



Universitat Autònoma de Barcelona

ADVERTIMENT. L'accés als continguts d'aquesta tesi queda condicionat a l'acceptació de les condicions d'ús establertes per la següent llicència Creative Commons:  http://cat.creativecommons.org/?page_id=184

ADVERTENCIA. El acceso a los contenidos de esta tesis queda condicionado a la aceptación de las condiciones de uso establecidas por la siguiente licencia Creative Commons:  <http://es.creativecommons.org/blog/licencias/>

WARNING. The access to the contents of this doctoral thesis it is limited to the acceptance of the use conditions set by the following Creative Commons license:  <https://creativecommons.org/licenses/?lang=en>



**Universitat Autònoma
de Barcelona**

**Exploiting the Interplay between Visual and Textual Data
for Scene Interpretation**

A dissertation submitted by **Raúl Álamo Gómez
Bruballa** at Universitat Autònoma de Barcelona to
fulfil the degree of **Doctor of Philosophy**.

Bellaterra, July 9, 2020

Director	Dr. Dimosthenis Karatzas Universitat Autònoma de Barcelona Dept. Ciències de la Computació Centre de Visió per Computador
Co-Director	Dr. Lluís Gómez Universitat Autònoma de Barcelona Dept. Ciències de la Computació Centre de Visió per Computador
Co-Director	Dr. Jaume Gibert Eurecat: Centre Tecnològic de Catalunya Unitat de Tecnologies Multimèdia
Thesis committee	Dr. Josep Lladós Centre de Visió per Computador Dr. Diane Larlus Naver Labs Europe Dr. Francesc Moreno Universitat Politècnica de Catalunya Institut de Robòtica i Informàtica Industrial
International evaluators	Dr. Nicu Sebe University of Trento Dr. Andrew D. Bagdanov University of Florence



This document was typeset by the author using $\text{\LaTeX}2_{\epsilon}$.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona and Eurecat: Centre Tecnològic de Catalunya.

Copyright © MMXIX by **Raúl Álamo Gómez Bruballa**. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the author.

ISBN

Printed by Ediciones Gráficas Rey, S.L.

Agradecimientos

Para comenzar, doy las gracias a mi familia por apoyarme totalmente en el camino que escogí, instigarme a recorrerlo con esfuerzo, y enseñarme a hacerlo con disfrute. También a su mecenazgo durante mis años de carrera y máster sin el cual, desafortunadamente, nunca hubiera conseguido el contrato de doctorando.

Agradezco también a mis compañeros de carrera en Telecos UPC, de residencia en el Penyafort, y de piso en Sajierro, junto a los cuales me zambullí en el apasionante mundo de la ciencia y la tecnología hablando sobre Fourier entre cacahuets salados, cerveza barata, y muchas otras distracciones maravillosas. Esos primeros años en Barcelona fueron frenéticos, un cúmulo de nuevas experiencias en el cual los sacrificios hechos para sacarse la carrera se diluían y volatilizaban. Gracias a todo el entorno que lo hizo posible, y enhorabuena a todos los que lo disfrutaron conmigo.

Thanks also to all the PhD students in CVC I have worked with, discussed about research or about things that should be improved in our community. One of the hardest things of pursuing a PhD is that you spend a lot of time working on stuff you cannot explain to anyone except to a few colleagues from your research community. And I think doing so is very important, both for research and for our health. Actually, and speaking about things that should be improved, my opinion is that we should move from the individualistic research model we are in to a team work model.

Thanks also to my PhD supervisors in CVC who have guided my training as a researcher. They have provided me good guidelines while giving me enough freedom to follow my own research ideas, which has been crucial to maintain my motivation.

Gràcies també al meu supervisor i als companys d'Eurecat, que m'han permès centrar-me en la investigació i han col·laborat en aquesta, a la vegada integrant-me en l'equip per col·laborar en projectes relacionats. Sé que, tot i que els doctorands industrials haurien de dedicar el seu temps a investigació, encara que aplicada, això no és sempre així.

Por último, agradecer su esfuerzo a todas aquellas personas que, desde dentro o desde fuera, trabajan para que tengamos un sistema educativo público de calidad que garantice la igualdad de oportunidades de la cual, desafortunadamente, tan lejos estamos. En mi caso, sin el apoyo económico de mi familia y aún disfrutando de becas, no hubiera podido vivir en Barcelona y aún menos en una residencia estudiantil. Os aseguro que sin esa posibilidad no estaría escribiendo esta tesis. Tampoco hubiera podido permitirme dejar el trabajo para estudiar el máster que luego me dió acceso al contrato de doctorando. No es una cuestión de esfuerzo, es una cuestión de oportunidades.

Abstract

Machine learning experimentation under controlled scenarios and standard datasets is necessary to compare algorithms performance by evaluating all of them in the same setup. However, experimentation on how those algorithms perform on unconstrained data and applied tasks to solve real world problems is also a must to ascertain how that research can contribute to our society.

In this dissertation we experiment with the latest computer vision and natural language processing algorithms applying them to multimodal scene interpretation. Particularly, we research on how image and text understanding can be jointly exploited to address real world problems, focusing on learning from Social Media data.

We address several tasks that involve image and textual information, discuss their characteristics and offer our experimentation conclusions. First, we work on detection of scene text in images. Then, we work with Social Media posts, exploiting the captions associated to images as supervision to learn visual features, which we apply to multimodal semantic image retrieval. Subsequently, we work with geolocated Social Media images with associated tags, experimenting on how to use the tags as supervision, on location sensitive image retrieval and on exploiting location information for image tagging. Finally, we work on a specific classification problem of Social Media publications consisting on an image and a text: Multimodal hate speech classification.

Resum

L'experimentació en aprenentatge automàtic en escenaris controlats i amb bases de dades estàndards és necessària per a comparar el rendiment entre algoritmes avaluant-los sota les mateixes condicions. Però també és necessària l'experimentació en com es comporten aquests algoritmes quan són entrenats amb dades menys controlades i aplicats a problemes reals per indagar en com els avanços en recerca poden contribuir a la nostra societat.

En aquesta tesi, experimentem amb els algoritmes més recents de visió per ordinador i processament del llenguatge natural aplicant-los a la interpretació d'escenes multimodals. En particular, investiguem en com la interpretació automàtica d'imatges i text es pot explotar conjuntament per resoldre problemes reals, enfocant-nos en aprendre de dades de xarxes socials.

Encarem diverses tasques que impliquen informació visual i textual, discutim les seves particularitats i reptes i exposem les nostres conclusions experimentals. Primer treballem en la detecció de text en imatges. A continuació, treballem amb publicacions de xarxes socials, fent servir els subtítols textuais associats a imatges com a supervisió per aprendre característiques visuals, que apliquem a la cerca d'imatges semàntica amb consultes multimodals. Després, treballem amb imatges de xarxes socials geolocalitzades amb etiquetes textuais associades, experimentant en com fer servir les etiquetes com a supervisió, en cerca d'imatges sensible a la localització, i en explotar la localització per l'etiquetatge d'imatges. Finalment, encarem un problema de classificació específic de publicacions de xarxes socials formades per una imatge i un text: Classificació de discurs de l'odi multimodal.

Resumen

La experimentación en aprendizaje automático en escenarios controlados y con bases de datos estándares es necesaria para comparar el desempeño entre algoritmos evaluándolos en las mismas condiciones. Sin embargo, también es necesaria experimentación en cómo se comportan estos algoritmos cuando son entrenados con datos menos controlados y aplicados a problemas reales para indagar en cómo los avances en investigación pueden contribuir a nuestra sociedad.

En esta tesis experimentamos con los algoritmos más recientes de visión por ordenador y procesamiento del lenguaje natural aplicándolos a la interpretación de escenas multimodales. En particular, investigamos en cómo la interpretación automática de imagen y texto se puede explotar conjuntamente para resolver problemas reales, enfocándonos en aprender de datos de redes sociales.

Encaramos diversas tareas que implican información visual y textual, discutimos sus características y retos y exponemos nuestras conclusiones experimentales. Primeramente trabajamos en la detección de texto en imágenes. A continuación, trabajamos con publicaciones de redes sociales, usando las leyendas textuales de imágenes como supervisión para aprender características visuales, que aplicamos a la búsqueda de imágenes semántica con consultas multimodales. Después, trabajamos con imágenes de redes sociales geolocalizadas con etiquetas textuales asociadas, experimentando en cómo usar las etiquetas como supervisión, en búsqueda de imágenes sensible a localización, y en explotar la localización para el etiquetado de imágenes. Finalmente, encaramos un problema de clasificación específico de publicaciones de redes sociales formadas por una imagen y un texto: Clasificación de discurso del odio multimodal.

Contents

1	Introduction	1
1.1	Visual and Textual Data	1
1.1.1	Images	1
1.1.2	Language: Speech and Text	2
1.1.3	Image and Text Compositions	3
1.1.4	Image and Text Semantic Relations	4
1.2	Images Associated Text as Supervision	5
1.3	Outline, Research Questions and Contributions	6
2	Background	11
2.1	Cross-Entropy Loss	12
2.1.1	Classification Tasks	12
	Multi-Class Classification	12
	Multi-Label Classification	12
2.1.2	Activation Functions	13
	Sigmoid	13
	Softmax	13
2.1.3	Formulation	13
	Categorical Cross-Entropy Loss	14
	Binary Cross-Entropy Loss	14
2.2	Ranking Loss	15
2.2.1	Metric Learning	15
2.2.2	Pairwise Ranking Loss	16
2.2.3	Triplet Ranking Loss	17
2.2.4	Siamese and Triplet Nets	19
	Siamese Nets	19
	Triplet Nets	19
2.3	Text Representations	20
3	Scene Text Segmentation and Style Transfer	23
3.1	Pixel Level Segmentation of Text in Images	23
3.1.1	Background on Fully Convolutional Networks	23
3.1.2	TextFCN: Learning Text Visual Features	24
3.1.3	Methodology	24

3.1.4	Results	24
3.1.5	TextFCN to Improve Scene Text Detection	25
	Background on Scene Text Detection Pipelines	25
	Methodology	26
	Results	27
	Conclusions	27
3.2	Selective Text Style Transfer For Text	28
3.2.1	Introduction	28
3.2.2	Background on Text Style Transfer	29
3.2.3	Style Transfer	30
3.2.4	Selective Style Transfer for Text	30
	Two-Stage Architecture	31
	End-to-end Architecture	31
3.2.5	Experiments	33
	Scene Text	33
	Machine Printed Text	34
	Handwritten Text	34
	Cross Domain	34
3.2.6	Data Augmentation	38
3.2.7	Conclusions	39
4	Exploiting Images Associated Text as Supervision	41
4.1	Introduction	41
4.1.1	Self-Supervised Learning	41
4.1.2	Exploiting Multimodal Web Data	42
4.2	Background on Multimodal Image and Text Embeddings	43
4.3	Semantic Multimodal Image Retrieval	43
4.3.1	Multimodal Text-Image Embedding	44
4.3.2	Datasets	45
	InstaCitiesIM	45
	WebVision	46
4.3.3	Results	46
	Error Analysis	48
	CNN Activation Maps Visualization	49
	Visualizing the Learned Semantic Space with t-SNE	49
4.3.4	Comparing the Image and Text Embeddings	56
	Experiment Setup	56
	Results and Conclusions	57
4.4	Learning from Barcelona Instagram Data	58
4.4.1	Text Supervision in Social Media for Tourism Analysis	58
4.4.2	Dataset	58
4.4.3	Textual Analysis	59
4.4.4	Visual Analysis	60
	Img2NeighCtx	61
	Images Associated to Districts	61

Beyond Districts: Img2Word2Vec	63
4.4.5 Applications to Tourism Industry	63
4.4.6 #Barcelona Deep Dream	64
4.5 Conclusions	66
5 Multimodal Learning with Images, Text and Geolocations	67
5.1 Introduction	67
5.2 Background on Location Sensitive Image Retrieval and Tagging	68
5.3 Methodology	69
5.3.1 Learning with Tags Supervision	69
Multi-Label Classification	69
Multi-Class Classification	70
Hashtag Embedding Regression	70
5.3.2 Location Sensitive Model (<i>LocSens</i>)	71
Balancing Location Influence on Ranking.	72
5.4 Experiments	73
5.4.1 Dataset	73
5.4.2 Image by Tag Retrieval	74
5.4.3 Location Sensitive Image by Tag Retrieval	74
Challenging queries	77
5.4.4 Image Tagging	78
5.4.5 Conclusions	81
6 Multimodal Hate Speech Detection	83
6.1 Introduction	83
6.2 Background	84
6.2.1 Background on Multimodal Hate Speech Detection	84
6.2.2 Background on Visual and Textual Data Fusion	85
6.3 The MMHS150K Dataset	85
6.4 Methodology	87
6.4.1 Unimodal Treatment	87
Images	87
Tweet Text	87
Image Text	88
6.4.2 Multimodal Architectures	88
Feature Concatenation Model (FCM)	88
Spatial Concatenation Model (SCM)	89
Textual Kernels Model (TKM)	89
Training	89
6.5 Results	90
6.6 Conclusions	92
7 Conclusions	93
Bibliography	101

