DEPARTAMENT DE LLENGUATGES I SISTEMES INFORMÀTICS

UNIVERSITAT JAUME I

Representaciones de vocabularios en tareas de traducción automática mediante modelos conexionistas.

TESIS DOCTORAL

PRESENTADA POR:

GUSTAVO A. CASAÑ NUÑEZ

DIRECTOR:

DR. JOSÉ MARTÍNEZ SOTOCA

Castellón, Abril de 2011

A mi madre.

Índice.

Índice de Figuras

Capítulo 1.

Introducción

Resumen: En este capítulo se expone la motivación subyacente al trabajo realizado, así como los objetivos de la tesis y su organización. Además se describen las herramientas utilizadas a lo largo del proceso experimental.

1.1. Motivación

Actualmente las Redes Neuronales Artificiales (*Artificial Neural Networks*, ANNs), también llamadas Modelos Conexionistas o simplemente Redes Neuronales, constituyen un área de gran interés desde la perspectiva de la ingeniería. Aunque su existencia data de mediados del siglo pasado, el descubrimiento reciente de hallazgos importantes[1] ha provocado un auge espectacular en las aproximaciones neuronales en los últimos veinticinco años.

El principal atractivo de las ANNs radica, por un lado, en su capacidad de aprendizaje a partir de ejemplos del mundo real y, por otro lado, en su habilidad para resolver problemas en los que aparecen dependencias temporales, llevándonos esto último a las Redes Neuronales Recurrentes (*Recurrent Neural Networks*, RNNs).

Estas características convierten a las arquitecturas conexionistas en herramientas interesantes para abordar tareas relacionadas con el procesamiento del lenguaje. Entre estas aplicaciones podemos distinguir el reconocimiento de voz, el tratamiento del lenguaje natural, la comprensión del lenguaje o la traducción automática entre lenguajes.

El reto definitivo en el campo de la Traducción Automática (*Automatic Translation*, AT), es el diseño de sistemas que permitan una verdadera comunicación multilingua persona-a-persona en la que el lenguaje no sea un obstáculo. No obstante, en la actualidad aún no se puede considerar que los problemas generales de Traducción Automática estén satisfactoriamente resueltos. Sólo si se restringen los objetivos de estos sistemas a tareas de dominio restringido[2] se pueden conseguir sistemas apropiados.

---

[1] Como nuevas arquitecturas, métodos de entrenamiento y su posible integración en circuitos.

[2] Con léxicos de talla pequeña, estructuras sintácticas sencillas y un universo del discurso acotado.

Desde mediados de los años 90 han comenzado a realizarse experiencias en Traducción Automática con ANNs. Cabe notar los realizados en [Castaño, 1997], donde se aborda el problema de la traducción con una aproximación conexionista sencilla, denominada RECONTRA (del inglés *REcurrent CONnectionist TRAnslator*). En esta aproximación se obvian procesos de análisis sintácticos y semánticos de las frases a traducir y traducidas, permitiendo abordar de manera directa y sin un lenguaje intermedio, la traducción entre el lenguaje fuente y el destino.

Los resultados obtenidos con RECONTRA en [Castaño, 1997] al abordar una tarea sencilla de dominio restringido y utilizar codificaciones locales resultaron muy prometedores, pero la metodología empleada tiene serias limitaciones. Al emplear codificaciones locales para representar los vocabularios, la red crece enormemente en tareas más cercanas a la realidad y por tanto más grandes. Se demostró empíricamente cómo la utilización de determinados tipos de codificaciones distribuidas creadas manualmente, permite abordar problemas con vocabularios más amplios.

1.2. Objetivos

El objetivo fundamental de la tesis es explorar la posibilidad de utilizar RECONTRA con tareas de traducción reales, más complejas, y no limitadas a dominios concretos. Esto implica resolver el problema de las codificaciones e intentar mejorar el propio traductor. A continuación se muestran los siguientes objetivos parciales:

1. Desarrollar un método basado en ANNs para crear codificaciones distribuidas automáticamente adecuadas a tareas de traducción.
2. Abordar nuevas y más complejas tareas de traducción[3]. Además, abordar nuevos pares de lenguas, para ampliar la fiabilidad del estudio.
3. Mejorar los resultados con el modelo RECONTRA y su funcionalidad, adaptándolo a cada tarea mediante la utilización de distintos parámetros y métodos de entrenamiento.
4. Explorar la utilidad y resultados de variantes del modelo RECONTRA con diferente número de capas ocultas y palabras de salida.
5. Explorar también la posibilidad de combinar distintas redes RECONTRA en un único sistema de traducción.

1.3. Organización de la tesis

La tesis está dividida en tres bloques principales que agrupan varios capítulos y unos apéndices. El primer bloque explora aspectos teóricos, el segundo la metodología de la experimentación y las tareas abordadas, y finalmente los resultados experimentales y las conclusiones.

El primer bloque está formado por los capítulos 1 al 5 donde se presentan los aspectos teóricos dentro de los cuales se enmarca la tesis. Así, se habla en general del problema de la traducción

[3] Con vocabularios más amplios y no limitadas en el ámbito del discurso.

automática y los diversos sistemas desarrollados para abordarla, para luego hablar de los sistemas conexionistas en general y en concreto de RECONTRA. Finalmente, se presenta el problema de cómo representar el conocimiento necesario para la tarea, a través de la codificación de los vocabularios y varias posibles soluciones.

En el segundo bloque formado por los capítulos 6 y 7, se introducen los aspectos experimentales generales: las tareas de traducción que se han abordado, con que métodos se han entrenado las ANNs y que características tienen estas, así como los criterios utilizados para tomar estas decisiones.

El tercer bloque es el más extenso y comprende los capítulos 8 al 12. En esta parte se presenta la experimentación realizada, dividida en distintos capítulos según el factor o factores en los que se concentraba el estudio: el tipo de codificaciones empleadas, los métodos de obtención de codificaciones automáticas, los métodos de entrenamiento de los modelos, las variantes sobre el modelo RECONTRA básico, o la combinación de redes.

Finalmente, un capítulo de conclusiones finales de la tesis y los apéndices, que aportan información complementaria.

1.4 Herramientas utilizadas

Para la realización del trabajo de la tesis se han utilizado diversas herramientas preexistentes, además de código propio. Con el modelo RECONTRA y los Perceptrones Multicapa (*Multi Layer Perceptrons*, MLPs), la experimentación se desarrolló mediante un simulador de redes neuronales de uso público denominado *Stuttgart Neural Network Simulator* (SNNS) [Zell, 1995]. Este simulador ha sido desarrollado por el Instituto para Sistemas Paralelos y Distribuidos de Alto Rendimiento (*Institut für Parallele und Verteilte Höchstleistungsrechner*) de la Universidad de Stuttgart, en Alemania, desde 1989.

El SNNS implementa los tipos de ANNs y métodos de entrenamiento más comunes, y al ser de libre distribución, permite modificarlos y añadir nuevos métodos con una relativa facilidad, dado que está completamente implementado en ANSI-C.

Este simulador puede ser utilizado de tres formas: de forma gráfica e interactiva con el usuario; como librerías llamadas desde programas en C/C++ y también mediante scripts y en modo batch. Estas dos últimas facilitan la ejecución de trabajos que requieren mucho tiempo de cálculo.

También se emplea el código utilizado en [Miikkulainen, 1989] llamado FGREP, que amablemente los autores han colocado a disposición del público.

En el aspecto hardware mencionar que se han utilizado diversas máquinas de cálculo y clusters de distintos tamaños y características. Por ejemplo, Anubis, un servidor *RS/6000 S70 Advanced* y Nuvol, un servidor *HP 9000 SMP Clase-K* o Medusa, un *cluster* con una media de 8 computadoras personales bajo Linux.

Capítulo 2.

La Traducción Automática

Resumen: En este capítulo se presenta el campo de la Traducción Automática, su posible definición, problemas e historia, para posteriormente comentar las dos estrategias en Traducción Automática: la deductiva y la inductiva, y las principales tendencias que se agrupan bajo ellas.

2.1. Traducción Automática entre lenguajes naturales

La Traducción Automática (*Automatic Translation*, AT) pretende obtener, a partir de un mensaje en un idioma fuente, un mensaje en otro idioma (destino) que conserve todo el significado e intencionalidad del primero. Es evidente que la AT puede provocar un profundo cambio en la sociedad, haciendo que la comunicación entre personas sin un lenguaje común ya no sea una dificultad casi insalvable.

El estudio moderno de la AT surgió oficialmente en 1949 con el memorandum que Warren Weaver envió a un grupo de colegas de distintas disciplinas [Nirenburg, 2003], en el cual expresaba la necesidad y las posibilidades de la entonces llamada *Machine Translation*.

Durante los primeros años se desarrolló con mucho optimismo y a partir de la formalización matemática de lo que se consideraba un lenguaje. Esta formalización permitía relacionar directamente los lenguajes naturales y los ordenadores (ver "El Lenguaje y el Entendimiento" [Chomsky, 1968]).

Originalmente se creía que el problema de la traducción automática estaría resuelto en unos pocos años, pero muy pronto se vio que era mucho más complejo de lo inicialmente supuesto, y hoy en día incluso se duda de si es posible resolverlo completamente. Como ya dijo Chomsky "*The existence of deep-seated formal universals… does not, for example, imply that there must be some reasonable procedure for translation between languages*" [Chomsky, 1968].

La complejidad del problema de la AT es tremenda dado que el lenguaje natural se caracteriza por su flexibilidad, diversificación y variación, y sólo lo limita la mutua comprensión entre los hablantes. Los lenguajes naturales cambian con el tiempo, hasta el

punto de transformarse tanto que se les considera un lenguaje distinto del original. Los lenguajes suelen organizarse en términos de familias. Así, se hablan de lenguas 'hermanas' o 'madres' de otras (por ejemplo el Latín como lengua madre del Castellano).

Por el contrario, el lenguaje artificial es un lenguaje en el cual las reglas están establecidas explícitamente antes de su uso, y cuya función generalmente es suplementar el lenguaje natural dada su mayor especificidad de referencia, mayor economía de expresión y mayor adecuación para su comunidad de usuarios. En el campo de la AT se suele hablar de un lenguaje pseudonatural cuando tenemos una versión 'simplificada' de uno natural, generada automáticamente mediante gramáticas.

La última tendencia son los llamados lenguajes controlados derivados de los lenguajes naturales, en los cuales los "usuarios" humanos siguen una serie de normas de simplificación (del vocabulario y la gramática) con el deseo de mejorar la comprensión y eliminar posibles problemas de ambigüedad.

En el resto del capítulo intentaremos definir qué es la AT, así como mostrar a grandes pinceladas su historia y algunos de los problemas que se deben afrontar cuando intentamos traducir entre distintos idiomas. Además, explicaremos las principales técnicas utilizadas en AT y el estado del arte en el campo. Dado que existen dos estrategias generales para construir un sistema de AT, la basada en reglas o deductiva y la basada en el corpus o inductiva, agruparemos las distintas técnicas en estos puntos.

2.1.1. ¿Qué es la Traducción Automática?

Existen muchas definiciones distintas de qué es la AT. Sin embargo, es importante aclarar que no existe una definición general que sea aceptada por todos los investigadores o usuarios, y es sencillo entender el porqué.

Por ejemplo, según el diccionario de la Real Academia Española de la Lengua, traducir es: 'Traducir. (Del lat. *traducere*, hacer pasar de un lugar a otro). tr. Expresar en una lengua lo que está escrito o se ha expresado antes en otra. || 2. Convertir, mudar, trocar. || 3. Explicar, interpretar. MORF. conjug. c. *conducir*.'

Así, siguiendo la primera acepción, podríamos hacer la extensión natural de que la Traducción Automática consiste en '*Expresar en una lengua lo que está escrito o se ha expresado antes en otra*' de forma automática, es decir, que una máquina realiza esta acción. Las acepciones segunda y tercera sólo añaden confusión, dado que nos hacen darnos cuenta de que expresar en una lengua lo que está expresado en otra requiere (o puede requerir) un alto grado de interpretación y conversión, lo cual inmediatamente incluye cierto grado de subjetividad.

Un problema con las diversas definiciones de la AT es que se puede considerar que la AT es cualquier tipo de traducción hecha por una máquina. Por ejemplo, en [Arnold, 1994] se dice que la AT es *"...the attempt to automate all, or part of the process of translating from one human language to another*". Sin embargo, usualmente se habla de traducción cuando se trabaja a mayor nivel que una palabra, es decir, que una aplicación

que implemente un simple diccionario bilingüe no se considera actualmente una aplicación de AT.

También se discute a partir de qué grado de automatización del proceso de traducción se considera que ésta es automática. Aunque en la definición del párrafo anterior no se menciona explícitamente, está claro que como mínimo el ser humano debe indicarle a la máquina qué se desea traducir. Dependiendo de la intervención humana en la traducción se suelen diferenciar tres niveles de automatización:

- MAHT (del inglés *Machine Assisted Human Translation*): la traducción es realizada por un ser humano que utiliza la computadora como un asistente. El caso más sencillo podría ser un revisor ortográfico para el lenguaje destino o un diccionario automático.
- HAMT (del inglés *Human Assisted Machine Translation*): la traducción la hace una máquina pero con intervención humana. Cuando la máquina se encuentra un problema, como por ejemplo una ambigüedad léxica o estructural, se detiene el proceso de traducción y se pide ayuda al usuario humano.
- FA(HQ)MT (del inglés *Fully Automatic (High Quality) Machine Translation*): la traducción es realizada completamente por una máquina sin limitaciones en el lenguaje. El aspecto *High Quality* se da por la exigencia de que la traducción sea correcta en casi todos los casos, comparable a los errores que pueda cometer un experto humano.

El otro aspecto fundamental al considerar un sistema de AT es la amplitud de los problemas de traducción que se abordan. Podemos considerar dos tipos:

- **General**: cualquier nivel de comunicación humana en cualquier situación.
- **Restringida a dominios**: se delimita un dominio, sea en el tamaño del vocabulario, la estructura de las frases o el campo al cual se aplica. Por ejemplo, las interacciones en la recepción de un hotel. Aquí es donde suelen aparecer las tareas en lenguaje pseudonatural. Algunas tareas comprenden un dominio tan amplio (como Hansards o EUROPAR[4], las actas de las intervenciones en los parlamentos canadiense y europeo, respectivamente) que se acepta que la diferencia con una tarea general es mínima.

Observando cualquiera de las herramientas que están accesibles para un usuario se puede creer que el objetivo último de la AT no está tan lejano, pero esto no es así. La mayor parte de los sistemas actuales son de los dos primeros tipos (MAHT y HAMT), y funcionan mejor contra más limitado sea su dominio. Esto es aún más evidente cuando

[4]En http://www.parl.gc.ca y http://europa.eu/institutions/inst/parliament/index_es.htm respectivamente.

hablamos de FA(HQ)MT, dado que sólo en dominios limitados[5] se logran buenos resultados.

2.1.2. Visión histórica

Históricamente, la preocupación por el lenguaje y la comunicación se pueden remontar cientos de años. Los primeros intentos, anteriores al siglo XX, consistieron en la creación de diccionarios mecánicos que traducían de un lenguaje a otro.

Esos intentos no tuvieron mucha repercusión, pero en los años treinta del siglo XX se realizaron varias patentes que señalan el interés por la AT e intentan crear algo más que un diccionario que no tenga forma de libro. Por ejemplo, Petr Smirnov-Troyanski en Rusia plantea y patenta un traductor que utiliza analizadores léxicos y un lenguaje intermedio.

En 1949 Warren Weaver y Andrew D. Booth plantearon la utilización de ordenadores para traducción basándose aún en la idea de que existía una base universal para el lenguaje, y de que ésta podía utilizarse como una lengua intermedia en el proceso de traducción. Entre otras plantearon tres ideas que se han mantenido hasta nuestros días:

- Si conocemos las suficientes palabras que preceden y siguen a una palabra en la frase, podemos obtener su traducción correcta: a esto se le ha llamado ventana o contexto de una palabra.
- El texto origen y el texto destino dicen lo mismo: es posible en dos idiomas distintos expresar la misma información, lo único que cambia entre uno y otro es la codificación de la información.
- Existe o puede crearse un lenguaje que permite la representación de cualquier información que pueda comunicarse entre seres humanos. Este 'lenguaje universal' o interlingua puede utilizarse para traducir: del lenguaje A se pasa a la interlingua y de allí al lenguaje B sin pérdida de información.

En 1951 Yoshua Bar-Hillel fue el primer investigador a tiempo completo en AT y en 1952 fue uno de los impulsores de la primera conferencia en el campo. Ya en 1954 se realizó la primera demostración de un sistema de AT, desarrollado en colaboración por la Universidad de Georgetown e IBM. El sistema constaba de 250 palabras de vocabulario y 6 reglas gramaticales. En este momento se pensaba que era sólo una cuestión de escala que el problema fuese resuelto.

A pesar de los problemas en desarrollar sistemas de AT prácticos durante estos primeros años, los desarrollos teóricos fueron muy importantes: tal vez el más importante fue Noam Chomsky cuando en 1957 revolucionó el estudio de las lenguas con la publicación de su *Syntactic Structures*.

---

[5] O cuando los lenguajes implicados son muy similares, como en el caso del traductor castellano/catalán InterNostrum (disponible en http://www.internostrum.com).

En 1960 Bar-Hillel publicó un artículo recomendando objetivos menos ambiciosos que FA(HQ)MT de texto sin límites. Según él, este objetivo estaría fuera de nuestro alcance, dado que la máquina necesitaría conocimiento del mundo real equivalente al de un ser humano, por lo que recomendaba objetivos más realizables.

Se puede argumentar incluso que es un problema irresoluble hasta que seamos capaces de crear inteligencias artificiales similares a las propias de los seres humanos, dada la necesidad de comprensión del mensaje y de su contexto social, cultural e incluso individual para poder realizar la traducción.

Citando a Douglas Hofstadter *"... Un programa ... pronto se verá irreversiblemente atascado en ambigüedades y significados múltiples. Incluso las personas –dotadas de una inmensa ventaja frente a las computadoras, ya que están totalmente equipadas con una comprensión del mundo- encuentran casi imposible traducir texto a su propio idioma, cuando emplean para conseguirlo solamente un diccionario del idioma que no conocen."* [Hofstadter, 1987].

No olvidemos que incluso un traductor humano que conozca perfectamente los lenguajes implicados puede cometer errores si desconoce el contexto del mensaje.

El teórico más importante de la idea de que existe una base universal para el lenguaje es el ya mencionado Noam Chomsky, con sus ideas de una Gramática Universal común a todos los seres humanos [Chomsky, 1968]. Aunque sus ideas tienen muchos críticos especialmente en los detalles (ver *El Instinto del Lenguaje* [Pinker, 1995]), todos los intentos de justificar la traducción de una lengua a otra pasando por algún tipo de representación intermedia se apoyan en sus ideas.

A pesar de la importancia sostenida de Chomsky, se considera que en 1966 Estados Unidos dejó de estar a la cabeza de la investigación en AT, a raíz de la publicación de un informe del *Automatic Language Processing Advisory Comitee* que aconsejaba abandonar la AT por poco eficaz. Siguiendo esta directriz, la mayor parte de los fondos públicos desaparecieron y Estados Unidos pronto dejó de ser la potencia líder en este campo.

La disciplina avanzó considerablemente con el incremento en potencia de cálculo, abaratamiento de los equipos informáticos, y los nuevos diseños de estructuras de datos y lenguajes de programación de alto nivel.

Los principales centros de investigación de la década de los 70 fueron los europeos, especialmente desde que en 1976 la Comisión de la Comunidad Europea instala '*Systran*', un sistema de AT Inglés-Francés. Muy pronto se desarrollan más sistemas para otros pares de lenguajes y se crea el proyecto '*Eurotra*'.

La Union Europea necesitaba limitar los problemas presentados por la existencia de docenas de lenguas distintas dentro de su territorio, con lo que apoyó multitud de iniciativas para crear sistemas de AT.

En 1976, investigadores del grupo ATUM (del francés *Traduction Automatique de l'Université de Montréal*) presentaron el sistema MÉTÉO, que traducía partes meteorológicos del inglés al francés. Es un sistema que ha hecho historia por la idoneidad

de la aplicación y diseño, y el porqué diversas versiones de él estuvieron en funcionamiento desde 1981 hasta 2001 para la traducción de partes meteorológicos para el gobierno canadiense.

Los años 80 se caracterizan por un fuerte desarrollo de los métodos simbólicos y una gran vitalidad de la investigación en sintaxis (gramáticas basadas en la unificación de rasgos) y en semántica (formalismos basados en la lógica de predicados). No obstante, los avances en el plano teórico no acababan de trasladarse al terreno de los resultados prácticos.

Hasta este momento los sistemas de AT se habían basado en la transferencia entre un lenguaje y otro, dependiendo del conocimiento lingüístico de expertos humanos. En los años ochenta en la universidad Carnegie Mellon se desarrollaron nuevas aproximaciones en la AT basadas en los avances en sistemas de comprensión del lenguaje natural.

A finales de los ochenta y principios de los noventa, diferentes grupos de investigación en Estados Unidos y Japón empezaron a desarrollar sistemas de AT puramente estadísticos y sistemas basados en corpus de ejemplos de traducción.

Por el lado institucional, la CE decidió cancelar definitivamente la financiación de EUROTRA; por el lado empresarial, la empresa PHILIPS inesperadamente da por terminado uno de los proyectos de más prestigio entre los especialistas, ROSETA. Paralelamente en Japón, se aplica una política de moderación presupuestaria tras las fabulosas inversiones de los años precedentes.

En este contexto de declive generalizado, hace su aparición en el mercado un nuevo tipo de producto de traducción asistida de diseño muy distinto a los anteriores. Son los programas de gestión de memorias de traducción dados a conocer primeramente por la empresa IBM (“TranslationManager”), y posteriormente llevados al gran público por las empresas alemanas TRADOS (“Translator's Workbench”) y SATR (TRANSIT), y la española ATRIL (DÉJÀ-VU).

El objetivo de muchos de estos sistemas ya no es la AT “pura”, sino proporcionar al traductor una herramienta que simplifique su trabajo. El éxito de estos sistemas, su uso generalizado y la satisfacción de los usuarios, es indiscutible. Otro aspecto destacable de esta etapa es el desarrollo de Internet. De repente, casi cualquier información está disponible en la red, pero el usuario debe ser capaz de comprenderla. La necesidad de un sistema de AT resulta evidente.

En la actualidad estamos en una nueva etapa. La globalización de empresas y mercados lleva asociada la necesidad de adaptar localmente productos y servicios (por ejemplo los manuales de los electrodomésticos caseros). Un buen número de tecnologías desarrolladas a lo largo de los últimos 50 años se están integrando y complementando en arquitecturas híbridas que conciben la traducción como un eslabón más en el complejo ciclo de la información en un medio que ahora es completamente electrónico.

Un ejemplo podría ser EDITerm que se concibe como una herramienta genérica con distintas utilidades para distintos usuarios:

"*EDITerm's primary purpose is to ensure that three types of users have access to consistent terminology: 1. terminologists… 2. unilingual writers… 3. translators…*" (de http://www.editerm.com/profile.asp 11/1/2008).

Otro ejemplo importante son los dos proyectos METIS [Carl, 2008], que entre 2001 a 2007, y patrocinados por la Unión Europea, han intentado desarrollar sistemas de traducción de texto libre sin el uso de ningún tipo de texto bilingüe.

2.1.3. Problemas lingüísticos en traducción

El problema fundamental en la traducción es que los significados y estructuras de significado de un lenguaje no tienen porque corresponderse con los del otro. Desde un punto de vista lingüístico se podría decir que un lenguaje está lleno de "agujeros" en relación con el otro lenguaje. Si no se quiere perder mensaje en la traducción, un traductor humano experto logra cubrir estos "agujeros", por ejemplo expresando mediante una frase o párrafo lo que en el idioma origen era una única palabra.

Además de este problema, existen una serie de problemas que son ineludibles en la traducción, sea esta automática o no, y que añaden mayor complejidad. Los cuatro principales son:

- **Morfología**: en la mayor parte de los lenguajes, las palabras se forman a partir de una base a la cual se añaden letras que concretan su significado en cada uso e indican características como el género, el número o el tiempo verbal[6]. Sin embargo, en todos los lenguajes no se indican las mismas características o funciones[7].
- **Ambigüedad léxica**: ésta se presenta de varias formas; la primera sería en la categoría, cuando una palabra puede tener varias funciones en una frase[8]; la segunda sería la ambigüedad holográfica, cuando dos o más palabras completamente distintas en significado se escriben igual; y por último la ambigüedad en la traducción, cuando una palabra tiene varias palabras equivalentes en otro idioma.
- **La ambigüedad estructural**: se produce cuando una frase puede tener diversas estructuras internas que cambian su sentido. Por ejemplo "*Flying planes can be dangerous"* tiene dos sentidos, pilotar un avión puede ser peligroso o los aviones volando pueden ser peligrosos. O en castellano "*Juan mencionó la carta que envié a Susana",* otra vez aparece con dos sentidos: Juan le dijo a Susana que envié una carta o por el contrario Juan dijo (no sabemos a quién), que le envié una carta a Susana. Son frases que tienen más de una interpretación según como las interprete el lector.

---

[6] La conjugación de los verbos en castellano sería un ejemplo, como lo es la declinación latina o alemana.

[7] Por ejemplo, mientras que el castellano, el valenciano y la mayor parte de las lenguas latinas diferencian el género de una palabra, en inglés no es así, lo cual además provoca que cierta información se pierda o deba incorporarse en el proceso de traducción.

[8] Por ejemplo en inglés, donde casi cualquier palabra puede utilizarse como nombre y como verbo.

- **Reordenación**: el orden de las palabras en distintas lenguas no tiene porque ser el mismo, con lo que el traductor debe además buscar una palabra en un diccionario, reordenar la frase en el idioma destino. Por ejemplo, en castellano y en inglés los nombres y los adjetivos que los califican no tienen el mismo orden: "*la casa verde*" vs. "*the green house*".

Todos estos problemas hacen que el problema de la AT genérica esté lejos de ser resuelto. No obstante, muchas tareas de interés para la industria y la ciencia son de dominio restringido y tienen una solución más fácil. Así, el léxico requerido es de tamaño más reducido y el universo del discurso es limitado, con lo que el número de posibles ambigüedades es menor, cuando no nulo.

Por ejemplo, en los manuales de uso de una aplicación informática se busca deliberadamente que no puedan existir varias interpretaciones de las frases, con lo que se facilita sin pretenderlo la tarea de traducción[9]. Esta acotación del problema ha permitido el desarrollo de sistemas automáticos de traducción de gran utilidad dentro de las limitaciones impuestas, como el ya mencionado MÉTÉO.

2.2. La traducción automática basada en el conocimiento humano: estrategia deductiva[10]

La aproximación más tradicional en el campo de la AT es la deductiva. En ella, el conocimiento lingüístico de expertos humanos es transferido al sistema en forma de reglas adecuadas al problema, por lo que los modelos desarrollados bajo esta aproximación se denominan genéricamente "Sistemas de Traducción Basados en Reglas" [Arnold, 1994]. Aunque esta aproximación apunta hacia una traducción de propósito general, éste es todavía un objetivo inalcanzado, proporcionando resultados aceptables únicamente en sistemas de AT de ayuda al experto humano o en dominios restringidos.

Esto se debe a que para dominios más amplios se acumulan un gran número de reglas, por lo que aparece el problema de la búsqueda y decisión de cual es la regla más apropiada para cada caso. Los lenguajes naturales parecen tener demasiadas reglas y excepciones para ser fácilmente abordables con este método.

Si bien tradicionalmente todos los sistemas basados en reglas se han construido a partir del conocimiento proporcionado por traductores y lingüistas humanos, también se han desarrollado sistemas en los cuales estas reglas o los diccionarios bilingües, han sido creados a partir de ejemplos de la tarea de traducción. Por ejemplo, se pueden desarrollar

[9] Si verdaderamente logran esta claridad de exposición es discutible, aún en los casos en los que se emplean lenguajes controlados, como ASD-STE100 Simplified Technical English.

[10] La fuente principal de este punto ha sido *Introduction to Machine Translation* [Hutchins, 1992].

diccionarios bilingües a partir de los resultados de alineamiento de los denominados modelos IBM [Brown, 1993].

La filosofía subyacente a los sistemas basados en reglas ha sido considerar la traducción como un problema, principalmente, de equivalencia semántica. Esta premisa se asienta en el supuesto de que todas las lenguas del mundo comparten una misma subestructura lógica. De esta creencia se extiende que, si fuéramos capaces de descubrir y formalizar esta subestructura, el problema de la traducción estaría resuelto.

Las principales divisiones entre los distintos métodos se basan en como estos métodos intentan resolver el problema de la equivalencia conceptual. Así, se habla de métodos "directos" cuando se trabaja con los dos idiomas implicados directamente, y de métodos "indirectos" cuando se trata de resolver el problema de la equivalencia conceptual de una forma indirecta, desarrollando algún tipo de representación o representaciones intermedias. En la Figura 2.1, podemos ver la pirámide clásica de organización de métodos de traducción basados en reglas.

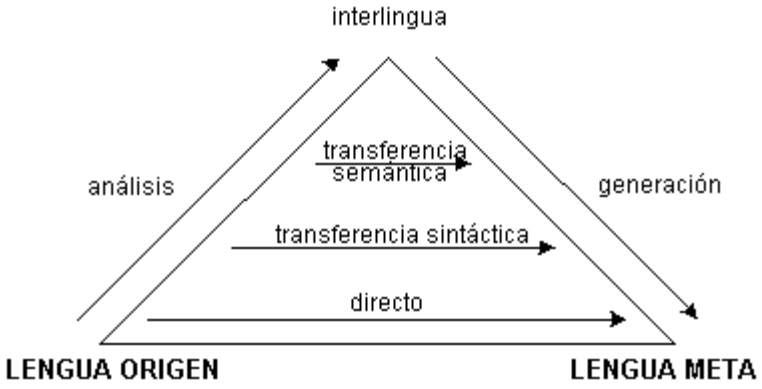

Figura 2.1. Organización de los métodos de traducción basados en reglas, adaptado de [Hutchins, 1992].

Los enfoques indirectos a la Traducción Automática también son llamados de "conocimiento lingüístico". La idea común a todos los enfoques indirectos es que, para conseguir AT de alta calidad, es indispensable tener información tanto de la lengua de origen como de la lengua destino (meta). También es común la necesidad de una representación intermedia que capture el "significado" de la oración del texto de origen, para generar una oración en el texto destino que sea equivalente en significado.

A continuación, comentaremos algunas de las técnicas más comunes dentro de estas dos estrategias.

2.2.1. Técnica directa

En su forma más pura, la traducción directa conlleva la traducción palabra por palabra junto con un proceso de equivalencias de cadenas y reordenación del texto destino. La traducción se realiza en un único paso: se consultan las palabras de la lengua de origen en el diccionario y directamente se generan sus correspondencias en la lengua destino,

aplicando sólo algunas reglas ad hoc de flexión[11], concordancia y reordenamiento, solucionando los principales problemas a la hora de traducir.

Los problemas inherentes a tal metodología son evidentes: el sistema no toma en consideración la estructura sintáctica de la frase ni las relaciones semánticas que existen entre las palabras. Además, no existe ninguna forma de asegurar la correcta formación de las expresiones del lenguaje destino, ya que no existen reglas gramaticales explicitas del lenguaje.

De tal modo, una frase como "*Juan la tocó*" podría ser traducida al inglés por un sistema de traducción directa como "*John the touched*" debido a la homonimia que exhibe la palabra española "*la*" (pronombre – artículo). Además, la palabra castellana "*tocó*" puede referirse no sólo a tocar físicamente un objeto, sino también a tocar un instrumento musical o una canción, con lo que posibles traducciones correctas serían "*John played it*", "*John touched it*" y "*John touched her*". A esto habría que añadir que aquí también se ha traducido el nombre propio "*Juan*" por "*John*", lo cual puede considerarse incorrecto.

La idea principal idea de un sistema de traducción directa es transformar las frases de la lengua origen en frases de la lengua destino de la manera más simple posible, reemplazando palabras de la lengua origen con sus correspondientes de la lengua destino, siguiendo un determinado diccionario bilingüe, y aplicando las reglas de transformación de las palabras o frases que contiene dicho diccionario.

Históricamente, una característica fundamental de este tipo de arquitecturas es la utilización de gramáticas muy simples, de tal modo que en la mayoría de los casos el sistema no es capaz de decidir si una determinada frase de la lengua destino es correcta gramaticalmente o no, limitándose a traducir los distintos componentes por él hallados.

Por supuesto, el léxico de uno de estos sistemas está acorde, en cuanto a calidad de información léxica con el tipo de gramática. Se trata típicamente de un diccionario con equivalencias de traducción de uno a uno, en donde no cabe la ambigüedad semántica. Las distintas entradas carecen de cualquier tipo de información aparte de la morfológica, ya que la sintaxis se implementa directamente en los algoritmos de *parking* reordenando las palabras traducidas con formulas preestablecidas.

A pesar de todo lo anterior, es sencillo entender que con ciertas modificaciones este tipo de sistemas es capaz de obtener resultados bastante buenos. Por ejemplo, incorporando algún tipo de sistema que modele la lengua destino y permita seleccionar entre distintas traducciones posibles combinado con un diccionario que ofrezca varias posibilidades de equivalencia de una palabra, o ampliar el diccionario para que incorpore expresiones formadas por varias palabras.

[11] En lingüística se denomina flexión a las modificaciones que sufre un lexema para expresar su función en la estructura gramatical.

2.2.2. Técnica de interlingua

El método de interlingua plantea la traducción a través de una única representación intermedia, común a las lenguas entre las que se va a traducir, lo que permite que la traducción se realice en sólo dos fases: "análisis" del texto a traducir y "generación" del texto traducido.

En el "análisis" se le asigna una estructura al texto de origen, usando para ello únicamente información de la lengua origen. Esta estructura objetivo es una oración en un lenguaje universal (interlingua) que representa el "significado" del texto. En teoría, partiendo de esta estructura, se puede generar el texto correspondiente en cualquier otra lengua sin importar cual fue la lengua de origen. Esta separación entre el conocimiento de la lengua de origen y la de destino es la principal característica y motivación de la AT mediante interlingua.

Suelen recomendar este método quienes mantienen que para traducir un texto antes hay que comprenderlo. La información semántica suele estar recogida en una base de conocimientos mediante un modelo del mundo accesible durante el proceso de traducción.

Una ventaja de estos métodos es que en un sistema multilingüe, la traducción en cualquier dirección para *n* lenguas mediante interlingua requiere únicamente *2n* módulos (análisis y síntesis para cada lengua), dado que la interlingüa sería común para todos.

No obstante, uno de los problemas más importantes que presenta este enfoque, es la elección del vocabulario de la interlingua [Arnold, 1994], es decir, los conceptos primitivos de la representación del significado.

Por ejemplo, se puede elegir un concepto "corner" para hacer referencia a la palabra inglesa *corner*; esto puede parecer natural a un hablante de la lengua inglesa, pero no a un hablante de lengua española, que contempla dos vocablos distintos según la orientación: "esquina" y "rincón". Para reflejar el castellano deberíamos contemplar dos conceptos primitivos distintos: *inside-corner* y *outside-corner*.

Se dan casos todavía más curiosos. Por ejemplo, en castellano se distingue entre "gato" y "gata", el macho y la hembra de la especie, pero en inglés, existen la palabra genérica *cat* y luego *tom* para el macho, y la menos habitual de *queen* para la hembra. ¿Qué conceptos primitivos deberíamos considerar? Al fin y al cabo el castellano tiene dos palabras y el inglés tres, con diferentes matices.

Otro ejemplo podría ser el ruso y los colores. En ruso no existe ninguna traducción para "azul", sino que existe *goluboi* (azul claro) y *sinii* (azul oscuro), ¿cuál sería la representación en interlingua de azul? ¿Sería necesario definir un rango de ondas luminosas que se corresponda con este color?

Por otra parte, si todos los lenguajes implicados son lenguajes humanos se debe poder expresar lo mismo con cualquiera de ellos, aunque algunos de ellos den facilidades para según qué temas. ¿Necesitaríamos un lenguaje universal en el cuál todo fuese sencillo de expresar? Parece que sí. Estos problemas no existen en los sistemas de transferencia que

veremos en la siguiente sección, puesto que las discrepancias entre lenguas se resuelven en el componente de transferencia del par de lenguas en cuestión.

2.2.3. Técnica de transferencia

Un sistema de AT basado en el modelo de transferencia incorpora, por una parte, una primera fase de análisis del texto origen, al cual le sigue una segunda fase que utiliza conocimiento bilingüe. El sistema de representación que transporta la información sobre el texto entre los dos análisis son las llamadas estructuras normalizadas, intermedias, o de interfaz.

En un sistema de transferencia típico, la cadena del texto de origen se analiza primero morfológicamente, y el resultado se analiza sintácticamente dando como resultado una representación sintáctica superficial. Esta representación se transforma después en una representación más abstracta que ignora algunos fenómenos no relevantes para el proceso de traducción, aportando a su vez una representación más apropiada de otros tipos de información.

A continuación, un módulo de transferencia convierte esta representación (que está aún ligada a la lengua de origen) a una representación al mismo nivel de abstracción pero ligada al lenguaje destino. A partir de esta representación el proceso se invierte: los módulos de síntesis generan una representación sintáctica del texto meta y finalmente la cadena de texto en la lengua destino.

El aspecto distintivo del método de transferencia sobre el método de interlingua es que propone dos representaciones intermedias, una por cada lengua del par de traducción. Sobre estas representaciones se organizan el análisis del texto de origen y la generación del texto de destino, de modo que la traducción se realiza en tres fases: análisis, transferencia y generación.

Por tanto, la principal característica de los sistemas de transferencia que los diferencia de los sistemas de interlingua, es la existencia de un módulo adicional de transferencia que proyecta representaciones intermedias del texto de origen sobre representaciones intermedias del texto de destino. Este módulo de transferencia puede trabajar en distintos niveles de análisis lingüístico, por lo que se pueden distinguir tres tipos de transferencia [Hutchins, 1992]:

Transferencia sintáctica: En ella se genera automáticamente un "árbol sintáctico"[12] de la frase a traducir y las operaciones que se realizan sobre las cadenas de caracteres son transformaciones de árbol a árbol que "transfieren" las estructuras sintácticas de la lengua origen, a las estructuras sintácticas de la lengua destino. En la figura 2.2 se puede ver un ejemplo de árbol sintáctico para una frase en castellano.

[12] Un *árbol* es un grafo simple unidireccional sin ciclos, en un árbol sintáctico las hojas de dicho árbol son palabras y las aristas muestran su posición dentro de la frase.

Dos de los sistemas más conocidos basados en transferencia sintáctica son METAL y ARIANE. En el primero, las transformaciones árbol-a-árbol se usan indexándolas a reglas gramaticales individuales, que son aplicadas después del análisis; en el segundo se construye una estructura interfaz de la lengua origen que después se transfiere a la estructura interfaz de la lengua destino, y se somete a una reestructuración posterior.

Este tipo de transferencia utiliza representaciones intermedias en forma de árboles sintácticos, sin prestar atención a las relaciones funcionales subyacentes. Hoy en día está generalmente admitido que la inclusión de tales relaciones es indispensable para un correcto análisis del texto origen. Sin embargo, algunos de los primeros sistemas daban por sentado que este grado de profundidad era suficiente para establecer satisfactoriamente transformaciones estructurales entre un par de lenguas.

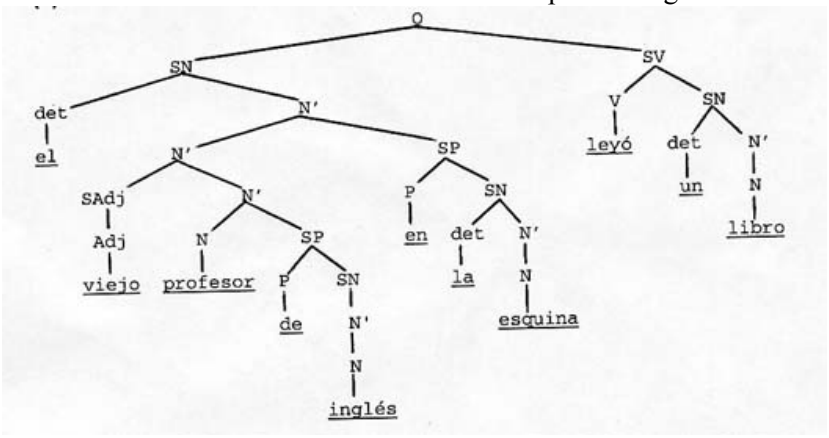

Figura 2.2. Árbol sintáctico para la frase en castellano: *el viejo profesor de inglés en la esquina leyó un libro*.

Transferencia semántica: En ella se transforman representaciones profundas, como patrones de casos, redes semánticas, o estructuras lógicas, que son dependientes del lenguaje origen. Esta representación puede consistir en una serie de estructuras argumentales o algún otro formalismo de representación del significado. En estos sistemas, la transferencia se realiza principalmente mediante la traducción de predicados.

Transferencia mixta: En ella las relaciones de transferencia se construyen con información sintáctica, funcional, semántica y algunas veces incluso pragmática. Puesto que el sistema utiliza múltiples niveles de información, el sistema puede codificar las equivalencias de traducción en el nivel más apropiado para las lenguas en cuestión.

Sea cual sea el tipo de transferencia que el sistema lleva a cabo, hay una característica común a todos los sistemas basados en transferencia y es el elevado número de módulos que deberán ser implementados según aumenta el número de lenguas. En concreto, para un sistema que tradujese entre `n` lenguas deberían ser implementados al menos `n*(n-1)` módulos de transferencia, es decir, cada lengua requerirá sus propios componentes de análisis y generación, y además, existirá un módulo de transferencia (dos en el casos de sistemas bidireccionales) para cada par de lenguas.

2.3. La traducción automática basada en corpus: estrategia inductiva

En las aproximaciones inductivas para la AT, el sistema aprende automáticamente el mecanismo o modelo de traducción a partir de ejemplos de las traducciones a realizar: "Sistemas de Traducción Basados en Ejemplos". Así, se puede prescindir de la intervención directa del traductor humano para la creación de los sistemas de traducción. Este modelo debe, de alguna forma, explicar los ejemplos e idealmente generalizar a partir de ellos, permitiendo la traducción de frases no presentes en los ejemplos.

Esta aproximación conlleva dos fases. En la primera fase se produce un proceso de aprendizaje, donde el modelo/sistema es generado a partir de los ejemplos. Generalmente, en esta fase también se incluye una parte de generalización del modelo. En la segunda fase se produce un proceso de reconocimiento, donde a partir de una frase de entrada se utiliza el modelo obtenido para obtener la traducción más adecuada.

El campo de la traducción basada en corpus se divide en tres tendencias principales: la traducción estadística, que se basa principalmente en la frecuencia de aparición de palabras y combinaciones de palabras, la traducción por ejemplos, que se basa en la extracción y combinación de sentencias u otros fragmentos reducidos de texto y la basada en sintaxis que crea modelos sintácticos de los lenguajes implicados a partir de los ejemplos.

Como es evidente dentro de estas tendencias podemos tener sistemas de traducción directa, indirecta o incluso de transferencia, pero hay otro tipo de AT basada en corpus, la AT mediante modelos conexionistas**,** que comentaremos con detalle en el siguiente capitulo, dado que la mejora de uno de estos modelos es el objetivo de esta tesis.

2.3.1. La traducción basada en ejemplos

La traducción basada en ejemplos (*Example Based Machine Translation*) se basa en la idea de realizar traducciones imitando los ejemplos de traducción de frases similares presentes en un corpus. Estos ejemplos se almacenan en una base de datos que facilite la posterior localización a partir de cualquiera de las palabras contenidas en los ejemplos. El elemento diferenciador con otras técnicas es que no se crea un modelo de traducción explícito.

En [Turcato, 2001] se define un sistema de traducción basado en ejemplos como aquel que cumple las siguientes condiciones:

- Utiliza un corpus bilingüe como base principal de conocimiento.
- Utiliza este corpus bilingüe en tiempo de ejecución, no sólo en la fase inicial de construcción de un modelo, como hacen otros métodos.

Según [Cranias, 1994] se pueden distinguir tres aspectos claves que permiten trabajar a los sistemas de traducción basados en ejemplos:

Correspondencia: establecer las correspondencias entre las unidades a partir de un corpus bi o multilingüe, sea a nivel de palabra, frase o segmento intermedio, según la unidad de texto empleada.

Unidad de texto: dada una entrada, establecer un mecanismo de recuperación de las unidades que coincidan mejor con esta entrada dentro de la base de datos. Es fundamental establecer la "unidad de texto" que determina si la búsqueda se realizará a nivel de sentencia o subsentencia. También será necesaria, la definición de una métrica de similitud entre dos unidades de texto, y por tanto, la reducción del espacio de búsqueda.

Búsqueda y utilización: explotar los ejemplos de traducción obtenidos de la base de datos para producir la traducción de la frase original.

El problema de establecer correspondencia entre corpus paralelos ha sido ampliamente estudiado por muchos autores, y existen diversas técnicas y metodologías que proporcionan buenos resultados [Brown, 1993]. En este trabajo los autores tratan el problema de alinear las palabras de dos frases bilingües mediante métodos estadísticos.

Ya se ha comentado cómo la selección de la unidad de texto determina si la búsqueda se realizará a nivel de frase o subfrase. Una elección obvia sería utilizar como unidad de texto la frase, dado que podríamos obtener la frontera entre unidades de texto de forma no ambigua. Sin embargo, las sentencias pueden ser a veces demasiado largas para realizar búsquedas en un tiempo razonable.

Además, el utilizar sentencias ocasionaría que se perdiese flexibilidad, dado que encontrar una frase completa exactamente igual a la del conjunto de evaluación en la base de datos es menos probable. En el otro extremo, si seleccionamos una única palabra como unidad de texto, estaríamos trabajando con un diccionario y la tarea de combinar las palabras y producir la frase resultado se complicaría.

Si por el contrario se selecciona la subfrase como unidad de texto, aparecen varios problemas. Para empezar será necesario un proceso que alinee textos paralelos a este nivel. También resultará muy importante utilizar algún método que estime la longitud óptima de las subsentencias. Por otra parte, hay que tener en cuenta que pueden aparecer problemas de ambigüedad cuando la longitud de los segmentos es demasiado pequeña. Si la decisión de las fronteras de división no es la adecuada, cuando posteriormente intentemos combinarlas, ocasionará una traducción incorrecta.

A pesar de que muchos de los sistemas que están funcionando en la actualidad utilizan como unidad de texto la sentencia, que es el caso más sencillo, el potencial de los sistemas basados en ejemplos está en explotar fragmentos de texto más pequeños que la sentencia, combinándolos para producir sentencias completas. El alineamiento automático de subsentencias es un problema todavía no completamente resuelto, y la combinación de las subsentencias producidas para generar una única traducción tampoco es evidente.

Una de las técnicas que está demostrando más utilidad para combinar subsentencias es la utilización de árboles de palabras o sintácticos. Aunque existen diversos tipos de

técnicas, la estrategia general se concentra en el análisis sintáctico de los ejemplos para crear árboles. Al representar los ejemplos de esta forma se pueden utilizar multitud de técnicas desarrolladas para el trabajo con modelos de árboles. De especial interés son las técnicas de búsqueda dentro del bosque formado por las posibles traducciones de una frase de entrada, como el algoritmo de búsqueda A*.

Otra aproximación estrechamente relacionada con la traducción basada en ejemplos son las memorias de traducción [Sato, 1990] [McTait, 2001]. Mediante esta técnica normalmente no se tratar de construir un sistema totalmente automático de traducción, sino que se orienta hacia la construcción de sistemas de apoyo al experto humano.

La idea consiste en buscar en el corpus bilingüe ejemplos de cómo se han traducido frases similares. Una vez visualizada esta información, el traductor humano puede utilizarla como referencia para realizar su traducción. Es habitual que el traductor humano pueda incorporar las nuevas traducciones al corpus bilingüe, con lo que éste se adapta a las preferencias del usuario y a las necesidades de la tarea.

En definitiva un programa de traducción basado en ejemplos o en memorias funciona de la siguiente forma:

1. El usuario introduce un texto del que se pretende obtener una traducción.
2. El sistema fragmenta esta entrada en “unidades de texto”.
3. El sistema busca en la base de datos los ejemplos almacenados que coinciden en mayor o menor grado con estas unidades.
4. Se decide de alguna forma cuál es la mejor traducción.

   En el caso de un sistema de memorias de traducción, toda la información localizada que se considera de interés será mostrada para que un traductor humano tome las decisiones oportunas y construya las frases de salida más adecuadas.

   En el caso de un sistema basado en ejemplos, un mecanismo automático combina los fragmentos encontrados y produce una sentencia de salida.

2.3.2. La traducción estadística basada en el modelo del canal

Uno de los primeros planteamientos inductivos ensayados para resolver el problema de la AT fueron las técnicas estadísticas. La aplicación de modelos estadísticos a la AT es el intento de capturar regularidades del lenguaje natural utilizando distribuciones probabilísticas de eventos lingüísticos, como la aparición de palabras dentro de un contexto.

Muy pronto se percibieron las limitaciones: resulta muy difícil capturar relaciones a larga distancia en una frase cuando, por ejemplo, trabajamos con n-gramas, aún utilizando cuatro o cinco palabras para modelar éstos.

A pesar de estos problemas, en los últimos años los modelos estadísticos de AT han experimentado un resurgimiento muy importante por tres factores principales:

- Existe una gran cantidad de textos masivos de fácil acceso, y por tanto, de corpora de entrenamiento.

- En el campo del reconocimiento del habla, los métodos estadísticos proporcionan muy buenos resultados, resultando muy interesante combinar en un único sistema el reconocedor de voz y el traductor.
- Al disponer de máquinas más rápidas y con mayor capacidad de cálculo se pueden utilizar modelos estadísticos más complejos.

La investigación actual utilizando estos modelos se basan en los llamados "modelos IBM" [Brown, 1990]. La filosofía de estos trabajos se basa en el modelo de canal ruidoso de Shannon. En la Figura 2.3 podemos observar gráficamente el funcionamiento de este modelo. Dado una secuencia `e` de texto correcto que atraviesa un canal de comunicación ruidoso, obtenemos a la salida una secuencia `f` de texto distorsionado. Podemos obtener entonces, una función que estime la probabilidad de que se produzca dicha entrada a partir de dicha salida, `p(e|f)`, utilizando la ecuación (2.1).

$$p(e \mid f) = \frac{p(e) \cdot p(f \mid e)}{p(f)} \qquad (2.1)$$

Entonces sería posible reconstruir la entrada más probable, `e`, probando todas las posibles entradas de `e`, y seleccionando la de mayor valor.

Figura 2.3. Modelo del canal ruidoso de Shannon: una entrada *e* es trasmitida y modificada hasta convertirse en una salida *f*, distinta de la original.

Y para obtener la traducción que máximice la probabilidad de la ecuación (2.2) tenemos:

$$p(e) \cdot p(f \mid e) \qquad (2.2)$$

Los denominados modelos IBM se desarrollaron en el centro de investigación IBM T.J.Watson [Brown, 1990]. Estos cinco modelos son utilizados hoy en día como referencia en el resto de investigaciones en este campo. La idea principal de estos modelos es conseguir un alineamiento entre la frase a traducir y la frase traducida. Siguiendo el modelo del canal ruidoso, se aproxima este problema de forma inversa, utilizando un modelo de lenguaje para el lenguaje destino (generalmente mediante n-gramas) y un modelo de alineamiento de las palabras de la frase de salida con las palabras de la frase de entrada.

Intuitivamente, un alineamiento nos indica qué palabras de la frase de origen han generado cada una de las palabras de la frase de destino. Se pueden definir muchos tipos de alineamientos, además del evidente de una palabra a otra. Por ejemplo, se puede permitir que una palabra genere la palabra vacía o que se puedan generar varias palabras a

partir de una sola. Incluso que varias palabras de origen generen (juntas) una o varias palabras destino o que haya palabras que se generen "espontáneamente".

En los modelos IBM se propone utilizar concretamente la siguiente definición de alineamiento, donde, siendo `a` una posible frase destino, a_j es la palabra destino que está alineada con la palabra en la posición de origen `j`:

$$a=a_1 \ldots a_{|j|} \tag{2.3}$$

Este concepto nos ayuda en la definición de la probabilidad condicional `p(f|e)`, que puede ser formalizado matemáticamente de varias formas [Brown, 1993; Vogel, 1996; Och, 1999; Ney, 2001].

Una vez definido un alineamiento podemos formular el modelo de traducción de la siguiente forma:

$$p(e \mid f) = \sum_a p(f, a \mid e) \tag{2.4}$$

Sin pérdida de generalidad, podemos obtener la probabilidad de una frase de salida `f`, dada una palabra de entrada `a` que forma parte de la frase de entrada `e`, `p(f,a|e)` como:

$$p(f, a \mid e) = p(|f| \mid e) \prod_{j=1}^{|f|} (p(a_j \mid a_1^{j-1}, f_a^{j-1}, |f|, e)\, p(f_j \mid a_1^{j}, f_a^{j-1}, |f|, e)) \tag{2.5}$$

donde `|f|` es la longitud de la frase `f`. Con esta descomposición, obtenemos tres distribuciones de probabilidad diferentes: la probabilidad de la longitud de la frase, la probabilidad de alineamiento y la probabilidad léxica. Como no pueden ser calculadas de forma directa, se simplifican asumiendo que ciertos parámetros son dependientes. Cada conjunto de asunciones define uno de los cinco modelos IBM. Sin entrar en detalles, veamos sus características:

IBM1: Asume que la distribución de alineamiento es uniforme y que la probabilidad léxica de f_j depende exclusivamente de la palabra con la cual es alineada.

IBM2: Añade una nueva distribución, de distorsión, que trata de estimar que una palabra origen en posición `j` sea alineada con otra palabra destino en la posición `i`.

IBM3: Introduce el parámetro de fertilidad, que modela la probabilidad de que una palabra destino pueda estar alienada con varias palabras origen.

IBM4: La distribución de distorsión es sustituida de forma que la posición de una palabra destino alineada con una de origen depende de la posición de la palabra destino alienada a la palabra origen que precede a la palabra origen actual. Se intenta que las palabras que aparecen juntas en la frase de origen aparezcan juntas en la frase destino. En muchos casos, para paliar el problema de la escasez de datos de entrenamiento se utilizan clases de palabras en lugar de palabras individuales.

IBM5: Los modelos 3 y 4 tienen el problema de que la suma de probabilidades para todos los posibles alineamientos válidos no sume uno. El modelo 5 es el modelo 4 con este problema solucionado.

Los modelos IBM se han convertido en la base de la AT basada en métodos estadísticos a pesar de sus deficiencias, y gran parte de la investigación de los últimos años se ha dedicado a intentar paliarlas.

2.3.3. Traducción basada en sintaxis

Dentro de la traducción basada en sintaxis se pueden agrupar multitud de técnicas, pero todas tienen un punto en común: intentan de una forma o de otra, tener en cuenta la sintaxis de los lenguajes implicados en la tarea de traducción. Esto pueden hacerlo de muchas formas: teniendo en cuenta el contexto de aparición de las palabras, utilizando árboles sintácticos, autómatas de estados finitos o simplemente incorporando algún tipo de modelo de lenguaje.

Muchos de estos métodos intentan incorporar algún tipo de conocimiento sintáctico a los modelos de Brown, fundamentalmente utilizando métodos de alineamiento de varias palabras a la vez y no de una sola, pero también se ha explorado la posibilidad de incorporar modelos del lenguaje para mejorar los resultados de la traducción.

2.3.3.1. Modelos estadísticos basados en máxima entropía

Como alternativa a la aproximación basada en un canal ruidoso, algunos autores proponen utilizar una aproximación basada en el concepto de máxima entropía [Och, 2002]. En el marco de la máxima entropía, la probabilidad de que una traducción sea correcta es expresada a partir de M funciones características h_m, $m = 1,\ldots,M$.

Estas funciones características tratan de recoger aquellas propiedades que pueden ser de utilidad a la hora de definir el modelo estadístico, lo cual nos ofrece una gran versatilidad a la hora de incorporar información al modelo básico. Para cada función característica se dispone de un parámetro λ_m. Esta aproximación obtiene la siguiente regla de decisión:

$$\sum_{m=1}^{M} \lambda_m \, h_m(e,f) \tag{2.6}$$

En [Och, 2002] se propone utilizar esta aproximación utilizando dos funciones características: un modelo de traducción directo, junto con el modelo de lenguaje (la razón por la cual este método se ha incluido dentro de los basados en sintaxis):

$$h_1(e,f) = \log p(e) \tag{2.7}$$

$$h_2(e,f) = \log p(e \mid f) \tag{2.8}$$

Obteniendo la siguiente regla de decisión:

$$p(e)^{\lambda_1} p(e \mid f)^{\lambda_2} \tag{2.9}$$

La ecuación (2.9) es similar a la obtenida en la aproximación por canal ruidoso (2.2), aunque utilizando un modelo directo de traducción. Experimentalmente se ha comprobado [Och, 2002] que la aplicación de esta versión proporciona mejores resultados de traducción. La optimización de estos parámetros λ_1 y λ_2 sería el equivalente

a la optimización de los factores de escalado, usados en las áreas de reconocimiento de voz y reconocimiento de patrones.

Esta aproximación presenta la ventaja de que los valores λ_m pueden obtenerse por medio del algoritmo GIS (*Generalized Iterative Scaling*) [Darroch, 1972] y sus múltiples variantes [Goodman, 2002].

Esta aproximación también se ha utilizado para obtener modelos léxicos refinados, en los que la probabilidad de traducción de una palabra se obtiene, no solamente conociendo la palabra con la que se alinea, sino que además se tiene en cuenta la posibilidad de que aparezca una determinada palabra en el contexto de la palabra destino.

Distintas funciones características son definidas según esta palabra se encuentre a la izquierda, a la derecha o en una posición cercana a la palabra destino.

2.3.3.1. Modelos de alineamiento

La principal deficiencia de los modelos de alineamiento basados en palabras es que ignoran el contexto donde aparece la palabra a traducir. Para solucionar esta limitación, en los últimos años, se han creado diversos modelos que intentan ampliar las capacidades de los modelos IBM. Fundamentalmente, estos modelos intentan alinear secuencias de palabras (*phrase-based models*) en lugar de palabras individuales.

Casi todos los modelos de alineamiento basados en secuencias de palabras se derivan del modelo de alineamiento de plantillas (*alignment template*) [Och, 1999] [Och, 2002] y [Och, 2004].

- Los modelos de alineamiento **basados en plantillas** trabajan con agrupaciones de palabras, llamadas plantillas o estructuras. Una plantilla es una tripleta `z=(F,E,A)` que describe el alineamiento `A`, entre una secuencia de origen de clases de palabras `F` y una secuencia de destino de clases de palabras `E`. Dos ejemplos de plantillas podrían ser `S1`: it, he, she,... y `T1`: comienza, llega,… La principal diferencia con otros modelos de alineamiento es que se trabaja con clases de palabras.
- Los modelos de alineamiento **basados en secuencias de palabras** lo que proponen es sustituir los parámetros léxicos del modelo, que se basan en palabras individuales, por parámetros léxicos que tengan en cuenta secuencias de palabras. Existen muchos intentos de crear este tipo de modelos [Marcu, 2002], [Tillmann, 2003], [Koehn, 2003], [Voguel, 2003], [Tomás, 2001] y [Tomás, 2004].

En [Zens, 2008] se hace un estudio comparativo de distintos modelos. En general, en una primera fase la frase de entrada es segmentada en secuencias de palabras, en la segunda fase se escoge la traducción de cada secuencia, y finalmente, se concatenan las secuencias traducidas. Como es evidente estas fases no son sencillas de completar, y tienen relación con las necesidades de la AT basada en memorias de traducción.

Existen otras estrategias para alinear los lenguajes, como los modelos de alineamiento **basados en gramáticas**. Estos modelos se caracterizan por realizar un análisis sintáctico sobre la frase de entrada como paso previo a la traducción. Este tipo de análisis da lugar a

árboles sintácticos, y lo que se intenta es alinear estos árboles sintácticos con aquellos que producirán la frase de salida más probable.

En esta aproximación se asumen asociaciones probabilísticas entre los símbolos no terminales o las reglas de los modelos de los lenguajes de entrada y salida (gramáticas estocásticas), y se intentan aprender estas asociaciones automáticamente, a partir de pares de frases de entrada-salida de la tarea considerada. Ejemplos de esta aproximación son [Yamada, 2001], [Prat, 2001a] y [Prat, 2001b].

Sin embargo, el modelo presenta varias limitaciones. La principal de ellas es su nula capacidad de generalización. Si se intenta traducir una secuencia de palabras que no ha sido vista en el entrenamiento, el modelo no dispone de los mecanismos necesarios para obtener una salida en función de otras secuencias similares.

No obstante, este modelo ha demostrado que puede ser eficaz en dominios restringidos o entre pares de lenguas que no requieran una gran reordenación. Aunque la gran simplicidad de estos modelos facilita su entrenamiento y permite el uso de algoritmos de búsqueda muy amplios, se requiere un número de parámetros muy elevado.

Estos no son los únicos métodos de alineamiento estadístico que se han desarrollado, dado que es aún un problema sin resolver de forma completamente satisfactoria. En [Och, 2003] se hace una buena revisión de las tendencias en el campo.

2.3.3.2. Modelos basados en autómatas de estados finitos

Un autómata de estados finitos es un tipo de modelo matemático de un sistema que recibe una cadena constituida por símbolos de un alfabeto, y determina si esa cadena pertenece al lenguaje que el autómata reconoce. Existen varios tipos de automatas, según las propiedades que cumplen. Un ejemplo de uno de ellos, el transductor de estados finitos, puede verse en la Figura 2.4.

Desde sus inicios, en el campo de la traducción se ha trabajado con diversos tipos de máquinas de estados finitos, como las máquinas de Mealy y de Moore, pero creadas manualmente. Actualmente, las técnicas de traducción más utilizadas son los traductores basados en modelos de estados finitos deterministas [Castellanos, 1998].

Los autómatas de estados finitos presentan un amplio abanico de posibilidades. Por un lado, la comunidad interesada en el procesado de lenguaje natural ha utilizado a menudo los modelos de estados finitos en muchas aplicaciones, principalmente en dominios restringidos.

Por otro lado, en reconocimiento del habla el uso de modelos de n-gramas, que no es más que un tipo muy simple de modelo estocástico de estados finitos, es la técnica más utilizada en dominios restringidos.

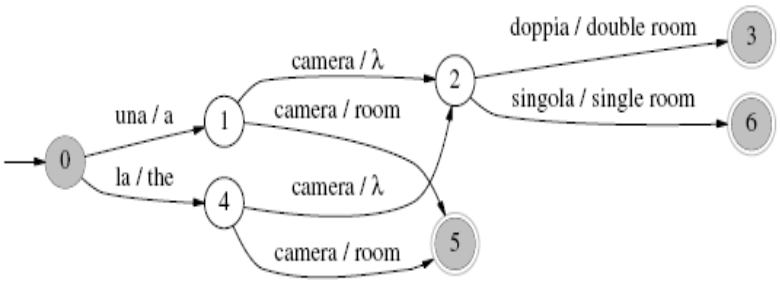

Figura 2.4. Un transductor subsecuencial de estados finitos generado con GIATI para Italiano/Inglés (tomado de [Casacuberta, 2004], página 213).

Tradicionalmente la información contenida en estos formalismos es introducida de forma deductiva, aunque desde los años 90 se han desarrollado multitud de técnicas inductivas para crearlos automáticamente y pueden ser formuladas bajo un marco estadístico [Vidal, 1997], [Knight, 1998], siendo esta la razón por la que han sido incluidos dentro de este apartado.

Uno de los tipos de máquinas de estados finitos más utilizados son los transductores de estados finitos, y dentro de esta clase tenemos los transductores subsecuenciales. A su vez dentro esta clase tenemos los transductores secuenciales. Un transductor secuencial es aquel que preserva la longitud de los prefijos de las cadenas de entrada-salida a medida que son analizadas. Este tipo de transductores sólo es aplicable a tareas muy sencillas como la traducción palabra-a-palabra.

Por su parte, los transductores subsecuenciales [Oncina, 1991], [Castellanos, 1994], [Castellanos, 1998], [Vilar, 1999] y [Amengual, 2001] son aquellos en los que los símbolos de su salida son generados sólo cuando se han visto suficientes símbolos en la entrada para garantizar una salida correcta. En la figura 2.4 se puede observar un ejemplo sencillo.

Los transductores subsecuenciales pueden ser aprendidos a partir de una adecuada representación positiva de ejemplos de entrada-salida, utilizando un algoritmo muy eficiente denominado OSTIA (*Onward Subsequential Transducer Inference Algorithm*) [Oncina, 1991].

La idea básica de esta aproximación consiste en construir inicialmente un árbol aceptor de prefijos compatible con el conjunto de pares de entrenamiento entrada-salida. A continuación, se fusionan pares de estados con el objetivo de asignar las cadenas de salida a los arcos más cercanos posibles al estado inicial, y obtener así un único transductor subsecuencial hacia adelante que sea mínimo.

Varios trabajos han utilizado este algoritmo para el aprendizaje de transductores en dominios restringidos [Jiménez et al, 1995], [Vidal, 1997] y [Castellanos et al, 1998]. El algoritmo OSTIA ha sido mejorado para permitir la utilización de información adicional

que puede ser extraída del propio corpus de aprendizaje. Este nuevo algoritmo se ha denominado Omega [Vilar, 1998].

Una de las principales ventajas de los sistemas de traducción basados en transductores de estados finitos es la posibilidad de inferirlos a partir de pares de muestras de entrenamiento [Vidal, 1989] y [Oncina, 1993]. Sin embargo, se dispone de un gran número de técnicas que permiten inferir gramáticas regulares a partir de un conjunto de cadenas de entrenamiento de un lenguaje [Vidal, 1995].

En [Casacuberta, 2000] se propone el método GIATI para inferir transductores de estados finitos basandose en la relación formal entre los transductores de estados finitos y las gramáticas regulares. La idea consiste en inferir una gramática regular a partir del corpus bilingüe de entrenamiento alineado, y transformarla posteriormente en un transductor de estados finitos.

Uno de los aspectos de mayor interés de las máquinas de estados finitos es su fácil integración en sistemas de reconocimiento del habla y la eficiencia de sus motores de traducción. Esto facilita el desarrollo de sistemas de traducción del habla. En [Casacuberta, 2004] se puede obtener una visión con mayor profundidad del campo.

2.3.3.3. Modelos basados en árboles sintácticos

Otra alternativa de AT son los modelos basados en árboles sintácticos. El interés por estos modelos se origina por el deseo de realizar un análisis más profundo de la estructura sintáctica de la frase.

Existen muchas variantes de árboles sintácticos que han sido utilizadas para traducción, dado que originalmente los árboles se utilizaban dentro de la estrategia deductiva. Por ejemplo, los árboles sintácticos que se utilizaban en muchos sistemas de traducción basados en transferencia sintáctica.

La idea fundamental de esta alternativa es que cada frase de ejemplo se convierte inicialmente en un árbol. En una etapa posterior se combinan los árboles, de forma que los árboles finales tendrán varias ramas o alternativas. Así, se obtendrán árboles modelo de uno de los lenguajes que se aprovecharán para realizar la tarea de traducción. Existen dos alternativas fundamentales:

- La traducción se produce mediante algunas reglas de conversión de un árbol en el lenguaje origen a otro árbol en el lenguaje destino [Yamada, 2001] [Yamada, 2002], [Galley, 2004], [Galley, 2006] y [Cowan, 2006], entre muchos otros.
- Los árboles son bilingües: en el momento de crearlos se utiliza algún tipo de alineamiento para hacer correspondencia de palabras entre el lenguaje origen y el de destino [Wu, 1996], [Wu, 1997], [Melamed, 2004], [Chiang, 2005], [Zollman, 2006] y [Chiang, 2007].

Capítulo 3.

Redes Neuronales Artificiales (ANNs)

Resumen: Este capítulo se realiza una breve descripción de la historia de las ANNs, para posteriormente profundizar en su conexión con las redes biológicas, su implementación práctica y mostrar distintas variedades. Además, se muestra su utilización actual en el campo del tratamiento del lenguaje natural.[13]

3.1. Visión histórica

En 1936, Alan Turing fue el primero en estudiar la arquitectura neuronal para aplicarla al mundo de la computación. No obstante, quienes primero formalizaron los primeros fundamentos de la computación neuronal fueron Warren McCulloch y Walter Pitts. En 1943, en su trabajo *A logical calculus of the ideas immanent in nervous activity*, intentaron explicar el funcionamiento del cerebro humano por medio de un modelo de células conectadas entre sí, de tal manera que una célula nerviosa o neurona no es más que un dispositivo con entradas y salidas.

El modelo consistía en la suma de las señales de la entrada a la red, multiplicadas por unos valores de pesos escogidos aleatoriamente. La entrada era comparada con un patrón preestablecido para determinar la salida de la red. Si la suma de las entradas multiplicadas por los pesos era mayor o igual al patrón preestablecido, la salida de la red era 1, en caso contrario, la salida era 0.

Seis años después, el fisiólogo Donald O. Hebb expuso que las redes neuronales podían aprender. Su propuesta tenía que ver con la conductividad de la sinapsis, es decir, con las conexiones entre neuronas. Hebb expuso que "cuando un axón de la célula A excita la célula B y participa en su activación, se produce algún proceso de desarrollo o

[13] Una buena fuente para obtener una visión histórica más amplia (y la principal del capítulo) es David Medler [Medler, 1998], especialmente por la cantidad de referencias que se pueden extraer de ella. La mayor parte de la información general acerca del funcionamiento de las ANNs está tomada de [Rojas, 1996].

cambio metabólico en una o en ambas células, de manera que la eficacia de A, como célula excitadora de B, se intensifica".

Bajo la dirección del científico Burrhus Frederic Skinner, Marvin Minsky diseñó varias máquinas que imitaban el comportamiento biológico. Los conocimientos adquiridos le sirvieron como base para crear en 1951 la primera máquina de redes neuronales.

En 1957 Frank Rosenblatt inició el desarrollo del Perceptrón, un identificador de patrones ópticos binarios con salida binaria. Las capacidades del Perceptrón se extendieron al desarrollar la "regla de aprendizaje delta", que permitía emplear señales continuas de entrada y salida.

En 1959 Bernard Widrow publicó una teoría sobre la adaptación neuronal y dos modelos inspirados en esa teoría: Adaline (*Adaptative Linear Neuron*) y Madaline (*Multiple Adaline*). Estos modelos fueron utilizados en numerosas aplicaciones y permitieron usar, por primera vez, una red neuronal en un problema importante del mundo real: filtros adaptativos para eliminar ecos en las líneas telefónicas.

En 1969 Marvin Minsky y Seymour Papert publican su libro *Perceptrons: An introduction to Computacional Geometry*. En dicho libro, se realiza una fuerte y detallada crítica del Perceptrón de Rosenblatt, resaltando la principal limitación del Perceptrón: su naturaleza lineal. Se presentó el famoso ejemplo de su incapacidad para representar la función XOR. Como consecuencia, se generaron serias dudas sobre las capacidades de los modelos conexionistas y esto provocó un abandono general del campo.

James Anderson fue uno de los pocos investigadores que continuó con el desarrollo de ingeniería neuronal. En 1977 presentó sus estudios realizados con modelos de memorias asociativas. Teuvo Kohonen continuó su trabajo al desarrollar modelos de aprendizaje competitivo basados en el principio de inhibición lateral. Su principal aportación consiste en un procedimiento para conseguir que unidades físicamente adyacentes aprendieran a representar patrones de entrada similares.

Por otro lado, Stephen Grossberg realizó un importante trabajo teórico basándose en principios fisiológicos, y aportando importantes innovaciones con su modelo ART (*Adaptative Resonance Theory*). Junto a Cohen elaboró un importante teorema sobre la estabilidad de las redes recurrentes en términos de una función de energía.

Durante los años 80 ocurrió el resurgimiento en el estudio de las ANNs. En 1982 el físico John Hopfield elaboró un modelo de red basado en unidades de proceso interconectadas [Hopfield, 1982], al cual aplicó los principios de estabilidad desarrollados por Grossberg.

En 1986 Rumelhart, McClelland y Hinton crearon el grupo PDP (*Parallel Distributed Processing*). Como resultado de los trabajos de este grupo, surgieron los manuales: *Parallel Distributed Processing. Explorations in the Microstructure of Cognition* y, *Explorations in Parallel Distributed Processing. A Handbook of Models. Programs and Exercices*. En estos manuales destacan los capítulos dedicados al algoritmo de retropropagación que soluciona los problemas planteados por Minsky y Papert, y

extienden enormemente el campo de aplicación de los modelos conexionistas, convirtiéndolos de esta forma en una de las áreas de la inteligencia artificial más ampliamente estudiadas en todo el mundo.

Desde entonces y hasta hoy en día la creación de nuevos modelos de ANNs, nuevos métodos de entrenamiento y la aplicación a nuevos campos han sido prácticamente constantes.

3.2. El modelo biológico

La teoría y modelado de redes neuronales está inspirada en la estructura y funcionamiento del sistema nervioso humano, donde la neurona es el elemento fundamental. Se estima que el sistema nervioso humano contiene en total más de cien mil millones de neuronas y sinapsis, con conexiones del orden de 10^{15}. La estructura de una neurona biológica se muestra en la Figura 3.1. En cualquier caso, hoy en día gran parte del trabajo con ANNs ignora sus raices biológicas y no busca inspiración en la naturaleza.

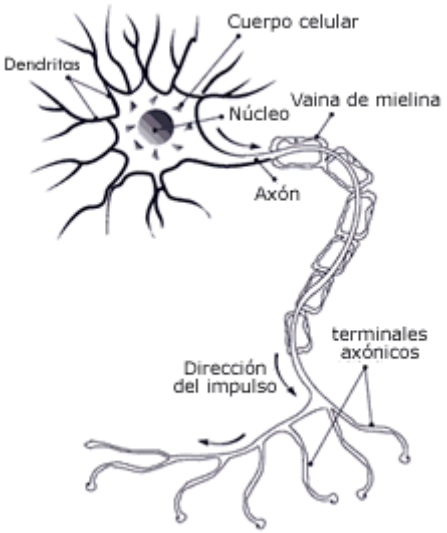

Figura 3.1. Dibujo de una neurona biológica genérica.

Una de las principales características de las neuronas es su capacidad de comunicarse. Su tamaño y forma es variable, pero con las mismas subdivisiones que muestran en la Figura 3.1. Esto es:

1. El **cuerpo** de la neurona o **soma** contiene el núcleo de la célula. El cuerpo de una neurona es más o menos esférico, de 5 a 10 micras de diámetro, y de él salen una rama principal, el axón y varias ramas más cortas llamadas dendritas. El cuerpo se encarga de todas las actividades metabólicas de la neurona y recibe la información de otras neuronas vecinas a través de las conexiones sinápticas, las combina e integra y emite señales de salida. Algunas neuronas se comunican sólo con las más cercanas, mientras que otras se conectan con miles, que pueden estar a centímetros de distancia.