List of Tables

3.1	Detection rates of TextProposals and TextProposals with TextFCN Suppression considering the top 10, 100, 1000, and all proposals on COCO-Text datasets	28
3.2	Mean execution time per image (seconds) considering all proposals generated by the different proposed strategies (64k for baseline and 14k with the TextFCN suppression). Word recognition is performed with [36], which takes about 0.4 ms per bounding box. Experiments were conducted on a system with a GPU GeForce GTX TITAN and an Intel® Core™2 Quad CPU Q9300@ 2.50GHz processor	28
3.3	Results (F-score) of the EAST text detector with different training data, evaluated on ICDAR 2015 Challenge 4 and ICDAR 2013 Challenge 2. *Result reported on the original EAST github.	40
4.1	Performance on InstaCities1M and WebVision. First column shows the mean P@5 for all the queries, second for the simple queries and third for complex queries.	47
4.2	Performance on transfer learning. First column shows the mean P@5 for all the queries, second for the simple queries and third for complex queries.	47
4.3	Performance on InstaCities1M using GloVe tf-idf introducing noise by changing the indicated % of captions by random captions from the training set.	47
5.1	Location sensitive hashtag based image retrieval: $P@10$. A retrieved image is considered correct if its groundtruth hashtags contain the queried hashtag and the distance between its location and the queried one is smaller than a given threshold (in kilometers in the table)	76
5.2	Image tagging: $A@1$, $A@10$, %pred and %cpred of the frequency baseline, location agnostic prediction and the location sensitive model	79
6.1	Performance of the proposed models, the LSTM and random scores. The Inputs column indicates which inputs are available at training and testing time.	91

List of Figures

1.1	Grayscale image pixel values (0-255). RGB images are encoded similarly but with one matrix per RGB color. Image source: https://openframeworks.cc/	2
1.2	An image with an associated article, caption, and tags.	3
1.3	Examples of handwritten text (left), machine printed text (center), and scene text (right).	4
1.4	Images and associated text with different semantic relations.	5
1.5	Results of pixel level text detection (explained in Chapter 3)	7
1.6	Results of text style transfer on scene text images (explained in Chapter 3)	7
1.7	Image retrieval results for multimodal queries composed by an image and text (explained in Chapter 4)	8
1.8	Image retrieval results for <i>bridge</i> and different query locations (explained in Chapter 5)	8
1.9	Multimodal hate speech detection setup (explained in Chapter 6)	9
2.1	Multi-class and multi-label classification tasks examples in a setup with three classes: Sun, moon and cloud.	12
2.2	Categorical Cross Entropy Loss applied to a Multi-Class Classification problem. Note that each Softmax activation function depends on all the network outputs, and that their outputs sum up to one, and therefore can be interpreted as class probabilities. Only the positive class (cloud in this example) keeps its term in the loss, since the others are multiplied by 0.	15
2.3	Binary Cross Entropy Loss applied to a Multi-Label Classification problem. Note that each class is treated as a binary classification problem dependent on a single CNN output, and that all the classes keep their term in the final loss.	15
2.4	Pairwise ranking loss use case example to train an face image verification Convolutional Neural Network. The CNN is trained to embed images of the same person nearby.	17
2.5	Triplet Ranking Loss uses case example to train an image face verification Convolutional Neural Network.	18
2.6	Representation of three types of negatives for a triplet with anchor "a" and positive "p" samples.	19

2.7	Illustration of the main features and typical use cases of the explained text representation methods.	22
3.1	Architecture of the Fully Convolutional Network used in our pipeline for text prediction.	25
3.2	Input images and output TextFCN heatmaps showing the pixel level text probability.	26
3.3	Top 500 ranked regions by TextProposals (in red) and by TextProposals with the TextFCN suppression strategy (in green).	27
3.4	Results stylizing the whole image with three different text styles and the two-stage architecture.	31
3.5	Two-stage Selective Text Style Transfer pipeline.	32
3.6	End-to-end Selective Text Style Transfer pipeline.	33
3.7	Applying different styles to various scene text images using the two-stage and end-to-end architectures with the cropped word styles.	35
3.8	Extrapolating from “Coco-Cola” to “Rock” style on COCO-Text images.	36
3.9	Results of the machine printed text model. The styles of the images on top are transferred to the images on the left.	36
3.10	Results of the handwritten model. The styles of the images on top are transferred to the images on the left.	36
3.11	Results of the scene text model applied to machine printed text (top) and handwritten text (bottom) images.	37
3.12	Handwritten text model transferring styles (top) to machine text images (left).	38
3.13	Arial text has been stylized with the original image scene text style (left) and manually inserted (right).	38
3.14	Results of transferring machine printed text styles to scene images (top) and handwritten images (bottom).	39
4.1	Pipeline of the visual embedding model training and the image retrieval by text.	44
4.2	Top-ranked results of combined text queries related with haircuts by our semantic image retrieval model. The learnt joint image-text embedding permits to learn a rich semantic manifold even for previously unseen concepts even though they might not be explicitly present in the training set.	48
4.3	First retrieved images for complex queries related with basketball with Word2Vec on InstaCites1M.	49
4.4	First retrieved images for textual queries with Word2Vec on InstaCites1M.	50
4.5	First retrieved images for complex textual queries with Word2Vec on InstaCites1M.	51
4.6	First retrieved images for complex queries (left), city related complex queries (top-right) and non-object queries (bottom-right) with Word2Vec on InstaCites1M.	52

4.7	First retrieved images for multimodal queries (concepts are added or removed to bias the results) with Word2Vec on WebVision.	52
4.8	First retrieved images for multimodal complex queries with Word2Vec on WebVision.	53
4.9	First retrieved images for simple (left and right columns) and complex weighted queries with Word2Vec on InstaCites1M.	53
4.10	First retrieved images for text queries using Word2Vec on WebVision. Concepts are removed to bias the results.	53
4.11	Top-5 activations for five units in pool5 layer of GoogleNet model trained from scratch with InstaCites1M using GloVe tf-idf as self-supervision and their activation maps.	54
4.12	Visualization of the joint embedding with Word2Vec on InstaCites1M dataset.	55
4.13	Text embeddings distance (X) vs the images embedding distance (Y) of different random image pairs for LDA, Word2Vec and GloVe embeddings trained with InstaCites1M. Distances have been normalized between [0,1]. Points are red if the pair does not share any tag, orange if it shares 1, light orange if it shares 2, yellow if it shares 3 and green if it shares more. R^2 is the coefficient of determination of images and texts distances.	56
4.14	Training procedure of Img2NeighCtx. The CNN is trained to maximize the difference of distances between the image and the positive caption and the image and the negative caption in the <i>Neighborhood Space</i> space until a certain margin.	60
4.15	Img2NeighCtx image by neighborhood retrieval results for different neighborhoods in each of the languages.	62
4.16	Img2Word2Vec image by text retrieval results for different queries in each of the languages.	64
4.17	Deep Dream applied to Img2Word2Vec.	65
4.18	Deep Dream applied to another layer of Img2Word2Vec.	65
5.1	The proposed <i>LocSens</i> multimodal scoring model trained by triplet ranking (bars after concatenation indicate fully connected + group normalization + ReLu activation layers). During training, location information is processed and inputted to the model with different strategies.	72
5.2	Top retrieved image by <i>LocSens</i> , our location sensitive model, for the query hashtag “ <i>temple</i> ” at different locations.	75
5.3	Top retrieved image by <i>LocSens</i> , our location sensitive model, for the query hashtag “ <i>bridge</i> ” at different locations.	75
5.4	Left: $P@10$ of the location sampling strategy for different σ and models with zeroed and raw locations. Right: $P@10$ difference respect to $\sigma = 1$	77
5.5	Query hashtags with different locations and top 3 retrieved images.	78
5.6	Query hashtags with different locations where some queries are incompatible because the hashtag refers to a concept which does not occur in the query location.	79

5.7	Images with their locations and groundtruth hashtags and the corresponding top 5 hashtags predicted by the location agnostic MCC model and <i>LocSens</i>	80
5.8	<i>LocSens</i> top predicted hashtags for images with different faked locations. .	80
5.9	<i>LocSens</i> top predicted hashtags for images with different faked locations. .	81
6.1	Percentage of tweets per class in MMHS150K.	86
6.2	Percentage of hate and not hate tweets for top keywords of MMHS150K. .	86
6.3	FCM architecture. Image and text representations are concatenated and processed by a set of fully connected layers.	88
6.4	TKM architecture. Textual kernels are learnt from the text representations, and convolved with the image representation.	90

Chapter 1

Introduction

1.1 Visual and Textual Data

1.1.1 Images

Humans obtain information from the environment through their senses and process it in the brain. Furthermore, we are able to remember that processed information and use it in further reasoning. Particularly, the human visual system allows us to perceive light, which we define as the portion of the electromagnetic spectrum that can be perceived by human eyes. Analyzing the light emitted or reflected by objects in our surroundings we are able to get information about them as their shape or color, which is nothing else than our interpretation of wavelengths and amplitude of the electromagnetic waves reflected by them. This provides us with valuable information to interpret the scene; to interpret our surroundings.

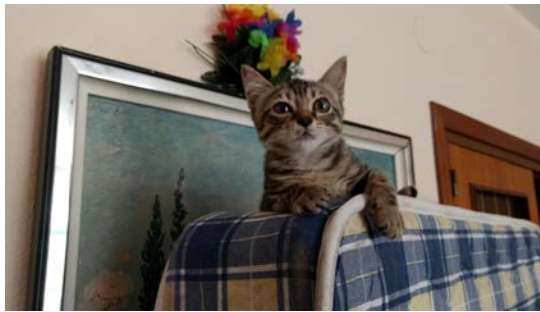
Humans have developed sensors able to measure the characteristics of the electromagnetic waves they receive. Digital cameras use image sensors which produce a certain electrical charge depending on the electromagnetic waves received. Cameras can be designed to be sensible to any wavelength, but the most common camera is sensible to similar wavelengths as our eye. That is because its objective is to capture the same data our eye is sensible to, in order to be able later to display it to be interpreted by a human eye. Light sensors cannot distinguish between the different wavelengths they receive, but our eye does, so to get a color image we need to capture that information. To do that, a camera splits the light in red, green and blue wavelengths and captures each one with different sensors. We call the signal captured by each set of red, green and blue light sensors an image channel, and combining them we recreate the different color sensations humans can perceive. Cameras generate images with a certain resolution that depends on the number of sensors and it is measured in pixels. Each pixel lighting is encoded by three values which measure the red, green and blue

1.1.3 Image and Text Compositions

In this thesis we research on machine learning models that jointly interpret images and text. We define different image and text compositions depending on the type of text and on the association they have, which is an important aspect in this research, as we will see in Chapter 4.

Text Associated to Images

With text associated to images we refer to paired image and textual data. That is a common data type in nowadays Web and Social Media, where webs, blog posts, or Social Media publications tend to combine both data modalities. The text associated to images can be very diverse, but in here we define three categories that group the most common data and are relevant in our further experiments. An example of an image with those three types of associated text is shown in Figure 1.2.



Article	Caption	Tags
Young gray and white cats usually climb to high positions when they get to new places. They do so to get a general view and understanding of the new location. Those cats, usually ...	Our new cat on top of a mattress inspecting his new home.	cat, mattress, welcome, flatmate barcelona

Figure 1.2: An image with an associated article, caption, and tags.

- **Article:** A long text, consisting on several sentences, associated to an image that appears on it. Examples would be a Wikipedia article, a news article or a blog post.
- **Caption:** A sentence attached to an image. It can describe the image content or contain other information. Examples would be Instagram posts or Twitter posts consisting on an image and a short associated text.

- **Tag:** An image can be accompanied (tagged) with tags, also called hashtags. Those are words that the user associates to images and that generally indicate categories, which can refer to the content or be more abstract. Examples are Flickr or Instagram tags, which are used to made images searchable.

Text in Images

Text can also appear in the image content. We define three different categories of text in images, which are common in the computer vision community. Figure 1.3 shows an example of them.



Figure 1.3: Examples of handwritten text (left), machine printed text (center), and scene text (right).

- **Handwritten Text:** Handwritten text which is photographed in foreground.
- **Machine Printed Text:** Machine printed text which is photographed in foreground.
- **Scene Text:** Text that appears integrated in a scene with other visual information.

1.1.4 Image and Text Semantic Relations

Images and text, regardless of their composition, can have different semantic relations between them. Figure 1.4 shows an example of an image with associated captions with different semantic relations.

Synergetic

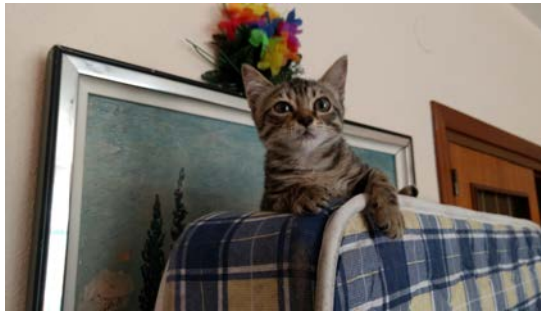
Image and text carry overlapping information. An example would be an image of a bus and a text describing the bus appearance.

Complementary

Image and text carry different information that together make a context, which explains more than any modality on its own. An example would be an image of a bus with a caption mentioning its destination.

Not Relevant Semantic Relation

Text does not add relevant context to interpret the scene. An example would be an image of a bus with a caption about climate change.



Synergetic	Complementary	Not Relevant
A gray and white cat on top of a mattress.	The most common pet in Barcelona	Hello summer!

Figure 1.4: Images and associated text with different semantic relations.

1.2 Images Associated Text as Supervision

Supervised machine learning learns from samples with annotations. In the image understanding field, a typical supervised learning scenario is learning from images with associated labels. Those labels indicate which categories (or classes) an image belongs to, and usually have an associated textual descriptor. The labels have a synergetic semantic relation with the image, since they refer to categories visually identifiable in the image. A drawback of supervised learning, and specifically supervised deep learning, is that a huge amount of annotated data is required to achieve good performances, and manual data annotation is an expensive process. An alternative to supervised learning is to learn from images with associated text. That is possible if images and text have a synergetic semantic relation. In that scenario, the images and the text carry overlapping information, as it happens in supervised learning with images and labels. Similarly as we do in the later, we can exploit the text as supervision to learn visual features.

The text supervision setup closer to a traditional supervised scenario is when the associated text are image tags. Tags, or hashtags, are also words associated to images, but should not be confused with supervised labels: The differences are that hashtags are not limited to a predefined set of classes, do not have a fix semantic relation with the image, and are not intentionally assigned to images for learning purposes. Text with other formats associated to images can be also used as supervision, as we will see in Chapter 4. The benefits of learning with text supervision compared to traditional supervised learning, is that paired images and text are available in the Web and Social Media unlimited. The challenges are that image and text associations are noisy and unconstrained: For a successful learning with text supervision, the paired images and text are required to have a synergetic semantic relation, but that is hard to guarantee in Web and Social Media data.

1.3 Outline, Research Questions and Contributions

In this section we summarize the content and the opened research questions of each one of the chapters of this thesis.

Chapter 2

We explain background on neural networks optimization, explaining deeply Cross-Entropy Loss and Ranking Loss, which are the ones we experiment with later in our research. We also explain briefly state of the art text representation methods.

Chapter 3

Research Question 1: *What kind of visual features does a neural network trained to detect text in an image learn?*

Research Question 2: *Can we train a neural network to learn and transfer text styles?*

We develop models to detect pixels containing text in an image, analyze which visual features a neural network learns to detect text, and improve the former state of the art scene text detection pipeline. An important observation is that we learn texture textual features hence text detection can be script independent, as shown in Figure 1.5. Also, we propose a model able to change the text style in an image learning from a single sample (see Figure 1.6), and demonstrate that it is a useful data augmentation technique to train an scene text detector.

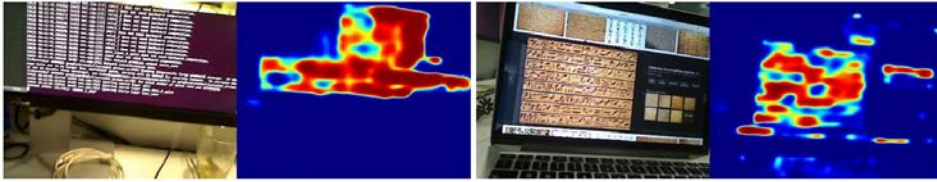


Figure 1.5: Results of pixel level text detection (explained in Chapter 3)



Figure 1.6: Results of text style transfer on scene text images (explained in Chapter 3)

Chapter 4

Research Question 3: *How can we exploit Social Media images with associated captions to learn visual features?*

Research Question 4: *What are the possible applications of learning with text supervision from Social Media data?*

We work with Social Media data consisting of images with an associated textual caption, using the text as supervision to learn visual features. We learn image semantic embeddings in different textual semantic embedding spaces and present results on multimodal image retrieval, as the ones shown in Figure 1.7, which show the potential of this methodology. The presented system is able to retrieve images semantically related to a given word, an arithmetic combination of words or a multimodal query formed by a word and an image. Additionally, we present a more specific application of that methodology, where we work with Instagram data related to Barcelona to learn which visual features different language speakers associate with the neighbourhoods of Barcelona.



Figure 1.7: Image retrieval results for multimodal queries composed by an image and text (explained in Chapter 4)

Chapter 5

Research Question 5: *Which is the best method to learn from images with associated tags?*

Research Question 6: *Can location information be exploited to boost image tagging performance?*

Research Question 7: *Can we train a retrieval system to find images associated to a given tag and near to a given location?*

We extend the work on joint visual and textual modeling to an additional modality: location. Learning from a large scale dataset of Flickr images with tags and location in form of latitude and longitude. First, we benchmark different methods to learn visual features from images with associated tags. Second, we propose a model able to exploit location information in the image tagging task and to retrieve images related to a given tag and near to a given location, as shown in Figure 1.8.

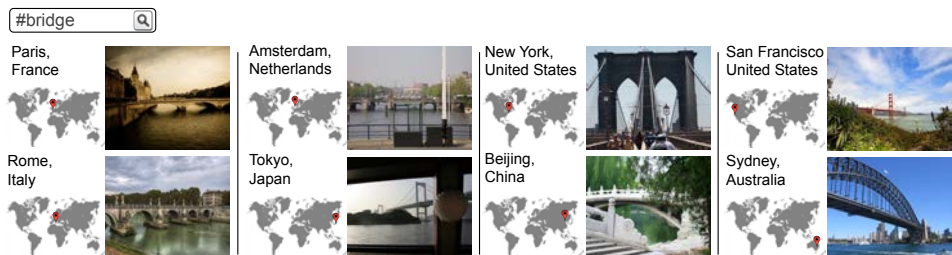


Figure 1.8: Image retrieval results for *bridge* and different query locations (explained in Chapter 5)

Chapter 6

Research Question 8: *Can we exploit the joint context provided by textual and image data for hate speech classification?*

We address the task of hate speech detection on Twitter publications formed by a text and an image. We create and annotate the first dataset for multimodal hate speech detection and propose different models that jointly analyze visual and textual information. We consider both the text associated to the images (in this case the tweet text) and the text that might appear in the image (see Figure 1.9).

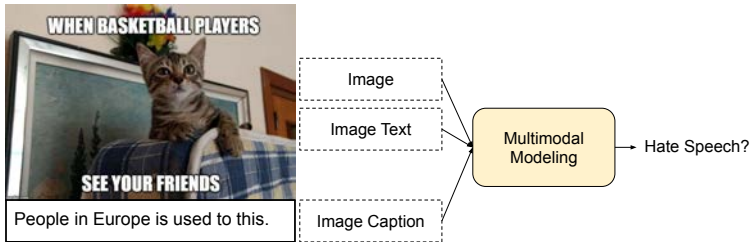


Figure 1.9: Multimodal hate speech detection setup (explained in Chapter 6)

Chapter 2

Background

Before the deep learning revolution image understanding pipelines were based on handcrafted features. That is, on manually designed feature extractors, such as HOG [11] or SIFT [58], that aimed to extract image representations useful for image understanding tasks. In 2012 Alex Krizhevsky *et al.* won by a big margin the Imagenet [37] image classification competition with AlexNet [46], a Convolutional Neural Network. Since then, CNN and deep learning based approaches have outperformed traditional techniques in all the image understanding tasks, such as image classification, object detection or image retrieval. CNN do not require human designed feature extractors. They take as inputs raw image pixels data and are optimized to solve a given objective task. They automatically learn, from raw image data, to extract image features useful for the optimization, features that have been proven to be very powerful for many different tasks, even when the CNN has not been specifically optimized for them.

The drawback of CNN, and in general of deep learning techniques, is that they are data hungry, much more than traditional image understanding techniques. To learn features from raw image pixels that generalize, they need to be trained with a big amount of annotated data, which is usually expensive to obtain. In the Chapters 3 and 4 of this dissertation, however, we propose methods to learn from images with associated text we can find in Web and Social Media, which allows to learn powerful image representations disregarding image annotations.

Neural Networks are trained by optimizing loss functions which have as inputs the network outputs. At each training forward pass the gradient of the loss functions with respect to the network outputs is computed, and then the gradient with respect to the rest of network neurons, using the chain rule, until the inputs. In the backward pass, the parameters of the network are updated to reduce the loss for that forward pass, a process which is known as backpropagation [76]. A multitude of loss functions have been proposed to optimize neural networks. However, there are two losses, or loss families, that are widely used: Cross-Entropy Loss and Ranking Loss. In this dissertation we experiment with them and compare their results in different tasks and setups.

This thesis covers different tasks involving textual and visual data. To ease its reading, we include at the beginning of each chapter a background section covering the basis and the related work of the tasks related to the Chapter, while in this Background Chapter we review Cross-Entropy Loss, Ranking Loss and Text Representations, which are general concepts we will go back to across the dissertation.

2.1 Cross-Entropy Loss

2.1.1 Classification Tasks

We start explaining two classification tasks where Cross-Entropy Loss is commonly used: multi-class classification and multi-label classification, illustrated in Figure 2.1.

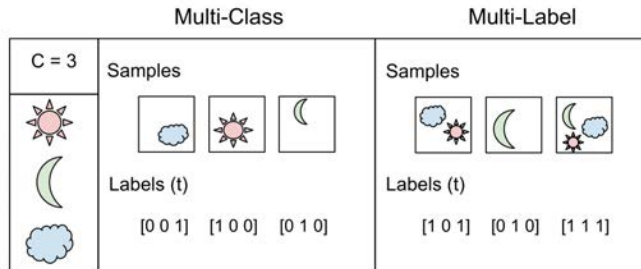


Figure 2.1: Multi-class and multi-label classification tasks examples in a setup with three classes: Sun, moon and cloud.

Multi-Class Classification

One-of-many classification. Each sample can belong to one of C classes. The network will have C output neurons that can be gathered in a vector of scores s . The target (ground truth) vector t will be a one-hot vector with one positive class and $C - 1$ negative classes. This task is treated as a single classification problem of samples in one of C classes.

Multi-Label Classification

Each sample can belong to more than one class. The network will have as well C output neurons. The target vector t can have more than a positive class, so it will be a vector of 0s and 1s with dimensionality C . This task is treated as C different binary and independent classification problems, where each output neuron decides if a sample belongs to a class or not.

2.1.2 Activation Functions

Activation functions are sometimes applied to the outputs of a network before inputting them to the loss. Two activation functions are commonly used:

Sigmoid

It squashes a vector in the range $[0, 1]$. It is applied independently to each element of the scores vector s . It's also called logistic function. The Sigmoid function formulation is:

$$(s_i) = \frac{1}{1 + e^{-s_i}} \quad (2.1)$$

Softmax

It squashes a vector in the range $[0, 1]$ and all the resulting elements add up to 1. It is applied to the output scores s . As scores in s represent a class, they can be interpreted as class probabilities. The Softmax function cannot be applied independently to each element s_i in s , since it depends on all elements of s . For a given class s_i , the Softmax function can be computed as:

$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \quad (2.2)$$

2.1.3 Formulation

The Cross-Entropy loss is defined as:

$$CE = - \sum_i^C t_i \log(s_i) \quad (2.3)$$

Where t_i and s_i are the groundtruth and the network score for each class i in C . As usually an activation function (Sigmoid or Softmax) is applied to the scores before the Cross-Entropy Loss computation, we write $f(s_i)$ to refer to the activations.

In a binary classification problem, where $C' = 2$, the Cross Entropy Loss can be

defined also as:

$$CE = - \sum_{i=1}^{C'=2} t_i \log(s_i) = -t_1 \log(s_1) - (1 - t_1) \log(1 - s_1) \quad (2.4)$$

Where it is assumed that there are two classes: C_1 and C_2 . $t_1 \in [0, 1]$ and s_1 are the groundtruth and the scores for C_1 , and $t_2 = 1 - t_1$ and $s_2 = 1 - s_1$ are the groundtruth and the score for C_2 . That is the case when we split a multi-label classification problem in C binary classification problems.

Often two different types of Cross-Entropy Loss are distinguished, depending on the activation function they are preceded by.