2. Las **dendritas** parten del soma y se encargan de la recepción de señales de otras células a través de conexiones llamadas sinápticas. Si pensamos en términos computacionales podemos decir que las dendritas son las conexiones de entrada de la neurona.
3. El **axón** es la "salida" de la neurona y se utiliza para enviar impulsos o señales a los terminales axónicos de las dendritas de otras neuronas. Cuando el axón está cerca de sus células destino, se divide en muchas ramificaciones que forman sinapsis con el soma o axones de otras células.

   Estas señales son de dos tipos: eléctricas y químicas. La señal generada por la neurona y transportada a lo largo del axón es un impulso eléctrico, pero al llegar a los axones, para superar la separación entre distintas células, el terminal genera varias substancias químicas que son percibidas por las dendritas al superar ciertas concentraciones. El efecto puede ser "inhibidor" o "excitador" de la neurona receptora según el transmisor que las libere. Se estima que cada neurona recibe de 10.000 a 100.000 sinapsis y el axón realiza una cantidad de conexiones similares.

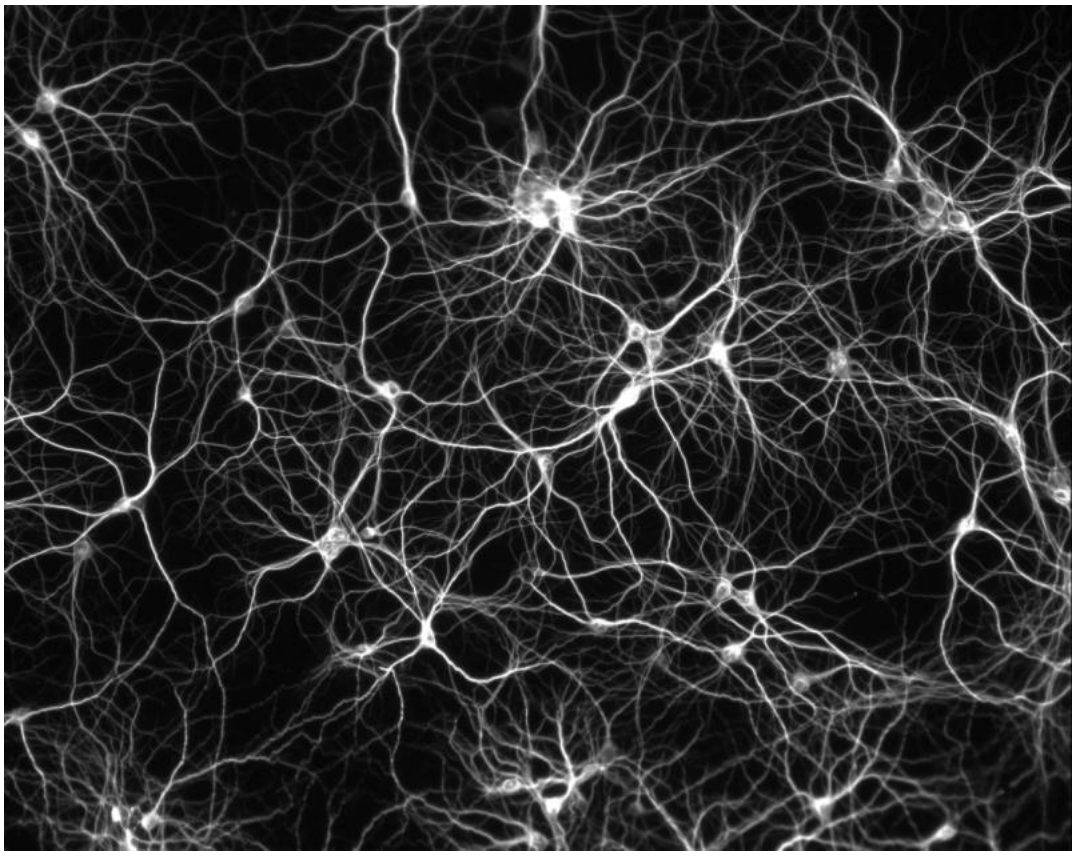

Figura 3.2. Fotografía de un corte del hipocampo de una rata (Paul de Konink Laboratory).

Esta explicación del comportamiento neuronal no refleja la auténtica complejidad de las neuronas presentes en el cerebro humano. Todavía hoy en día existen factores del funcionamiento de las redes neuronales biológicas mal comprendidos. Una muestra especialmente curiosa de las teorías sobre el comportamiento del cerebro humano puede

verse en el trabajo de Penrose [Penrose, 1990] [Penrose, 1995], así como la teoría de este sobre la conciencia y el funcionamiento a nivel cuántico de las neuronas.

3.3. La Red Neuronal Artificial

La definición de una Red Neuronal Artificial (*Artificial Neural Network*, ANN) podría ser la siguiente: las redes neuronales artificiales son sistemas de aprendizaje y procesamiento automático que, a través de modelos matemáticos recreados mediante mecanismos artificiales, pretenden imitar en pequeña escala la forma de funcionamiento de las neuronas que forman el cerebro humano. Ofrecen numerosas ventajas sobre los sistemas informáticos convencionales, entre las cuales podemos destacar:

- **Aprendizaje Adaptativo**: En el proceso de aprendizaje, los enlaces ponderados de las neuronas se ajustan de manera que se obtengan unos resultados específicos. Posteriormente, se mantienen fijos durante su aplicación, aunque también existen redes que continúan aprendiendo a lo largo de su uso, después de completado el periodo inicial de entrenamiento.
- **Autoorganización**: Esta autoorganización permite que las ANNs respondan apropiadamente cuando se les presentan datos o situaciones a las que no habían sido expuestas anteriormente. Por otra parte muchas ANNs tienen la capacidad de incrementar y disminuir tanto su número de neuronas como de conexiones, haciéndose más eficientes a la hora de tratar un determinado problema.
- **Respuesta ante nuevas entradas**: El modelo creado por las ANNs es continuo, con lo cual ante entradas desconocidas que la red no ha experimentado durante el entrenamiento, pueden ser capaces de generar una salida correcta o bastante próxima a la correcta.
- **Tolerancia a Fallos**: Debido a que tienen su información distribuida en las conexiones entre neuronas, en caso que se produzca un fallo en algunas de las neuronas, el comportamiento del sistema se ve afectado, pero no sufre una caída repentina. Esta tolerancia también la encontramos cuando hay problemas como ruido, distorsión de los datos o información incompleta en los datos de entrada.
- **Operación en Tiempo Real**: Los modelos neuronales pueden realizar ciertas operaciones en paralelo, y se pueden diseñar y fabricar máquinas con hardware especial para aprovechar esta capacidad.

3.3.1. Características de las ANNs

Existen cuatro aspectos que caracterizan una red neuronal: su topología, el mecanismo de aprendizaje, el tipo de asociación realizado entre la información de entrada y salida, y la forma de representación de esta información.

3.3.1.1. Topología de las ANNs

La topología o arquitectura de las redes neuronales consiste en la organización y disposición de las neuronas formando niveles o capas. Se conoce como capa o nivel a un conjunto de neuronas cuyas entradas provienen de la misma fuente, y cuyas salidas se dirigen al mismo destino (ver Figura 3.3).

Se denomina capa de entrada a la capa a la cual se presentan los datos y generalmente no tendrá entradas de otras capas. La capa de salida, por tanto, será aquella que produce los datos que se consideran la salida de la red.

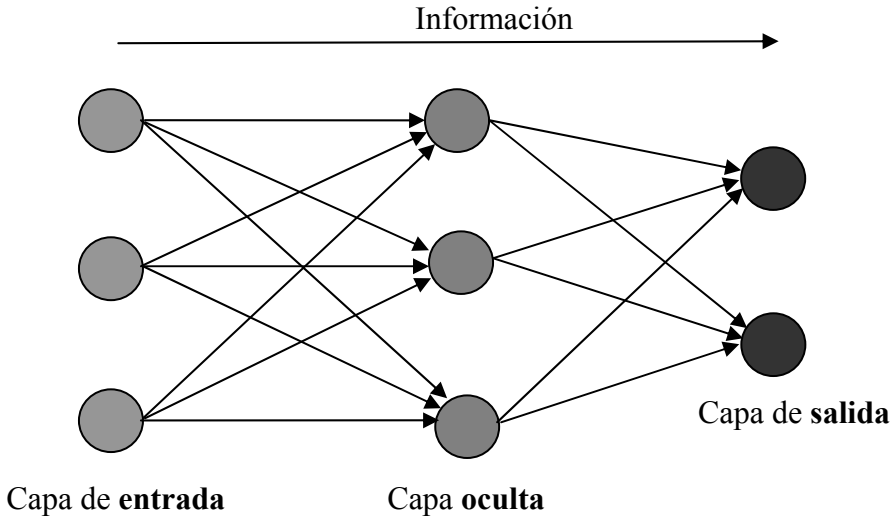

Figura 3.3. Topología de una red neuronal artificial sencilla, de izquierda a derecha se ve la capa de entrada, la capa oculta y la de salida (derecha).

Los parámetros fundamentales de la red desde el punto de vista de la arquitectura son: el número de capas, el número de neuronas por capa, el grado de conectividad y el tipo de conexiones entre neuronas. Considerando estos parámetros, podemos establecer diferentes taxonomías de las ANNs:

Atendiendo al número de capas las redes pueden clasificarse en:

Redes monocapa: Estas redes cuentan con una sola capa de neuronas que constituyen a un tiempo la entrada y salida del sistema. Se utilizan en tareas relacionadas con lo que se conoce como autoasociación. Un ejemplo de este tipo de redes es el perceptrón simple.

Redes multicapa: Disponen de conjuntos de neuronas agrupadas y jerarquizadas en distintos niveles o capas, con al menos una capa de entrada, otra de salida, y, una o varias capas intermedias también denominadas capas ocultas. Por otro lado, el número de capas ocultas está directamente relacionado con la capacidad de la red.

Una forma de distinguir la capa a la que pertenece una neurona consiste en fijarse en el origen de las señales que recibe a la entrada y en el destino de la señal de salida. Cuando ninguna salida de las neuronas es entrada de neuronas del mismo nivel o de niveles precedentes, la red se describe como de propagación hacia delante o "*feedforward"*, y cuando las salidas pueden ser conectadas como entradas de neuronas de

niveles previos o del mismo nivel, incluyéndose ellas mismas, la red es de propagación hacia atrás o "*feedback"*.

Redes modulares *(adaptive mixture of experts* o *neural network ensembles)*: Este enfoque originalmente basa su estructura en la modularidad que tiene el sistema nervioso humano, en el cual cada región cerebral tiene una función específica, pero a su vez, las regiones se interconectan entre sí. Por ello, se dice que una ANN es modular si la computación realizada por la red puede verse descompuesta en dos o más módulos o subsistemas que trabajan de forma independiente sobre los mismos datos o parte de ellos. Cada uno de estos módulos puede ser considerado como neuronas en la red en su conjunto. En su implementación más básica todos los módulos son de un mismo tipo, pero pueden utilizarse esquemas diferentes o combinarse con módulos que utilicen técnicas distintas de las ANNs.

Otra forma de ver las redes modulares es como un tipo de ensembles: un ensemble de ANNs es un conjunto de ANNs diferentes que juntas resuelven un problema [Zhi-Hua, 2002]. El elemento diferenciador de las redes modulares es que ellas utilizan otra ANN para combinar los resultados de cada módulo, y los ensembles pueden utilizar otros métodos.

La arquitectura más utilizada es la que cuenta con una capa de módulos de entrada, además de un módulo integrador. De manera que cada módulo aporta una solución o parte de ella a un mismo problema, y el módulo integrador determina la solución global a partir de las soluciones individuales de cada red.

Algunas de las ventajas que estructuras modulares de este tipo tienen sobre modelos multicapa, son las siguientes:

1. **Velocidad de aprendizaje**. Si una función compleja se descompone en un conjunto de funciones más simples, una red modular puede implementar dicha descomposición y obtener un aprendizaje más rápido, en comparación a cuando se utiliza una única red para aprender la función sin descomponer.
2. **Tratamiento de la información**. Las redes modulares son bastante útiles cuando se trabaja con fuentes de información diferentes, o cuando los datos han sido preprocesados con diferentes técnicas.
3. **Distribución del conocimiento**. En una red modular, los módulos de la red tienden a especializarse mediante el aprendizaje de diferentes regiones del espacio de entrada.
4. **Fácil integración con otras técnicas**. Dado que en gran parte cada módulo es independiente de los demás, se pueden utilizar tipos muy distintos de ANNs e incluso otras técnicas completamente distintas.

Sin embargo, las redes modulares también presentan diferentes dificultades. La principal dificultad es cómo dividir el problema en partes para cada módulo. Si la división no es la adecuada, los módulos no van a aprender de forma adecuada.

Atendiendo al tipo de conexiones las redes pueden clasificarse en:

ANNs recurrentes, en las cuales una neurona o capa está enlazada consigo misma (su salida se convierte en una de sus entradas). Esto implica que las redes reciben información acerca de su comportamiento anterior, lo cual permite que aborden tareas en las cuales existe una componente temporal. Existen muchos tipos de ANNs recurrentes, y uno de ellos, la red de Elman [Elman, 1990], es la base del traductor RECONTRA.

ANNs no-recurrentes, en las cuales las conexiones se producen de la entrada a la salida sin aparición de bucles. Estas son las redes que se desarrollaron en primer lugar y las que se consideran clásicas.

3.3.1.2. Atendiendo a los mecanismos de aprendizaje

Hay diversos criterios posibles para diferenciar las reglas de aprendizaje de las redes. Un criterio se basa en considerar si la red puede aprender durante su funcionamiento habitual, o si el aprendizaje supone la desconexión del aprendizaje de la red.

En los casos que se trabaja con problemas estables y que se dispone de un gran número de muestras de entrenamiento, no hace falta que la red continúe aprendiendo durante su funcionamiento. Cuando el conjunto de entrenamiento es muy reducido o el problema altamente variable, puede resultar más interesante dejar que la red se adapte a los nuevos datos que va recibiendo.

Otro criterio suele considerar dos tipos de reglas de aprendizaje: las de aprendizaje supervisado y las correspondientes a un aprendizaje no supervisado. Estas reglas dan pie a una clasificación.

Las **ANNs con aprendizaje supervisado** requieren de un conjunto de datos de entrada cuya respuesta objetivo se conoce. El proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor comprueba la salida de la red y, en el caso de que ésta no coincida con la deseada, se procede a modificar los pesos de las conexiones, a fin de conseguir que la salida se aproxime a la deseada (Perceptrón simple, red Adaline y Perceptrón Multicapa).

Las **redes que utilizan aprendizaje no supervisado** no requieren influencia externa para ajustar los pesos de las conexiones entre neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada es o no correcta. Así, existen varias posibilidades para interpretar la salida de estas redes.

3.3.1.3. Atendiendo al tipo de asociación entre la información de entrada y la salida:

Existen dos formas primarias de realizar la asociación de entrada/salida en una red. Esto da lugar a una clasificación en dos tipos de redes neuronales, las redes heteroasociativas y las autoasociativas.

- Una red **heteroasociativa** se refiere al caso en el que la red aprende parejas de datos $[(A_1,B_1),(A_2,B_2),\ldots,(A_n,B_n)]$, de tal forma que cuando se presente cierta información de entrada A_i, deberá responder generando la correspondiente

salida B_i. Existen redes heteroasociativas con conexiones feedforward, feedforward/feedback y redes con conexiones laterales. También existen redes heteroasociativas multidimensionales y su aprendizaje puede ser supervisado o no supervisado.

- Una red **autoasociativa** se da cuando la red aprende ciertas informaciones A_1, A_2,…, A_n, de tal forma que cuando se le presenta una información de entrada, A_i' realizará una autocorrelación, respondiendo con uno de los datos almacenados, el más parecido al de la entrada A_i. Pueden implementarse con una sola capa, existen conexiones laterales o también autorecurrentes, y habitualmente son de aprendizaje no supervisado.

3.3.1.4. Atendiendo a la representación de la información de entrada y salida:

En esta categoría se distinguen tres tipos de redes: analógicas, discretas e híbridas.

- Las **redes analógicas** procesan datos de entrada de naturaleza analógica, valores reales continuos, habitualmente acotados y usualmente en el rango [-1,1] o en el [0,1], para dar respuestas también continuas. Las redes analógicas suelen presentar funciones de activación continuas, habitualmente lineales o sigmoides. Entre estas redes neuronales destacan las redes de *Backpropagation*, la red continua de Hopfield, la de Contrapropagación, la Memoria Lineal Asociativa, la *Brain-State-in-Box*, y los modelos de Kohonen (SOM y LVQ).
- Las **redes discretas** procesan datos de naturaleza discreta, habitualmente valores lógicos booleanos {0,1}, para acabar emitiendo una respuesta discreta. En este caso, las funciones de activación de las neuronas son de tipo escalón. Entre las redes binarias destacan la máquina de Boltzman, la máquina de Cauchy, la red discreta de Hopfield, el Cognitrón y el Neocognitrón.
- Finalmente, las **redes híbridas** procesan entradas analógicas para dar respuestas binarias; entre ellas destacan el Perceptrón, la red Adaline y la Madaline.

3.3.2. Funcionamiento

Dada la gran variedad posible de redes neuronales artificiales centraremos la explicación en el Perceptrón Multicapa (*Multi Layer Perceptron*, MLP), una de las ANNs más utilizadas y que resulta la base de gran parte de las ANNs modernas. En la Figura 3.4 se muestra un esquema conceptual de la neurona artificial. Como vemos en ella, una neurona i recibe una serie de entradas x_j a través de conexiones afectadas por un peso w_{ij} y emite una salida y_i. Esta salida viene dada por dos funciones:

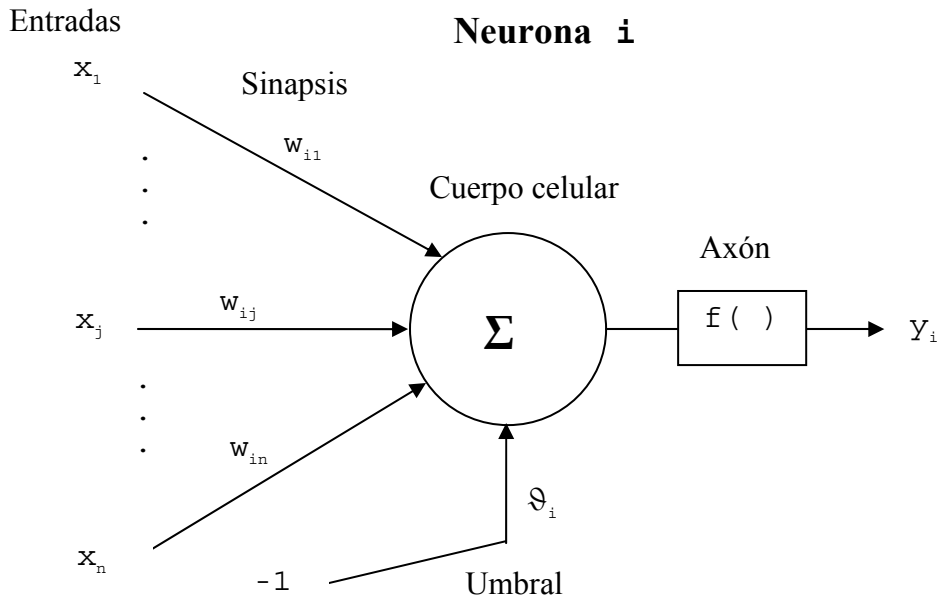

Figura 3.4. Neurona artificial (se pueden observar los nombres de los componentes biológicos, sinapsis, dendritas, cuerpo celular y axón, junto a los "equivalentes" artificiales).

Una función de **excitación** o propagación (ver ecuación 3.1), consistente en un sumatorio donde cada entrada es multiplicada por el peso de su interconexión. Las señales que llegan a la sinapsis son las entradas a la neurona; éstas son ponderadas de acuerdo a la sinapsis correspondiente. Se considera que el efecto de cada señal es aditivo, de tal forma que la entrada neta que recibe una neurona es la suma del producto de cada señal de entrada por el valor de la sinapsis que conecta ambas neuronas. En algunos casos se añade un umbral ϑ_i constante a la función. Señalar que esta, por lo general, es una función lineal.

$$\sum_j w_{ij} \cdot x_j - \vartheta_i \qquad (3.1)$$

Una función de **transferencia** para la activación o salida de la neurona: Esta función está asociada con cada neurona, de tal manera que para cada neurona i hay una función de salida que transforma el estado actual de activación en una señal de salida y_j. De este modo, las señales de entrada pueden excitar a la neurona (sinapsis con peso positivo) o inhibirla (peso negativo).

Si la suma de las entradas ponderadas (ver ecuación 3.1) es igual o mayor que el umbral de la neurona, indica que la relación entre las neuronas es excitadora, y en este caso, la neurona se activa (da salida). Si la suma es menor que el umbral, la sinapsis será inhibidora. En este caso, si la neurona está activada, se desactivará.

Finalmente, si la suma es 0, se supone que no hay conexión entre ambas neuronas. Esta es una situación de todo o nada: cada neurona se activa o no se activa.

La salida final de la neurona viene dada por el resultado de la función de transferencia f.

$$y_i = f\left(\sum_j w_{ij} \cdot x_j - \vartheta_i\right) \tag{3.2}$$

que puede ser lineal:

$$f(neur_i) = c \cdot neur_i \tag{3.3}$$

donde `neur`$_i$ es la salida de la neurona `i`, y `c` un valor constante; una función escalón:

$$f(neur_i) = \begin{cases} +1 & \text{si} \quad neur_i > 0 \\ -1 & \text{si} \quad neur_i < 0 \end{cases} \tag{3.4}$$

o no lineal, como la función sigmoidea, una de las más utilizadas es la función definida por:

$$f(neur_i) = \frac{1}{1 + \exp(-a \cdot neur_i + b)}, \quad a > 0 \tag{3.5}$$

donde `a` y `b` son valores constantes, `a` el parámetro de inclinación de la pendiente de la función y `b` el sesgo o bias.

Es importante tener en cuenta que para aprovechar la capacidad de las ANNs de aprender relaciones complejas o no lineales entre variables, es necesario utilizar funciones no lineales al menos en las neuronas de la capa oculta, dado que sin ellas las ANNs no podrán aprender correctamente el problema no lineal. Por su parte, la elección de la función de activación en las neuronas de la capa de salida depende del tipo de actividad que realicen.

Por ejemplo, en tareas de clasificación, las neuronas normalmente toman una función de activación sigmoidal o sigmoide (dado que su derivada es fácil de calcular y la función tiene un único punto de inflexión). Así, cuando se presenta un patrón que pertenece a una categoría particular, los valores de salida tienden a dar como valor 1 para la neurona de salida que representa la categoría de pertenencia del patrón, y 0 ó -1 para las otras neuronas de salida, dependiendo de que la función sigmoidal esté definida en el rango [0,1] ó [-1,1].

3.3.3. Aprendizaje y entrenamiento

Al igual que en el sistema biológico, el aprendizaje de una ANN puede ser explicado como la modificación del comportamiento producido por la interacción con el entorno y como resultado de las experiencias sufridas, conduciendo al establecimiento de nuevos modelos de respuesta a estímulos externos.

Para que se dé el aprendizaje, se debe partir de un conjunto de datos de entrada suficientemente significativo, para conseguir que la red aprenda automáticamente las propiedades deseadas. Durante este proceso, los parámetros de la red se adecuan a la resolución del problema, realizando ajustes en los pesos existentes entre las neuronas. Los cambios que se producen durante el proceso de aprendizaje “normal” se reducen a la modificación de los valores de los pesos, lo cual supone la modificación y destrucción de conexiones entre neuronas, cuando estas conexiones se ponen a cero.

Existen métodos más avanzados de aprendizaje que permiten la creación de conexiones entre neuronas, y la creación y destrucción de las propias neuronas. Se puede afirmar que el proceso de aprendizaje ha finalizado cuando los valores de los pesos permanecen estables.

Por otro lado, las modificaciones de los pesos pueden realizarse de dos formas: después de haber presentado todos los patrones de entrenamiento (aprendizaje por lotes o modo *batch*), o actualizando los pesos tras la presentación de cada patrón de entrenamiento (aprendizaje en serie o modo *on line*). En este último modo, es importante forzar a que la presentación de los patrones sea de forma aleatoria. Dado que si siempre se sigue un mismo orden, el entrenamiento estaría viciado a favor del último patrón del conjunto de entrenamiento, cuya actualización por ser la última, siempre predominaría sobre las anteriores.

Este entrenamiento repetido para todos los valores de entrada y salida, origina una representación interna del objeto en la red, que considera todas las irregularidades y generalidades del mismo.

Como hemos visto, uno de los métodos de entrenamiento de las ANNs consiste en reducir el error entre la salida esperada y la producida, pero aunque está más que demostrado que las redes convergen a un mínimo local, este mínimo no tiene por qué ser uno adecuado. La selección y tratamiento de los parámetros de entrenamiento es fundamental para lograr que las ANNs converjan a un mínimo local que obtenga buenos resultados.

3.4. Modelos conexionistas para el tratamiento del lenguaje natural.

Los modelos conexionistas han sido utilizados en multitud de tareas relacionadas con el tratamiento del lenguaje natural. Entre otras tareas determinar si una frase es correcta o no, predecir la siguiente palabra de una cadena, reconocer la lengua hablada o traducir entre distintos lenguajes.

Los modelos conexionistas han demostrado ser capaces de inducir lenguajes naturales [Lawrence, 1996] [Giles, 2001] en la forma de reconocedores (autómatas) de estados finitos, y de manera más general, traductores sencillos de estados finitos como máquinas de Mealy e incluso traductores más complejos, como las máquinas de Mealy extendidas.

En estas últimas, frente a las clásicas máquinas de Mealy, no se preserva ni la longitud ni la sincronía entre la frase a traducir y la frase traducida, algo fundamental si lo que se quiere es abordar la traducción entre lenguajes naturales. Ya en [Allen, 1987] se demostraba la capacidad de traducción de las ANNs abordando una tarea sencilla de traducción entre inglés y castellano.

3.4.1. Modelado del lenguaje natural

El modelado del lenguaje es el intento de caracterizar, capturar y explotar las regularidades del lenguaje natural. Este es un campo de amplia aplicación, dado que se pueden utilizar modelos del lenguaje para mejorar el rendimiento de los sistemas de reconocimiento del habla o en sistemas de traducción. En muchas aplicaciones resulta interesante restringir las posibles salidas del sistema a las que se correspondan con las posibilidades del modelo.

Generalmente, estos modelos están basados en n-gramas, de los cuales se calculan las probabilidades a partir de bases de datos con gran número de ejemplos de frases correctas. Una alternativa al cálculo de n-gramas a través de sus frecuencias de aparición son las ANNs.

El tiempo de entrenamiento es normalmente muy elevado, pero presentan dos ventajas interesantes a la aproximación convencional: un suavizado implícito de sus estimaciones y el número de parámetros no crece exponencialmente en función de la variable `n`.

Los primeros modelos conexionistas en utilizarse fueron MLPs [Kamura, 1989] y empíricamente se demostró que el modelo conexionista es equivalente al modelo de n-gramas convencional. Otros experimentadores llegaron a conclusiones similares [Castro, 2001], e incluso algunos lograron mostrar que ANNs podían aprender un modelo del lenguaje con un comportamiento incluso mejor que los modelos de n-gramas [Xu, 2000].

Otros modelos de ANNs ampliamente utilizados para modelar lenguajes son las redes recurrentes, especialmente las redes de Elman, que han demostrado su capacidad de inferir modelos de lenguaje adecuados para gramáticas regulares. Una de las ventajas de aplicar redes neuronales recurrentes, es que en principio se elimina la necesidad de decidir a priori el tamaño de `n`.

Experimentalmente, los modelos conexionistas han demostrado ser capaces de inducir lenguajes naturales [Lawrence, 1996], [Giles, 2001] en la forma de reconocedores autómatas de estados finitos. Ejemplos de aplicación al problema de reconocimiento del habla podrían ser [Bengio, 2001], [Schwenk, 2002] y [Bengio, 2003]. Otros ejemplos de estas capacidades pueden encontrarse en [Castaño, 1998] y [Rodriguez, 2003].

También se han empleado nuevos tipos de ANNs, por ejemplo redes LSTM (*Long Short Term Memory*) [Hochreiter, 1997] que han demostrado su capacidad de resolver problemas complejos relacionados con el lenguaje humano (por ejemplo, [Woellmer, 2009]) y las basadas en *Reservoir Computing* [Lukosevicius, 2007] que tienen capacidad de recordar.

En resumen, las ANNs pueden aprender modelos de lenguaje con resultados similares o mejores que los obtenidos con modelos estadísticos convencionales basados en n-gramas.

3.4.2. Creación de representaciones

Una de las tareas para las cuales también se han empleado las ANNs ha sido para la creación de representaciones de palabras o frases.

Las máquinas RAAM (*Recurrent AutoAsociative Memories*) han sido utilizadas en [Pollack, 1990] para obtener representaciones de secuencias, tanto de letras formando palabras como de palabras formando frases, y en [Chrisman, 1991] para obtener representaciones de frases. Con el modelo RAAM se han generado las codificaciones de las frases a partir de las activaciones que se han desarrollado en la capa oculta para fragmentos progresivamente mayores de ellas.

La arquitectura RAAM permite representar datos estructurados de tamaño variable usando una red de tamaño fijo. La RAAM básica puede codificar estructuras de tipo árbol de profundidad variable mientras el factor de ramificación esté limitado. En teoría no existe ningún límite a la profundidad del árbol o el grado de ramificación. En la práctica depende del tamaño de la red.

Los elementos (letras, palabras) de una lista (palabra, frase) se presentan a la red uno a uno de izquierda a derecha, junto con la representación desarrollada por la red para todos los elementos precedentes. Al presentarse el último elemento, se obtendrá una representación de toda la frase.

Decodificar la información consiste en ir presentando en la capa oculta la representación desarrollada por la red. A la salida de la red obtendremos una palabra y la representación del resto de la lista (frase). Este proceso se repite hasta obtener las codificaciones locales de todos los elementos (palabras) de la lista (frase).

Tanto Pollack como Chrisman [Pollack, 1990] y [Chrisman, 1991] obtuvieron buenos resultados de traducción con esta arquitectura, pero siempre en tareas con un léxico muy reducido. Los requerimientos de memoria cuando los vocabularios crecen son considerables.

Cuando las redes crecen demasiado, en algunos casos aparecen problemas de convergencia a un mínimo local aceptable, lo cual hace todavía menos aconsejable este método.

El método FGREP (*Forming Global Representations with Extended backPropagation*) fue presentado en [Miikkulainen, 1991] y se basa en considerar la capa de entrada como una capa de pesos más a entrenar. Así, se considera que las unidades de entrada son unidades que tienen como función de activación la función identidad.

Inicialmente, a cada palabra de las frases se le asigna una codificación al azar, es decir, la misma palabra tiene distintas codificaciones en cada frase. Según progresa el entrenamiento, las codificaciones de cada palabra tienden a converger[14]. Con el modelo

[14] Excepto cuando una palabra puede tener funciones distintas en distintas frases, en cuyo caso se tiende a desarrollar codificaciones distintas para cada función.

FGREP original, la red recibe como entrada una frase completa y a la salida, las palabras se representan con las mismas codificaciones.

En esta aproximación, la creación de las codificaciones y el aprendizaje de la tarea en sí se realizan a la vez. Este método no se puede aplicar directamente a tareas de traducción dado que la entrada y la salida de la red no son las mismas, y el método debería permitir generar a la vez la representación de los dos lenguajes. Sin embargo, se puede emplear como base para un traductor en el que se traduzcan frases completas de un lenguaje origen a un lenguaje destino, con un modelo FGREP proporcionando la representación de cada frase.

Una de las tareas más habituales a las cuales se destina el modelo es la codificación de palabras, pero con el objetivo de localizar palabras con la misma o muy similar representación, y por lo tanto que pertenezcan a la misma "clase" o "tipo".

Redes con dependencia de izquierda a derecha: Uno de los últimos desarrollos en codificación de datos utilizando ANNs se ha producido al adaptar la idea de redes de dependencia de izquierda a derecha (*left-to-right*) a las ANNs, creando una nueva topología (ver Figura 3.9).

Estas redes disponen a su entrada de una secuencia de palabras, pero no se realiza una conectividad completa entre la capa de entrada y la capa oculta, sino sólo parcial. Así, una palabra está conectada con la palabra en su mismo orden en la capa oculta y las correspondientes palabras a su derecha en la secuencia.

En diversos experimentos realizados por Bengio con MLPs con esta conectividad [Bengio, 2001] y [Bengio, 2003], este método ha demostrado su capacidad de desarrollar representaciones adecuadas para gran número de datos, como parte de un modelo del lenguaje para reconocimiento del habla.

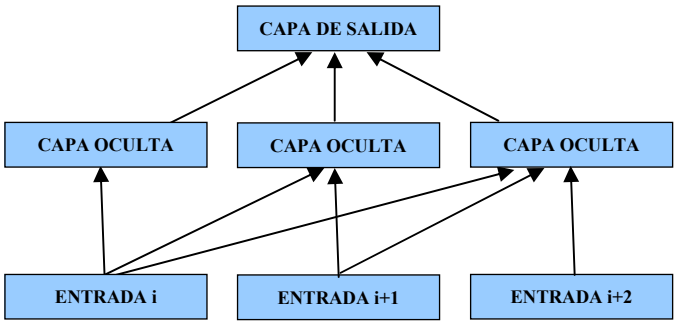

Figura 3.9. Perceptrón Multicapa con *adaptación de Left-to-Right* para tres palabras, *i-1*, *i* e *i+1*.

3.4.3. Traducción

Ya en [Allen, 87] se demostraba la capacidad de traducción de las ANNs abordando una tarea sencilla de traducción entre inglés y castellano. L. Chrisman [Chrisman, 1991] aproximaba el problema de la traducción mediante unas máquinas RAAM.

Para ello, inicialmente se codificaban por separado las frases del lenguaje destino en dos máquinas RAAM, y a partir de allí, se resolvían las traducciones siguiendo dos aproximaciones diferentes. La primera, la Traducción Transformacional, consistía en

entrenar otra herramienta conexionista que transformaba las representaciones desarrolladas por el codificador de las frases origen, en las correspondientes representaciones proporcionadas por el codificador de las frases destino.

La segunda, la Traducción Confluente, partiendo de las codificaciones de las frases origen y destino, entrenaba otra máquina RAAM doble que obtenía la misma codificación interna para la frase origen y la frase destino.

Los resultados alcanzados fueron prometedores con una tarea simple, con un vocabulario de 36 palabras en castellano y 36 en inglés y 216 pares de frases. No obstante, sería preferible un mecanismo que resolviera directamente y de manera más natural las traducciones.

Por otro lado, N. Koncar y G. Guthire [Koncar, 1994] plantearon la construcción incremental de un Perceptrón Multicapa, al que presentaban simultáneamente todas las palabras de la frase a traducir debiendo proporcionar a la salida, también en un solo paso, la frase traducida completa. Pese al éxito obtenido con una tarea sencilla (37 palabras en Inglés y 64 en Servo-Croata, con 10000 pares de frases de entrenamiento), ni la herramienta conexionista empleada (un perceptrón estático), ni el planteamiento para resolver la traducción parecen apropiados para enfocar procesos de AT más complejos.

Se necesitan, por lo tanto, modelos neuronales dinámicos capaces de manejar la componente del tiempo implícita en el procesamiento del lenguaje.

Dentro del proyecto JANUS [Waibel, 1991] se desarrolló una compleja arquitectura modular, jerárquica y recurrente que resolvía el problema de las secuencias de tiempo. Los aspectos sintácticos y semánticos de cada lenguaje (fuente y destino) se trataban separadamente, y por otro lado, se utilizaba un lenguaje artificial intermedio (o "interlingua") para las traducciones entre el lenguaje origen y el lenguaje destino.

En [Castaño, 1997] se presenta una aproximación intermedia entre los dos últimos planteamientos, que resuelve el problema de AT de una manera directa (sin interlingua) y con herramientas conexionistas dinámicas y sencillas. A su vez, en [Castaño, 1998] y [Castaño, 1999] se realizó un primer estudio preliminar de las aplicaciones de codificaciones con esta herramienta de traducción. Estos trabajos constituyen el punto de partida de la investigación realizada en el presente trabajo.

En los últimos años no ha habido muchos más trabajos en AT que utilicen únicamente ANNs. Sin embargo, sí que se han seguido empleandose en conjunción con otros métodos, como el presentado en [Prat, 2001a] y [Prat, 2001b], que utiliza ANNs en combinación con gramáticas para crear un sistema de traducción.

En [Langlais, 2005] se crea un sistema de traducción del habla, en el que una ANN sencilla, un Perceptrón Multicapa, ordena la lista de posibles transcripciones en función de su corrección y verosimilitud antes de pasarla a otro módulo del sistema. Esto se puede ver como una ANN que modela el lenguaje, y dada una frase, informa de hasta que punto es correcta dentro de este lenguaje.

3.5. Métodos de entrenamiento de los MLPs Codificadores

Como ya se ha mencionado en el apartado 3.4.2, para obtener representaciones en la capa oculta de la entrada de una ANN, el codificador debe ser entrenado para producir a la salida la misma palabra que se le presentaba a la entrada. En este trabajo se van a emplear MLPs para realizar la tarea de codificación de los vocabularios.

Los conjuntos de entrenamiento para los MLP codificadores se construyen a partir de los correspondientes corpora de aprendizaje de los traductores. Para ello, se extraen de estos conjuntos las diferentes palabras del vocabulario de un lenguaje dado junto con los posibles contextos en los que éstas aparecen.

Los tres métodos de entrenamiento que se han empleado en este trabajo con MLP fueron métodos supervisados: "Retro Propagación del Error" (*Backward Error Propagation*, BP), *Resilent Backpropagation* (RPROP) y *Scalated Conjugated Gradient* (SCG).

El método más utilizado de los tres en la literatura es el de Retro Propagación del Error. Este método puede presentar problemas de estabilidad, moviéndose en torno a un mínimo local sin ser capaz de localizarlo exactamente. Esto a que en la experimentación llevada a cabo se intente combinar BP con los otros dos métodos, especialmente SCG, intentando conseguir mantener la velocidad inicial de BP hacía un mejor mínimo local y la precisión posterior de SCG para localizarlo exactamente.

3.5.1. RetroPropagación del Error

El algoritmo de Retro Propagación del Error (*Backward Error Propagation* o *BP*) [Rumelhart, 1986] es el método de entrenamiento supervisado más común para MLP. En sí, el método es una implementación de la regla Delta, que se utiliza para modificar los pesos de un perceptrón de una única capa, dado el error producido por la red calculado a partir de la salida producida por la red (y_j) para una entrada (x_i) y la salida esperada (t_j) y un parámetro llamado factor de aprendizaje (α). En su forma simplificada para un perceptrón con una función lineal de activación es:

$$\Delta w_{ij} = \alpha \cdot \left(t_j - y_j\right) \cdot x_i \tag{3.6}$$

Definiendo el error para el perceptrón como

$$E = \sum_{j=1}^{n} \left(t_j - y_j\right)^2 \tag{3.7}$$

dónde n es el número de salidas del perceptrón. Podemos minimizar la función del error así definida utilizando el método de descenso del gradiente, que es un método de optimización.

Para encontrar el mínimo local de una función, el método de descenso del gradiente toma pasos proporcionales al negativo del gradiente (o la aproximación al gradiente) de la función en el punto actual. Evidentemente, el método requiere conocer la salida deseada

para cualquier entrada[15] y que la función de activación utilizada por las neuronas artificiales sea diferenciable.

La función final para calcular el gradiente con la derivada `g'` de la función de activación `g` para la neurona `i`, donde h_i es la salida de dicha neurona, es la siguiente:

$$\frac{\partial E}{\partial w_{ji}} = -\left(t_j - y_j\right) \cdot g'(h_j) \cdot x_i \qquad (3.8)$$

Por tanto la ecuación final para actualizar los pesos, tras añadir un valor de proporcionalidad (el factor de aprendizaje) y eliminar el símbolo negativo para poder minimizar el error y no optimizarlo es

$$\Delta w_{ji} = \alpha \cdot \left(t_j - y_j\right) \cdot g'(h_j) \cdot x_i \qquad (3.9)$$

Como implica su nombre, los errores (y por tanto la modificación de los pesos y el aprendizaje) se propagan hacia atrás desde las neuronas de salida hacia las internas. Técnicamente hablando el método de *backpropagación* se utiliza para calcular el gradiente del error de la red con respecto a los pesos modificables de la red. Este gradiente se utiliza entonces en un simple sistema descenso del gradiente para encontrar los pesos que minimizan el error.

Sumario de la técnica:

1. Se presenta un ejemplo de entrenamiento a la red neuronal.
2. Se compara la salida de la red al valor deseado para ese ejemplo. Se calcula el error para cada neurona de salida.
3. Para cada neurona se calcula que salida debería haber producido y un factor de escala que indica cuanto debe ajustarse la salida para coincidir con la deseada. Este es el error local.
4. Se ajustan los pesos de cada neurona para decrementar el error local.
5. Se reparte la "culpa" del error local de esta neurona a las neuronas que tiene de entrada (las del nivel previo) teniendo en cuenta los pesos de la neurona.
6. Se vuelve al paso 3 con las neuronas del nivel previo y utilizando como error la "culpa" asignada.

Para aumentar la rapidez de convergencia y evitar caer en mínimos locales muy malos, se utilizó también un momentum (η) para actualizar los valores de los pesos. Este momentum es un factor constante que tiene en cuenta la variación anterior de cada peso para la nueva actualización. De esta forma se refuerza el movimiento en un sentido y se limita la oscilación de los pesos. Así, la ecuación 3.9 se convierte en la expresión 3.10:

$$\Delta w_{ji}(t) = \alpha \cdot \left(t_j - y_j\right) \cdot g'(h_j) \cdot x_i + \eta \cdot \Delta w_{ji}(t-1) \qquad (3.10)$$

[15] Se trata por tanto de un método de entrenamiento supervisado.

La elección de estos parámetros (α, factor de aprendizaje y η, momentum) se puede realizar de muchas formas. En nuestra experimentación se realiza dentro del espacio bidimensional que se define, analizando el Error Cuadrático Medio (ECM) residual de un MLP entrenado tras un número determinado de presentaciones al azar del conjunto completo de entrenamiento.

3.5.2. RPROP

Resilent Backpropagation (RPROP) es un esquema de aprendizaje adaptativo local que realiza aprendizaje *batch* para MLP [Riedmiller, 1993]. Su principio básico es eliminar la influencia dañina del tamaño de la derivada parcial en el paso de modificación de los pesos. En consecuencia, sólo se considera el signo de la derivada para indicar la dirección de la modificación de los pesos en BP.

Cada vez que la derivada parcial cambia de signo, se interpreta como indicando que la última actualización era demasiado grande y el algoritmo ha saltado sobre un mínimo local. Por tanto, el valor del parámetro de actualización se reduce en el factor η^-. Si la derivada mantiene su signo, el valor η^+ se incrementa ligeramente para incrementar la velocidad de convergencia en las regiones llanas. La expresión resultante tiene el siguiente aspecto:

$$\Delta w_{ji}(t) = \begin{cases} \eta + \Delta w_{ji}(t-1) & , \text{si } \dfrac{\partial E(t-1)}{\partial w_{ji}} \cdot \dfrac{\partial E(t)}{\partial w_{ji}} > 0 \\ \eta - \Delta w_{ji}(t-1) & , \text{si } \dfrac{\partial E(t-1)}{\partial w_{ji}} \cdot \dfrac{\partial E(t)}{\partial w_{ji}} < 0 \\ \Delta w_{ji}(t-1), & \text{en cualquier otro caso} \\ & \text{donde } 0 < \eta - 1 < \eta + \end{cases} \qquad (3.11)$$

3.5.3. SCG

El *Scalated Conjugated Gradient* o SCG [Möller, 1993] es un método de aprendizaje supervisado para MLP y es un miembro de la clase de Métodos del Gradiente Conjugado (MGC). Los MGCs son técnicas de segundo orden de propósito general, es decir, que estos métodos utilizan la segunda derivada de la función objetivo para encontrar un buen mínimo local, pero a costa de un mayor coste computacional.

Los MGCs intentan acercarse al mínimo, pero no siempre se desplazan hacia abajo en el gradiente de la función del error, sino en la dirección que está conjugada (*conjugate*) a la dirección del paso previo. Así, la minimización realizada en un paso no es parcialmente deshecha en el siguiente paso, como ocurre en otros métodos. Es un método *batch*, es decir, que realiza la actualización de los pesos tras la presentación completa del conjunto de entrenamiento.

Más formalmente: sea w_i un vector definido en el espacio R^N, donde N es el número de pesos de la red. Sea E el error que deseamos minimizar.

Cada iteración k se calcula

$$w_{k+1} = w_k + \beta_k \cdot p_k \tag{3.12}$$

dónde p_k es la nueva dirección conjugada, y β_k es el tamaño del paso en esa dirección. En realidad, β_k es una función de la segunda derivada de la función del error, $E''(w_k)$ (matriz Hessiana). Dada la complejidad de calcular esta matriz, se evita su cálculo mediante la utilización de una aproximación del término s_k, fundamental en el cálculo de β_k:

$$s_k = E''(w_k) \cdot p_k \approx \frac{E'(w_k + \sigma_k \cdot p_k) - E'(w_k)}{\sigma_k}, \quad 0 < \sigma_k << 1 \tag{3.13}$$

La matriz Hessiana no tiene porqué estar definida positivamente, lo cual puede provocar que el algoritmo no tenga buenos resultados. Para mejorar el método se utiliza un escalar λ_k que regulariza la indefinición de la Hessiana y que debe ajustarse en cada iteración:

$$s_k = \frac{E'(w_k + \sigma_k \cdot p_k) - E'(w_k)}{\sigma_k} + \lambda_k \cdot p_k \tag{3.14}$$

3.5.4. Métodos de Poda del Perceptrón Multicapa

El proceso de poda en una ANN es la eliminación de las conexiones entre neuronas y/o de las propias neuronas que se consideran "innecesarias" por algún criterio. Para nuestros propósitos nos interesa eliminar neuronas en la capa oculta, reduciendo así el tamaño de las codificaciones que posteriormente extraeremos.

En este trabajo exploramos dos métodos distintos para podar las codificaciones: *Skeletonization* [Mozer, 1990] y el método de Poda de Unidades No Contributivas [Dow, 1991].

El método denominado *Skeletonization* funciona de la siguiente forma: dado un MLP determinado, este método decide qué unidad eliminar analizando el cambio que se produciría en la función del error si se eliminase cada unidad. Para cada neurona i se introduce una "fuerza de atención" λ_i que lleva a una expresión diferente para la entrada de cada unidad j:

$$net_j = \sum_{i=1}^{n} \omega_{ij} \cdot \lambda_i \cdot y_j \tag{3.15}$$

dónde net_j es la entrada a la unidad j de la red MLP, $\bullet_{ij}$ es el peso correspondiente a la conexión entre las neuronas i y j, λ_i es la fuerza de atención de la unidad i (podrá tomar los valores 0 ó 1) e y_i es la salida de la unidad i.

Esto permite definir la relevancia de una unidad i (ρ_i) como el cambio en la función del error que se produce al remover la unidad (λ_i toma el valor 0) se obtiene

$$\rho_i = E\ \lambda_i \tag{3.16}$$

dónde $E\lambda_i$ es la función del error. Así, la relevancia de una neurona i nos permite calcular el efecto que produce a la red la eliminación de dicha neurona.

Para calcular la medida de salida o expulsión de una neurona se utiliza la función lineal del error[16]

$$E = \sum_{j=1}^{n} \left| t_{pj} - y_{pj} \right| \tag{3.17}$$

dónde t_{pj} es el valor esperado para la neurona j de la capa p e y_{pj} es el valor producido como salida por la neurona.

La forma en la que este método se ha aplicado en la experimentación puede verse en el algoritmo 3.1:

Dado MLP_0 un MLP inicial, siendo CO_{MLPi} el tamaño de la capa oculta de un MLP_i., para $i=0$ inicialmente.
Mientras $CO_{MLP_i} \neq 0$
    Se entrena MLP_i
    Se aplica *Skeletonization* sobre la capa oculta de MLP_i
FinMientras
Se selecciona el MLP_i con menor ECM residual entre i=0,1,… CO_{MLPi}

Algoritmo 3.1. Método de aplicación del algoritmo de poda *Skeletonization* ([Mozer, 1990]) sobre MLP en los experimentos.

Con el método de poda de "Unidades No Contributivas", la idea fundamental es eliminar las neuronas que realizan la misma función que otras de la misma capa. Para determinar si dos neuronas realizan el mismo proceso se tiene en cuenta la salida producida ante las mismas entradas.

En concreto se aplican una serie de reglas para determinar que unidades eliminar:

- Las neuronas que no varían la salida, es decir, que tienen una salida constante.
- Las neuronas que muestran siempre la misma salida que otra neurona de la misma capa.
- Las neuronas que muestran siempre la salida opuesta que otra neurona de la misma capa.

La forma en la que este método se ha aplicado en la experimentación puede verse en el algoritmo 3.2:

[16] Observemos que en la formula (3.7) se eleva al cuadrado esta diferencia, mientras que la ecuación (3.17) simplemente cambia el signo caso de ser negativa. El objetivo es el mismo: trabajar siempre con números positivos.

Dado MLP_0 un MLP inicial, siendo CO_{MLPi} el tamaño de la capa oculta de un MLP_i., para $i=0$ inicialmente.
Mientras $CO_{MLP_i} \neq 0$ y $ECMresidual_i <= ECMresidual_{i-1}$
    Se entrena MLP_i hasta que se cumple uno de los criterios sobre las unidades de la capa oculta de MLP_i
FinMientras
Se selecciona el MLP_i con menor ECM residual entre i=0,1,… CO_{MLPi}

Algoritmo 3.2. Método de aplicación del algoritmo de poda Unidades No Contributivas ([Dow, 1991]) sobre MLP en los experimentos.

El problema fundamental que tiene este método en comparación con *Skeletonization* es que el tiempo de entrenamiento necesario es mucho mayor. Hay que tener en cuenta que el entrenamiento entre podas debe continuar hasta que se cumplan una de las condiciones. No tiene porqué ser un número reducido de iteraciones. Para limitar esto se ha incluido en el algoritmo 3.2 una condición de parada en función del ECM residual, como en el método de *Skeletonization*.

3.5.5. Entrenamiento del modelo FGREP codificador

Se realizaron ciertos experimentos con FGREP, siguiendo los experimentos originales de Miikkulainen [Miikkulainen, 1989]. Como en ellos se utilizó una adaptación del algoritmo de Retropropagación del Error [Rumelhart, 1986] que modifica también los valores de las entradas y las salidas a la red. Esto se logra simplemente haciendo que BP considere los valores de entrada como una unidad más.

Más formalmente, si definimos el error como:

$$\partial_{1i} = \sum_j \partial_{2j} \cdot \omega_{1ij} \tag{3.18}$$

dónde $\bullet_{xy}$ es el error para la unidad `y` en la capa `x`, y ω_{iij} es el peso entre la unidad `i` en la capa de entrada (`1`) y `j` en la primera capa oculta. Ahora las representaciones se pueden modificar de acuerdo con este error:

$$\Delta r_{ci} = \alpha \cdot \partial_{1i} \tag{3.19}$$

dónde r_{ci} es el componente de representación `i` del objeto `c`, y δ_{1i} es el error para la unidad `i` en la primera capa (`1`), y α el factor de aprendizaje.

Estos valores pueden ser muy grandes (o muy pequeños), puesto que los pesos no tienen límites de tamaño lo cual lleva a que se pongan límites superior e inferior[17] a los valores que toman las codificaciones.

[17] Aunque en el artículo original no se menciona es evidente que habitualmente estos parámetros toman valores 0 ó 1.

3.5.6. Métodos para detener el entrenamiento

Existen varios criterios utilizados en la bibliografía para decidir cuándo se ha de detener el proceso de aprendizaje:

- El proceso de entrenamiento se puede detener cuándo el Error Cuadrático Medio cometido sobre el conjunto de aprendizaje (ECM residual) se haya estabilizado, incluso aunque este error no llegue a tomar el valor cero.
- El entrenamiento finalizará cuando la red traduzca correctamente un determinado número de cadenas consecutivas, generalmente llamado en la bibliografía "margen".
- Se utiliza un conjunto de cadenas no vistas en el entrenamiento como validación, de forma que se detiene el entrenamiento cuando el ECM sobre este conjunto alcanza un mínimo. A este método se le denomina Validación Cruzada (*Cross Validation*) y se considera especialmente efectivo, a pesar de que requiere unos conjuntos de entrenamiento grandes. Dado el tamaño de las tareas estas técnicas no se emplearon excepto de forma puntual.
- Tras un número predeterminado de presentaciones del corpus de entrenamiento.

En la mayor parte de los experimentos presentados en este trabajo, dada la facilidad de uso del método y para facilitar la comparación con experimentos anteriores presentados en [Castaño, 1997], [Castaño, 1998] y [Castaño, 1999], se emplea una combinación del primer y cuarto métodos. Así, se detiene el entrenamiento del traductor RECONTRA cuando el ECM ya no disminuye por debajo de un umbral o tras un número predeterminado de presentaciones del corpus de entrenamiento.

Cuando se utiliza SCG, dada su precisión a encontrar un mínimo local antes de alcanzar el máximo número de iteraciones permitidas, también se detiene el entrenamiento cuando el ECM se queda fijo en un valor.

Los valores concretos de parada se deciden para cada tarea tras un experimento inicial y observar la evolución del ECM y los errores de traducción.

Detener el entrenamiento tras un número predefinido de iteraciones puede llevar en algunos casos a problemas de sobreentrenamiento, dependiendo de la tarea y el experimento concreto. Sin embargo para la creación de codificaciones mediante redes, una vez producido el entrenamiento, se extraen las codificaciones para cada palabra y se prescinde de las redes. Dado que no van a aparecer nuevas palabras en los vocabularios la importancia de un posible sobreentrenamiento o sobreajuste a estas palabras es menor. Ciertamente las palabras pueden aparecer en el test en contextos no vistos en el conjunto de entrenamiento, pero incluso en ese caso la ventana de entrada de RECONTRA (y la de salida, cuando presente) aporta información al traductor y puede compensar este efecto.

Con RECONTRA el problema puede tener mayor importancia, pero la elección de los valores es conservadora, por lo que el entrenamiento tiende a detenderse antes de llegar al máximo.

Capítulo 4.

El modelo RECONTRA

Resumen: En este capítulo se explican las características del modelo RECONTRA, así como el estado del arte en su utilización antes del trabajo que ha dado lugar a esta tesis. Veremos las características del modelo, como se entrena y su funcionamiento en general; además hablaremos acerca de cómo integrar diversos modelos. Finalmente, se muestran los resultados previos de traducción obtenidos con RECONTRA[18].

4.1 Topología del modelo

El traductor conexionista RECONTRA presentado en [Castaño, 1997], está basado en la Red Neuronal Recurrente Simple (*Simple Recurrent Neural Network*, SRNN) de Elman [Elman, 1990].

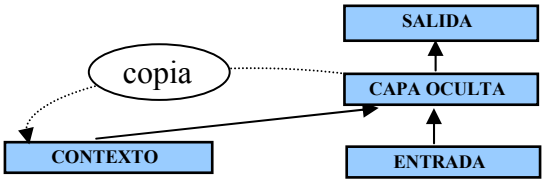

Figura 4.1. Red de Elman.

La arquitectura desarrollada por Elman incorpora una recurrencia simple en un MPL mediante la reintroducción en la capa oculta de las activaciones de esta capa oculta en el instante de tiempo anterior. Así, se consigue que la capa oculta tenga una memoria explícita de su salida en instantes precedentes. Esta arquitectura puede observarse en la Figura 4.1. La capacidad de "recordar" las anteriores activaciones de la capa oculta de la red, le permite abordar con éxito problemas en los cuales la componente temporal es importante. Algunos ejemplos de varias tareas en las cuales se ha empleado con éxito están recogidos en [Übeyli, 2008]. Las redes de Elman han sido utilizadas para abordar

[18] Constituye un resumen muy condensado de los resultados presentados en [Castaño, 1998].

diversas tareas de tratamiento del Lenguaje Natural (LN), especialmente en el modelado del lenguaje como ya se ha comentado en el punto 3.4.

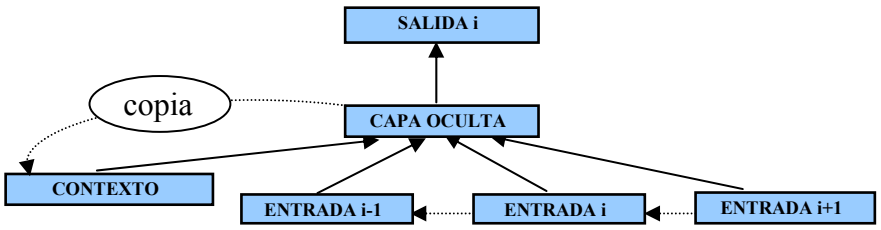

Figura 4.2. Traductor conexionista RECONTRA con ventana de entrada de tamaño 3.

El traductor conexionista RECONTRA es una red de Elman a la que se incorpora una ventana con retrasos y adelantamientos de tiempo en la entrada de la red. La topología resultante puede verse en la Figura 4.2. Así pues, en RECONTRA se utilizan dos métodos para representar el tiempo, redes dinámicas y ventanas deslizantes en la entrada. Con las recurrencias simples de la red de Elman se consigue que la capa oculta tenga una memoria explícita de su salida en instantes precedentes. La incorporación de una ventana a la entrada de la red permite incrementar la información contextual de la señal de entrada, pudiendo tener en cuenta tanto las palabras precedentes como las posteriores a la palabra de entrada ,en la frase del lenguaje origen a traducir.

Al traductor se le presentan secuencialmente las palabras de la frase de entrada y se entrena para que proporcione, también secuencialmente y de manera continua, las palabras que constituyen la frase traducida. Los retrasos de tiempo introducidos en la capa de entrada permitirán ver a la red simultáneamente `m+1+n` palabras consecutivas de la frase de entrada, las palabras `i-m,...,i-1,i,i+1,...,i+n`; en el siguiente paso se presentarán las palabras `i-m+1,...,i-1,i,i+1,...,i+n+1` y así sucesivamente; esto es, el traductor irá viendo la frase de entrada a través de una ventana de `m+1+n` palabras que se desplaza palabra a palabra, y proporcionará una a una las sucesivas palabras de la correspondiente frase traducida.

La Tabla 4.1 muestra una posible presentación a la red de la frase en español "¿ les importaría bajar el equipaje a recepción ?*"* y las correspondientes palabras en inglés que debería mostrar la red en cada momento a la salida (activaciones de salida deseadas), asumiendo que la traducción al inglés de dicha frase es "*would you mind sending the luggage down to reception ? FinFrase*".

Paso	Ventana de palabras de entrada			Salida
1	…….	¿	Les	would
2	¿	les	importaría	you
3	les	importaría	bajar	mind
4	importaría	bajar	el	sending
5	bajar	el	equipaje	the
6	el	equipaje	A	luggage
7	equipaje	a	recepción	down
8	a	recepción	?	to
9	recepción	?	…….	reception
10	?	……..	…….	?
11	…….	…….	…….	FinFrase

Tabla 4.1. Sucesivas entradas y salidas deseadas de un traductor RECONTRA con una ventana de 3 (1+1+1) palabras ante una frase de entrada.

Nótese que el tamaño de la ventana tiene que ser suficiente para que la red siempre haya visto la palabra que debe presentar traducida a la salida. Sin embargo, no parece preciso que la palabra o palabras del lenguaje origen relacionadas con la palabra traducida en la salida estén justamente en la ventana de entrada, ya que en principio, la red dispone de memoria y es capaz de recordar eventos pasados.

Con el fin de identificar el final de la frase traducida por la red, se añade al vocabulario del lenguaje destino una palabra específica, "FinFrase", dedicada a tal efecto. De esta manera, el proceso de presentación de la frase de entrada finalizará cuando la red proporcione dicha palabra como salida, o en su defecto, tras la presentación a la red de un número predeterminado de ventanas de entrada completamente vacías (con palabras en blanco).

El traductor RECONTRA puede clasificarse como un traductor de tipo unidireccional y bilingüe [Hutchins, 1992], pudiendo traducir únicamente entre dos lenguajes y en un sentido. No obstante, cuando ampliamos la consideración del sistema para incluir los codificadores automáticos, podemos ver claramente como el sistema puede considerarse multilenguaje, teniendo una codificación para cada lenguaje implicado en la tarea y un modelo RECONTRA para realizar la traducción entre cada par de lenguajes y en cada sentido.

4.2. Algoritmos de entrenamiento

El principal algoritmo adoptado en este trabajo para entrenar las ANNs de Elman que constituyen el traductor RECONTRA es una adaptación del algoritmo de retropropagación del error [Rumelhart, 1986]. Este algoritmo está adaptado para no tener en cuenta las conexiones recurrentes de la red, y por tanto, trunca el gradiente del error. Pese a no calcular el gradiente de forma exacta, en la práctica este algoritmo truncado da buenos resultados y se ha utilizado en multitud de campos.

En la experimentación presentada en este trabajo los pesos de RECONTRA se actualizan entrada-a-entrada *(on-line)* tras procesar cada par de entrenamiento *ventana de*

entrada # palabra de salida traducida, procediendo así hasta que se acaba el conjunto de entrenamiento.

El traductor ha de mostrar a la salida secuencialmente las sucesivas palabras de la frase traducida. Obviamente, la presentación de cada palabra de la frase de entrada y de cada palabra de la frase destino se realiza a partir de las codificaciones creadas para cada experimento y cada vocabulario.

Se añade una nueva palabra al vocabulario de los lenguajes tanto de origen como de destino, "PalabraVacía" (@), con el fin de rellenar la ventana de entrada por la izquierda y por la derecha cuando sea necesario, y reducir el problema de la diferencia de tamaños entre las frases del lenguaje origen y destino.

Para RECONTRA, cuando exista una ventana de entrada, por ejemplo de tamaño `2` a la izquierda y `2` a la derecha, el vector `i-ésimo` de entrada a RECONTRA de dicha frase consistirá en el elemento (palabra) `i-2`, el elemento `i-1`, el elemento `i`, el `i+1` y el `i+2` de la frase y el vector de salida será el elemento (palabra) `i-ésimo` de la frase destino: `i-2 i-1 i i+1 i+2`. Si no existen estas palabras, se empleará la palabra especial P*alabraVacía* (@).

Una de las variantes de RECONTRA desarrolladas en este trabajo presenta una diferencia fundamental: la presencia de una ventana de salida en el traductor (la red resultante puede observarse en la figura 4.3).

El hecho de que el traductor no produzca una única palabra sino varias lo aproxima conceptualmente a ciertos tipos de traductores basados en la traducción de fragmentos de frases (*phrase-based*) y no en la traducción palabra a palabra.

Desde un punto de vista más conexionista, al obligar a la red a producir varias palabras a la vez se puede interpretar como que se la está obligando a tener más en cuenta el lenguaje de salida en la traducción.

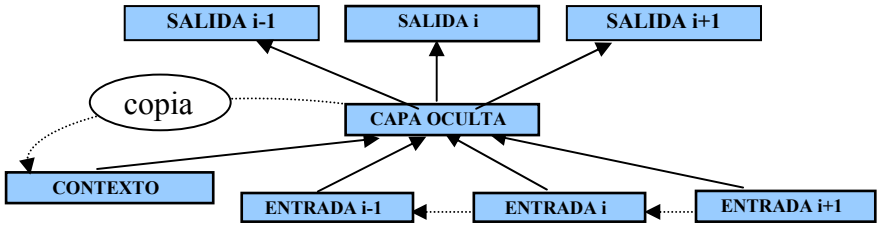

Figura 4.3. Traductor conexionista RECONTRA con ventana de entrada y de salida de tamaño 3.

En cualquier caso, al comienzo de cada frase de entrada se procede a la inicialización de las unidades de contexto con el fin de que la red no disponga inicialmente de información contextual alguna. Teniendo en cuenta que en la experimentación se adoptará la función sigmoide asimétrica definida entre 0 y 1 como función de activación no lineal, el valor inicial de cada neurona de contexto será de 0.5. Por otro lado, antes de comenzar el entrenamiento de la red, las conexiones de ésta se inicializan a valores aleatorios dentro del rango [-0.01,+0.01] ó [-0.1,+0.1].

Existen dos parámetros principales a seleccionar en los modelos RECONTRA, el tamaño de la capa oculta y el tamaño de la ventana de entrada y dos parámetros en el algoritmo de entrenamiento, el factor de aprendizaje y el momentum.

4.3. Ensembles de RECONTRA

Uno de los objetivos de la tesis es la mejora de los resultados de traducción. En general, una forma de obtener mejores prestaciones con ANNs es mediante la combinación de varias redes distintas en una sola. Esta combinación de redes puede realizarse de muchas formas y ha recibido varios nombres: *ensembles*, *mixture of experts* o *redes modulares* [Jacobs, 1991] (ya mencionadas en el punto 3.3.1). Parte del trabajo presentado en la tesis consiste en la creación y uso de diversos tipos de ensembles de modelos RECONTRA para mejorar los resultados de traducción.

Recordemos que un ensemble de ANNs es un conjunto de ANNs diferentes que juntas resuelven un problema [Zhi-Hua, 2002]. Para que los resultados del ensemble sean mejores que los proporcionados por los componentes, estos deben ser diversos y debe tenerse un método adecuado para combinar los resultados.

La necesaria diversidad en los componentes[19] se puede lograr de muchas formas: alterando el conjunto de entrenamiento para cada componente o utilizando distintas topologías o tipos de redes para cada componente. Por ejemplo, en el campo de la clasificación, se ha comprobado como la combinación de clasificadores produce mejores resultados que un único clasificador [Kuncheva, 2004].

Desde el punto de vista del problema de la clasificación, dada una entrada $\mathbf{x}$, cada clasificador (traductores en nuestro caso) puede generar una salida distinta, algunas de las cuales serán correctas y otras no. El método de combinación seleccionará en este momento una de las salidas o las combinará de alguna forma para formar la salida final. El conjunto de técnicas combinatorias se ha extendido desde el campo de la clasificación a otros campos [Zhi-Hua, 2002].

Una de las ventajas principales de los ensembles es que permiten utilizar técnicas de traducción diferentes, sin necesidad de limitarse a las ANNs. Por ejemplo, para un ensemble de dos componentes se podría utilizar directamente un diccionario como uno de de ellos y el traductor conexionista para el otro, con otra ANN decidiendo que salida es la correcta, la producida por el diccionario o por la red.

Así para nuestro propósito, la traducción, vamos a combinar distintos modelos RECONTRA, aunque podríamos combinar tipos distintos de traductores.

Una discusión permanente es si está justificado el uso de ensembles. Al combinar clasificadores buscamos unos mejores resultados de clasificación a costa de un incremento de la complejidad. Algunos investigadores consideran que en lugar de buscar

[19] El procedimiento no tendría sentido si cada componente fuese igual y proporcionase siempre los mismos resultados.

las mejores características y el mejor clasificador, con ensembles se busca el mejor conjunto de clasificadores y el mejor método de combinarlos, olvidando que existen problemas básicos por solucionar para cada clasificador.

Existen tres razones principales para utilizar ensembles [Kuncheva, 2004]:

Estadísticas: podemos tener varios traductores con resultados de entrenamiento similares pero distinta generalización. Esto supone que algunos darán resultados de traducción sobre el conjunto/s de test peores que otros, o incluso muy malos. Obtener la media de varios clasificadores elimina este problema.

Computacionales: algunos algoritmos encuentran mínimos locales[20]. Si logramos encontrar una forma de combinar o agregar estos mínimos, esto podría acercarnos al mínimo global.

Representacionales: entrenar un ensemble para llegar a una precisión determinada es mejor que entrenar directamente un clasificador de gran complejidad.

Como resulta evidente las tres razones son aplicables para RECONTRA:

- Las ANNs encuentran un mínimo local, no el mínimo global.
- Dado que los mínimos locales probablemente serán distintos, las generalizaciones también lo serán. Con lo cual combinarlas con éxito amplia la generalización del sistema.
- Experimentalmente se ha comprobado en multitud de problemas que al aumentar el tamaño de las redes se incrementan los problemas de convergencia y estabilidad.

En general hay cuatro posibilidades distintas para construir ensembles, según en qué parte del proceso decidamos centrarnos:

1. **Diseñar diferentes combinadores**: la primera idea para combinar las salidas de diversos componentes sería simplemente calcular algún tipo de votación.
2. **Usar traductores distintos**: la utilización de distintos parámetros de entrenamiento o topologías ya crearía traductores distintos.
3. **Usar diferentes espacios de características**: aunque no son estrictamente distintos subespacios, utilizar distintas codificaciones para las palabras sí que supone tener distintas visiones del conjunto de datos.
4. **Usar diferentes subconjuntos de datos**: dividir el conjunto de entrenamiento en tantas partes como componentes del ensemble deseemos.

Genéricamente, las estrategias para combinar clasificadores son la fusión y la selección.

[20] Los que se utilizan en el entrenamiento de redes neuronales, por ejemplo.

En la **fusión** cada elemento del ensemble se supone que debe conocer el espacio completo de entrenamiento y las salidas proporcionadas por cada elemento pueden combinarse.

En la **selección** cada elemento está especializado en una sección de los datos y/u objetos e idealmente, según la sección a la cual pertenezcan los datos de entrada, se debe seleccionar la salida del componente especializado en esa sección.

En la experimentación presentada en esta tesis se han utilizado varios métodos distintos, tanto de fusión como de selección.

En nuestra experimentación utilizamos varios de estos métodos para crear nuevos sistemas de traducción, que podemos organizar en tres principios distintos: sistemas basados en mezcla de expertos, sistemas basados en distancia y sistemas con ANNs como integradoras.

4.3.1. Sistemas expertos

En la experimentación realizada en estos sistemas, disponemos de dos redes distintas entrenadas con conjuntos de entrenamiento distintos.

Se utiliza el conocimiento humano para separar los pares de frases interrogativas (aquellas en las cuales aparece el símbolo '¿') de aquellas no interrogativas. Esto nos sirve también como método de combinación de resultados, indicando la presencia del símbolo '¿' si se debe aceptar como respuesta la salida de una red o de la otra siendo así un método de selección.

Por otra parte se podría clasificar el sistema de traducción resultante como un método mixto, dado que aplica conocimiento "externo" y técnicas automáticas para abordar la tarea, o también que se abordan dos tareas distintas más sencillas, una con las frases interrogativas y otra con las no interrogativas. En la figura 4.4 puede observarse un diagrama de este sistema.

Como es evidente, al utilizar el conocimiento humano para separar los pares de frases interrogativas de aquellas no interrogativas, se podría considerar el sistema como uno en el cual se abordan dos tareas distintas más sencillas que la combinada, una con las frases interrogativas y otra con las no interrogativas.

Es interesante señalar que la diferencia entre los dos tipos de frases no se limita a este símbolo, sino que la estructura tanto en castellano como en inglés también es distinta: las interrogativas "Auxiliar Sujeto Verbo Objeto" y las frases afirmativas tienden a tener la estructura "Sujeto Verbo Objeto"[21].

[21] Aunque no todas las no-interrogativas son afirmativas y siguen esta estructura.

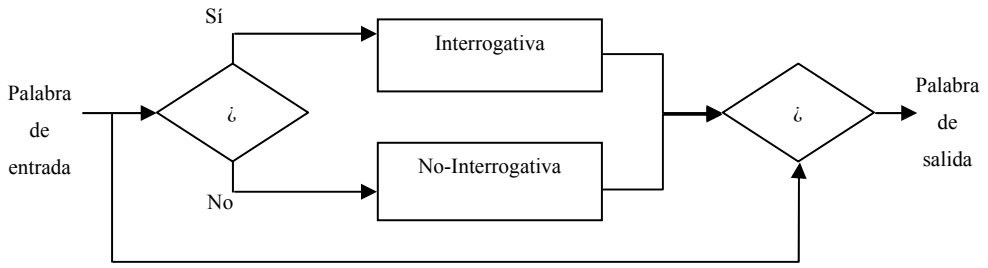

Figura 4.4. El sistema de expertos: una regla sencilla (la presencia de '¿') y dos modelos RECONTRA distintos.

4.3.2. Sistemas basados en distancia

Si consideramos que los modelos RECONTRA producen una salida real que interpretamos como la palabra con la codificación más próxima del vocabulario, la distancia entre estos valores (el producido y el de la codificación más próxima) puede interpretarse como una medida de la confianza que la red tiene en la salida.

La forma más sencilla de utilizar este criterio es comparar las distancias obtenidas con las distintas salidas y quedarnos con la red (palabra) cuya distancia sea mínima para cada palabra. Así, eliminaríamos las palabras de las cuales las redes no se sienten "seguras".

Se proponen dos variantes de este sistema, en las cuales, en lugar de comparar las distancias palabra a palabra, se comparan las distancias acumuladas, en un caso frase a frase, y en el otro, palabra a palabra.

Sin embargo, hay varios aspectos a tener en cuenta:

1. Si las codificaciones tienen distinto tamaño, para poder compararlas, es necesario normalizarlas dividiendo esta distancia entre el número de unidades/bits que se utilizan para cada codificación.
2. Se debe decidir si se quiere seleccionar la salida del sistema palabra a palabra o toda la frase, en cuyo caso sumaríamos las distancias acumuladas de cada palabra para cada red.
3. En caso de decidir seleccionar toda la frase, se presenta la decisión de si se deben tener en cuenta las palabras generadas tras la aparición de "FinFrase"[22].

Más formalmente, para seleccionar una palabra $\texttt{pal}_i$ dadas $\texttt{m}$ redes, debemos seleccionar aquella cuya distancia a la codificación ($\texttt{dispal}_i$) sea mínima, dada por la ecuación 4.1.

$$\text{distpal}_i = \min\left(\text{dispal}_1, \text{dispal}_2, ..., \text{dispal}_m\right) \qquad (4.1)$$

Como ya se ha comentado anteriormente, si las codificaciones no tienen el mismo tamaño debemos normalizarlas. Así, si llamamos $\texttt{t}_i$ al tamaño de la codificación empleada en la red $\texttt{i}$, la expresión resultante es (4.2):

[22] Como ya se ha comentado, esta es la palabra especial que marca el final de la frase de salida.

$$\text{distpal}_i = \min\left(\frac{\text{dispal}_1}{t_1}, \frac{\text{dispal}_2}{t_2}, \ldots, \frac{\text{dispal}_m}{t_m}\right) \quad (4.2)$$

Otro método de distancia es mediante uso del fragmento de las frases previamente producido por las redes, que sirve para seleccionar las palabras una a una ($\texttt{pal}_j$) a partir de las distancias acumuladas hasta esa palabra (`j`), formalmente se aplica la formula 4.3.

$$\text{distpal}_j = \min\left(\sum_{i=0}^{j}\text{dispal}_{1i}, \sum_{i=0}^{j}\text{dispal}_{2i}, \ldots \sum_{i=0}^{j}\text{dispal}_{mi}\right) \quad (4.3)$$

Para seleccionar una frase completa, se aplica la fórmula 4.4, donde `n` es el tamaño de la frase, `m` el número de redes que forman el ensemble (siempre igual o mayor que 2) y $\texttt{dispal}_{xi}$ la distancia entre la salida producida por la red `x` para la palabra de entrada `i` y la palabra más cercana del vocabulario dividida por el número de unidades utilizadas en la codificación para esa red (es decir, ya normalizada). Esta fórmula nos devuelve la distancia menor entre las frases implicadas y a partir de ella seleccionamos la frase completa (`fs`) correspondiente.

$$\text{distfs} = \min\left(\sum_{i=0}^{n}\text{dispal}_{1i}, \sum_{i=0}^{n}\text{dispal}_{2i}, \ldots \sum_{i=0}^{n}\text{dispal}_{mi}\right) \quad (4.4)$$

Es importante señalar que con esta fórmula (4.4) no se detiene la suma al aparecer la palabra especial “FinFrase”, sino que se consideran todas las salidas producidas por la red[23]. Definiendo $\texttt{pos}_{jf}$ como la posición en la cual se encuentra la palabra “FinFrase” en la red `j` se logra la ecuación (4.5).

$$\text{distfs} = \min\left(\frac{\sum_{i=0}^{\text{pos}_{1f}}\text{dispal}_{1i}}{\text{pos}_{1f}}, \frac{\sum_{i=0}^{\text{pos}_{2f}}\text{dispal}_{2i}}{\text{pos}_{2f}}, \ldots, \frac{\sum_{i=0}^{\text{pos}_{mf}}\text{dispal}_{mi}}{\text{pos}_{mf}}\right) \quad (4.5)$$

$$\text{pos}_{jf} = \text{FinFrase} \quad (j = 1, 2, \ldots, m)$$

Intuitivamente, con las distancias (4.1), (4.2) y (4.3) seleccionamos palabra a palabra de redes potencialmente distintas, mientras que con las distancias (4.4) y (4.5), seleccionamos todas las palabras de una misma red.