Categorical Cross-Entropy Loss

Is a name used for the combination of a Softmax activation function and a Cross-Entropy Loss. This setup is also called Softmax Loss. When we use this loss, we train a neural network to output a probability over the C classes for each sample. This loss is commonly used in multi-class classification tasks, where the labels are one-hot and only the positive class C_p keeps its term in the loss. Figure 2.2 shows an example of Categorical Cross-Entropy Loss applied to a Multi-Class Classification problem. There is only one element of the target vector t which is not zero (t_p) so, discarding the elements of the summation which are zero due to target labels, we can write the loss as:

$$CE = -\log \left(\frac{e^{s_p}}{\sum_j^C e^{s_j}} \right) \quad (2.5)$$

Binary Cross-Entropy Loss

Binary Cross-Entropy loss is a name used for the combination of a Sigmoid activation function and a Cross-Entropy Loss. This setup is also called Logistic Loss. Unlike Categorical Cross-Entropy Loss it is independent for each output of the neural network, meaning that the loss for each one of them is not affected by other network output values. It is commonly used for multi-label classification, where the insight of an element belonging to a certain category should not influence the decision for other classes. Figure 2.3 shows an example of Binary Cross-Entropy Loss applied to a Multi-Label Classification problem. It is called Binary Cross-Entropy Loss because it sets up a binary classification problem between $C' = 2$ classes for every class in C , as explained above. So when using this loss, the expression of Cross-Entropy Loss for binary problems is

often used. Being $f()$ the Sigmoid function, it can be expressed as:

$$CE = - \sum_{i=1}^{C'=2} t_i \log(f(s_i)) = -t_1 \log(f(s_1)) - (1 - t_1) \log(1 - f(s_1)) \tag{2.6}$$

Which can also be written as:

$$CE = \begin{cases} -\log(f(s_1)) & \text{if } t_1 = 1 \\ -\log(1 - f(s_1)) & \text{if } t_1 = 0 \end{cases} \tag{2.7}$$

Where $t_1 = 1$ means that the class C_1 is positive for this sample.

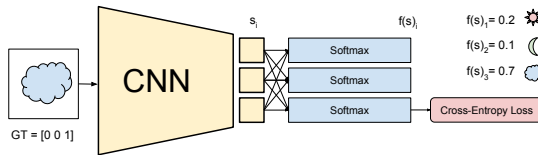


Figure 2.2: Categorical Cross Entropy Loss applied to a Multi-Class Classification problem. Note that each Softmax activation function depends on all the network outputs, and that their outputs sum up to one, and therefore can be interpreted as class probabilities. Only the positive class (cloud in this example) keeps its term in the loss, since the others are multiplied by 0.

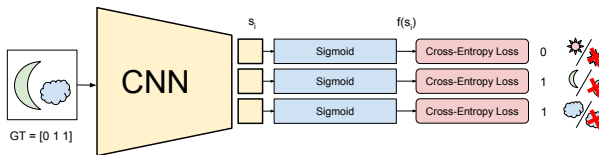


Figure 2.3: Binary Cross Entropy Loss applied to a Multi-Label Classification problem. Note that each class is treated as a binary classification problem dependent on a single CNN output, and that all the classes keep their term in the final loss.

2.2 Ranking Loss

2.2.1 Metric Learning

Unlike other loss functions, such as Cross-Entropy Loss, whose objective is to learn to predict directly a label, a value, or a set of values given an input, the objective of

Ranking Losses is to predict relative distances between inputs. This task is often called metric learning. Ranking Losses are very flexible in terms of training data: We just need a similarity score between data points to use them. That score can be binary: similar or dissimilar.

To use a Ranking Loss function we first extract features from two (or three) input data points and get an embedded representation for each one of them. Then, we define a metric function to measure the similarity between those representations, for instance the Euclidian distance. Finally, we train the feature extractors to produce similar representations for both inputs, in case the inputs are similar, or distant representations for the two inputs, in case they are dissimilar. We do not care about the values of the representations, only about the distances between them. However, this training methodology has demonstrated to produce powerful representations for different tasks.

Different names are used for Ranking Losses, but their formulation is simple and invariant in most cases. We distinguish two kinds of Ranking Losses for two different setups: When we use pairs of training data points or triplets of training data points. Both of them compare distances between representations of training data samples

2.2.2 Pairwise Ranking Loss

In this setup positive and negative pairs of training data points are used. Positive pairs are composed by an anchor sample x_a and a positive sample x_p , which is similar to x_a in the metric we aim to learn, and negative pairs composed by an anchor sample x_a and a negative sample x_n , which is dissimilar to x_a in that metric. This loss is sometimes called Contrastive Loss, referring to the fact that is contrasting two or more data point representations. If we wanted to use it to train a network for face image verification, positive pairs would be formed by two images of the same person, while negative pairs by two images of different people, as shown in Figure 2.4.

The objective is to learn representations with a small distance d between them for positive pairs, and greater distance than some margin value m for negative pairs. Pairwise Ranking Loss forces representations to have 0 distance for positive pairs, and a distance greater than a margin for negative pairs. Being r_a , r_p and r_n the sample representations and d a distance function, we can write:

$$L = \begin{cases} d(r_a, r_p) & \text{if } \textit{PositivePair} \\ \max(0, m - d(r_a, r_n)) & \text{if } \textit{NegativePair} \end{cases} \quad (2.8)$$

For positive pairs, the loss will be 0 only when the net produces representations for both the two elements in the pair with zero distance between them, and the loss (and therefore, the corresponding net parameters update) will increase with that distance.

For negative pairs, the loss will be 0 when the distance between the representations of the two pair elements is greater than the margin m . But when that distance is not

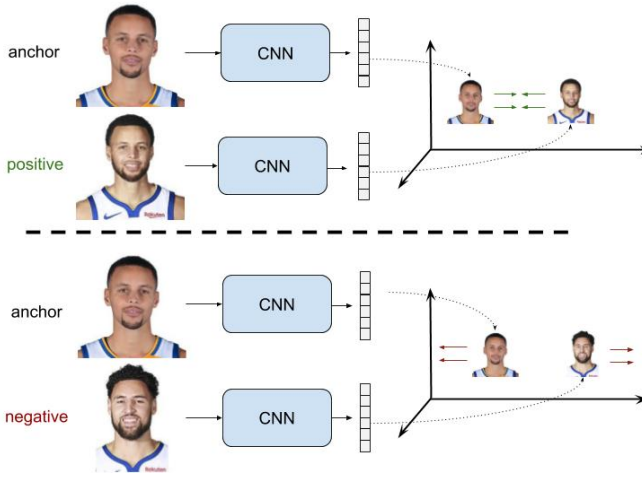


Figure 2.4: Pairwise ranking loss use case example to train an face image verification Convolutional Neural Network. The CNN is trained to embed images of the same person nearby.

bigger than m , the loss will be positive, and net parameters will be updated to produce more distant representation for those two elements. The loss value will be at most m , when the distance between r_a and r_n is 0. The function of the margin is that, when the representations produced for a negative pair are distant enough, no efforts are wasted on enlarging that distance, so further training can focus on more difficult pairs.

If r_0 and r_1 are the representations of a pair of samples, t is a binary flag equal to 0 for a negative pair and to 1 for a positive pair and the distance d is the euclidian distance, we can equivalently write:

$$L(r_0, r_1, t) = t \|r_0 - r_1\| + (1 - t) \max(0, m - \|r_0 - r_1\|) \quad (2.9)$$

2.2.3 Triplet Ranking Loss

This setup generally outperforms the former by using triplets of training data samples, instead of pairs. It was introduced in [94, 81]. The triplets are formed by an anchor sample x_a , a positive sample x_p and a negative sample x_n . It is sometimes called Triplet Loss. The objective is that the distance between the anchor sample and the negative sample representations $d(r_a, r_n)$ is greater (and bigger than a margin m) than the distance between the anchor and positive representations $d(r_a, r_p)$. If we wanted to use it to train a network for face image verification triplets would be formed by an anchor image of a given person, a positive image of the same person, and a negative image of

another person, as shown in Figure 2.5. With the same notation, we can write:

$$L(r_a, r_p, r_n) = \max(0, m + d(r_a, r_p) - d(r_a, r_n)) \quad (2.10)$$

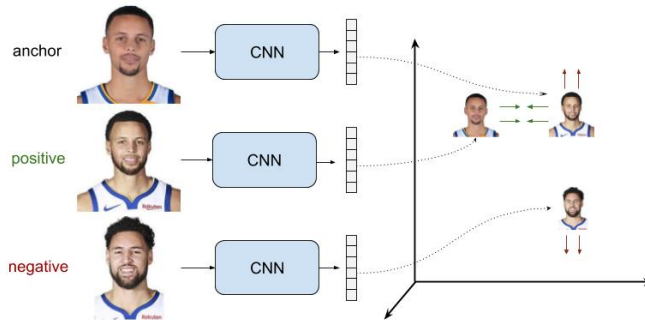


Figure 2.5: Triplet Ranking Loss uses case example to train an image face verification Convolutional Neural Network.

Let's analyze three situations of this loss (shown in Figure 2.6):

- **Easy Triplets:** $d(r_a, r_n) > d(r_a, r_p) + m$. The negative sample is already sufficiently distant from the anchor sample with respect to the positive sample in the embedding space. The loss is 0 and the net parameters are not updated.
- **Hard Triplets:** $d(r_a, r_n) < d(r_a, r_p)$. The negative sample is closer to the anchor than the positive. The loss is positive (and greater than m).
- **Semi-Hard Triplets:** $d(r_a, r_p) < d(r_a, r_n) < d(r_a, r_p) + m$. The negative sample is more distant to the anchor than the positive, but the distance is not greater than the margin, so the loss is still positive (and smaller than m).

An important decision of training with Triplet Ranking Loss is negatives selection or triplet mining. The strategy chosen will have a high impact on the training efficiency and final performance. An obvious appreciation is that training with Easy Triplets should be avoided, since their resulting loss will be 0. Early strategies used offline triplet mining, which means that triplets were defined at the beginning of the training, or at each epoch. Later, online triplet mining, meaning that triplets are defined for every batch during the training, was proposed and resulted in better training efficiency and performance. The optimal way for negatives selection is highly dependent on the task.

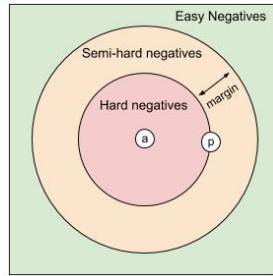


Figure 2.6: Representation of three types of negatives for a triplet with anchor "a" and positive "p" samples.

2.2.4 Siamese and Triplet Nets

Siamese and triplet nets are training setups where Pairwise Ranking Loss and Triplet Ranking Loss are used. But those losses can be also used in other setups. In these setups, the representations for the training samples in the pair or triplet are computed with identical nets with shared weights (which practically means with the same network).

Siamese Nets

Are built by two identical networks with shared weights (both networks have the same weights). Each one of these nets processes a sample and produces a representation. Those representations are compared and a distance between them is computed. Then, a Pairwise Ranking Loss is used to train the network, such that the distance between representations produced by similar samples is small, and the distance between representations of dissimilar samples is big. Since in a siamese net setup the representations for both elements in the pair are computed by the same network, being $f(x)$ that network, we can write the Pairwise Ranking Loss as:

$$L(x_0, x_1, y) = y \|f(x_0) - f(x_1)\| + (1 - y) \max(0, m - \|f(x_0) - f(x_1)\|) \quad (2.11)$$

Triplet Nets

The idea is similar to a siamese net, but a triplet net has three branches (three networks with shared weights). The model is trained by simultaneously giving a positive and a negative image to the corresponding anchor sample, and using a Triplet Ranking Loss. That lets the net learn better which samples are similar and different to the anchor image. In the case of triplet nets, since the same network $f(x)$ is used to compute the

representations for the three triplet elements, we can write the Triplet Ranking Loss as:

$$L(x_a, x_p, x_n) = \max(0, m + \|f(x_a) - f(x_p)\| - \|f(x_a) - f(x_n)\|) \quad (2.12)$$

Extended explanations of Cross-Entropy Loss and Ranking Loss are available in the author's blog ¹

2.3 Text Representations

Text representation methods are diverse in terms of architecture and the text structure they are designed to deal with. Some methods are oriented to learn representations for individual words and others for full texts or paragraphs. They have shown to generate powerful semantic representations, and in this work we do not focus on improving them, but experimenting on how they can be exploited to learn visual features when using text associated to images as supervision, as explained in Chapter 4. Here we explain briefly the main characteristics of the top-performing text representation methods.

LDA

Latent Dirichlet Allocation [7] learns latent topics from a collection of text documents and maps words to a vector of probabilities over those topics. It can describe a document by assigning topic distributions to it, which in turn have word distributions assigned. An advantage of this method is that it provides interpretable topics. LDA representations for documents tend to be sparse, meaning that most values are zero, so we can easily interpret to which topics a document is related to by inspecting the positive values. It is trained on collections of documents and, even though it can generate representations for words or captions, it is mainly used to represent documents, meaning long texts (articles).

Word2Vec

Word2Vec [62] Learns representations for words based on their context using a single hidden layer feed-forward neural network. It has two variants: In the CBOW (Continuous Bag of Word) approach, the neural network is trained to predict a word given as input its surrounding context (surrounding words). In the Skip-gram model, opposite

¹The author of this dissertation wrote 2 blog posts explaining Cross-Entropy and Ranking Losses which were referenced in the fast.ai *Deep Learning* course and the deeplearning.ai *Introduction to TensorFlow course*, and received considerable attention and feedback.

Cross-Entropy Loss: https://gombru.github.io/2018/05/23/cross_entropy_loss

Ranking Loss: https://gombru.github.io/2019/04/03/ranking_loss/

to the CBOW model, the neural network is trained to predict a word context given that word as an input. In the research presented here we used the most extended and efficient CBOW approach.

Differently from LDA, Word2Vec vectors are dense and not interpretable. Word2Vec learns a word embedding space with semantic structure, where we can apply arithmetics between word representations and get the corresponding semantic result: As an example, if we sum the Word2Vec representation of *king* and the one of *woman* we'll get a vector close to the Word2Vec representation of *queen*. This is not a unique feature of Word2Vec, the other text representations methods also learn embeddings with a semantic structure, but Word2Vec is commonly known for this fact, since it was the first demonstrating its strong semantic representations. WordVec is trained with plain text, and it generates representations for words. However, it has been widely used to generate representations for captions or articles by averaging their words representations, as we do in Chapter 4.

Doc2Vec

Doc2Vec [49] Extends the Word2Vec idea to documents, being able to create a numeric representation for them, regardless of their length. Extending Word2Vec CBOW model, it adds another input vector to the input context, which is the paragraph identifier. When training the word vectors, the document vector is trained as well, and at the end it holds a numeric representation of the whole document. As with Word2Vec, in this research we used the CBOW approach.

GloVe

GloVe [69] is a count-based model. It learns the vectors by essentially doing dimensionality reduction on the co-occurrence counts matrix. Training is performed on aggregated global word-word co-occurrence statistics from a corpus.

Despite its essential differences with Word2Vec, GloVe has some similarities with Word2Vec regarding its word representations: They are not interpretable and they also have a strong semantic structure. GloVe is trained in plain text and, despite it only generates word representations, it is also commonly used to represent captions or articles by averaging their words representations, as we do in Chapter 4.

FastText

FastText [8] is an extension of Word2Vec which treats each word as composed of character n-grams, learning representations for n-grams instead of words. The idea is to take into account and exploit the morphology of words. Each word is split in n-grams which are all input separately to the model, which can be trained using the CBOW or

the skip-gram approach. The vector for each word is made of the sum of its character n grams, so it can generate embeddings for out of vocabulary words. By exploiting the words' morphology, FastText tries to generate better embeddings for rare words, assuming their character n-grams are shared with other words. It also allows to generate embeddings for out of vocabulary words. In this research we used the originally proposed and most extended skipgram approach.

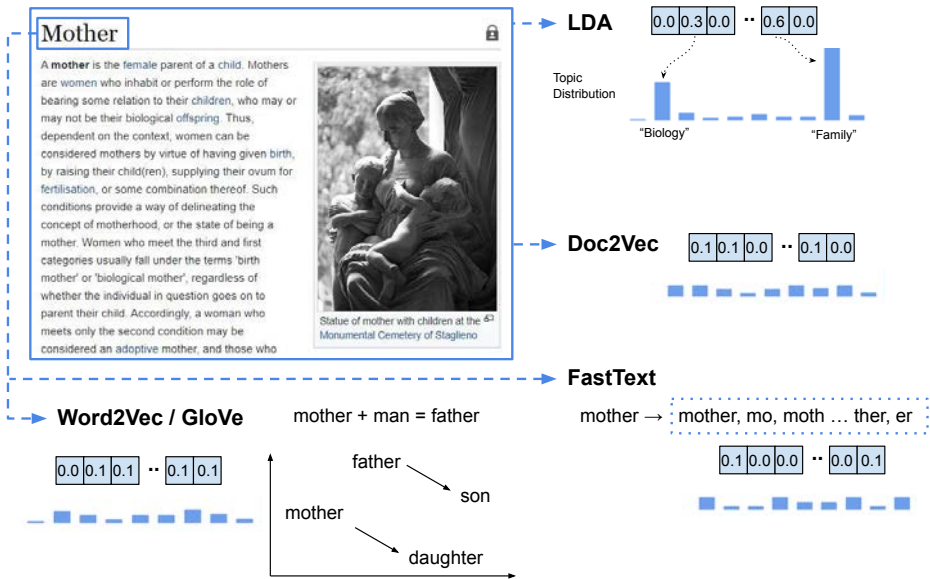


Figure 2.7: Illustration of the main features and typical use cases of the explained text representation methods.

Figure 2.7 shows the main features of these text representations methods. Note that its purpose is only to illustrate a simplification of the typical use cases of each of them, but they are not limited to them (for instance LDA can also be used to generate representations for single words). In this section we have broadly described state of the art text representations, which we exploit in Chapter 4 for using text as supervision and multimodal retrieval. However, this dissertation focuses on exploiting this text representations to learn visual features, and not on deeply analyze them. We refer the reader to the original text representation publications for that.

Chapter 3

Scene Text Segmentation and Style Transfer

Research Question 1: *Which visual features does a neural network trained to detect text in an image learn?*

Research Question 2: *Can we train a neural network to learn and transfer text styles?*

In this Chapter we present *TextFCN*, a Fully Convolutional Network trained to segment scene text in images at pixel level. We analyze the visual features it learns to detect text concluding that it learns generic text texture features which are robust to any language or alphabet, instead of features for each character. Then we show how *TextFCN* can be used to improve the efficiency of a text detection performance consisting on an object proposals algorithms followed by a text recognizer. Finally, we propose a text style transfer algorithm which exploits those generic texture text features to learn to transfer text styles learning the styles from a single image, and present results on handwritten text, machine printed text and scene text. Text style transfer is proven to be an effective data augmentation method to train scene text detectors.

3.1 Pixel Level Segmentation of Text in Images

3.1.1 Background on Fully Convolutional Networks

A Fully Convolutional Network (FCN) is a Convolutional Neural Network (CNN) without fully connected layers. Its architecture allows inputting images of varying sizes and output a tensor of the input's dimension. FCNs are often used for object segmentation, a task that consists on, given an input image, detect at pixel level certain objects

appearing in it: In other words, detect the pixels belonging to objects. In this setup, FCN's output a tensor of the same spacial dimensions as the input image, but having as much channels as the number of classes it aims to detect. Therefore, for each class it outputs a value per image pixels, which will be the score of that pixel for the class.

FCNs can be trained with multi-label or multi-class classification setups but, differently from entire images classification, to train FCNs we set an independent classification problem per each image pixel. Therefore, if we have an input image with $M \times N$ spatial dimensions and C classes to detect, in a multi-label segmentation setup we would have $M \times N \times C$ independent binary classification problems, while in a multi-class segmentation setup we would have $M \times N$ classifications problems depending in the scores off the different classes for a given pixel. Applying the Softmax activation function for a given class in the channel dimension, we get a $M \times N$ containing the probabilities of each pixel to belong to that class, which is often called heatmap. *TextFCN* is trained to predict the probability of each pixel in an image to be text.

3.1.2 TextFCN: Learning Text Visual Features

3.1.3 Methodology

TextFCN is based on the FCN proposed in [57], which is transformed from a VGG network [82]. We adapt it to output a tensor of $M \times N \times 2$ dimensions, where $M \times N$ are the input images size and each channel contains an score for each pixel and one of the two classes we define: text and no-text. We train the model in a multi-class setup consisting on a Softmax activation function plus a Cross-Entropy loss, as shown in Figure 3.1. Note that, in inference stage, applying the Softmax operator to the text class will provide a heatmap containing per-pixel text probabilities. We train *TextFCN* in the COCO-Text [89] dataset. It contains a total of 22,184 images with scene text (both legible and illegible) with axis-oriented bounding boxes annotations. *TextFCN* code and trained model are available on Github ¹.

3.1.4 Results

Figure 3.2 shows the heatmaps with pixel level text probability computed by TextFCN for different input images, where dark red represents a high text probability and dark blue a low one. The two images on top show how *TextFCN* is robust detecting texts of different shapes, sizes, colors or orientations. The result on mid-left shows, however, how it struggles to detect sets of lines of text that are equal and aligned. Result on mid-right shows how it successfully detects Egyptian hieroglyphs, and results on bot show that it also scores as text drawings with text-like textures. From the observation of this results, we can conclude that *TextFCN*, instead of patterns for specific characters, words or alphabets, is learning generic textual patterns to segment text. The reason

¹TextFCN code and model: <https://github.com/gombru/TextFCN>

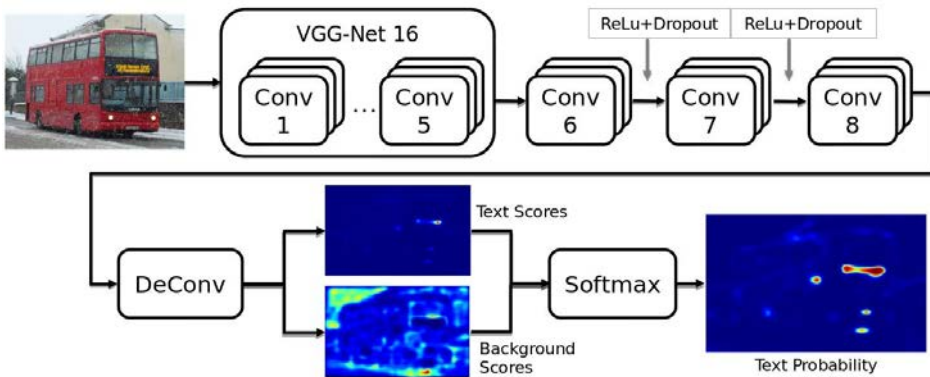


Figure 3.1: Architecture of the Fully Convolutional Network used in our pipeline for text prediction.

of the failure on the equal and aligned lines of the result in mid-left, is probably that *TextFCN* has learnt that text texture is not equal line-wise, and therefore, despite of the presence of characters, it scores that area as no-text, while scoring other areas without characters but with text-like texture as text, as seen on the results on bot.

3.1.5 TextFCN to Improve Scene Text Detection

Background on Scene Text Detection Pipelines

At the time this research was conducted, the state of the art scene text detection pipeline consisted on an object proposals method followed by a text recognizer. The object proposals methods were usually based in simple handcrafted features and their objective was to found bounding boxes susceptible to contain words with a high recall. Then, a text recognizer evaluated those bounding boxes, either recognizing a word or discarding it as not textual. Specifically, the pipeline that used TextProposals [21] algorithm for finding text boxes and the word recognizer proposed in [36] achieved state of the art results on end-to-end scene text recognition.

One drawback of that method is that TextProposals is computationally expensive. The second is that TextProposals [21] is able to find text boxes with high recall but is not able to rank them properly and achieve a decent precision. Therefore, in the end-to-end text understanding pipeline, the word recognizer [36] needs to evaluate a huge amount of bounding boxes, which is also computationally costly. In [5, 4] we proposed a method that exploits *TextFCN* by integrating it in TextProposals to palliate those two drawbacks.

After this work, the object proposals methods were replaced in state of the art text understanding pipelines by CNNs that performed text localization with high precision

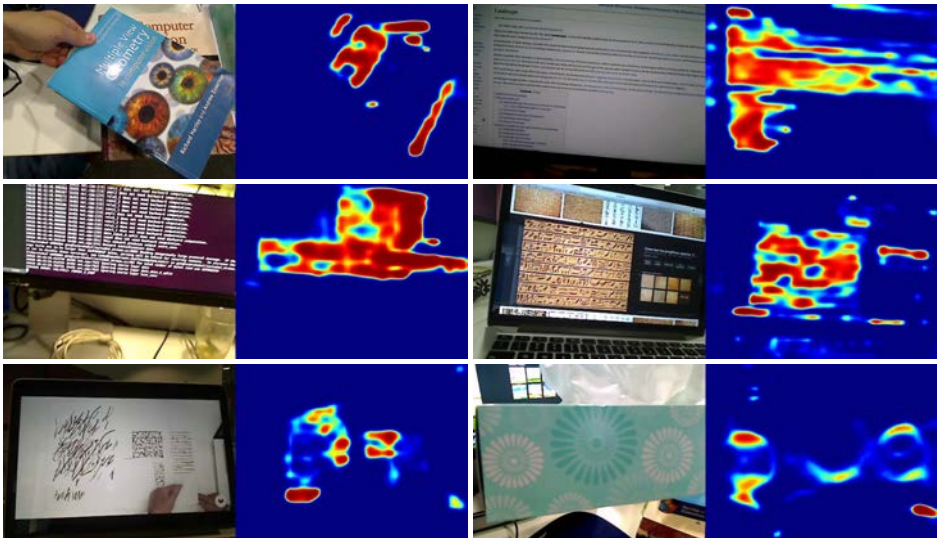


Figure 3.2: Input images and output TextFCN heatmaps showing the pixel level text probability.

while keeping a high recall. Thus methods, from which EAST [110] is an example, directly predict bounding boxes coordinates from image pixels. I experienced the transition from object proposals methods to CNN based text localizers while organizing the ICDAR 2017 Robust Reading Challenge on COCO-Text [27].