Por otro lado se puede combinar la división de la tarea en frases interrogativas y no-interrogativas y la utilización de distancias. Para esto existen tres posibilidades:

1. Dados dos (o más) sistemas expertos se pueden utilizar los criterios de distancia para combinar las distintas salidas, como si los sistemas expertos fuesen un traductor más.

[23] Las distintas redes tienen siempre el mismo número de palabras de salida, mientras que *FinFrase* puede aparecer en momentos distintos, lo cual implica normalizar las distancias.

2. En un sistema experto, en lugar de emplear la regla de decisión (presencia de '¿' en la entrada) para seleccionar la salida correcta, se emplea alguna medida de distancia.
3. Cada uno de los dos componentes de un sistema experto puede estar formado por varias redes distintas. A las salidas de estas redes se les aplica alguno de los métodos de distancia, y posteriormente la regla decide cual de las salidas de los dos componentes es seleccionada. En la Figura 4.5 puede observarse un ejemplo de la red resultante.

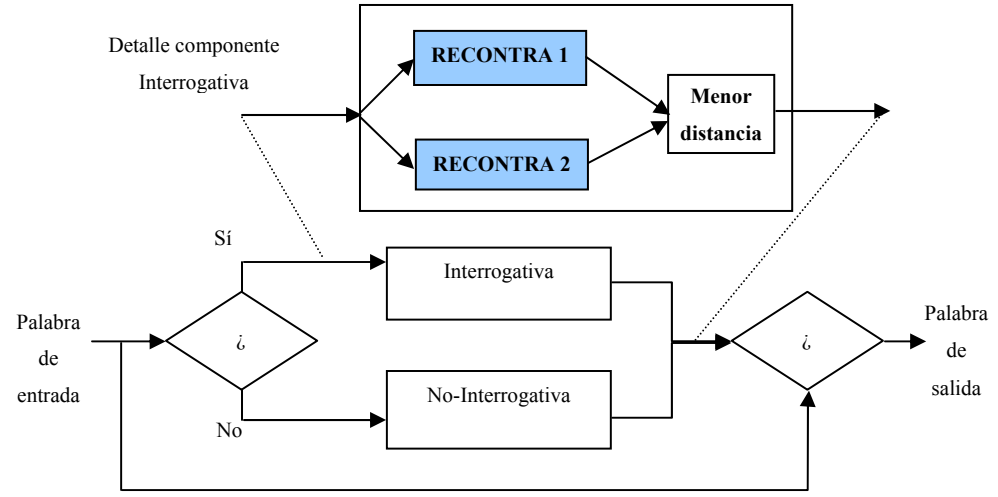

Figura 4.5. Se muestra un sistema de expertos con distancias: una regla sencilla (la presencia de '¿') con *dos modelos RECONTRA* distintos formando cada componente.

4.3.3. ANNs como integradoras

Otra forma es emplear una nueva ANN para integrar las salidas de varios traductores, creando una red claramente modular. Se puede ver como una forma de intentar incorporar un modelo del lenguaje a la salida del sistema de traducción, dado que la nueva red debe combinar las salidas de los traductores para crear una frase correcta.

Evidentemente, cualquier tipo de ANN puede utilizarse como red integradora, por ejemplo MLP o redes de Elman. Para el problema que nos interesa, las redes de Elman por sus capacidades de memoria, parecen más adecuadas. En la Figura 4.6 se puede observar el caso con dos modelos RECONTRA y una red de Elman como integradora.

4.4. Estado del arte

Las primeras experiencias realizadas con el traductor conexionista RECONTRA se centraron en la tarea ALM (Adquisición de un Lenguaje Miniatura) [Feldman, 1990], en la que los vocabularios implicados tenían una talla del entorno de 20 palabras. Ampliaciones del vocabulario de esta tarea demostraron ser también accesibles mediante el modelo RECONTRA [Castaño, 1998].

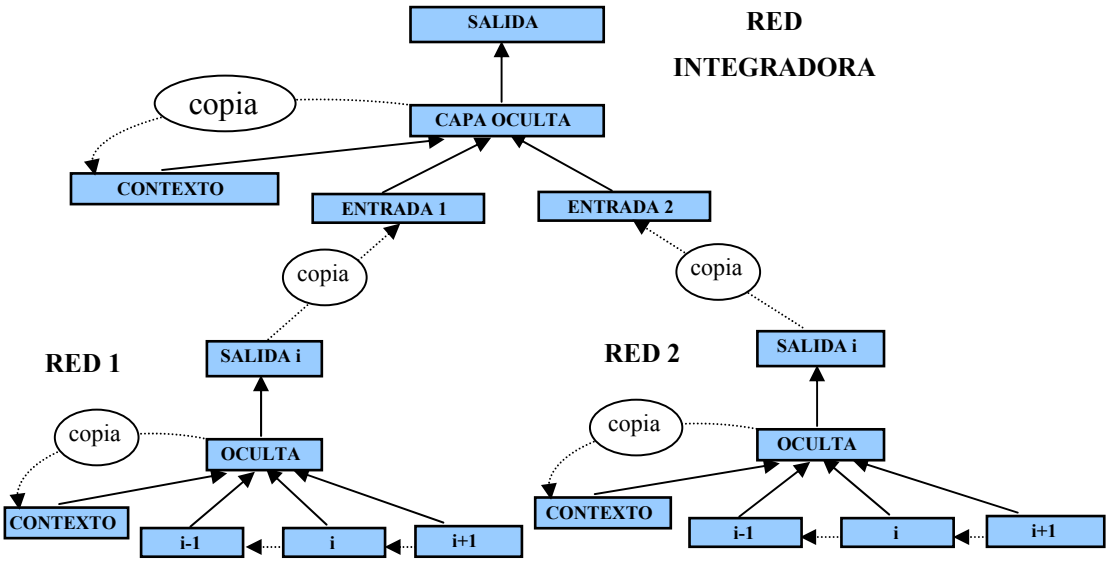

Figura 4.5. Una red de Elman como *integradora de dos modelos RECONTRA distintos* con tres palabras de entrada (*i-1*, *i* e *i+1*)

La tarea ALM fue propuesta inicialmente como una tarea interdisciplinar que afectaba a áreas como Visión, Lenguaje Natural, Inferencia y Aprendizaje. Posteriormente fue adaptada para ser abordada como un problema de AT [Castellanos, 1994]. Para ello se partía de descripciones de escenas en un lenguaje fuente dado, a las que había que asociar las correspondientes descripciones en un lenguaje destino dado; esto es, frases en distintos lenguajes que describen la misma escena, y por tanto, son traducciones la una de la otra.

En las escenas aparecían figuras geométricas con diferentes formas, tamaños y sombreado, situadas dentro de los límites visibles permitidos y con la restricción adicional de que los objetos no podían ocultarse ni solaparse unos con otros. Una primera extensión de la tarea es denominada tarea ALM extendida en [Castaño, 1998].

Además, también consideraba la eliminación e inclusión de objetos dentro de la escena mediante formas reflexivas. El lenguaje en castellano de dicha extensión de la tarea tenía 29 palabras, y el inglés, 25.

Ejemplos de estas frases en español junto con su correspondiente traducción en inglés son:

Español: un cuadrado mediano y claro y un círculo tocan a un círculo claro y un cuadrado mediano

Inglés: a medium light square and a circle touch a light circle and a medium square

Español: se elimina el círculo grande que está encima del cuadrado y del triángulo mediano

Inglés: the large circle which is above the square and the medium triangle is removed

La complejidad de esta tarea radica fundamentalmente en el grado de asincronía existente entre la traducción de las palabras de la frase origen, y la traducción de las palabras de la frase destino. Así por ejemplo, en inglés los adjetivos suelen preceder al

nombre, al contrario que en castellano (*large triangle* vs. *triángulo grande*). Por otro lado, una o varias palabras origen pueden ser traducidas por un número distinto de palabras destino (*por debajo de* vs. *below*). La utilización de verbos reflexivos aumenta enormemente el problema, dado que en castellano aparecen al principio de la frase (*se elimina*), mientras en inglés, la correspondiente forma pasiva se coloca justo al final de la sentencia (*is removed*).

Queda pues patente que la tarea a resolver no resulta trivial; es restringida pero lo suficientemente rica para poder realizar pruebas iniciales con experimentos de AT.

Los resultados con esta tarea y el traductor conexionista RECONTRA adoptando con una "codificación local" para los datos [Castaño, 1997] eran excelentes. En la Tabla 4.2 pueden observarse algunos de los resultados más interesantes obtenidos previamente con codificaciones distribuidas manuales [Castaño, 1998], como *Palabras Bien Traducidas* (PBT) y *Frases Bien Traducidas* (FBT). Se aportaba información sobre la función de una palabra en la frase (nombre, verbo y adjetivo) y sus características (número y género) permitiendo obtener mejores resultados de traducción.

Carácterísticas de las codificaciones			**Porcentajes**	
Número	**Género**	**Tipo de palabra**	**FBT**	**PBT**
No	No	Sí	75.6%	96.2%
No	No	No	21.6%	80.5%
Sí	No	No	26.7%	83.0%
Sí	Sí	No	61.2%	92.7%
Sí	Sí	Sí	79.0%	96.7%

Tabla 4.2. Resultados para la tarea ALM con codificaciones distribuidas manuales de tamaño 10.

La segunda tarea que se ha abordado anteriormente con el traductor conexionista RECONTRA es un subconjunto de la denominada tarea del *Turista* ("Traveller task") [Amengual, 2001], con una talla de vocabularios de unas 150 palabras. Los resultados con codificaciones locales también fueron muy buenos, pero las redes neuronales crecían enormemente y consecuentemente el tiempo requerido para su entrenamiento, debido al aumento de las tallas de los vocabularios.

La razón de esto radica en que las codificaciones locales consumen una gran cantidad de recursos y no son apropiadas para experimentos de AT con vocabularios grandes. La solución que se encontró fue adoptar codificaciones distribuidas que reducían el tamaño de las ANNs, a pesar de que los resultados del traductor empeoraban. Pese a todo, tras experimentar con gran número de combinaciones se lograron codificaciones distribuidas que incorporaban conocimiento que resolvían el problema casi perfectamente, como puede comprobarse en la Tabla 4.3.

|Castellano|/|Inglés|	**FBT**	**PBT**
50/37	98.80%	99.80%
25/25	98.40%	99.72%

Tabla 4.3. Resultados para la subtarea del Turista con codificaciones distribuidas manuales.

De todos los resultados previos se pueden extraer varias conclusiones:

1. Crear una codificación distribuida que sea adecuada para abordar tareas de traducción mediante RECONTRA no es una tarea trivial.
2. En general, los resultados de traducción tienden a empeorar al disminuir el número de unidades de entrada y de salida disponibles para su codificación.
3. En [Castaño, 1998] se logró establecer que codificaciones similares para palabras que aparecían en contextos similares (por ejemplo, los verbos) mejoraban los resultados, y como codificaciones idénticas para palabras equivalentes en el lenguaje de entrada y en el de salida (por ejemplo, *cuadrado* y *square* en la tarea ALM) también lograban mejorar los resultados de traducción.
4. La inclusión de conocimiento en las codificaciones distribuidas de las palabras mejoraba los resultados respecto a los obtenidos con codificaciones distribuidas creadas aleatoriamente.

Estas conclusiones proporcionan un punto de partida sobre las características de las codificaciones y algunas limitaciones, principalmente que no podemos dejar al azar las codificaciones. Al incluir en las codificaciones de las palabras conocimiento sobre su función en la frase y facilitar el reconocimiento de similitudes entre ellas, se facilita la tarea de traducción.

Capítulo 5.

Codificación de los vocabularios

Resumen: En este capítulo se comentan los problemas que se presentan al intentar representar los elementos que procesan los modelos conexionistas, prestando una especial atención a como representar frases y palabras. Así, se hablará de los distintos tipos de codificaciones, para posteriormente presentar modelos conexionistas que se han utilizado con anterioridad para crear representaciones. Además, se presentarán los distintos modelos que se han utilizado en la tesis para crear codificaciones adecuadas para tareas de traducción.

5.1. Introducción

Las representaciones de los elementos que procesa una ANN tienen una importancia crítica en los modelos conexionistas. Estas representaciones (codificaciones) determinan en gran parte la capacidad de dicho sistema. Las representaciones son una fuente de capacidades y debilidades de los modelos conexionistas.

Representaciones cuidadosamente construidas pueden proporcionar propiedades útiles a estos sistemas, como:

- Robustez ante fallos.
- Habilidad para aprender.
- Generalización automática a partir de los ejemplos de entrenamiento.
- Degradación paulatina, etc.

Representaciones inadecuadas pueden hacer que las redes sean incapaces de abordar con éxito el problema.

En esta sección se da una visión de los problemas que se presentan cuando hablamos de la representación de los elementos que procesan los modelos conexionistas, prestando una especial atención, como no, al problema de representar frases y palabras como entradas y salidas de modelos conexionistas.

Si las representaciones no son capaces de representar los objetos implicados en las tareas, el modelo será incapaz de realizar cualquier tipo de cálculo sobre él. Para tareas

que implican procesar el lenguaje natural como son las tareas de traducción abordadas en este trabajo, hay que tener en cuenta que necesitamos ser capaces de representar una frase completa, que no posee un tamaño fijo y en la que el orden de los elementos es importante.

El modelo RECONTRA, al tratarse de una red recurrente y con ventana de entrada es capaz de mantener en memoria parte de la frase, por lo que en principio podríamos representar cada palabra por separado, esto es, la red permite representar implícita y explícitamente dependencias temporales inherentes en el problema de traducción.

No obstante, como los experimentos han demostrado, el modelo funciona mejor cuando las representaciones han sido desarrolladas teniendo en cuenta qué funciones cumplen en la frase.

5.2. Tipos de codificaciones

A continuación se realiza una exploración de los tipos de codificación de los elementos que ha de procesar una ANN en la entrada/salida [Miikkulainen, 1991], [Plate, 1994] y [Castaño, 1998]. Es importante tener presente que, aunque aquí se presentan como dos tipos de representación completamente separados, en gran parte de la bibliografía la codificación local y la codificación distribuida se consideran como parte de un continuo, sin existir una clara frontera de separación entre ellas.

5.2.1. Codificación local

En la **codificación local** cada neurona (unidad) de entrada o de salida representa a una de las palabras del vocabulario del lenguaje origen o destino, respectivamente. Para ello, una neurona está activa (valor 1) y las demás inactivas (valor 0). Así, los valores adoptados en la experimentación para codificar localmente las palabras serían binarios, 0 ó 1.

Por ejemplo, en un vocabulario de 10 palabras, dos posibles palabras de este vocabulario, “círculo” y “pirámide”, podrían representarse como sigue:

círculo	*1 0 0 0 0 0 0 0 0 0*
pirámide	*0 1 0 0 0 0 0 0 0 0*

Las principales ventajas de las codificaciones locales son su representación explicita de los componentes de una tarea, y la facilidad de creación y comprensión para el usuario, así como la facilidad de manipularlas.

Sus problemas son inherentes al tipo de representación utilizada: se produce una gran ineficiencia cuando se trabaja con conjuntos de elementos grandes, puesto que se fuerza a multiplicar el número de conexiones de la red, y en muchos casos estas conexiones son redundantes.

5.2.2. Codificación distribuida

La **codificación distribuida** supone que cada palabra se representa mediante un patrón de activación distribuido en toda o parte de la entrada y/o salida de la red[24]. Esto significa que cada unidad puede intervenir en la representación de más de una palabra. Cuando todas las unidades forman parte de la representación de cada elemento, hablamos de una codificación holográfica.

Siguiendo con el ejemplo anterior, una posible codificación holográfica adoptando cuatro unidades de representación, en las que cada una de las cuatro unidades toma un valor entre 0 y 1, podría ser:

círculo	*0.1*	*0.1*	*0.1*	*0.1*
pirámide	*0.2*	*0.2*	*0.2*	*0.2*

Hay dos tipos principales de codificación distribuida:

La **codificación distribuida simbólica**, en la que cada unidad activada corresponderá a una microcaracterística (siempre de mayor nivel de abstracción que el conjunto de conceptos que queremos codificar) que representa una categoría semántica.

La **codificación distribuida subsimbólica**, en la que cada entrada no es semánticamente interpretable.

Continuando con el ejemplo anterior, una codificación distribuida subsimbólica podría ser:

círculo	*1*	*1*	*1*	*0*	*0*	*0*	*0*
pirámide	*0*	*1*	*1*	*0*	*1*	*0*	*0*

En una codificación simbólica, cada unidad tendría un significado. Así, posibles características serían: objeto material, animal, mamífero, carnívoro, terrestre, masculino, solitario, etc. Cada una de estas características se correspondería con una unidad que tomaría un valor según la palabra descrita posea o no esa característica, en cuyo caso se podrían utilizar simplemente valores binarios (1 ó 0), o según el grado en que la posea, valores reales entre 0 y 1. Un ejemplo de codificaciones binarias simbólicas puede observarse en la Tabla 5.1.

Existe todavía otra forma de ver las codificaciones distribuidas: se denomina representación "tosca" (*coarse representation*), a una codificación en la que cada palabra es representada por varias unidades activas consecutivas o agrupadas. Generalmente, se agrupan según algún tipo de criterio con sentido para el experto humano, pero no tiene porque ser así.

[24] Es importante señalar que en el caso de que esta "parte de la entrada" fuese una única unidad tendríamos una representación local.

En el siguiente ejemplo la agrupación de unidades se realiza según la función de cada una de estas palabras en la frase, si son verbos o sustantivos:

círculo	*1*	*1*	*1*	*0*	*0*	*0*	*0*
pirámide	*0*	*1*	*1*	*0*	*0*	*0*	*0*
está	*0*	*0*	*0*	*0*	*0*	*1*	*1*
tocan	*0*	*0*	*0*	*0*	*1*	*0*	*1*

Una codificación local podría convertirse en una codificación distribuida "tosca" simplemente activando varias unidades contiguas en lugar de una sola.

Palabras	**Características**						
	Objeto	**Animal**	**Mamífero**	**Carnívoro**	**Terrestre**	**Masculino**	**Solitario**
Perro	0	1	1	1	1	1	0
Gato	0	1	1	1	1	1	1

Tabla 5.1. Ejemplo de codificación distribuida simbólica con valores binarios (1 ó 0).

Aunque en la mayor parte de los ejemplos que se han mostrado hasta ahora los valores de las unidades eran 0 ó 1, esto no tiene porqué ser así. Como ocurría en las representaciones holográficas, podemos tener valores reales como activaciones de las unidades. Así, podría haber codificaciones con la descripción codificaciones simbólicas y subsimbólicas[25] holográficas toscas[26].

Las principales ventajas de las representaciones distribuidas son:

- Pueden representar aspectos relevantes de los objetos (sus propiedades o características).
- Permiten que objetos similares puedan tener representaciones similares (características interesantes para el tipo de tareas que deseamos abordar).
- Son redundantes (con lo que son más tolerantes a fallos).
- Hacen un uso (más) efectivo de los recursos de representación (unidades).
- Pueden estar distribuidas en un espacio continuo de vectores.

Por supuesto también presentan una serie de desventajas:

- Cuanto tienen que representar asociaciones arbitrarias o variables (por ejemplo una palabra que puede tener varias acepciones).
- Cuando hay que representar secuencias de tamaño variable[27].
- A la hora de representar el orden de estos elementos[28].

[25] Unas unidades tendrían significado y otras no.

[26] Varias unidades participan en la representación, pero no todas.

[27] Como las frases en una tarea natural, que pueden tener longitudes muy distintas, desde una palabra a cientos de ellas.

[28] En nuestro caso, palabras, dado que según el orden en que se presenten cambia completamente el significado.

Como ya se ha comentado, en nuestra experimentación gran parte estos problemas son asumidos y solucionados por el modelo conexionista utilizado.

5.3. Modelos conexionistas como codificadores

Una de las principales características de las ANNs es su capacidad como clasificador automático [Rojas, 1996]. Por otra parte, en diversos trabajos anteriores se demostró que la primera capa de una Red Neuronal actúa como un detector de características de la entrada [Mirchandani, 1989].

Estas características han sido aprovechadas en distintas variantes de ANNs para obtener representaciones de un conjunto de datos (las entradas/salidas de la red). Y ya en [Ackley, 1985] se presenta el denominado "problema del codificador" (*the Encoder problem*) en el cual se utiliza una ANN para crear representaciones de las entradas/salidas.

Es importante señalar que en gran parte de la investigación realizada, la creación de las codificaciones está integrada dentro de la tarea a abordar. Así, por ejemplo en [Bengio, 2001], en tareas de reconocimiento del habla, tenemos una única red que desarrolla su propia codificación a partir de la entrada, y al mismo tiempo reconoce la palabra y genera la salida correspondiente.

5.3.1. Perceptrón Multicapa

La principal característica del Perceptrón Multicapa (MLP) es su capacidad como clasificador automático [Rojas, 1996]. Un MLP es un tipo de red neuronal no recurrente que se considera uno de los tipos básicos de ANN (Figura 5.1). Un MLP entrenado como clasificador, dado un patrón a la entrada proporcionará como salida la clase a la que cree que pertenece. *El Perceptrón Multicapa se ha utilizado como compresor de información, dadas sus características como detector de características significativas de la entrada*. Visto desde otro punto de vista, se han utilizado MLP para modelar conjuntos de datos discretos de alta dimensionalidad [Bengio, 2003].

Diversas variantes del MLP se han utilizado como codificador y decodificador de representaciones para palabras/frases (dos de las cuales, FGREP y máquinas RAAM se explicarán con más detalle en los puntos siguientes).

En la aproximación adoptada en este trabajo, se intenta modelar el lenguaje fuente y el lenguaje destino mediante MLP para posteriormente entrenar una red de Elman (el modelo RECONTRA) para establecer una correspondencia entre puntos de los dos modelos, como ya se realizó en [Castaño, 1998]. Sin embargo, existen una diferencia fundamental con las experiencias anteriores: la utilización de ventanas de salida en los MLP codificadores.

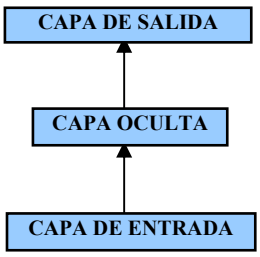

Figura 5.1. Perceptrón Multicapa.

Un aspecto interesante es que las codificaciones de entrada y salida de los MLPs pueden ser tanto codificaciones locales como codificaciones distribuidas. Como en el caso de RECONTRA, el uso de codificaciones distribuidas en lugar de locales permite la reducción del tamaño de la ANN y por tanto el tiempo de entrenamiento.

5.3.2. Maquinas RAAM

Las máquinas RAAM (*Recurrent AutoAsociative Memories*) han sido utilizadas en [Pollack, 1990] para obtener representaciones de secuencias, tanto de letras formando palabras como de palabras formando frases, y en [Chrisman, 1991] para obtener representaciones de frases. Con el modelo RAAM se generaban las codificaciones de las frases a partir de las activaciones que se desarrollaban en la capa oculta para fragmentos progresivamente mayores de ellas.

La arquitectura RAAM permite representar datos estructurados de tamaño variable usando una red de tamaño fijo. La RAAM básica puede codificar estructuras de tipo árbol de profundidad variable mientras el factor de ramificación esté limitado. En teoría no existe ningún límite a la profundidad del árbol o el grado de ramificación. En la práctica depende del tamaño de la red.

Los elementos (letras, palabras) de una lista (palabra, frase) se presentan a la red uno a uno de izquierda a derecha, junto con la representación desarrollada por la red para todos los elementos precedentes. Al presentarse el último elemento, se obtendrá una representación para toda la frase.

Decodificar consiste en ir presentando en la capa oculta la representación. A la salida de la red obtendremos una palabra y la representación del resto de la lista (frase). Este proceso se repite hasta obtener las codificaciones locales de todos los elementos (palabras) de la lista (frase).

La red está entrenada para autoasociar las entradas deseadas utilizando el método de Retropropagación del Error. El elemento de una lista, junto con la codificación del trozo de lista anterior, se coloca a la entrada y la red es entrenada para reproducir el mismo patrón a la salida.

En el proceso la red es forzada a desarrollar representaciones compactas en las unidades ocultas. Las activaciones de las unidades ocultas son extraídas y utilizadas para codificar listas más largas, y el método de Retropropagación del Error es aplicado hasta

que se llega al final de la lista. Según aprende la red, la codificación de la capa oculta cambia y aparece una forma de aprendizaje por objetivo móvil.

Tanto Pollack como Chrisman [Pollack, 1990] y [Chrisman, 1991] obtuvieron buenos resultados de traducción y en varias tareas de tratamiento del lenguajes natural con esta arquitectura, pero siempre en tareas con un léxico muy reducido. Los requerimientos de memoria cuando los vocabularios crecen considerablemente hacen este método de difícil aplicación. Además, cuando las redes crecen demasiado, en algunos casos aparecen problemas de convergencia a un mínimo local aceptable.

5.3.3. FGREP

El método FGREP (*Forming Global Representations with Extended backPropagation*) fue presentado en [Miikkulainen, 1991] se basa en un principio distinto a los anteriores. En lugar de extraer las codificaciones de las representaciones desarrolladas en la capa oculta del MLP, este método se basa en considerar la capa de entrada como una capa de pesos más a entrenar.

Así, consideran que las unidades de entrada son unidades que tienen como función de activación la función identidad. Dado que la derivada de la función identidad es uno, la ecuación del error de una señal para una unidad de entrada consiste en la expresión:

$$\delta_i = \sum_j \delta_j \cdot \omega_{ij} \tag{5.1}$$

donde δ_i es la señal de error para la unidad `i` en la capa de entrada, δ_j es la señal de error para la unidad `j` en la primera capa oculta de la red y ω_{ij} es el peso entre ellas. Así, las representaciones de entrada son modificadas de acuerdo con la señal del error siguiente:

$$\Delta\tau c_i = \eta\delta_i \tag{5.2}$$

donde τc_i es el *i*-ésimo elemento de la representación del objeto `c` y η es el factor de aprendizaje.

Inicialmente a cada palabra se le asigna una codificación al azar, es decir, inicialmente la misma palabra tiene distintas codificaciones en cada frase. Según progresa el entrenamiento, las codificaciones de cada palabra tienden a converger[29]. Con FGREP original, la red recibe como entrada una frase completa y a la salida las palabras se representan con las mismas codificaciones.

En esta aproximación, la creación de las codificaciones y el aprendizaje de la tarea en sí se realizan a la vez. Este método no se puede aplicar directamente a tareas de traducción, dado que la entrada y la salida de la red no son las mismas, y el método no permite generar a la vez la representación de los dos lenguajes.

[29] Excepto cuando una palabra puede tener funciones distintas en distintas frases, en cuyo caso se tiende a desarrollar codificaciones distintas para cada función.

Se puede emplear como base para un traductor en el que se traduzcan frases completas de un lenguaje origen a un lenguaje destino, con un modelo FGREP proporcionando la representación de cada frase. La traducción se puede realizar, por ejemplo, mediante un MLP.

A pesar de que se puede emplear para codificar frases completas, una de las tareas más habituales a las cuales se destina el modelo es la codificación de palabras, pero con el objetivo de localizar palabras con la misma representación y por tanto que pertenezcan a la misma "clase" o "tipo". Dado que la tarea de traducción debe distinguir entre palabras individuales, este comportamiento supone un problema.

5.3.4. Adaptación de left-to-right

Uno de los últimos desarrollos en codificación de datos utilizando ANNs se ha producido al adaptar la idea de redes de dependencia de izquierda a derecha (*left-to-right*) a las ANNs, en concreto a los MLPs, creando una nueva topología [Bengio, 2001].

En esta nueva topología la capa oculta se divide en varios grupos de neuronas, tantas como grupos de neuronas hay a la entrada, y cada grupo de neuronas de la capa de entrada se conecta a la agrupación correspondiente en la capa oculta y a todas las agrupaciones a su derecha (ver Figura 5.2). En diversos experimentos [Bengio, 2001] y [Bengio, 2003] este método ha sido aplicado a tareas de reconocimiento del habla, y ha demostrado su capacidad de desarrollar representaciones adecuadas para gran número de datos.

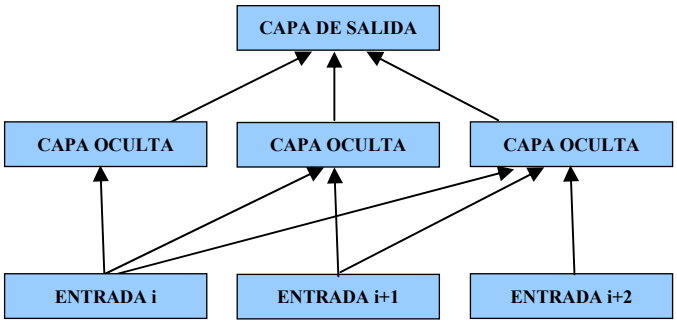

Figura 5.2. Perceptrón Multicapa con *adaptación de left-to-right* con tres agrupaciones.

En esta aproximación, la creación de las codificaciones de entrada y el aprendizaje de la tarea en sí se realizan a la vez, dejando sin resolver el problema de la codificación de salida.

Capítulo 6.

Aspectos experimentales

Resumen: En este capítulo se proporciona toda la información necesaria para comprender la experimentación presentada en este trabajo. En primer lugar se habla de las tareas de traducción abordadas, para posteriormente explicar las características de las ANNs utilizadas, los métodos de entrenamiento, los criterios de éxito empleados e incluso las nomenclaturas de las codificaciones desarrolladas.

6.1. Introducción

En este capítulo se describen las tareas de traducción empleadas en este trabajo, así como las características de la experimentación. Entre otras características tenemos:

- Topologías de las redes empleadas para cada tarea.
- Métodos de selección de los parámetros de entrenamiento.
- Mecanismos de parada de la experimentación.

6.2. Tareas de Traducción

Se han empleado varios tareas de traducción en la realización de este trabajo, participando en la creación de dos ellas, ALM Extendida II y Hansards-SIESTA.

6.2.1. La tarea Adquisición de un Lenguaje Miniatura (ALM)

La tarea ALM (Adquisición de un Lenguaje Miniatura) fue propuesta inicialmente en [Feldman, 1990] como una tarea interdisciplinar que afectaba a áreas como Visión, Lenguaje Natural, Inferencia y Aprendizaje. Posteriormente fue adaptada para ser abordada como un problema de AT [Castellanos, 1994].

Para ello se partía de descripciones de escenas en un lenguaje (natural) origen dado, a las que había que asociar las correspondientes descripciones en un lenguaje (natural) destino dado; esto es, frases en distintos lenguajes que describen la misma escena y que, por tanto, son traducciones la una de la otra.

En las escenas aparecían figuras geométricas con diferentes formas, tamaños y sombreado, situadas dentro de los límites visibles permitidos y con la restricción adicional de que los objetos no podían ocultarse ni solaparse unos con otros. En la Figura 6.1 puede observarse un ejemplo de una de estas escenas.

Una primera extensión de la tarea (denominada tarea ALM extendida), también consideraba la eliminación e inclusión de objetos dentro de la escena. El lenguaje en castellano de dicha extensión de la tarea tenía 29 palabras, y en inglés 25 palabras. A pesar del pequeño tamaño de los vocabularios, los resultados obtenidos previamente demuestran que la tarea no es "obvia". Aunque con RECONTRA se obtuvieron un 99.9% de aciertos a nivel de palabras (Palabras Bien Traducidas, PBT) [Castaño, 1997] y con Asociación de Gramáticas del 99.8%[30], con OSTIA-DR los mejores resultados eran del 85.9% [Castaño, 1997].

Para este trabajo se realizó una nueva extensión de la tarea denominada "Extendida-II", que mantenía el tipo de frases pero ampliaba la talla de los vocabularios hasta 50 palabras en castellano y 39 palabras en inglés, dado el interés de este trabajo en estudiar el comportamiento de RECONTRA ante vocabularios mayores. En el anexo B puede verse el vocabulario completo de esta tarea.

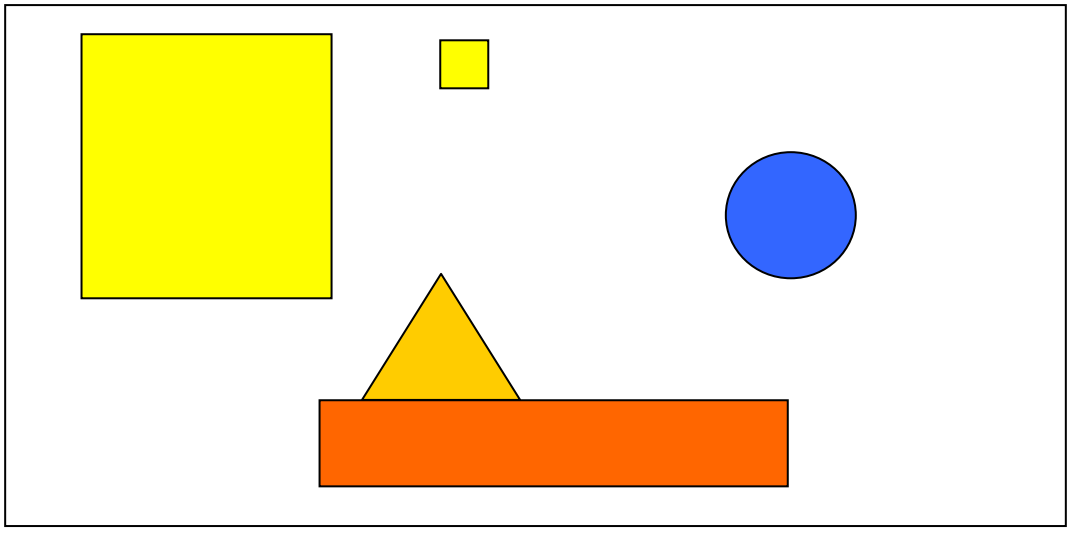

Figura 6.1. Imagen ejemplo de la tarea ALM.

Con el fin de restringir de manera implícita el dominio conceptual de la tarea, Feldman y sus colaboradores crearon una gramática libre de contexto para generar frases que describían en inglés las escenas visuales. Las frases de la tarea constaban de un sujeto, y opcionalmente, de un verbo seguido de un complemento directo. Tanto el sujeto como el complemento directo podían ser uno o dos elementos. Por otro lado, el movimiento de objetos se expresaba mediante formas reflexivas [Castellanos, 1994] de la forma *Se elimina*…en castellano y *…is removed* en inglés.

[30] Utilizando ANNs y poda [Prat, 2001a].

Los pares de frases de traducciones castellano-inglés se generaron a partir de un esquema de traducción dirigido por la sintaxis y gobernado por la gramática libre de contexto para el inglés.

Ejemplos de estas frases en español junto con su correspondiente traducción en inglés pueden observarse en la Tabla 6.1.

	Pares de frases
Español:	un cuadrado mediano y claro y un círculo tocan a un círculo claro y un cuadrado mediano
Inglés:	a medium light square and a circle touch a light circle and a medium square
Español:	se elimina el círculo grande que está encima del cuadrado y del triángulo mediano
Inglés:	the large circle which is above the square and the medium triangle is removed

Tabla 6.1. Ejemplos de pares de frases de la tarea ALM extendida II.

Como ya se ha comentado anteriormente la complejidad de esta tarea radica fundamentalmente en el grado de asincronía existente entre la traducción de las palabras de la frase fuente y la traducción de las palabras de la frase destino.

Los conjuntos de aprendizaje y prueba utilizados para abordar la tarea ALM Extendida II con el traductor RECONTRA constaban respectivamente de 3000 y 2000 pares de frases. Los conjuntos de entrenamiento para los MLPs se generaban a partir de estos dos conjuntos de frases, eliminando todos los contextos repetidos, de forma que para cada palabra sus diferentes contextos no se repetían.

La longitud media de las frases fue de 13.5 palabras para el castellano y 12.4 palabras para el inglés.

6.2.2. La tarea del Turista (EUTRANS-I)

La tarea del Turista (en inglés *Traveller task* or *Tourist task*) es una tarea de Traducción Automática (AT) diseñada en el marco del proyecto EuTrans-I [Amengual, 2001], y financiado por la Comunidad Económica Europea. El objetivo de este proyecto consistía en demostrar la viabilidad de ciertas técnicas basadas en ejemplos (concretamente aquellas centradas en transductores subsecuenciales) para abordar problemas de AT más cercanos a la realidad que los abordados hasta entonces con estas metodologías.

6.2.2.1. La tarea completa

La tarea del Turista aborda situaciones típicas en las que puede encontrarse un turista que visita un país extranjero y cuya lengua desconoce. El escenario contemplado para las posibles traducciones es la comunicación persona a persona en la recepción de un hotel.

En este entorno se contemplan frases en las que:

- El turista informa al recepcionista que ha reservado previamente habitaciones.
- Solicita la reserva de habitaciones.
- Pregunta sobre cómo rellenar la hoja de registro.
- Solicita información sobre las características de las habitaciones.
- Se queja de ellas, etc.

Esta tarea presenta el atractivo de permitir realizar experimentaciones progresivas en las que se va incrementando el tamaño de los vocabularios.

Aunque en el proyecto Eutrans-I se contemplaron tres pares de lenguajes en las traducciones: español-inglés, español-alemán y español-italiano, nosotros consideraremos únicamente el caso español-inglés. Ejemplos de estas frases pueden verse en la Tabla 6.2 y en la Tabla 6.5 como pares de frases de la subtarea sin categorizar.

Los pares de frases para entrenar y probar los traductores se generaron a partir de cuatro esquemas de traducción dirigidos por las sintaxis[31]. Cada esquema generaba pares de frases dentro de un subdominio de las traducciones posibles para la tarea del Turista. Los vocabularios de los lenguajes implicados en las traducciones tenían 686 (castellano) y 513 (inglés) palabras y la longitud media de las frases estaba entre 8 y 13 palabras.

Los corpora adoptados estaban formados por pares de frases texto a texto. Cada par consistía en una frase en español y la correspondiente traducción al inglés. Los traductores de la tarea no categorizada se entrenaron con 10.000 pares de frases y los resultantes modelos entrenados se evaluaron sobre 3.000 pares diferentes.

Pares de frases	
Español:	¿ Qué precio tienen una habitación doble para dos días incluyendo desayuno ?
Inglés:	What is the price for a double room including breakfast for two days ?
Español:	¿ Está incluido el teléfono en la factura ?
Inglés:	Is the phone bill included in the bill ?
Español:	He de marcharme el día veintisiete de febrero a las siete y media de la tarde .
Inglés:	I should leave on February the twenty-seventh at half past seven in the afternoon.
Español:	Por favor, dénos las llaves de la doscientos veintidós .
Inglés:	Please, give us the keys to room Lumber two two two .
Español:	Reservé una habitación tranquila para hoy a nombre de la señorita Rosa Soler .
Inglés:	I booked a quiet room for today for Miss Rosa Soler .

Tabla 6.2. Ejemplos de pares de frases de la tarea del *Turista*.

Con esta tarea se ha realizado una gran experimentación con diversos métodos, aunque no con RECONTRA previamente a este trabajo. En la Tabla 6.3 se muestran algunos de los mejores resultados (como Palabras Bien Traducidas, PBT) para la tarea completa.

[31] Así, la tarea es pseudonatural.

Métodos	PBT
OMEGA con categorías	96.1%
Alignment templates	95.6%
OMEGA sin categorias	93.4%
MGTI	93.2%
Asociación de Gramáticas con modelo Loco_C	93.0%
OSTIA	91.7%
Asociación de Gramáticas con ANN para cada estado y poda	92.5%
Búsqueda casi monótona	89.2%
Búsqueda iterativa basada en DP	86.1%

Tabla 6.3. Mejores resultados de traducción (Palabras Bien Traducidas, PBT) con distintos métodos para la tarea del Turista completa, cogidos de [Prat, 2001a].

Dada la gran diferencia relativa que se produce entre el tamaño de la tarea ALM y el tamaño de la tarea Turista, se utilizaron dos subtareas de menor tamaño como pasos intermedios. Estas subtareas se explican en los dos puntos siguientes. En el anexo B se pueden observar los vocabularios completos de la tarea y en la Tabla 6.6 las principales características de la tarea y las subtareas.

6.2.2.2. La subtarea del Turista categorizada

En la subtarea únicamente se contemplan pares de frases español-inglés generados mediante uno de los cuatro esquemas de traducción dirigidos por la sintaxis considerada para ese par de lenguajes. En ella se incluyen frases en las que el turista notifica su partida, solicita la factura, se queja o pregunta sobre la factura o pide que, ante su marcha, sea trasladado su equipaje.

Existen dos variantes de esta subtarea, la "categorizada" y la "no categorizada". En la subtarea categorizada se consideraron dos categorías correspondientes a horas y fechas genéricas representadas, respectivamente, por $HORA y $FECHA[32]. Al utilizar categorías la complejidad de la tarea se reduce, así como el tamaño del vocabulario. El vocabulario castellano de esta subtarea categorizada tiene 132 palabras (incluyendo las palabras que representan a las categorías), y el vocabulario inglés tiene 82 palabras. Ejemplos de las frases implicadas pueden verse en la Tabla 6.4.

Pares de frases	
Español:	Nos marchamos hoy mismo a $HORA por la noche .
Inglés:	We are leaving today at $HORA in the evening .
Español:	Me voy el día $FECHA a $HORA de la mañana .
Inglés:	I am leaving on $FECHA at $HOUR in the morning .

Tabla 6.4. Ejemplos de pares de frases de la subtarea del *Turista categorizada*.

Los traductores de la tarea categorizada se entrenaron con 5000 pares de frases y los resultantes modelos entrenados se evaluaron sobre otros 1000 pares.

De los 5000 pares de entrenamiento categorizados, 2687 pares eran diferentes. En el corpus de test, 771 de los 1000 pares categorizados eran diferentes. El 54% de los pares

[32] Estas categorías se crearon manualmente.

del conjunto de test categorizado formaban parte de la muestra de entrenamiento categorizada. El número de palabras de las frases categorizadas variaba entre 3 y 13 palabras para el castellano y entre 3 y 12 palabras para el inglés.

6.1.2.3. La subtarea del Turista no categorizada

En la subtarea empleada únicamente se contemplan pares de frases español-inglés generados mediante uno de los cuatro esquemas de traducción dirigidos por la sintaxis considerados para ese par de lenguajes. El vocabulario castellano sin categorizar tiene 178 palabras y el vocabulario inglés, 140 palabras. Ejemplos de las frases implicadas pueden verse en la Tabla 6.5.

Pares de frases	
Español:	He de marcharme el día veintisiete de febrero a las siete y media de la tarde .
Inglés:	I should leave on February the twenty-seventh at half past seven in the afternoon.
Español:	¿ Podemos abonar en efectivo ?
Inglés:	Can we pay in cash ?

Tabla 6.5. Ejemplos de pares de frases de la subtarea del *Turista sin categorizar*.

Los traductores de la tarea no categorizada se entrenaron con 5.000 pares de frases y los resultantes modelos entrenados se evaluaron sobre 1.000 pares diferentes[33]. Estos corpora de entrenamiento y prueba se categorizaron posteriormente y fueron empleados para la tarea categorizada.

De los 5000 pares de entrenamiento no categorizados, 3425 pares eran diferentes. En el corpus de test, 991 de los 1000 pares no categorizados eran diferentes.

Como era de esperar la longitud de las frases en la subtarea no categorizada era mayor que en la categorizada. En castellano la longitud oscilaba entre 3 y 20 palabras, y en inglés, entre 3 y 17. En la Tabla 6.6 pueden observarse algunas de las características de la tarea y las subtareas.

Tarea	Vocabularios	Entrenamiento	Prueba
Subtarea Categorizada	132/82	5000	1000
Subtarea No-Categorizada	178/140	5000	1000
Tarea Completa	686/513	10000	3000

Tabla 6.6. Características de las subtareas de la tarea del Turista.

Los mejores resultados previos para la subtarea no categorizada con RECONTRA[34] son del 98.6% [Castellano, 1999], pero existen métodos que aportan mejores resultados, como Asociación de Gramáticas con 99.6% [Prat, 2001a].

[33] No había solapamiento entre los corpora de aprendizaje y prueba no categorizados, al contrario que en los categorizados.

[34] La tarea más compleja abordada previamente a este trabajo.

6.2.3. La tarea HANSARDS

La denominada tarea Hansards en realidad se trata de una serie de tareas de traducción del lenguaje natural basadas en las actas del parlamento canadiense, los Hansards. Por las características propias del país, estas actas están escritas en dos idiomas, inglés y francés y son una de las mayores (sino la mayor) recopilaciones de textos bilingües existentes en el mundo. Un subconjunto de estas actas está disponible en la web (en http://www.parl.gc.ca) y diversos investigadores las han recogido y utilizado en su experimentación como por ejemplo [Brown, 1990] y las han puesto a disposición de otros investigadores, como en http://www.isi.edu/natural-language/download/handard/.

En esta tesis se han utilizado los conjuntos creados para el proyecto SIESTA[35] (Sistemas Estadísticos para la Traducción Automática). Básicamente, los conjuntos originales de Hansards han sido reducidos para quedarse con pares de frases formadas por las 5000 palabras, en concreto 4968 palabras en francés y 4762 palabras en Inglés, más habituales de las que aparecen en el conjunto original.

Uno de los motivos para seleccionar esta tarea ha sido que el par de lenguajes empleados (Francés-Inglés) es distinto a los anteriores (Castellano-Inglés).

Se ha realizado un preprocesado básico de los conjuntos. Así, los números han sido sustituidos por una etiqueta genérica y el tamaño máximo de las frases seleccionadas es de 50 palabras (la frase más pequeña consta de una sola palabra). Además, se han sustituido por etiquetas los caracteres no alfanuméricos, para facilitar el procesado, como se puede ver en la Tabla 6.7.

[35] El doctorando ha sido uno de los investigadores del proyecto (proyecto Bancaja P1·1B2002-15). Estos conjuntos están disponibles en http://monica.act.uji.es/siesta/Corpus/Hansards.

Caracteres	Etiquetas	Nombre del carácter
.	Pun	Punto
,	Com	Coma
;	PyC	punto y coma
:	DPu	dos puntos
"	DCo	comilla doble
'	SCo	comilla simple (o apóstrofe)
-	Gui	Guión
/	Bar	Barra
(	APa	abre paréntesis
)	CPa	cierra paréntesis
[	ACo	abre corchetes
]	CCo	cierra corchetes
«	ACA	abre comillas angulares
»	CCA	cierra comillas angulares
{	ALl	abre llaves
}	CLl	cierra llaves
¡	AEx	abre exclamación
!	CEx	cierra exclamación
¿	AIn	abre interrogación
?	CIn	cierra interrogación
$	Dol	Dólar
%	Por	Porcentaje
&	And	"ampersand"
@	Arr	Arroba
*	Ast	Asterisco
`	Bac	"backquote"

Tabla 6.7. Sustituciones de caracteres no alfanuméricos por *etiquetas*.

El tamaño medio es de 12.17 palabras para el francés y de 10.59 palabras en inglés para el conjunto de entrenamiento. Para el conjunto de test los valores medios son ligeramente mayores: 12.77 palabras para el francés y 11.06 palabras para el inglés.

Los pares de frases se dividieron en dos partes: un conjunto de entrenamiento de aproximadamente 80000 pares (en concreto 79537, de las cuales 40639 son diferentes) y un conjunto de prueba de 1000 pares. Esta distribución es la seguida dentro del proyecto SIESTA y se mantuvo para hacer posible una comparación directa. Ejemplos de estas frases pueden verse a continuación, en la Tabla 6.8. Entre ellos pueden verse ejemplos de frases formadas por una sola palabra y también de frases que al ser muy formales son prácticamente indistinguibles de un idioma a otro.

Pares de frases	
Francés:	Ajournement
Inglés:	Adjournment
Francés:	Comunication de la residence du goveneur general
Inglés:	Communication from government house
Francés:	Hon . lorna milne :
Inglés:	Hon . lorna milne :
Francés:	projet de loi donnant effet à l ‘ exigence de clarté formulée par la cour suprême du canada Dans son avis sur le renvoi sur la sécession du québec
Inglés:	bill to give effect to the requirement for clarity as set out in the opinion of the supreme court of canada in the quebec secession reference
Francés:	honorables sénateurs , ma question s ‘ adresse au leader du gouvernement au sénat .
Inglés:	honourable senators , my question is for the leader of the government in the senate .
Francés:	l ‘ honorable j . bernard boudreau (leader du gouvernement) :
Inglés:	hon . j . Bernard boudreau (leader of the government) :
Francés:	d ‘ après les renseignements les plus récents que je possède , aucune demande n ‘ a été faite au canada pour qu ‘ il participe au projet , et la question n ‘ est pas activement à l ‘ étude .
Inglés:	the most recent information i have is that no request was made of canada to join in the program and that the issue is not under active consideration .
Francés:	le programme d ‘ échange de pages avec la chambre des comunes
Inglés:	pages exchange program with house of commons

Tabla 6.8. Ejemplos de pares de frases de la tarea *Hansards*.

Es importante señalar la dificultad que supone esta tarea, dado que en ella existen:

- Frases muy largas y muy cortas.
- Palabras que aparecen en muy pocas ocasiones (fundamentalmente nombres propios, de personas o lugares).
- Expresiones y palabras que tienen distintas traducciones.
- Siglas que no se utilizan siempre.
- E incluso malas traducciones o errores tipográficos.

Los mejores resultados dentro del proyecto y para esta tarea han sido del 12.5% para *Word Error Rate*, o 77.5% para PBT y se han obtenido con un sistema de traducción basado en transductores subsecuenciales. Omega, integrando además un nuevo tipo de categorización léxica bilingüe [Prat, 2006] en el informe final del proyecto.

Sin embargo, en resultados no publicados[36], se han obtenido resultados mucho más diversos para esta tarea con Omega. Para empezar, se emplearon dos nuevos sistemas de categorización bilingüe[37], dado que el empleado anteriormente para abordar otras tareas no ofrecía buenos resultados (resultados con “Básico”, “Dejando uno fuera” y “Extensión B” en la Tabla 6.9); sólo cuando se utilizó un nuevo método para determinar cuál era la mejor de las traducciones posibles, *Elige*, se obtuvieron mejores resultados. Incluso en ese caso el método es tremendamente dependiente del número de clases y el resultado de

[36] Comunicación privada el 29 de Diciembre de 2008 con los doctores Barrachina y Vilar.

[37] Para crear las clases se utilizó el paquete GIZA.

12.5% parece ser excepcional. En la Tabla 6.9 se puede observar un resumen de los resultados.

Método	Número de categorías	PBT
Básico	-	52.12%
Básico	2000	45.04%
Básico	4000	47.29%
Básico	8000	48.67%
Extensión B	-	52.12%
Extensión B	2000	45.88%
Extensión B	4000	47.72%
Extensión B	8000	48.04%
Extensión B con Elige	-	52.11%
Extensión B con Elige	2000	75.24%
Extensión B con Elige	4000	87.50%
Extensión B con Elige	8000	48.06%
Dejando uno fuera con Elige	-	52.11%
	2000	44.53%
	4000	46.20%
	6000	46.91%
Dejando uno fuera y Extensión B con Elige	-	52.11%
	2000	75.24%
	4000	75.86%
	6000	73.04%

Tabla 6.9. Resultados de traducción con Omega y el método básico de crear las clases ("Básico"), ampliando este método ("Extensión B") o agrupando con *leaving-one-out* ("Dejando uno fuera"). Además, se puede utilizar o no un nuevo método de búsqueda de la traducción correcta, *Elige*, basado en un modelo del lenguaje.

6.2.3. Resumen

Para facilitar la comparación entre ellos, la Tabla 6.10 se pueden observar las características más importantes de cada tarea[38].

Tarea	|Vocabulario origen|	|Vocabulario destino|	|Frases Entrenamiento|	|Frases Test|
ALM Extendida II	39	50	3000	2000
Turista	686	513	10000	2996
Turista Interrogativa	686	513	4593	1228
Turista No Interrogativa	686	513	5407	1768
MiniTurista Con Categorias	132	82	5000	1000
MiniTurista Sin Categorias	178	140	5000	1000
Hansards	4968	4762	79537	1000

Tabla 6.10. Características de las tareas empleadas en este trabajo.

[38] Se incluyen también los datos de Turista No Interrogativa y Turista Interrogativa, que son divisiones de la tarea Turista completa que se han empleando en experimentación con sistemas expertos.

6.3. Topología de las ANNs

6.3.1. ANNs Codificadores

Las características generales de las ANNs codificadores utilizados en la experimentación son las siguientes:

- Arquitectura Perceptrón multicapa (MLP).
- Tres capas, una de entrada, una oculta, y una de salida, con conexión completa entre las capas.

Además, existen una serie de características que pueden varias considerablemente en los distintos experimentos:

- Salida de la red con distintos formatos, desde una única palabra de entrada a varias palabras, formando una ventana.
- Codificaciones locales o distribuidas a la entrada y la salida de las redes.
- Método de entrenamiento: Retropropagación del Error, SCG o RPROP.
- El número de iteraciones, generalmente 1000 o 3000 con Retropropagación del Error y RPROP, según la tarea. Con SCG, el entrenamiento se detiene al estabilizarse el ECM residual.
- Sin o con poda en la capa oculta. Se han empleado dos métodos distintos de poda, *Skeletonization* o el método de Unidades No Contributivas.

Para cada experimento se realizó una fase de estimación de los parámetros factor de aprendizaje y momentum, tras lo cual se procedió a entrenar la red.

De forma puntual se han realizado experimentos con FGREP. Estas redes son MLPs, y han sido entrenadas con una variación del algoritmo de Retropropagación del Error que modifica directamente las codificaciones de entrada y salida de la red.

También se han realizado unos pocos experimentos con MLPs con una conectividad modificada, siguiendo el método *Left-to-right* entre la capa de entrada y la capa oculta.

Además también se ha experimentado con MLPs con las mismas características (tres capas ocultas, métodos de entrenamiento, ventana de salida…) con ventana de entrada, fundamentalmente para obtener codificaciones de varias palabras a la vez.

Para facilitar la comparación entre ellos, en la Tabla 6.11 se pueden observar las características más habituales de los MLPs para cada tarea.

Tarea	Aprendizaje	|Capa Oculta|	Poda	|Salida|
ALM Extendida II	Retropropagación	6 o 10	No	1, 2, 3, 4, 5, 6 u 8
Turista	Retropropagación, SCG, RPROP, BEP+SCG	varios	Skeletonization o Unidades No Contributivas	4, 6 u 8
MiniTurista Con Categorias	Retropropagación	25	Skeletonization	4, 6 u 8
MiniTurista Sin Categorias	Retropropagación	25	Skeletonization	4, 6 u 8
Hansards	Retropropagación, BEP+SCG	60 o 240	Skeletonization	4

Tabla 6.11. Características de los MLPs empleados en este trabajo.

6.3.2. Modelos RECONTRA

Aunque de forma puntual se han utilizado otras ANNs como traductores, el modelo utilizado principalmente ha sido RECONTRA, una red de Elman con ventana de entrada. Las características generales de las redes fueron:

- Tres capas, una capa de entrada, una oculta y una de salida. Al tratarse de una red de Elman, la salida de la capa de entrada se realimenta sobre la misma capa.
- Ventanas de entrada deslizantes, de forma que en cada momento la red ve un fragmento (varias palabras) de la frase de entrada.

Sin embargo, además de las variaciones propias de cada tarea (tamaño de la ventana de entrada y número de unidades en la capa oculta) y experimento (factor de aprendizaje y momentum), se han desarrollado dos variaciones en el modelo:

- Un modelo RECONTRA con más de una capa oculta: generalmente dos capas ocultas, dado que la primera de ellas se especializará en la creación de representaciones de la entrada.
- Un modelo RECONTRA con una ventana de salida. Se utilizaron distintos tamaños: dos, tres y cuatro palabras.

Además, existen una serie de características que pueden varias considerablemente en los distintos experimentos:

- El método de entrenamiento empleado en general es el de Retropropagación del Error con truncamiento del error para evitar problemas con la recurrencia. Sin embargo también se han empleado una versión de RPROP y de forma puntual, Quickprop.

Para facilitar la comparación entre ellos, la Tabla 6.12 se pueden observar las características más importantes de las redes para cada tarea.

Tarea	|Entrada|	Capas Ocultas	|Capa Oculta|	|Salida|	|Iteraciones|
ALM Extendida II	14	1	140	1	500
Turista	9	1, 2 o 3	450	1, 2, 3 o 4	150
MiniTurista Con Categorias	9	1	140	1	500
MiniTurista Sin Categorias	9	1	140	1	500
Hansards	5 o 9	1 o 2	900	1 o 2	50

Tabla 6.12. Características de los modelos RECONTRA empleados en este trabajo.

6.4. Entrenamiento de las ANNs

El entrenamiento de las ANNs empleadas en este trabajo es similar, tanto para las redes codificadoras como para las traductoras. En este apartado en primer lugar en este apartado se comentarán las características de los métodos de entrenamiento empleados en la experimentación con los dos tipos de redes: codificadoras y traductoras. Posteriormente

se explica como se ha realizado la selección de parámetros, los criterios de parada empleados y los métodos de poda de los MLPs empleados como codificadores.

A continuación puede verse el esquema básico de entrenamiento:

- Selección de codificaciones y características de las redes.
- Estimación de parámetros: factor de aprendizaje y momentum.
- Entrenamiento durante un número determinado de iteraciones o hasta que se alcanza un mínimo en el ECM residual[39].
- Obtención de resultados.

6.4.1. Entrenamiento de las redes Codificadoras

Los codificadores empleados principalmente han sido MLPs de tres capas, a los cuales se presenta de forma estática una palabra de entrada y como salida la misma palabra, habitualmente acompañada de su contexto. Para ser capaz de generar la misma palabra de entrada a la salida, las redes se ven forzadas a desarrollar representaciones en su capa oculta de dichas palabras. Estas representaciones son las que se extraen de los MLPs para emplearlas en la tarea de traducción correspondiente.

Como ya se ha comentado al emplear contexto de salida, las representaciones no sólo representan una palabra sino también todos los contextos vistos en el entrenamiento. Que las representaciones contengan información de los contextos en los cuales aparece una palabra las hace más adecuadas para la tarea de traducción.

Para evitar que palabras que aparecen en contextos similares acaben siendo iguales (o en la práctica muy parecidas), se debe dar más importancia a la palabra que deseamos representar (de entrada). Esto se logra mediante una simple repetición de dicha palabra en la salida. En la Tabla 6.13 pueden observarse ejemplos de los principales formatos de ventana de salida empleados en los codificadores para una palabra de la tarea del Turista.

Formato	Entrada	Salida
x_i	la	la
$x_{i-1}\ x_i\ x_{i+1}$	la	en la habitación
$x_{i-1}\ x_i\ x_i\ x_{i+1}$	la	en la la habitación
$x_{i+1}\ x_i\ x_i\ x_i\ x_i\ x_{i+1}$	la	en la la la la habitación
$x_{i-2}\ x_{i-1}\ x_i\ x_i\ x_i\ x_i\ x_{i+1}\ x_{i+2}$	la	teléfono en la la la la habitación .
$x_{i-1}\ x_i\ x_i\ x_i\ x_i\ x_i\ x_i\ x_{i+1}$	la	en la la la la la la habitación

Tabla 6.13. Ejemplo de entrada y salida para un codificador con la tarea del Turista para la palabra x_i "la", donde i es la posición en una frase.

Los MLPs empleados como codificadores se han entrenado empleando fundamentalmente el método de Retropropagación del Error, ya explicado anteriormente en el apartado 3.5. Además también se han empleado puntualmente los métodos RPROP y SCG, también comentados en el capítulo 3, además de combinaciones de estos métodos.

[39] Algo que no sucedió generalmente.

6.4.2. Entrenamiento del traductor RECONTRA

El procedimiento de presentación de cada frase a traducir es secuencial, es decir, palabra a palabra (aunque como ya se ha comentado, la red realmente observa en un instante determinado no sólo la palabra de entrada en curso, sino las *m* palabras anteriores, y las *n* posteriores, asumiendo una ventana de tamaño *m* a la izquierda y una ventana de tamaño *n* a la derecha). El traductor ha de mostrar a la salida también secuencialmente las sucesivas palabras de la frase traducida. Obviamente, la presentación de cada palabra de la frase de entrada y de cada palabra de la frase destino se realiza a partir de las codificaciones creadas para cada experimento y vocabulario.

En la Tabla 6.14 se puede observar un ejemplo para una ventana con formato *2+1+2* para el par de frases "*Quiero una habitación* ." y "*I want a room* .".

Salida	Entrada i-2	Entrada i-1	Entrada i	Entrada i+1	Entrada i+2
I	@	@	Quiero	una	Habitación
want	@	Quiero	una	habitación	.
a	Quiero	una	habitación	.	@
room	una	habitación	.	@	@
.	habitación	.	@	@	@
FinFrase	.	@	@	@	@
@	@	@	@	@	@
@	@	@	@	@	@

Tabla 6.14. Ejemplo de entrada y salida para un modelo RECONTRA con *ventana de entrada con formato 2+1+2* y el par de frases "*Quiero una habitación . # I want a room* .".

La estimación inicial de los parámetros de entrenamiento mediante la observación del ECM residual es generalmente suficiente para producir una rápida aproximación a un mínimo local adecuado. En algunos casos que esto no se producía. Sin embargo, se exploró la posibilidad de modificar estos parámetros durante el entrenamiento (ver sección 6.5). Se busca reducir el movimiento en torno al mínimo local de la función y por tanto se localice con más precisión e idealmente que los resultados sean mejores.

6.4.3. Otros métodos

De forma puntual se utilizaron otros dos métodos distintos de Retropropagación para el entrenamiento de los modelos RECONTRA.

El primero es una adaptación de RPROP, ya explicado para MLP, en el que el error se trunca al no seguir la recurrencia de la red.

El segundo es *Quickprop*, un método para el entrenamiento de redes neuronales en el cual se utiliza información acerca de la curvatura de la superficie del error. Esto requiere el cálculo de las derivadas de segundo orden de la función del error, lo cual supone un mayor coste computacional. Quickprop asume que la superficie del error es cuadráticamente local e intenta "saltar" en un paso de la posición actual al mínimo de la parábola.

Quickprop calcula las derivadas en la dirección de cada peso. Después de calcular el gradiente con Retropropagación normal, se intenta dar un paso directo al error mínimo mediante el empleo de la fórmula 6.1, que indica la variación que se debe aplicar al peso.

$$\Delta(t+1)\omega_{ij} = \frac{S(t+1)}{S(t)-S(t+1)}\Delta(t)\omega_{ij} \qquad (6.1)$$

dónde ω_{ij} es el peso entre las unidades `i` y `j`, `Δ(t+1)` es cambio en el peso actual (según Retropropagación), `S(t+1)` es la derivada parcial de la función del error para ω_{ij}, y `S(t)` es la última derivada parcial.

Cómo en el caso de Retropropagación aplicado a redes de Elman, se trunca el cálculo del error para evitar caer en ciclos infinitos (recordemos que Elman tiene un grado de recurrencia en la capa oculta).

6.4.4. Criterio de traducciones correctas

Las codificaciones de los vocabularios, ya sean desarrolladas manual o automáticamente, se evalúan indirectamente a través de RECONTRA analizando las tasas de traducción obtenidas.

Para determinar la palabra producida por el traductor se emplean dos métodos según se trabaje con codificaciones locales o con codificaciones distribuidas. Con codificaciones locales, la palabra de salida viene determinada por cual de las salidas de red presenta un valor más elevado. Con codificaciones distribuidas, se calcula la distancia entre la salida producida y las codificaciones del vocabulario, seleccionando la palabra con menor distancia.

Para evaluar la calidad de la traducción producida por la red existen múltiples posibilidades, divididas en métodos automáticos y no automáticos. En general los métodos no automáticos son más informativos para el usuario humano, dado que intentan medir hasta que punto una frase es comprensible para el futuro usuario, aunque no sea completamente correcta o que parte del mensaje original se logra transmitir. Estos métodos son mucho más complicados de utilizar e incluso son subjetivos, puesto que dependen de un evaluador humano, por lo que no se han utilizado en este trabajo.

Entre los métodos automáticos se decidió utilizar dos clásicos para medir la corrección de las traducciones:

1. Una frase se considera bien traducida (FBT) si las palabras de salida proporcionadas por la red (entrenada) coinciden exactamente con las palabras esperadas de la frase destino. Es importante señalar que no se explora la posibilidad de que la frase sea o no correcta en el idioma destino, que porcentaje de la frase es incorrecta o que exista un error en la palabra pero no en la función en la frase de la palabra traducida (por ejemplo traducir '*perro*' por '*gato*' o '*Juan*' por '*Andrés*').
2. A nivel de palabra el porcentaje (real) de palabras acertadas (bien traducidas) (PBT) se obtiene comparando cada frase traducida por la red con su traducción

esperada y utilizando un algoritmo de edición basado en una estrategia de Programación Dinámica [Marzal, 1993], que calcula la siguiente expresión:

$$\frac{N_a}{N_s + N_i + N_b + N_a} \cdot 100 \qquad (6.2)$$

dónde N_a es el número de palabras acertadas en las traducciones dadas por RECONTRA y N_s, N_i y N_b indican el número mínimo de palabras que hace falta sustituir, insertar y borrar, respectivamente, para obtener las traducciones esperadas a partir de las emitidas por la red.

Para analizar la importancia estadística de los resultados de todos los métodos empleados en la comparativa, se ha realizado un test Friedman [Friedman, 1937]. Esta es una técnica no paramétrica para medir la importancia de la diferencia estadística de varios algoritmos que proporcionan resultados para el mismo problema, utilizando un ranquing de los resultados obtenidos por los algoritmos a comparar.

Para cada subconjunto de características se ordenan en el ranquing de uno al número de métodos. El método con los mejores resultados tendrá la posición 1, el siguiente la 2, etc. En el caso de que dos o más métodos tengan los mismos resultados, se les asigna una media del ranquing.

El estimador de Friedman sigue una distribución de Fisher que permite analizar la importancia estadística de los resultados. El estimador se expresa como:

$$X_r^2 = \frac{12}{Nk(k+1)} \sum_{j=1}^{k} R_j^2 - 3N(k+1) \qquad (6.3)$$

Donde k es el número de resultados (columnas), N es el número de métodos (filas), y R_j es el número en el orden de los resultados. A los valores obtenidos se les aplica un grado de confianza del 95%.

6.4.5. Métodos para detener el entrenamiento

De todos los métodos posibles para decidir cuándo se ha de detener el proceso de aprendizaje (comentados en el apartado 3.5.6) se seleccionó uno de los más sencillos: tras un número predeterminado de iteraciones del conjunto completo o cuando se alcanzaba un mínimo en el ECM residual, se detenía el entrenamiento y se procedió a obtener los resultados sobre el conjunto de prueba. Para cada tarea se seleccionó este número de iteraciones de forma independiente, tras examinar la evolución del error de un experimento inicial.

6.5. Selección y evolución de parámetros de entrenamiento

Los algoritmos de entrenamiento de ANNs garantizan la convergencia a un mínimo local de la función del error. Sin embargo no garantizan que el mínimo que encuentran sea el mínimo global o incluso que se encuentre entre los mejores de la función.

Como ya se comenta en [Sutton, 1986] existen dos problemas en los métodos de entrenamiento basados en descenso del gradiente.

El primero es que si la superficie del error no es uniforme, las direcciones con una inclinación más abrupta serán las que se seguirán, provocando que se quede 'atrapado' en un mínimo local.

El segundo es que los patrones de entrenamiento pueden interferir entre ellos, puesto que las unidades que han resultado más útiles hasta este momento son las más susceptibles a ser modificadas, con la consiguiente perdida de resultados.

Aparte de las características propias de cada problema hay varios factores que pueden influir en la 'calidad' del mínimo local encontrado y por tanto de los resultados finales: la inicialización y los parámetros de entrenamiento utilizados. El factor de aprendizaje y el momentum determinan en las redes neuronales entrenadas con el método de RetroPropagación la rapidez con la cual la red evoluciona hacia el mínimo de la función del error.

Pero el algoritmo de RetroPropagación presenta problemas cuando se acerca a un mínimo local, especialmente cuando se utiliza un (relativo) gran factor de aprendizaje, como permite hacer la búsqueda en el espacio bidimensional. Esto puede provocar que las codificaciones extraídas de los MLP no sean adecuadas para la tarea de traducción: cuando la función del error es demasiado abrupta, la red puede oscilar en torno a él sin acabar de localizarlo. En RECONTRA, los resultados de traducción oscilan sin disminuir, por mucho de tiempo que se dedique al entrenamiento.

En este trabajo no hemos realizado ningún experimento serio con distintos métodos de inicializar las redes, pero como ya se ha comentado anteriormente, sí que se han utilizado dos métodos para trabajar con parámetros: selección y evolución.

6.5.1. Selección de parámetros.

Para cada experimento con el algoritmo de *Retropropagación del Error* se realizó la estimación de dos de los parámetros de entrenamiento de la red codificadora (el factor de aprendizaje, α, y el momentum, η).

El procedimiento empleado consistió en entrenar la ANN correspondiente (fuese un MLP o el modelo RECONTRA) hasta 10 presentaciones del conjunto de entrenamiento con distintas combinaciones de factor de aprendizaje y momentum tomadas de la lista 0.1, 0.3, 0.5, 0.7 y 0.9, valores comprendidos dentro del rango de valores posibles para estos parámetros [0,1]. Se seleccionan los parámetros que en este punto del entrenamiento producen un ECM residual menor (recordemos que es el ECM sobre el conjunto de entrenamiento). Así, para un experimento típico el factor de aprendizaje α, podría tomar el valor 0.1 y el momentum η, el valor 0.7.

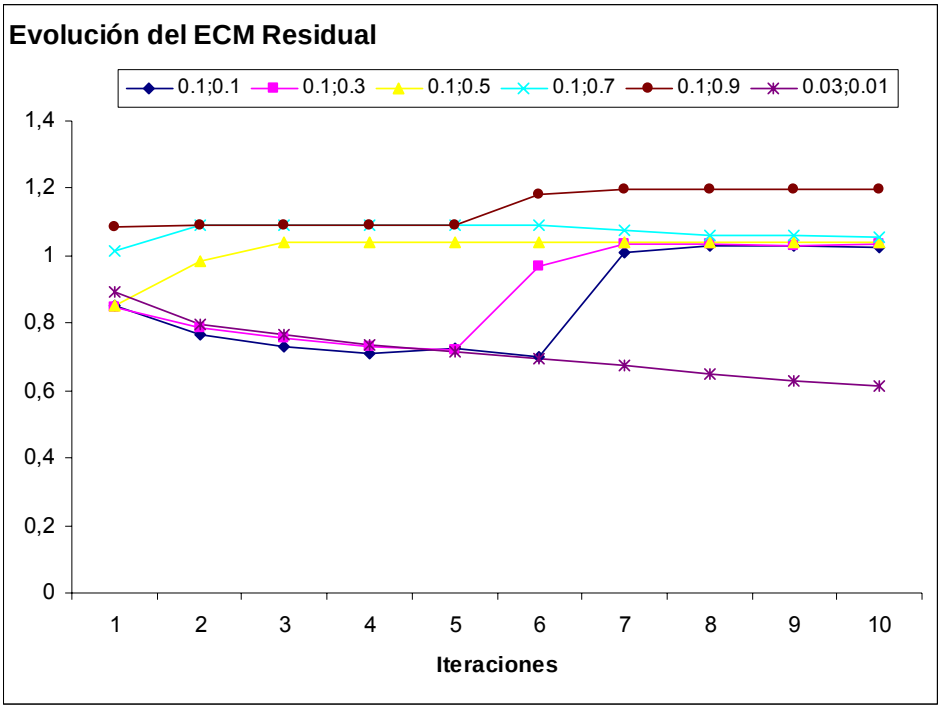

Figura 6.2. Evolución del ECM residual para la tarea del Turista y una red de *Elman* con ventana de entrada de tamaño 9 y 450 unidades en la capa oculta durante el proceso de estimación de parámetros de entrenamiento (Learning Rate y Momentum) (con codificaciones *T_VI_pmv4_r2_pS_t156_115_bp3000*).

Experimentalmente se comprobó que en ciertos casos la evolución del ECM residual en redes de Elman mostraba un aumento tras cierto número de iteraciones de entrenamiento, y los correspondientes resultados de traducción en ese punto eran próximos a cero. Claramente la red había iniciado su aprendizaje y luego había 'olvidado' lo aprendido (este fenómenos ya habían sido observados por [Sutton, 1996]).

Este problema se presenta con relativa frecuencia cuando los parámetros de entrenamiento son demasiado grandes, así que en algunos casos se repitió el proceso de estimación con valores menores: 0.01, 0.03, 0.05, 0.07 y 0.09 y en la mayoría de los casos se resolvió el problema. En algunos casos concretos[40] aún se tuvo que repetir el proceso de estimación con valores menores: 0.001, 0.003, 0.005, 0.007 y 0.009. Por ejemplo, los parámetros de un experimento con Hansards fueron α=0.03 y η=0.01. Se puede ver un ejemplo gráfico de este fenómeno en la figura 6.2, donde esta combinación de parámetros son los únicos que no producen un empeoramiento en el ECM residual a partir de cierto punto del entrenamiento.

Señalar que por la forma de realizar la estimación no se permiten combinaciones de valores de los parámetros de distintos órdenes de magnitud, del tipo α=0.3 y η=0.01, por ejemplo.

[40] Fundamentalmente con la tarea Hansards, como ya veremos en la presentación de los resultados experimentales.

6.5.2. Evolución de parámetros

Como ya se ha mencionado el método utilizado para seleccionar los parámetros de entrenamiento es un método de prueba y error al inicio del entrenamiento. Sin embargo, los parámetros que inicialmente son adecuados (son los que se acercan más rápidamente al mínimo), pueden no serlo según avanza el entrenamiento. Si estos valores son demasiado grandes, la red es incapaz de encontrar un mínimo local.