Methodology

TextProposals algorithm consist on an initial MSER (Maximally Stable Extremal Regions, which performs an oversegmentation of the input image from which we obtain a set of regions, and then a bottom-up agglomeration processes of that regions based on textual cues. That bottom-up agglomeration is a computationally expensive method, given that MSER produces a huge amount of regions, in both textual and not textual areas.

TextFCN computes a pixel-level text probability map which can be used to compute a text probability of each one of the MSER proposed regions very efficiently by simple averaging the text probability of their pixels. By eliminating not textual regions from the initial MSER segmentation setting and empirical text probability threshold we are able to reduce drastically the number of regions considered in the bottom-up agglomeration processes, and thus reduce the computational load. The number of suppressed regions, and thus the efficiency gain, will be ofcourse dependent on the textual area of the image. Given that that TextProposal is not accurate at ranking text boxes, it tends to include boxes in non textual regions in high ranking positions. Therefore,

the suppression of non textual areas in the process also results in a big reduction of the number of boxes that have to be evaluated by the text detector.

Results

Figure 3.3 shows the top 500 ranked regions of the baseline TextProposals methods and TextProposals with *TextFCN* suppression strategy. Note that TextProposals produces a high number of regions in not textual areas in the top 500 ranked results, which the TextFCN suppression avoids.

Table 3.1 proves that the former observation results in better text detection rates when a limited number of text regions is evaluated, and a small decay in performance when all regions are. Note that the number of regions proposed by TextProposals is huge, so evaluating all of them is not a realistic scenario, as we can conclude from the times shown in next Table 3.2. The later shows the execution times of each stage of the end-to-end text detection pipeline of the baseline pipeline using TextProposals and the pipeline incorporating TextFCN suppression considering all the regions proposed in the region proposal stage. Results prove that *TextFCN* suppression reduces drastically TextProposals execution time and, given the reduction of the number of output regions, word recognizer time.



Figure 3.3: Top 500 ranked regions by TextProposals (in red) and by TextProposals with the TextFCN suppression strategy (in green).

Conclusions

We have shown that *TextFCN*, a neural network trained for pixel level text segmentation, learns generic textual textures instead of character level features. This fact is ex-

Table 3.1: Detection rates of TextProposals and TextProposals with TextFCN Suppression considering the top 10, 100, 1000, and all proposals on COCO-Text datasets

#of Proposals	10	100	1000	All
Baseline	0.13	0.34	0.63	0.92
+ TextFCN Suppression	0.20	0.48	0.77	0.87

Table 3.2: Mean execution time per image (seconds) considering all proposals generated by the different proposed strategies (64k for baseline and 14k with the TextFCN suppression). Word recognition is performed with [36], which takes about 0.4 ms per bounding box. Experiments were conducted on a system with a GPU GeForce GTX TITAN and an Intel[®] Core[™]2 Quad CPU Q9300@ 2.50GHz processor

	TextFCN	TextProposals	Word Recognizer	Total
Baseline	0	4.11	25.69	29.80
+ TextFCN Suppression	0.15	1.33	5.58	7.06

exploited in next section for text style transfer. Also, we have seen that *TextFCN* is useful to dramatically increase the performance of an object proposals plus a word recognizer text understanding pipeline.

3.2 Selective Text Style Transfer For Text

3.2.1 Introduction

Style transfer is the task of combining the *style* of one image with the *content* of another image. Although the content of an image can be defined by the objects and the general scenery, the style of an image is not well defined. The style can be understood as the brush stroke of a painting, the color distribution, certain dominant forms and shapes or even a combination of all the above[20]. Previous style transfer works have focused on transferring paintings styles, where the features to be transferred encode the brush strokes, the cubist patterns or the color palette achieving fascinating results[20, 38, 15]. However, text characters are very particular objects for which the common understanding of content and style cannot be adopted. Instead we define the style of the text as the shape, color and background of the characters and the content as the transcription of the text.

In this section we present a text style transfer method which learns a text style from a single image and is able to transfer it to other images with text preserving their content. We propose two methods: A two-stage method and an end-to-end method. The

first one uses *TextFCN* to find the text pixels in the image where the style has to be applied. The second one is a model that learns to do text segmentation and text styling at once. Both methods exploit global textual texture features (as seen in Figure 3.2) to learn a text style from a single image and transfer it to any text. Instead of learning the specific style of each character, it learns and transfers general style features such as line width, tile, curvature, color or shadow.

The proposed method is able to automatically change the style of text regions in natural scene images, generating realistic images with the same textual content but with different text styles. In machine printed text images, it is able to train models that stimulate a change of the text font. In handwritten text, it is able to transfer the writing style of a particular writer to another. Furthermore, we demonstrate that such an approach is useful as a data augmentation technique to train an scene text detector.

3.2.2 Background on Text Style Transfer

Gatys *et al.* [20] propose a method using a pretrained VGG network [82] for style transfer by extrapolating a randomly generated image to a stylized image. The stylized image is obtained by computing the backward propagation on the resulting pixel values which is computationally demanding. Johnson *et al.* [38] recast the problem as an image transformation task, where a single, fixed learnt painting style is applied to an arbitrary image. A CNN is trained to alter a corpus of content images to match the style of a painting, eventually allowing to stylize images in real time. Simultaneously, Ulyanov *et al.* [87] introduced the idea of Instance Normalization which is a modified version of Batch Normalization to have computationally less demanding models. A drawback of these works[38, 87] is that an independent model has to be trained for each source style. Dumolin *et al.* [15] overcomes it by proposing a single CNN that can learn to transfer different source styles (up to 32 in their experiments), allowing to generate images with combined styles.

Although style transfer has not been applied to text, other works have targeted the task of changing text style with different approaches. Liu *et al.* [56] proposed a pipeline to transform scene text into machine printed text within a scene text recognition model. Abe *et al.* [1] proposed a Generative Adversarial Network model to create new machine printed text fonts. Aksan *et al.* [2] propose a generative model to disentangle content and style of handwritten text represented as temporally ordered strokes, and apply it to handwriting synthesis and style transfer. More related to our work, Ankan *et al.* [48] focus on font to font translation in images of printed documents using a GAN architecture. Also, Azadi *et al.* [3] propose a conditional GAN to style machine printed text to more complex scene text fonts, learning each character style independently. Yang *et al.* [104] recently proposed a method able to change the text content of an scene text image keeping its original style. It is based on a content and style disentanglement method and a generative architecture that combines the extracted disentangled features.

3.2.3 Style Transfer

We use the model proposed by Dumoulin *et al.*[15] as our baseline. The style transfer task is usually defined as finding an image p which is produced by an encoder-decoder image transformation network, whose content is similar to a source content image c but whose style is similar to a source style image s . A key point of style transfer is the definition of both content and style. In [15], two images are considered to have a similar content if their high-level features extracted by a trained classifier are close in Euclidean distance. On the other hand, two images are similar in style if their gram matrices of low-level features as extracted by a trained classifier are close under the Frobenius norm.

More formally, let ϕ be the transformer network and γ^l the output of the l^{th} layer of a CNN pretrained on ImageNet[37]. In our case γ is the VGG-16[82]. The training process is as follows: we initially forward the content image c through the transformer network ϕ to obtain the stylized image p . All images c, s, p are then forwarded through γ , and features for them are extracted: correspondingly, F_c for the content image from the m^{th} layer, F_s for the style image from n^{th} layer and, F_{pm}, F_{pn} for the stylized image from both m^{th}, n^{th} layers where $m \geq n$. The content loss L_c is defined as the mean squared error between F_c and F_p . The style loss L_s is computed as the mean squared error between corresponding Gram matrices G_s and G_p of the features F_s and F_p . The final loss L_{total} that directs the model training is a weighted average of the content and the style losses. In summary:

$$\begin{aligned}
 p &= \phi(c) \\
 F_c &= \gamma^m(c), F_s = \gamma^n(s) \\
 F_{pm} &= \gamma^m(p), F_{pn} = \gamma^n(p) \\
 L_c &= \sum_k (F_{ck} - F_{pmk})^2 \\
 G_s &= F_s \cdot F_s^T, G_p = F_{pn} \cdot F_{pn}^T \\
 L_s &= \sum_k (G_{sk} - G_{pnk})^2 \\
 L_{total} &= \lambda_1 L_c + \lambda_2 L_s
 \end{aligned} \tag{3.1}$$

3.2.4 Selective Style Transfer for Text

Selective style transfer refers to automatically detecting the relevant areas in the image (in our case, areas where text is present) and restricting the application of style to the detected areas only, leaving the rest of the image unchanged. Accordingly, we design and describe two models that can perform selective style transfer for text.

Two-Stage Architecture

To stylize only textual areas of the images, we exploit *TextFCN*. To transfer a text style to an input image, we first stylize the whole image. The result of stylizing entire images is shown in Figure 3.4. Then, we compute the pixel-level heatmap of the original image with TextFCN. To generate the final image where the style is transferred only to textual areas, we do a blending of the original image and the stylized image weighted with the TextFCN heatmap (see Figure 3.5). This procedure allows to obtain realistic images, ensuring that non-textual areas are kept unchanged.

More formally, given a stylized image p and a pre-trained TextFCN δ , we process the content image c with δ to obtain the per-pixel text probability map P_t . Then we get only textual areas of the stylized image p by taking its Hadamard product with P_t , and do the same with c and $1 - P_t$ to get the content of non-textual areas. We sum up the results to get the final image p_{text} :

$$\begin{aligned}
 P_t &= \delta(c) \\
 p_{text} &= P_t \odot p + (1 - P_t) \odot c
 \end{aligned}
 \tag{3.2}$$



Figure 3.4: Results stylizing the whole image with three different text styles and the two-stage architecture.

End-to-end Architecture

The proposed end-to-end architecture that is capable of performing selective style transfer on text without needing any text detector. Our model is inspired by the distillation strategy from [32]. The basic idea of distillation is to pass the learnt information of various networks, which are able to solve different tasks, into a single model. In our

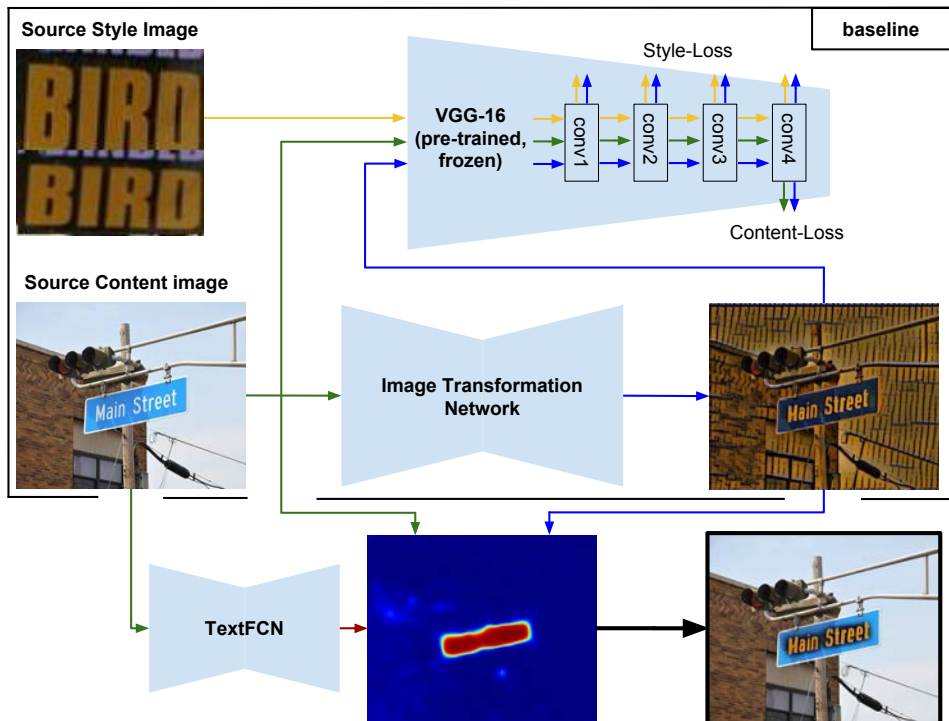


Figure 3.5: Two-stage Selective Text Style Transfer pipeline.

case, we combine the image style transformation network with the text detector. We take the pretrained image style transformation network and the ground truth annotations for the text to train a randomly initialized image transformation network with mean squared error loss (see Figure 3.6).

More formally, let ϕ be the pretrained image style transformation network, M the masks for the text regions where $M_{ij} \in \{0, 1\}$ and, η the same network as ϕ but randomly initialized. We first obtain the stylized image p forwarding the content image c through ϕ . We then obtain \hat{p}_{text} as the output of η after feeding it with the content image c . To get the ground truth for content and for style, θ_c and θ_s respectively, we apply the Hadamard product:

$$\begin{aligned}
 p &= \phi(c) \\
 \hat{p}_{text} &= \eta(c) \\
 \theta_s &= p \odot M \\
 \theta_c &= c \odot (1 - M)
 \end{aligned}$$

(3.3)

We use mean squared error as our loss to train the selective style transfer net η . There are two key points in our loss calculation. First of all, we need to apply the mask M for text regions, and $1 - M$ for content regions to \hat{p}_{text} to make sure our model learns to differentiate between text and background. Secondly, since text regions in the image are significantly smaller compared to background, we weight loss contributions of text and background pixels with two parameters λ_1, λ_2 , with $\lambda_1 > \lambda_2$. The loss is defined as:

$$L = \lambda_1 \sum (\hat{p}_{text} \odot M - \theta_s)^2 + \lambda_2 \sum (\hat{p}_{text} \odot (1 - M) - \theta_c)^2 \quad (3.4)$$

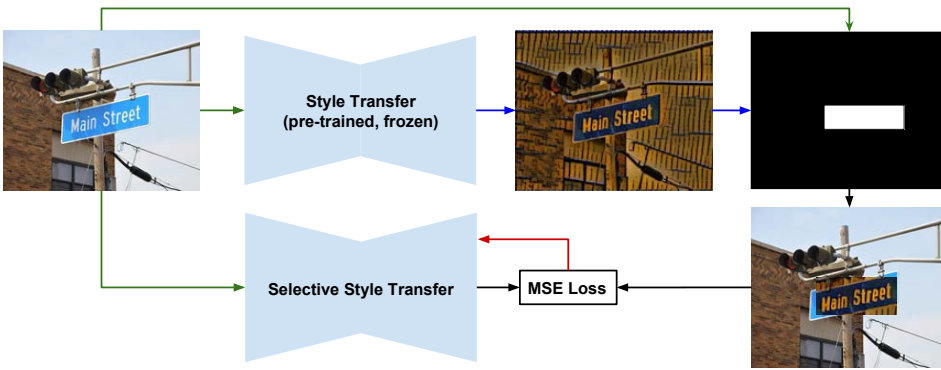


Figure 3.6: End-to-end Selective Text Style Transfer pipeline.

3.2.5 Experiments

To explore the capabilities of text style transfer in various text domains, we train 3 models, namely, a model to transfer scene text styles, a model to transfer machine printed text fonts styles, and a model to transfer handwritten styles.

Scene Text

We trained our baseline style transfer model for two-stage architecture with 34 scene text styles from COCO-Text dataset [89] using cropped word images as source styles and ImageNet [37] as the training dataset. Then, we trained our end-to-end model for selective text style transfer using the COCO-Text [89] dataset (only the images containing legible text).

Figure 3.7 shows results of the scene text model applied to COCO-Text scene images, using both the two-stage architecture and the end-to-end architecture to get the selective text style transfer results. The performance is appealing, transferring the

source styles with high fidelity in both character shapes and colors to a wide diversity of scene texts. The text content is preserved quite well in most images, and only in some cases where the task is very complex due to the original text size or tangled style the result is illegible. Figure 3.8 shows results of transferring two weighted styles to get intermediate text styles.

Machine Printed Text

We train our baseline model with 8 machine printed source text fonts: Arial, Times, Lubster, Corsiva, Caveat, Pacifico, Consolas and Syncopate and Imagenet [37] training images. Figure 3.9 shows results of the model applied to machine text. It transfers successfully the main features of the source font style, such as line width, text orientation, and main font character style. However, it fails transferring the specific styles of some characters, and the final output is influenced by the initial image.

Handwritten Text

The baseline model is trained with 8 styles from different writers, using images from the IAM dataset [61] as source styles and the ImageNet [37] dataset as training images. Figure 3.10 shows results of the model applied to IAM dataset images. The model transfers correctly the main features of the text, as the tight characters and the thick stroke of the style in the first column, and the elongated and italic style of the writer in the second column. However, it fails on transferring more fine-grained characteristics of the source writer style, and some words of the resulting text are blurry.

Cross Domain

We can go one step further and test the capability of our text style transfer models to transfer style to images of other text domains.

Machine Text to Scene Text.

Transferring scene text styles to machine printed text has a huge potential in augmented reality scenarios and as a data augmentation technique, generating synthetic images with a given text style but different text content. Figure 3.11 shows results of styling machine printed text with the scene text model. The model successfully transfers scene text style to machine printed text with high fidelity. Figure 3.13 shows some content augmentation results, where Arial text has been stylized with the destination scene text font, and the stylized text has been inserted in the image manually.

Handwritten to Scene Text.

Figure 3.11 shows results of styling handwritten text with the scene text model. This task is quite complex, since the scene text tends to be thick and detached while handwritten text is tangled formed by tangled thin strokes. However, the scene text style

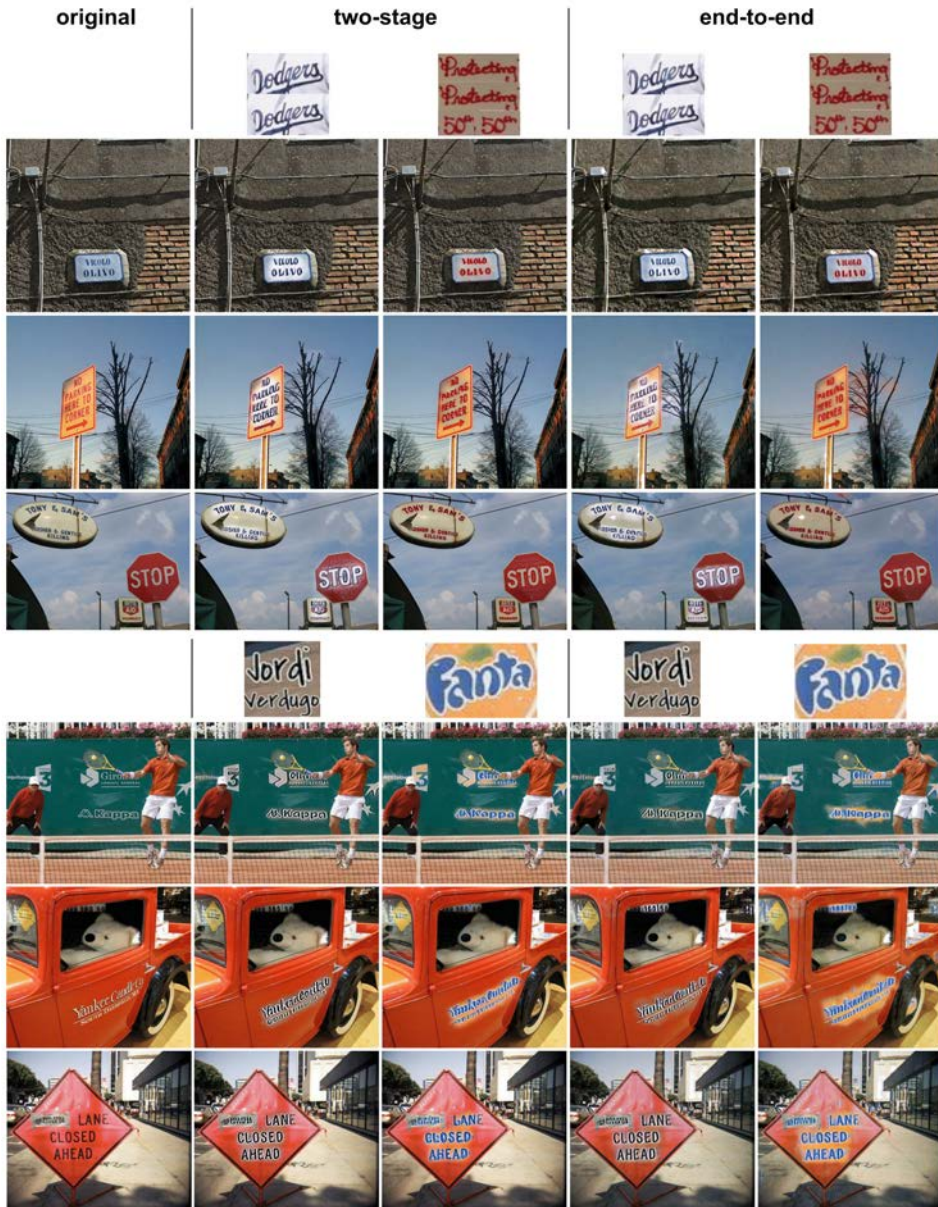


Figure 3.7: Applying different styles to various scene text images using the two-stage and end-to-end architectures with the cropped word styles.

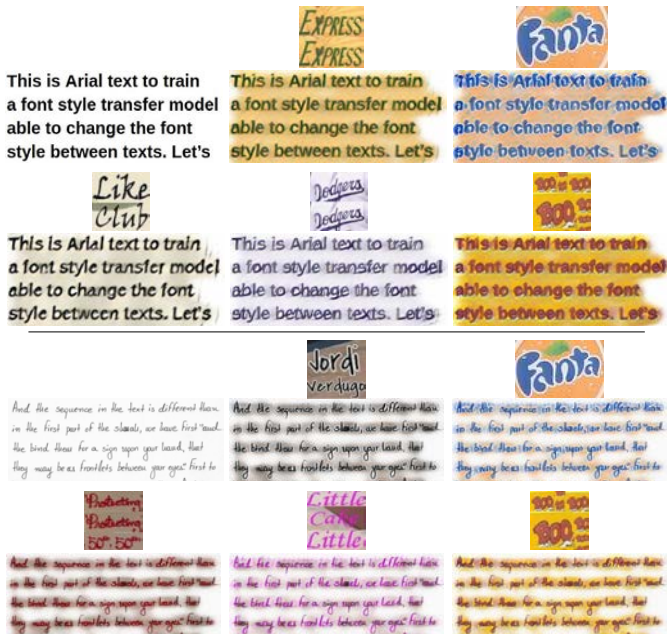


Figure 3.11: Results of the scene text model applied to machine printed text (top) and handwritten text (bottom) images.

transfer model successfully transfers some style features to it keeping it legible, and results are a combination of the transferred scene text style and the original handwritten style.

Scene Text to Machine Text. A model capable to convert any scene text to machine printed text would be a nice tool to improve scene text understanding pipelines. The machine printed text model allows us to do so, as shown in Figure 3.14. It correctly transfers the machine text source style if the scene text is simple, but it fails when scene text has a complex font, is too small or rotated. Note that the artifacts in those images are due to noisy high responses of the text detector.

Handwritten to Machine Text. Converting handwritten text to machine printed text could be very useful in a handwritten text understanding pipeline. Our machine text model transfers style features from machine fonts to handwritten text, but it breaks the content, resulting in illegible images, as shown in Figure 3.14.

Machine Text to Handwritten. Converting machine text to handwritten text can be a great tool to generate synthetic data to train handwritten text understanding models. However, our handwritten text model fails transferring the font styles to machine text, as shown in Figure 3.12. It only achieves to copy some general handwritten style features to some machine text fonts closer to handwritten styles, like Caveat.

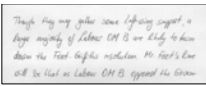
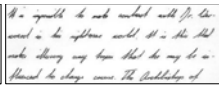
		
This is Arial text to train a font style transfer model able to change the font style between texts. Let's	This is Arial text to train a font style transfer model able to change the font style between texts. Let's	This is Arial text to train a font style transfer model able to change the font style between texts. Let's
<i>This is Caveat text to train a font style transfer modelable to change the font style between texts. Let's explain how</i>	<i>This is Caveat text to train a font style transfer modelable to change the font style between texts. Let's explain how</i>	<i>This is Caveat text to train a font style transfer modelable to change the font style between texts. Let's explain how</i>

Figure 3.12: Handwritten text model transferring styles (top) to machine text images (left).



Figure 3.13: Arial text has been stylized with the original image scene text style (left) and manually inserted (right).

3.2.6 Data Augmentation

Next Experiments demonstrate the usefulness of the proposed selective scene text style transfer as a data augmentation tool to improve text detectors' performance. In the experiments, we use the consolidated and widely used EAST [110] text detector². We consider the following datasets:

- **ICDAR 2013** [40]: the dataset contains 229 training images and 229 test images that capture focused text on sign boards, posters, etc.
- **ICDAR 2015** [39]: the dataset contains 1000 training images and 500 testing images with incidental scene text, which means text that appears in the scene without the user focusing on it.