Una forma de mejorar los resultados de las redes neuronales es cambiar estos parámetros, de forma que su valor vaya disminuyendo al acercarse a un mínimo. El problema es que a priori no se puede conocer este mínimo o cuando nos acercamos a él. Existen muchas posibilidades para determinar como se modifican los parámetros y varias se han explorado en la tesis.

Para modificar el valor del factor de aprendizaje (α) y el momentum (η) durante el proceso de entrenamiento se han probado varios métodos clásicos [Darken, 1991][41].

El primero es el más evidente: en puntos fijos del entrenamiento, se reducía el valor de los parámetros hasta un décimo de su valor inicial.

Sin embargo, incluso desde el punto de vista conceptual, este método deja mucho que desear, por lo que se utilizaron dos métodos que tenían en cuenta el número de iteraciones de entrenamiento en la cual se encontraba la red. Esto se basa en las siguientes dos fórmulas distintas del tipo "busca-luego-converge":

$$\eta(t) = \frac{\eta(t-1)}{1+\frac{t}{\lambda}} \tag{6.4}$$

y

$$\eta(t) = \frac{\eta(0)}{1+\frac{t}{\lambda}} \tag{6.5}$$

La diferencia entre las expresiones (6.14) y (6.15) es que la primera depende del valor previo del factor de aprendizaje, y tiende a reducirlo más rápidamente y en algunos casos se detiene en un mal mínimo local.

Reducir siempre los parámetros y solo tener en cuenta las iteraciones de entrenamiento no es una buena idea en todos los casos. La evolución del ECM no sigue completamente el número de iteraciones y en algunos casos el ECM puede incrementarse con más entrenamiento, y reducir el factor de aprendizaje no ayuda a que la red salga de esta zona, sino al contrario. Otro problema es la aparición de un nuevo parámetro λ.

Una simple solución es tomar en cuenta si el ECM residual se ha reducido o incrementado tras la última iteración para reducir o aumentar los parámetros por un factor fijo. El método [Silva, 1990] utilizado en los experimentos es el siguiente:

[41] Aunque en el original se aplicaban sólo sobre el factor de aprendizaje, dado que la versión del algoritmo de retropropagación del error no utilizaba el parámetro de momentum.

Si $(ECM_{t-1} - ECM_{t-2}) > (ECM_{t-2} - ECM_{t-3})$

$$\eta(t) = \eta(t-1) \cdot 0.96$$
$$\sigma(t) = \sigma(t-1) \cdot 0.96 \qquad (6.6)$$

sino

$$\eta(t) = \eta(t-1) \cdot 1.04$$
$$\sigma(t) = \sigma(t-1) \cdot 1.04$$

Siendo *t* la iteración actual. Como es evidente hay dos nuevos parámetros a los cuales se han asignado los valores 0.96 y 1.04.

Se han empleado dos varias variantes de este método, según esta comparación se realiza cada iteración o cada 10 ellas (para evitar un posible fenómeno de oscilación en los parámetros).

6.6. Nomenclatura de las codificaciones

La nomenclatura utilizada para nombrar las codificaciones tiene dos objetivos:

- La identificación de las codificaciones de forma unívoca.
- La descripción de estas.

Esta nomenclatura se explica en detalle en los capítulos de resultados experimentales según es necesaria su aplicación. En este apartado se realiza una introducción general.

En letras mayúsculas aparece la inicial o iniciales de la tarea abordada:

- ALM: A.
- Turista: T
- MiniTurista con Categorías: MinTcC
- MiniTurista sin Categorías: MinTsC
- Hansards: H.

A continuación un número latino en mayúsculas y la letra `t`, de tamaño, seguida por los tamaños de las codificaciones origen y destino, todo ello separado por guiones.

Así, *A_I_t10_10* sería la codificación I de la tarea ALM, con codificaciones de entrada y salida de tamaño 10 y *T_I_t30_30* la codificación I de la tarea del Turista con tamaños 30. De esta forma podemos identificar todo tipo de codificaciones distribuidas al azar y conocer sus características más importantes: la tarea a la que pertenecen y el tamaño (número de unidades) empleado.

Cuando queremos identificar codificaciones generadas por algún método automático, información acerca del método y las características empleadas se incluyen en el nombre de la codificación.

Para dar nombre a las codificaciones extraídas de MLP se extiende esta nomenclatura de forma que lo primero que tenemos sea la inicial de la tarea abordada, seguida por el número latino asignado a la codificación. Inmediatamente se indica que se ha utilizado un MLP mediante las letras `pm`, el tamaño de la ventana de salida empleada (`v`) y el número de repeticiones de la palabra de entrada en dicha ventana de salida (`r`). A continuación,

como anteriormente, se indica el tamaño de las codificaciones mediante la letra `t` y sus tamaños. Lo que falta es indicar el método de entrenamiento del MLP y el número de iteraciones de entrenamiento para cada lenguaje.

Así, una codificación extraída de un MLP con ventana de salida de tamaño 8 y 4 repeticiones de la palabra de entrada a a la salida y codificaciones iniciales *T_I* con tamaño en la capa oculta 30, las redes entrenadas con Retropropagación durante 3000 iteraciones tomaría el nombre *T_I_pmv8_r4_t30_30_bp3000*.

Cuando se aplica un método de poda se indica antes del tamaño mediante una *p*, seguida por las iniciales del método empleado (*S* o *UNC*). Así, otro ejemplo podría ser *T_I_pmv8_r4_pS_t23_34_bp3000*, para codificaciones extraídas de MLP con ventana de salida de tamaño 8 y 4 repeticiones de la palabra de entrada a a la salida y codificaciones iniciales *T_I* con poda mediante *Skeletonization* que dio los tamaños 23 y 34 para Castellano e Inglés, entrenadas las redes con Retropropagación durante 3000 iteraciones.

Cuando se han empleado otros métodos de entrenamiento se han utilizado para entrenar los MLP y durante cuantas iteraciones. Así, para indicar que en el entrenamiento del MLP se ha utilizado RPROP y SCG se utilizan las letras *RPROP* y *SCG* seguidas por el número de iteraciones. Para indicar que se han empleado las fórmulas 6.4, 6.5 o 6.6 se añaden tras el número de iteraciones *f1*, *f2* o *f3*, respectivamente.

La nomenclatura utilizada para los nombres de las codificaciones generadas con este método amplia la ya existente. Así, en primer lugar tenemos el nombre completo de la codificación utilizada para entrenar el MLP, seguida por la letra *R* mayúscula y las características propias de esta red (es decir, no indicamos la ventana de salida ni el tamaño de las codificaciones resultantes), fundamentalmente el método de entrenamiento y el número de iteraciones.

Un ejemplo podría ser la codificación *T_L_pmv4_r2_pS_t30_30_bp3000_R_bp3000*, que para reducir el tamaño normalmente se expresará simplemente como *L_pmv4_r2_pS_t30_30_bp_R_bp3000*. Dado que si empleamos realimentación para varios MLP podríamos acabar con codificaciones de nombres muy largos (incluso con la versión reducida del nombre, *L_pmv4_r2_pS_t30_30_bp_R_bp_R_bp_R_bp3000*, es muy largo), aprovechamos que en la mayoría de los casos el método de entrenamiento se repite en los siguientes experimentos para indicar simplemente el nivel de realimentación cón un número, así, para las codificaciones del ejemplo anterior, tendríamos *L_pmv4_r2_pS_t30_30_bp_R3_bp3000*.

En los experimentos en los que en lugar de MLP “clásicos” se ha utilizado FGREP, simplemente se sustituye *pm* por *fgrep*, y se añade el dato del tamaño de la capa oculta de la red empleada.

En algunos experimentos concretos no se ha empleado el conjunto normal para entrenar los MLPs. Esto se indica añadiendo el nombre de dicho conjunto al final del nombre de la codificación. Un ejemplo podría ser el par de codificaciones *H_IV_pmv4_r2_t240_240_bp3000_contexto1*. Como su nomenclatura indica estas codificaciones son para la tarea Hansards, y se extraen de MLPs con ventana de tamaño

4, codificaciones iniciales *H_IV_t240_240*, 240 unidades en la capa oculta y entrenadas con Retropropagación del Error durante 3000 iteraciones con el conjunto de entrenamiento *contexto1*.

En la Tabla 8.22 pueden verse los tamaños de los conjuntos completos así como para un único contexto por palabra "contexto1" y para tres contextos por palabra "contexto3". Estos contextos, en principio, se eligieron al azar sin ningún criterio de selección. Sin embargo, también se realizaron experimentos con un conjunto, "contextoM1" creado con el contexto que aparecía más veces en el conjunto de entrenamiento para cada palabra. En teoría esto debería facilitar la tarea del traductor, al ser este contexto más representativo del uso de la palabra en el conjunto de entrenamiento.

6.7. Ensembles de RECONTRA

En el punto 4.3 ya se ha comentado que en la experimentación presentada en este trabajo se utilizaron distintos tipos de e*nsembles*. En el aspecto práctico de cómo entrenar las arquitecturas conexionistas que forman los componentes del ensemble, los algoritmos de entrenamiento y el procedimiento fueron los mismos que en los otros casos: tras una estimación de parámetros (factor de aprendizaje y momentum) basados en el ECM residual, se prosiguió el entrenamiento hasta un número predefinido de iteraciones. En ese momento se obtuvieron los resultados de traducción para cada módulo sobre el conjunto de test correspondiente.

Con ensembles se añade un paso, dado que en este momento se deben combinar las salidas o simplemente decidir cuál es la correcta. Aquí se presentan tres casos, ya comentados en el punto 4.3:

1. Mezcla de **expertos**: aplicamos la regla de decisión predefinida (presencia de '¿' a la entrada del sistema).
2. Criterios de **selección:** basados en la distancia a la palabra del vocabulario más próxima. Aplicamos la formula adecuada según el método de fusión deseado (si deseamos seleccionar palabras individuales o frases completas, etc…).
3. Puramente **modular**: se utilizan las salidas de los componentes como entradas a la red integradora, que es la que nos produce la palabra de salida del sistema.

El único de estos tres casos en el que es necesario aclarar algún aspecto es el tercero, el puramente modular. Como ya se ha visto, conceptualmente es sencillo, dado que las salidas de los traductores individuales son la entrada de una nueva red que produce la salida correcta. Así, esta red puede verse como que crea un modelo del lenguaje de salida que se utiliza para decidir cual de las salidas producidas por las redes componentes es la correcta.

En la práctica primero se entrenaron completamente los componentes (por separado) y luego se entrenó la red integradora (siguiendo el proceso habitual) con las salidas producidas por la redes con el conjunto de entrenamiento.

Capítulo 7.

Experimentación con codificaciones manuales

Resumen: En este capítulo se muestra la experimentación sobre codificaciones manuales que se realiza con RECONTRA. En primer lugar se concretan las características de los experimentos realizados, para posteriormente presentar los resultados con codificaciones al azar para todas las tareas, que servirán de base para la comparación de resultados. A continuación, se presentan los experimentos con codificaciones en las que se incorpora conocimiento humano mostrando algunas conclusiones respecto al uso de las mismas.

7.1. Características de los experimentos

Este capítulo se presenta la experimentación realizada con RECONTRA utilizando codificaciones manuales, creadas al azar o por un ser humano. La razón de este análisis es obtener unos resultados básicos a partir de los cuales poder compararse con los resultados obtenidos con codificaciones automáticas. Además, estos experimentos se han utilizado para darnos información de las características de RECONTRA más adecuadas para cada tarea.

La nomenclatura utilizada para nombrar las codificaciones es muy sencilla: en letras mayúsculas se muestra la inicial de la tarea abordada (ALM, *A*, Mini Turista sin Categorías, *MinTsC*, Mini Turista con Categorías, *MinTcC*, Turista, *T* y Hansards, *H*), un número latino en mayúsculas y la letra *t* de tamaño, seguida por los tamaños de las codificaciones origen y destino (el número de bits empleados), todo ello separado por guiones. Así, *A_I_t10_10* sería la codificación *I* de la tarea ALM, con codificaciones de tamaño 10 y *T_I_t30_30* la codificación *I* de la tarea del Turista con tamaño 30. Cuando no existe posibilidad de confusión, las codificaciones son nombradas únicamente por su número latino. En el capítulo 6 se recoge la nomenclatura completa utilizada en este trabajo.

En este capítulo se han empleado modelos RECONTRA con distinto número de unidades en la capa oculta y número de palabras en la capa de entrada, según las necesidades de la tarea abordada en cada caso. Dado que no se dispone de un método

seguro para determinar las características ideales de RECONTRA para cada tarea, se realizaron distintas pruebas con distintos modelos. En general, las tareas más grandes como Hansards tienden a necesitar un mayor número de unidades en la capa oculta. Además, las tareas con mayor asincronía y por tanto más complejas, necesitan mayor número de palabras en la ventana de entrada.

Para cada modelo se ha realizado una búsqueda de los parámetros de la red (factor de aprendizaje y momentum) que proporcionan el menor ECM residual, empleando el método explicado en el capítulo 6.

Por último señalar que los resultados presentados en el capítulo han dado lugar a las siguientes publicaciones: [Casañ, 1999] y [Casañ, 2003c].

7.2. Codificaciones al azar

En este apartado se presentan una serie de experimentos con codificaciones creadas al azar. Recordemos que este tipo de codificaciones emplean valores 0 y 1 sin seguir ningún tipo de regla ni criterio, como se puede ver en el ejemplo siguiente para varias palabras de la tarea ALM.

rojo	*1 1 0 0 1 0 1*
verde	*0 1 1 0 1 0 1*
cuadrado	*1 1 0 0 1 1 0*

7.2.1. La tarea ALM

Las codificaciones locales de los vocabularios han demostrado experimentalmente [Castaño, 1999] ser capaces de proporcionar a RECONTRA muy buenos resultados de traducción, cercanos al 100% de frases bien traducidas (FBT) con la tarea ALM básica y extendida.

Tomando como base las características de las redes encontrados en [Castaño, 1999] se utiliza el modelo RECONTRA con ventana de entrada de tamaño 14 y formato 6+1+7 (las 6 palabras precedentes, la palabra actual y las 7 palabras siguientes), 140 unidades en la capa oculta y 500 iteraciones del conjunto de entrenamiento.

En la Tabla 7.1 se observan los resultados de traducción obtenidos sobre el conjunto de test con codificaciones locales y distribuidas binarias “al azar”, en las cuales no se ha introducido ningún tipo de conocimiento acerca de la tarea.

Se utilizó un tamaño de 10 unidades para las codificaciones. Puede verse como al utilizar codificaciones distribuidas al azar los resultados obtenidos en general son peores, como se muestra en la tabla.

Nombre	FBT	PBT
A_Local	98.8%	99.9%
A_I_t10_10	21.6%	80.5%

Tabla 7.1. Porcentajes de frases y palabras bien traducidas (FBT y PBT) para la tarea ALM extendida utilizando codificaciones *locales* y codificaciones *binarias al azar* con 10 unidades.

7.2.2. La tarea del Mini Turista sin Categorías

Para la experimentación realizada con la tarea del Mini Turista sin categorías, se utilizó RECONTRA con ventana de entrada de tamaño 6 palabras y 140 unidades en la capa oculta. El tamaño de las codificaciones al azar es de 25 unidades. En comparación con la tarea ALM, la tarea Turista en general necesita una ventana de entrada de menor tamaño, dado que su grado de asincronía es menor. En la Tabla 7.2 pueden observarse los resultados obtenidos.

Nombre	FBT	PBT
MinTsC_I_t25_25	33.7%	89.16%
MinTsC_II_t61_52	39.4%	91.15%

Tabla 7.2. Porcentajes de frases y palabras bien traducidas (FBT y PBT) para la tarea del Mini Turista sin Categorías utilizando codificaciones binarias al azar con 25 unidades y RECONTRA con ventana de entrada de tamaño 6 y 140 unidades en la capa oculta tras 150 iteraciones.

7.2.3. La tarea del Turista

Para la experimentación realizada con la tarea del Turista completa, se utilizó RECONTRA con ventana de entrada de tamaño 9 palabras y 450 unidades en la capa oculta. En comparación con la tarea ALM, la tarea Turista necesita sólo una ventana de entrada de tamaño 9 palabras, dado que su grado de asincronia es menor. Recordemos que en ALM aparecen frases pasivas que complican la tarea de traducción.

El proceso de entrenamiento también sufrió una modificación respecto a la tarea ALM, deteniéndose el entrenamiento tras 150 iteraciones y no 500 iteraciones. La intención es evitar el sobreentrenamiento y reducir el tiempo dedicado a cada experimento individual. Por regla general en 150 iteraciones las redes para esta tarea ya han alcanzado su resultado máximo o están muy cerca de él (ver Figura 7.1).

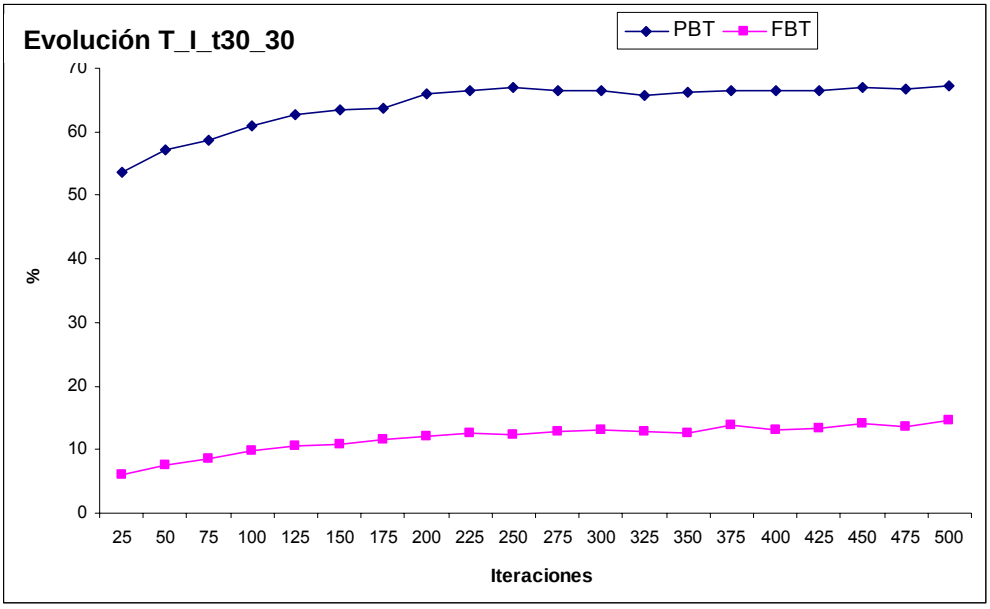

Figura 7.1. Evolución del porcentaje de PBT hasta 500 iteraciones para el experimento con codificaciones *T_I_t30_30* de la Tabla 7.3.

Se han seleccionado tres pares de tamaños distintos para las codificaciones, 30 unidades, 60 unidades y 172 (castellano) y 129 (inglés) unidades (aproximadamente una cuarta parte del tamaño de los vocabularios). En la Tabla 7.3 podemos observar los resultados obtenidos tras 150 presentaciones del conjunto de entrenamiento con estas codificaciones.

Hay todavía una distinción más entre las codificaciones: si estas son binarias o reales. En experimentos previos en tareas de tratamiento del lenguaje natural[42] se sostiene que con codificaciones reales se disminuye la inestabilidad de las redes y se obtiene mejores resultados. Así, se decidió realizar algunos experimentos con ambos tipos de codificaciones para tamaño 30 de entrada y 30 de salida, creando varias codificaciones distintas para cada tipo. Como se puede ver en la Tabla 7.3, los resultados obtenidos fueron similares.

Al utilizar codificaciones binarias de tamaños 172/129 se obtienen resultados muy pobres de traducción. Tras varias iteraciones de entrenamiento en las cuales el ECM residual se reduce, este vuelve a aumentar exageradamente y la red se queda atascada en estos valores. En la Tabla 7.3 se muestran los mejores resultados obtenidos con parámetros de entrenamiento buscados dentro del rango [0.01, 0.1].

Nombre	Codificación	FBT	PBT
T_I_t30_30	Sí	11.0%	63.4%
T_II_t30_30	Sí	12.2%	67.5%
T_III_t30_30	Sí	11.1%	67.4%
T_IV_t60_60	Sí	6.4%	66.2%
T_V_t172_129	Sí	0.0%	26.3%
T_VI_t172_129	Sí	0.0%	26.7%
T_VII_172_129	Sí	0.0%	26.1%
T_VIII_t30_30	No	12.0%	63.6%
T_IX_t30_30	No	12.7%	64.1%

Tabla 7.3. Porcentajes de FBT y PBT tras 150 iteraciones para la tarea del Turista utilizando codificaciones al azar de distintos tamaños y tipos (binarias y reales) para RECONTRA.

7.2.4. La tarea Hansards

Para abordar la tarea Hansards se ha decidido utilizar inicialmente 4 pares de codificaciones binarias al azar de tres tamaños distintos: 30 unidades (codificaciones *H_I_t30_30* y *H_II_t30_30*), 60 (codificación *H_III_t60_60*) y 240 unidades (codificación *H_IV_t240_240*).

Dada la disparidad entre el tamaño de los vocabularios y el de las codificaciones en esta tarea, se aumento el número de unidades a 240, a partir de la relación tamaño del vocabulario/número de unidades por palabra que aparecía en la tarea Turista. Así, en Turista tenemos una relación de tamaño del vocabulario/número de bits, de 23 palabras por unidad (686/30 para castellano), mientras que para Hansards obtenemos una de 21 palabras (5000/240 para francés).

[42] De especial interés es [Lawrence, 1996], dado que utilizaba redes de Elman.

En un primer momento se decidió que la red de Elman tendría similares características a las que se habían empleado con éxito en la tarea del Turista completa: una ventana de entrada de tamaño 9 (con formato 3 + 1 + 5).

La tarea Hansards tiene una longitud media de frases (12.17 palabras) mayor que Turista. Esto se debe a la presencia de unas pocas frases muy largas (hasta un máximo de 50 palabras), las cuales no se espera que el traductor sea capaz de traducir correctamente en su totalidad, por lo que este tamaño nos parece razonable.

Por otro lado, 450 unidades en la capa oculta parecía un tamaño demasiado reducido para la tarea Hansards dado el tamaño de su vocabulario, utilizando 600 y 900 unidades en la capa oculta. Se decidió combinar las codificaciones más grandes (240 unidades) sólo con las capas ocultas más grandes. Tras alargar el entrenamiento del experimento de menor tamaño (codificaciones de tamaño 30 y 450 unidades en la capa oculta) hasta 150 iteraciones se decidió detenerlo en los siguientes experimentos tras 50 iteraciones del conjunto de entrenamiento, dada la falta de mejora en siguientes iteraciones. Además, esto permitía reducir el coste temporal del entrenamiento.

En la Tabla 7.4 se presentan los resultados para distintas combinaciones de codificaciones y número de unidades en la capa oculta. Los mejores resultados, del 36.70% de PBT, se han obtenido para la codificación *H_III_T60_60* con 900 unidades en la capa oculta. Son de destacar los malos resultados con las codificaciones *H_I_t30_30* y *H_IV_t240_240*. Tras una evolución normal de los resultados incrementándose ligeramente la precisión en la traducción y llegando a superar el 39% de PBT para *H_IV_t240_240*, el ECM residual empezó a subir y los resultados de traducción a bajar. Se tomaron valores más reducidos del factor de aprendizaje dentro del intervalo [0.001, 0.009] observándose un comportamiento similar.

Nombre	| CO |	FBT	PBT
H_I_t30_30	450	30.23%	33.22%
H_I_t30_30	600	0.00%	8.35%
H_II_t30_30	450	30.93%	28.87%
H_II_t30_30	600	30.23%	29.97%
H_III_t60_60	450	32.13%	33.36%
H_III_t60_60	900	34.03%	36.70%
H_IV_t240_240	600	0.00%	13.45%
H_IV_t240_240	900	0.00%	11.98%

Tabla 7.4. Resultados con Hansards y codificaciones binarias al azar de diversos tamaños: 30/30, 60/60 y 240/240 y distinto número de unidades en la Capa Oculta (CO).

Se produce el fenómeno curioso de que en algunos casos obtenemos similares resultados de traducción a nivel de frase que a nivel de palabra. Esto se debe a que existe un porcentaje de frases de muy pequeño tamaño que aparecen repetidas en varias ocasiones en la tarea, con lo que probablemente la red consigue aprenderlas “de

memoria"[43]. En la Tabla 7.5 podemos observar varios ejemplos de frases traducidas correctamente, de menor a mayor tamaño.

Es importante señalar que junto a frases mal traducidas sin ningún tipo de sentido sintáctico, hay otras traducciones incorrectas que son perfectamente comprensibles, como por ejemplo (a la izquierda la traducción esperada, a la derecha la producida por la red, separadas por el símbolo #, en negrita la palabra mal traducida):

hon <Pun> lorna milne <DPu> FinFrase # hon <Pun> ***elderly*** *milne <DPu> FinFrase*

Está claro que la red está aprendiendo al menos parte de la tarea. Sin embargo, los resultados son ligeramente inferiores a los obtenidos con el método Omega sin la utilización de clases, y muy pobres comparados con los mejores obtenidos con otras técnicas.

Tamaño	Frase bien traducida
2	ajournement FinFrase adjournment FinFrase
	interpellation FinFrase inquiry FinFrase
3	les affaires étrangères FinFrase foreign affairs FinFrase
	la défense nationale FinFrase nacional defence FinFrase
	première lecture FinFrase first reading FinFrase
	deuxième lecture FinFrase second reading FinFrase
	troisième lecture FinFrase third reading FinFrase
4	report of committee FinFrase referred to committee FinFrase
5	ordre du jour FinFrase orders of the day FinFrase
	business of the senate FinFrase les travaux du sénat FinFrase
	interpellation <Gui> ajournement du débat FinFrase inquiry <Gui> debate adjourned FinFrase
6	deuxième lecture <Gui> suite du débat FinFrase second reading <Gui> debate continued FinFrase
	deuxième lecture <Gui> ajournement du débat finFrase second reading <Gui> debate adjourned FinFrase
	volume <NUM> <Com> numéro <NUM> FinFrase volume <NUM> <Com> issue <NUM> FinFrase
7	<NUM>nd session <Com> <NUM>th parliament <Com> FinFrase <NUM>nd session <Com> <NUM>th parliament <Com> FinFrase
8	débats du sénat <APa> hansard <CPa> debates of the senate <APa> hansard <CPa> FinFrase
	l <SCo> honorable j <Pun> michael forrestall <DPu> hon <Pun> j <Pun> michael forrestall <DPu> FinFrase

Tabla 7.5. *Ejemplos de frases bien traducidas* para la tarea Hansards y codificaciones binarias al azar *H_I_t30_30* (se muestra el original y la frase producida por la red).

[43] Señalar que en los resultados proporcionados por el Dr. Barachina (en comunicación privada y sin publicar) y comentados en el apartado 6.2.3 se producía un fenómeno similar.

El tamaño de la tarea[44] provoca que se tarde mucho tiempo en entrenar las redes. Como hay un porcentaje muy importante de frases que se repite, esto abre la posibilidad de eliminar las repeticiones y disminuir enormemente el tamaño del conjunto de entrenamiento (de 79537 a 40639 pares de frases), y por tanto, el tiempo necesario para entrenar la red. Se decidió comprobar el comportamiento del traductor con este tamaño más reducido, seleccionando la codificación *H_II_t30_30* y entrenándo una red con varios tamaños de la capa oculta (450, 600 y 750 unidades).

Los resultados en la Tabla 7.6 al compararlos con resultados con el conjunto de entrenamiento original (ver Tabla 7.4), muestran que los resultados en PBT son bastante similares para 600 unidades en la capa oculta. Llama la atención que los resultados en FBT son cero para 450 unidades en la capa oculta, lo cual indica claramente que en este caso la red no se ha limitado a aprender las frases que aparecen repetidas un número muy elevado de veces en el conjunto de entrenamiento.

Nombre de la Codificación	| CO |	FBT	PBT
H_II_t30_30	450	0.00%	14.64%
H_II_t30_30	600	24.12%	31.54%
H_II_t30_30	750	0.00%	9.10%

Tabla 7.6. Resultados con la codificación *H_II_t30_30* para la tarea Hansards con distintos tamaños de la capa oculta y el conjunto de entrenamiento sin frases repetidas.

Se realizaron experimentos adicionales con ventanas de entrada a RECONTRA con tamaños menores (5 palabras y formato 2+1+2) y mayores (14 palabras y formato 6+1+7) para la codificación *H_II_t30_30*. Los resultados obtenidos fueron similares a cuando se utilizaron 9 palabras en la ventana de entrada (ver Tabla 7.4). Esto sugiere que el tamaño de la ventana de entrada no es el factor más importante en el comportamiento de la red. La tabla resumen de resultados el lector puede encontrarla en el Anexo A (Tabla A.1).

7.2.5. Conclusiones

Los resultados con codificaciones al azar para las distintas tareas de traducción muestran claramente la capacidad de RECONTRA de abordar con éxito tareas de traducción en tareas no excesivamente complejas, cuando modelos y codificaciones son adecuadas.

El tamaño de las codificaciones, de la ventana de entrada y el número de unidades en la capa oculta son aspectos importantes en los resultados finales obtenidos.

Se pueden establecer tres conclusiones importantes referentes a la codificación:

1. Crear una codificación distribuida al azar para representar los vocabularios implicados en las tareas afecta al resultado.
2. Utilizar codificaciones reales o binarias no parece suponer una diferencia significativa en los resultados, a pesar de resultados previos en tareas de

[44] La cantidad de frases que la componen, el tamaño de las codificaciones y el vocabulario.

tratamiento del lenguaje natural [Lawrence, 1996], donde se decía que las codificaciones reales mejoraban los resultados.

3. El tamaño de las codificaciones es un factor importante en los resultados obtenidos, pero que una codificación sea más grande que otra no implica que sea mejor para nuestros propósitos. Codificaciones grandes crean redes grandes, que son más difíciles de entrenar por su inestabilidad.

Es importante recalcar el comportamiento de las redes en la tarea Hansards. En tareas con vocabularios grandes las ANNs tienen a hacerse inestables al aumentar de tamaño. El ECM residual oscila sin lograr mejorar significativas, a pesar de haber modificado los parámetros iniciales buscando un buen comportamiento. Se plantea la necesidad de buscar estrategias que solucionen o al menos mitiguen este problema.

7.3. Conocimiento a priori en las codificaciones

La inclusión de conocimiento a priori sobre los lenguajes implicados en la traducción de las codificaciones, es una posibilidad interesante y que puede contribuir a proporcionar buenos resultados de traducción [Castaño, 1998]. No obstante, incluir este conocimiento puede presentar dos problemas:

1. No es una garantía completa de buenos resultados.
2. Se continúa dependiendo de un experto humano para crear las codificaciones.

En este punto se experimentó con varias tareas que se prestaban más claramente a la creación de este tipo de codificaciones, en concreto la tarea ALM Extendida II y las distintas subtareas del Turista.

7.3.1. La tarea ALM

En estos experimentos se utilizaron codificaciones distribuidas binarias en una combinación de representaciones simbólicas-subsimbólicas para el vocabulario de 50 palabras en castellano y 39 en inglés de la tarea ALM extendida II.

Recordemos que subsimbólicas quiere decir que las unidades de la codificación no incorporan información de ningún tipo sobre las funciones de cada palabra en la frase, mientras que las codificaciones simbólicas se definen por que cada unidad tiene información. Así, las codificaciones empleadas en este apartado de la experimentación tienen dos tipos de unidades: una serie de unidades que únicamente diferencian entre palabras, sin ningún significado especial y otras unidades específicas para indicar:

- Género de la palabra: masculino o femenino (0/1).
- Número de la palabra: singular o plural (0/1).
- Tipo de palabra: verbo, sustantivo, adjetivo/adverbio o una clase genérica (para representar el resto). Si es un verbo está activo (1), sino esta a cero (0).

El tamaño de las codificaciones utilizadas es de 10 unidades, de las cuales varias se destinaron a cumplir la funciones ya mencionadas. En el caso extremo se utilizaron 6 unidades para características y 4 para diferenciar entre palabras con los mismos atributos.

Por ejemplo, si tenemos las palabras "círculo" y "cuadrado", que deberían tener las mismas unidades simbólicas activas: verbo (0), sustantivo (1), adjetivo (0), genérica (0), masculino (0) y singular (0), con las cuatro unidades restantes para diferenciarlas:

círculo	*0 1 0 0 0 0 0*	*0 0 1*
cuadrado	*0 1 0 0 0 0 0*	*0 1 0*

También se tuvieron en cuenta dos características globales que se intentaron mantener en todas las codificaciones:

- Palabras similares en un lenguaje tienen codificaciones similares, creando codificaciones hasta cierto punto "toscas", en las que un grupo de unidades se utilizan con preferencia para unos tipos de palabras u otros. Sólo una de las codificaciones creadas es puramente "tosca" (la *A_V_t10_10* en la Tabla 7.7). Unos ejemplos de objetos y verbos podrían ser:

círculo	*1 1 1 0 0 0 0 0*
pirámide	*0 1 1 0 0 0 0 0*
está	*0 0 0 0 0 0 1 1*
tocan	*0 0 0 0 0 1 0 1*

- Palabras equivalentes en el idioma fuente y destino tienen la misma codificación (por ejemplo, "círculo" y "*circle*"):

círculo	*1 0 0 0 1 0 0 0 1 0*	*circle*
pirámide	*1 0 0 0 1 0 0 0 1 1*	*pyramid*

Los modelos RECONTRA empleados tienen las mismas características que los empleados en la sección 7.2.1:

- Se utilizaron modelos con 14 palabras en la ventana de entrada, con formato 6 + 1 + 7 palabras.
- El tamaño de la capa oculta fue de 140 unidades.
- El entrenamiento de los modelos se produjo con Retropropagación y se detuvo tras 500 iteraciones de entrenamiento.

En la Tabla 7.7 se pueden observar los resultados de traducción obtenidos (Palabras Bien Traducidas, PBT y Frases Bien Traducidas, FBT) sobre el conjunto de test para codificaciones con varias combinaciones de estas características.

Nombre	Características de las Codificaciones				Porcentajes de traducciones correctas	
	Tosca	Número	Género	Tipo de palabra	FBT	PBT
A_I_t10_10	No	No	No	No	21.6%	80.5%
A_II_t10_10	No	Sí	No	No	26.7%	83.0%
A_III_t10_10	No	Sí	Sí	No	61.2%	92.7%
A_IV_t10_10	No	Sí	Sí	Sí	79.0%	96.7%
A_V_t10_10	Sí	No	No	No	75.6%	96.2%

Tabla 7.7. Porcentajes de frases y palabras bien traducidas (FBT y PBT) para la tarea ALM utilizando diversas codificaciones manuales con tamaño 10 unidades que incorporan conocimiento a priori.

Al observar los resultados de la Tabla 7.7, la incorporación del conocimiento de un experto humano en las codificaciones de los vocabularios permite crear codificaciones distribuidas que ofrecen buenos resultados de traducción, mejores que los obtenidos con codificaciones al azar (*A_I_t10_10*).

Como era de esperar, los mejores resultados se obtienen al incorporar más conocimiento a las codificaciones. Al incorporar unidades específicas para las características de las palabras (número, género o para el tipo de palabra) o especificar el tipo de palabras mediante un agrupamiento de unidades, los resultados mejoran.

7.3.2. La tarea del Mini Turista

Para las dos variantes de la tarea del Mini Turista, la categorizada (MinTcC) y la sin categorizar (MinTsC), se crearon codificaciones distribuidas binarias manuales. Se intentó en la medida de lo posible que las palabras sintácticamente parecidas en un lenguaje tuviesen codificaciones parecidas, y también que las codificaciones de palabras con el mismo significado en inglés y castellano fuesen iguales.

Sin embargo, no se dedicaron unidades a especificar ni el género, ni el número ni el tipo de palabra como se había hecho en algunas codificaciones para la tarea ALM Extendida II.

Los modelos RECONTRA utilizados en la experimentación tienen las siguientes características:

- Una ventana a la entrada de 6 palabras (2 + 1 + 3).
- Una capa oculta de 140 unidades.
- Entrenada con Retropropagación, se detuvo el entrenamiento tras 500 iteraciones.

Los mejores resultados de traducción obtenidos pueden verse en la Tabla 7.8. Dadas las dificultades encontradas al intentar crear este tipo de codificaciones, no se intentó crear una codificación equivalente para la tarea del Turista completa y tampoco para Hansards.

Nombre	Tarea	FBT	PBT
MinTcC_I_t25_25	Mini Turista con categorías	98.4%	99.7%
MinTsC_III_t61_52	Mini Turista sin categorías	90.4%	98.6%

Tabla 7.8. Porcentajes de frases y palabras bien traducidas (FBT y PBT) para la tarea del Mini Turista con y sin categorías utilizando codificaciones manuales que incorporan conocimiento a priori.

7.3.3. Conclusiones

Experimentalmente se ha comprobado como al añadir información lingüística a las codificaciones se mejoran los resultados de traducción. Este conocimiento puede tomar la forma de bits dedicados a características de las codificaciones, como el género o el número; un parecido entre palabras mediante codificaciones "similares" o utilizar las mismas codificaciones en el lenguaje origen y el destino.

No obstante, que un experto humano tenga que crear estas codificaciones (decidir que información utilizar y cual ignorar, como introducirla en las codificaciones, etc…) es inabordable, por lo que no se continuaron estos experimentos con el resto de tareas.

La única posibilidad que parece intuirse para continuar esta línea de investigación sería mediante la aplicación de algún método automático de detección de características de las palabras. Por ejemplo, se podría intentar determinar la función de cada palabra de los vocabularios dentro de las frases y utilizar estas etiquetas para determinar parte de la codificación para dichas palabras, dedicando un bit o unidad a cada función detectada.

Capítulo 8.

Experimentación con codificaciones automáticas

Resumen: Se presentan los experimentos realizados para obtener codificaciones automáticas de los vocabularios y los resultados de traducción obtenidos con ellas para las distintas tareas. Trabajaremos con codificaciones locales y MLP con ventana de salida. Se explora la aplicación de métodos de poda para determinar el tamaño de la capa oculta y por tanto de las codificaciones extraídas. Además, se utilizan codificaciones distribuidas con MLP, buscando reducir los recursos empleados. Se refinan las codificaciones mediante la aplicación sucesiva de varios MLPs en un esquema recurrente. También se realiza una comparativa con el método FGREP.

8.1. Características de los experimentos

En este capítulo se explora como crear automáticamente codificaciones para abordar tareas de traducción con RECONTRA. Nos centraremos en la creación de estas codificaciones mediante ANNs simples a partir de las posibilidades teóricas presentadas en el capítulo 5.

Como ya se ha comentado, una buena característica a tener en las codificaciones es que palabras similares (con contexto sintáctico similar) tengan codificaciones similares. Esta característica facilita la tarea del traductor. Una forma de lograr codificaciones con esta característica es proporcionar una ventana de salida al MLP utilizado como codificador (ver la Figura 8.1), de forma que la codificación interna del MLP refleje los contextos en los que aparece dicha palabra.

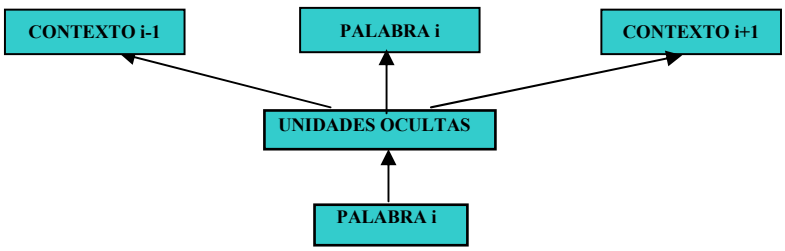

Figura 8.1. Perceptrón Multicapa con ventana de salida de tamaño 3.

No obstante, que en un MLP con ventana de salida las activaciones de las neuronas de la capa oculta reflejen la codificación de la palabra de entrada y su contexto no tiene sólo ventajas. Con el suficiente entrenamiento, la red tiende a representar clases de palabras definidas por su contexto, más que la palabra de entrada.

Para potenciar el peso de la palabra de entrada recurrimos a un procedimiento muy sencillo: la palabra de entrada se repite varias veces en la ventana de salida, para compensar el contexto, como se puede ver en la figura 8.2.

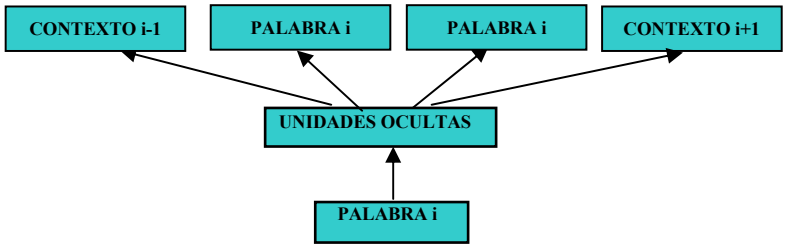

Figura 8.2. MLP con ventana de salida de tamaño 4 y la palabra de entrada *i* repetida 2 veces a la salida.

El modelo neuronal codificador que se adoptó en todos los experimentos fue:

- Topología tipo Perceptrón Multicapa (MLP) con dos capas de conexiones (capa oculta y capa de salida).
- Una palabra de entrada y una o varias (3, 4, 5, 6 u 8 palabras) de salida, en algunos casos con repetición de la palabra de entrada en ella.
- Distintos tamaños de la capa oculta, con valores fijos predefinidos o dados por los resultados de la poda del MLP.
- Diversos algoritmos de entrenamiento: Retropropagación, RPROP, SCG y combinaciones de métodos.
- Los parámetros de entrenamiento se seleccionaron siguiendo el procedimento habitual.
- Función sigmoide de activación definida en el intervalo [0,1].
- Rangos [-0.01,0.01] ó [-0.1,0.1] para inicializar los pesos de las redes.

Dado que existe un contexto de salida para cada palabra (un fragmento de la frase de la tarea de traducción), a veces fue necesario colocar palabras vacías de relleno para proporcionar este contexto.

Generalmente, se detenía el entrenamiento del MLP tras un número predeterminado de presentaciones completas (1000 o 3000 según la tarea) del conjunto de entrenamiento. Este valor se obtenía por un método de prueba y error en los experimentos iniciales para cada tarea. A la red así entrenada se le presentaban todas las palabras del vocabulario de la tarea, extrayendo las activaciones de la capa oculta para cada una de ellas.

A partir de estas activaciones, las representaciones de las palabras del vocabulario podían ser:

- Las codificaciones reales obtenidas directamente por el MLP codificador.

- Las codificaciones del MLP codificador discretizadas (binarizadas) a las que se añaden bits para distinguir entre palabras.
- Las codificaciones reales del MLP codificador a las que se añaden bits para potenciar la diferencia entre palabras (codificaciones mixtas).

En este capítulo nos centraremos en los resultados obtenidos con codificaciones reales.

Para dar nombre a las codificaciones extraídas del MLP y diferenciarlas de las codificaciones manuales del capítulo anterior, se extiende esta nomenclatura de forma que lo primero que tenemos sea la inicial de la tarea abordada, seguida por el número latino asignado a la codificación.

Inmediatamente se indica que se ha utilizado un MLP y el tamaño de la ventana de salida empleada (*v*) y el número de repeticiones de la palabra de entrada en dicha ventana de salida (*r*). A continuación se indica el tamaño de las codificaciones mediante *t* y sus tamaños. Finalizamos con el método de entrenamiento empleado con el MLP y el número de iteraciones empleadas.

Así, una codificación denominada *T_I_pmv8_r4_pS_t23_34_bp3000* significa:

1. Una codificación para la tarea del Turista "T".
2. Qué la codificación inicial es la *I* (*T_I*) y que se ha realizado automáticamente con un MLP "pm" con ventana de salida de tamaño 8 palabras "v8", con cuatro repeticiones de la palabra de entrada en la ventana de salida "r4".
3. Se ha empleado el método de poda *Skeletonization* sobre la capa oculta del MLP "pS".
4. El tamaño de las codificaciones es 23 unidades para el lenguaje origen y 34 para el lenguaje destino "t23_34".
5. Se ha entrenado el MLP con el método de Retropropagación del Error "bp" durante 3000 iteraciones "bp3000".

Por último señalar que la investigación mostrada en este capítulo ha dado lugar a una serie de publicaciones: [Casañ, 2003a], [Casañ, 2003b], [Casañ, 2003c], [Casañ, 2004], [Casañ, 2005a], [Casañ, 2005b] y [Casañ, 2007].

8.2. Experimentación con codificaciones de tamaño fijo

En un primer momento se trabaja con MLPs entrenados únicamente con el método de Retropropagación del Error y codificaciones locales a la entrada y a la salida de la red. El tamaño de la capa oculta viene dado a priori y no se aplica poda.

En esta experimentación no se trabaja con la tarea Hansards dado que se emplean codificaciones locales con los MLPs codificadores, y estos tendrían un tamaño demasiado grande[45] para resultar prácticos.

8.2.1. La tarea ALM

Para facilitar la lectura de este apartado, dada la gran cantidad de experimentación que se presenta en él, está dividido en dos subsecciones.

En la primera se agrupan una serie de experimentos realizados con MLPs codificadores en los que el contexto tenía más importancia que la palabra de entrada. Recordemos que dar menos importancia a la palabra de entrada que a su contexto supone, en términos del simulador neuronal utilizado, que la palabra de entrada se presenta una única vez en la ventana de salida. Este método crea codificaciones en las que, en su forma extrema, la representación no es de la palabra en sí, sino de su contexto.

En la segunda se presentan una serie de experimentos en los que se da igual o mayor importancia a la palabra de entrada sobre su contexto. Esto supone, en términos del simulador neuronal utilizado, que la palabra de entrada se repite varias veces a la salida[46].

Las características de las ANNs empleadas en los experimentos con ALM son las siguientes:

Para el MLP Codificador:

- 6 ó 10 unidades ocultas.
- 1000 presentaciones del conjunto de entrenamiento.
- Salida sin ventana (1 palabra) o con ventanas de tamaños 3, 4, 5, 6 u 8, con o sin repeticiones.
- Codificaciones locales a la entrada y a la salida.
- Método de entrenamiento Retropropagación del Error.

Para el Traductor:

- Ventana de entrada: 14 (6+1+7) palabras.
- Capa oculta: 140 unidades.
- 500 presentaciones del conjunto de entrenamiento.

8.2.1.1. Dando en el MLP menos importancia a la palabra de entrada que a su contexto.

Se realizaron una serie de experimentos entrenando MLP con codificaciones locales y ventanas de salida sin repetición de la palabra de entrada (con tamaños de tres y cinco

[45] Un MLP con 240 unidades en la capa oculta y ventana de salida de tamaño 4 tendría 5961600 pesos que entrenar para francés y 5716800 para inglés.

[46] Podrían haberse utilizado pesos específicos (no entrenables) para calcular el error durante el entrenamiento y lograr el mismo efecto, pero era más sencillo hacerlo de esta forma, dado el simulador.

palabras). Además, para poder apreciar la contribución de la ventana de salida, se realizó un experimento con MLP sin ventana (algo ya realizado en [Castellano, 1998] para la tarea ALM sin ampliar).

Los resultados pueden observarse en la Tabla 8.1 y muestran claramente que una ventana de salida mejora la calidad de las codificaciones, reflejada en un incremento en los resultados de traducción. Es prematuro hablar acerca del mejor tamaño de la ventana de salida, pero parece que la ventana de menor tamaño (3) produzca mejores codificaciones que la de mayor (5). Probablemente esto es debido a que las activaciones de la capa de neuronas ocultas del MLP codificador no nos reflejan sólo la codificación de la palabra de entrada, sino la de toda la salida.

Por otra parte, los resultados obtenidos al aplicar directamente las codificaciones reales proporcionadas por el MLP a la tarea de traducción ALM Extendida-II no son tan buenos como los logrados con las codificaciones binarias manuales incorporando conocimiento (número, género, tipo de palabra o codificaciones "toscas") mostrados en la Tabla 7.7 aunque sí mejores que codificaciones al azar del mismo tamaño.

Nombre	FBT	PBT
A_L_pmv1_t10_10_bp1000	0,30%	48,51%
A_L_pmv3_r1_t10_10_bp1000	20,70%	86,80%
A_L_pmv5_r1_t10_10_bp1000	10,60%	83,26%
A_L_pmv5_r1_t6_6_bp1000	5,30%	77,08%

Tabla 8.1. Porcentajes de FBT y PBT tras 500 presentaciones del conjunto de entrenamiento para diversos experimentos de traducción con codificaciones basadas en la codificación real extraída de MLPs con tamaños de la ventana de salida de tamaños 1, 3 y 5 y 6 y 10 unidades en la capa oculta.

8.2.1.2. Dando en el MLP igual o más importancia a la palabra de entrada que a su contexto.

En esta experimentación se dio igual o más importancia a la palabra de entrada que a su contexto en la salida del MLP. Para lograr esto se repitió en la ventana de salida la palabra de entrada. Dado el gran número de experimentos realizados y la similitud entre todos ellos, nos limitaremos a presentar dos de ellos con detalle y un resumen del resto.

Un resumen de los porcentajes de PBT y FBT para todos los experimentos realizados puede observarse en las Tablas 8.2 (para codificaciones de tamaño 6) y 8.3 (para codificaciones de tamaño 10).

Al examinarlos se puede comprobar cómo los resultados son sensiblemente mejores a los obtenidos sin utilizar repetición de la palabra de entrada en la salida para codificaciones del mismo tamaño (comparar Tablas 8.1 y 8.3), incluso cuando el tamaño de la codificación es menor (comparar Tablas 8.1 y 8.2).

El tamaño del contexto utilizado en la ventana de salida del MLP para obtener la codificación real (2 ó 4 palabras) no parece ser un factor excesivamente significativo en los porcentajes de traducción obtenidos. Al mismo número de repeticiones de la palabra

de entrada a la salida, un contexto mayor parece mejorar ligeramente el porcentaje de frases bien traducidas.

Nombre	FBT	PBT
A_L_pmv4_r2_t6_6_bp1000	36.75%	86.87%
A_L_pmv6_r4_t6_6_bp1000	48.10%	90.88%
A_L_pmv8_r4_t6_6_bp1000	53.60%	91.39%
A_L_pmv8_r6_t6_6_bp1000	58.00%	91.84%

Tabla 8.2. Porcentajes de FBT y PBT obtenidos tras 500 presentaciones del conjunto de entrenamiento al traductor RECONTRA para codificaciones de tamaño 6 extraídas de MLPs.

Nombre	FBT	PBT
A_L_pmv4_r2_t10_10_bp1000	61.80%	93.66%
A_L_pmv6_r4_t10_10_bp1000	72.90%	95.21%
A_L_pmv8_r4_t10_10_bp1000	74.40%	96.06%
A_L_pmv8_r6_t10_10_bp1000	70.30%	94.46%

Tabla 8.3. Porcentajes de FBT y PBT obtenidos tras 500 presentaciones del conjunto de entrenamiento al traductor RECONTRA para codificaciones de tamaño 10 extraídas de MLPs.

8.2.2. La tarea del Mini Turista categorizada

Con el fin de obtener codificaciones distribuidas automáticas[47] para los vocabularios de la tarea categorizada, se entrenaron MLPs con ventanas de salida de distintos formatos: distinto tamaño y distinto número de repeticiones de la palabra de entrada en la salida. A continuación, se extrajeron de ellas las representaciones reales conseguidas para cada palabra de los vocabularios de la tarea de AT.

Las características de las ANNs empleadas en los experimentos con la tarea del MiniTurista categorizada son las siguientes:

Para el MLP Codificador:

- 25 unidades ocultas.
- 1000 presentaciones del conjunto de entrenamiento
- Ventana de salida de tamaños 4, 6 u 8 palabras, con repeticiones.
- Codificaciones locales a la entrada y a la salida.
- Método de entrenamiento de Retropropagación del Error.

Para el Traductor:

- Ventana de entrada: 6 (2+1+3) palabras.
- Capa oculta: 140 unidades.
- 500 presentaciones del conjunto de entrenamiento.

Como ya se ha comentado, se continuaron utilizando las mismas características para el MLP codificador del lenguaje origen que para el MLP codificador del lenguaje destino, es decir, que las codificaciones reales para el vocabulario castellano extraídas de un MLP

[47] De tamaño 25, para poder comparar los resultados con los obtenidos a partir de las codificaciones creadas manualmente.

con ventana de salida de tamaño 4 y 2 repeticiones de la palabra de entrada se emparejan con las codificaciones para el vocabulario inglés extraídas de un MLP con ventana de salida de tamaño 4 y 2 repeticiones de la palabra de entrada.

Una vez entrenados los traductores con dichas codificaciones (cuyos nombres aparecen en la Tabla 8.4), se extrajeron las tasas de traducción sobre el conjunto de prueba.Los resultados mostrados en la Tabla 8.4 fueron similares a los obtenidos con codificaciones distribuidas manuales del mismo tamaño (codificaciones *MinTcC_I_t25_25*) que se repiten en dicha tabla.

Nombre	Manual	FBT	PBT
MinTcC_I_t25_25	Sí	98.40%	99.72%
MinTcC_L_pmv4_r2_t25_25_bp1000	No	98.00%	99.64%
MinTcC_L_pmv6_r4_t25_25_bp1000	No	97.70%	99.59%
MinTcC_L_pmv8_r4_t25_25_bp1000	No	97.60%	99.49%
MinTcC_L_pmv8_r6_t25_25_bp1000	No	96.70%	99.51%

Tabla 8.4. Porcentajes de FBT y PBT de la tarea categorizada utilizando codificaciones automáticas de tamaño fijo (25) extraídas de MLP con diversos formatos de la ventana de salida e MinTcC_I_t25_25.

8.2.3. La tarea del Mini Turista sin categorías

Para la tarea del Mini Turista sin categorías, se entrenaron MLP con ventanas de salida de distinto tamaño (4, 6 y 8 palabras) y distinto número de repeticiones de la palabra de entrada en la salida (2, 4 y 6 repeticiones) durante 1000 iteraciones del conjunto de aprendizaje (alargadas en algunos casos hasta 3000). Luego se extrajeron las representaciones reales obtenidas para cada palabra y se emplearon como codificaciones de los vocabularios de la tarea de traducción.

Con el cambio de tarea también se modificaron las características del modelo RECONTRA a emplear. Así, se utilizó principalmente una ventana de entrada de 6 elementos y una capa oculta de 140 unidades, aunque también se exploraron otras combinaciones.

En resumen, las características de las ANNs empleadas en los experimentos con la tarea del MiniTurista sin categorías son las siguientes:

Para el MLP Codificador:

- 25 unidades ocultas.
- 1000 (o 3000 en algunos casos) presentaciones del conjunto de entrenamiento.
- Ventana de salida de tamaños 4, 6 u 8 palabras, con repeticiones.
- Método de entrenamiento Retropropagación del Error.
- Codificaciones locales a la entrada y a la salida.

Para el Traductor se realizaron una serie de pruebas con diversas combinaciones de la ventana de entrada y el tamaño de la capa oculta:

- Ventana de entrada: 6 (2+1+3) u 8 (3+1+4) palabras.

- Capa oculta: 140, 160 o 180 unidades.
- 500 presentaciones del conjunto de entrenamiento.

Una vez entrenados los traductores con dichas codificaciones, se calcularon las tasas de traducción sobre el conjunto de test. Los resultados son mejores que los obtenidos con codificaciones al azar del mismo tamaño (ver Tabla 7.2) y peores que los obtenidos con codificaciones manuales que incorporaban conocimiento (en la Tabla 7.8).

Al analizar los resultados obtenidos (ver Tabla 8.5) se puede observar que son peores que los alcanzados con la tarea categorizada. Esto se debe a que RECONTRA confundía sistemáticamente ciertas palabras que tenían las mismas funciones semánticas (principalmente los días del mes).

Nombre	FBT	PBT
MinTsC_L_pmv4_r2_t25_25_bp1000	18.70%	88.22%
MinTsC_L_pmv6_r4_t25_25_bp1000	62.40%	94.16%
MinTsC_L_pmv8_r4_t25_25_bp1000	45.40%	91.25%
MinTsC_L_pmv8_r6_t25_25_bp1000	50.40%	92.68%

Tabla 8.5. Porcentajes de FBT y PBT de la tarea no categorizada con RECONTRA con ventana de entrada de tamaño 6 y 140 unidades ocultas utilizando codificaciones automáticas de tamaño fijo (25) .

Con el fin de mejorar las codificaciones, se realizaron nuevos experimentos a partir de las codificaciones MinTsC_L_pmv6_r4_t25_25_bp1000, aumentando el tamaño de la capa oculta, modificando la ventana de entrada del modelo RECONTRA y el número de iteraciones. Los mejores resultados se obtuvieron al ampliar la ventana de entrada del traductor a 8 palabras, con 140 unidades en la capa oculta, consiguiendo 71.20% de FBT y 95.76% de PBT. Un resumen puede encontrarse en la tabla A.2 del anexo A.

8.2.4. La tarea del Turista

Siguiendo los experimentos realizados sobre la tarea ALM y la tarea del Mini Turista se procedió a entrenar dos MLP con ventanas de salida de formato *x-1* ***x x*** *x+1*, 30 unidades en la capa oculta utilizando codificaciones locales. El par de codificaciones resultantes se denominan *T_L_pmv4_r2_t30_30_bp3000*. El tamaño de las codificaciones decidió ampliarse respecto a las 25 unidades de MiniTurista, dado el mayor tamaño del vocabulario, y también se aumentó el tiempo de entrenamiento de los MLP[48].

En resumen, las características de las ANNs empleadas en los experimentos con la tarea del Turista son las siguientes:

Para el MLP Codificador:

- 30 unidades ocultas.
- 3000 presentaciones del conjunto de entrenamiento.

[48] Aunque aquí no se muestran, como parte del proceso de búsqueda de parámetros se obtuvieron resultados para codificaciones con 1000 y 5000 iteraciones, y fueron peores que con 3000.

- Ventana de salida de tamaño 4, con 2 repeticiones de la palabra de entrada en ella.
- Método de entrenamiento Retropropagación del Error.
- Codificaciones locales a la entrada y a la salida.

Para el Traductor se realizaron una serie de pruebas con diversas combinaciones de la ventana de entrada y el tamaño de la capa oculta, intentando determinar la mejor topología de RECONTRA para la tarea:

- Ventana de entrada: 8 (3+1+4), 9 (3+1+5) o 10 (4+1+5) palabras.
- Capa oculta: 180, 250, 300, 350, 450, 500, 550, 600 o 650 unidades.
- 150 presentaciones del conjunto de entrenamiento.

Es importante recalcar que las diferencias en el número de pesos, y por tanto el tiempo de entrenamiento, no son triviales para estas redes. En la Tabla 8.6 se puede observar claramente como el número de pesos (|Pesos| en la tabla) se puede doblar de una red a otra.

Ventana entrada RECONTRA	**CO**	**Pesos**
8	180	81000
9	180	86400
9	250	137500
9	300	180000
9	350	227500
9	450	337500
9	500	400000
9	550	467500
9	600	54000
9	650	617500
10	180	91800

Tabla 8.6. Número de unidades (|Pesos|) de modelos RECONTRA con distinto tamaño de la ventana de entrada y número de unidades en la capa oculta (|CO|), para codificaciones de tamaño 30 (para castellano e inglés).

En la Tabla 8.7 se pueden observar los resultados de traducción obtenidos con diversos modelos RECONTRA, que nos llevaron a decidir emplear una red de Elman con ventana de entrada de tamaño 9 (3+1+5) y 450 unidades en la capa oculta.

Nombre	Ventana entrada RECONTRA	CO	FBT	PBT
T_L_pmv4_r2_t30_bp3000	8	180	1.8%	46.0%
T_L_pmv4_r2_t30_bp3000	9	180	2.0%	47.5%
T_L_pmv4_r2_t30_bp3000	9	250	3.9%	53.0%
T_L_pmv4_r2_t30_bp3000	9	300	1.8%	45.8%
T_L_pmv4_r2_t30_bp3000	9	350	4.8%	55.8%
T_L_pmv4_r2_t30_bp3000	9	450	8.4%	60.7%
T_L_pmv4_r2_t30_bp3000	9	500	9.9%	61.2%
T_L_pmv4_r2_t30_bp3000	9	550	10.3%	63.0%
T_L_pmv4_r2_t30_bp3000	9	600	10.8%	61.9%
T_L_pmv4_r2_t30_bp3000	9	650	8.5%	58.7%
T_L_pmv4_r2_t30_bp3000	10	180	1.3%	46.0%

Tabla 8.7. Resultados de traducción tras 150 iteraciones de entrenamiento con codificaciones T_L_pmv4_r2_t30_bp3000 y distintas características de RECONTRA (ventana de entrada y tamaño de capa oculta, CO).

8.2.5. Conclusiones

En este punto se ha presentado un método para crear codificaciones automáticas para tareas de traducción. Tras explorar varias posibilidades sobre la salida de la red, ha quedado patente que una ventana de salida con repeticiones de la palabra de entrada en ella es la mejor opción para obtener buenas codificaciones.

De la experimentación realizada se pueden extraer varias conclusiones importantes:

- Los resultados son esperanzadores, dado que se ha logrado obtener codificaciones de forma completamente automática.
- La presencia de una ventana de salida mejora las codificaciones respecto a no tenerla en los MLP codificadores. Esto nos muestra claramente que, si no se utiliza ventana de contexto a la salida del MLP, lo que obtenemos son únicamente las representaciones internas de cada palabra pero desarrolladas al azar, sin tener en cuenta el contexto en el que se utilizan. Si se utiliza ventana de contexto, palabras que aparecen en el mismo contexto tendrán codificaciones similares.
- Los resultados son mejores al aumentar la importancia de la palabra de entrada en la salida, es decir, al repetir la palabra de entrada en la ventana de salida, debido a la tendencia de las redes a crear codificaciones para representar las clases de palabras con contextos similares.
- Se vuelve a confirmar que a mayor tamaño de la codificación, mejores resultados se obtienen en la tarea de traducción.

Hay que recordar que en los experimentos realizados hasta ahora, la codificación del vocabulario en inglés se derivaba directamente de las codificaciones desarrolladas para el vocabulario en castellano, ya que en [Castaño, 1998] se demostraba que utilizar las mismas codificaciones en castellano e inglés, producía mejores resultados de traducción. Sin embargo, al crear las codificaciones automáticamente hay que establecer una equivalencia entre una palabra en castellano y una en inglés, lo cual no es evidente

cuando más de una palabra en castellano se puede traducir por la misma palabra en inglés (por ejemplo: “el” y “la” se traducen por *the* según las circunstancias) o cuando, en otras tareas, no existe ninguna palabra en castellano que se corresponda con una determinada palabra en inglés.

El método aún tiene varios aspectos poco explorados y que son relativamente problemáticos:

- El tamaño de las codificaciones viene dado a priori, por el tamaño de la capa oculta del MLP.
- El entrenamiento de los MLP es largo y a veces no logra encontrar un mínimo local tan bueno como nos gustaría.
- El tamaño de los MLP tiende a dispararse con tareas grandes (y por tanto el tiempo de entrenamiento), dado que utiliza codificaciones locales a la entrada y a la salida de la red: este es motivo por el que no se han empleado MLP y codificaciones locales para la tarea Hansards.

En los puntos siguientes se muestra como se han intentado resolver estos problemas.

Aunque no se han incluido en las tablas de resultados, vale la pena indicar que se realizaron algunos experimentos con la tarea ALM en los cuales se binarizaban las codificaciones extraídas de los MLP, pero los resultados obtenidos fueron decepcionantes, estando en algun caso cercanos al 0% de FBT. En muchos casos, al binarizar distintas palabras acababan teniendo la misma codificación.

Se exploró como resolver este problema añadiendo nuevas unidades (bits) con valores al azar, para diferenciar entre palabras con la misma codificación binaria, pero se lograron peores resultados que con codificaciones reales del mismo tamaño. Los resultados obtenidos muestran claramente que en los bits añadidos a una codificación extraída de un MLP con ventana de salida, de alguna forma, pueden eliminar o enturbiar el aprendizaje realizado por dicho MLP. Incluso cuando los bits, en lugar de intentar mantener las clases diferenciando las palabras, codificaban información sintáctica[49], sólo en un caso mejoraron ligeramente los resultados respecto a la codificación original real de tamaño 6.

8.3. Experimentación con codificaciones podadas

En la experimentación llevada a cabo en el anterior punto, un paso del proceso de traducción continuaba siendo manual. A pesar de que las codificaciones se obtenían automáticamente de MLP entrenados al efecto, el tamaño de estas codificaciones aún tenía que ser determinado por el experto humano.

Para automatizar este proceso se puede utilizar un algoritmo de poda que elimine progresivamente neuronas de la capa oculta del MLP, y reduzca el tamaño de las

[49] De forma similar a como se hizo al crear codificaciones con conocimiento humano: singular/plural, femenino/masculino, etc…

codificaciones. Esto implica que el tamaño de las codificaciones en los idiomas fuente y destino puede ser distinto.

Un ejemplo de como la poda puede afectar el tamaño de las redes puede observarse en la Tabla 8.8, donde se muestra el número de pesos para la tarea del Turista con RECONTRA con 450 unidades en la capa oculta y ventana de entrada de tamaño 9, y MLP con ventana de salida de tamaño 4, para distintos tamaños de las codificaciones.

Codificación Inicial	Codificación extraída del MLP	Pesos a entrenar en MLP	Pesos a entrenar en Elman
686	-	-	3211650
686	100	343000	652500
686	34	116620	352800

Tabla 8.8. Número de pesos en RECONTRA y en MLP para la tarea del Turista con codificaciones de diversos tamaños: locales, de tamaño 100 y 34/28 (castellano/inglés).

En este apartado se aplicó un único método de poda, descrito previamente en los capítulos 3 y 6, *Skeletonization*, que se basa en la eliminación de la unidad que está contribuyendo menos en el cálculo del ECM residual.

En la figura 8.3 puede observarse un ejemplo de la evolución el ECM residual durante el proceso de poda. Como se explicaba en el Algoritmo 3.1, la red seleccionada para proseguir el entrenamiento es la que tiene un ECM menor de las creadas durante el proceso de poda.

Todos los experimentos presentados emplean codificaciones locales como entrada y salida de los MLP, por lo que se continúa sin trabajar con la tarea Hansards. Además, vistos los buenos resultados obtenidos anteriormente se abandonó la tarea ALM.

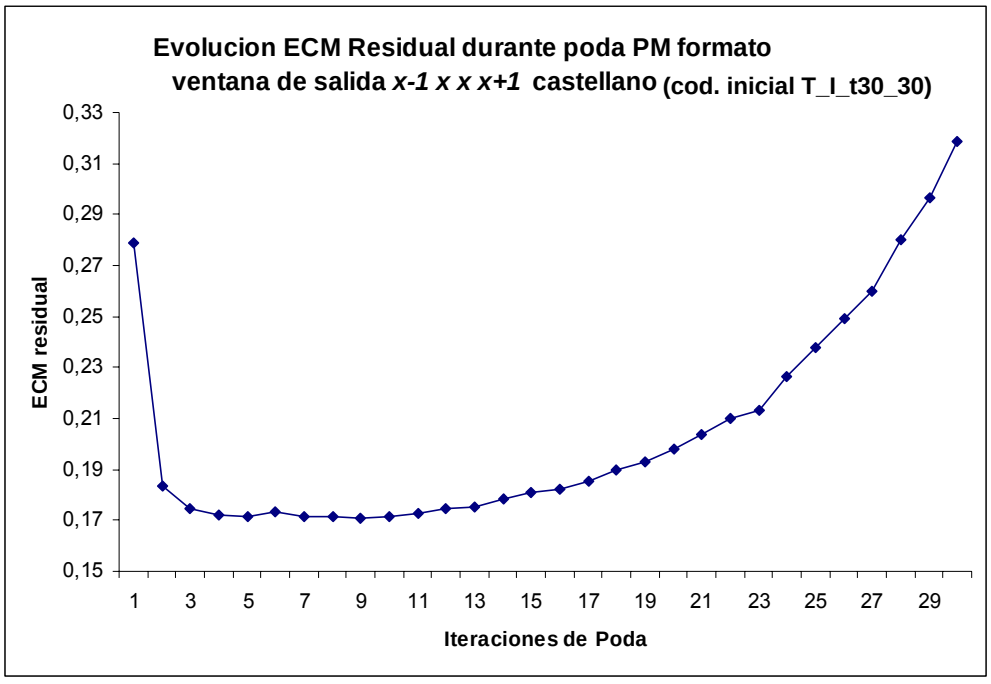

Figura 8.3. Evolución del ECM residual para MLP con ventana de salida de tamaño 4 e inicialmente 30 unidades en la capa oculta para el vocabulario castellano de la tarea del Turista, con codificación inicial *T_I_t30_30*.

8.3.1. La tarea del Mini Turista categorizada

Se seleccionaron los MLPs (para castellano e inglés) con ventana de salida de tamaño 4 y 2 repeticiones de la palabra de entrada a la salida y los de ventana de salida de tamaño 8 y 4 repeticiones de la palabra de entrada. Estos MLP tenían inicialmente 25 unidades en la capa oculta, que se podaron con el método de *Skeletonization* sucesivamente cada 10 presentaciones completas de la muestra de aprendizaje.

En cada uno de los MLP se seleccionó la red cuyo ECM residual era mínimo y se extrajeron de ella las codificaciones correspondientes. Para el MLP codificador del castellano se consiguió un tamaño de 9 unidades ocultas y para el de inglés, de 10. Las codificaciones así obtenidas se denominaron *MinTcS_L_pmv4_r2_pS_t9_10_bp1000* y *MinTcC_L_pmv8_r4_pS_t11_12_bp1000.*

En resumen, las características de las ANNs empleadas en los experimentos con la tarea del Mini Turista categorizada son las siguientes:

Para el MLP Codificador:

- 25 unidades ocultas inicialmente.
- Método de poda *Skeletonization*.
- 1000 o 3000 presentaciones del conjunto de entrenamiento.
- Ventanas de salida de tamaños 4, 6 u 8, con repeticiones de la palabra de entrada en ellas.
- Método de entrenamiento Retropropagación del Error.
- Codificaciones locales a la entrada y a la salida.

Para el traductor:

- Ventana de entrada: 6 (2+1+3) palabras.
- Capa oculta: 180 unidades.
- 500 presentaciones del conjunto de entrenamiento.

Algunos experimentos demostraron que las codificaciones obtenidas tras podar MLP entrenados durante 1000 presentaciones del conjunto de aprendizaje no proporcionaban buenas tasas de traducción. Así pues, las redes codificadoras podadas arriba indicadas se entrenaron hasta 3000 iteraciones (*MinTcC_L_pmv4_r2_pS_t9_10_bp3000* y *MinTcC_L_pmv8_r4_pS_t11_12_bp3000*), bajo la suposición de que a los MLP les costaba más tiempo encontrar un mínimo local.

Las tasas de traducción obtenidas con las codificaciones extraídas de los MLPs aparecen en la Tabla 8.9. Como puede observarse, los porcentajes de traducción obtenidos son ligeramente mejores para 3000 iteraciones del MLP.

Por otro lado, si comparamos con los resultados sin poda (Tabla 8.4), los resultados que se observan en la Tabla 8.9 muestran claramente que este método reduce el tamaño de la red y el tiempo de cómputo de forma significativa, manteniendo los porcentajes de traducción, aunque aumenta la inestabilidad de las redes.

Nombre	|Iteraciones MLP|	FBT	PBT
MinTcC_L_pmv4_r2_pS_t9_10_bp1000	1000	96.50%	99.00%
MinTcC_L_pmv4_r2_pS_t9_10_bp3000	3000	95.20%	98.88%
MinTcC_L_pmv8_r4_pS_t11_12_bp1000	1000	53.20%	94.32%
MinTcC_L_pmv8_r4_pS_t11_12_bp3000	3000	96.10%	99.15%

Tabla 8.9. Porcentajes de FBT y PBT de la tarea del Mini Turista categorizada con codificaciones extraídas tras podar MLP.

8.3.2. La tarea del Mini Turista sin categorias

Se seleccionaron MLPs con distintos tamaños de ventana de salida y las mismas características que en experimentos de la tarea categorizada y se les aplicó el algoritmo de poda. En cada caso se seleccionó el MLP cuyo ECM residual era mínimo y se extrajeron de él las codificaciones correspondientes tras entrenarlo durante 3000 presentaciones del conjunto de entrenamiento.

Estas codificaciones se utilizaron para abordar la tarea no categorizada mediante traductores RECONTRA con las mismas características que en experimentos anteriores, ventana 6 y 140 unidades ocultas. En la Tabla 8.10 se pueden observar las tasas de traducción alcanzadas sobre el corpus de test. Los resultados muestran que se ha producido un descenso en los porcentajes de FBT y PBT[50].

Nombre	FBT	PBT
MinTsC_L_pmv4_r2_pS_t11_9_bp3000	25.80%	88.79%
MinTsC_L_pmv6_r4_pS_t11_11_bp3000	32.00%	90.22%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	40.30%	90.23%
MinTsC_L_pmv8_r6_pS_t11_9_bp3000	27.20%	88.05%

Tabla 8.10. Porcentajes de FBT y PBT para la tarea no categorizada obtenidos con un traductor RECONTRA con ventana de entrada 6 y 140 unidades ocultas usando codificaciones extraídas de MLPs podados.

Dado que en los experimentos sin poda los resultados de traducción habían sugerido que un mayor tamaño de la ventana de entrada de RECONTRA mejoraba las tasas de traducción, se emplearon traductores RECONTRA con diferentes tamaños de la ventana de entrada y distinto número de unidades ocultas con las codificaciones *MinTsC_L_pmv6_r4_pS_t11_11_bp3000* y *MinTsC_L_pmv8_r4_pS_t14_13_bp3000*.

Los resultados sobre el corpus de test pueden verse en la Tabla 8.11 y reflejan que el incremento de estos dos parámetros de RECONTRA mejoraba el proceso de traducción. En la Tabla 8.12 se pueden ver los resultados obtenidos para las mismas codificaciones empleadas en los experimentos de la Tabla 8.10 pero con RECONTRA con ventana de entrada de tamaño 8 (3+1+4) y 180 unidades en la capa oculta. Los resultados son ligeramente superiores.

[50] Como se puede ver al comparar con los resultados de la Tabla 8.5, donde se muestran los porcentajes alcanzados con codificaciones sin poda.

Nombre	| CO |	| Entrada |	FBT	PBT
MinTsC_L_pmv6_r4_pS_t11_11_bp3000	140	6	32.00%	90.22%
MinTsC_L_pmv6_r4_pS_t11_11_bp3000	140	10	29.20%	89.27%
MinTsC_L_pmv6_r4_pS_t11_11_bp3000	180	8	42.30%	91.89%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	140	6	40.30%	90.23%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	180	8	51.105	92.17%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	220	8	54.60%	93.59%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	250	8	52.90%	93.73%

Tabla 8.11. Porcentajes de FBT y PBT para la tarea no categorizada obtenidos con traductores con diversos tamaños de la ventana de entrada y capa oculta.

Nombre	FBT	PBT
MinTsC_L_pmv4_r2_pS_t11_9_bp3000	32.80%	90.62%
MinTsC_L_pmv6_r4_pS_t11_11_bp3000	42.30%	91.89%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	51.10%	92.17%
MinTsC_L_pmv8_r6_pS_t11_9_bp3000	39.50%	90.84%

Tabla 8.12. Porcentajes de FBT y PBT para la tarea no categorizada obtenidos con un traductor RECONTRA con ventana de entrada de tamaño 8 y 180 unidades en la capa oculta.

Una forma de mejorar las codificaciones sería volver a estimar el factor de aprendizaje y el momentum del MLP tras la poda (al fin y al cabo es una nueva red). Se hizo esta operación con MLPs con ventana de salida de tamaño 8. Tras el entrenamiento del MLP con los nuevos parámetros (durante 3000 iteraciones con el conjunto de aprendizaje) se extrajeron las codificaciones de la capa oculta. Al proceso de poda lo hemos denominada "pS_ep". La codificación resultante se denomina *MinTsC_L_pmv8_r4_pS_ep_t14_13_bp3000*. Con ellas se entrenó un modelo RECONTRA de las mismas características que los empleados anteriormente:

- 180 neuronas ocultas.
- Una ventana de entrada de tamaño 8 (3+1+4) palabras.

Los resultados de traducción alcanzados sobre el corpus de test pueden verse en la Tabla 8.13. Como en ella se aprecia, la estimación de parámetros sobre el MLP podado y no únicamente sobre el MLP original sin podar mejoraron las traducciones. En toda la experimentación posterior con codificaciones podadas se realizó esta estimación tras la poda de los MLPs y no se indica en el nombre de las codificaciones.

Nombre	Estimación tras poda	FBT	PBT
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	No	51.10%	92.17%
MinTsC_L_pmv8_r4_pS_ep_t14_13_bp3000	Sí	67.40%	95.34%

Tabla 8.13. Porcentajes de FBT y PBT para la tarea no categorizada obtenidos con codificaciones extraídas de MLPs con ventana de salida 8, con reestimación y sin reestimación de los parámetros de los MLPs podado.

Se decidió ampliar el tamaño de la capa oculta de los MLPs de 25 a 100 unidades en una serie de experimentos. El tamaño tras poda fue de 49 (castellano) y 72 (inglés) unidades. Los resultados se muestran en la Tabla 8.14 y fueron mejores.

Nombre	Iteraciones	FBT	PBT
MinTsC_L_pmv8_r4_pS100_ep_t49_72_bp1000	1000	89.10%	97.72%
MinTsC_L_pmv8_r4_pS100_ep_t49_72_bp3000	3000	90.30%	98.25%

Tabla 8.14. Resultados de traducción con RECONTRA con ventana de tamaño 8 y 180 unidades ocultas con codificaciones extraidas de MLP inicialmente con 100 unidades en la capa oculta.

8.3.3. La tarea del Turista

En el caso de la tarea Turista las características de las redes empleadas fueron las siguientes:

Para el MLP Codificador:

- 100 unidades ocultas inicialmente (300 en un experimento).
- Método de poda *Skeletonization*.
- 3000 presentaciones del conjunto de entrenamiento.
- Ventanas de salida de tamaños 4, 6 u 8, con repeticiones de la palabra de entrada en ellas.
- Método de entrenamiento Retropropagación del Error.
- Codificaciones locales a la entrada y a la salida.
- Se realiza estimación de parámetros tras la poda.

Para el traductor:

- Ventana de entrada: 9 (3+1+5) palabras.
- Capa oculta: 450 unidades.
- 150 presentaciones del conjunto de entrenamiento.

En la Tabla 8.15 se pueden ver los nombres de las codificaciones, que reflejan estas características, y los resultados de traducción obtenidos. En la tabla también se pueden ver los resultados obtenidos con codificaciones extraidas de MLPs con 300 unidades ocultas iniciales.

Los resultados de traducción fueron similares los obtenidos sin poda (Tabla 8.7) para *T_L_pmv4_r2_pS_t34_28_bp3000* y *T_L_pmv8_r4_pS_t44_43_bp3000*, y peores para el resto.

Nombre	| CO |	FBT	PBT
T_L_pmv4_r2_pS_t34_28_bp3000	100	12.0%	62.8%
T_L_pmv6_r4_pS_t24_26_bp3000	100	6.1%	51.6%
T_L_pmv8_r4_pS_t44_43_bp3000	100	5.2%	62.8%
T_L_pmv8_r6_pS_t21_23_bp3000	100	4.4%	49.6%
T_L_pmv4_r2_pS_300_t55_79_bp3000	300	0.3%	46.2%

Tabla 8.15. Porcentajes de PBT y FBT tras 150 iteraciones para la tarea del Turista obtenidos con RECONTRA usando codificaciones extraídas de MLP podados.

8.3.4. Conclusiones

En la experimentación llevada a cabo en esta sección se ha procedido a hacer completamente automático el proceso de creación de codificaciones mediante MLP. De esta forma:

- La incorporación de un método de poda de la capa oculta de los MLP reduce el tamaño de la red sin perder de forma significativa su rendimiento. Se ha comprobado experimentalmente que las codificaciones extraídas son adecuadas para abordar tareas de traducción. En general los resultados son similares con poda y sin poda.
- El tamaño inicial de la capa oculta continúa teniendo que decidirse a priori.

Sin embargo, como se comprueba con la tarea Mini Turista sin categorías, las redes podadas pueden resultar más inestables. Una solución que reduce esta inestabilidad y conduce a mejores resultados es la estimación de los parámetros tras la poda.

Cuando el algoritmo de poda reduce demasiado la capa oculta del MLP, se produce un empeoramiento significativo en el traductor, como se puede observar en la tarea Turista con la codificación *T_L_pmv8_r6_pS_t21_23_bp3000* (ver Tabla 8.15).

Señalar que aunque el tiempo total de obtención de resultados de traducción deberia aumentar enormemente (al fin y al cabo tenemos que entrenar 3 redes: dos MLP para las codificaciones y una de Elman para la traducción), experimentalmente hemos comprobado que los tiempos totales en realizar el proceso son similares.

8.4. Utilización de codificaciones distribuidas en el entrenamiento del MLP

En esta sección se plantea la posibilidad de utilizar codificaciones distribuidas para el entrenamiento de los MLPs. Aunque se aplique poda para reducir el tamaño de la capa oculta, la entrada y salida locales hacen que la red tenga un tamaño considerable y por tanto necesite un gran tiempo de entrenamiento.

Utilizar codificaciones distribuidas permite reducir el tamaño de los MLPs con la consiguiente reducción de tiempo de entrenamiento. En la Tabla 8.16 pueden observarse ejemplos de los tamaños obtenidos con y sin poda. Se muestra el número de pesos para la tarea del Turista con RECONTRA con 450 unidades en la capa oculta y ventana de entrada de tamaño 9, y MLP con ventana de salida de tamaño 4 para el castellano.

Codificación Inicial	Codificación MLP	MLP	Elman
686	-	-	3211650
686	100	343000	652500
686	34	116620	352800
30	30	4500	337500
30	17	2550	279000

Tabla 8.16. Número de pesos en RECONTRA y en MLP para la tarea del Turista con codificaciones de diversos tamaños: Locales, de tamaño 100, 34 y 17.

La solución adoptada es la misma que la adoptada para RECONTRA: la utilización de codificaciones distribuidas que sean más pequeñas que las locales. Sí, como esperabamos, los MLPs continuan siendo capaces de generar buenas codificaciones cuando se le presentan codificaciones distribuidas en lugar de locales, se lograría reducir enormemente el tamaño de los MLPs y el tiempo de entrenamiento.

En la experimentación, además de adoptar codificaciones distribuidas al azar para entrenar los MLPs, en algunos casos se ha utilizado el algoritmo de poda para determinar el tamaño de la capa oculta y por tanto el de las codificaciones resultantes.

En este apartado la experimentación se realiza sobre las tareas Turista y Hansards, dado que son tareas más grandes y las codificaciones distribuidas permiten abordarlas.

8.4.1. La tarea del Turista

Para la tarea del Turista se utilizaron algunas de las codificaciones binarias distribuidas al azar empleadas en el apartado 7.2.3[51] para entrenar el MLP. Estos tenían una ventana de salida de tamaño 4. Las características tanto en el traductor como en el codificador fueron las mismas que las del apartado 8.3.3.

Se obtuvieron dos series de codificaciones:

- Una sin aplicar ningún método de poda y con el mismo número de unidades en la capa oculta que en las codificaciones originales
- Y otra aplicando poda mediante *Skeletonization* a la capa oculta de los MLPs.

Los resultados de traducción fueron bastante irregulares, como puede verse en la Tabla 8.17. Aparentemente, al aplicar el método de poda, en muchos casos se obtenían codificaciones de un tamaño demasiado pequeño. Los resultados al mantener la capa oculta del mismo tamaño que las codificaciones iniciales fueron casi siempre mejores, como puede verse en la misma tabla.

Estos resultados sugieren que, a partir de cierto tamaño mínimo, el traductor tiene problemas para obtener buenos resultados. Esto no implica que codificaciones de tamaños grandes produzcan siempre mejores resultados.

[51] Los resultados originales de traducción obtenidos con estas codificaciones (*T_I_t30_30*, *T_II_t30*, *T_III_t30_30*, *T_IV_t60_60* y *T_IX_t30_30*) pueden observarse en la Tabla 7.3.

Por otra parte señalar que los resultados de traducción obtenidos a partir de las codificaciones extraidas de MLPs entrenados con codificaciones reales (ver el caso de la codificación *T_IX_t30_30*), son peores que los obtenidos a partir de codificaciones binarias. Probablemente necesiten más tiempo de entrenamiento o simplemente un método mejor para entrenar el MLP.

Nombre	| Castellano| / |Inglés|	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	30/30	20.7%	76.5%
T_I_pmv4_r2_pS_t17_14_bp3000	17/14	6.8%	52.5%
T_II_pmv4_r2_t30_30_bp3000	30/30	23.2%	75.3%
T_II_pmv4_r2_pS_t18_13_bp3000	18/13	5.2%	62.3%
T_III_pmv4_r2_t30_30_bp3000	30/30	24.6%	76.3%
T_III_pmv4_r2_pS_t15_22_bp3000	15/22	14.5%	70.1%
T_IX_pmv4_r2_t30_30_bp3000	30/30	12.6%	68.9%
T_IV_pmv4_r2_t30_30_bp3000	60/60	20.3%	75.7%
T_IV_pmv4_r2_pS_t49_46_bp3000	49/46	27.7%	79.7%

Tabla 8.17. Porcentajes de PBT y FBT para la tarea del Turista obtenidos con RECONTRA usando codificaciones extraídas de MLPs entrenados con codificaciones distribuidas.

Dado que un factor importante en los resultados parecía ser el tamaño final de las codificaciones tras poda, se reutilizó una de las codificaciones distribuidas al azar de mayor tamaño para entrenar MLPs con distintos formatos de la ventana de salida y poda. En concreto se utilizó la codificación *T_V_t172_129*. Los resultados obtenidos no fueron satisfactorios (ver Tabla Anexo A.3).

También se probó con otra codificación del mismo tamaño, *T_VI_t172_129* con y sin poda, sin encontrar una mejora (ver Tabla Anexo A.4).

Hasta ahora hemos utilizado el método de *Skeletonization* para realizar la poda de los MLPs. En la siguiente serie de experimentos se utiliza otro método de poda, el de "Unidades No Contributivas", denotado en la nomenclatura de las codificaciones por "pUNC".

Se aplicó este método a diversos MLPs con varios tamaños de ventanas de salida y codificaciones iniciales *T_I_t30_30* y *T_V_t30_30*. En la Tabla 8.18 se pueden observar los resultados obtenidos. A nivel de traducción se muestra una clara mejora respecto a los resultados obtenidos con codificaciones podadas mediante *Skeletonization*. En este mecanismo de poda los tiempos de poda fueron mayores.

Nombre	FBT	PBT
T_I_pmv4_r2_pUNC_t21_21_bp3000	16.8%	71.9%
T_V_pmv4_r2_pUNC_t76_31_bp3000	13.3%	75.8%
T_V_pmv6_r4_pUNC_t77_48_bp3000	11.6%	72.3%
T_V_pmv8_r4_pUNC_t87_69_bp3000	0.1%	41.4%
T_V_pmv4_r2_pUNC_t76_49_bp3000	3.0%	65.3%

Tabla 8.18. Porcentajes de PBT y FBT para la tarea del Turista obtenidos con RECONTRA usando codificaciones extraídas de MLPs podados mediante el método de Unidades No Contributivas.

De forma puntual se realizaron experimentos con *cross-validation* (dividiendo el conjunto de entrenamiento en cuatro y diez partes y dedicando una a validación) y las codificaciones *T_I_pmv4_r2_t30_30_bp3000* pero los resultados de traducción obtenidos fueron peores que los originales. Esto sugiere que las redes necesitan todas las muestras presentes para esta tarea.

8.4.2. La tarea Hansards

Los resultados obtenidos con codificaciones al azar sugerían que el tamaño de 240 unidades era el más adecuado de los probados para abordar la tarea, y este fue uno de los que se adoptó para el entrenamiento de los MLP. En el caso de la tarea Hansards las características de las redes empleadas fueron las siguientes:

Para el MLP Codificador:

- 240 unidades ocultas inicialmente.
- Método de poda *Skeletonization*.
- 1000 o 3000 presentaciones del conjunto de entrenamiento.
- Ventanas de salida de tamaño 4 con repeticiones de la palabra de entrada en ella.
- Método de entrenamiento Retropropagación del Error.
- Codificaciones distribuidas a la entrada y a la salida.
- Se realiza estimación de parámetros tras la poda.

Para el traductor:

- Ventana de entrada: 9 (3+1+5) o 5 (2+1+2) palabras.
- Capa oculta: 900 unidades.
- 50 presentaciones del conjunto de entrenamiento (aunque en algunos casos se detuvo antes el entrenamiento).

Se adoptó la codificación *H_IV_t240_240*, ya utilizada para abordar directamente la tarea (los resultados se presentaron en la sección 7.2.4), para entrenar MLP con 240 unidades en su capa oculta. También se adoptó una codificación de menor tamaño, con 60 bits, *H_III_t60_60*, en algunos experimentos.

Los resultados de traducción pueden verse a continuación, en la Tabla 8.19, y no representan una mejora respecto a los resultados originales (en la Tabla 7.4). Es importante señalar que tras un periodo de oscilación del ECM residual y los resultados de traducción, la red empeoró drásticamente su comportamiento y se detuvo el entrenamiento. Los mejores resultados de esta secuencia son de 29.45% de PBT.

Nombre	| CO |	FBT	PBT
H_IV_pmv4_r2_t240_240_bp1000	900	0.00%	6.39%

Tabla 8.19. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con RECONTRA usando codificaciones extraídas de MLP sin podar entrenados con codificación inicial *H_IV_t240_240*.

Gran parte del problema es la estabilidad de RECONTRA, dado que los resultados fueron cercanos del 30% de PBT antes de descender y que se detuviese el entrenamiento.

Las modificaciones en la inicialización y los parámetros de entrenamiento (incluso disminuyéndolos hasta 0.001) no lograron diferencias significativas. Tras una evolución "normal", en la cual se incrementaban los porcentajes de traducciones correctas y disminuía el ECM residual, la tendencia se invertía.

En este punto se hizo otra vez evidente un problema fundamental: el tamaño de Hansards. Entrenar dos MLPs con estimación de parámetros previa y durante un número significativo de iteraciones del conjunto completo de entrenamiento, añade un tiempo de entrenamiento enorme (semanas) al ya elevado de entrenar RECONTRA con codificaciones de tamaño 240. Una posible solución es la reducción del tamaño de los conjuntos de entrenamiento de los MLPs.

Así, se decidió limitar el tamaño seleccionando únicamente parte de todos los posibles contextos de cada palabra, sin modificar el formato de ventana de salida. Esto supone una pérdida de información importante para la red y por tanto para las codificaciones finales.

En la Tabla 8.20 pueden verse los tamaños de los conjuntos completos así como los tamaños para un único contexto por palabra "contexto1" y para tres contextos por palabra "contexto3". Estos contextos, en principio, se eligieron al azar sin ningún criterio de selección, como ya se comentó en el capítulo 6.

Además, también se realizaron experimentos con un conjunto, "contextoM1" creado con el contexto que aparecía más veces en el conjunto de entrenamiento para cada palabra. En teoría esto debería facilitar la tarea del traductor, al ser este contexto más representativo del uso de la palabra en el conjunto de entrenamiento.

Nombre	Francés	Inglés
Completo	246916	244109
contexto1	4965	4759
contextoM1	4965	4759
contexto3	13844	12783

Tabla 8.20. Tamaños de los conjuntos de entrenamiento de los MLPs con ventana de tamaño 4.

Algunas palabras aparecen en un único contexto en el conjunto de entrenamiento, con lo que la pérdida de información no es tan acusada como podría parecer en un primer momento.

Para tener más posibilidades de comparación en este punto y confirmar los resultados con 240 unidades, también se empleó una codificación al azar de menor tamaño, *H_III_t60_60* con el conjunto contexto1.

Los resultados de traducción con ambas codificaciones iniciales y modelos RECONTRA con las mismas características pueden observarse en la Tabla 8.21. Al compararlos podemos ver una ligera mejora con contexto1, y un empeoramiento con contexto3.

Con respecto a las codificaciones obtenidas tras entrenar un MLP con todo el conjunto de contextos que aparecen en la tarea (ver Tabla 8.19) se observa una clara mejora, lo cual sugiere que a partir de ahora se empleen este tipo de conjuntos para entrenar los MLPs.

Nombre	| CO |	FBT	PBT
H_IV_t240_240	900	34.63%	39.78%
H_IV_pmv4_r2_t240_240_bp3000_contexto1	900	34.54%	41.80%
H_IV_pmv4_r2_t240_240_bp3000_contexto3	900	33,83%	36.62%
H_III_t60_60	900	34.03%	36.70%
H_III_pmv4_r2_t60_60_bp3000_contexto1	900	36.04%	34.48%

Tabla 8.21. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con RECONTRA usando codificaciones extraídas de MLPs sin podar con codificaciones iniciales *H_IV_t240_240* y *H_III_t60_60*.

Se procedió a aplicar el método de poda *Skeletonization* con el conjunto de entrenamiento contexto1 y codificación inicial *H_IV_t240_240*, a MLPs. Para el francés se produce un descenso significativo en el tamaño, pasando de 240 a 192 unidades, pero para el inglés se detuvo en 213 unidades[52]. También se intentó aplicar el método UNC, pero resultó ser demasiado costoso.

Las codificaciones resultantes del entrenamiento con Retropropagación fueron extraídas de los MLPs y aplicadas a la tarea de traducción. Los resultados obtenidos pueden observarse en la Tabla 8.22 para modelos RECONTRA con 900 unidades en la capa oculta y 9 y 5 palabras en la capa de entrada. También pueden verse los resultados originales de la codificación *H_IV_t240_240* y los de las codificaciones sin poda en los MLPs (*H_IV_pmv4_r2_t240_240_bp3000_contexto1*).

Nombre	Entrada	| CO |	FBT	PBT
H_IV_t240_240	9	900	34.63%	39.78%
H_IV_pmv4_r2_t240_240_bp3000_contexto1	9	900	34.54%	41.80%
H_IV_pmv4_r2_t240_240_bp3000_contextoM1	5	900	34.23%	39.34%
H_IV_pmv4_r2_pS_t192_213_bp3000_contexto1	9	900	34.23%	42.52%
H_IV_pmv4_r2_pS_t192_213_bp3000_contexto1	5	900	35.24%	43.28%

Tabla 8.22. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con RECONTRA usando codificaciones extraídas de MLPs podados con codificaciones iniciales *H_IV_t240_240*.

Se puede ver claramente como las codificaciones podadas aportan mejores resultados de traducción, pero las diferencias son relativamente pequeñas. Esto sugiere claramente que no hay un problema con el tamaño de las codificaciones[53], sino con los modelos RECONTRA implicados y el proceso de entrenamiento o con la calidad de las codificaciones[54]. En experimentación posterior se intentan resolver (o por lo menos mitigar) ambos problemas.

[52] Al intentar aplicar este método a MLP con las codificaciones H_III_t60_60, el ECM residual nunca llegaba a descender, aunque se producía poda, lo cual sugiere que este tamaño es demasiado pequeño para abordar la tarea.

[53] Recordemos que para la tarea del Turista el proceso de poda había producido algunas codificaciones muy pequeñas, que dificultaban la labor de aprendizaje de la tarea.

[54] Y seguramente una combinación de ambos factores.

8.4.3. Conclusiones

Utilizar codificaciones distribuidas para entrenar los MLPs reduce el tamaño de las redes y por tanto su tiempo de entrenamiento.

Para la tarea del Turista se pueden comparar directamente el uso de codificaciones locales y distribuidas en los MLPs, y en esta tarea se obtienen en algunos casos mejores resultados con codificaciones distribuidas que con codificaciones locales.

Existen dos situaciones donde las codificaciones distribuidas no funcionan correctamente. En la primera, el método de poda ha generado unas codificaciones muy pequeñas con las que el traductor no puede trabajar correctamente. En la segunda, se obtienen codificaciones de tamaño relativamente grande que producen inestabilidad en el traductor debido a que algunas palabras no están bien codificadas.

Para la tarea Hansards cabe señalar que los intentos de reducir el tamaño del conjunto de entrenamiento de los MLPs han mejorado las codificaciones (deducidas de una mejora de los resultados de traducción).

Finalmente, comentar como se han probado (para la tarea del Turista) topologías distintas para los MLPs, con ventana de entrada, pero los resultados han sido claramente peores. Esta experimentación puede verse en el anexo A.2 en el apartado “Otras topologías de los MLPs”.

8.5. Otros métodos de entrenamiento del MLP

En esta sección se experimenta con otros algoritmos de entrenamiento de los MLPs ya discutidos en el apartado 6.4, cuya convergencia alcanza el mínimo de la función más rápidamente. Se emplearon los métodos RPROP y SCG, además de combinaciones de Retropropagación y RPROP, y Retropropagación y SCG.

Además, se propone utilizar varios métodos de modificación de los parámetros de entrenamiento (factor de aprendizaje y momentum) de Retropropagación durante el entrenamiento. Estos métodos, en general, se pueden aplicar en función del ECM residual o del tiempo de entrenamiento transcurrido. Una lectura más extensa de los métodos empleados para modificar los parámetros de entrenamiento puede verse en el apartado 6.4.

Así, se modificaron los parámetros en puntos fijos del entrenamiento y aplicando las tres ecuaciones propuestas en (6.17), (6.18) y (6.19).

Recordemos que los nuevos experimentos supondrán una ampliación de la nomenclatura de las codificaciones, que ahora deberán indicar que métodos de entrenamiento se han utilizado para entrenar los MLPs y durante cuantas iteraciones. Así, para indicar que en el entrenamiento del MLP se ha utilizado RPROP y SCG se utilizan las letras “rp” y “scg” seguidas por el número de iteraciones. Si se emplea una combinación de métodos, el primero va seguido por el número de iteraciones, una

guiónbajo, el nuevo método y el número de iteraciones con él, como "bp100_scg500_300.

Para indicar que se han empleado las ecuaciones (6.17), (6.18) y (6.19) se añaden tras el número de iteraciones "f1", "f2" o "f3", respectivamente.

8.5.1. Algoritmos de entrenamiento

En este apartado se aplicaron los métodos de entrenamiento del MLP explicados anteriormente, RPROP y SCG, con la esperanza de mejorar los resultados de traducción.

8.5.1.1. La tarea del Mini Turista sin categorías

Para esta tarea se decidió emplear MLPs con ventana de salida de tamaño 8, manteniendo las mismas características que los empleados en la sección 8.3.2.

Los resultados con estas codificaciones para RECONTRA con 8 palabras en la ventana de entrada y 180 unidades en la capa oculta pueden verse en la Tabla 8.23. Los resultados originales de traducción con las codificaciones extraídas de MLP entrenados empleando únicamente Retropropagación ya han podido verse en la sección 8.4 y pueden verse también en la tabla.

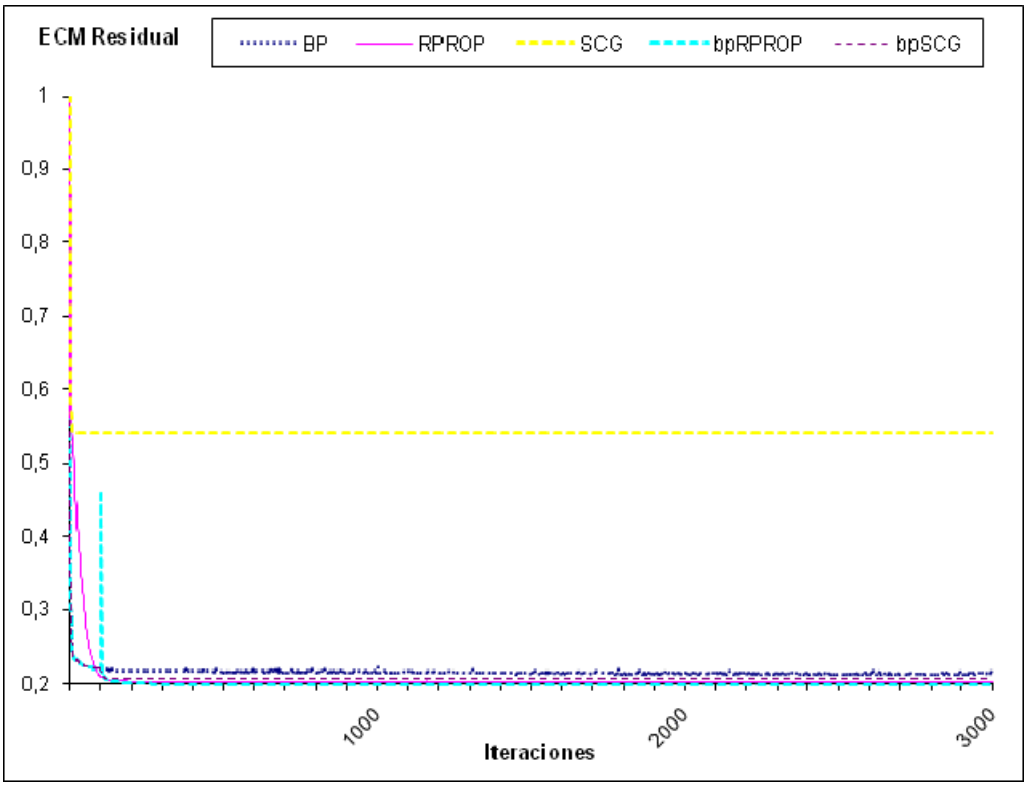

Figura 8.4. Evolución del ECM residual para un MLP (castellano) con ventana de salida *x-2 x-1 x x+1 x+2* entrenada con Retropropagación, RPROP, SCG y una combinación de BP y RPROP y BP y SCG con codificación inicial Local.

Como se puede comprobar en la Tabla 8.23, el algoritmo RPROP obtuvo resultados similares al método de Retropropagacióón, y el algoritmo SCG mucho peores, seguramente debido a que se queda estancado en un mínimo local (un ejemplo de la

evolución del ECM residual para los distintos algoritmos puede observarse en la figura 8.4).

Codificaciones	Algoritmo	FBT	PBT
MinTsC_L_pmv8_4_pS_t14_13_bp1000	BP	51.90%	93.10%
MinTsC_L_pmv8_4_pS_t14_13_bp3000		51.10%	92.17%
MinTsC_L_pmv8_4_pS_ep_t14_13_bp3000		67.40%	95.34%
MinTsC_L_pmv8_4_pS_t14_13_rp1000	RPROP	48.80%	92.98%
MinTsC_L_pmv8_4_pS_t14_13_rp3000		53.80%	93.94%
MinTsC_L_pmv8_4_pS_t14_13_scg1000	SCG	22.10%	85.96%
MinTsC_L_pmv8_4_pS_t14_13_scg3000		28.00%	89.23%
MinTsC_L_pmv8_4_pS_t14_13_bp100_rp1000	BP + RPROP	46.30%	92.69%
MinTsC_L_pmv8_4_pS_t14_13_bp100_rp3000		51.90%	93.02%
MinTsC_L_pmv8_4_pS_t14_13_bp100_scg1000	BP + SCG	64.10%	95.10%
MinTsC_L_pmv8_4_pS_t14_13_bp100_scg3000		58.605	94.44%

Tabla 8.23. Resultados tras 500 iteraciones para varias codificaciones extraídas de MLPs entrenados con distintos algoritmos de entrenamiento: Retropropagación, RPROP y SCG; para RECONTRA con ventana de entrada de 8 palabras y 180 unidades en la capa oculta.

Otra posibilidad es utilizar una combinación entre Retropropagación y otro método. Así, se consideró que tras 100 iteraciones, Retropropagación ya ha estabilizado la red y no se van a producir cambios importantes en el ECM residual. A continuación se propone el uso de otra técnica para afinar este mínimo local. Otra forma de verlo es como si se emplease Retropropagación para realizar una inicialización de los pesos de la red anes de proceder al entrenamiento.

La nomenclatura de las codificaciones se amplió para reflejar las diversas combinaciones de métodos de entrenamiento. Así, se denominaron *MinTsC_L_pmv8_r4_pS_300_t49_72 _bp1000* cuando se utiliza Retropropagación durante 1000 iteraciones, *_bp100_scg900*, cuando se utiliza una combinación de Retropropagación (100 iteraciones) y SCG (hasta 1000 iteraciones en total o hasta que el ECM residual se queda fijo en un valor).

Experimentalmente se ha comprobado que la combinación de Retropropagación y SCG obtiene los mejores resultados.

Si combinamos Retropropagación y RPROP los resultados de traducción no se modifican sensiblemente a pesar de que en la figura 8.3 se puede observar que el ECM residual obtenido es menor.

Un problema que se presenta al combinar dos métodos es decidir en que momento se debe realizar el cambio de uno a otro. Se realizó un estudio de este problema, comparando los resultados deteniendo el entrenamiento con Retropropagación en 100 iteraciones y en 500. Los resultados en 500 son similares a los obtenidos únicamente con Retropropagación, mientras que en 100 son mejores (ver Tabla 8.24).

Nombre	BP + SCG	FBT	PBT
MinTsC_L_pmv8_r4_pS_300_t49_72_bp1000	1000 + 0/0	89.1%	97.7%
MinTsC_L_pmv8_r4_pS_300_t49_72_bp3000	3000 + 0/0	90.3%	98.3%
MinTsC_L_pmv8_r4_pS_300_t49_72_bp100_scg900	100 + 900/900	26.2%	89.8%
MinTsC_L_pmv8_r4_pS_300_t49_72_bp100_scg2900	100 + 2900/2900	21.0%	89.0%
MinTsC_L_pmv8_r4_pS_300_t49_72_bp100_scg400_300	100 + 400/300	88.0%	97.7%
MinTsC_L_pmv8_r4_pS_300_t49_72_bp500_scg500	500 + 500/500	89.4%	97.4%

Tabla 8.24. Resultados de traducción con redes RECONTRA con ventana de entrada de tamaño 8 y 180 unidades ocultas para diversas codificaciones extraídas de MLP entrenados con el algoritmo de Retropropagación del Error (BP) o combinaciones de BP y SCG con distinto número de iteraciones.

Es de especial interés que los mejores resultados son prácticamente iguales a los mejores resultados obtenidos con codificaciones manuales que incorporaban conocimiento de mayor tamaño (98.6% PBT en la Tabla 7.8).

Los resultados sugieren que utilizando SCG, el ECM residual se estanca tras un reducido número de iteraciones. Continuar entrenando la red con SCG una vez alcanzado un mínimo local, es decir, cuando el ECM residual permanece invariante parece proporcionar peores codificaciones, como demuestran los resultados de traducción de la Tabla 8.24.

8.5.1.2. La tarea del Turista

Para esta tarea se repitió la experimentación presentada en la sección anterior con la tarea Mini Turista con el fin de confirmar que la combinación de BP + SCG era la que mejores resultados presentaba. Esto se realizó utilizando codificaciones distribuidas como entrada a los MLPs. Se mantuvieron las mismas características para las redes que las empleadas en la sección 8.4.1.

En la figura 8.5 podemos observar la evolución del ECM residual para 3000 presentaciones del conjunto de entrenamiento para castellano con la codificación *T_V_t30_30* con Retropropagación y con BP + SCG para un MLP con ventana de salida de tamaño 4.

Se observa como la combinación de BP y SCG ha aprendido más que sólo Retropropagación (sus valores finales son muy inferiores) y como la evolución del ECM se detiene llegado un punto del entrenamiento, con lo cual ya se pueden extraer las codificaciones.

Así, se entrena el MLP con SCG hasta que el ECM residual deja de variar (comprobando su valor en intervalos de 100 iteraciones) o se alcanza el número máximo de iteraciones permitido.

Como hacía previsible la evolución del ECM residual, este proceso de entrenamiento generó unas codificaciones que mejoraron sensiblemente los resultados de traducción respecto a los obtenidos utilizando sólo Retropropagación, como se puede ver comparando la Tablas 8.25 y 8.17 del capítulo 8, y las Tablas A.4 y A.5 del anexo A.

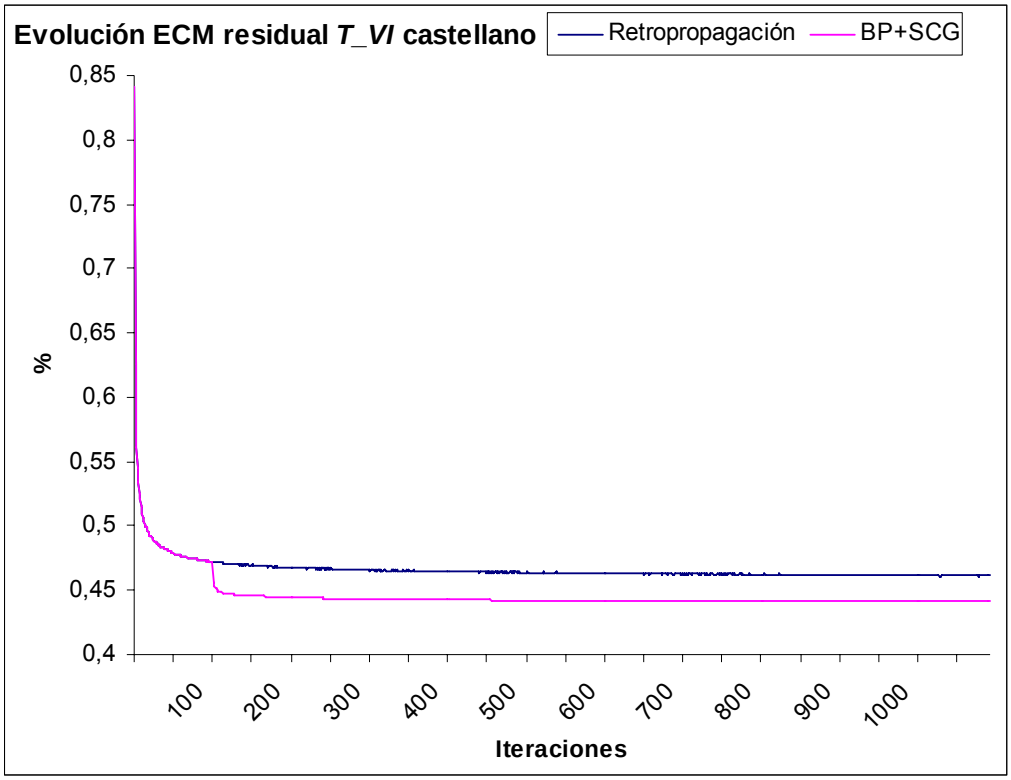

Figura 8.5. Evolución parcial (las primeras 1100 iteraciones) del ECM residual para un MLP podado con ventana de salida de tamaño 4 entrenado con Retropropagación (BP) y con una combinación de BP y SCG. La codificación inicial es *T_VI_t172_129*, una codificación distribuida al azar.

Nombre	BP + SCG	FBT	PBT
T_I_pmv4_r2_t30_30_scg1200_1300	0+1200/1300	6.5%	62.1%
T_I_pmv4_r2_t30_30_bp100_scg1100_1000	100+1100/1000	11.2%	69.2%
T_I_pmv4_r2_pS_t17_14_scg1200_1300	0+1200/1300	6.5%	57.0%
T_I_pmv4_r2_pS_t17_14_bp100_scg1200_900	100+ 1200/900	6.6%	58.1%
T_V_pmv4_r2_pS_t110_89_scg2300_2200	0+2300/2200	23.1%	75.8%
T_V_pmv4_r2_pS_t110_89_bp100_scg2900	100 + 2900/2900	2.3%	61.5%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	100 + 600/900	7.9%	70.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	100 + 600/600	22.6%	78.2%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	100 + 400/500	5.6%	65.8%
T_V_pmv8_r6_pS_t132_93_scg2400_2300	0 + 2400/2300	6,7%	63,8%
T_V_pmv8_r6_pS_t132_93_bp100_scg600_600	100 + 500/500	17.8%	73.8%
T_VI_pmv4_r2_pS_t156_115_bp100_scg600_800	100+600/800	23.6%	77.9%
T_V_pmv4_r2_pUNC_t76_31_bp100_scg600_900	100+600/900	32.2%	81.1%
T_V_pmv6_r4_pUNC_t77_48_bp100_scg700_1100	100+700/1100	30.9%	82.0%
T_V_pmv8_r4_pUNC_t87_69_bp100_scg600_800	100+600/800	37,9%	82,6%
T_V_pmv8_r6_pUNC_t76_49_bp100_scg1600_1200	100+1600/1200	35.4%	82.6%

Tabla 8.25. Porcentajes de PBT y FBT para la tarea del Turista obtenidos con RECONTRA usando codificaciones extraídas de MLP podados con distintos formatos y entrenados con SCG y combinaciones de BP y SCG.

8.5.1.3. La tarea Hansards

Para la tarea Hansards se continuaron los experimentos con tamaño reducido del conjunto de entrenamiento de los MLPs, con las mismas características de las redes (ver sección 8.4.2). Así, se adoptaron las codificaciones *H_III_t60_60* y *H_IV_t240_240* para entrenar MLP con ventana de tamaño 4. También se realizó un proceso de poda mediante *Skeletonization* a partir de *H_IV_t240_240*.

En todos los casos se entrenó la red mediante una combinación de Retropropagación (durante 100 iteraciones) y SCG, con los conjuntos de entrenamiento para los MLP "contexto1" y "contexto3". Los resultados de traducción pueden verse a continuación, en la Tabla 8.26, y en algunos casos representan una ligera mejora respecto a los resultados con las codificaciones originales (*H_III_t60_60* y *H_IV_t240_240* en la Tabla 7.4) o las obtenidas de MLP mediante Retropropagación y los mismos conjuntos. También se realizaron pruebas con ventanas de entrada 1, 2 y 5 (en la Tabla 8.27).

Como se puede ver en las tablas, la mejora de resultados es muy reducida. También es importante recalcar que aparentemente no hay diferencias significativas entre emplear una ventana de tamaño 5 o una ventana de tamaño 9 en RECONTRA.

Nombre	FBT	PBT
H_IV_pmv4_r2_t240_240_bp3000_contexto1	34.54%	41.80%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	34.84%	42.76%
H_IV_pmv4_r2_t240_240_bp3000_contexto3	33,83%	36.62%
H_IV_pmv4_r2_t240_240_bp100_scg1800_1800_contexto3	33.93%	37.49%
H_IV_pmv4_r2_pS_t192_213_bp3000_contexto1	34.23%	42.52%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	34.54%	42.17%
H_III_pmv4_r2_t60_60_bp3000_contexto1	36.04%	34.48%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	34.84%	33.93%

Tabla 8.26. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con un traductor con ventana de entrada de tamaño 9 y 900 unidades en la capa oculta usando codificaciones extraídas de MLP sin podar y entrenados con una combinación de BP y SCG con dos conjuntos distintos.

Nombre	| Ventana |	| CO |	FBT	PBT
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	9	900	34.84%	42.76%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	1	900	14.62%	31.62%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	2	900	15.72%	32.37%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	5	900	34.94%	42.49%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	5	1800	35.34%	44.80%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	5	900	34.53%	42.74%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	5	900	33.83%	32.90%

Tabla 8.27. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con un traductor con distintas ventanas de entrada, de tamaños 1, 2, 5 y 9 y 900 unidades en la capa oculta usando codificaciones extraídas de MLP y entrenados con una combinación de BP y SCG.

8.5.2. Análisis de los parámetros durante el entrenamiento

En esta sección se proponen utilizar varios métodos de modificación de los parámetros factor de aprendizaje y momentum del método de Retropropagación durante el entrenamiento. Estos métodos se pueden aplicar en función del ECM residual o del tiempo de entrenamiento transcurrido. Una lectura más extensa de los métodos empleados para modificar los parámetros de entrenamiento puede verse en el apartado 6.4.

En una primera experimentación se trabajó en la tarea Turista con reducción de los parámetros en puntos fijos y en un función de las iteraciones. Los resultados de esta experimentación no fueron consistentemente mejores que la combinación de BP + SCG. El lector puede ver un resumen de la experimentación en el anexo A.2, en la sección "Reducción de parámetros durante el entrenamiento de los MLPs".

Así, únicamente reducir los parámetros de entrenamiento no es una buena idea, y además, hacerlo únicamente en función del número de iteraciones de entrenamiento, sin tener en cuenta las circunstancias concretas de cada experimento no produce buenos resultados.

Parece mucho más razonable disminuir e incrementar los valores de los parámetros según las necesidades indicadas por la evolución del ECM residual. La idea básica es que, en las zonas donde el error disminuye se le dan mayores valores para que la red avance más rápido hacia el mínimo, y en las que aumenta, menores, para encontrar el mínimo que se ha superado. La expresión empleada es la (6.19) ya comentada en el capítulo 6.

8.5.2.1. La tarea del Turista

Para la tarea del turista se seleccionaron un gran número de codificaciones iníciales *T_I_t30_30*, *T_V_t172_129* y *T_VIII_t30_30* y se entrenaron MLP con distintas ventanas de salida. Los resultados originales con Retropropagación con parámetros fijos y variándolos según la expresión (6.19) pueden verse en la Tabla 8.28, y en la mayor parte de los casos muestran una mejora en los resultados de traducción.

Sin embargo, más interesante que la simple mejora de resultados es el hecho de la consistencia del método. Así, con independencia del tamaño de las codificaciones iniciales de los MLPs o del tamaño resultante tras aplicar el método de poda, las codificaciones generadas tienden a proporcionar resultados similares o mejores.

Un ejemplo evidente es el comportamiento de las codificaciones extraídas de MLPs con codificaciones iniciales *T_V_t172_129*, las cuales mejoran considerablemente con este método.

Nombre	Métodos	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	BP	20.7%	76.5%
T_I_pmv4_r2_t30_30_bp3000f3	(6.19)	10.9%	68.1%
T_I_pmv4_r2_pS_t17_14_bp3000	BP	6.8%	52.5%
T_I_pmv4_r2_pS_t17_14_bp3000f3	(6.19)	7.2%	58.9%
T_VIII_pmv4_r2_t30_30_bp3000	BP	12.6%	68.9%
T_VIII_pmv4_r2_t30_30_bp3000f3	(6.19)	15.2%	68.1%
T_V_pmv4_r2_pS_t110_89_bp3000	BP	0.0%	23.8%
T_V_pmv4_r2_pS_t110_89_bp3000f3	(6.19)	37.2%	82.2%
T_V_pmv4_r2_pUNC_t76_31_bp3000	BP	13.3%	75.8%
T_V_pmv4_r2_pUNC_t76_31_bp3000f3	(6.19)	28.2%	80.2%
T_V_pmv6_r4_pS_t121_94_bp3000	BP	0.0%	34.0%
T_V_pmv6_r4_pS_t121_94_bp3000f3	(6.19)	18.9%	77.3%
T_V_pmv8_r4_pS_t137_88_bp3000	BP	0.0%	32.3%
T_V_pmv8_r4_pS_t137_88_bp3000f3	(6.19)	19.6%	77.8%
T_V_pmv8_r6_pS_t132_93_bp3000	BP	0.1%	40.6%
T_V_pmv8_r6_pS_t132_93_bp3000f3	(6.19)	11.7%	73.6%

Tabla 8.28. Porcentajes de PBT y FBT para la tarea del Turista obtenidos con un traductor RECONTRA con ventana de entrada de tamaño 9 y 450 unidades en la capa usando codificaciones extraídas de MLP y entrenados sólo con Retropropagación (BP) y aplicando (6.19).

8.5.2.2. La tarea Hansards

Para la tarea Hansards se mantuvieron las características de las redes en secciones anteriores de esta tarea. Así, se adoptaron codificaciones *H_III_t60_60* y *H_IV_t240_240* para entrenar MLP con Retropropagación y expresión (6.19). También se empleó el conjunto completo de entrenamiento (durante 1000 iteraciones) además de "contexto1" (3000 iteraciones) y "contexto3" (3000 iteraciones). Se entrenó la red mediante Retropropagación aplicando la ecuación (6.19) y los resultados pueden verse a continuación, en la Tabla 8.29. Presentan una ligera mejora respecto a los resultados con MLP entrenados sólo con Retropropagación (en las Tablas 8.21 y 8.22).

Nombre	Método	FBT	PBT
H_IV_pmv4_r2_t240_240_bp1000	BP	0.00%	6.39%
H_IV_pmv4_r2_t240_240_bp1000f3	(6.19)	31.23%	31.20%
H_IV_pmv4_r2_t240_240_bp3000_contexto1	BP	34.54%	41.80%
H_IV_pmv4_r2_t240_240_bp3000f3_contexto1	(6.19)	22.93%	43.10%
H_IV_pmv4_r2_t240_240_bp3000_contexto3	BP	33.83%	36.62%
H_IV_pmv4_r2_t240_240_bp3000f3_contexto3	(6.19)	34.13%	36.69%
H_IV_pmv4_r2_pS_t192_213_ bp3000_contexto1	BP	34.23%	42.52%
H_IV_pmv4_r2_pS_t192_213_ bp3000f3_contexto1	(6.19)	34.74%	44.19%
H_III_pmv4_r2_t60_60_ bp3000_contexto1	BP	36.04%	34.48%
H_III_pmv4_r2_t60_60_ bp3000f3_contexto1	(6.19)	34.23%	33.53%

Tabla 8.29. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con un traductor con ventana de entrada de tamaño 9 y 900 unidades en la capa oculta usando codificaciones extraídas de MLP sin podar y entrenados con Retropropagación y un método de cambio de parámetros (6.19).

Se realizaron pruebas con otras ventanas de entrada de RECONTRA (tamaños 1 y 5) y diversos tamaños de la capa oculta con las codificaciones anteriores. Además, se crearon codificaciones de mayor tamaño. Así, se entrenaron MLP con 480 unidades en la capa

oculta y codificaciones iniciales *H_IV_t240_240*, y se creó una nueva codificación distribuida al azar de este tamaño para entrenar MLPs (la llamamos *H_V_t480_480*).

Los resultados obtenidos[55] pueden verse en la Tabla 8.30, y tampoco mejoraron sensiblemente respecto al entrenamiento únicamente con Retropropagación y en algunos casos empeoraron.

Nombre	|Ventana|	|CO|	FBT	PBT
H_IV_pmv4_r2_t240_240_bp2000f3	5	900	30.83%	31.09%
H_IV_pmv4_r2_t240_240_bp2000f3	5	1800	28.53%	29.20%
H_IV_pmv4_r2_bpS_t192_213_ bp1000f3	1	900	14.82%	26.68%
H_IV_pmv4_r2_bpS_t192_213_ bp1000f3	5	900	13.11%	25.31%
H_IV_pmv4_r2_bpS_t192_213_ bp1000f3	5	1800	31.93%	28.35%
H_IV_pmv4_r2_t480_480_ bp3000f3_contexto1	1	900	15.51%	31.61%
H_IV_pmv4_r2_t480_480_ bp3000f3_contexto1	5	900	32.23%	37.38%
H_V_pmv4_r2_t480_480_ bp3000f3_contexto1	5	900	32.93%	37.84%
H_III_pmv4_r2_bpS_t60_60_ bp3000f3_contexto1	5	900	33.43%	33.23%

Tabla 8.30. Porcentajes de PBT y FBT para la tarea Hansards obtenidos con traductores con ventanas de entrada de tamaños 1 y 5 y diversas unidades en la capa oculta usando codificaciones extraídas de MLP con y sin podar y entrenados con Retropropagación y un método de cambio de parámetros (6.19).

8.5.3. Conclusiones

Las pruebas realizadas con distintos métodos de entrenamiento del MLP: Retroprogación, SCG y RPROP, muestran claramente que por sí solos, Retropropagación y RPROP son los más adecuados para la tarea. SCG obtiene unos resultados poco satisfactorios, debidos a la forma de la función del error y su tendencia a quedarse "atascado" en el primer mínimo local que encuentra.

Sin embargo, experimentalmente se ha comprobado que la combinación de BP (Retropropagación) + SCG es el método que obtiene los mejores resultados. Esto es muy sencillo de explicar: Retropropagación se acerca muy rápidamente a un buen mínimo local y posteriormente SCG encuentra exactamente este mínimo, sin "oscilar" en torno a él como hace Retropropagación, especialmente con los parámetros relativamente grandes que se emplean en gran parte de la experimentación.

La combinación de métodos es una buena forma de obtener las ventajas de ambos y reducir los inconvenientes, a pesar de ser muy poco elegante.

Cuando se reducen los parámetros de entrenamiento mediante (6.17) y (6.18), en algunos casos los resultados son mejores a los obtenidos al utilizar Retropropagación sin modificar. Sin embargo, son muy dependientes del valor seleccionado para el parámetro λ.

[55] Dada la evolución del ECM residual y los resultados se detuvo su entrenamiento muy pronto.

Cuando se emplea la expresión (6.19) para modificar los parámetros de entrenamiento en función del ECM residual, los resultados son más consistentes, y tienden a ser similares a las combinaciones de BP + SCG.

En el caso de la base de datos Hansards también se producen mejoras en la traducción, en un comportamiento similar al observado con otras tareas, pero estas son muy reducidas. Se han probado un gran número de variantes de RECONTRA para la tarea Hansards y si hay algo que defina su comportamiento es la inestabilidad. Hay que explorar otros métodos en entrenamiento más rápidos y seguros y tal vez, nuevas topologías más adecuadas para la tarea.

Además, hay parámetros que aún son decididos por el experto humano (la ventana de contexto o las codificaciones iniciales), y otros para los que se buscan valores por un método de prueba y error (factor de aprendizaje y momentum).

Se exploraron otras posibilidades en la creación de codificaciones de forma automática, pero no han aportado mejoras significativas.

Se intentaron emplear codificaciones iguales en ambos idiomas para las palabras. En el caso teórico extremo esto supondría reducir el problema de traducción a un problema de reordenamiento de las palabras. No obstante, asignar codificaciones a palabras rompe los métodos automáticos empleados, con lo que aparecen nuevos problemas. En el apartado A.2 del anexo A, "Codificaciones iguales en codificaciones automáticas", puede verse parte de los resultados obtenidos y los problemas que se presentan.

Por otra parte también se añadieron unidades a las codificaciones extraídas de MLP para diferenciar una palabras de otras (en concreto "Nombres Propios" en la tarea del Turista), dado que el traductor parecía confundirlas. Esto parece en teoría una buena idea para solucionar el problema de diferenciación entre dichas palabras en el momento de la traducción.

Como se puede comprobar en las tablas A.10 y A.11 dentro del anexo A, en el subapartado "Codificaciones mixtas: añadiendo unidades a las codificaciones automáticas", las nuevas unidades ayudan a diferenciar entre los nombres propios, pero provocan confusión en el resto de palabras del vocabulario.

También se ha empleado el método FGREP para generar codificaciones para las tareas Turista y Hansards, con distintas topologías, obteniendo resultados peores que con MLPs con ventana de salida. En el subapartado "Utilización de FGREP como codificador", dentro del apartado A.2 del anexo A, se muestran los resultados obtenidos.

8.6. Realimentación de las codificaciones

Dado que en muchos casos los experimentos utilizando codificaciones binarias distribuidas como entradas a los MLPs habían producido codificaciones automáticas que daban buenos resultados de traducción, un posible paso extendiendo esta idea era utilizar estas codificaciones obtenidas automáticamente para entrenar MLPs y obtener nuevas codificaciones. En el capítulo 6 ya se comentó esta idea.

El utilizar estas codificaciones debería facilitar la tarea de aprendizaje de los MLPs y permitir obtener mejores codificaciones, en muchos casos con MLP de menor tamaño. Este esquema sería similar a la recurrencia presente en las máquinas RAAM, o podría considerarse equivalente a utilizar un modelo FGREP, en el cual las representaciones internas desarrolladas por la red son utilizadas posteriormente como entrada y salida.

Una vez obtenida una codificación automática, para simplificar y hacer más rápido el proceso de obtención de las codificaciones, se mantienen las características del MLP original. Así, no se produce una poda del MLP, sino que se parte del tamaño de codificación anterior. Además, los MLPs utilizados en la realimentación tienen las mismas ventanas que los originales.

La nomenclatura utilizada para los nombres de las codificaciones generadas con este método amplia la ya existente, incluyendo la letra *R* mayúscula y las características propias de esta red, fundamentalmente el método de entrenamiento y el número de iteraciones. Un ejemplo podría ser la codificación *T_L_pmv4_r2_pS_t30_30_bp3000_R_bp3000.*

8.6.1. La tarea del Turista

Para la tarea del Turista se seleccionaron una serie de codificaciones extraídas de MLPs entrenados con codificaciones locales y distribuidas y varios métodos en entrenamiento, y se entrenaron MLPs para obtener nuevas codificaciones. Como métodos de entrenamiento de estas nuevas redes se utilizaron tanto Retropropagación como la combinación de BP + SCG.

Los resultados de traducción con un nivel de realimentación y codificaciones distribuidas generadas a partir de codificaciones locales se muestran en la Tabla 8.31. En la mayoría de los casos muestran mejoras respecto a los obtenidos con las mismas codificaciones base (en la misma tabla).

En general, los resultados de traducción obtenidos con las codificaciones logradas tras aplicar este esquema de realimentación fueron mejores al utilizar Retropropagación que una combinación de BP + SCG, sugiriendo que la función del error es compleja y con variados mínimos locales, en uno de los cuales se queda SCG (a pesar del inicio del entrenamiento con Retropropagación). Parece que el uso de realimentación en las codificaciones hace a los MLPs más inestables.

Nombre	Realimentación	FBT	PBT
T_L_pmv4_r2_pS_t30_30_bp3000	No	8.4%	60.7%
T_L_pmv4_r2_pS_t30_30_bp3000_R_bp3000	Sí	15.3%	63.3%
T_L_pmv4_r2_pS_t30_30_bp3000_R_bp100_scg1100_1200	Sí	10.5%	58.5%
T_L_pmv4_r2_pS_t34_28_bp3000	No	12.0%	62.8%
T_L_pmv4_r2_pS_t34_28_bp3000_R_bp3000	Sí	14.3%	61.8%
T_L_pmv4_r2_pS_t34_28_bp3000_R_bp100_scg600_900	Sí	10.1%	57.0%

Tabla 8.31. Porcentajes de FBT y PBT obtenidos al traductor RECONTRA para dos codificaciones reales extraídas de MLPs con distintas características y entrenados con BP y una combinación de BP y SCG.

Las codificaciones usadas en los experimentos en las Tablas 8.15 y 8.25 fueron utilizadas para entrenar MLPs con Retropropagación obteniendo nuevas codificaciones que se aplicaron a la tarea de traducción. Los resultados, en las correspondientes Tablas 8.32 y 8.33 muestran que en la mayoría de los casos se ha producido una mejora en la traducción respecto a los originales.

Los motivos de la mejora son evidentes: la tarea de creación de representaciones de los MLP se ve facilitada por el hecho de que las codificaciones iniciales ya forman una representación adecuada (más adecuada que codificaciones distribuidas al azar, por lo menos).

Nombre	Realimentación	FBT	PBT
T_L_pmv4_r2_pS_t30_30_bp3000	No	8.4%	60.7%
T_L_pmv4_r2_pS_t30_30_bp3000_R_bp3000	Sí	15.4%	63.3%
T_L_pmv4_r2_pS_t34_28_bp3000	No	12.0%	62.8%
T_L_pmv4_r2_pS_t34_28_bp3000_R_bp3000	Sí	14.3%	61.8%
T_L_pmv6_r4_pS_t24_26_bp3000	No	6.1%	51.6%
T_L_pmv6_r4_pS_t24_26_bp3000_R_bp3000	Sí	6.6%	51.4%
T_L_pmv8_r4_pS_t44_43_bp3000	No	5.2%	62.8%
T_L_pmv8_r4_pS_t44_43_bp3000_R_bp3000	Sí	22.8%	74.9%
T_L_pmv8_r6_pS_t21_23_bp3000	No	4.4%	49.6%
T_L_pmv8_r6_pS_t21_23_bp3000_R_bp3000	Sí	8.0%	52.0%

Tabla 8.32. Porcentajes de FBT y PBT para Turista con RECONTRA utilizando codificaciones extraídas de MLPs podados con distintas ventanas de salida, entrenadas con BP, con o sin realimentación.

Nombre	Realimentación	FBT	PBT
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	No	7.9%	70.3%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900_R_bp3000	Sí	3.1%	67.7%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	No	22.6%	78.2%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_R_bp3000	Sí	38.6%	83.7%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	No	5.6%	65.7%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500_R_bp3000	Sí	14.1%	76.3%
T_V_pmv8_r6_pS_t132_93_bp100_scg600_600	No	17.8%	73.8%
T_V_pmv8_r6_pS_t132_93_bp100_scg600_600_R_bp3000	Sí	31.9%	82.3%

Tabla 8.33. Porcentajes de FBT y PBT para Turista con RECONTRA utilizando codificaciones extraídas de MLPs entrenadas con BP + SCG (sin realimentación) y con BP en la realimentación.

Se seleccionaron las codificaciones que habían proporcionado los mejores resultados en las dos tablas anteriores y se repitió el proceso de realimentación. Esto se reflejó en el nombre de las codificaciones con la notación *R2* entre las características del entrenamiento del primer MLP y el segundo.

Los porcentajes de traducción obtenidos pueden observarse en la Tabla 8.34 y revelan que los resultados no mejoraron en todos los casos, lo cual apunta a que la mejora puede detenerse tras el primer nivel de realimentación. Para confirmar que prolongar la realimentación no asegura una mejora continúa, se seleccionó un experimento y se prolongó su entrenamiento con el método de realimentación, tanto con Retropropagación como con combinaciones de BP + SCG.

Nombre	FBT	PBT
T_L_pmv4_r2_pS_t30_30_bp3000_R2_bp3000	18.6%	67.8%
T_L_pmv4_r2_pS_t34_28_bp3000_R2_bp3000	16.9%	66.3%
T_L_pmv8_r4_pS_t44_43_bp3000_R2_bp3000	29.7%	79.4%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_R2_bp3000	30.1%	80.7%

Tabla 8.34. Porcentajes de FBT y PBT para Turista con RECONTRA con codificaciones obtenidas tras dos niveles de realimentación.

Los resultados mostrados en la Tabla 8.35 confirman que la mejora no era progresiva según se añadían más niveles de realimentación. Por otra parte, se confirma que los resultados con codificaciones obtenidas de MLP entrenados con Retropropagación son mejores que cuando se usa una combinación de BP y SCG.

En la figura 8.6 se puede observar la evolución de los porcentajes de PBT para toda esta serie de experimentos, así como el del experimento original (denominado *Ini*) hasta 500 iteraciones.

Hay que remarcar que en algunos casos la realimentación no logró mejorar los resultados, especialmente tras varias aplicaciones del método. Una posibilidad es que, como le sucede al método FGREP, las codificaciones tienden a representar clases de palabras, con lo que se dificulta la tarea de traducción.

Nombre	FBT	PBT
T_L_pmv4_r2_pS_t30_30_bp3000	8.4%	60.7%
T_L_pmv4_r2_pS_t30_30_bp3000_R_bp3000	15.4%	63.3%
T_L_pmv4_r2_pS_t30_30_bp3000_R_bp100_scg1100_1200	10.5%	58.5%
T_L_pmv4_r2_pS_t30_30_bp3000_R2_bp3000	18.6%	67.8%
T_L_pmv4_r2_pS_t30_30_bp3000_R2_bp100_scg500_900	8.8%	56.3%
T_L_pmv4_r2_pS_t30_30_bp3000_R3_bp3000	11.2%	60.7%
T_L_pmv4_r2_pS_t30_30_bp3000_R3_bp100_1600_1200	5.8%	51.4%
T_L_pmv4_r2_pS_t30_30_bp3000_R4_bp3000	16.1%	65.8%
T_L_pmv4_r2_pS_t30_30_bp3000_R4_bp100_scg900_2200	3.2%	47.7%

Tabla 8.35. Porcentajes de FBT y PBT para Turista con RECONTRA con codificaciones obtenidas tras varios niveles de realimentación (R) a partir de la codificación T_L_pmv4_r2_pS_t30_30_bp3000.

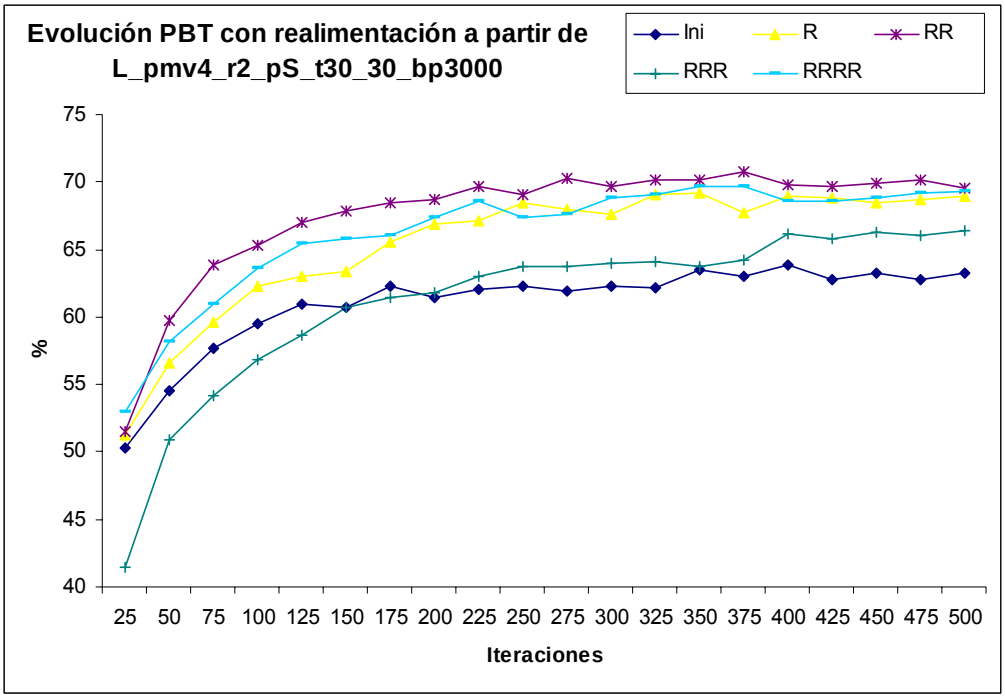

Figura 8.6. Porcentajes de PBT para la cadena de *experimentos con realimentación* con MLP con formato de ventana de salida *x-1* **x x** *x+1*, codificaciones de tamaños 30/30 y BP como método de entrenamiento. Iniciadas con las codificaciones *T_L_pmv4_r2_pS_t30_30_bp3000 (Ini)*.

8.6.2. La tarea Hansards

Como para la tarea del Turista, se utilizaron codificaciones extraídas de MLP para entrenar nuevos MLP y obtener codificaciones.

En concreto se seleccionaron las codificaciones que partían de *H_IV_t240_240* y habían sido extraídas de MLP entrenados con BP y SCG y Retropropagación con la expresión (6.19), algunas de las que habían ofrecido mejores resultados. En ambos casos se trabajó con y sin poda. De estas cuatro codificaciones se entrenaron MLP para los tres métodos de entrenamiento principales: Retropropagación, una combinación de BP y SCG y Retropropagación con modificación de parámetros mediante (6.19). Para todos los casos se trabaja con el conjunto de entrenamiento (para MLP) contexto1.

Los resultados de utilizar estas codificaciones con modelos RECONTRA con distintas ventanas de entrada y número de capas ocultas pueden verse en la Tabla 8.41 y no muestran ninguna tendencia clara.

En la mayoría de los casos los resultados de RECONTRA no sólo oscilan sino que llegado un punto del entrenamiento, incluso disminuyen. Esto, como en casos anteriores, motivó la decisión de detener el entrenamiento de los modelos RECONTRA, dada la inutilidad de continuar de la misma forma.

Nombre	|Ventana|	FBT	PBT
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	9	34.84%	42.76%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	5	34,94%	42.49%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp3000_contexto1	9	35.24%	40.37%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp3000_contexto1	5	34.64%	40.05%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp3000f3_contexto1	9	34.23%	42.32%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1	9	34.54%	42.70%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1	5	0.00%	40.91%
H_IV_pmv4_r2_t240_240_bp3000f3_contexto1	9	22.93%	43.10%
H_IV_pmv4_r2_t240_240_bp3000f3_R_bp3000_contexto1	9	34.54%	40.14%
H_IV_pmv4_r2_t240_240_bp3000f3_R_bp3000f3_contexto1	9	35.04%	43.96%
H_IV_pmv4_r2_t240_240_bp3000f3_R_bp3000f3_contexto1	5	4.81%	45.32%
H_IV_pmv4_r2_t240_240_bp3000f3_R_bp100_scg2700_3000_contexto1	9	33.43%	42.72%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	9	34.54%	42.17%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	5	34.53%	42.74%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_R_bp3000_contexto1	5	34.54%	39.94%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_R_bp3000f3_contexto1	5	34.23%	42.75%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_R_bp100_scg1000_1000_contexto1	5	35.04%	42.89%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_contexto1	9	34.74%	44.19%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_R_bp3000_contexto1	9	35.34%	40.69%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_R_bp3000_contexto1	5	34.64%	41.15%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_R_bp3000f3_contexto1	9	34.93%	44.72%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_R_bp100_scg3000_3000_contexto1	9	34.33%	43.95%

Tabla 8.41. Porcentajes de FBT y PBT obtenidos con RECONTRA con codificaciones obtenidas tras realimentación (R) de MLPs entrenados con Retroprogación, BP + SCG y Retroprogación con (6.19) con diferentes codificaciones iniciales. También aparecen los resultados originales.

Este comportamiento es especialmente decepcionante dado que cuando las redes aprendían de la forma usual, los resultados eran los mejores obtenidos hasta este punto. Se ha intentado solucionarlo de varias formas:

1. Para empezar, se intentaron diversas inicializaciones de las redes, sin ninguna diferencia significativa.
2. También se ha probado con una variedad de topologías, con ventanas de entrada de tamaños 5 y 9 y variados tamaños de la capa oculta (450, 600, 900, 1800 y 2160 unidades). Algunos de los resultados obtenidos se muestran en la Tabla 8.42, pero el comportamiento parece empeorar con redes mayores, y los resultados

disminuyen para redes más pequeñas (ver los casos con 450 y 600 unidades en la capa oculta).

3. Se ha empleado la formula (6.19) para modificar los parámetros factor de aprendizaje y momentum y evitar que las redes queden bloqueadas en un mal mínimo local. Incluso se han reducido deliberadamente el factor de aprendizaje y el momentum justo antes de la caída del ECM en algunos experimentos.

Nombre	|Ventana|	|CO|	FBT	PBT
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 3000_contexto1	9	600	34.54%	37.20%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 3000_contexto1	9	2160	36.44%	41.49%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 3000_contexto1	5	900	34.64%	40.05%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 3000f3_contexto1	9	1800	36.14%	43.89%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 3000f3_contexto1	9	2160	34.54%	43.39%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 100_scg2700_3000_contexto1	9	1800	36.64%	45.24%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 100_scg2700_3000_contexto1	9	2160	35.84%	40.71%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 100_scg2700_3000_contexto1	5	450	0.2%	41.24%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp 100_scg2700_3000_contexto1	5	1800	3.10%	45.85%
H_IV_pmv4_r2_t240_240_bp3000f3_R_bp3000f3_con texto1	9	2160	0.0%	5.72%

Tabla 8.42. Porcentajes de FBT y PBT obtenidos con codificaciones con realimentación (R) de MLPs a partir de dos codificaciones para diversos tamaños de la capa oculta y ventana de entrada de RECONTRA.

8.6.3. Conclusiones

Reutilizar codificaciones reales distribuidas extraídas de MLP para entrenar nuevos MLP y obtener nuevas codificaciones ha demostrado ser una idea que funciona bien para la tarea del Turista. Los resultados de traducción mejoran consistentemente al utilizar estas nuevas codificaciones en el traductor.

Con la tarea Hansards la situación no es tan clara. Aunque el problema principal parece presentarse en el modelo RECONTRA, y no en las codificaciones proporcionadas por los MLP no se puede tener una completa seguridad de que sea así.

La mejora producida por la utilización de codificaciones “entrenadas” en MLP no se extiende indefinidamente, es decir, utilizar la codificación extraída de un MLP para entrenar otro, no puede repetirse indefinidamente antes de que los resultados de traducción dejen de mejorar. Hay diversas causas posibles, fundamentalmente en la interacción entre un par de codificaciones y el traductor que tiene que interpretarlas, pero ninguna que podamos señalar definitivamente.

Una de las posibilidades que se tuvo en cuenta para mejorar los resultados de traducción fue la utilización de otros métodos de entrenamiento de RECONTRA distintos de Retropropagación del Error. Así, en el anexo A del apartado A.2, dentro del subapartado "Distintos métodos de entrenamiento de RECONTRA" se muestra la experimentación realizada en este sentido.

Capítulo 9.

Sistemas integrados: codificador y traductor

Resumen: En este capítulo se presenta la experimentación destinada a crear un traductor basado en RECONTRA que pueda desarrollar sus propias codificaciones para los vocabularios, al tiempo que resuelve el problema de traducción.

9.1. Introducción

Como ya se ha comentado anteriormente cuando hablamos del MLP como generador de codificaciones en los capítulos 3 y 5, una capa de neuronas tiene la capacidad de desarrollar una representación que represente la entrada a dicha capa. Así, la capa oculta de nuestro traductor no sólo está realizando la tarea de traducir sino que también está desarrollando sus propias representaciones de las palabras de entrada.

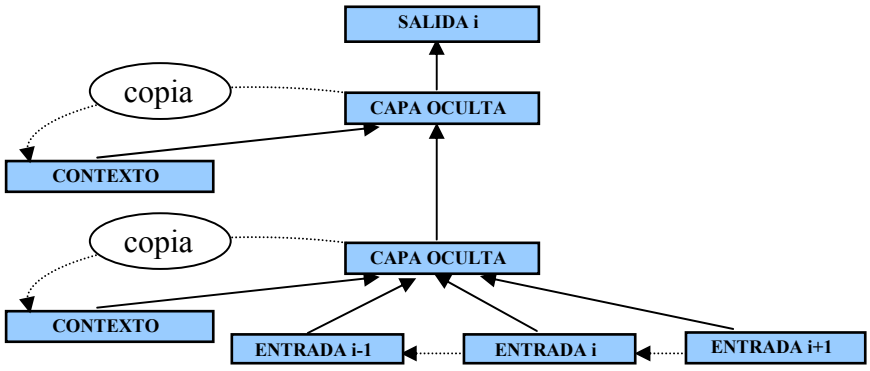

Figura 9.1. Modelos RECONTRA con dos capas ocultas.

Añadir una nueva capa de entrada al traductor tiene el potencial de ayudarle a desarrollar mejor la tarea. La primera capa oculta tiene la capacidad de especializarse en la creación de representaciones específicas para la tarea, mientras que la otra puede especializarse en la tarea de traducción en sí. En la figura 9.1 se puede observar la topología resultante.

Por desgracia incluso teóricamente esta topología sólo nos permite prescindir de la necesidad de codificaciones de entrada, no de salida. Añadir una tercera capa no debería

conducir a una mejora mayor. Pese a esto, se han realizado algunos experimentos con este tipo de redes, para asegurar que la mejora en los resultados no se produce simplemente por añadir más capas ocultas.

Desde otro punto de vista la idea de esta capa extra para mejorar la capacidad de la red está inspirada por un topología similar desarrollada por Bengio [Bengio, 2001] y destinada a abordar problemas de reconocimiento del habla.

En el anexo A, dentro del apartado A.3, en la subsección "Modelos de Bengio" se pueden observar los resultados obtenidos al emplear directamente el modelo de Bengio (e incluso MLP clásicos) para abordar la tarea, siendo los resultados peores a los obtenidos con RECONTRA. Tambien se incluye en la subsección "RECONTRA y modelos de Bengio" del anexo A.3, los resultados obtenidos al modificar los modelos RECONTRA para que tengan la topología "*Left-to-right*" empleada por Bengio.

También se esploró otra posibilidad para la creación de representaciones en el traductor del vocabulario de salida mediante una topología en la que se realimenta a la última capa su salida, como puede verse en la figura 9.2.

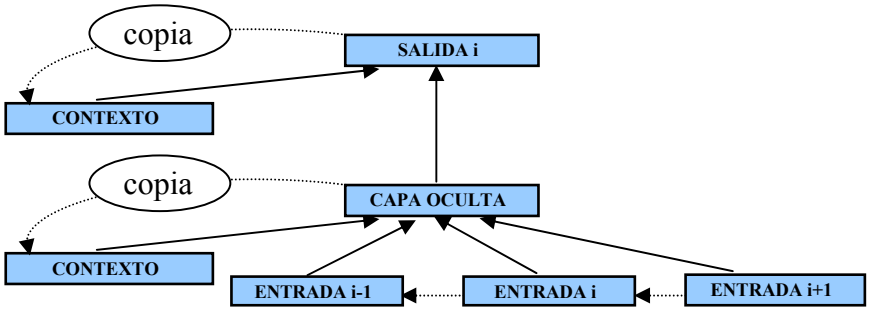

Figura 9.2. Red de Elman (RECONTRA) con ventana de entrada de tamaño 3 (i-1, i, i+1) y realimentación de la salida.

Dado que la última capa recibe su propia salida, se desarrollan representaciones de ella. Sin embargo esta topología presenta dos problemas evidentes:

1. La salida realimentada es la del momento anterior.
2. No es la única entrada de la capa, con lo que la representación no será únicamente de la palabra de salida.

Los resultados mostrados en el anexo A.3, en la subsección "Realimentación de la capa de salida" son claramente peores que los obtenidos con redes sin esta realimentación y las mismas características.

Parte de los resultados presentados en este apartado han dado lugar a la publicación [Casañ, 2008b].

9.2. RECONTRA como codificador/traductor

Como ya se ha comentado, al incorporar nuevas capas ocultas en RECONTRA esperamos que estas capas desarrollen representaciones de la entrada específicas al

problema, ahorrando considerable tiempo de entrenamiento (el tiempo empleado en entrenar el MLP) y mejorando los resultados de traducción.

9.2.1. La tarea del MiniTurista sin categorías

Aunque los resultados obtenidos para la tarea del MiniTurista sin categorías ya eran muy buenos, se decidió intentar mejorarlos aún más incorporando otra capa oculta en el traductor del mismo tamaño que la capa de entrada, siguiendo la serie de experimentos que se habían realizado para la tarea del Turista completa.

Las características de los modelos RECONTRA empleados fueron similares a las de las redes utilizadas en los experimentos del capítulo 8 con esta tarea:

- Dos capas ocultas, la segunda siempre con 180 unidades.
- Una ventana de entrada de tamaño 8 (3+1+4) palabras.

Se seleccionaron dos pares de codificaciones de entre las que habían producido mejores resultados con la tarea, *MinT_L_pmv6_r4_pS_t11_11_bp3000*, y *MinT_L_pmv8_r4_pS_t14_13_bp3000* y se entrenaron modelos RECONTRA con características similares a las utilizadas en los experimentos originales: 8 palabras en la ventana de entrada y 180 unidades en la (segunda) capa oculta. El tamaño de la primera capa oculta es el mismo que el de la capa de entrada. Así, es de 88 unidades en un caso y 112 en el otro.

Los resultados obtenidos pueden observarse en la Tabla 9.1 y no muestran mejoras respecto a los obtenidos con el traductor con una única capa oculta y las mismas características (mostrados en la Tabla 8.10). Incluso se puede hablar de ligero empeoramiento de los resultados. Esto parece deberse a una mayor inestabilidad de la red, con lo que el mínimo local encontrado es ligeramente peor.

Nombre	| 1ª CO |	| 2ª CO |	FBT	PBT
MinTsC_L_pmv6_r4_pS_t11_11_bp3000	88	180	28.0%	90.54%
MinTsC_L_pmv8_r4_pS_t14_13_bp3000	112	180	41.0%	91.62%

Tabla 9.3. Resultados de traducción con dos capas ocultas en RECONTRA con ventana de entrada de tamaño 8 y 180 unidades ocultas en la segunda capa oculta para dos codificaciones distintas, *MinT_L_pmv6_r4_pS_t11_11_bp3000* y *MinT_L_pmv8_r4_pS_t14_13_bp3000*.

9.3.2. La tarea del Turista

Para la tarea del Turista se exploraron diversas posibilidades con las redes, tanto en cantidades de capas ocultas (dos o tres) como en el número de neuronas por capa, siempre con codificaciones ya utilizadas en otros experimentos.

9.2.2.1. Diferentes tamaños de las capas ocultas.

En primer lugar, con propósitos de comparación con los modelos de Bengio, se crearon redes de Elman con dos capas ocultas. Las codificaciones empleadas fueron las codificaciones *T_I_t30_30*[56].

Las características de los modelos RECONTRA fueron variadas, pero se mantuvieron varias características en común con experimentos anteriores:

- Ventana de entrada de tamaño 9 palabras.
- Método de entrenamiento de Retropropagación del Error durante 150 iteraciones.

En la Tabla 9.2 pueden observarse los resultados obtenidos para distintas combinaciones de tamaños de las dos capas ocultas (CO) y RECONTRA con ventana de entrada de tamaño 9 palabras y codificaciones *T_I_t30_30*. En la tabla también se puede observar la cantidad total de pesos de cada red. Si comparamos estos resultados con los resultados originales (también en la tabla) puede comprobarse como en algunos casos se produce una mejora considerable, y como simplemente un mayor número de pesos no ofrece mejores resultados.

Sin embargo puede afirmarse con seguridad que una primera o segunda capa muy pequeñas son responsables de reducir la efectividad de la red.

| 1ª CO |	**| 2ª CO |**	**| Pesos |**	**FBT**	**PBT**
-	450	337500	11.0%	63.4%
30	450	238500	3.9%	52.3%
90	450	288900	8.7%	61.2%
270	450	483300	18.1%	71.3%
450	450	742500	26.2%	76.9%
600	450	1008000	24.9%	77.7%
750	450	1318500	25.9%	77.7%
270	270	299700	12.3%	68.7%
450	30	339300	3.5%	54.8%
450	60	356400	6.3%	62.1%

Tabla 9.2. Porcentajes de FBT y PBT obtenidos con modelos RECONTRA con una o dos capas ocultas de distintos tamaños, ventana de entrada de tamaño 9 y codificación *T_I_t30_30*.

Los resultados pueden verse en la tabla siguiente (9.3) para distintos tamaños de estas tres capas y son peores que los obtenidos con dos capas.

Esta tercera capa oculta no está creando una representación para el vocabulario de salida de la tarea.

[56] Los intentos de utilizar codificaciones locales para facilitar la comparación directa con los experimentos con modelos de Bengio fueron infructuosos, debido a la inestabilidad de las redes resultantes, alcanzándose resultados de 0.0% de PBT.

1ª CO	2ª CO	3ª CO	Pesos	FBT	PBT
270	450	30	485100	4.4%	57.4%
270	450	60	502200	5.1%	56.3%
270	450	270	672300	13.9%	69.2%
270	450	450	888300	15.9%	70.2%
270	270	450	710100	13.0%	68.5%

Tabla 9.3. Porcentajes de FBT y PBT obtenidos con modelos RECONTRA con tres capas ocultas de distintos tamaños, ventana de entrada de tamaño 9 y codificación *T_I_t30_30*.

Los resultados muestran claramente que simplemente añadir capas no supone una mejora (si comparamos entre tres, dos y una capa oculta) y que un número de unidades muy pequeño en alguna de las capas ocultas también empeora los resultados. Con dos capas ocultas, la primera de ellas, cuando no es muy pequeña, es capaz de mejorar los resultados creando representaciones internas de las entradas.

Nombre	1ª CO	Pesos	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	-	337500	20.7%	76.5%
T_I_pmv4_r2_t30_30_bp3000	270	483300	29.1%	80.9%
T_I_pmv4_r2_t30_30_bp3000	450	742500	32.5%	82.5%
T_I_pmv4_r2_t30_30_bp3000	-	277650	6.8%	52.5%
T_I_pmv4_r2_t30_30_bp3000	270	444510	11.4%	61.7%
T_I_pmv4_r2_t30_30_bp3000	450	682650	17.6%	68.0%
T_I_pmv4_r2_pUNC_t21_21_bp3000	-	297000	16.8%	71.9%
T_I_pmv4_r2_pUNC_t21_21_bp3000	189	639450	18.7%	75.3%
T_II_t30_30	-	337500	12.2%	67.5%
T_II_t30_30	270	483300	19.4%	76.1%
T_II_pmv4_r2_t30_30_bp3000	-	337500	23.2%	75.3%
T_II_pmv4_r2_t30_30_bp3000	270	483300	31.4%	79.4%
T_III_t30_30	-	337500	11.1%	67.4%
T_III_t30_30	270	483300	20.1%	75.9%
T_III_t30_30	750	1318500	23.5%	78.1%
T_III_pmv4_r2_t30_30_bp3000	-	337500	24.6%	76.3%
T_III_pmv4_r2_t30_30_bp3000	270	483300	26.8%	79.1%
T_III_pmv4_r2_t30_30_bp3000	-	421650	27.7%	79.7%
T_III_pmv4_r2_t30_30_bp3000	270	536670	30.1%	82.0%
T_IV_pmv4_r2_pS_t49_46_bp3000	441	810612	33.8%	83.4%
T_IV_pmv4_r2_pS_t49_46_bp3000	450	826650	35.0%	84.2%
T_IV_pmv4_r2_pS_t49_46_bp3000	750	1453950	37.5%	84.9%
T_IV_pmv4_r2_pS_t49_46_bp3000	-	734850	22.6%	78.2%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	270	733230	24.4%	81.4%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	1089	3106692	34.4%	84.2%
T_V_pmv4_r2_pUNS_t76_31_bp3000	-	524250	13.3%	75.8%
T_V_pmv4_r2_pUNS_t76_31_bp3000	270	595530	17.8%	78.6%
T_V_pmv6_r4_pS_t121_94_bp_scg600_600_R_bp3000	-	734850	38.6%	83.7%
T_V_pmv6_r4_pS_t121_94_bp_scg600_600_R_bp3000	270	733230	34.5%	83.9%
T_V_pmv6_r4_pS_t121_94_bp_scg600_600_R_bp3000	450	1139850	36.0%	84.5%
T_V_pmv8_r6_pUNC_t76_49_bp3000	-	532350	3.0%	65.3%
T_V_pmv8_r6_pUNC_t76_49_bp3000	270	603630	7.5%	71.8%

Tabla 9.4. Porcentajes de FBT y PBT obtenidos con modelos RECONTRA con una y dos capas ocultas, la primera de distintos tamaños y la segunda siempre de 450 unidades para una variedad de codificaciones.

Se decidió repetir los experimentos con una variedad mayor de codificaciones, tanto distribuidas al azar como extraídas de MLPs entrenados al efecto. Si internamente el

traductor conseguía desarrollar una codificación adecuada, los resultados de traducción deberían ser mejores que los obtenidos originalmente con las codificaciones distribuidas al azar o las derivadas de estas tras entrenar MLP. En la Tabla 9.4 podemos observar los resultados obtenidos y como en casi todos los casos los resultados de traducción mejoraron.

Experimentalmente se ha confirmado como añadir una capa oculta mejoraba los resultados, pero el tiempo de entrenamiento empezaba a crecer considerablemente en los casos en los que las codificaciones y por tanto las redes, tendían a ser más grandes. En la Tabla 9.4 se puede observar cómo el número de pesos aumenta considerablemente al añadir una primera capa oculta de un tamaño "razonable".

Recalcar que no es la red de mayor tamaño la que obtiene los mejores resultados, aunque ciertamente, sí que parece que exista una relación entre los dos factores, tamaño de la red y porcentajes de traducción.

9.2.2.2. Diferentes codificaciones de entrada.

Finalmente, se realizaron una serie de experimentos con la misma codificación de salida (*T_I_pmv4_r2_t30_30_bp3000*) y varias codificaciones de entrada, con redes con dos capas ocultas, la primera de tamaño 270 unidades y la segunda 450. Los resultados, de la Tabla 9.5, muestran como el nuevo RECONTRA es en su mayor parte insensible a las codificaciones de entrada, obteniendo resultados similares en todos los casos, excepto en dos:

1. Cuando se utiliza una codificación de entrada muy pequeña (de tamaño 17, *T_I_pmv4_r2_pS_t17_14_bp3000*), en que los resultados son peores.
2. Y cuando la codificación de entrada es "Local", que son mejores.

Esto significa que RECONTRA extendido ha reducido la dependencia de las codificaciones de entrada, aunque el tamaño de estas aún parece importante.

Codificación de Entrada	|Pesos|	FBT	FBT
T_I_pmv4_r2_t30_30_bp3000	483300	29.1%	80.9%
T_I_pmv4_r2_pS_t17_14_bp3000	451710	21.7%	77.4%
T_I_t30_30	483300	31.9%	81.7%
T_Local_t686_513	2077380	43.8%	86.7%
T_II_t30_30	483300	30.4%	81.7%
T_IV_pmv4_r2_pS_t49_46_bp3000	529470	30.8%	82.6%

Tabla 9.5. Resultados de traducción para redes RECONTRA con dos capas ocultas de tamaños 270 y 450, la misma codificación a la salida (*T_I_pmv4_r2_pS_t30_30_bp3000*) y diversas codificaciones a la entrada.

Estas mejoras sugieren utilizar siempre codificaciones locales a la entrada, pero la red se hace muy grande, muy lenta, y propensa a alcanzar mínimos locales poco adecuados. El tiempo de entrenamiento con la codificación local a la entrada es mayor, comparado con el resto de experimentación, con una perdida de resultados no muy exagerada.

A pesar del tiempo de entrenamiento que consume, se realizaron unos pocos experimentos con una codificación local a la entrada (*T_Local*) y diversas codificaciones

de salida, con distinto número de unidades en las capas ocultas. Como es evidente, el tamaño de las redes resultante es muy grande, y esto provocó problemas de estabilidad.

Como se puede observar en la Tabla 9.6, los resultados fueron en general muy buenos (en la Tabla 9.4 pueden observarse los originales para codificación *T_I_pmv4_r2_t30_30_bp3000*), y cercanos al 90% de PBT. Sin embargo, en un caso concreto los resultados fueron bastante peores[57], debido precisamente a esta inestabilidad.

En algunos de los experimentos se producía un comportamiento inestable en las redes, con oscilaciones considerables en el ECM residual. En uno de estos casos, con codificación de salida *T_I_pmv4_r2_t30_30_bp3000* se redujo a un 10% los parámetros en las últimas iteraciones, logrando mejores resultados, del 45.6% de FBT y 87.3% de PBT.

Codificación de Salida	| 1ª CO |	| Pesos |	FBT	FBT
T_I_pmv4_r2_t30_30_bp3000	-	2994300	37.3%	85.9%
T_I_pmv4_r2_t30_30_bp3000	270	2077380	43.8%	86.7%
T_I_pmv4_r2_t30_30_bp3000	450	2158380	48.1%	88.8%
T_V_pmv4_r2_pUNC_t76_31_bp3000f3	270	2077830	1.0%	58.1%

Tabla 9.6. Resultados de traducción para redes RECONTRA con la primera capa oculta de tamaños distintos (sin ella, 270 o 450 unidades), la segunda de 450 unidades, la misma codificación a la entrada (*T_Local*) y diversas codificaciones a la salida.

A continuación, podemos ver algunos ejemplos de frases mal traducidas para la codificación de entrada *T_Local*. En ellos se puede observar como, en muchos casos, la red produce frases gramaticalmente correctas para el idioma destino pero que no son la traducción correcta.

Así, las confusiones entre números y nombres propios son comunes, como también los casos de confusión singular/plural. Esto sugiere que, por ejemplo, utilizando algún tipo de clases para los números y nombres propios, mejorarían mucho los resultados. Una vez cometido un error, por otra parte, el traductor parece tender a producir palabras "aleatoriamente".

would you mind sending down my suitcase to room number four five four , please ? FinFrase# would you mind sending down my suitcase to room number four four , please ? FinFrase

I made a reservation for Asunción Rivera . FinFrase# I made a reservation for Federico Ruiz . FinFrase

I leave on June the fourteenth at a quarter past one in the afternoon . FinFrase# I leave on June the twenty-first at a quarter past one in the afternoon . FinFrase

a mistake has been made in our bill for room number four one eight . FinFrase# a mistake has been made in our bill for room number one eight oh . Barber'a

could you give me the keys to room number seven one two ? FinFrase# could you give me the keys to room number two eighteen ? FinFrase

[57] Los resultados originales con codificaciones *T_V_pmv4_r2_pUNC_t76_31_bp3000f3* y una capa oculta (con 450 unidades) eran de 80.2% PBT. Distintas inicializaciones y parámetros de entrenamiento no lograron mejorar los resultados.

I have booked a room with a minibar . FinFrase# I have booked a room with a telephone . FinFrase

would you mind waking me up at three , please ? FinFrase# would you mind waking me up at twenty-seventh , please ? FinFrase

I have made a reservation for Mr Llopis . FinFrase# I have made a reservation for Mr Lidia . FinFrase

could you wake me up at half past two ? FinFrase# could you wake me up at half past eleven ? FinFrase

I want you to wake me up at a quarter to seven . FinFrase# I want you to wake me up at a quarter to twenty-six . FinFrase

my name is César García . FinFrase# my name is Óscar Barberá . FinFrase

I want you to give us the key to our room . FinFrase# I want you to give us the key to the room . FinFrase

9.2.3. La tarea Hansards

Siguiendo la serie de experimentos que se habían realizado para la tarea del Turista completa se realizaron varios experimentos similares con dos capas ocultas con codificaciones binarias al azar de distintos tamaños, en concreto con las codificaciones ya empleadas en el capítulo 8, *H_I_t30_30* , *H_II_t30_30*, *H_III_t60_60* y *H_IV_t240_240*.

Se utilizaron modelos RECONTRA con distinto número de unidades en la primera capa oculta (270, 750 y 900 unidades) y en la segunda (450 y 900 unidades). Los resultados obtenidos pueden observarse en la Tabla 9.7 y muestran una ligera variación pero no una mejora respecto a los obtenidos con el traductor con una única capa oculta y las mismas características (en Tabla 7.3).

Nombre	|1ª CO|	|2ª CO|	F.A. Mom.	| Pesos |	FBT	PBT
H_I_t30_30	-	450	0.05 0.05	337500	30.23%	32.22%
H_I_t30_30	270	450	0.03 0.03	483300	29.63%	32.48%
H_I_t30_30	750	450	0.03 0.03	1318500	23.42%	29.20%
H_II_t30_30	-	450	0.05 0.01	337500	30.93%	27.87%
H_II_t30_30	270	450	0.05 0.01	483300	32.13%	29.00%
H_III_t60_60	-	900	0.05 0.03	1350000	34.03%	36.70%
H_III_t60_60	540	900	0.05 0.03	1933200	34.43%	34.58%
H_III_t60_60	900	900	0.05 0.03	2646000	32.83%	35.21%
H_IV_t240_240	-	900	0.01 0.03	2970000	0.00%	11.98%
H_IV_t240_240	900	900	0.01 0.03	4590000	30.63%	34.90%

Tabla 9.7. Resultados de traducción (FBT y PBT) con dos capas ocultas de distintos tamaños en RECONTRA (con ventana de entrada de tamaño 9 palabras) y varias codificaciones para la tarea Hansards.

Probablemente este comportamiento se debe a dos factores diferenciados:

1. Se consideran unos tamaños demasiado pequeños para las capas, y como sucedía en algunos casos en la tarea del Turista, el traductor se ve incapaz de aprender.
2. Las redes son tan grandes que se producen problemas de estabilidad, como viene sucediendo con gran parte de la experimentación con Hansards. También en la Tabla 9.7 se puede observar como el tamaño de las redes está creciendo enormemente.

En la Figura 9.3 puede verse la evolución de los porcentajes de PBT para un experimento cuyo entrenamiento se alargó hasta 150 iteraciones, que refuerzan la idea de la presencia de problemas de aprendizaje[58].

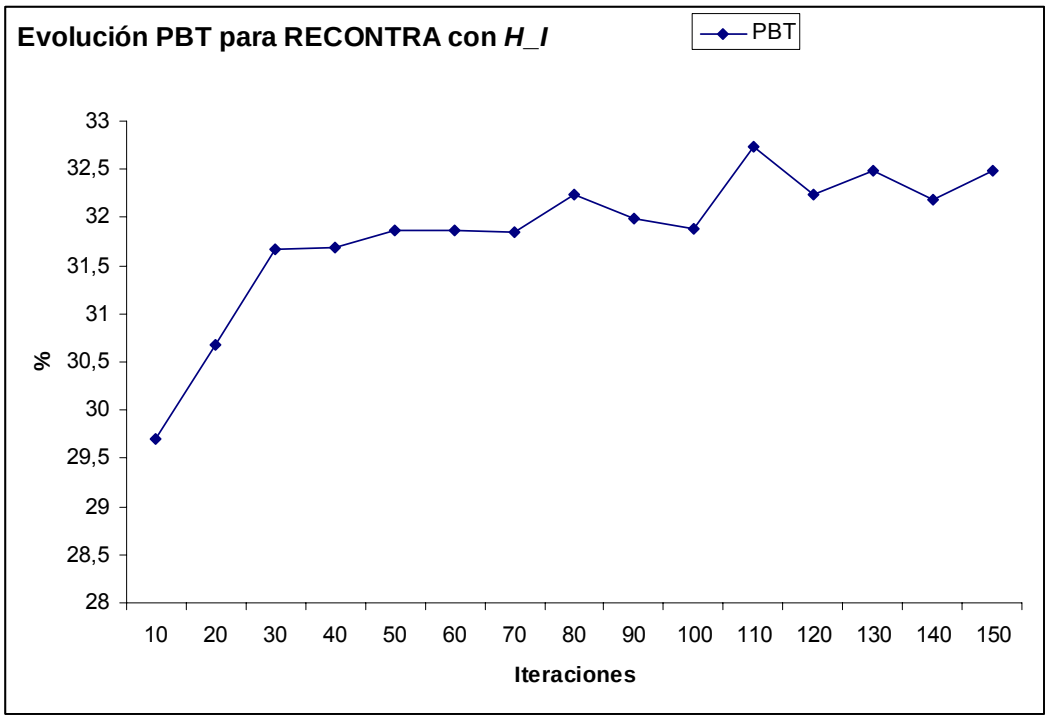

Figura 9.3. Evolución PBT hasta 150 iteraciones para RECONTRA con dos capas ocultas, de 270 y 450 unidades para las codificaciones *H_I_t30_30*.

Se realizaron más experimentos con dos capas ocultas y el tamaño reducido del conjunto de entrenamiento de RECONTRA para las codificaciones *H_I_t30_30*, como puede verse en la Tabla 9.8. Los resultados fueron mucho mejores que los originales (también en la tabla), pero estos resultados son engañosos, dado que se deben a la inestabilidad de la red en el experimento original.

Nombre	1ª CO	2ª CO	Pesos	FBT	PBT
H_I_t30_30	-	450	337500	0.00%	2.37%
H_I_t30_30	270	450	483300	20.42%	28.61%
H_I_t30_30	450	450	742500	21.32%	29.41%

Tabla 9.8. Resultados con dos capas ocultas en RECONTRA (de distintos tamaños) y codificaciones *H_I_t30_30* para la tarea Hansards con conjunto de entrenamiento reducido para el traductor.

Se realizaron diversos experimentos con codificaciones extraídas de MLP entrenados y dos capas ocultas de distintos tamaños en RECONTRA (así como distintos tamaños de la ventana de entrada, 1, 5 y 9). Los resultados de traducción obtenidos pueden observarse en la Tabla 9.9, y en algunos casos muestran ciertas mejoras respecto a los

[58] El aparente considerable ascenso de los porcentajes de traducción se debe más a la escala empleada (28 al 33%) que a una mejora de los resultados.

originales, también mostrados en la tabla. Para hacer posible la lectura de la tabla, el nombre de las codificaciones ha sido sustituido por un número.

Así, *H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1* es llamada 1 en la tabla, *H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1* es llamada 2, *H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1* es 3, *H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1* es 4 y *H_IV_pmv4_r2_ pS_t192_213_bp3000f3_R_bp3000f3_contexto1* es 5 .

Sin embargo, los problemas de estabilidad de las redes de gran tamaño, hacen que estas mejoras sean casi puntuales, y en la mayor parte de los casos, muy reducidas. Se intentaron utilizar factor de aprendizaje y momentum de menor tamaño (0.001), pero en los casos en los que se redujo la inestabilidad, la evolución del aprendizaje fue muy lenta, sin llegar a buenos resultados[59].

Nombre	Entrada	1ª CO	2ª CO	Pesos	FBT	PBT
1	5	-	900	1134000	33.83%	32.90%
1	5	300	900	1414000	33.63%	32.95%
2	1	-	900	1242000	14.62%	31.62%
2	1	900	900	2862000	17.22%	34.73%
2	5	-	900	2106000	34.94%	42.49%
2	5	900	900	2862000	35.34%	40.23%
2	5	1200	2400	12096000	34.63%	39.57%
3	9	-	2160	9849600	35.84%	40.71%
3	9	2160	2160	19180800	33.13%	41.21%
3	5	-	900	2106000	0.00%	40.91%
3	5	900	900	3726000	0.00%	16.41%
4	9	-	900	2556900	35.14%	43.80%
4	9	900	900	3366900	34.84%	41.99%
5	9	-	900	2556900	35.04%	43.47%
5	9	900	900	3366900	31.03%	43.16%

Tabla 9.10. Resultados con dos capas ocultas en RECONTRA (de distintos tamaños) y codificaciones para modelos RECONTRA con distintas ventanas de entrada.

Por otra parte señalar que otra vez se produce el fenómeno del gran número de frases completas bien traducidas, debido al número de repeticiones y el pequeño tamaño estas frases.

9.2.4. Conclusiones

Al incorporar una nueva capa oculta en RECONTRA, la primera capa oculta desarrolla representaciones de la entrada específicas al problema, en muchos casos ahorrando tiempo de entrenamiento (al prescindir de un MLP) y mejorando los resultados de traducción.

[59] Por ejemplo, con codificaciones *H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1* y modelos RECONTRA con ventana de entrada de tamaño 5 y capas ocultas de tamaño 300 y 900 unidades respectivamente, para factor de aprendizaje 0.001 y momentum 0.001 se lograron resultados en torno al 15% de PBT.

Esta nueva topología permite ahorrar tiempo de entrenamiento (tiempo empleado en entrenar el MLP para el lenguaje origen).

En algunos casos, sin embargo, aparecen dificultades:

1. Experimentalmente se ha comprobado que si una de las capas ocultas es muy pequeña, la labor de la red se complica y empeoran los resultados.
2. Dado que las redes crecen, necesitan más tiempo de entrenamiento.
3. Al crecer presentan problemas de estabilidad, especialmente con la tarea Hansards.
4. Además, esta "nueva" red sólo resuelve parcialmente el problema de la creación de codificaciones para RECONTRA, mientras que la codificación de entrada pierde importancia, la de salida no.

Es importante señalar que la red continua dependiendo de la codificación de salida, la cual se convierte en el principal factor en los resultados obtenidos.

Esta "nueva" red resuelve parcialmente el problema de la creación de codificaciones para RECONTRA. Los resultados experimentales muestran claramente que dos capas ocultas proporcionan mejores resultados que una (o tres); la primera capa desarrolla una representación de la entrada (el vocabulario fuente) más adecuada para la tarea.

Utilizar codificaciones locales a la entrada parece aportar unos resultados ligeramente mejores. Sin embargo la inestabilidad y el tiempo de entrenamiento se incrementan considerablemente, con lo que su uso es problemático.

Hay varios problemas a solucionar en el futuro:

1. La dependencia en la codificación de salida (que el traductor aún no aprende).
2. El incremento de tamaño de la red resultante, con los problemas de inestabilidad que esto supone.
3. Decidir el tamaño de la primera capa oculta, dado que un tamaño demasiado pequeño empeora claramente los resultados, pero un tamaño grande provoca inestabilidad.

Es importante recalcar que la mejora en los resultados de traducción no se produce simplemente porque añadir una nueva capa oculta mejore la capacidad de la red. Dejando de lado que teóricamente se conoce que la capacidad de la red no aumenta al tener más capas ocultas, experimentalmente se ha comprobado como al añadir dos nuevas capas ocultas (hasta un total de tres) los resultados no mejoran.

Se han intentado aplicar MLP a la tarea y se ha demostrado una vez más que un MLP multicapa no es el tipo de red adecuada para abordar una tarea de traducción. Sin embargo, la sencilla modificación que Bengio utiliza para otras tareas de tratamiento del lenguaje natural, presenta resultados mejores que el MLP "clásico" aunque muy lejos de algo que inspire a continuar la investigación por ese camino.

Por otra parte, se ha explorado la posibilidad de utilizar realimentación de la capa de salida, con la idea, sino de creación de representaciones de las palabras de salida, si por lo

menos, de que la red cree un modelo del lenguaje. Los resultados sugieren que no se logra.

Los problemas con la tarea Hansards continúan presentes: la inestabilidad de las redes más grandes hace que sea complicado establecer si RECONTRA con dos capas ocultas logra mejorar los resultados de traducción.

Capítulo 10.

RECONTRA extendido

Resumen: En este capítulo se amplia la topología de los modelos RECONTRA intentando mejorar los resultados de traducción. Así, se estudian los efectos de utilizar modelos con ventanas de salida, con ventanas de salida y realimentación de esta, y ventana de salida y varias capas ocultas.

10.1. Introducción

Como parte de la búsqueda de mejores resultados de traducción se volvió a modificar la topología básica de RECONTRA de varias formas: con ventanas de salida, con ventanas de salida y realimentación de esta, y con ventanas de salida y varias capas ocultas.

En el apartado 10.2, en primer lugar se amplia la salida de la red para incluir una ventana de retrasos, de forma similar a la entrada (como ya se comentó en el capítulo 4). De esta forma en cada instante la red produce como salida un fragmento de la frase destino.

Un ejemplo de la topología resultante ha podido verse en la Figura 4.3 y un ejemplo de la entrada y la salida para el par de frases "*Voy a marcharme hoy por la tarde . # I am leaving today in the afternoon .*" puede verse en la Tabla 10.1 (con formato de ventana de entrada $x_{i-2}\ x_{i-1}\ x_i\ x_{i+1}\ x_{i+2}$, y formato de ventana de salida $y_{j-1}\ y_j$, siendo y_j la palabra de salida esperada para x_i).

Incluir una ventana de salida en el traductor tiene dos ventajas:

1. Fuerza al traductor a reconocer sus salidas previas y tener en cuenta todas las palabras en la entrada.
2. Esta forma de producir la frase traducida es más similar a como trabaja un traductor humano, el cual no traduce palabra a palabra, sino un párrafo o frase a la vez. Cuando se ve forzado a trabajar con conocimiento incompleto, ante la aparición de nuevo conocimiento, el traductor o intérprete tiende a volver atrás y cambiar la traducción de palabras individuales para mejorar la comprensión global. Además, el uso de varias palabras de salida en lugar de una sola está

relacionado con los métodos de traducción *phrase-based* (mencionados en el capítulo 2).

Entrada x_{i-2} x_{i-1} x_i x_{i+1} x_{i+2},					**Salida y_{j-1} y_j**	
@	@	Voy	a	marcharme	@	I
@	Voy	a	marcharme	hoy	I	am
Voy	a	marcharme	hoy	por	Am	leaving
a	marcharme	hoy	por	la	Leaving	today
marcharme	hoy	por	la	tarde	Today	in

Tabla 10.1. Ejemplo para el par de frases: *Voy a marcharme hoy por la tarde . # I am leaving today in the afternoon .* con formato de ventana de entrada x_{i-2} x_{i-1} x_i x_{i+1} x_{i+2}, y formato de ventana de salida y_{j-1} y_j, siendo y_j la palabra de salida esperada para la palabra x_i.

Experimentalmente se exploraron diversos tamaños en la ventana de salida: 2, 3 y 4 palabras. Para esto se seleccionaron una serie de codificaciones que se habían utilizado en experimentos anteriores.

Posteriormente, en el apartado 10.3, se utilizaron combinaciones de varias capas ocultas (ya empleadas en el capítulo 9) y ventanas de salida. En la figura 10.1 se puede ver un ejemplo de las redes empleadas, con dos capas ocultas y dos palabras de salida.

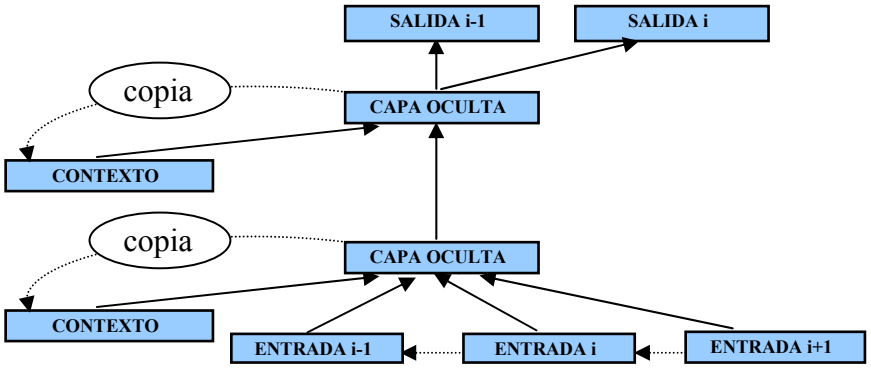

Figura 10.1. RECONTRA con ventana de salida (de dos palabras de tamaño) y dos capas ocultas.

También se ha explorado experimentalmente como interpretar las salidas de la red. Los resultados obtenidos pueden observarse dentro del anexo A.4 en la tabla A.12, en el subapartado "Integración de multiples salidas de una red" del anexo A.

Esta investigación ha dado lugar a las siguientes publicaciones [Casañ, 2008a] y [Casañ, 2009].

10.2. Ventana de salida

En este apartado se presentan los experimentos realizado con redes con varias palabras de salida. Este apartado, a su vez, se divide en tres subapartados según el número de palabras de salida de la red: 2, 3 y 4 palabras. Se muestran tanto los resultados originales con el modelo RECONTRA básico como los obtenidos con los nuevos modelos.

Dado que en este momento no aplicamos ninguna técnica para integrar o interpretar la salida, los resultados de traducción se presentan como si cada palabra de salida perteneciese a una red RECONTRA distinta.

10.2.1. La tarea del Turista

Se realizaron una serie de experimentos con la tarea del Turista, diversas codificaciones y varios tamaños de la ventana de salida del traductor: 2, 3 y 4 palabras. En las tablas, los resultados se muestran como si cada palabra de salida procediera de una red distinta.

10.2.1.1. Dos palabras de salida

Se realizaron experimentos con modelos RECONTRA con dos palabras de salida. Como podemos observar al ver los resultados en la Tabla 10.2 (para PBT) y compararlos con los obtenidos con los originales con una única palabra de salida (también en la tabla), se produjo una mejora considerable en los resultados de traducción. Es importante hacer notar que una de las palabras de salida consistentemente produce mejores resultados que la otra.

En la tabla también se pueden observar el tamaño de cada red y como este se incremente respecto a los experimentos originales con una única palabra de salida.

Nombre	Tamaño	2ª	1ª
T_I_t30_30	337500	-	63.4%
T_I_t30_30	351000	70.5%	75.8%
T_I_pmv4_r2_t30_30_bp3000	337500	-	76.5%
T_I_pmv4_r2_t30_30_bp3000	351000	79.7%	81.1%
T_I_pmv4_r2_pS_t17_14_bp3000	277650	-	52.5%
T_I_pmv4_r2_pS_t17_14_bp3000	283950	62.5%	67.5%
T_IV_pmv4_r2_pS_t49_46_bp3000	421650	-	79.7%
T_I_pmv4_r2_pS_t17_14_bp3000	442350	81.3%	83.4%
T_V_pmv6_r4_pS_t121_94_bp3000	734850	-	78.2%
T_V_pmv6_r4_pS_t121_94_bp3000	777150	72.9%	83.1%
T_IV_pmv6_r4_pS_t121_94_bp100_scg600_600_r_bp3000	734850	-	83.7%
T_IV_pmv6_r4_pS_t121_94_bp100_scg600_600_r_bp3000	777150	84.1%	85.8%
T_I_pmv4_r2_pUNC_t21_21_bp3000	297000	-	71.9%
T_I_pmv4_r2_pUNC_t21_21_bp3000	306450	77.4%	79.2%

Tabla 10.2. Resultados de traducción (PBT) con RECONTRA básico y extendido con dos palabras de salida (1ª y 2ª en la tabla), 9 palabras de entrada y 450 unidades en la capa oculta para la tarea del Turista.

10.2.1.2. Tres palabras de salida

Dados los buenos resultados obtenidos para la tarea del Turista, añadir una nueva palabra de salida era el paso lógico. Observando los resultados (en la Tabla 10.3), sin embargo, podemos observar que la mejora en los resultados no fue tan grande como al realizar el paso de una a dos palabras de salida.

Parte del problema parece provenir del incremento en tamaño de las redes y los problemas de estabilidad que aparecen.

Dado que una forma sencilla de reducir los problemas de estabilidad puede ser utilizar parámetros más pequeños, se realizó en algunos experimentos con cuatro palabras de salida (en los cuales se esperaban problemas mayores) con relativo éxito (Tabla 10.4).

Nombre	Tamaño	3ª	2ª	1ª
T_I_t30_30	364500	69.9%	75.5%	77.6%
T_I_pmv4_r2_t30_30_bp3000	364500	80.3%	82.5%	82.8%
T_I_pmv4_r2_pS_t17_14_bp3000	290250	64.1%	70.1%	73.0%
T_IV_pmv4_r2_pS_t49_46_bp3000	463050	80.6%	83.6%	84.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg 600_600	616950	51.2%	72.5%	82.8%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600 _r_bp3000	616950	80.0%	83.5%	84.6%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600 _r_bp3000	616950	81.8%	84.7%	85.7%

Tabla 10.3. Resultados de traducción con RECONTRA extendido con tres palabras de salida (1ª, 2ª y 3ª en la tabla) y varias codificaciones.

10.2.1.3. Cuatro palabras de salida

Finalmente, se probaron modelos RECONTRA con cuatro palabras de salida y en algunos casos los resultados de traducción aún mejoraron (en Tabla 10.4), pero en este punto los problemas de estabilidad se hicieron generales, especialmente para las codificaciones de mayor tamaño (*T_V_pmv6_r4_pS_t121_94_bp_scg* y *T_V_pmv6_r4_pS_t121_94_bp_scg_R_bp3000*), además de resultar su tiempo de entrenamiento muy largo.

Teniendo también en cuenta los problemas de asociar fragmentos de texto tan largos entre el lenguage fuente y el destino no se continuó aumentando el tamaño de la ventana de salida.

Nombre	FA	Mom	Tamaño	4ª	3ª	2ª	1ª
T_I_t30_30	0.1	0.1	378000	69.3%	77.2%	78.1%	79.3%
T_I_pmv4_r2_t30_30_bp3000	0.1	0.1	378000	70.6%	82.2%	82.8%	82.6%
T_I_pmv4_r2_pS_t17_14_bp3000	0.1	0.1	296550	64.3%	70.4%	72.7%	74.1%
T_IV_pmv4_r2_pS_t49_46_bp3000	0.1	0.3	483750	80.4%	84.1%	83.4%	84.8%
T_IV_pmv4_r2_pS_t49_46_bp3000	0.07	0.09	483750	80.0%	83.2%	84.0%	84.7%
T_V_pmv6_r4_pS_t121_94_bp100_ scg600_600	0.1	0.1	659250	56.3%	69.3%	62.5%	74.7%
T_V_pmv6_r4_pS_t121_94_bp100_ scg600_600	0.05	0.01	659250	60.1%	76.9%	80.4%	82.0%
T_V_pmv6_r4_pS_t121_94_bp100_ scg600_600_R_bp3000	0.1	0.1	659250	76.6%	81.3%	83.4%	84.3%

Tabla 10.4. Resultados de traducción para RECONTRA extendido con cuatro palabras de salida (1ª, 2ª, 3ª y 4ª en la tabla) y varias codificaciones.

10.2.2. La tarea Hansards

Como era de esperar, también se realizaron experimentos para la tarea Hansards con varias palabras de salida en el traductor y distintas codificaciones.

10.2.2.1. Dos palabras de salida

Inicialmente se realizaron pruebas con modelos RECONTRA con dos palabras de salida. Sin embargo, como podemos observar en los resultados de la Tabla 10.5 (para PBT), no se produjo una mejora consistente en los resultados de traducción.

A veces se mejoraban los resultados ligeramente, sobre un 2% de PBT y a veces empeoraban (en porcentajes similares)[60]. La razón parece clara: las redes son demasiado inestables, por su gran tamaño y la complejidad de la función del error.

Esto llevó a que no se realizasen experimentos con un mayor número de palabras de salida y esta tarea, dado que no se esperaba una mejora en los resultados.

Nombre	|Entrada|	|CO|	Tamaño	2ª	1ª
H_II_t30_30	5	450	283500	-	26.47%
H_II_t30_30	5	450	297000	25.75%	26.72%
H_II_t30_30	5	900	972000	-	0.03%
H_II_t30_30	5	900	999000	23.84%	24.29%
H_III_pmv4_r2_t60_60_bp3000f3_contexto1	5	900	1134000	-	33.23%
H_III_pmv4_r2_t60_60_bp3000f3_contexto1	5	900	1188000	31.56%	32.44%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	1	900	1242000	-	31.62%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	1	900	1458000	28.88%	33.44%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_r_bp3000f3_contexto1	9	900	2556900	-	42.32%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_r_bp3000f3_contexto1	9	900	3186000	44.23%	44.58%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	9	900	2556900	-	43.96%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	9	900	3186000	40.55%	41.35%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	5	900	2106000	-	44.87%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	5	900	2106000	41.52%	42.20%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_contexto1	1	900	1174500	-	31.20%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_contexto1	1	900	1366200	25.37%	27.15%

Tabla 10.5. Resultados de traducción (PBT) con RECONTRA básico y extendido con dos palabras de salida (1ª y 2ª en la tabla) para la tarea Hansards, diversas ventanas de entrada (1, 5 o 9 palabras), métodos de entrenamiento y diversas unidades en la capa oculta.

Se realizaron varios intentos de mejorar los resultados con técnicas que ya habían proporcionado buenos resultados en anteriores experimentos:

1. Por un lado aumentar el tamaño de la capa oculta para aumentar la capacidad de la red.
2. Y por otro utilizar métodos de evolución de los parámetros de entrenamiento para acelerar el aprendizaje y reducir la inestabilidad.

Los resultados pueden verse en la Tabla 10.6 y no muestran un comportamiento claro.

[60] Excepto en algún caso concreto en el que la inestabilidad de las redes había llevado a obtener resultados muy malos.

Nombre	|Entrada|	|CO|	Método	2ª	1ª
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_contexto1	1	900	BP	-	31.62%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_contexto1	1	900	BP	28.88%	33.44%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_contexto1	1	900	(6.19)	30.03%	33.71%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_r_bp3000f3_contexto1	9	900	BP	-	42.32%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_r_bp3000f3_contexto1	9	900	BP	41.07%	41.32%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_r_bp3000f3_contexto1	9	900	(6.19)	41.58%	41.19%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_r_bp3000f3_contexto1	9	2160	BP	-	43.39%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_r_bp3000f3_contexto1	9	2160	BP	44.13%	44.48%
H_IV_pmv4_r2_t240_240_bp3000f3 _r_bp3000f3_contexto1	9	900	BP	-	43.96%
H_IV_pmv4_r2_t240_240_bp3000f3 _r_bp3000f3_contexto1	9	900	BP	40.55%	41.35%
H_IV_pmv4_r2_t240_240_bp3000f3 _r_bp3000f3_contexto1	9	900	(6.19)	41.56%	41.39%
H_IV_pmv4_r2_t240_240_bp3000f3 _r_bp3000f3_contexto1	9	2160	BP	-	5.72%
H_IV_pmv4_r2_t240_240_bp3000f3 _r_bp3000f3_contexto1	9	2160	BP	40.41%	40.15%

Tabla 10.6. Resultados de traducción (PBT) para cada palabra de salida con RECONTRA básico y extendido con dos palabras de salida (1ª y 2ª en la tabla) para la tarea Hansards y diversas codificaciones, métodos de entrenamiento Retropropagación y Retropropagación con (6.19) y distinto número de unidades en la capa oculta (900 o 2160).

10.2.3. Conclusiones

Las distintas versiones de RECONTRA con ventana de salida deslizante (con dos, tres y cuatro palabras de salida) que hemos presentado en esta experimentación presentan dos comportamientos diferenciados según las tareas. Así, para Turista suponen siempre una mejora en los resultados, por regla general considerable. Para Hansards, los resultados empeoran o mejoran ligeramente (entre 1 y 2%), lo cual parece deberse más a la inestabilidad de las redes que a cualquier otro factor.

Señalar que para la tarea Hansards la causa de la inestabilidad no es únicamente el tamaño de las redes (hay algunas redes de tamaños similares para la tarea Turista que tienen un comportamiento más estable y dan buenos resultados) sino la mayor complejidad de la tarea.

Por supuesto, estas modificaciones no eliminan todos los problemas que se presentan al intentar abordar tareas de traducción e incluso generan algunos, especialmente evidentes en la tarea Hansards.

No se ha proseguido el aumento en el número de palabras en la capa de salida dado que la mejora parece reducirse progresivamente y empiezan a aparecer comportamientos

extraños en las redes, asociados a la inestabilidad. Esto se puede observar en la Tabla 10.7 para las codificaciones *T_I_pmv4_r2_t30_30_bp3000* para la tarea Turista con distintas ventanas de salida.

Nombre	Tamaño	4ª	3ª	2ª	1ª
T_I_pmv4_r2_t30_30_bp3000	337500	-	-	-	76.5%
T_I_pmv4_r2_t30_30_bp3000	351000	-	-	79.7%	81.1%
T_I_pmv4_r2_t30_30_bp3000	364500	-	80.3%	82.5%	82.8%
T_I_pmv4_r2_t30_30_bp3000	378000	70.6%	82.2%	82.8%	82.6%

Tabla 10.7. Resultados de traducción para RECONTRA extendido con distinto número de palabras de salida (1ª, 2ª, 3ª y 4ª en la tabla) y codificaciones T_I_pmv4_r2_t30_30_bp3000.

Por otra parte se confirman las conclusiones de capítulos anteriores: gran parte de los problemas que se presentan vienen dados por las limitaciones del algoritmo de entrenamiento, hay que crear/utilizar uno más adecuado: más estable y más rápido.

También mencionar que se realizaron algunos experimentos con realimentación de la capa salida, pero los resultados no supusieron una mejora respecto a las redes sin realimentación (ver tabla A.17).

10.3. Varias capas ocultas y ventana de salida

Una vez observados los buenos resultados que se producían cuando utilizábamos dos capas ocultas (apartado 9.3) y una ventana de salida (apartado 10.2) por separado, el paso siguiente consistía claramente en combinarlos. Así, tendremos modelos RECONTRA con dos capas ocultas y varias palabras de salida. Como veremos experimentalmente, los problemas de estabilidad continuaban apareciendo y el tiempo de entrenamiento volvía a aumentar, pero los resultados mejoraron ligeramente otra vez.

10.3.1. La tarea del Turista

Se exploraron diversas combinaciones de codificaciones y de redes, siempre con dos capas ocultas pero con distinto número de unidades en cada capa y distinto número de palabras de salida (como en los experimentos del apartado 10.2, para dos, tres y cuatro palabras). Los resultados pueden observarse en las Tablas 10.8 (dos palabras), 10.9 (tres palabras) y 10.10 (cuatro palabras), y en general muestran mejores respecto a los experimentos originales, sólo con dos capas ocultas o sólo con varias palabras de salida.

Nombre	|1ª CO|	Tamaño	2ª	1ª
T_I_t30_30	270	496800	75.8%	79.4%
T_I_t30_30	450	756000	76.7%	80.4%
T_I_pmv4_r2_t30_30_bp3000	270	496800	82.3%	83.8%
T_I_pmv4_r2_t30_30_bp3000	450	756000	83.3%	84.5%
T_I_pmv4_r2_pS_t17_14_bp3000	270	450810	69.1%	74.0%
T_I_pmv4_r2_pS_t17_14_bp3000	450	688950	67.4%	71.6%
T_IV_pmv4_r2_pS_t49_46_bp3000	270	557370	83.1%	84.8%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	270	775530	79.6%	85.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_R_bp3000	270	775530	83.9%	86.0%

Tabla 10.8. Resultados de traducción (PBT) para redes RECONTRA con dos capas ocultas de tamaños diversos (la segunda siempre 450 unidades) y dos palabras de salida (1ª y 2ª en la tabla) para diversas codificaciones.

Nombre	Tamaño	3ª	2ª	1ª
T_I_t30_30	510300	75.4%	80.0%	81.3%
T_I_pmv4_r2_t30_30_bp3000	510300	81.4%	83.5%	83.8%
T_I_pmv4_r2_pS_t17_14_bp3000	457110	67.5%	72.5%	75.1%
T_IV_pmv4_r2_pS_t49_46_bp3000	578070	82.9%	85.4%	86.4%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	817830	74.0%	79.5%	84.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_R_bp3000	817830	82.2%	84.6%	85.4%

Tabla 10.9. Resultados de traducción (PBT) para redes RECONTRA con dos capas ocultas de tamaños 270 y 450 unidades respectivamente y tres palabras de salida (1ª, 2ª y 3ª en la tabla) para diversas codificaciones.

Es interesante señalar que al incrementar el número de palabras de salida o el tamaño de las capas ocultas se puede observar una clara tendencia a la mejora de los resultados. Sin embargo, con cuatro palabras de salida los resultados (Tabla 10.10) empeoran excepto en el caso de las codificaciones más pequeñas (codificaciones *T_I_pmv4_r2_pS_t17_14_bp3000*). Como en otros casos parece deberse a los problemas de inestabilidad que aparecen al crecer las redes.

Nombre	Tamaño	4ª	3ª	2ª	1ª
T_I_t30_30	523800	74.1%	78.5%	80.0%	81.2%
T_I_pmv4_r2_t30_30_bp3000	523800	80.9%	82.8%	83.3%	83.6%
T_I_pmv4_r2_pS_t17_14_bp3000	463410	69.0%	74.0%	76.5%	78.1%
T_IV_pmv4_r2_pS_t49_46_bp3000	598770	82.1%	84.2%	85.2%	85.4%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	860130	63.6%	75.7%	77.8%	79.0%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_R_bp3000	860130	80.3%	83.8%	85.4%	85.7%

Tabla 10.10. Resultados de traducción (PBT) para redes RECONTRA con dos capas ocultas de tamaños 270 y 450 unidades respectivamente y cuatro palabras de salida (1ª, 2ª, 3ª y 4ª en la tabla) para diversas codificaciones.

De forma puntual se realizaron dos experimentos para modelos RECONTRA con dos capas ocultas, dos palabras de salida y codificación de entrada *T_Local*, dado que los experimentos originales con esta combinación de codificaciones y diversas redes habían

ofrecido muy buenos resultados. En la Tabla 10.11 se pueden observar los resultados obtenidos junto con los de los experimentos originales con una única palabra de salida.

Los resultados obtenidos son claramente mejores (en realidad los mejores obtenidos para la tarea), pero si se observa el tamaño de las redes (también en la tabla) se puede ver claramente como este ha aumentado exageradamente de tamaño.

Tamaño	|1ª CO|	|2ª CO|	2ª	1ª
2994300	-	450	-	86.6%
2077380	270	450	-	86.7%
2158380	450	450	-	88.8%
2090880	270	450	87.7%	88.7%
3075300	450	450	89.0%	89.9%

Tabla 10.11. Resultados de traducción (PBT) para redes RECONTRA con una y dos capas ocultas con una palabra de salida y dos capas ocultas de tamaños 270 y 450 unidades la primera capa y 450 unidades la segunda y dos palabras de salida (1ª y 2ª en la tabla) con codificación de entrada *T_Local* y codicación de salida *T_I_pmv4_r2_t30_30_bp3000*.

Los resultados sugieren que podría existir algún tipo de límite máximo dado por ciertas palabras o frases problemáticas. En la Tabla 10.12 se pueden observar ejemplos de frases mal traducidas con modelos RECONTRA con dos palabras de salida y dos capas ocultas (tamaños 270 y 450 unidades) y codificación *T_Local* a la entrada y *T_I_pmv4_r2_t30_30_bp3000* a la salida, que sugieren que las redes tienen problemas para diferenciar entre determinadas palabras con características comunes. En concreto los nombres propios y los números.

Traducción correcta	Traducción producida
I have a reservation for a quiet , single room with a good view , a bath and a telephone for C'esar Borillo . FinFrase	I have a reservation for a quiet , single room with a good view *and* a *and for* a *shower* for *V'azquez*
I have booked a room for twenty days . FinFrase	I have booked a room for *twenty-first* days . FinFrase
could you give me the keys to our room , please ? FinFrase	could you *wake* me *we* keys to our room , please ? FinFrase
would you mind sending down my suitcase to room number four five four , please ? FinFrase	would you mind sending down my suitcase to room number four *four* four , please ? FinFrase
I made a reservation for Asunci'on Rivera . FinFrase	I made a reservation for *Rosario Mira* . FinFrase
I leave on June the fourteenth at a quarter past one in the afternoon . FinFrase	I leave on June the *station* at a quarter past one in the afternoon . FinFrase
a mistake has been made in our bill for room number four one eight . FinFrase	a mistake has been made in our bill for room number *one* one . FinFrase
could you give me the keys to room number seven one two ? FinFrase	could you give me the keys to room number *five* one *five* ? FinFrase

Tabla 10.12. Ejemplos de frases mal traducidos para un modelo RECONTRA con dos capas ocultas con dos capas ocultas de tamaños 270 y 450 unidades cada capa y codificación de entrada Local y codicación de salida *T_I_pmv4_r2_t30_30_bp3000*. Las palabras mal traducidas aparecen sombreadas.

10.3.2. Conclusiones

Para la tarea del Turista añadir una capa oculta y varias palabras de salida ha demostrado una mejora respecto a los resultados de traducción obtenidos con las mismas codificaciones y el modelo RECONTRA básico, así como a las dos modificaciones por separado.

En algunos casos, las nuevas redes aumentan tanto de tamaño que presentan dificultades de entrenamiento. Este problema se ve exacervado para la tarea Hansards. Por contra, en otros, con codificaciones de pequeño tamaño, compensa utilizar estas redes: los resultados son mejores con un tamaño de red y problemas de entrenamiento reducidos.

Al utilizar dos capas ocultas, se le facilita a la red la creación de su propia representación de las codificaciones de entrada. Esto provoca un aumento de los resultados, al generarse mejores representaciones de las que le podemos proporcionar a la red. Por otra parte se disminuye enormemente la necesidad de obtener una buena codificación para los vocabularios de entrada, dado que la red compensa los posibles problemas que pueda tener la codificación.

Con la tarea Hansards se han realizado algunos experimentos que sólo han destacado los problemas de estabilidad presentes en estas redes tan grandes.

Se realizaron algunos experimentos con una variante de la topología de RECONTRA en la que la capa de salida se realimentaba a sí misma. Esta variante ya había sido sugerida por Elman ([Elman, 1990]) como alternativa a la utilización de ventanas de salida. Esta realimentación permitiría a la red tener en cuenta sus salidas anteriores y en teoría mejorar su capacidad para resolver este tipo de tareas, en las que existe un orden en la secuencia de salida. En la figura A.2 del anexo A se puede ver la topología de esta red para una ventana de salida de dos palabras.

Sin embargo, los resultados obtenidos (en Tabla A.20) fueron peores que los logrados sin realimentación y pueden observarse en la subsección “Realimentación de la capa de salida” del anexo A.3.

Capítulo 11.

Ensembles de RECONTRA

Resumen: En este capítulo se presentan varios nuevos sistemas de traducción basados en sistemas expertos y ensembles de modelos RECONTRA. Se exploran varias posibilidades de selección y combinación de las salidas de las redes, basadas en reglas formando así un sistema experto y en métodos de votación y medidas de distancias.

11.1. Introducción

Como ya se ha comentado en el apartado 4.3 una posible forma de obtener mejores prestaciones con ANNs es mediante la combinación de varias redes distintas en una sola red. Esta combinación puede realizarse de muchas formas y en este capítulo exploramos algunas alternativas.

En primer lugar, se muestran los resultados de un sistema experto, cuya estructura ya fue presentada en el apartado 4.3.1. Así, los dos componentes del sistema tienen igual topología pero son entrenados con dos subconjuntos de pares de frases distintos: los pares de frases interrogativos (aquellos en los cuales aparece un interrogante, '¿' en la frase de entrada) y los no-interrogativos (aquellos en los que no aparece).

A continuación, se presentan los resultados que se obtuvieron en una serie de experimentos con ensembles de 2, 3, 4 y 5 componentes y seis criterios de decisión:

- **Votación** para ensembles de 3 o más componentes. Así, se empleó una simple regla de la votación de la mayoría simple para la selección de cada palabra.
- Criterios básicos de **distancia**: los modelos RECONTRA producen una salida real que interpretamos como la palabra con la codificación más próxima del vocabulario. En el apartado 4.3.2 se presentaron los criterios que se han utilizado en la experimentación de este capítulo, reflejados en las ecuaciones (4.2), (4.3) y (4.5).

- **Votación más distancia**: cuando falla la regla del votación de la mayoría simple, se aplica una de las formulas de selección de palabras basada en distancias, (4.2) o (4.3). En las tablas estos métodos se denominan *V1* y *V2* respectivamente.

Recordemos que la idea básica es que distancia entre estos valores (el producido por la red y el de la codificación más próxima) puede interpretarse como una medida de la confianza que la red tiene en la salida. La forma más sencilla de utilizar este criterio es comparar las distancias obtenidas con las distintas salidas y quedarnos con la red (palabra) cuya distancia sea mínima según la ecuación (4.1). Así, eliminaríamos las palabras de las cuales las redes no se sienten "seguras".

Sin embargo, si las codificaciones tienen distinto tamaño para poder compararlas es necesario normalizarlas: así se divide esta distancia entre el número de unidades/bits que se utilizan para cada codificación (ecuación (4.2)):

$$\text{distp}_i = \min\left(\frac{\text{disp}_1}{t_1}, \frac{\text{disp}_2}{t_2}, \ldots, \frac{\text{disp}_m}{t_m}\right)$$

En el siguiente ejemplo pueden observarse las palabras y distancias para dos redes distintas, la primera con codificación de tamaño 49, y la segunda de tamaño 30 para la frase en inglés "*please wake us up tomorrow at a quarter past seven .*".

please 1.767642 wake 2.167500 us 1.598883 up 1.374740 tomorrow 0.721543 one 1.052814 a 0.558218 quarter 1.886124 past 1.379472 for 1.420147 . 3.766006 FinFrase 15.895749

please 1.402408 wake 1.167278 us 0.895170 up 0.749546 tomorrow 1.090379 at 0.747640 a 0.709439 quarter 1.320870 past 1.029576 seven 2.928242 . 4.052036 FinFrase 3.926232

La salida que este criterio produciría sería la siguiente:

please 1.402408 wake 1.167278 us 0.895170 up 0.749546 tomorrow 0.721543 at 0.747640 a 0.558218 quarter 1.320870 past 1.029576 for 1.420147 . 3.766006 FinFrase 3.926232

En este ejemplo se pueden observar dos de las características principales de este método:

- En muchos casos no importa que palabra seleccionamos puesto que las dos redes proporcionan la misma traducción.
- Se tenderá siempre a seleccionar la palabra con codificación menor si no se normalizan las distancias.

A continuación puede observarse el resultado de normaliza las distancias para el ejemplo anterior:

please 0.036074 wake 0.044235 us 0.032630 up 0.028056 tomorrow 0.014725 one 0.021486 a 0.011392 quarter 0.038492 past 0.028152 for 0.028983 . 0.076857 FinFrase 0.324403

please 0.046747 wake 0.038909 us 0.029839 up 0.024985 tomorrow 0.036346 at 0.024921 a 0.023648 quarter 0.044029 past 0.034319 seven 0.097608 . 0.135068 FinFrase 0.130874

Se empleó también una variante de este sistema, en la cual, en lugar de comparar las distancias palabra a palabra, se comparaban las distancias acumuladas normalizadas (en función del tamaño de las codificaciones) hasta la palabra actual *j* ecuación (4.3).

$$distp_j = \min\left(\sum_{i=0}^{j} disp_{1i}, \sum_{i=0}^{j} disp_{2i}, \ldots \sum_{i=0}^{j} disp_{mi}\right)$$

Es una forma de dar mayor importancia a los componentes que han proporcionado mejores resultados anteriormente.

Para la selección de frases completas se utilizó una variante de acumulación de distancias normalizadas ecuación (4.5), en la cual se tenía en cuenta la aparición de *FinFrase* en cada red.

$$distfs = \min\left(\frac{\sum_{i=0}^{pos_{1f}} disp_{1i}}{pos_{1f}}, \frac{\sum_{i=0}^{pos_{2f}} disp_{2i}}{pos_{2f}}, \ldots \frac{\sum_{i=0}^{pos_{mf}} disp_{mi}}{pos_{mf}}\right),$$

$$pos_{jf} = FinFrase \quad (j = 1, 2, \ldots m)$$

Intuitivamente, con la ecuación (4.5) seleccionamos todas las palabras de una misma red, mientras que con las ecuaciones (4.2) y (4.3) seleccionamos palabra a palabra de redes potencialmente distintas.

En la experimentación presentada en el punto 11.4 se combinó la división de la tarea en frases interrogativas y no-interrogativas y la utilización de distancias. Así, se exploró la opción de utilizar un criterio de distancia para combinar la salida de varias redes especializadas dentro del mismo subconjunto (frases interrogativas o no-interrogativas).

Parte de esta experimentación ha sido presentada en [Casañ, 2009].

11.2. Sistema Experto: frases Interrogativas/No-Interrogativas

11.2.1. La tarea del Turista

Se creó un ensemble formado por dos redes RECONTRA, una de ellas entrenada con pares de frases interrogativas, y el otro con frases no interrogativas. Se seleccionaron una serie experimentos con redes de Elman con ventana de entrada de tamaño 9 palabras y 450 unidades en la capa oculta, que utilizaban codificaciones extraídas de MLP entrenados con codificaciones distribuidas al azar, en concreto las codificaciones extraídas de MLP entrenados con retropropagación, codificaciones locales iniciales y podadas.

Recordemos que estas codificaciones, tras la poda de los MLPs, tienen tamaños distintos. Además se han utilizado codificaciones extraídas de MLP entrenadas con codificaciones distribuidas, poda, y combinaciones de métodos de entrenamiento. Para obtener algunas de estas codificaciones se ha empleado realimentación en los MLP.

Los resultados originales y los aportados por el sistema pueden verse en la Tabla 11.1, y muestran una clara mejora del sistema respecto a los obtenidos con un único modelo RECONTRA de las mismas características.

Nombre	Sistema	FBT	PBT
T_L_pmv4_r2_pS_t34_28_bp3000	Ensemble	16.3%	65.4%
T_L_pmv4_r2_pS_t34_28_bp3000	Original	12.0%	62.8%
T_L_pmv6_r4_pS_t24_26_bp3000	Ensemble	14.0%	63.4%
T_L_pmv6_r4_pS_t24_26_bp3000	Original	6.1%	51.6%
T_L_pmv8_r4_pS_t44_43_bp3000	Ensemble	12.2%	70.4%
T_L_pmv8_r4_pS_t44_43_bp3000	Original	5.2%	62.8%
T_L_pmv8_r6_pS_t21_23_bp3000	Ensemble	13.5%	63.4%
T_L_pmv8_r6_pS_t21_23_bp3000	Original	4.4%	49.6%
T_I_pmv4_r2_t30_30_bp3000	Ensemble	27.5%	79.8%
T_I_pmv4_r2_t30_30_bp3000	Original	20.7%	76.5%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	Ensemble	16.2%	77.1%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	Original	7.9%	70.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	Ensemble	30.0%	82.0%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	Original	22.6%	78.2%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	Ensemble	14.3%	70.9%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	Original	5.6%	65.7%
T _V_pmv8_r6_pS_t132_93_bp100_scg600_600	Ensemble	38.6%	84.0%
T _V_pmv8_r6_pS_t132_93_bp100_scg600_600	Original	17.8%	73.8%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_r_bp3000	Ensemble	41.2%	85.5%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_r_bp3000	Original	38.6%	83.9%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_r2_bp3000	Ensemble	40.0%	85.5%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_r2_bp3000	Original	30.1%	80.7%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (salida)	Ensemble	46.1%	88.8%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (salida)	Original	43.8%	86.7%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (2 pal. salida)	Ensemble	47.0%	89.5%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (2 pal. salida)	Original	46.9%	88.6%

Tabla 11.1 Resultados de traducción para redes RECONTRA con diversas codificaciones distribuidas, excepto los experimentos con codificación de entrada *T_Local* y codificación de salida *T_I_pmv4_r2_t30_30_bp3000*, para los que se emplearon dos capas ocultas de 270 y 450 unidades, y en un caso, dos palabras de salida.

Si observamos los resultados divididos por subconjunto o componente (No-interrogativo e Interrogativo) para algunos de los casos, la mejora es especialmente notable en las frases interrogativas. Este fenómeno se debe a que su estructura es menos variable y presentan menos palabras `problemáticas` (no aparecen, por ejemplo, nombres propios y números).

Este método, como ya se ha comentado no es automático, dado que el conocimiento del experto humano ha sido utilizado en la selección de los dos conjuntos de entrenamiento. Sin embargo, existe otras posibilidades de división de la tarea que podrían aportar ventajas. Por ejemplo se podría dividir la tarea en función de la longitud de las frases, lo que permitiría utilizar ventanas de entrada de distintos tamaños, más adecuadas a cada subproblema.

Nombre	Conjuntos	FBT	PBT
T_L_pmv4_r2_pS_t34_28_bp3000	No-interrogativo	7.9%	60.8%
T_L_pmv4_r2_pS_t34_28_bp3000	Interrogativo	28.3%	72.0%
T_L_pmv4_r2_pS_t34_28_bp3000	Ensemble	16.3%	65.4%
T_L_pmv4_r2_pS_t34_28_bp3000	Completo	12.0%	62.8%
T_L_pmv6_r4_pS_t24_26_bp3000	No-interrogativo	7.7%	58.6%
T_L_pmv6_r4_pS_t24_26_bp3000	Interrogativo	23.0%	70.4%
T_L_pmv6_r4_pS_t24_26_bp3000	Ensemble	14.0%	63.4%
T_L_pmv6_r4_pS_t24_26_bp3000	Completo	6.1%	51.6%
T_L_pmv8_r4_pS_t44_43_bp3000	No-interrogativo	6.9%	67.4%
T_L_pmv8_r4_pS_t44_43_bp3000	Interrogativo	19.9%	74.8%
T_L_pmv8_r4_pS_t44_43_bp3000	Ensemble	12.2%	70.4%
T_L_pmv8_r4_pS_t44_43_bp3000	Completo	5.2%	62.8%
T_L_pmv8_r6_pS_t21_23_bp3000	No-interrogativo	9.8%	60.0%
T_L_pmv8_r6_pS_t21_23_bp3000	Interrogativo	18.8%	68.2%
T_L_pmv8_r6_pS_t21_23_bp3000	Ensemble	13.5%	63.4%
T_L_pmv8_r6_pS_t21_23_bp3000	Completo	4.4%	49.6%
T_I_pmv4_r2_t30_30_bp3000	No-interrogativo	20.4%	76.3%
T_I_pmv4_r2_t30_30_bp3000	Interrogativo	35.6%	84.8%
T_I_pmv4_r2_t30_30_bp3000	Ensemble	27.5%	79.8%
T_I_pmv4_r2_t30_30_bp3000	Completo	20.7%	76.5%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	No-interrogativo	12.1%	73.9%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	Interrogativo	19.8%	81.8%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	Ensemble	16.2%	77.1%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	Completo	7.9%	70.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	No-interrogativo	20.3%	77.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	Interrogativo	40.4&	87.3%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	Ensemble	30.0%	82.0%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600	Completo	22.6%	78.2%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_ r_bp3000	No-interrogativo	39.9%	82.0%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_ r_bp3000	Interrogativo	50.5%	90.2%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_ r_bp3000	Ensemble	41.2%	85.5%
T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_ r_bp3000	Completo	38.6%	83.7%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (salida)	No-interrogativo	39.5%	87.1%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (salida)	Interrogativo	55.5%	91.3%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (salida)	Ensemble	46.1%	88.8%
T_Local (entrada) y T_I_pmv4_r2_t30_30_bp3000 (salida)	Completo	43.8%	86.7%

Tabla 11.2 Resultados de traducción para dos redes *RECONTRA* componentes de un sistema de experto *Int/No-Int* para diversas codificaciones distribuidas.

Otro aspecto interesante a señalar es que los componentes no tienen porqué ser iguales. Dado que experimentalmente hemos comprobado las frases no interrogativas son más complejas, utilizar un componente distinto para ellas (una capa oculta mayor, o dos capas ocultas) resulta razonable.

Los resultados obtenidos utilizando redes con un número mayor de neuronas en la capa oculta de RECONTRA con dos capas ocultas (como ya vimos en el apartado 9.2) y con una o dos palabras de salida (explorado en el capítulo 10) pueden verse en la Tabla 11.3 y muestran una clara mejora respecto a los originales. Así en la tabla se recogen las siguientes características: con capas ocultas de varios tamaños (450, 600 y 750), con dos capas ocultas (segunda capa de tamaño 450 y 270, 450 o 750 unidades en la primera capa) y con dos palabras de salida y dos capas ocultas (270 o 450 unidades en la primera capa y 450 en la segunda).

Todos estos experimentos se realizaron con las codificaciones básicas *T_I_pmv4_r2_t30_30_bp3000* y diversas redes: con una o dos capas ocultas de varios tamaños, con una o dos palabras de salida y con dos palabras de salida y dos capas ocultas.

Como es evidente se podrían haber hecho experimentos con más codificaciones y con combinaciones de redes distintas, pero consideramos que estos resultados ya eran bastante representativos.

Para facilitar la comparación, en la Tabla 11.3 se incluyen los resultados obtenidos con los experimentos originales (Conjunto *Completo*) para las mismas codificaciones y tipos de redes. En la Tabla 11.3 también se pueden observar los resultados con el sistema experto original (con RECONTRA con una única capa oculta de 450 unidades).

|1ª CO|	|2ª CO|	|Salida|	Conjunto	FBT	PBT
-	450	1	Completo	20.7%	76.5%
270	450	1	Completo	29.1%	80.9%
450	450	1	Completo	32.5%	82.5%
-	450	2	Completo	30.1%	81.1%
270	450	2	Completo	32.4%	83.8%
450	450	2	Completo	35.0%	84.5%
-	450	1	Interrogativo	35.6%	84.8%
-	450	1	No- interrogativo	20.4%	76.3%
-	550	1	Interrogativo	36.9%	84.7%
-	550	1	No- interrogativo	21.2%	77.4%
-	600	1	No- interrogativo	24.0%	78.1%
-	600	1	Interrogativo	38.1%	84.8%
-	750	1	No- interrogativo	24.0%	79.1%
-	750	1	Interrogativo	39.2%	85.6%
270	450	1	Interrogativo	39.5%	86.4%
270	450	1	No- interrogativo	24.0%	79.5%
450	450	1	Interrogativo	41.6%	88.1%
450	450	1	No- interrogativo	27.2%	81.9%
750	450	1	Interrogativo	44.4%	88.5%
-	450	2	Interrogativo	42.3%	87.7%
-	450	2	No- interrogativo	28.9%	83.4%
270	450	2	Interrogativo	42.2%	87.7%
270	450	2	No- interrogativo	28.0%	87.7%
450	450	2	Interrogativo	46.0%	89.5%
450	450	2	No- interrogativo	30.3%	84.1%

Tabla 11.3 Resultados de traducción para las codificaciones *T_I_pmv4_r2_t30_30_bp3000* y diversas redes y ensembles de dos redes RECONTRA. Se muestran los resultados con una única red básica, con dos capas ocultas, con dos palabras de salida y con dos palabras de salida y dos capas ocultas.

Es interesante señalar que siempre se produce una mejora con las frases interrogativas, pero con las no-interrogativas las mejoras tienden a ser inferiores y en algunos casos inexistentes.

11.2.2. Conclusiones

Crear un nuevo sistema de traducción basado en ensembles y la división de la tarea según la aparición o no del símbolo '¿' ha demostrado mejorar los resultados de traducción. Esta división saca provecho de las características de los idiomas implicados en la traducción y no depende de un costoso análisis sintáctico anterior al proceso de traducción.

Aunque la experimentación sólo se ha realizado para la tarea del Turista, esta estrategia es claramente aplicable a otras tareas de traducción del lenguaje natural, y en teoría cuanto mayor sea el número de pares de frases de la tarea, mejores resultados proporcionará dado que habrá suficientes ejemplos de cada subtarea.

Dependiendo de los lenguajes implicados, las tareas se podrían dividir en varias subtareas utilizando distintos criterios, cada una de las cuales se aproximaría con una red especializada. Por ejemplo se podría emplear la división de las frases en función de la aparición de la palabra “no” (frases negativas) o el símbolo de exclamación '!'.

Sin embargo, esto depende en gran parte de los lenguajes implicados en las tareas, puesto que no será aplicable a todos ellos y, por otra parte y de forma evidente, de un experto humano que debe seleccionar la forma de dividir la tarea.

11.3. Ensembles: votación y distancia a la codificación

Se realizaron una serie de experimentos con ensembles utilizando varios criterios de unión y distinto número de componentes. Así, se realizaron experimentos con sistemas de dos, tres, cuatro y cinco componentes. Para ello se seleccionaron redes con distintas características (codificaciones empleadas, inicialización, parámetros de entrenamiento, ventana de entrada, número de capas ocultas, tamaño de la capa oculta, ventana de salida, etcétera) generadas en experimentos anteriores y se procedió a combinarlas.

Como criterios de unión se emplearon versiones de la distancia entre la codificación de salida proporcionada por las redes y la palabra más próxima en el vocabulario, además de un criterio de votación por mayorías y dos combinaciones de criterios de votación y distancia.

Uno de los problemas del método de votación es cómo generar como salida cuando no existe una palabra ganadora (que tenga la mayoría). Al combinar con los métodos de distancia para seleccionar palabras, se obtienen dos nuevos criterios cuando falla el sistema de votación.

Para facilitar la comprensión de los resultados y la comparación entre ellos, en la Tabla 11.4 se pueden observar las características de las redes[61] y resultados obtenidos con dichas redes para la tarea Turista: número de iteraciones de entrenamiento (*Iter*), tamaño

[61] Nombre de las codificaciones utilizadas, ventana de salida, número de unidades en la primera capa oculta (si existe), número de unidades en la segunda capa oculta, etc…

de la/s capa/s oculta/s y número de palabras de salida, así como el nombre asignado (*Red*).

Señalar que existen una serie de componentes (de *1A* a *1E*) con sólo una diferencia: el número de iteraciones de entrenamiento, por tanto las redes y los resultados que proporcionan deberían ser muy similares. Otra característica especial de estas redes es que a la entrada utilizan codificación *T_Local* y a la salida la codificación inglesa de *T_I_pmv4_r2_t30_30_bp3000*.

Red	Iter.	1ªCO	2ªCO	Nombre	Salida	PBT
1A	55	270	450	T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	1	86.9%
1B	70	270	450	T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	1	87.0%
1C	85	270	450	T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	1	86.9%
1D	140	270	450	T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	1	86.9%
1E	150	270	450	T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	1	86.7%
2	150	-	450	T_I_pmv4_r2_t30_30_bp3000	2	81.1%
3	150	-	450	T_I_pmv4_r2_pUNC_t21_21_bp3000	2	79.2%
4	150	-	450	T_V_pmv8_r6_pUNC_t76_49_bp3000	1	65.3%
5	150	-	450	T_V_pmv8_r6_pUNC_t76_49_bp3000f3	1	78.6%
6	150	270	450	T_I_pmv4_r2_t30_30_bp3000	1	80.9%

Tabla 11.4. Características de las redes (componentes) utilizados en los ensembles y los resultados obtenidos en los experimentos originales para la tarea Turista.

En la Tabla 11.5 se pueden observar las caraterísticas de las redes empleadas en ensembles para la tarea Hansards. Como para Turista, se seleccionaron redes con distintas características, no necesariamente las que proporcionaban mejores resultados: codificaciones empleadas *H_V_pmv4_r2_pS_t192_213_bp3000f3_r_bp3000f3_contexto1* (llamadas *cod1* en la tabla para facilitar la lectura), *H_IV_pmv4_r2_r240_240_bp100_scg1600_900_r_bp100_scg3000_3000_contexto1* (*cod2*), *H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1* (*cod3*) y *H_III_pmv4_r2_t60_60_bp3000f3_contexto1* (*cod4*), número de unidades en la capa oculta, número de capas ocultas (1 o 2), número de iteraciones de entrenamiento y número de palabras de salida (1 o 2).

Para facilitar la diferenciación con los componentes empleados con la tarea Turista, se les coloca la letra “H” mayúscula antes del número, para identificar la tarea a la que pertenecen.

Red	| Iter. |	|Entrada|	|1ªCO|	|2ªCO|	Algoritmo	Nombre	|Salida|	PBT
H1	23	9	900	900	BP	cod1	1	47.26%
H2	24	9	900	900	BP	cod1	1	47.12%
H3	25	9	900	900	BP	cod1	1	47.58%
H4	13	9	-	1800	BP	cod2	1	45.73%
H5	25	9	-	2160	BP	cod2	1	48.38%
H6	26	9	-	2160	BP	cod2	1	48.38%
H7	12	9	-	1200	(6.19)	cod2	1	43.85%
H8	13	9	-	1200	(6.19)	cod2	1	44.23%
H9	11	9	-	2160	(6.19)	cod2	1	46.49%
H10	7	5	900	900	BP	cod3	1	33.80%
H11	6	5	-	1800	BP	cod3	1	33.14%
H12	18	5	300	900	BP	cod4	2	32.59%
H13	20	5	300	900	BP	cod4	2	36.14%

Tabla 11.5. Características de las redes utilizadas en los ensembles: tamaño de la/s capa/s oculta/s, palabras de entrada, palabras de salida, método de entrenamiento utilizado en RECONTRA y el par de codificaciones usados, nombre asignado a cada componente (*Red*) y los resultados obtenidos en los experimentos originales para Hansards.

Para la tarea Hansards también se han empleado redes entrenadas con conjuntos distintos de entrenamiento como método de crear diversidad en las redes. Estos conjuntos son subconjuntos del conjunto completo:

- Una simple división en dos de todo el conjunto de entrenamiento, aproximadamente por la mitad: "conjuntoInicial", con las 40000 primeras frases del conjunto y "conjuntoFinal" con el resto.
- Tres conjuntos en función de la longitud de las frases 10, 20 o 30 palabras: corto10, corto20 y corto30.

Las características de cada red pueden observarse en la Tabla 11.6. En todos los casos se emplea Retropropagación del Error como método de entrenamiento y las redes tienen una única palabra de salida.

Red	Nombre	|Entrada|	|CO|	Conjunto	|Iter|	PBT
H14	cod1	9	900	conjuntoInicial	12	40.23%
H15	cod1	9	900	conjuntoInicial	27	39.88%
H16	cod1	9	900	conjuntoFinal	2	43.99%
H17	cod1	9	900	conjuntoFinal	4	44.47%
H18	cod2	9	2160	conjuntoInicial	4	43.25%
H19	cod2	5	1800	corto10	20	41.60%
H20	cod2	5	900	corto20	15	45.31%
H21	cod2	5	1800	corto30	20	42.34%

Tabla 11.6. Características de las redes utilizados en los ensembles para la tarea Hansards: par de codificaciones usados y el conjunto de entrenamiento empleado, nombre asignado a cada componente y los resultados obtenidos en los experimentos originales para Hansards.

11.3.1. Dos componentes

Para ensembles de dos componentes sólo se emplearon los métodos basados en distancia (4.2), (4.3) y (4.5), dado que con sólo dos componentes no puede realizarse ninguna votación significativa.

11.3.1.1. La tarea del Turista

En la Tabla 11.7 pueden observarse los resultados obtenidos al emplear los métodos (4.2), (4.3) y (4.5) para combinar dos componentes, además de la media para cada ensemble y por cada método.

Los resultados obtenidos por el ensemble son siempre mejores que los resultados de traducción obtenidos por el peor componente, pero no siempre se mejoran los resultados respecto al mejor de los dos. Aunque si observamos la comparativa, las medias son similares sin existir una diferencia estadística significativa, y con una ligera ventaja para el método (4.2).

Red		**PBT**		
1	**2**	**(4.2)**	**(4.3)**	**(4.5)**
1A	1B	88.2%	88.4%	88.2%
1A	1E	88.4%	88.3%	88.4%
1A	2	87.2%	87.5%	87.5%
1A	3	86.8%	87.3%	87.0%
1B	1D	88.6%	88.7%	88.6%
1B	6	87.1%	87.6%	87.2%
1C	4	76.1%	69.5%	75.8%
1C	1D	88.3%	88.5%	88.4%
1C	6	87.0%	87.5%	87.1%
2	3	83.9%	83.8%	83.6%
4	2	66.8%	63.5%	64.7%
4	5	64.6%	59.4%	63.1%
4	1D	73.3%	73.1%	72.5%
4	6	68.2%	63.8%	66.1%
5	1D	86.4%	86.9%	86.7%
5	6	83.3%	82.5%	83.1%
Media		**81.5%**	80.4%	81.1%

Tabla 11.7. Resultados de traducción para ensembles de dos redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando *tres criterios de distancia distintos*.

11.3.1.2. La tarea Hansards

En la Tabla 11.8 pueden observarse los resultados obtenidos al emplear los métodos (4.2), (4.3) y (4.4) para combinar dos componentes y la media por ensemble y método.

Los resultados muestran comportamientos similares a los obtenidos para la tarea del Turista, con mejoras en el sistema respecto al componente con el peor resultado, y pocas variaciones respecto a los resultados de los componentes.

En este caso, las medias plantean que el método (4.5) es el mejor sin que la diferencia estadística sea significativa para confirmar esta ventaja respecto a los otros dos métodos.

Red		PBT		
1	**2**	**(4.2)**	**(4.3)**	**(4.5)**
H1	H2	47.45%	47.16%	47.26%
H1	H5	48.72%	48.36%	47.96%
H1	H4	47.34%	47.08%	47.18%
H1	H8	46.34%	46.34%	46.91%
H1	H9	47.82%	47.62%	47.34%
H1	H7	46.49%	46.12%	46.89%
H2	H5	48.71%	48.65%	48.16%
H2	H4	47.44%	47.14%	46.98%
H2	H8	46.51%	46.17%	46.60%
H2	H9	47.84%	47.61%	47.11%
H2	H7	46.70%	46.00%	46.53%
H5	H4	47.71%	47.63%	48.06%
H5	H8	46.97%	46.78%	47.61%
H5	H9	47.46%	47.76%	47.70%
H5	H7	47.33%	47.56%	48.08%
H4	H8	45.91%	45.46%	45.20%
H4	H9	47.03%	46.59%	46.34%
H4	H7	45.98%	45.38%	45.26%
H14	H15	40.96%	40.25%	40.09%
H14	H16	43.30%	43.00%	42.25%
H14	H17	43.68%	43.53%	42.55%
H14	H19	39.03%	43.98%	44.02%
H19	H15	38.57%	44.23%	44.00%
H19	H16	40.24%	45.24%	45.37%
H19	H17	40.50%	43.33%	45.45%
Media		45.44%	45.96%	**46.04%**

Tabla 11.8. Resultados de traducción para ensembles de dos redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando tres criterios de distancia distintos para la tarea Hansards.

11.3.2. Tres componentes

11.3.2.1. La tarea del Turista

En la Tabla 11.9 se muestran los resultados obtenidos al emplear los métodos de Votación (*V* en la tabla), (4.2), (4.3) y (4.5) y Votación con (4.2) (*V1*) o (4.3) (*V2*) para decidir la salida de sistemas con tres componentes.

Para ver la diferencia estadística entre los diferentes métodos, hemos aplicado un estadístico de Friedmann siguiendo la distribución de Fisher F con el fin de analizar la significancia estadística de los resultados. Si consideramos que el número de métodos M = 6 y el número de bases de datos D = 17, podemos aplicar la distribución de Fisher con (M -1) = 5 y (M-1)*(D-1) = 80 grados de libertad. Utilizaremos un nivel de confianza α =0.05, 95 % de confidencia a la hora de fijar el valor crítico de la distribución de Fisher, obteniendo un valor critico de F(5,80) = 2.3. En la comparativa de métodos de la Tabla 11.9 hemos obtenido un valor para el Test de Friedmann de 9.2 con lo que la diferencias son significativas a nivel estadístico entre los métodos.

En general, los métodos basados en votación obtienen mejores resultados que cualquiera de los basados en distancia. Esto sucede cuando uno de los componentes produce resultados sensiblemente peores que el resto (componente *4*).

Como se puede comprobar facilmente en las medias, aparentemente el método de votación compensa el error cometido por este componente si los otros dos están de acuerdo en otra palabra. Sin embargo, en los métodos de distancia este componente puede estar más seguro de su salida que los otros dos, con peores resultados.

Red			PBT					
1	**2**	**3**	**V**	**(4.2)**	**(4.3)**	**(4.5)**	**V1**	**V2**
1A	4	1D	84.4%	74.8%	74.3%	74.5%	86.1%	85.6%
1A	4	6	80.0%	72.4%	68.9%	71.4%	83.0%	81.6%
1A	7	1D	85.3%	87.5%	88.1%	88.1%	88.1%	87.8%
1A	7	6	82.3%	86.4%	87.1%	87.0%	87.0%	86.3%
1A	5	3	82.1%	86.1%	86.6%	86.1%	87.1%	86.6%
1B	4	1D	84.8%	75.2%	74.7%	75.0%	86.5%	85.9%
1B	4	3	79.7%	72.3%	69.1%	70.3%	83.0%	81.4%
1B	4	6	80.2%	73.2%	70.2%	71.9%	83.3%	82.0%
1B	5	1D	85.8%	87.8%	88.2%	88.4%	88.3%	88.1%
1B	5	3	82.2%	86.4%	87.1%	86.5%	87.3%	87.0%
1B	5	6	82.5%	86.6%	87.5%	87.1%	87.2%	86.6%
1C	4	1d	84.9%	75.1%	74.8%	74.6%	86.4%	86.0%
1C	4	6	80.1%	73.3%	71.2%	72.6%	83.3%	82.2%
1C	5	1D	85.7%	87.5%	88.2%	88.3%	88.2%	87.9%
1C	5	6	82.2%	86.6%	87.3%	87.1%	87.2%	86.4%
1D	2	3	81.9%	87.6%	88.1%	88.0%	87.8%	87.8%
1D	4	6	80.2%	74.8%	74.3%	74.6%	83.3%	82.9%
Media			82.6%	80.8%	80.3%	80.7%	**86.1%**	85.4%

Tabla 11.9. Resultados de traducción para ensembles de tres redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando un criterio de votación, tres criterios de distancia distintos y dos criterios de votación y distancia para la tarea del Turista.

11.3.2.2. La tarea Hansards

En la Tabla 11.10 pueden observarse los resultados obtenidos al emplear los métodos de Votación (*V* en la tabla), (4.2), (4.3) y (4.5) y Votación con (4.2) (*V1*) o (4.3) (*V2*) para decidir la salida de sistemas con tres componentes, además de las medias de los resultados.

Los resultados con los distintos métodos son similares a los obtenidos para la tarea del Turista con una importante diferencia. El efecto de un componente con mucho peores resultados que el resto del ensemble (como H12 y H13 en la tabla) no parece ser tan pronunciado.

Para ver la diferencia estadística entre los diferentes métodos, hemos aplicado de nuevo un Test de Friedmann siguiendo la distribución de Fisher F con el fin de analizar la significancia estadística de los resultados. En este caso, el número de métodos es M = 6 y el número de bases de datos D = 27, podemos aplicar la distribución de Fisher con (M -1) = 5 y (M-1)*(D-1) = 135 grados de libertad. Utilizaremos un nivel de confianza α =0.05, 95 % de confidencia a la hora de fijar el valor crítico de la distribución de Fisher,

obteniendo un valor critico de F(5,135) = 2.28. En la comparativa de métodos de la Tabla 11.10 hemos obtenido un valor para el Test de Friedmann de 27.3 con lo que la diferencias son significativas a nivel estadístico entre los métodos. Viendo que los resultados eran suficientemente concluyentes y se mantenian el mismo comportamiento estadístico para 4 y 5 componentes, decidimos no repertir el test en los dos próximos apartados.

Red			PBT					
1	**2**	**3**	**V**	**(4.2)**	**(4.3)**	**(4.5)**	**V1**	**V2**
H1	H2	H5	46.74%	48.62%	47.91%	47.78%	47.79%	47.91%
H1	H2	H4	46.62%	47.49%	47.45%	47.12%	47.48%	47.45%
H1	H2	H8	46.49%	46.61%	47.19%	46.98%	47.18%	47.19%
H1	H2	H9	46.58%	47.86%	47.62%	46.92%	47.64%	47.62%
H1	H2	H7	46.60%	46.96%	47.41%	46.78%	47.54%	47.41%
H1	H5	H4	45.97%	48.07%	48.29%	47.78%	48.52%	48.29%
H1	H5	H8	45.11%	47.46%	47.58%	46.98%	47.77%	47.58%
H1	H5	H9	46.50%	47.95%	48.38%	46.92%	48.45%	48.38%
H1	H5	H7	45.36%	47.94%	48.25%	46.78%	48.18%	48.24%
H1	H4	H8	44.21%	47.02%	46.81%	46.98%	47.08%	46.81%
H1	H4	H9	44.95%	47.83%	47.79%	46.92%	48.10%	47.79%
H1	H4	H7	44.37%	47.31%	47.13%	46.78%	47.55%	47.13%
H1	H8	H9	44.22%	47.07%	46.91%	46.92%	47.16%	46.91%
H1	H8	H7	44.17%	46.10%	46.08%	46.78%	46.10%	46.08%
H1	H9	H7	44.54%	47.53%	47.78%	46.78%	47.99%	47.78%
H3	H5	H12	41.38%	43.83%	43.74%	36.70%	45.41%	43.74%
H3	H5	H13	41.39%	43.77%	43.77%	33.05%	45.4%	43.77%
H3	H20	H14	38.59%	38.86%	43.62%	48.17%	43.95%	42.62%
H3	H20	H15	38.54%	39.29%	43.79%	48.17%	44.21%	43.79%
H3	H20	H16	42.04%	39.43%	45.13%	48.17%	45.67%	45.13%
H3	H20	H17	41.75%	39.53%	45.14%	48.17%	45.60%	45.14%
H3	H20	H18	41.51%	39.40%	44.61%	48.17%	45.05%	44.61%
H3	H20	H19	36.47%	39.40%	40.80%	42.95%	40.72%	40.80%
H3	H20	H21	38.82%	38.62%	40.57%	42.96%	40.87%	40.57%
H14	H16	H20	38.34%	37.79%	42.15%	40.59%	42.45%	42.15%
H14	H19	H20	36.30%	40.36%	40.50%	36.41%	40.98%	40.50%
	Media		42.98%	44.31%	45.63%	45.34%	**45.96%**	45.59%

Tabla 11.10. Resultados de traducción para ensembles de tres redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando un criterio de votación, tres criterios de distancia distintos y dos criterios de votación y distancia para la tarea Hansards.

11.3.3. Cuatro componentes

11.3.3.1. La tarea del Turista

En la Tabla 11.11 pueden observarse los resultados obtenidos con ensembles de cuatro componentes y los seis métodos de decisión/integración empleados anteriormente: V, (4.2), (4.3), (4.5), V1 y V2. Como es evidente cuando comparamos con los resultados obtenidos para ensembles de dos y tres componentes, los resultados han mejorado.

Sin embargo con el método de votación aparecen varios malos resultados (columna *V* de la tabla) mientras que con los métodos que emplean votación *y* distancia se continúan

obteniendo resultados similares o mejores a los obtenidos con distancia. Este comportamiento es evidente al observar las medias de los resultados obtenidos con los distintos métodos.

Red				PBT					
1	**2**	**3**	**4**	**V**	**(4.2)**	**(4.3)**	**(4.5)**	**V1**	**V2**
4	5	1D	1A	81.5%	75.3%	74.6%	75.3%	87.3%	87.8%
4	5	1D	1B	82.2%	88.0%	88.9%	89.0%	87.2%	87.6%
4	5	1D	1C	76.0%	75.4%	74.7%	75.7%	82.4%	82.0%
4	5	6	1A	78.8%	72.8%	69.2%	72.2%	84.2%	84.5%
4	5	6	1B	76.8%	87.2%	88.2%	88.1%	84.3%	84.6%
4	5	6	1C	72.6%	73.0%	71.5%	73.4%	79.7%	78.6%
1A	1C	1B	1D	84.0%	88.8%	89.3%	89.4%	88.3%	88.3%
1A	1C	1B	6	82.3%	88.0%	88.8%	88.8%	87.5%	87.6%
1A	1C	4	1D	75.8%	75.8%	75.2%	76.0%	82.3%	81.7%
1A	1C	4	6	73.8%	75.0%	72.3%	74.2%	79.4%	77.7%
1A	1C	5	6	76.6%	87.2%	88.2%	88.1%	80.6%	80.0%
1B	1C	4	1D	81.1%	75.6%	75.6%	76.2%	84.3%	84.6%
1B	1C	4	6	76.4%	74.3%	72.9%	74.6%	84.3%	80.0%
Media				78.3%	79.7%	79.2%	80.1%	**84.0%**	83.5%

Tabla 11.11. Resultados de traducción para ensembles de cinco redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida comparando los seis métodos de decisión

En parte esto se debe a que alcanzar la mayoría para una palabra resulta más complicado con un número par de componentes.

Una posible solución es modificar el método de votación de forma que se seleccione una salida simplemente si la mitad (dos) de los componentes coinciden (*V'* en la tabla). Experimentalmente en la Tabla 11.12 se puede comprobar como esto proporciona mejores resultados.

Red				V		V'	
1	**2**	**3**	**4**	**FBT**	**PBT**	**FBT**	**PBT**
4	5	1D	1A	40.4%	81.5%	45.6%	88.2%
4	5	1D	1B	40.5%	82.2%	45.6%	88.1%
4	5	1D	1C	28.5%	76.0%	36.3%	84.0%
4	5	6	1A	34.1%	78.8%	44.0%	87.4%
4	5	6	1B	34.2%	76.8%	44.1%	87.3%
4	5	6	1C	23.8%	72.6%	32.4%	82.1%
1A	1C	1B	1D	42.0%	84.0%	45.7%	88.4%
1A	1C	1B	6	39.3%	82.3%	45.2%	88.0%
1A	1C	4	1D	28.2%	75.8%	36.9%	84.0%
1A	1C	4	6	23.5%	73.8%	31.7%	82.1%
1A	1C	5	1D	37.6%	81.4%	45.1%	88.0%
1A	1C	5	6	29.3%	76.6%	43.2%	87.2%
1B	1C	4	1D	37.8%	81.1%	44.7%	87.8%
1B	1C	4	6	28.2%	76.4%	42.7%	87.0%
Media				33.4%	78.5%	**41.7%**	**86.4%**

Tabla 11.12. Resultados de traducción para ensembles de cuatro redes RECONTRA con distintas características, aplicando el criterio de votación (*V*) y el criterio de votación modificado *V'* (dos redes iguales suficiente*)*.

11.3.3.2. La tarea Hansards

En la Tabla 11.13 pueden observarse los resultados obtenidos al emplear los métodos de Votación (*V* en la tabla), (4.2), (4.3) y (4.5) y Votación con (4.2) (*V1*) o (4.3) (*V2*) para decidir la salida de sistemas con cuatro componentes.

Red				PBT					
1	**2**	**3**	**4**	**V**	**(4.2)**	**(4.3)**	**(4.5)**	**V1**	**V2**
H1	H2	H5	H4	42.10%	48.00%	48.09%	48.01%	44.63%	44.62%
H1	H2	H5	H8	41.84%	47.50%	47.44%	47.80%	44.27%	44.09%
H1	H2	H5	H9	41.79%	47.93%	48.06%	47.86%	44.22%	44.14%
H1	H2	H5	H7	41.75%	47.94%	48.01%	48.07%	44.39%	44.47%
H1	H2	H5	H6	41.07%	48.62%	48.50%	48.08%	43.01%	42.85%
H1	H2	H5	H3	45.55%	48.73%	48.65%	48.37%	47.60%	47.45%
H1	H2	H5	H10	40.12%	42.48%	40.69%	41.22%	41.90%	41.86%
H1	H2	H5	H11	40.01%	42.61%	40.71%	44.35%	41.77%	41.87%
H1	H2	H5	H12	39.95%	43.76%	43.15%	47.82%	42.19%	42.42%
H1	H2	H5	H13	40.02%	43.74%	42.96%	46.81%	42.25%	42.44%
H1	H2	H4	H8	41.22%	47.12%	46.86%	47.16%	43.45%	43.15%
H1	H2	H4	H9	41.43%	47.88%	47.67%	47.43%	44.05%	43.72%
H5	H8	H7	H3	40.84%	47.72%	47.35%	48.16%	47.70%	47.00%
H5	H8	H7	H10	39.08%	41.61%	39.87%	39.19%	42.70%	42.21%
H5	H8	H7	H11	38.78%	41.64%	39.90%	41.49%	42.65%	42.15%
H5	H8	H7	H12	38.72%	42.85%	42.07%	47.05%	43.49%	43.54%
H5	H8	H7	H13	38.72%	42.78%	41.92%	45.38%	43.40%	43.32%
H5	H8	H6	H3	43.09%	47.88%	47.65%	48.13%	48.19%	47.86%
Media				40.89%	45.60%	44.98%	**46.24%**	43.99%	43.84%

Tabla 11.13. Resultados de traducción para ensembles de cinco redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando un criterio de votación y tres criterios de distancia distintos para la tarea Hansards.

También se realizaron algunos experimentos con votación modificada, produciéndose una mejora considerable, como puede verse en la Tabla 11.14, donde la media de los resultados muestran claramente que los métodos modificados son mejores.

Red				PBT					
1	**2**	**3**	**4**	**V**	**V1**	**V2**	**V'**	**V'1**	**V'2**
H1	H2	H5	H4	42.10%	44.63%	44.62%	47.25%	47.63%	47.48%
H1	H2	H5	H8	41.84%	44.27%	44.09%	47.21%	47.51%	47.33%
H1	H2	H5	H9	41.79%	44.22%	44.14%	47.55%	47.76%	47.69%
H1	H2	H5	H7	41.75%	44.39%	44.47%	47.38%	47.81%	47.78%
H1	H2	H5	H6	41.07%	43.01%	42.85%	47.95%	47.96%	47.95%
H1	H2	H5	H3	45.55%	47.60%	47.45%	47.39%	47.39%	47.34%
H1	H2	H5	H10	40.12%	41.90%	41.86%	46.93%	47.05%	47.18%
H1	H2	H5	H11	40.01%	41.77%	41.87%	46.96%	46.97%	47.13%
H1	H2	H5	H12	39.95%	42.19%	42.42%	46.75%	47.02%	47.23%
H1	H2	H5	H13	40.02%	42.25%	42.44%	46.76%	47.05%	47.19%
H1	H2	H4	H8	41.22%	43.45%	43.15%	47.05%	47.34%	47.23%
H1	H2	H4	H9	41.43%	44.05%	43.72%	47.13%	47.49%	47.34%
Media				41.40%	43.64%	43.59%	47.19%	**47.42%**	47.41%

Tabla 11.14. Resultados de traducción para ensembles de cuatro redes RECONTRA con distintas características, aplicando el criterio de votación *(V*) y sus derivados (*V1* y *V2*) y el el criterio de votación modificado *V'* (dos redes iguales suficiente) y sus derivados (*V'1* y *V'2*) para la tarea Hansards.

11.3.4. Cinco componentes

11.3.4.1. La tarea del Turista

En la Tabla 11.15 pueden observarse los resultados obtenidos al emplear los seis métodos de decisión con ensembles de cinco componentes.

Comparando los métodos se puede comprobar cómo los mejores resultados de traducción se obtienen con el método (4.5). Pero hay ciertos casos en los que el método de Votación (*V*) obtiene mejores resultados que cualquiera de los basados en distancia. Como en ensembles de menor tamaño esto sucede cuando uno de los componentes produce resultados sensiblemente peores que el resto (componente *4*). Aparentemente el método de votación compensa el error cometido por este componente si los otros dos están de acuerdo en otra palabra.

Sin embargo en los métodos de distancia este componente parece estar más seguro de su salida que los otros dos, con malos resultados. En conjunto la combinación de votación y distancia parece ser la mejor solución al problema de la integración.

Se repite la poca variación en los resultados según las características de las redes que forman los ensembles. Así, ensembles formadas por redes que se diferencian únicamente por la cantidad de entrenamiento a la que han sido sometidas (redes *1A* a *1E*) ofrecen un comportamiento similar a los obtenidos con redes más diversas (número de capas ocultas, de palabras de salida, codificaciones, conjuntos de entrenamiento, etc).

Red					PBT					
1	**2**	**3**	**4**	**5**	**V**	**(4.2)**	**(4.3)**	**(4.5)**	**V1**	**V2**
1A	1B	1C	4	5	84.3%	74.7%	72.8%	74.8%	86.6%	86.1%
1A	1B	1C	4	1D	85.6%	76.2%	76.0%	76.9%	87.5%	87.2%
1A	1B	1C	4	6	84.3%	75.1%	73.5%	75.5%	86.8%	86.2%
1A	1B	1C	5	1D	86.2%	88.1%	89.0%	89.3%	88.9%	89.2%
1A	1B	1C	5	6	84.9%	87.5%	88.6%	88.6%	88.6%	88.8%
1A	1B	1C	1D	6	86.0%	88.2%	89.2%	89.3%	89.0%	89.1%
1A	1B	1C	1D	1E	86.9%	88.9%	89.5%	89.6%	89.3%	89.2%
1A	1B	4	5	1D	84.2%	75.9%	75.5%	76.5%	86.7%	86.2%
1A	1B	4	5	6	81.6%	74.2%	71.7%	74.3%	85.1%	84.3%
1A	1B	4	1D	6	84.2%	76.1%	76.0%	77.0%	86.8%	86.4%
1A	1B	5	1D	6	84.9%	87.7%	88.9%	89.0%	88.6%	89.0%
1A	4	5	1D	6	81.5%	75.7%	75.3%	76.5%	85.4%	84.6%
1A	1C	4	5	1D	84.1%	75.9%	75.5%	76.4%	86.7%	86.3%
1A	1C	4	5	6	81.6%	74.5%	72.5%	74.7%	85.3%	84.5%
1A	1C	4	1D	9	84.3%	76.1%	76.0%	77.0%	86.8%	86.4%
1B	1C	4	5	1D	84.3%	76.1%	75.8%	76.7%	86.9%	86.4%
1B	1C	4	4	6	81.8%	74.7%	73.1%	75.0%	85.4%	84.6%
1B	1C	4	1D	6	84.8%	76.3%	76.3%	77.2%	87.0%	86.3%
1B	1C	4	1E	3	84.2%	76.1%	75.8%	75.9%	86.9%	86.4%
1B	4	5	1D	6	81.8%	75.9%	75.7%	76.8%	85.5%	84.9%
1C	4	5	1D	6	81.8%	75.8%	75.7%	76.7%	85.6%	85.6%
Media					84.0%	78.6%	78.2%	79.2%	**86.9%**	86.6%

Tabla 11.15. Resultados de traducción para ensembles de cinco redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando un criterio de votación *y* tres criterios de distancia distintos para la tarea del Turista.

11.3.4.2. La tarea Hansards

En la Tabla 11.16 pueden observarse los resultados obtenidos al emplear los seis métodos de decisión con ensembles de cinco componentes y las medias de resultados por método y ensemble.

Si comparamos con los resultados con ensembles de cuatro elementos podemos comprobar como los resultados son similares o peores, por lo que se decidió no continuar la experimentación de mayor tamaño.

Red					PBT					
1	**2**	**3**	**4**	**5**	**V**	**(4.2)**	**(4.3)**	**(4.5)**	**V1**	**V2**
H1	H2	H5	H4	H8	44.68%	47.37%	47.43%	47.81%	48.01%	47.70%
H1	H2	H5	H4	H9	45.26%	47.79%	48.005	47.89%	48.53%	48.32%
H1	H2	H5	H4	H7	44.74%	47.67%	47.82%	48.01%	48.32%	48.15%
H1	H2	H5	H4	H6	45.93%	47.98%	48.19%	47.99%	48.57%	48.49%
H1	H2	H5	H4	H3	46.55%	48.18%	48.17%	48.26%	47.97%	47.60%
H1	H2	H5	H4	H10	43.06%	43.26%	40.98%	41.18%	45.81%	44.93%
H1	H2	H5	H4	H11	43.31%	43.31%	41.11%	44.28%	45.74%	45.19%
H1	H2	H5	H4	H12	42.89%	44.63%	43.56%	47.89%	46.57%	46.23%
H1	H2	H5	H4	H13	42.87%	44.64%	43.51%	46.84%	46.56%	46.18%
H1	H2	H5	H8	H9	44.88%	47.24%	47.37%	47.77%	47.84%	47.67%
H1	H2	H5	H8	H7	44.81%	47.44%	47.31%	47.78%	47.92%	47.56%
H1	H2	H5	H8	H6	45.25%	47.52%	47.55%	47.81%	47.97%	47.67%
H1	H2	H5	H8	H3	46.45%	47.68%	47.59%	48.05%	47.67%	47.38%
H1	H2	H5	H8	H10	42.87%	43.17%	41.40%	41.15%	45.68%	45.03%
H1	H2	H5	H8	H11	42.91%	43.29%	41.34%	44.26%	45.70%	45.12%
H1	H2	H5	H8	H12	42.59%	44.25%	43.81%	47.67%	46.15%	46.07%
H1	H2	H5	H8	H13	42.57%	44.20%	43.73%	46.72%	46.20%	46.03%
H1	H2	H8	H19	H20	40.49%	39.88%	31.53%	47.04%	45.90%	45.56%
H1	H2	H8	H14	H20	41.65%	39.28%	32.53%	46.62%	45.87%	45.67%
H1	H3	H11	H17	H20	39.14%	37.70%	43.52%	44.12%	44.01%	44.29%
Media					43.65%	44.82%	43.82%	46.46%	**46.85%**	46.54%

Tabla 11.16. Resultados de traducción para ensembles de cinco redes RECONTRA con distintas codificaciones, distintas topologías y varias palabras de salida, aplicando un criterio de votación, tres criterios de distancia distintos y combinaciones de votación *y* distancia para la tarea Hansards.

11.3.5. Conclusiones

En este apartado se han presentado los resultados organizados por número de componentes de los ensembles con los diferentes métodos de unión. Hay varias conclusiones que se pueden extraer de ellos.

- El sistema de votación no es aplicable con dos componentes y presenta problemas cuando se intenta aplicar a un número par de componentes, al requerir que un mayor número de ellos se ponga de acuerdo.
- Añadir más componentes no es una garantia de mejora de los resultados. Es más importante el tener los componentes adecuados que un mayor número de ellos. Así, los ensembles de cinco componentes no suponen una mejora respecto a los de cuatro, e incluso proporcionan resultados ligeramente peores.

- Comparando los seis métodos empleados se puede comprobar que, aunque los mejores resultados de traducción tienden a obtenerse con los métodos basados en distancias, hay ciertos casos en los que los métodos de votación son claramente mejores y en general tienden a ser más estables. Esto sucede cuando uno de los componentes produce resultados sensiblemente peores que el resto (componente *4*). Aparentemente el método de votación compensa el error cometido por este componente si los otros dos están de acuerdo en otra palabra.
- Los métodos basados en distancia son más versátiles, dado que pueden emplearse con ensembles de dos componentes. Diferenciando entre ellos, claramente se obtienen mejores resultados cuando se emplean las distancias acumuladas, respecto a no emplearlas, dado que una red que ya ha producido buenas traducciones tenderá a seguir produciéndolas.

Es importante señalar la poca variación en los resultados de los ensembles según las características de las redes que forman los componentes. Así, ensembles formadas por redes que se diferencian únicamente por la cantidad de entrenamiento a la que han sido sometidas y que a priori son muy similares ofrecen un comportamiento similar a los obtenidos con redes más diversas. Esto sugiere que los ensembles sufren de un problema de diversidad, es decir, que todas las redes están aprendiendo fundamentalmente de la misma forma, y que se podrían obtener mejores resultados.

Se han realizado algunos experimentos preliminares con otras redes, en concreto con MLP, MLP con conectividad de Bengio (izquierda-a-derecha), RECONTRA con conectividad de Bengio y redes de Jordan [Jordan, 1986] como componentes de los ensembles, principalmente para la tarea del Turista completo. Pero en el mejor de los casos (RECONTRA con Bengio) los resultados han sido similares, siendo peores para las MLP, dado que son redes estáticas y no pueden tratar adecuadamente el problema de traducción.

Se necesita explorar la posibilidad de ensembles con otras redes e incluso otros métodos distintos para lograr mayor diversidad de componentes, pero esto se deja para un trabajo futuro.

11.4. Frases Interrogativas/No-Interrogativas y Distancias

11.4.1. La tarea del Turista

Se realizaron una serie de experimentos combinando las dos técnicas principales: división del conjunto de entrenamiento en dos, frases interrogativas y no-interrogativas y distancia entre tres redes especializadas en cada subconjunto.

Las características de las redes empleadas y la nomenclatura empleada pueden verse en la Tabla 11.17.

Los resultados de traducción obtenidos con estas componentes para cada conjunto pueden verse en las Tablas 11.18 (No-Interrogativo) y 11.19 (Interrogativo). Si

comparamos con los resultados originales en la Tabla 11.2 aplicando sólo la división en dos subconjutos, podemos comprobar como se han producido mejoras en los resultados.

Nombre	Red	|Iterac|	|1ªCO|	|2ªCO|	Conjunto	PBT
T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	I7	100	270	450	Interrogativo	90.2%
T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	N7	100	270	450	No-interrogativo	84.5%
T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	I8	150	270	450	Interrogativo	91.3%
T_Local (entrada) + T_I_pmv4_r2_t30_30_bp3000 (salida)	N8	150	270	450	No-interrogativo	87.1%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	I9	100	-	450	Interrogativo	74.5%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	N9	100	-	450	No-interrogativo	64.7%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	I10	125	-	450	Interrogativo	75.8%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	N10	125	-	450	No-interrogativo	66.7%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	I11	150	-	450	Interrogativo	75.5%
T_V_pmv8_r4_pS_t137_88_bp100_scg400_500	N11	150	-	450	No-interrogativo	67.7%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	I12	100	-	450	Interrogativo	79.6%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	I13	150	-	450	Interrogativo	81.8%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	I14	150	270	450	Interrogativo	84.8%

Tabla 11.17. Resultados de traducción por componentes de un sistema experto con distintas características: modelos RECONTRA con una o dos capas ocultas (270 y 450 unidades) en distintos momentos del entrenamiento y para varias codificaciones.

Existe otra posibilidad de combinar los dos métodos, utilizando el criterio de distancias para unir las salidas de las redes en lugar de la regla de aparición del símbolo '¿'. Como es evidente los resultados de traducción son bastante peores y no se muestran.

Red			PBT					
1	2	3	V	(4.2)	(4.3)	(4.5)	V1	V2
N7	N8	N9	85.9%	87.5%	87.9%	86.3%	87.9%	87.9%
N7	N8	N10	86.1%	87.5%	87.9%	86.3%	87.9%	87.9%
N7	N8	N11	86.1%	87.5%	87.9%	86.4%	87.9%	88.0%
N7	N9	N10	68.3%	86.2%	86.4%	86.3%	70.1%	70.1%
N7	N9	N11	69.3%	86.2%	86.5%	86.4%	71.4%	71.5%
N7	N10	N11	69.2%	86.3%	86.5%	86.4%	70.8%	70.8%
N8	N9	N10	68.5%	86.5%	87.1%	87.1%	70.2%	70.2%
N8	N9	N11	69.3%	86.6%	87.1%	87.1%	70.9%	70.9%
N9	N10	N11	66.6%	65.1%	66.8%	66.1%	66.9%	66.9%

Tabla 11.18. Resultados de traducción para la componente No Interrogativa de un sistema experto Int/No-Int con ensemble de tres redes RECONTRA, distintas topologías y palabras de salida, con criterio de votación, tres criterios de distancia distintos y dos criterios de votación y distancia para la tarea del Turista.

Red			PBT					
1	2	3	V	(4.2)	(4.3)	(4.5)	V1	V2
I7	I8	I9	89.7%	91.5%	91.9%	90.0%	91.7%	91.6%
I7	I8	I10	89.8%	91.5%	91.9%	89.9%	91.7%	91.6%
I7	I8	I11	89.9%	91.4%	91.8%	89.6%	91.7%	91.6%
I7	I8	I12	90.1%	91.3%	91.9%	90.2%	91.7%	91.7%
I7	I8	I13	90.2%	91.3%	91.9%	90.2%	91.6%	91.8%
I7	I8	I14	90.3%	91.4%	91.8%	90.1%	91.8%	91.8%
I8	I9	I10	77.3%	91.3%	91.3%	91.0%	77.5%	77.5%
I8	I9	I11	77.2%	90.9%	91.2%	90.9%	78.6%	78.7%
I8	I9	I14	85.9%	91.3%	91.3%	91.3%	89.0%	88.9%
I8	I3	I14	86.1%	91.1%	91.3%	91.3%	88.7%	88.7%

Tabla 11.19. Resultados de traducción para la componente Interrogativa de un sistema experto Int/No-Int para un ensemble de tres redes RECONTRA con distintas topologías y palabras de salida, con criterio de votación, tres criterios de distancia distintos y dos criterios de votación y distancia para la tarea del Turista.

11.4.2. Conclusiones

Si dividir la tarea en dos según que las frases fuesen interrogativas o no interrogativas permitía mejorar los resultados de traducción respecto a las mismas codificaciones y redes de las mismas características, utilizar varias redes para cada conjunto de entrenamiento y combinar las salidas mediante alguno de los criterios de distancia y/o votación tiende a mejorar aún más los resultados.

Por desgracia, los resultados no mejoran tanto como desearíamos, sugiriendo:

- Por un lado que la diversidad de las redes no es suficiente para formar un buen ensemble.
- Existe un conjunto reducido de frases y/o palabras de difícil traducción dentro de la tarea del Turista.
- En la tarea Hansards la situación es distinta, con muy pocas mejoras en los resultados.
- Como trabajo futuro, se debe aumentar la diversidad de los componentes. Utilizar otro tipo de ANNs es una posibilidad interesante, aunque las pruebas iniciales con redes de Jordan y variaciones de RECONTRA con conexiones “izquierada-a-derecha” no han ofrecido mejoras significativas, pero tal vez un método completamente distinto sea más productivo.

Capítulo 12.

Conclusiones

Resumen: En este capítulo final se presentan las conclusiones a las que se han llegado al realizar esta tesis, tanto con ANNs como codificadores como con modelos conexionistas como traductores. Además se presentan las publicaciones a las que ha dado lugar la tesis y el posible trabajo futuro.

12.1. Introducción

En este capítulo se presentan las conclusiones del trabajo relizado para esta tesis y las aportaciones realizadas, así como posibles líneas de trabajo futuro. Se finaliza con la lista de publicaciones a las que se ha dado lugar en esta tesis.

La utilización de ANNs en tareas de traducción no se había producido previamente en tareas de gran tamaño. Los problemas de escala que aparecían al abordar tareas con grandes vocabularios parecían inabordables. En esta tesis se ha explorado como solucionar estos problemas mediante la creación de codificaciones adecuadas, además de incorporar mejoras a las redes. Así, se ha demostrado experimentalmente su capacidad de abordar tareas más complejas, ya pertenecientes al ambito del discurso natural.

12.2. Aportaciones de esta tesis

Además de la aproximación para tareas de traducción que nunca antes se habían abordado con métodos puramente conexionistas, las principales aportaciones de este trabajo pueden dividirse en tres secciones: la creación de codificaciones automáticas mediante ANNs, la creación y empleo de variantes de redes de Elman como traductores y el diseño de ensembles en ANNs como traductores.

12.2.1. Creación de codificaciones automáticas mediante ANNs.

Se ha demostrado como, con las codificaciones adecuadas, RECONTRA es capaz de aproximar con éxito tareas de traducción complejas. La obtención de estas codificaciones

no es evidente ni trivial, y se han diseñado y empleado diversas arquitecturas de red para este propósito. Los principales aspectos son:

- Se ha desarrollado una nueva topología, incorporando ventanas de salida a MLPs, que crea codificaciones más adecuadas que los MLPs clásicos para las palabras implicadas en tareas de traducción.
- Se han empleado métodos de poda sobre la capa oculta de los MLPs que permiten obtener automáticamente el tamaño de las codificaciones, sin depender de una decisión humana.
- Se han utilizado diversos MLPs en un esquema recurrente para refinar las codificaciones.
- Se han probado diversos métodos de entrenamiento de los MLPs, así como métodos para modificar los parámetros durante el aprendizaje.

12.2.2. Modificaciones de RECONTRA

El modelo RECONTRA básico es una red de Elman con ventana deslizante de entrada. Se han realizado varias modicaciones a este modelo y dos de ellas han demostrado experimentalmente sus ventajas:

- La utilización de dos capas ocultas, permite que una de ellas puede dedicarse a crear representaciones internas más adecuadas para la codificación de entrada. Añadir a esto el esquema de conectividad entre capas empleado por Bengio (Left-to-Right) no ha supuesto una mejora en los resultados.
- El uso de una ventana deslizante de salida fuerza al modelo a tener más presente las salidas anteriores que se han producido. La utilización de un contexto para esta capa de salida no ha mejorado los resultados, aunque sí que se ha podido incorporar el uso de dos capas ocultas con éxito.

12.2.3. Ensembles

Se han creado varios nuevos sistemas de traducción mediante Ensembles basados en modelos RECONTRA.

- Dividir la tarea en dos empleando una regla sintáctica (la aparición del signo “¿”) para abordarlas por separado mejora los resultados.
- El manejo de una distancia entre la salida producida por el traductor y la palabra más cercana del vocabulario como una medida de la confianza que el traductor tiene en su salida, permite emplear estos valores como un criterio de decisión entre las componentes del ensemble. Durante la tesis se han desarrollado varias metodologías para resolver este problema.
- Como no, la combinación de división de la tarea por la aparición del signo “¿” y el uso de ensembles para cada subconjunto mejora los resultados.

Además, se han abordado nuevas tareas más complejas, que no se habían abordado antes con RECONTRA ni con ningún otro método basado en ANNs.

12.3. Trabajo futuro

Existen muchas líneas innovadoras que no han sido suficientemente exploradas, que permiten el desarrollo de trabajo en el futuro. A continuación se comentan algunas de ellas:

- Abordar más tareas con lenguajes no indoeuropeos, más alejados de su norma y organización.
- Explorar como obtener el formato de ventana óptimo para la ventana de salida de los MLP codificadores, más allá del método de prueba y error empleado.
- Aplicar métodos de entrenamiento, poda e inicialización de los MLP distintos de los ya utilizados.
- Explorar nuevos y mejores métodos de entrenamiento de modelos RECONTRA, incluyendo métodos de poda y explorar las posibilidades de emplear distintas inicializaciones de los pesos.
- Incorporación de modelos del lenguaje para las salidas de las redes, así como para la integracion de los ensembles.
- Nuevos tipos de ANNs, por ejemplo redes LSTM (*Long Short Term Memory*) [Hochreiter, 1997] que han demostrado su capacidad de resolver problemas complejos relacionados con el lenguaje humano [Woellmer, 2009] y las basadas en *Reservoir Computing* [Lukosevicius, 2007].
- Emplear otros tipos de redes, como los ya mencionados (LSTM y Reservoir Computing) y ampliar la experimentación con redes de Jordan y variaciones de RECONTRA con conexiones "izquierda-a-derecha", como componentes de los ensembles.
- Saliendo del entorno conexionista, se podrían incorporar otros métodos de traducción no conexionistas a los ensembles, y especialmente métodos de agrupación (clasificación) de palabras que permiten reducir el tamaño de los vocabularios y simplifican la tarea de traducción de la red.

12.4. Publicaciones relacionadas con la tesis

[Casañ, 1999] G.A. Casañ, M.A. Castaño. Distributed Representation of Vocabularies in the RECONTRA Neural Translator. Procs. 6th European Conference on Speech Communication and Technology vol. 6, pag. 2423–2426, Budapest, 1999.

[Casañ, 2000] M.A. Castaño, S. Barrachina, G.A. Casañ. F. Prat y J.M. Vilar. Redes Neuronales y Técnicas Inductivas de Categorización Aplicadas a la Traducción. Las Tecnologías del Habla, Procs. I Jornadas en Tecnología del Habla. Sevilla, 2000.

[Casañ, 2001] G. A. Casañ, M. A. Castaño. Inference of Stochastic Regular Languages through Simple Recurrent Networks with Time Delays. Proceedings of the 6th International Workshop on Artificial Neural Networks (IWANN2001), vol. II, Ed. Springer-Verlag, pag. 671–675. Granada, 2001.

[Casañ, 2001] G.A. Casañ, M.A. Castaño. Application of a Hierarchical Clustering Algorithm to Word Codification in Machine Translation. Pattern Recognition and Image analysis, vol. II, pag. 349–354, 2001.

[Casañ, 2003a] G.A. Casañ, M.A. Castaño. Automatic Word Codification for the RECONTRA Connectionist Translator. LNCS 2652. Artificial Neural Nets Problem Solving Methods, Part II, pag. 766–773, 2003.

[Casañ, 2003b] G.A. Casañ, M.A. Castaño. Automatic Size Determination of Codifications for the Vocabularies of the RECONTRA Connectionist Translator. LNCS. Pattern Recognition and Image Analysis, pag. 168–175, 2003.

[Casañ, 2003c] G.A. Casañ, M.A. Castaño. Generación Automática de Codificaciones de Palabras para Tareas de Lenguaje Natural. Revista Ibero-americana de Inteligencia Artificial, monográfico "Acceso a Información Multilingüe", 2003.

[Casañ, 2004] G.A. Casañ, M.A. Castaño. Improvements on Automatic Word Codification for Connectionist Machine Translation. Procs. of the 16th European Conference on Artificial Intelligence ECAI, pag. 576–580, Valencia, 2004.

[Casañ, 2005a] G.A. Casañ, M.A. Castaño. Automatic Distributed Codifications with Feedback for the RECONTRA Neural Translator. Procs. of the 6th International Symposium on Natural Language Processing, vol. I, pag. 67–72, Thailandia, 2005.

[Casañ, 2005b] G.A. Casañ, M.A. Castaño. A New Approach to Codifications for the RECONTRA Neural Translator. Procs. of the 9th IASTED International Conference Artificial Intelligence and Soft Computing, pag. 147–152, Mallorca, 2005.

[Casañ, 2007] G.A. Casañ, M.A. Castaño. Tuning word codifications for the RECONTRA translator. Borrajo et al. (Eds.) Actas de 12th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2007, vol. II, pag. 351–354, Salamanca, España, Noviembre 2007.

[Alejo, 2008] R. Alejo, J. M. Sotoca, G.A. Casañ. An Empirical Study for the Multi-class Imbalance Problem with Neural Networks. LNCS 5197, Progress in Pattern Recognition, Image Analysis and Applications, pag. 479–486, 2008.

[Casañ, 2008a] G.A. Casañ, M.A. Castaño. An Improved Connectionist Translator between Natural Languages. LNAI 5290, H. Geffnet et al. (Eds.), Advances in Artificial Intelligence 11th Ibero-American Conference on Artificial Intelligence (IBERAMIA 2008), pag. 282–291, Springer, Lisboa, Portugal, Octubre 2008.

[Casañ, 2008b] G.A. Casañ, M.A. Castaño. A Connectionist Automatic Encoder and Translator for Natural Languages, Advances in Soft Computing, no. 50, Corchado at al. (Eds.), International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008), pag. 415–423, Springer, Salamanca, España, 2008.

[Alejo, 2009] R. Alejo, J. M. Sotoca, R. M. Valdovinos, G. A. Casañ. The Multi-class Imbalance Problem: Cost Functions with Modular and Non-Modular Neural Networks. Advances in Intelligent and Soft Computing, no. 56, The Sixth International Symposium on Neural Networks 2009, Springer Verlag, pag. 421–431, 2009.

[Casañ, 2009] G.A. Casañ, M.A. Castaño. An Ensemble Based Translator for Natural Languages. LNCS 5518, S. Omatu et Al. (Eds.), Distributed Computind, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, 10th International Work-Conference on Artificial Neural Networks, IWANN 2009 Workshops, part II, pag. 264–271, Springer, Salamanca, España, Junio 2009.

Capítulo 13.

Bibliografía

[Ackley, 1985] D. Ackely, G. Hinton, T. Sejnowski. *Learning in Boltzman machines*. Cognitive Science, 9, pag. 147–169, 1985.

[Andrés-Ferrer, 2007] J. Andrés-Ferrer, A. Juan-Císcar. *A phrase-based hidden markov model approach to machine translation*. Procs. New Approaches to Machine Translation, pag. 57–62, 2007.

[Arnold, 1994] D. Arnold, L. Balkan, S. Meijer, R.Lee Humphreys y L. Sadler. *Machine Translation: An Introductory Guide*. NCC Blackwell, London, 1994.

[Amengual, 1997] J.C. Amengual, J.M. Benedi, K. Beulen, F. Casacuberta, M.A. Castaño, A. Castellanos, D. Llorens, A. Marzal, H. Ney, F. Prat, E. Vidal y J.M. Vilar. *Speech Translation Based on Automatically Trainable Finite-State Models.* Procs. of the 5th European Conference on Speech Communication and Technology (EUROSPEECH-97), vol. 1, pag. 91–95, Rhodes, Greece, 1997.

[Amengual, 2001] J.C. Amengual, M.A. Castaño, A. Castellanos, D. Llorens, A. Marzal, F. Prat, J.M. Vilar J.M. Benedi, F. Casacuberta, M. Pastor y E. Vidal. *The EUTRANS-I Spoken Language Translation System.* Machine Translation, no. 15, pag. 75–103. Kluwer Academic Publishers. 2001.

[Bengio, 2001] Y. Bengio, R. Ducharme y P. Vincent. *A Neural Probabilistic Language Model.* In Advances in Neural Information Processing Systems, vol. 13, pag. 932–938*,* 2001.

[Bengio, 2003] Y. Bengio, R. Ducharme, P., Vincent y C. Jauvin. *A Neural Probabilistic Language Model.* Journal of Machine Learning Research, vol. 3, pag. 1137–115, Mit Press, 2003.

[Brown, 1990] P. F. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, R. Mercer y P. Roossin. *A Statistical Approach to Machine Translation.* Computational Linguistics, vol. 16, no. 2, pag. 79-95, 1990.

[Brown, 1993] P.F. Brown, S.A. Della Prieta y R.L. Mercer. *The Mathematics of Statistical Machine Translation: Parameter Estimation.* Computational Linguistics, vol. 19, no. 2, pag. 263–311, 1993.

[Carl, 2008] M. Carl, M. Melero, T. Badia, V. Vandefhinste, P. Dirix, I. Schuurman, S. Markantonatou, S. Sofianopoulus, M. Vassiliou y O. Yannoutsou. *METIS-II: low resource machine translation*. Machine Translation, vol. 22, pag. 67–99, Springer, 2008.

[Castaño, 1997] M. A. Castaño y F. Casacuberta. *A Connectionist Approach to Machine Translation.* Procs. of the 5th European Conference on Speech Communications and Technology (EUROSPEECH-97), vol. 1, pag. 91–94, Rhodes, Greece, 1997.

[Castaño, 1998] M. A. Castaño. *Redes Neuronales Recurrentes para Inferencia Gramatical y Traducción Automática.* Tesis doctoral, Dpto. Lenguajes y Sistemas Informáticos, Universidad Politécnica de Valencia, Valencia, 1998.

[Castaño, 1999] M. A. Castaño y F. Casacuberta. *Text-to-Text Machine Translation Using the RECONTRA Connectionist Model*. Lecture Notes in Computer Science: Applications of Bio-Inspired Artificial Neural Networks, vol. 1607, pag. 683–692, José Mira, Juan Vincente Sánchez-Andrés (Eds.), Springer-Verlag, 1999.

[Castellanos, 1994] A. Castellanos, I. Galiano y E. Vidal. *Applications of OSTIA to Machine Translation Tasks*. Lecture notes in Computer Science-Lecture Notes in Artificial Intelligence: Grammatical Inference and Applications, vol. 862, pag. 93–105, R.C. Carrasco and J. Oncina (Eds.), Springer-Verlag, 1994.

[Castellanos, 1998] A. Castellanos, E. Vidal, M. A. Varó y J. Oncina. *Language Understanding and Subsequential Transducer Learning*. Computer Speech and Language, no. 12, pag. 193–228. Academic Press, 1998.

[Castro, 2001] M.J. Castro, V. Polvoreda y F. Prat. *Connectionist N-gram Models by using MLPs*. Procs. of the Second Workshop on Natural Language Processing and Neural networks (NLPNN'01), pag. 16–22, 2001.

[Chalmers, 1990] D. J. Chalmers. *Syntactic Transformations on Distributed Representations*. Connection Science, vol. 2, pag. 53–62, 1990.

[Chan, 2007] Yee Seng Chan, Hwee Tou Ng y David Chiang. *Word sense disambiguation improves statistical machine translation*. Procs. Assoc. for Computational Linguistics (ACL), 2007.

[Chiang, 2005] D. Chiang. *A Hierarchical Phrase-Based Model for Statistical Machine Translation*. Procs. 43rd Annual Meeting of the Assoc. for Computational Linguistics (ACL), pag. 263–270, 2005.

[Chiang, 2007] D. Chiang. *Hierarchical Phrase-Based Translation*. Computational Linguistics, vol. 33, no. 2, pag. 201–228, 2007.

[Chomsky, 1968] N. Chomsky. *El lenguaje y el entendimiento*. Obras Maestras del Pensamiento Contemporáneo. Ed. Planeta-Agostini. Barcelona. 1968.

[Chrisman, 1991] L. Chrisman. *Learning Recursive Distributed Representation for Holistic Computation.* Connection Science, vol. 3, no. 4, pag. 345–366, 1991

[Cowan, 2006] B. Cowan, I. Kucerova y M. Collins. *A Discriminative Model for Tree-to-Tree Translation.* Procs. Conference on Empirical Methods in Natural Language Processing (EMNL 2006), pag. 232–241, 2006.

[Cranias, 1994] L. Cranias, H. Papageorgiou y S. Peperidis. *A matching technique in example-based machine translation*. Procs. of the Fifteenth International Conference on Computational Linguistics, pags. 100–104, 1994.

[Darken, 1991] C. Darken y J. Moody. *Note on learning rate schedules for stochastic optimization*. Neural Information Processing Systems 3, pag. 832–838. Morgan Kaufmann, 1991.

[Darroch, 1972] J.N. Darroch y D. Ratcliff. *Generalized iterative scaling for log-linear models*. The Annals of Mathematical Satatistics, vol. 43, pag. 1470–1480, 1972.

[Desai, 2002] R. Desai. *Bootstrapping in miniature language adquisition*. Cognitive Systems Research, vol. 3, no. 1, pag. 15–23, 2002.

[Dow, 1991] R. Dow y J. Sietsma. *Creating Artificial Neural Networks That Generalize*. Neural Networks, 4 (1), pag. 67–79, 1991.

[Elman, 1990] J.L. Elman. *Finding Structure in Time.* Cognitive Science, vol. 2, no. 4, pag. 279–311, 1990.

[Elman, 1991] J.L. Elman. *Distributed Representations, Simple Recurrent Networks, and Grammatical Structure*. Machine Learning, vol. 7, pag. 195–225, 1991.

[Feldman, 1990] J.A. Feldman, G. Lakoff y A. Stolcke, S.H. Weber. *Miniature Language Adquisition: A Touchstone for Cognitive Science.* Technical Report nº TR-90-0009, Int. Computer Science Institute, Berkeley, California, 1990.

[Friedman, 1937] M. Friedman. The Use of ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. Journal of the American Statistical Association, vol. 32, nº 200, pag. 675-701, 1937.

[Galley, 2006] M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, I. Thayer: *Scalable Inference and Training of Context-Rich Syntactic Translation Models*. Procs. 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL), pag. 961–968, 2006.

[Galley, 2004] M. Galley, M. Hopkins, K. Knighty D. Marcu. *What's in a translation rule?* Proc. Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLT-NAACL), pag. 273–280, 2004.

[Giles, 2001] C.L. Giles, S. Lawrence y A. C. Tsoi. *Noisy Time Series Prediction using Recurrent Neural Networks and Grammatical Inference*. Machine Learning, vol. 44, 161–183, 2001.

[Goodman, 2002] J. Goodman. *Sequential Conditional Generalized Iterative Scaling*. In Actas of 40th Meeting of the Association for computational Linguistics (ACL), pag. 9–16, 2002.

[Hochreiter, 1997] S. Hochreiter y J. Schmidhuber. *LSTM can solve hard long time lag problems*. En M. C. Mozer, M. I. Jordan, T. Petsche, eds., Advances in Neural Information Processing Systems 9, NIPS'9, pag. 473-479, MIT Press, Cambridge MA, 1997.

[Hofstadter, 1987] D.R. Hofstadter. *Gödel, Escher, Bach un Eterno y Grácil Bucle*. Ed. Tusquets Editores, 1987.

[Hutchins, 1992] J. Huttchins y H.L. Somers. *An Introduction to Machine Translation*. London: Academic Press, 1992.

[Jordan, 1986] M. I. Jordan. *Attractor dynamics and parallelism in a connectionist sequential machine*. En Proc. Of the Eighth Annual Conference of the Cognitive Science Society, pag. 531-546, Hillsdale, NJ, 1986.

[Koehn, 2003] P. Koehn y F.J. Och, D. Marcu. *Statistical Phrase-Based Translation*. Proc. Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLT-NAACL), pag. 127–133, 2003.

[Koncar, 1994] N. Koncar y G. Guthire. *A Natural Language Translation Neural Network.* Procs. of the Int. Conf. On New Methods in Language Processing, pag. 71–77, 1994.

[Kuncheva, 2004] L. I. Kuncheva. *Combining Pattern Classifiers*. John Wiley & Sons, Inc., 2004.

[Lamclais, 2005] P. Lanclais, S. Grandrabur, T. Leplu y G. Lapalme. *The Long-Term Forecast for Wather Bulletin Translation*. Machine Translation, vol. 19, pag. 83–112, Springer, 2005.

[Lawrence, 1996] S. Lawrence, S. Fond y C. L. Giles. *Natural Language Grammatical Inference: A Comparison of Recurrent Neural Networks and Machine Learning Methods*. Symbolic, Connectionist, and Statistical Approaches to Learning for Natural Language Processing, Lecture Notes in AI, (Eds.) S. Wermter, E. Riloff y G. Scheler , pag. 33–47, Springer Verlag ,1996.

[Lukosevicius, 2009] M. Lukosevicius y H. Jaeger. *Reservoir Computing Approaches to Recurrent Neural Network Training*. Computer Science Review 3(3), pag, 127-149,2009.

[Marcu, 2002] D. Marcu y W. Wong. *A Phrase-Based, Joint Probability Model for Statistical Machine Translation*. Procs. Conf. on Empirical Methods for Natural Language Processing (EMNLP), pag. 133–139, 2002.

[Marzal, 1993] A. Marzal y E. Vidal. *Computation of Normalized Edit Distance and Applications*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, no. 9, 1993.

[McTait, 2001] K. McTait. *Memory-Based Translation Using Translation Patterns*. Procs. of the 4th Annual CLUK Colloquium, pag. 43–52, 2001

[Medler, 1998] D.A. Medler. *A Brief History of Connectionism*. *Neural Computing Surveys*, vol. 1, no. 1, pag. 61–101, 1998.

[Melamed, 2004] I.D. Melamed. *Statistical Machine Translation by Parsing*. Procs. 42nd Annual Meeting of the Assoc. for Computational Linguistics (ACL), pag. 653–660, 2004.

[Miikkulainen, 1988] R. P. Miikkulainen y M. G. Dyer. *Forming Global Representations with Extended BackPropagation*. Procs. of the IEEE International Conference on Neural Networks, 1998.

[Miikkulainen, 1991] R. P. Miikkulainen y M. G. Dyer. *Natural Language Processing with Modular Neural Networks and Distributed Lexicon*. Cognitive Science, vol. 15, pag. 343–399, 1991.

[Mozer, 1990] M.C. Mozer y P. Smolensky. *Skeletonization: a Technique for Trimming the Fat from a Network via Relevance Assessment*. Advances in Neural Information Processing 1, D.S. Touretzky, Ed. Morgan Kaufmann, pag. 177–185, 1990.

[Nakamura, 1989] M. Nakamura y K. Shikano. *A study of English word category prediction based on neural networks*. Procs. of the International Conference on Acoustics, Speech and Signal Proceesing (ICASSP'89), pag. 731–734, 1989.

[Ney, 2001] H. Ney. *The Statistical Approach to Spoken Language Translation*. Procs. IEEE Automatic Speech Recognition and Understanding Workshop, Madonna di Campiglio, Trento, Italy, 8 pages, CD ROM, IEEE Catalog Nº 01EX544, 2001.

[Nirenburg, 2003] S. Nirengurg, H. Somers y Yorick Wilks (eds.) *Readings in Machine Translation*. A Bradford Book The MIT Press, 2003.

[Och, 1999] F.J. Och, C. Tillmann, H. Ney. *Improved Alignment models for Statistical Machine Translation*. Procs. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP99), pag. 20–28, 1999.

[Och, 2001] F.J. Och, N. Ueffing y H. Ney. *An Efficient A* Search Algorithm for Statistical Machine Translation*. ACL-EACL-2001: 39th Annual Meeting of the Association for Computational Lingustics: Procs. of the Workshop on Data-Driven Machine Translation, pag. 55–62, 2001.

[Och, 2002] F.J. Och y H. Ney. *Discriminative Training and Maximum Entropy models for Statistical Machine Translation*. ACL 2002: Procs. 40th Annual Meeting of the Association for Computational Lingustics, pag. 295–302, 2002.

[Och, 2003] F.J. Och y H. Ney. *A Systematic Comparison of Various Statistical Alignment Models*. Computational Linguistics, vol. 29, no. 1, pag. 19–51, 2003.

[Oncina, 1991] J. Oncina. *Aprendizaje de lenguajes regulares y funciones subsecuenciales*. Tesis doctoral. Universidad Politecnica de Valencia, 1991.

[Penrose, 1990] R. Penrose. *The Emperor's New Mind: Concerning Computers, Minds and the Laws of Physics*. Oxford University Press, 1990.

[Penrose, 1995] R. Penrose. *Shadows of the Mind: A Search for the Missing Science of Consciousness*. Oxford University Press, 1995.

[Pinker, 1995] S. Pinker. *El instinto del lenguaje*. (Original: The Language Instinct. How the Mind Creates Language. 1994) Alianza Editorial, 1995.

[Plate, 1994] Tony A. Plate, *Distribued Representations and Nested Compositional Structure*. Tesis doctoral, Graduate Department of Computer Science, University of Toronto, 1994.

[Pollack, 1990] J. B.Pollack. *Recursive Distributed Representations*. Artificial Intelligence, vol. 46, pag. 77–105, 1990.

[Prat, 2001a] F. Prat, F. Casacuberta y M. J. Castro, *Machine Translation with Grammar Association: Combining Neural Networks and Finite-State Models*, 2nd Workshop on Natural Language Processing and Neural Networks, pag. 53–61, 2001.

[Prat, 2001b] F. Prat, *Machine Translation with Grammar Association: Some Improvements and the Loco_C Model*, Procs. of the Association for Computational Linguistics 2001 Workshop on Data-driven Machine Translation, pag. 71–78, 2001.

[Riedmiller, 1993] M. Riedmiller y H. Braun. *A direct adaptative method for faster backpropagation learning: The RPROP algorithm*. Procs. IEEE International Conference on Neural Networks (ICNN 1993), vol. 1, pag. 586–591, 1993.

[Rodriguez, 2003] P. Rodriguez. *Comparing Simple Recurrent Networks and n-grams in a Large Corpus*. Journal of Applied Intelligence, vol. 19, nº 1-2, pag. 39-50,2003.

[Rojas, 1996] R. Rojas. *Neural Networks. A Systematic Introduction.* Springer-Verlag, 1996.

[Rumelhart, 1986] D.E. Rumelhart, G. Hinton y R. Williams. *Learning Sequential Structure in Simple Recurrent Networks.* Parallel distributed processing: Experiments in the microestructure of cognition, vol. 1. (Eds.) D.E. Rumelhart, J.L. McClelland y el PDP Research Group, MIT Press. Cambridge, 1986.

[Sato, 1990] S. Sato y M. Nagao. *Toward Memory-based Translation*. Procs. of Conference On Computational Linguistics (COLING), vol. 3, pag. 247–252, 1990.

[Scharzl, 2001] A. Schwarzl. *The (Im)Possibilities of Machine Translation*. (Ed.) Peter Lang. European University Studies, 2001.

[Schwenk, 2002] H. Schwenk y J-L. Gauvain. *Connectionist language modelling for large vocabulary continuous speech recognition*. Procs. of the International Conference on acoustics, speech, and Signal Processing (ICASSP'02), pag. 765–788, 2002.

[Sharkey, 1990] N.E. Sharkey. *Connectionist Representations for Natural Language: Old and New*. Procs. of the VI Congreso de la Sociedad Española para el Procesamiento del Lenguaje Natural (SEPLN), pag. 11-27, 1990.

[Silva, 1990] F. H. Silva y L. B. Almedia. Speeding Up Backpropagation. EnR. Eckmiller (Ed.) Advanced Neural Computers (Procs. of the) International Symposium on Neural Networks for Sensory and Motor Systems (NSMS). 1990.

[Sutton, 1986] R.S. Sutton. *Two problems with backpropagation and other steepest-descent learning procedures for networks*. Procs. of the Eighth Annual Conference of the Cognitive Science Society, pag. 823-831, 1986. Reinpreso en Artificial Neural Networks: Concepts and Theory, (Eds.) P. Mehra y B. Wah, IEEE Computer Society Press, 1992.

[Theodoridis, 1998] S. Theodoridis y K. Koutrumbas. *Pattern Recognition.* Academic Press, 1998.

[Tillmann, 2003] C. Tillmann y F. Xia. *A Phrase-based Unigram Model for Statistical Machine Translation*. Procs. Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLT-NAACL), Companion Volume: Short Papers, pag. 106–108, 2003.

[Tomás, 2001] J. Tomás y F. Casacuberta. *Monotone statistical translation using word groups*. Procs. of the Machine Translation Summit VIII, pags. 357–361, 2001

[Tomás, 2004] J. Tomás y F. Casacuberta. *Statistical Machine Translation Decoding Using Target Word Reordering*. Structural, Syntactic, and Statistical Pattern Recognition, vol. 3138 de Lecture Notes in Computer Science, pag. 734–743, Springer-Verlag, 2004.

[Turcato, 2001] D. Turcato y F. Popowich. *What is Example-Based Machine Translation?* Procs. of the Workshop on Example-Based Machine Translation, hosted by MT-Summit VIII, pag. 59-81, 2001.

[Übeyli, 2008] Elif D. Übeyli y Mustafa Übeyli. *Case Studies for Applications of Elman Recurrent Neural Networks*. Recurrent Neural Networks, (Eds.) Xiolin Hu y P. Balasubramaniam. Editorial IN-TECH, 2008.

[Vidal, 1997] E. Vidal. *Finite-State Speech-to-Speech Translation*. Procs. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), vol. 1, pag. 111–114, 1997.

[Vilar, 1999] J. M. Vilar, V. M. Jiménez, J. C. Amengual, A. Castellanos, D. Llorens y E. Vidal. *Text and Speech translation by means of subsequential transducers*. Extended finite state models of language, Ed. András Kornai. Cambridge University Press, 1999.

[Vogel, 1996] S. Vogel, H. Ney, C. Tillmann. *HMN-Based Word aligment in Statistical Translation.* Procs. of the Int. Conference on Computational Linguistics, pag. 836–841, Copenhagen, Denmark, 1996.

[Vogel, 2003] S. Vogel. *SMT Decoder Dissected: Word Reordering*. Procs. Int. Conf. on Natural Language Processing and Knowledge Engineering (NLP-KE), pag. 561–566, 2003.

[Waibel, 1991] A. Waibel, A.J. Jain, A.E. McNair, H. Saito, A.G. Hauptmann y J. Tebelskis. *JANUS: A Speech-to-Speech Translation System using Connectionist and Symbolic Processing Strategies*. Procs.. of the International Conferene on Acoustics, Speech and Signal Processing (ICASSP-91), vol. 2, pag. 793–796, 1991.

[Wang, 1998] Y. Wang y A. Waibel. *Modeling with structures in statistical machine translation*. Procs. of the 36th annual meeting on Association for Computational Linguistics, vol. 2, pag. 1357–1363, 1998.

[Woellmer, 2009] M. Woellmer, F. Eyben, A. Graves, B. Schuller y G. Rigoll. *A tandem BLSTM-DBN architecture for keyword spotting with enhanced context modeling*. Proc. of NOLISP 2009, ISCA Tutorial and Research Workshop on Non-Linear Speech Processing, Vic, Spain, 2009.

[Wu, 1996] D.Wu. *A Polynomial-Time Algorithm for Statistical Machine Translation*. Proc. 34th Annual Meeting of the Assoc. for Computational Linguistics (ACL), pag. 152–158, 1996.

[Wu, 1997] D. Wu. *Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora.* Computational Linguistics, vol. 23, no. 3, pag. 377–403, 1997.

[Xu, 2000] Wei Xu y A. Rudnicky. *Can Artificial Neural Networks Learn Language Models?* Procs. of the 6th International Conference in Spoken Language Processing (ICSLP'00), vol. 1, pag. 118-121, 2000.

[Yamada, 2001] K. Yamada y K. Knight. *A Syntax-Based Statistical Translation Model*. Procs. 39th Annual Meeting of the Assoc. for Computational Linguistics (ACL), pag. 523–530, 2001.

[Yamada, 2002] K. Yamada y K. Knight. *A Decoder for Syntax-based Statistical MT*. Procs. 40th Annual Meeting of the Assoc. for Computational Linguistics (ACL), pag. 303–310, 2002.

[Zell, 1995] A. Zell et al. *SNNS: Stuttgart Neural Network Simulator. User manual, Version 4.1*. Technical Report nº. 6195, Institute for Parallel and Distributed High Performance Systems, University of Stuttgart, 1995.

[Zens, 2002] R. Zens, F.J. Och y H. Ney. *Phrase-based statistical machine translation*. Procs. German Conference on Artificial Intelligence (KI), vol. 2479 de Lecture Notes in Artificial Intelligence (LNAI), pag. 18–32, 2002.

[Zens, 2008] R. Zens. *Phrase-based Statistical Machine Translation: Models, Search, Training*. Tesis doctoral, Aachen, Germany, Febrero 2008.

[Zollmann, 2006] A. Zollmann y A. Venugopal. *Syntax Augmented Machine Translation via Chart Parsing*. Procs. Human Language Technology Conf. / North American Chapter of the Assoc. for Computational Linguistics Annual Meeting (HLTNAACL): Workshop on Statistical Machine Translation, pag. 138–141, 2006.

Anexos

Experimentación secundaria

Anexo A. Experimentación

A.1. Experimentación adicional relacionada con el Capítulo 7

Codificación	Ventana	CO	Pesos	FBT	PBT
H_II_t30_30	9	450	337500	30.93%	27.87%
H_II_t30_30	9	600	540000	30.23%	29.97%
H_II_t30_30	9	900	1080000	31.53%	27.63%
H_II_t30_30	5	450	283500	30.13%	26.47%
H_II_t30_30	5	900	972000	0.00%	0.03%
H_II_t30_30	14	450	405000	27.93%	27.70%

Tabla A.1. Resultados con las codificaciones H_II_t30_30 para diversos tamaños de la capa oculta y tamaños de la ventana de entrada de RECONTRA 9, 5 y 14 palabras y la tarea Hansards.

A.2. Experimentación adicional relacionada con el Capítulo 8

Nombre	|CO|	|Entrada|	|Iter.|	PBT	FBT
MinTsC_L_pmv6_r4_t25_25_bp1000	140	6	500	62.40%	94.16%
MinTsC_L_pmv6_r4_t25_25_bp1000	140	6	1000	62.60%	94.10%
MinTsC_L_pmv6_r4_t25_25_bp1000	140	8	500	71.20%	95.76%
MinTsC_L_pmv6_r4_t25_25_bp1000	160	6	500	57.20%	93.04%
MinTsC_L_pmv6_r4_t25_25_bp1000	180	6	500	61.20%	93.50%
MinTsC_L_pmv6_r4_t25_25_bp1000	180	8	500	73.10%	95.74%
MinTsC_L_pmv6_r4_t25_25_bp3000	140	6	500	22.80%	88.67%

Tabla A.2. Porcentajes de FBT y PBT de la tarea del MiniTurista no categorizada con traductores con distintas características, usando codificaciones extraídas de MLP, tras distintos números de iteraciones (1000 y 3000).

Nombre	| Castellano| / |Inglés|	FBT	PBT
T_V_pmv4_r2_pS_t110_89_bp3000	110/89	0.0%	23.8%
T_V_pmv6_r4_pS_t121_94_bp3000	121/94	0.0%	34.0%
T_V_pmv8_r4_pS_t137_88_bp3000	137/88	0.0%	32.3%
T_V_pmv8_r6_pS_t132_93_bp3000	132/93	0.0%	40.6%

Tabla A.3. Porcentajes de PBT y FBT para la tarea del Turista obtenidos con RECONTRA con ventana de entrada de tamaño 9 y 450 unidades en la capa oculta usando codificaciones extraídas de MLPs podados con distintos formatos de ventana de salida y codificaciones iniciales distribuidas.

Nombre	FBT	PBT
T_VI_pmv4_r2_pS_t156_115_bp3000	0.0%	15.2%
T_VI_pmv4_r2_t172_129_bp3000	0.0%	7.5%

Tabla A.4. Porcentajes de PBT y FBT para la tarea del Turista obtenidos con RECONTRA con ventana de entrada de tamaño 9 y 450 unidades en la capa oculta usando codificaciones extraídas de MLPs podados con distintos formatos de ventana de salida y codificaciones iniciales distribuidas.

Otras topologías de los MLPs

Como ya se comentó en el apartado 5.3.2 se realizaron algunos experimentos con MLP con ventana de entrada y de salida, siguiendo la topología mostrada en la Figura 5.3[62].

Así, para los MLPs tenemos las siguientes características:

- Tres palabras en la ventana de entrada, sin repetición y con 30 unidades para cada palabra (codificaciones iniciales *T_I_t30_30*).
- Cada oculta de 30 unidades, sin poda.
- Ventana de salida igual a la ventana de entrada, con una palabra de contexto a izquierda y derecha (ventana de tamaño 3). A veces con repetición de la palabra más importante a la salida, obteniendo una ventana de tamaño 4.
- Algoritmo de entrenamiento de Retropropagación durante 3000 iteraciones tras estimación de parámetros.

Se adoptaron dos interpretaciones de las codificaciones desarrolladas:

1. La codificación extraída representa toda la entrada al MLP, de la cual utilizamos únicamente la palabra central *x*, ignorando su contexto (opción *1a* en Tabla A.5). También se utilizó este contexto en un sistema de votación (si dos de las tres posibles palabras para esa posición son iguales, esta es la palabra producida, sino, es la palabra central *x*) para decidir cual es la palabra generada por el sistema, como puede verse gráficamente en la Figura A.1.

[62] Con ventanas de entrada distintas, con un contexto mayor, por ejemplo. Ni siquiera se consideró necesario repetir esta experimentación con codificaciones locales.

2. La codificación extraída representa toda la entrada, es decir, el trío de palabras *x-1 x x+1* (opción 2). Esto supone que a la hora de traducir, ciertas combinaciones de palabras que aparecen en el conjunto de prueba y no aparecían en el conjunto de entrenamiento no van a tener representación, y por tanto no van a poder traducirse.

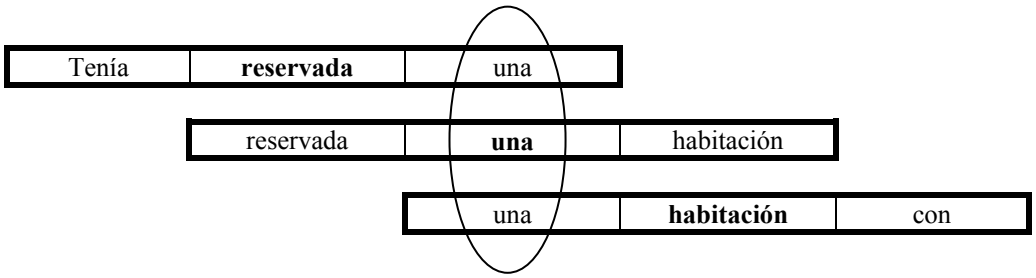

Figura A.1. Ejemplo gráfico del sistema de votación *1b* con el fragmento de frase en castellano *Tenía reservada una habitación con vistas a ...* y tres palabras "triples" de salida. Dentro del óvalo aparecen las tres palabras que aportan su voto, en negrita aparece la palabra central producida por cada activación del traductor, que tiene preferencia a la hora de seleccionar su salida.

En la Tabla A.5 se pueden observar los resultados. Las dos variantes (con repetición o no de la palabra de entrada central en la ventana de salida) no producen resultados significativamente distintos, pero sí inferiores a las producidas por codificaciones generadas por MLP con ventana de salida y repetición.

Nombre	Opción	FBT	PBT
T_I_t30_30	-	11.0%	63.4%
T_I_pmv4_r2_t30_30_bp3000	-	20.7%	76.5%
T_I_pmv3_v3_t30_30_bp3000	1ª	10.1%	63.5%
T_I_pmv3_v3_t30_30_bp3000	1b	6.8%	55.8%
T_I_pmv3_v3_t30_30_bp3000	2	4.1%	60.6%
T_I_pmv3_v4_r2_t30_30_bp3000	1ª	10.0%	64.8%
T_I_pmv3_v4_r2_t30_30_bp3000	1b	3.6%	57.7%
T_I_pmv3_v4_r2_t30_30_bp3000	2	1.0%	56.6%

Tabla A.5. Porcentajes de FBT y PBT tras 150 iteraciones de entrenamiento para la tarea del Turista con RECONTRA con 9 palabras de entrada y 450 unidades en la capa oculta.

Se realizaron nuevos experimentos con las mismas codificaciones y modelos RECONTRA con una característica diferente: la ventana de entrada tiene un tamaño de 3 palabras. Esto hace que para la opción 2, RECONTRA vea a la entrada 9 palabras reales de la frase, con lo que se puede considerar más parecido a los experimentos anteriores.

En la Tabla A.6 se pueden observar los resultados con RECONTRA, que son en general un poco mejores que los mostrados en la Tabla A.5.

Nombre	Opción	FBT	PBT
T_I_pmv3_v3_t30_30_bp3000	1a	14.0%	61.3%
T_I_pmv3_v3_t30_30_bp3000	1b	13.7%	59.7%
T_I_pmv3_v3_t30_30_bp3000	2	8.9%	65.1%
T_I_pmv3_v4_r2_t30_30_bp3000	1a	15.1%	64.6%
T_I_pmv3_v4_r2_t30_30_bp3000	1b	15.1%	64.5%
T_I_pmv3_v4_r2_t30_30_bp3000	2	4.2%	63.9%

Tabla A.6. Porcentajes de PBT para la tarea del Turista con RECONTRA con 3 palabras de entrada y 450 unidades en la capa oculta.

Reducción de parámetros durante el entrenamiento de los MLPs

Se realizaron experimentos para reducir los parámetros de Retropropagación en función del número de iteraciones de entrenamiento.

Inicialmente se realizaron varios experimentos en los cuales se redujeron a un 10% de sus valores iníciales los parámetros de entrenamiento de los MLP tras 1000 y 2000 iteraciones. Los resultados se pueden observar en la Tabla A.7. Puede verse que no se produce la mejora esperada.

Nombre	Reducción	FBT	PBT
T_II_pmv4_r2_t30_30_bp3000	No	22.4%	75.3%
T_II_pmv4_r2_t30_30_bp3000reduc	Sí	19.5%	71.2%
T_II_pmv4_r2_t30_30_bp3000reducEP	Sí (con EP)	21.0%	74.6%
T_III_pmv4_r2_t30_30_bp3000	No	22.4%	75.8%
T_III_pmv4_r2_t30_30_bp3000reduc	Sí	20.0%	72.4%
T_V_pmv4_r2_pS_t110_89_bp3000	No	0.0%	23.8%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	No	7.9%	70,3%
T_V_pmv4_r2_pS_t110_89_bp3000reduc	Sí	0.0%	30.1%
T_V_pmv8_r6_pS_t132_93_bp3000	No	0.1%	40.6%
T_V_pmv8_r6_pS_t132_93_bp100_scg600_600	No	17.8%	73.8%
T_V_pmv8_r6_pS_t132_93_bp3000reduc	Sí	0.1%	45.6%

Tabla A.7. Resultados de traducción al reducir en puntos fijos del entrenamiento el factor de aprendizaje y el momentum en los MLPs. También se incluyen los obtenidos con BP + SCG.

Se emplearon las ecuaciones (6.17) y (6.18). Se realizaron experimentos con MLPs con codificaciones iniciales distribuidas al azar y con y sin poda. La nomenclatura seguida añade "f1" o "f2" tras el número de iteraciones del MLP para indicar la ecuación de modificación de parámetros empleada.

En la Tabla A.8 pueden verse los resultados obtenidos con estos métodos, así como los resultados originales con Retropropagación y con la combinación de BP + SCG.

Se realizaron más experimentos con distintos valores de λ, en un intento de comprender porqué no se producían diferencias más significativas en los resultados con los otros métodos de entrenamiento.

Los resultados pueden verse en la Tabla A.9 y confirman que los métodos (6.17) y (6.18) pueden tener problemas para encontrar un buen mínimo local, y que un valor de λ adecuado puede paliar o acentuar este problema.

Nombre	Métodos	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	BP	20.7%	76.5%
T_I_pmv4_r2_t30_30_bp3000f1	(6.17)	10.1%	68.2%
T_I_pmv4_r2_t30_30_bp3000f2	(6.18)	10.7%	70.8%
T_I_pmv4_r2_pS_t17_14_bp3000	BP	6.8%	52.5%
T_I_pmv4_r2_pS_t17_14_bp3000f1	(6.17)	8.1%	59.4%
T_I_pmv4_r2_pS_t17_14_bp3000f2	(6.18)	8.5%	61.5%
T_VIII_pmv4_r2_t30_30_bp3000	BP	12.6%	68.9%
T_VIII_pmv4_r2_t30_30_bp3000f1	(6.17)	13.1%	67.5%
T_VIII_pmv4_r2_t30_30_bp3000f2	(6.18)	15.2%	66.5%
T_V_pmv4_r2_pS_t110_89_bp3000	BP	0.0%	23.8%
T_V_pmv4_r2_pS_t110_89_bp100_scg2900	BP+SCG	2.3%	61.5%
T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	BP+SCG	7.9%	70.3%
T_V_pmv4_r2_pS_t110_89_bp3000f1	(6.17)	25.9%	79.7%
T_V_pmv4_r2_pS_t110_89_bp3000f2	(6.18)	0.0%	27.5%
T_V_pmv6_r4_pS_t121_94_bp3000	BP	0.0%	34.0%
T_V_pmv6_r4_pS_t121_94_bp3000f1	(6.17)	0.1%	45.0%
T_V_pmv6_r4_pS_t121_94_bp3000f2	(6.18)	0.0%	32.7%

Tabla A.8. Resultados de traducción utilizando codificaciones obtenidas de MLP podados con codificaciones iníciales y entrenados con BP con dos métodos para reducir los parámetros de entrenamiento: (6.17) y (6.18) y λ con el mismo valor que el tamaño de los conjuntos de entrenamiento, así como los resultados originales sin modificación de parámetros.

Λ valor	Codificaciones	Método	FBT	PBT
-	T_I_pmv4_r2_pS_t17_14_bp3000	BP	6.8%	52.5%
100	T_I_pmv4_r2_pS_t17_14_bp3000f1_100	(6.17)	8.5%	59.8%
1000	T_I_pmv4_r2_pS_t17_14_bp3000f1_1000	(6.17)	7.8%	57.7%
Tamaño	T_I_pmv4_r2_pS_t17_14_bp3000f1	(6.17)	8.1%	59.4%
10	T_I_pmv4_r2_pS_t17_14_bp3000f2_100	(6.18)	7.7%	60.2%
1000	T_I_pmv4_r2_pS_t17_14_bp3000f2_1000	(6.18)	6.5%	60.4%
Tamaño	T_I_pmv4_r2_pS_t17_14_bp3000f2	(6.18)	8.5%	61.5%
-	T_V_pmv4_r2_pS_t110_89_bp3000	BP	0.0%	23.8%
-	T_V_pmv4_r2_pS_t110_89_bp100_scg2900	BP+SCG	2.3%	61.5%
-	T_V_pmv4_r2_pS_t110_89_bp100_scg600_900	BP+SCG	7,9%	70,3%
1000	T_V_pmv4_r2_pS_t110_89_bp3000f1_1000	(6.17)	34.8%	81.1%
Tamaño	T_V_pmv4_r2_pS_t110_89_bp3000f1	(6.17)	25.9%	79.7%
10	T_V_pmv4_r2_pS_t110_89_bp3000f2_10	(6.18)	35.7%	80.8%
100	T_V_pmv4_r2_pS_t110_89_bp3000f2_100	(6.18)	24.9%	79.2%
1000	T_V_pmv4_r2_pS_t110_89_bp3000f2_1000	(6.18)	0.0%	26.2%
Tamaño	T_V_pmv4_r2_pS_t110_89_bp3000f2	(6.18)	0.0%	27.5%
-	T_V_pmv8_r4_pS_t137_88_bp3000	BP	0.0%	32.3%
-	T_V_pmv8_r4_pS_t137_88_bp100_scg	BP+SCG	5.6%	65.7%
Tamaño	T_V_pmv8_r4_pS_t137_88_bp3000_f1	(6.17)	0.1%	51.7%
100	T_V_pmv8_r4_pS_t137_88_bp3000f2_100	(6.18)	0.1%	49.4%
1000	T_V_pmv8_r4_pS_t137_88_bp3000f2_1000	(6.18)	0.0%	28.1%
Tamaño	T_V_pmv8_r4_pS_t137_88_bp3000f2	(6.18)	0.0%	22.7%
-	T_V_pmv8_r6_pS_t132_93_bp3000	BP	0.1%	40.6%
-	T_V_pmv8_r6_pS_t132_93_bp100_scg600_600	BP+SCG	17.8%	73.8%
Tamaño	T_V_pmv8_r6_pS_t132_93_bp3000f1	(6.17)	13.5%	74.9%
1000	T_V_pmv8_r6_pS_t132_93_bp3000f2_1000	(6.18)	0.0%	24.4%
Tamaño	T_V_pmv8_r6_pS_t132_93_bp3000f2	(6.18)	0.0%	33.8%

Tabla A.9. Resultados de traducción con RECONTRA y codificaciones extraidas de MLP entrenados con distintos métodos (nombre de las codificaciones y los métodos empleados pueden verse en la tabla).

Con la expresión (6.17), al trabajar sobre los parámetros inmediatamente anteriores, tiende a disminuirlos demasiado rápido, con lo que las redes se quedan atascadas en el primer (y peor) mínimo local. Como al aumentar el valor de λ disminuye la reducción, se obtienen resultados mejores, pero sólo similares a los obtenidos sin reducción. Si hacemos que λ tienda a infinito, los parámetros variarán muy poco con cada iteración, sólo lo suficiente para que en las "últimas" se oscile menos en torno al ECM y se obtengan mejores resultados.

La expresión (6.18) soluciona parte de los problemas que presenta (6.17). Al reducirse siempre en función de los parámetros iniciales, no tiende a quedarse atascada en el primer mínimo local, y los valores de los parámetros no se reducen mucho hasta muy avanzado el entrenamiento, cuando se supone que la red ya esta oscilando en torno a un mínimo local mejor o por lo menos más cerca de él.

Codificaciones iguales en codificaciones automáticas

Uno de los aspectos que se revelaron como determinantes en las codificaciones para obtener buenos resultados de traducción, era que las palabras similares en los vocabularios de origen y destino tuviesen las mismas codificaciones [Castaño, 1998]. Sin embargo, establecer esta equivalencia entre palabras no es evidente, dado que no es siempre un proceso unívoco. Además, si se trabaja con codificaciones automáticas se presentan una serie de problemas añadidos:

- Dado que las codificaciones de los dos vocabularios se generan en dos MLP distintos, podemos tener codificaciones iguales (desarrolladas independientemente) para palabras que no tienen ninguna relación.
- El tamaño de las codificaciones en el lenguaje origen puede no ser el mismo que el lenguaje destino (por el proceso de poda).
- Las representaciones internas desarrolladas por los MLP deben tener en cuenta las codificaciones que queremos que se repitan en los dos vocabularios. Como posible solución se podrían adoptar las codificaciones finales en un idioma para el entrenamiento de los MLPs del otro idioma [Bengio, 2003a].

Se realizaron diversos experimentos con la tarea Turista que sólo utilizaban las mismas codificaciones en nombres propios (dado que la equivalencia es evidente) y se logró un muy ligero incremento (1-2% de PBT) en los resultados sobre las codificaciones equivalentes extraídas de MLPs.

Codificaciones mixtas: añadiendo unidades a las codificaciones automáticas

En muchos casos, los errores de traducción parecen presentarse en una serie de palabras que el traductor confunde entre sí consistentemente. Al entrenar MLP para general representaciones de los vocabularios, puede suceder que estas palabras tengan codificaciones muy similares y que el traductor tenga dificultades en diferenciar entre

ellas. Para Turista los resultados no son buenos debido a la existencia de un gran número de nombres propios distintos en los vocabularios, que aparecen en un bajo número de ocasiones en el conjunto de entrenamiento.

Se añadieron varias unidades binarias a codificaciones automáticas preexistentes con la intención de facilitar al traductor la tarea de distinguir entre unos nombres y otros, mientras que en el resto de palabras estas unidades adicionales tomaban el valor cero. Los resultados mostrados en la Tabla A.10 fueron variados dado que las nuevas unidades parecían confundir al traductor.

Nombre	Unidades Extras	FBT	PBT
T_L_pmv4_r2_pS_t30_30_bp3000	No	8.4%	60.7%
T_L_pmv4_r2_pS_t30_30_bp3000	Sí	7.4%	57.3%
T_L_pmv4_r2_pS_t30_30_bp_R_bp3000	No	15.4%	63.3%
T_L_pmv4_r2_pS_t30_30_bp_R_bp3000	Sí	14.4%	64.5%
T_L_pmv4_r2_pS_t30_30_bp_R_bp_scg1100_1200	No	10.5%	58.5%
T_L_pmv4_r2_pS_t30_30_bp_R_bp_scg1100_1200	Sí	9.5%	57.4%
T_V_pmv8_r6_pS_t132_93_bp100_scg600_600	No	17.8%	73.8%
T_V_pmv8_r6_pS_t132_93_bp100_scg600_600	Sí	5.0%	59.7%

Tabla A.10. Porcentajes de FBT y PBT obtenidos con el traductor RECONTRA para Turista con diversas codificaciones distribuidas extraídas de MLP y para estas mismas codificaciones con 8 unidades extras para diferenciar entre Nombres Propios.

También se entrenaron MLPs para que generaran automáticamente estas 8 unidades. Los resultados de traducción obtenidos (Tabla A.11) fueron similares. Como el tamaño de las codificaciones podía ser demasiado pequeño, se realizó un nuevo experimento con 30 unidades en el MLP, pero los resultados no mejoraron.

Método de entrenamiento	Unidades Extras	FBT	PBT
-	0	8.4%	60.7%
BP 3000 iteraciones	8	14.5%	64.1%
BP 100 SCG 1200/3000 iteraciones	8	14.3%	63.9%
BP 3000	30	8.6%	64.0%

Tabla A.11. Porcentajes de FBT y PBT obtenidos tras 150 iteraciones de entrenamiento a RECONTRA para Turista con diversas codificaciones (8 o 30 unidades) generadas a partir de las codificaciones *T_L_pmv4_r2_pS_t30_30_bp3000* y añadidas a estas para ayudar a distinguir entre Nombres Propios, obtenidas de MLPs con ventana de salida de tamaño 4.

Dados los resultados obtenidos no se realizaron experimentos con otras tareas, a pesar de que la idea de tener segmentos de las codificaciones dedicadas a un subconjunto concreto del vocabulario suena potencialmente interesante.

Utilización de FGREP como codificador

Para la tarea del Turista se realizaron diversos experimentos de traducción utilizando codificaciones producidas con el método FGREP (ya explicado en el capítulo 3). Se emplearon codificaciones del mismo tamaño que en experimentos previos (30/30, 44/43 y 121/94) y diversos tamaños de la capa ocultas (30, 50, 80, 120 y 200) así como diversas

iteraciones del conjunto de entrenamiento (100, 300, 1000, 2000 y hasta un máximo de 3,000, la misma cantidad utilizada en el entrenamiento de los MLPs).

En un primer momento se utilizaron redes FGREP con la siguiente topología: 3 palabras a la entrada (*x x+1 x+2*) y 4 a la salida (*x x+1 x+2 x+3*) en FGREP. En la tabla A.12 podemos ver los resultados de traducción obtenidos.

Como podemos comprobar, incluso las codificaciones al azar de este tamaño obtienen mejores resultados de traducción que las obtenidas mediante FGREP con el mismo número de iteraciones (3,000 iteraciones). Es sólo cuando deliberadamente entrenamos muy poco la red cuando FGREP ofrece resultados más aceptables (el número de unidades en la capa oculta no parece ser un factor importante).

Se adoptó una nueva topología para FGREP, exactamente la misma que en los experimentos originales de Miikkulainen para la tarea de predicción: así, una 3-tupla se presenta a la red que tiene que producir la siguiente palabra (*x x+1 x+2 / x+3*).

Los resultados de traducción (en la Tabla A.13) son mejores que los obtenidos con FGREP con topología 3-tupla / 4-tupla y en algunos casos mejores que los obtenidos con MLPs con codificaciones del mismo tamaño. Sin embargo los resultados aún son peores que los obtenidos con MLPs con realimentación, que tiene una estructura lógica más similar a FGREP.

Nombre	|CO|	Iteraciones	Inicialización	FBT	PBT
T_Fgrepv3_4_50_t30_30_bp100	50	100	-	3.6%	64.6%
T_Fgrepv3_4_50_t30_30_bp300	50	300	-	1.6%	55.3%
T_Fgrepv3_4_50_t30_30_bp1000	50	1000	-	1.3%	55.5%
T_Fgrepv3_4_50_t30_30_bp2000	50	2000	-	1.4%	48.7%
T_Fgrepv3_4_50_t30_30_bp3000	50	3000	-	0.5%	43.9%
T_Fgrepv3_4_50_t30_30_bp3000_i	50	3000	Distinta	0.7%	41.9%
T_Fgrepv3_4_50_t30_30_bp3000_i	50	3000	Distinta	0.6%	40.6%
T_Fgrepv3_4_30_t30_30_bp100	30	100	--	2.9%	60.2%
T_Fgrepv3_4_30_t30_30_bp3000	30	3000	-	0.7%	44.9%
T_Fgrepv3_4_80_t30_30_bp100	80	100	-	0.8%	48.4%
T_Fgrepv3_4_80_t30_30_bp3000	80	3000	-	0.0%	25.7%
T_Fgrepv3_4_120_t30_30_bp100	120	100	-	0.4%	39.7%
T_Fgrepv3_4_120_t30_30_bp3000	120	3000	-	0.0%	13.9%
T_Fgrepv3_4_200_t30_30_bp100	200	100	-	0.2%	31.3%
T_Fgrepv3_4_200_t30_30_bp3000	200	3000	-	0.0%	16.9%

Tabla A.12. Resultados de traducción con codificaciones generadas por FGREP con *topología x x+1 x+2 # x x+1 x+2 x+3*.

Estos nuevos experimentos (codificaciones) no muestran la clara dependencia del número de iteraciones de entrenamiento que muestran los anteriores.

Nombre	|CO|	BP	FBT	PBT
T_Fgrepv3_1_50_t30_30_bp100	50	100	7.0%	67.9%
T_Fgrepv3_1_50_t30_30_bp300	50	300	3.8%	65.4%
T_Fgrepv3_1_50_t30_30_bp600	50	600	5.8%	66.3%
T_Fgrepv3_1_50_t30_30_bp1000	50	1000	8.4%	68.7%
T_Fgrepv3_1_50_t30_30_bp2000	50	2000	9.7%	69.8%
T_Fgrepv3_1_50_t30_30_bp3000	50	3000	8.0%	67.2%
T_Fgrepv3_1_80_t30_30_bp100	80	100	10.5%	68.0%
T_Fgrepv3_1_80_t30_30_bp3000	80	3000	7.0%	68.1%
T_Fgrepv3_1_200_t30_30_bp100	200	100	5.8%	66.4%
T_Fgrepv3_1_200_t30_30_bp3000	200	3000	6.9%	68.6%
T_Fgrepv3_1_50_t44_43_bp100	50	100	10.0%	68.4%
T_Fgrepv3_1_50_t44_43_bp300	50	300	11.6%	69.7%
T_Fgrepv3_1_50_t44_43_bp600	50	600	8.8%	69.0%
T_Fgrepv3_1_50_t44_43_bp1000	50	1000	7.5%	70.1%
T_Fgrepv3_1_50_t44_43_bp2000	50	2000	10.4%	69.4%
T_Fgrepv3_1_50_t44_43_bp3000	50	3000	8.2%	70.4%
T_Fgrepv3_1_50_t121_94_bp100	50	100	26.9%	78.9%
T_Fgrepv3_1_50_t121_94_bp3000	50	3000	18.2%	74.9%
T_Fgrepv3_1_200_t121_94_bp100	200	100	28.2%	81.2%
T_Fgrepv3_1_200_t121_94_bp300	200	300	6.3%	69.1%
T_Fgrepv3_1_200_t121_94_bp600	200	600	12.1%	72.1%
T_Fgrepv3_1_200_t121_94_bp1000	200	1000	15.8%	73.6%
T_Fgrepv3_1_200_t121_94_bp2000	200	2000	9.9%	71.5%
T_Fgrepv3_1_200_t121_94_bp3000	200	3000	16.9%	75.4%

Tabla A.13. Resultados de traducción con codificaciones generadas por FGREP con *topología x x+1 x+2 # x+3*.

El factor determinante en los resultados parece ser la salida de la red FGREP[63]. Cuando se utiliza la salida *x x+1 x+2 x+3* (en la Tabla A.12), los resultados son peores que con MLPs. Sin embargo, los resultados con salida *x+3* son mejores (Tabla A.13), aunque aún son peores que los obtenidos con MLPs con realimentación.

Por último, se utilizaron dos topologías para FGREP extraídas directamente de MLPs: dada una palabra de entrada, FGREP debía producir a la salida esta palabra y la palabra precedente y la posterior. Los resultados de traducción pueden verse en la Tabla A.14, y permiten concluir con seguridad que FGREP es un método peor que MLPs con ventanas de salida para obtener codificaciones con propósitos de traducción.

Nombre	Ventana de salida	Iteraciones	FBT	PBT
Fgrepv1_3_50_t30_30_bp100	x-1 **x** x+1	100	4.0%	58.4%
Fgrepv1_3_50_t30_30_bp3000	x-1 **x** x+1	3000	2.2%	53.6%
Fgrepv1_4_50_t30_30_bp100	x-1 **x x** x+1	100	5.1%	64.8%
Fgrepv1_4_50_t30_30_bp3000	x-1 **x x** x+1	3000	2.7%	59.5%

Tabla A.14. Resultados de traducción con codificaciones generadas *por FGREP con topologías x # x-1 **x** x+1 y x # x-1 **x x** x+1* y 50 unidades ocultas.

Se realizó un único experimento para Hansards con codificaciones producidas por FGREP. Los resultados (en la Tabla A.15) sugieren que el método FGREP obtiene

[63] Aunque muchas iteraciones provocan peores resdultados, dado que las codificaciones de palabras con las mismas funciones sintácticas se hacen demasiado similares.

resultados similares a utilizar codificaciones al azar del mismo tamaño. Sin embargo, seguramente esto se debe a que el tamaño de las codificaciones es demasiado pequeño para abordar el problema con éxito.

Nombre	Ventana de salida	|Frances|/|Inglés|	FBT	PBT
H_Fgrepv1_4_50_t30_30_bp3000	x-1 **x x** x+1	30/30	0.0%	26.85%

Tabla A.15. Resultados con codificación generadas *con FGREP para la tarea Hansards*.

Se ha comprobado que una posibilidad para obtener codificaciones más adecuadas para tareas de traducción es modificar la salida de la red, creando una ventana de salida, es decir, aproximando los modelos FGREP a los MLP desarrollados en este trabajo. Sin embargo no parecen suponer una ventaja.

Distintos métodos de entrenamiento de RECONTRA

Se probaron dos métodos clásicos de entrenamiento de RECONTRA, modificados para tener en cuenta la recurrencia de la red: Quickprop (explicado en el apartado 6.3.2) y Rprop (explicada en el apartado 6.1.2[64]).

Los mejores resultados para la tarea del Mini Turista sin categorías pueden verse en la Tabla A.16. Con el método Rprop los resultados fueron peores que los obtenidos con Retropropagación, y en el caso de Quickprop, la red se vio incapaz de aprender.

Algoritmo de entrenamiento	FBT	PBT
Retropropagación	51.1%	92.2%
Quickprop	0.0%	0.3%
Rprop	27.0%	88.3%

Tabla A.16. Resultados de traducción para RECONTRA con ventana de entrada de 8 palabras y 180 unidades ocultas tras 500 iteraciones de entrenamiento con la codificación *MinTsC_L_pmv8_r4_pSep_14_14_bp3000* y Quickprop, Rprop y Retropropagación.

Como se puede observar en la tabla A.17 para la tarea del Turista, se repite el comportamiento de la tarea del Mini Turista sin categorías: con Rprop los resultados fueron peores que los obtenidos con Retropropagación, y la red no aprendió con Quickprop. Se repitieron los experimentos con distintas inicializaciones de la red, sin mejoras.

Algoritmo de entrenamiento	FBT	PBT
Retropropagación	20.7%	76.5%
Quickprop	0.0%	0.1%
Rprop	0.0%	26.0%

Tabla A.17. Los mejores resultados con distintas inicializaciones para RECONTRA con ventana de entrada 9 y 450 unidades ocultas con codificaciones *T_I_pmv4_r2_t30_30_bp3000 y métodos Quickprop, Rprop y Retropropagación*.

[64] En el apartado 6.1.2 se explica Rprop para su uso con MLP. La única diferencia es el truncamiento del error en las conexiones recurrentes, como en Retropropagación.

Con la tarea Hansards sólo se realizaron experimentos con el método Rprop. Los resultados (en la Tabla A.18) son peores que los obtenidos con Retropropagación y confirmaron que este método de entrenamiento no es adecuado para traducción.

Algoritmo de entrenamiento	FBT	PBT
Retropropagación	33.83%	32.90%
Rprop	9.61%	18.65%

Tabla A.18. RECONTRA con ventana de entrada 5 y 900 unidades ocultas para dos métodos distintos de entrenamiento con codificaciones *H_III_pmv4_r2_t60_60_bp100_scg2100_1400 y Rprop y Retropropagación*.

A.3. Experimentación adicional relacionada con el Capítulo 9

Realimentación de la capa de salida

Se realizaron experimentos utilizando redes con realimentación de la capa de salida sobre sí misma con la tarea del Turista con el fin de comprobar experimentalmente su comportamiento. La realimentación en la capa de salida puede interpretarse como un intento de forzar a la capa a crea un modelo del lenguaje destino, lo cual podría mejorar los resultados de traducción.

Se emplearon redes con 1 y 2 capas ocultas sin resultados satisfactorios (Tabla A.19).

| 1ª CO |	| 2ª CO |	Realimentación Salida	FBT	PBT
-	450	No	11.0%	63.4%
-	450	Sí	3.2%	45.4%
270	450	No	18.1%	71.3%
270	450	Sí	3.1%	51.0%

Tabla A.19. Resultados de traducción para la tarea Turista con modelos RECONTRA con o sin realimentación de la capa de salida, una o dos capa ocultas y ventana de entrada de tamaño 9 y codificaciones *T_I_t30_30*.

También se realizaron experimentos (en la Tabla A.20) con modelos RECONTRA con dos palabras de salida y realimentación (figura A.2).

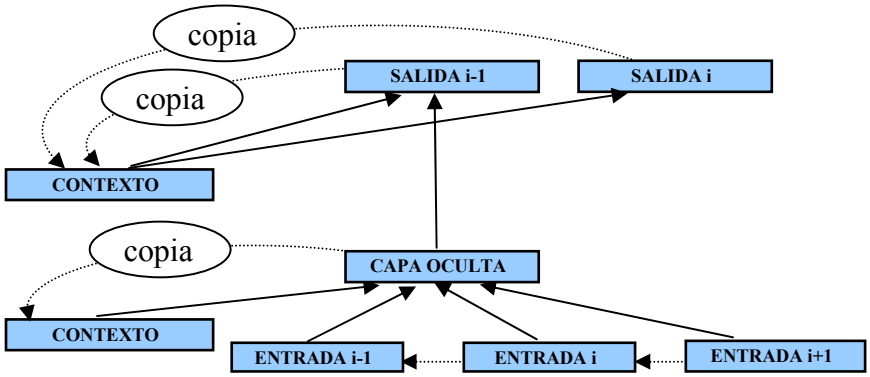

Figura A.2. RECONTRA con dos palabras de salida y realimentación de la capa de salida.

Nombre	Realimentación	2ª	1ª
T_I_t30_30	No	-	63.4%
T_I_t30_30	Sí	-	40.8%
T_I_t30_30	Sí	44.3%	45.2%

Tabla A.20. Resultados de traducción (PBT) para redes RECONTRA con *realimentación en la capa de salida,* una capa oculta y una o dos palabras de salida y codificaciones distintas, *I_t30_30,* así como los resultados con RECONTRA básico.

Modelos de Bengio

Los modelos de Bengio son MLPs con una diferencia en la conectividad de las neuronas (conectividad de-izquierda-a-derecha entre la capa de entrada y la capa de salida). Aunque no resuelven todos los problemas de los MLPs clásicos, han demostrado ser más adecuados para tratar secuencias. Con propósitos de comparación, en esta sección intentamos aplicarlos directamente a tareas de traducción para la tarea del Turista.

La primera capa oculta debería desarrollar algún tipo de representación de la entrada que la segunda capa oculta interpretaría para generar eventualmente la salida. Los mejores resultados obtenidos pueden observarse en la Tabla A.21 y son bastante peores a los resultados obtenidos con modelos RECONTRA.

Los resultados parecen indicar que, aunque Bengio es incapaz de resolver la tarea con éxito, presenta ventajas respecto a un MLP "clásico".

Tipo de red	| 1ª CO|	| 2ª CO |	| 3ª CO |	FBT	PBT
MLP Bengio	270	450	-	0.0%	33.0%
MLP Bengio	270	450	30	0.0%	4.5%
MLP clásico	270	450	-	0.0%	0.0%

Tabla A.21. Mejores porcentajes de FBT y PBT obtenidos con modelos de Bengio y un MLP con ventana de entrada de tamaño 9 y codificaciones locales (*T_Local*).

Se emplearon codificaciones distribuidas extraídas de MLPs, para asegurar que el comportamiento de este tipo de redes era consistente y no provocado por su relativo gran tamaño. Los mejores resultados obtenidos pueden verse en la Tabla A.22 y muestran una mejora considerable, aunque inferior a RECONTRA.

Tipo de red	| 1ª CO|	| 2ª CO |	FBT	PBT
RECONTRA	-	450	20.7%	76.5%
MLP Bengio	270	450	1.7%	56.6%

Tabla A.22. Mejores porcentajes de FBT y PBT obtenidos con modelos de Bengio con ventana de entrada de tamaño 9 y 270 y 450 unidades en sus capas ocultas y codificaciones *T_I_pmv4_r2_t30_30_bp3000*.

El modelo de Bengio no es apropiado para tareas de traducción, aunque por lo menos es mejor que utilizar un MLP "clásico" con las mismas características.

RECONTRA y modelos de Bengio

Se exploró la utilización de una topología distinta en RECONTRA, basado en los modelos de Bengio. Así, se creó una variante de RECONTRA en que la capa de entrada y la primera capa oculta tienen una conexión "de izquierda a derecha".

Los resultados obtenidos pueden observarse en la Tabla A.23, y si se comparan con los resultados para modelos de Bengio y modelos RECONTRA con dos capas ocultas de los mismos tamaños, se puede observar que mejoran respecto al MLP, como ya se esperaba, pero que no hay una diferencia sustancial con RECONTRA sin esta conexión.

Red	| 1ª CO |	| 2ª CO |	FBT	PBT
RECONTRA	-	450	20.7%	76.5%
RECONTRA izq a der	-	450	6.2%	70.5%
RECONTRA	270	450	29.1%	80.9%
RECONTRA izq a der	270	450	28.9%	81.1%
MLPBengio	270	450	1.7%	56.6%

Tabla A.23. Resultados de traducción con modelos RECONTRA con conexión de izquierda a derecha y dos capas ocultas de 270 y 450 unidades, sin ella y con MLP con conexión de izquierda a derecha (Bengio) y codificaciones *T_I_pmv4_r2_t30_30_bp3000*.

Se realizaron experimentos con la tarea Hansards (en la Tabla A.24), y comparados con los resultados para modelos RECONTRA con una o dos capas ocultas de los mismos tamaños, se puede observar un descenso de resultados respecto a los modelos RECONTRA sin esta conexión especial.

Conexiones	| 1ª CO |	| 2ª CO |	FBT	PBT
Estándar	-	450	30.23%	33.22%
De izquierda a derecha	-	450	5.41%	13.38%
Estándar	270	450	36.93%	31.02%
De izquierda a derecha	270	450	22.12%	27.20%

Tabla A.24. Resultados de traducción con modelos RECONTRA "normal" con conexión de izquierda a derecha y dos capas ocultas de 270 y 450 unidades y codificaciones *H_I_t30_30*.

A.4. Experimentación adicional relacionada con el Capítulo 10

Integración de multiples salidas de una red

Se realizaron varios experimentos que exploraban la integración de las salidas de una red con multiples salidas (para la tarea Turista). En experimentación previa con los modelos RECONTRA con ventana de salida, se había escogido una de las palabras de salida como la producida por la red, aprovechando la tendencia de esta de ofrecer siempre mejores salidas en una de ellas. Se utilizaron modelos RECONTRA con dos palabras de salida y dos técnicas distintas, con y sin pesos:

1. **Media**, en el que se calculaba la media aritmética de las dos codificaciones de salida y
2. **Distancia**, en la que se comparan la distancia entre las salidas producidas por la red y las codificaciones más cercanas, conservando la palabra cuya codificación está a menor distancia.

Los resultados, con diversos valores de los pesos, pueden verse en la Tabla A.25 y muestran claramente que ninguno de los métodos empleados es adecuado. Los mejores resultados se obtienen cuando, mediante pesos se da tanta importancia a la palabra que

iba a proporcionar los mejores resultados (*Primera*) que prácticamente se ignora *Segunda*.

Metodo	Segundo Peso	Primer Peso	PBT
Media	0.1	0.9	85.9%
Media	0.3	0.7	83.9%
Media	0.5	0.5	54.7%
Media	0.7	0.3	25.4%
Media	0.9	0.1	23.0%
Distancia	0.1	0.9	85.2%
Distancia	0.3	0.7	85.2%
Distancia	0.5	0.5	84.9%
Distancia	0.7	0.3	83.8%
Distancia	0.9	0.1	83.5%

Tabla A.25. Resultados de traducción con RECONTRA extendido RECONTRA con dos palabras de salida y dos métodos distintos de integración para el par de codificaciones denominadas *T_V_pmv6_r4_pS_t121_94_bp100_scg600_600_R_bp3000*.

Métodos más complejos, desarrollados para el trabajo con ensembles ([Kuncheva, 2004]) podrían ofrecer mejores resultados, pero los componentes de los ensembles necesitan cumplir unas condiciones de diversidad para ser útiles y en este caso al trabajar con la misma red no se cumplen.

A.5. Experimentación adicional relacionada con el Capítulo 11

Integración de varias redes mediante ANNs

Se realizaron una serie de experimentos con la tarea Turista para combinar la salida de dos modelos RECONTRA mediante una nueva ANN. Se utilizaron redes de Elman de dos palabras de entrada (una por cada red), una capa oculta y una palabra de salida. En la tabla A.26 podemos ver los malos resultados para dos modelos RECONTRA distintos con las mismas codificaciones. Parece que el sistema no tiene suficiente información para decidirse por ninguna de ellas e intenta calcular algún tipo de media. Se podría cambiar el diseño del sistema para que la ventana de entrada del traductor también fuese una entrada de la red integradora. También se podría modificar de forma que las palabras anteriores a la actual fuesen presentadas a la red, así, la ANN integradora debería cual de las dos palabras propuestas es más apropiada para ser la siguiente de la serie.

Unidades Ocultas	FBT	PBT
60	0.0%	13.0%
450	0.0%	23.5%

Tabla A.26. Resultados de traducción para ensembles de dos redes RECONTRA con codificaciones *T_I_pmv4_r2_t30_30_bp3000* , distintas tipologías (9 palabras de entrada y 450 unidades en la capa oculta y el otro con dos capas ocultas, ambas de 450 unidades) integradas con redes de Elman de distintas características: 2 palabras de entrada, una de salida y 60 o 450 unidades en la capa oculta.

Con cualquiera de los dos métodos la red tendría más información para decidir cuál de las dos redes ofrece la mejor salida.

A.6. Conjuntos no equilibrados en RECONTRA

En las tareas de traducción se producen situaciones de desequilibrio. A nivel de palabra, hay palabras que aparecen mucho más a menudo que otras, y a nivel de frase sucede lo mismo. En este punto se intenta explorar la posibilidad de disminuir este desequilibrio para obtener mejores resultados de traducción.

Hay diversos métodos de compensar este desbalance. Para la tarea del Turista el que empleamos es el muy sencillo de repetir las frases en las cuales aparecen algunas palabras poco habituales en la tarea.

En concreto se crearon dos conjuntos: uno con los Nombres Propios (NP) de personas y otro con todos los Números (NUM). Estos conjuntos se utilizaron en conjunto (DUP) y por separado para entrenar los traductores (junto con el conjunto "Normal"), y como se puede ver en la Tabla A.27, generalmente se obtuvieron ligeramente mejores resultados de traducción que sin utilizarlos.

Nombre	Conjunto	FBT	PBT
T_I_t30_30	Normal	11.0%	63.4%
T_I_t30_30	NP	10.7%	63.9%
T_I_t30_30	NUM	14.4%	64.5%
T_I_t30_30	DUP	11.6%	62.8%
T_I_t30_30	DUP Orden	11.9%	64.0%
T_III_t30_30	Normal	11.1%	67.4%
T_III_t30_30	NP	11.0%	68.2%
T_III_t30_30	NUM	13.5%	70.0%
T_III_t30_30	DUP	12.3%	68.9%
T_II_t30_30	Normal	12.2%	67.5%
T_II_t30_30	NP	12.9%	68.5%
T_II_t30_30	NUM	14.9%	70.5%
T_II_t30_30	DUP	14.1%	69.1%
T_II_pmv4_r2_t30_30_bp3000	Normal	23.2%	75.3%
T_II_pmv4_r2_t30_30_bp3000	Duplicado NP	22.8%	75.4%
T_II_pmv4_r2_t30_30_bp3000	Triplicado NP	21.9%	75.0%
T_II_pmv4_r2_t30_30_bp3000	NUM	25.8%	76.1%
T_II_pmv4_r2_t30_30_bp3000	DUP	24.0%	76.1%
T_IV_pmv4_r2_pS_t49_46_bp3000	Normal	27.7%	79.7%
T_IV_pmv4_r2_pS_t49_46_bp3000	NP	26.2%	79.2%
T_IV_pmv4_r2_pS_t49_46_bp3000	NUM	28.0%	79.6%
T_IV_pmv4_r2_pS_t49_46_bp3000	DUP	26.8%	79.4%

Tabla A.27. Resultados de traducción con diversas codificaciones y *repitiendo frases en las cuales aparecen números o Nombres Propios*.

Es interesante señalar que se realizó un único experimento en el cual RECONTRA era entrenado unicamente con las frases en las cuales aparecen números. Los resultados (en la Tabla A.28) son muy curiosos, dado que no empeoran tan drásticamente como sería de esperar respecto a los originales.

Nombre	Conjunto	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	Solo NUM	3.8%	55.4%

Tabla A.28. Resultados de traducción con diversas codificaciones y *repitiendo frase en las cuales aparecen números o Nombres Propios*.

Otra posibilidad que en principio es interesante para reducir el desequilibrio es repetir las frases en las que aparecen las palabras más raras del vocabulario: las que aparecen menos veces en el conjunto de entrenamiento. Este método no necesita de un experto humano que decida que existen algunas palabras/frases que necesitan una atención especial.

Sin embargo si modificamos excesivamente la distribución de probabilidad del conjunto de entrenamiento respecto al de prueba, empeoraran los resultados. En otras palabras, si unas determinadas frases y/o palabras aparecen en muy pocas ocasiones en el conjunto de entrenamiento no tienen porque aparecer más veces en el conjunto de prueba. Se realizaron experimentos para las frases con palabras que aparecen 5, 10, 15 o 20 veces o menos en el conjunto de entrenamiento (resultados en la Tabla A.29) muestran claramente este fenómeno, con unas caídas en los resultados considerables.

Palabras con N aparaciones	FBT	PBT
5	9.9%	61.0%
10	9.0%	59.6%
15	8.7%	61.9%
20	8.6%	59.9%

Tabla A.29. Resultados de traducción con codificaciones *T_I_t30_30* y *repitiendo frase en las cuales aparecen las palabras con menos apariciones: 5, 10, 15 o 20 veces*.

Para la tarea del Turista, reducir el desequilibrio mediante la repetición de frases puede mejorar los resultados en algunos casos: al repetir los números. Consistentemente, duplicar las frases con números mejora ligeramente los resultados de traducción.

Queda pendiente un trabajo mucho más profundo sobre el tema. Para empezar habría que detectar exactamente que forma toma el problema del desequilibrio, y con que frases/palabras se presenta. Y por supuesto, habría que emplear (y probablemente desarrollar) otras técnicas para compensar este desequilibrio y obtener mejores resultados de traducción.

A.7. Modificación de parámetros de RECONTRA

Como ya se comentó en el apartado 6.1.8, la evolución de parámetros de una red puede ayudar a encontrar un mínimo local mejor y por tanto proporcionar mejores resultados. Esto se debe a que el factor de aprendizaje y el momentum determinan en las redes neuronales entrenadas con el método de Retropropagación la rapidez con la cual la red evoluciona hacia el mínimo de la función del error.

Como en el caso de los MLP, el problema que se presenta en el método que utilizamos para seleccionar los parámetros es que puede seleccionar unos valores que aunque inicialmente son adecuados (son los que se acercan más rápidamente al mínimo) con el paso del tiempo (de las iteraciones del conjunto de entrenamiento) provocan oscilaciones en torno a ese mínimo.

Se realizaron tres series de experimentos, empezando siempre con los parámetros obtenidos tras la estimación:

1. En una serie de experimentos los parámetros se reducían tras un número fijo de iteraciones de entrenamiento, facilitando la comparación con los experimentos en los cuales no se producía esta reducción. En concreto se dividían los parámetros (factor de aprendizaje y momentum) por 10 o por 2 y se continuaba entrenando las redes.
2. En la segunda serie de experimentos se redujeron los parámetros durante el entrenamiento en función del número de iteraciones, aplicando distintos métodos (formulas (6.17), (6.18), (A.1) y (A.2)).
3. Y en la última serie de experimentos, los parámetros se modificaron, reduciéndolos o aumentándolos en función de los cambios en el ECM residual (formula (6.19)).

Esta experimentación ha dado lugar a varias publicaciones, como [Casañ, 2007], [Casañ, 2008a] y [Casañ, 2008b].

Reducción de parámetros en momentos fijos

En los experimentos con la tarea del Turista, los parámetros se reducían tras un número fijo de iteraciones de entrenamiento.

En concreto se dividían los parámetros (factor de aprendizaje y Momentum) por 10 o 2 y se continuaba entrenando las redes. Como se puede ver en los resultados de la Tabla A.30, cuando esta reducción se producía tras 125 iteraciones, los resultados de traducción para 150 eran mejores que cuando no se producía. Cuando esta reducción se producía en momentos anteriores del entrenamiento, los resultados no eran tan buenos. Se extendió esta idea para reducir varias veces los valores de los parámetros (en 25, 75 y 127 iteraciones) pero dividiendo por dos los valores, no por diez. Los resultados, otra vez en la Tabla A.30 muestran como en algunos casos se producen mejoras respecto a los experimentos sin cambio de parámetros y en otros casos no es así.

Los resultados parecen confirmar es que si se inicia la reducción de parámetros demasiado pronto la red se queda atascada en un mínimo local ‘peor’. En cualquier caso parece evidente que decidir a priori en que punto van a modificarse los parámetros no es una buena política.

Nombre	Reducción	Iteración	FBT	PBT
T_L_pmv8_r4_pS_t44_43_bp_R2_bp3000	No	-	29.7%	79.4%
T_L_pmv8_r4_pS_t44_43_bp_R2_bp3000	1/10	25	25.4%	77.6%
T_L_pmv8_r4_pS_t44_43_bp_R2_bp3000	1/2	75	28.4%	79.5%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_bp3000	No	-	38.6%	83.7%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_bp3000	1/10	125	37.6%	84.9%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_bp3000	1/2	24	38.7%	84.3%
T_I_pmv4_r2_t30_30_bp3000	No	-	20.7%	76.5%
T_I_pmv4_r2_t30_30_bp3000	1/10	125	23.1%	77.3%
T_II_t30_30	No	-	12.2%	67.5%
T_II_t30_30	1/10	125	14.0%	69.4%
T_II_t30_30	1/10	25	10.3%	66.9%
T_IV_pmv4_r2_pS_t49_46_bp3000	No	-	27.7%	79.7%
T_IV_pmv4_r2_pS_t49_46_bp3000	1/10	125	28.4%	80.4%

Tabla A.30. Resultados de traducción con diversas codificaciones y *reducción de los parámetros de entrenamiento a un 10% en momentos fijos* de este y resultados sin reducción, para facilitar la comparación.

También se aplicó a modelos RECONTRA modificados, con resultados similares (Tabla A.31), donde se presentan los resultados para distintas codificaciones y redes con dos capas ocultas o varias palabras de salida.

Nombre	|1ª CO|	|Salida|	Iteración	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	270	3	-	32.8%	83.8%
T_I_pmv4_r2_t30_30_bp3000	270	3	100	34.5%	85.3%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_ bp3000	450	-	-	36.0%	84.5%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_ bp3000	450	-	125	36.6%	85.0%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_ bp3000	270	2	-	35.3%	86.0%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_ bp3000	270	2	100	35.8	86.5%

Tabla A.31. Resultados de traducción con diversas codificaciones y *redes con dos capas ocultas o varias palabras de salida* y *reducción de los parámetros de entrenamiento a un 10% en momentos fijos* de este y resultados sin reducción, para facilitar la comparación.

Reducción de parámetros en función de las iteraciones

En la siguiente serie de experimentos, tras la estimación de parámetros, estos se modificaron siguiendo cuatro formulas. En concreto, las formulas (6.17) y (6.18) que ya habían sido utilizadas con MLP, y (A.1) que utiliza dos parámetros, `C1` y `C2`, el número total de iteraciones (`N`) tras las cuales se planea detener el entrenamiento y la iteración actual, `t`, además del parámetro inicial, $\eta(0)$, para determinar el nuevo valor de los parámetros.

$$\eta(t) = \frac{\eta(0)}{\frac{t}{N/2} + \frac{C1}{\max\left(1, C1 - \frac{\max\left(0, C1 * \left(t - C2 * N\right)\right)}{\left(1 - C2\right) * N}\right)}} \qquad (A.1)$$

Y la formula (A.2) prescinde del número máximo de iteraciones y utiliza un parámetro `C` y un parámetro `τ`, además de la iteración actual, `t`, y el valor del parámetro inicial, `η(0)`, para determinar el nuevo valor de los parámetros.

$$\eta(t) = \eta(0) * \frac{1 + \frac{C}{\eta(0)} * \frac{t}{\tau}}{1 + \frac{C}{\eta(0)} * \frac{t}{\tau} + \tau * \frac{t^2}{\tau^2}} \qquad (A.2)$$

Las características de las redes son:

- Modelos RECONTRA con 9 palabras de entrada.
- Una capa oculta de 900 unidades ocultas o dos capas, de 270 y 450 unidades.
- Algoritmo de Retropropagación del Error, con parámetros modificados durante el entrenamiento.
- Estimación de parámetros de entrenamiento inicial (excepto en casos concretos).
- Diversas codificaciones distribuidas.

Los resultados con la fórmula (6.17), dos codificaciones distintas, y distintos valores del parámetro **λ** pueden observarse en la Tabla A.32. Si los comparamos con los resultados obtenidos sin emplear este método (también en la tabla) se puede observar como en general no se produce una mejora en la traducción, y que la selección del parámetro **λ** es fundamental en el funcionamiento del método. Es interesante observar que en algunos casos el ECM residual deja de evolucionar, y por tanto el entrenamiento se puede detener (y se detuvo) antes de 150 iteraciones.

Señalar que también se realizaron tres experimentos sin estimación de parámetros para Retropropagación. Al fin y al cabo teniendo un método para modificarlos durante el entrenamiento, deberían tener menos importancia. Pero los resultados fueron peores, como también se puede ver en la Tabla A.32.

Nombre	λ	*t* final	FBT	PBT
T_I_t30_30	-	150	11.0%	63.4%
T_I_t30_30	0.1	25	0.0%	20.0%
T_I_t30_30	10	25	0.2%	27.5%
T_I_t30_30	100	25	2.1%	40.3%
T_I_t30_30	500	50	5.8%	51.3%
T_I_t30_30	1500	75	8.6%	57.9%
T_I_t30_30	5000	150	9.7%	62.2%
T_I_t30_30	10000	150	10.6%	64.0%
T_I_t30_30	50000	150	11.1%	65.6%
T_I_t30_30	100	25	1.1%	36.7%
(con factor de aprendizaje y	500	25	1.7%	39.7%
momentum iguales a 0.5)	10000	150	3.8%	47.7%
T_IV_pmv4_r2_pS_t49_46_bp3000	-	150	27.7%	79.7%
T_IV_pmv4_r2_pS_t49_46_bp3000	1500	75	19.5%	76.4%
T_IV_pmv4_r2_pS_t49_46_bp3000	10000	150	24.7%	78.9%

Tabla A.32. Resultados de traducción con diversas codificaciones y *reducción de los parámetros de entrenamiento mediante la fórmula (6.17) y distintos valores para el parámetro de dicha fórmula.*

Se realizaron más experimentos, esta vez con la fórmula (6.18) y, como se puede ver en los resultados presentados en la Tabla A.33, se produjeron unos resultados ligeramente mejores que únicamente con Retropropagación, y con mayor independencia del parámetro λ que con la fórmula (6.17).

Nombre	Λ	FBT	PBT
T_I_t30_30	-	11.0%	63.4%
T_I_t30_30	10	5.4%	53.3%
T_I_t30_30	100	9.3%	65.5%
T_I_t30_30	500	10.6%	65.5%
T_I_t30_30	1500	10.3%	65.0%
T_I_t30_30	10000	11.1%	63.0%
T_I_t30_30	50000	10.4%	62.3%
T_VIII_t30_30	-	12.7%	64.1%
T_VIII_t30_30	100	15.1%	65.6%
T_II_pmv4_r2_t30_30_bp3000	-	23.2%	75.3%
T_II_pmv4_r2_t30_30_bp3000	100	26.1%	77.4%
T_III_pmv4_r2_t30_30_bp3000	-	24.6%	76.3%
T_III_pmv4_r2_t30_30_bp3000	100	24.6%	77.3%

Tabla A.33. Resultados de traducción con diversas codificaciones y *reducción de los parámetros de entrenamiento mediante la fórmula (6.18)* para modelos RECONTRA con 450 unidades en la capa oculta y 9 palabras en la entrada.

Cuando estas fórmulas se aplican a modelos RECONTRA extendidos, los resultados son similares, como se puede comprobar en los resultados presentados en la Tabla A.34 para modelos con dos capas ocultas de 270 y 450 unidades.

Método	Λ	t final	FBT	PBT
-	-	150	18.1%	71.3%
(6.17)	1000	75	7.5%	60.6%
(6.17)	10000	150	13.9%	71.1%
(6.18)	100	150	10.7%	68.8%
(6.18)	10000	150	19.6%	74.0%

Tabla A.34. Resultados de traducción con codificaciones I y *reducción de los parámetros de entrenamiento mediante las fórmulas (6.17) y (6.18) y distintos valores del parámetro* λ para modelos RECONTRA con dos capas ocultas de 270 y 450 unidades y 9 palabras en la entrada.

Se seleccionaron codificaciones para la tarea Hansards y se aplicaron las formulas (6.17) y (6.18) con distintos valores del parámetro λ. Los resultados originales y los obtenidos con estos dos métodos pueden observarse en la Tabla A.35 y muestran que no se han producido diferencias importantes en los resultados. En unos pocos casos las redes no han mostrado inestabilidad, pero en esos casos tampoco se han obtenidos resultados mejores que los mejores resultados con las redes inestables.

Nombre	Método	Λ	FBT	PBT
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	-	-	33.83%	32.90%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	(6.17)	0.1	30.53%	29.52%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	(6.17)	10	33.13%	31.95%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	(6.17)	79616	34.13%	32.41%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	(6.18)	10	33.63%	32.03%
H_III_pmv4_r2_t60_60_bp100_scg2100_1400_contexto1	(6.18)	79616	24.43%	33.17%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.17)	-	3.00%	45.0%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.17)	1	5.10%	43.31%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.17)	10	7.51%	41.45%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.17)	100	0.30%	41.63%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.17)	1000	5.10%	45.40%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.17)	79616	2.90%	45.34%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.18)	10	3.10%	45.13%
H_IV_pmv4_r2_t240_240_bp3000f3_r_bp3000f3_contexto1	(6.18)	79616	0.00%	45.34%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_r_bp100_scg3000_3000_contexto1	-	-	34.54%	42.70%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_r_bp100_scg3000_3000_contexto1	(6.17)	10	33.93%	40.70%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	-	-	35.14%	43.80%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	(6.17)	10	35.74%	44.19%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_r_bp3000_contexto1	-	-	35.34%	40.69%
H_IV_pmv4_r2_pS_t192_213_bp3000f3_r_bp3000_contexto1	(6.17)	10	33.03%	37.33%

Tabla A.35. Resultados de traducción con diversas codificaciones extraídas de MLP y *reducción de los parámetros de entrenamiento mediante las fórmulas (6.17) y (6.18) y distintos valores del parámetro λ.* Además se muestran los resultados originales con Retropropagación y sin reducción.

En las tablas A.36 y A.37 pueden verse los resultados para diversas combinaciones de parámetros (de las fórmulas (A.1) y (A.2)). Comparando con los resultados con BEP básico (también mostrados en la tabla), podemos comprobar como, en el mejor de los casos, sólo se producen ligeras mejoras, y, dependiendo de dichos parámetros, se obtienen peores resultados de traducción.

C1	C2	FBT	PBT
-	-	11.0%	63.4%
5	0.65	11.2%	64.2%
50	0.35		
50	0.65	12.2%	65.3%
500	0.65	12.5%	64.3%

Tabla A.36. Resultados de traducción con codificaciones *T_I_t30_30* y *reducción de los parámetros de entrenamiento mediante la fórmula (A.1) y distintos valores para sus parámetros.*

C	T	FBT	PBT
-	-	11.0%	63.4%
5	0.05	0.0%	22.9%
5	0.35	1.1%	39.6%
5	0.65	3.0%	48.1%
5	0.95	4.9%	52.8%
50	0.65	3.1%	48.3%
500	0.65	1.1%	39.7%

Tabla A.37. Resultados de traducción con codificaciones *T_I_t30_30* y *reducción de los parámetros de entrenamiento mediante la fórmula (A.2) y distintos valores para sus parámetros.*

La reducción de parámetros en función del número de iteraciones se basa (fundamentalmente) en la suposición de que cuanto más se entrenan las redes más cerca se está de un mínimo local adecuado, y de que las redes pueden oscilar en torno a él.

Sin embargo, dependiendo de la forma concreta de la función del error, reducir la capacidad de desplazamiento de las redes puede provocar que se atasquen en un mal mínimo local. Al parecer esto es lo que está sucediendo con los distintos métodos de reducción de parámetros: o se reducen mucho y se quedan en un mínimo local malo, con lo que los resultados son peores o se reducen muy poco y los resultados son similares a no utilizarlos.

Reducción de parámetros en función del ECM

En la siguiente serie de experimentos, tras la estimación de parámetros, estos sólo se modificaron (reduciéndose) cuando el ECM residual tras una iteración era superior al de la iteración precedente. Así, se toma la red de la iteración precedente, se reducen los parámetros y se continúa el entrenamiento.

Los resultados, para modelos RECONTRA con 9 palabras de entrada y 450 unidades en la capa oculta y dos codificaciones distintas, son presentados en la Tabla A.38 muestran que en la mayoría de los casos se produce una ligera mejora en los resultados de traducción.

Nombre	Reducción	FBT	PBT
T_L_pmv8_r4_pS_t44_43_bp_R2_bp3000	No	29.7%	79.4%
T_L_pmv8_r4_pS_t44_43_bp_R2_bp3000	1/2	32.0%	81.3%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_bp3000	No	38.6%	83.7%
T_V_pmv6_r4_pS_t121_94_bp_scg_R_bp3000	1/2	39.2%	84.4%

Tabla A.38. Resultados de traducción con diversas codificaciones y *reducción de los parámetros de entrenamiento a un 50% tras una subida del ECM residual.*

Ya se había comentado la inestabilidad de ciertos experimentos con la tarea Hansards y los malos resultados que producían. Como una forma sencilla de solucionar el problema (o por lo menos de paliarlo), se redujo hasta un 10% los parámetros cuando el ECM crecía. Es decir, si el ECM en la iteración *t* era mayor que en la iteración *t-1*, entonces se continuaba desde *t-1* con valores *parámetros/10*. Los resultados son mejores, como se puede ver en la tabla A.39 para modelos RECONTRA básico y en la tabla A.41 para modelos RECONTRA con dos capas. En las tablas también se han colocado los resultados originales para facilitar la comparación.

Sin embargo la fuente de esta mejora parece deberse más a la reducción de la inestabilidad de las redes que a otros posibles factores.

Nombre	| CO |	Reducción	FBT	PBT
H_I_t30_30	900	No	0.00%	0.03%
H_I_t30_30	900	Sí	20.23%	25.74%
H_IV_t240_240	600	No	2.4%	24.33%
H_IV_t240_240	600	Sí	33.96%	38.98%
H_IV_t240_240	900	No	0.0%	11.98%
H_IV_t240_240	900	Sí	34.63%	39.79%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	900	No	35.34%	40.23%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	900	Sí	37.44%	44.36%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1	2160	No	35.84%	40.71%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1	2160	Sí	36.94%	48.32%
H_IV_pmv4_r2_t240_240_bp1000	900	No	0.00%	6.39%
H_IV_pmv4_r2_t240_240_bp1000	900	Sí	32.13%	29.19%
H_IV_pmv4_r2_pS_t192_213_bp3000f3	1800	No	0.00%	6.63%
H_IV_pmv4_r2_pS_t192_213_bp3000f3	1800	Sí	32.30%	30.13%

Tabla A.39. Resultados de traducción con diversas codificaciones y modelos RECONTRA básicos y *reducción de los parámetros de entrenamiento a un 10% tras una subida del ECM residual.*

Nombre	Reducción	FBT	PBT
H_IV_t240_240	No	30.63%	34.90%
H_IV_t240_240	Sí	34.64%	38.56%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1	No	0.00%	16.41%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_R_bp100_scg3000_3000_contexto1	Sí	0.00%	40.79%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	No	33.83%	38.59%
H_IV_pmv4_r2_pS_t192_213_bp100_scg2300_1300_contexto1	Sí	35.84%	43.54%

Tabla A.40. Resultados de traducción con diversas codificaciones y modelos RECONTRA con dos capas ocultas (de 900 unidades cada una) y *reducción de los parámetros de entrenamiento a un 10% tras una subida del ECM residual.*

Modificación de parámetros en función del ECM

En los experimentos del apartado anterior se intentó reducir los valores de los parámetros de entrenamiento con la esperanza de mejorar los resultados de traducción y tal vez de reducir el tiempo de entrenamiento. En la literatura, sin embargo, no se suele limitar la evolución de los parámetros de entrenamiento a la reducción, sino que, dependiendo del ECM residual se puede también producir su aumento.

La idea fundamental es que al aumentar los parámetros aceleramos el aprendizaje de la red cuando se encuentra en una zona de descenso del ECM, cuando se ha superado un mínimo local y ha aumentado el ECM se decrementan los parámetros para acercarse a este mínimo.

La fórmula empleada para modificar los parámetros es la que ya hemos empleado con MLP, (6.19).

Para la tarea del Turista se realizaron varios experimentos con distintas codificaciones y redes RECONTRA básicas y extendidas, con dos capas ocultas (la segunda siempre con 450 unidades ocultas). Los resultados experimentales, en la Tabla A.41, que se muestran junto a los originales, demostraron que utilizar este método no suponía diferencias significativas con no utilizarlo, mejorando o empeorando ligeramente los resultados.

Nombre	| 1ª CO|	Modificación	FBT	PBT
T_I_pmv4_r2_t30_30_bp3000	-	No	20.7%	76.5%
T_I_pmv4_r2_t30_30_bp3000	-	Sí	20.0%	75.4%
T_III_pmv4_r2_t30_30_bp3000	-	No	24.6%	76.3%
T_III_pmv4_r2_t30_30_bp3000	-	Sí	24.2%	76.7%
T_II_pmv4_r2_t30_30_bp3000	-	No	23.2%	75.3%
T_II_pmv4_r2_t30_30_bp3000	-	Sí	23.7%	75.8%
T_II_pmv4_r2_t30_30_bp3000	270	No	31.4%	79.4%
T_II_pmv4_r2_t30_30_bp3000	270	Sí	26.4%	78.3%
T_II_pmv4_r2_t30_30_bp3000	450	No	32.4%	80.5%
T_II_pmv4_r2_t30_30_bp3000	450	Sí	30.5%	80.7%

Tabla A.41. Resultados de traducción aplicando o no modificación de los parámetros en *función del ECM residual* con la formula (6.19) y diversas codificaciones y modelos RECONTRA básicos con 450 unidades ocultas y extendidos, con dos capas ocultas (450 unidades en la segunda capa oculta), y siempre con 9 palabras en la ventana de entrada.

Los resultados anteriores parecen indicar que la modificación de parámetros en función del ECM no aporta mejores resultados de traducción y puede incluso proporcionar peores. Se debería realizar un estudio más profundo para determinar que factores pueden determinan estos resultados, pero esta fuera del alcance de esta tesis.

Se realizaron varios experimentos con distintas codificaciones y redes RECONTRA básicas y extendidas, con dos capas ocultas. Los resultados experimentales, en la Tabla A.42 para modelos RECONTRA básicos, y en la Tabla A.43 para RECONTRA con dos capas ocultas, mostraron varios comportamientos diferenciados al utilizar este método.

En la mayoría de los casos sólo se produjeron pequeñas diferencias en los resultados, mejorando o empeorando ligeramente.

En muchos casos el método de modificación de los parámetros reduce la inestabilidad, pero no en todos ellos. En casos concretos este método no reduce la inestabilidad, con lo que tras una mejora de los resultados estos empeoran drásticamente. Un ejemplo serían los dos experimentos con codificaciones *H_IV_pmv4_r2_t240_240_bp1000*, que caen tras oscilar en torno al 29% de PBT (mostrados en la Tabla A.43).

Nombre	|CO|	|Ventana|	Modificación	FBT	PBT
H_III_pmv4_r2_t60_60_bp100_ scg2100_1400_contexto1	900	5	No	33.83%	32.90%
H_III_pmv4_r2_t60_60_bp100_ scg2100_1400_contexto1	900	5	Sí	33.93%	32.64%
H_III_pmv4_r2_t60_60_bp3000f3_con texto1	900	5	No	33.43%	33.23%
H_III_pmv4_r2_t60_60_bp3000f3_con texto1	900	5	Sí	35.53%	33.66%
H_IV_pmv4_r2_pS_t192_213_bp1000	900	1	No	14.82%	26.68%
H_IV_pmv4_r2_pS_t192_213_bp1000	900	1	Sí	14.82%	27.39%
H_IV_pmv4_r2_t240_240_bp1000	900	9	No	0.00%	6.39%
H_IV_pmv4_r2_t240_240_bp1000	900	9	Sí	1.26%	10.54%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_contexto1	900	1	No	14.62%	31.62%
H_IV_pmv4_r2_t240_240_bp100_ scg1600_900_contexto1	900	1	Sí	15.82%	31.37%
H_IV_pmv4_r2_t240_240_bp3000 f3_r_bp3000f3_contexto1	900	5	No	3.90%	45.00%
H_IV_pmv4_r2_t240_240_bp3000 f3_r_bp3000f3_contexto1	900	5	Sí	0.70%	44.02%
H_IV_pmv4_r2_t240_240_bp100_scg 1600_900_r_bp100_scg3000_3000_co ntexto1	1800	9	No	36.64%	45.24%
H_IV_pmv4_r2_t240_240_bp100_scg 1600_900_r_bp100_scg3000_3000_co ntexto1	1800	9	Sí	36.44%	46.22%
H_IV_pmv4_r2_t240_240_bp100_scg 1600_900_r_bp100_scg3000_3000_co ntexto1	2160	9	No	35.84%	40.71%
H_IV_pmv4_r2_t240_240_bp100_scg 1600_900_r_bp100_scg3000_3000_co ntexto1	2160	9	Sí	36.44%	46.49%
H_IV_pmv4_r2_t240_240_bp100_scg 1600_900_r_bp100_scg3000_3000_co ntexto1	900	5	No	0.00%	40.91%
H_IV_pmv4_r2_t240_240_bp100_scg 1600_900_r_bp100_scg3000_3000_co ntexto1	900	5	Sí	0.00%	44.11%

Tabla A.42. Resultados de traducción aplicando o no modificación de los parámetros en *función del ECM residual con formula (6.19)* con diversas codificaciones y modelos RECONTRA básicos con distintos tamaños en la capa oculta y con 9, 5 o 1 palabras en la ventana de entrada.

Nombre	Ventana	FBT	PBT
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	1	17.22%	34.73%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_contexto1	1	17.42%	34.84%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_r_bp100_scg3000_3000_contexto1	5	0.00%	16.41%
H_IV_pmv4_r2_t240_240_bp100_scg1600_900_r_bp100_scg3000_3000_contexto1	5	0.00%	8.35%

Tabla A.43. Resultados de traducción aplicando o no modificación de los parámetros en *función del ECM residual* con diversas codificaciones y modelos RECONTRA con dos capas ocultas (900 unidades en la ambas capas ocultas), y con 9, 5 o 1 palabras en la ventana de entrada.

Los distintos métodos que se han probado para modificar (aunque hablemos sólo de reducción) los parámetros de entrenamiento para modelos RECONTRA no han supuesto en general una mejora en los resultados respecto a su no utilización.

Con Hansards, sin embargo, la situación es más complicada. Estos métodos logran en muchas ocasiones disminuir la inestabilidad de las redes, lo cual proporciona una mejora considerable de los resultados.

Por otra parte, todos estos métodos suponen la inclusión, de una forma o de otra de nuevos parámetros, con el problema de su consiguiente selección.

Los distintos métodos de entrenamiento de RECONTRA que se han explorado en este capítulo han generado resultados muy diversos, sin embargo se pueden establecer una serie de conclusiones generales para cada grupo de métodos.

Reducir los parámetros de entrenamiento del algoritmo de Retropropagación en función del número de iteraciones:

1. Estos métodos suponen añadir nuevos parámetros al método de entrenamiento, lo cual no es una ventaja.
2. En general todos los métodos tienden a proporcionar resultados similares a los obtenidos al no utilizarlos (un poco peores, un poco mejores), excepto en casos concretos en los que por influencia de parámetros poco adecuados, los resultados son sensiblemente peores.
3. Por último señalar que los métodos empleados son sencillos, con lo que no aumenta apreciablemente el tiempo de entrenamiento de las redes.

Modificar los parámetros de entrenamiento del método de Retropropagación en función del ECM residual es teóricamente mejor que únicamente en función del número de iteraciones y los resultados en general lo demuestran.

1. También necesitan nuevos parámetros y son sensibles a los seleccionados, como los métodos que sólo tenían en cuenta el número de iteraciones.
2. Los resultados son similares (un poco mejores, un poco peores), excepto con la tarea Hansards, donde en muchos casos la reducción en la inestabilidad supone una mejora considerable en los resultados finales. En un caso concreto proporciona los mejores resultados para la tarea.

3. Cuando los parámetros pueden tanto disminuir como aumentar, aunque el comportamiento es similar a la simple reducción es de señalar que no se reduce la inestabilidad de las redes tanto como cuando sólo se reducen. Es posible que en parte esto se deba a que en la reducción se vuelve a la última red antes de la subida del ECM residual, “deshaciendo el error”.
4. Tampoco aumenta perceptiblemente el tiempo de entrenamiento, excepto en el método de reducción, en el cual se repite una iteración de la experimentación cada vez que se tiene que reducir los parámetros.

Anexo B. Frases mal traducidas

En este anexo se muestran ejemplos de las frases mal traducidas para las tareas.

B.1. Turista

I have a reservation for a quiet , single room with a good view , a bath and a telephone for C'esar Borillo . FinFrase# I have a reservation for a quiet , single room with a good view and a and for a shower for V'azquez

I have booked a room for twenty days . FinFrase# I have booked a room for twenty-first days . FinFrase

could you give me the keys to our room , please ? FinFrase# could you wake me we keys to our room , please ? FinFrase

would you mind sending down my suitcase to room number four five four , please ? FinFrase# would you mind sending down my suitcase to room number four four four , please ? FinFrase

I made a reservation for Asunci'on Rivera . FinFrase# I made a reservation for Rosario Mira . FinFrase

I leave on June the fourteenth at a quarter past one in the afternoon . FinFrase# I leave on June the station at a quarter past one in the afternoon . FinFrase

a mistake has been made in our bill for room number four one eight . FinFrase# a mistake has been made in our bill for room number one one . FinFrase

could you give me the keys to room number seven one two ? FinFrase# could you give me the keys to room number five one five ? FinFrase

what is the price for a single room including breakfast per week ? FinFrase# what is the price for a single room , please ? FinFrase

do they have air conditioning , a tv and hot water ? FinFrase# do the rooms have air conditioning , a tv and day eighteen

can you prepare the bill for room number two five four for me , please ? FinFrase# can you prepare the bill for room number two five four , please ? FinFrase

could you send up our suitcase to room number eight five four ? FinFrase# could you send up our suitcase to room number eight one four eighteen ? V'azquez

would you mind waking me up at three , please ? FinFrase# would you mind waking me up at one , please ? FinFrase

can you make out the bill for us , please ? FinFrase# can you make out the bill for me , please ? FinFrase

I made a reservation for Mrs Belenguer . FinFrase# I made a reservation for Mrs Alted . FinFrase

B.2. Hansards

is it your pleasure <Com> honourable senators <Com> to adopt the motion <CIn> FinFrase # is it your pleasure <Com> honourable senators <Com> to adopt the motion FinFrase

motion agreed to <Com> and bill read second time <Pun> FinFrase # motion agreed to and bill read second time <Pun> FinFrase

on motion of senator kenny <Com> bill referred to the standing senate committee on energy <Com> the environment and natural resources <Pun> FinFrase # on motion of senator senator <Com> bill referred to the standing senate committee on committee on on <Com> the causes and

sir john a <Pun> macdonald day bill FinFrase # bill bill bill bill defending one comment

honourable senator grafstein <CPa> <Pun> FinFrase # honourable senator <CPa> <Pun> FinFrase

on motion of senator hays <Com> for senator grafstein <Com> debate adjourned <Pun> FinFrase # on motion of senator hays <Com> for senator senator <Com> debate adjourned <Pun> FinFrase

some people have some concerns about this proposed legislation <Pun> FinFrase # <NUM> have causes causes causes

however <Com> those concerns are best addressed in the committee rather than on the floor of the senate chamber <Pun> FinFrase # however <Com> the committee in committee the the lynch observation ontario <Pun> criteria <Pun>

honourable senators <Com> it is true that this has already been debated and objections raised <Pun> FinFrase # honourable senators <Com> it is is that that lynch comment lynch lynch lynch and lynch lynch comment comment comment <Pun>

this is something we should at least point out <Com> even if i did not always agree with him <Pun> FinFrase # it is a a that we back comment lynch lynch comment <Com> lynch i mine condemned not lynch lynch comment comment one lynch comment

i have nothing against that <Com> but no one ever told us how much each of these changes costs <Pun> FinFrase # i have have <Com> every <Com> to <Com> <Com> is every has to every every to <Com> <Com> of

however <Com> that is another issue <Pun> FinFrase # deployment <Com> is a <Pun> <Pun> <Pun> <Pun>

this means that when the chief electoral officer receives parliament <SCo> s decision <Com> it will be all over <Pun> FinFrase # that that that to when the causes the comment the the causes <Com> <Com> will

i always raise the same objections <Pun> FinFrase # i <Com> i <Com> i the

at some point <Com> i will not only have objections <Com> i will also try to convince senators that i am right <Com> that no changes should be made to the names of ridings between elections <Pun> FinFrase # at a this <Com> <Com> not not not not

thought that that i causes i causes like causes the causes <Gui> <Gui> i that <Com> that that <Pun> that causes the causes tuesday mine comment comment lynch comment <Pun> comment one

honourable senators <Com> i should like to make a few comments with respect to the fourth report of the standing committee on privileges <Com> standing rules and orders <Pun> FinFrase # honourable senators <Com> i i should like to to to to every report the committee standing senate committee on thE1rE2se <Com> standing and and

the senate found a prima facie case of breach of privilege in each of those questions raised in the senate and referred those questions to the committee <Pun> FinFrase # the senate senate that that that every every canada every to every every that <Com> every every every every to <Pun>

the committee reviewed both questions of privilege in its proceedings <Pun> FinFrase # the committee to <Pun> to every every every <Pun> comment

the senate has taken the question of a breach of privilege very seriously indeed <Pun> FinFrase # the senate senate has the to the of the of

hopefully <Com> there will be very few matters in the future <Pun> FinFrase # the honourable the be to be to to these <Pun>

i want to be clear <Pun> FinFrase # i i will <Pun> FinFrase