²EAST text detector code: <https://github.com/argman/EAST>

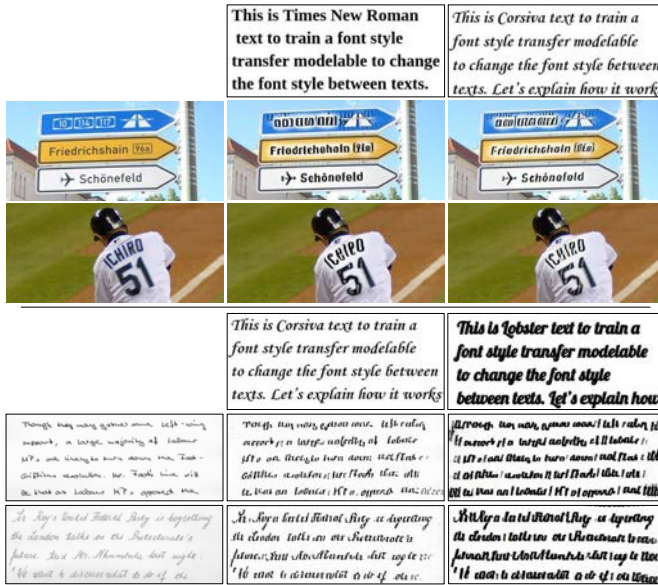


Figure 3.14: Results of transferring machine printed text styles to scene images (top) and handwritten images (bottom).

- **COCO-Text** [89]: the dataset contains 63k images from the COCO [53] dataset with text regions annotated.

For these experiments, we trained an additional scene text style transfer model using 96 different styles from the ICDAR 2015 training dataset, and used the two-stage architecture to perform selective text style transfer. We augment ICDAR 2013, ICDAR 2015 and COCO-Text datasets using the two-stage model with 1 or 4 random additional styles per image. We train the EAST text detector on the augmented and regular datasets. Stylizing COCO-Text with ICDAR 2015 training styles, allows to get a COCO-Text dataset closer to the target testing data, which is ICDAR 2015 testing set. To evaluate the trained models, we use the Robust Reading Competition framework (ICDAR 2015 Challenge 4: Incidental Scene Text Localization task and ICDAR 2013 Challenge 2: Focused Scene Text). Results in Table 3.3 show that text style transfer is a useful data augmentation technique, achieving an improvement in F-Score performance between 2-4% in all the setups just using 1 or 4 augmentations per image.

3.2.7 Conclusions

We have seen that a style transfer model is able to learn text styles as the characters shapes, line style, and colors, from a single image and transfer them to an input text preserving the original characters. To do so, it exploits generic texture text features, as

Training Dataset	Testing dataset	Augmentations	F
ICDAR 13	ICDAR 13	-	70.97
ICDAR 13	ICDAR 13	1 styles per image	74.55
ICDAR 13	ICDAR 13	4 styles per image	75.29
ICDAR 13+15	ICDAR 15	-	80.83*
ICDAR 15	ICDAR 15	-	78.74
ICDAR 15	ICDAR 15	1 style per image	80.60
ICDAR 15	ICDAR 15	4 styles per image	81.83
COCO-Text	ICDAR 15	-	68.05
COCO-Text	ICDAR 15	1 style per image	69.66
COCO-Text	ICDAR 15	4 styles per image	70.71

Table 3.3: Results (F-score) of the EAST text detector with different training data, evaluated on ICDAR 2015 Challenge 4 and ICDAR 2013 Challenge 2. *Result reported on the original EAST github.

TextFCN does to localize text. We have explored the performance of text style transfer in 3 text modalities: scene text, machine printed text and handwritten text and in cross-modal scenarios, proving the usefulness of text style transfer as a data augmentation technique to train scene text detectors.

Chapter 4

Exploiting Images Associated Text as Supervision

Research Question 3: *How can we exploit Social Media images with associated captions to learn visual features?*

Research Question 4: *What applications learning with text-supervision from Social Media data can have?*

4.1 Introduction

4.1.1 Self-Supervised Learning

Supervised learning refers to learning from samples with associated labels, and unsupervised learning refers to learn without labels. Those definitions are clear, but there are other terms referring to intermediate situations, such as weakly-supervised learning, semi-supervised learning, webly-supervised learning, or self-supervised learning, that have been using variably in the literature. In this chapter we research on how to learn from images with associated text using the text as supervision. In our previous publications, we have stated that the proposed techniques are self-supervised learning methods, based on the fact that we learn from raw data consisting on images and text associations that can be found in the Web, and the supervision comes from the raw data itself. However, we have found that some researchers consider only self-supervised learning methods that use solely one data modality, exploiting for instance it's structure as supervision as proposed in [13, 88, 14, 64, 55]. Other researchers admit as self-supervised learning methods that exploit paired data modalities such as paired image and audio [107, 75, 18, 74], but exclude text, as seen in this publication [9] related work section.

It is true that, when using text as supervision, the frontier between supervised learning and self-supervised learning is weak. As an example, if one learns from articles that include images, such as Wikipedia articles [22], the setup is clearly far from supervised learning, because Wikipedia articles cannot be considered labels. However, if one learns from images with tags [59], the setup is closer to supervised learning, since tags can easily be considered labels, even though they are noisy. In this chapter we learn from Web and Social Media data consisting on an image and an associated caption. We refer to the methods proposed as self-supervised learning techniques, because the supervision comes from the raw data found in the Web: its image and text associations. However, we understand the different definitions and understandings self-supervised learning can have. So from now on in this dissertation, we ask the reader to understand self-supervised learning as learning using images associated text as supervision.

4.1.2 Exploiting Multimodal Web Data

Lately, Web data has been used to build classification datasets, such as in the Web-Vision Challenge [50] and in this Facebook research [59]. In these works, to build a classification dataset, queries are made to search engines using class names and the retrieved images are labeled with the querying class. In such a configuration the learning is limited to some pre-established classes, thus it could not generalize to new classes. While working with image labels is very convenient for training traditional visual models, the semantics in such a discrete space are very limited in comparison with the richness of human language expressiveness when describing an image. Instead we define here a scenario where, by exploiting distributional semantics in a given text corpus, we can learn from every word associated to an image. The noisy and unstructured text associated to Web images provides information about the image content that we can use to learn visual features. A strategy to do that is to embed the multimodal data (images and text) in the same vectorial space.

Instagram is an image based social network where people tend to post high quality personal pictures accompanied by a caption. Captions are diverse, but they usually describe the photo content, the place where the photo was taken or the feelings the photo brings in. The objective of adding this text, which usually contains hashtags, is that other Instagram users can find the photo using one of the words and follow the author if they like what they post. Therefore, the semantic relation between the image and the text is usually synergetic, and that's why this data is adequate for self-supervised learning, were we aim to learnt from pair data modalities where pairs have an equivalent meaning. In this chapter we exploit Instagram data to learn the visual features users associate with words, and achieve impressive semantic multimodal image retrieval results learning solely from it. Also, we show that this technique has interesting and immediate applications: We learn which visual features Barcelona tourists and locals associate with its different neighbourhoods and show how that can be used to analyze tourism activity.

4.2 Background on Multimodal Image and Text Embeddings

Multimodal image and text embeddings have been lately a very active research area. The possibilities of learning together from different kinds of data have motivated this field of study, where both general and applied research has been done. DeViSE [65] proposes a pipeline that, instead of learning to predict ImageNet classes, learns to infer the Word2Vec [62] representations of their labels. The result is a model that makes semantically relevant predictions even when it makes errors, and generalizes to classes outside of its labeled training set. Gordo & Larlus [29] use captions associated to images to learn a common embedding space for images and text through which they perform semantic image retrieval. They use a *tf-idf* based BoW representation over the image captions as a semantic similarity measure between images and they train a CNN to minimize a margin loss based on the distances of triplets of query-similar-dissimilar images. Gomez, Patel *et al.* [22, 67] use LDA [7] to extract topic probabilities from a bunch of Wikipedia articles and train a CNN to embed their associated images in the same topic space. Wang *et al.* [96] propose a method to learn a joint embedding of images and text for image-to-text and text-to-image retrieval, by training a neural net to embed in the same space Word2Vec [62] text representations and CNN extracted features. Recently and related to our work, Vo *et al.* [93] work on image retrieval using multimodal queries formed by an image and a text. They compare different techniques to combine the image and textual queries and propose a new method to do so based on residual features.

Other than semantic retrieval, joint image-text embeddings have also been used in more specific applications. Patel *et al.* [68] use LDA [7] to learn a joint image-text embedding and generate contextualized lexicons for images using only visual information. Gordo *et al.* [28] embed word images in a semantic space relying in the graph taxonomy provided by WordNet [70] to perform text recognition. In a more specific application, Salvador *et al.* [79] propose a joint embedding of food images and their recipes to identify ingredients, using Word2Vec [62] and LSTM representations to encode ingredient names and cooking instructions and a CNN to extract visual features from the associated images.

4.3 Semantic Multimodal Image Retrieval

In this section we represent text using five different state of the art methods and eventually embed images in the learnt semantic space by means of a regression CNN. We compare the performance of the different text space configurations under a text based image retrieval task. We have published the research explained in this chapter in [25, 26]. Also, an online demo of the semantic multimodal image retrieval method pro-

posed here is available in ¹.

4.3.1 Multimodal Text-Image Embedding

The proposed pipeline is as follows: First, we train the text embedding model on a dataset composed by pairs of images and correlated texts (I, x) . Second, we use the text embedding model to generate vectorial representations of those texts. Given a text instance x , we denote its embedding by $\phi(x) \in \mathbb{R}^D$. Third, we train a CNN to regress those text embeddings directly from the correlated images. Given an image I , its representation in the embedding space is denoted by $\psi(I) \in \mathbb{R}^D$. Thereby the CNN learns to embed images in the vectorial space defined by the text embedding model. The trained CNN model is used to generate visual embeddings for the test set images. Figure 4.1 shows a diagram of the visual embedding training pipeline and the retrieval procedure.

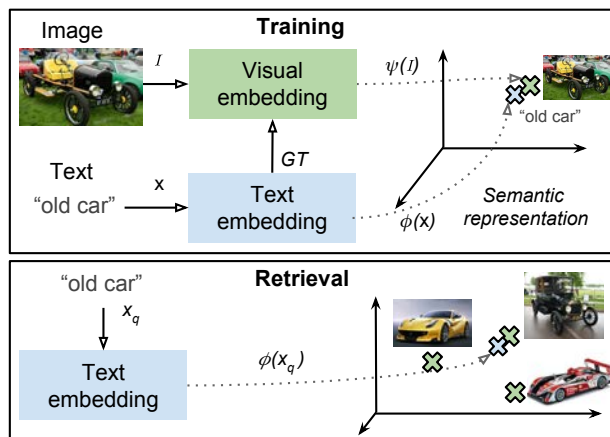


Figure 4.1: Pipeline of the visual embedding model training and the image retrieval by text.

In the image retrieval stage the vectorial representation in the joint text-image space of the querying text is computed using the text embedding model. Image queries can also be handled by using the visual embedding model instead of the text embedding model to generate the query representation. Furthermore, we can generate complex queries combining different query representations applying algebra in the joint text-image space. To retrieve the most semantically similar image I_R to a query x_q , we compute the cosine similarity of its vectorial representation $\phi(x_q)$ with the visual embeddings of the test set images $\psi(I_T)$, and retrieve the nearest image in the joint text-

¹Semantic Multimodal Image Retrieval Demo: <https://gombru.github.io/MMSemanticRetrievalDemo/>

image space:

$$\arg \min_{I_T \in \text{Test}} \frac{\langle \phi(x_q), \psi(I_T) \rangle}{\|\phi(x_q)\| \cdot \|\psi(I_T)\|} \quad (4.1)$$

State of the art text embedding methods trained on large text corpus are very good generating representations of text in a vector space where semantically similar concepts fall close to each other. The proposed pipeline leverages the semantic structure of these text embedding spaces training a visual embedding model that generates vectorial representations of images in the same space, mapping semantically similar images close to each other, and also close to texts correlated to the image content.

A CNN is trained to regress text embeddings from the correlated images minimizing a sigmoid cross-entropy loss. This loss is used to minimize distances between the text and image embeddings. Let $\{(I_n, x_n)\}_{n=1:N}$ be a batch of image-text pairs. If $\sigma(\cdot)$ is the component-wise sigmoid function, we denote $p_n = \sigma(\phi(x_n))$ and $\hat{p}_n = \sigma(\psi(I_n))$. Note $p_n, \hat{p}_n \in \mathbb{R}^D$ where D is the dimensionality of the joint embedding space. Let the loss be:

$$L = -\frac{1}{ND} \sum_{n=1}^N \sum_{d=1}^D [p_{n_d} \log \hat{p}_{n_d} + (1 - p_{n_d}) \log(1 - \hat{p}_{n_d})] \quad (4.2)$$

The GoogleNet architecture [85] is used, customizing the last layer to regress a vector of the same dimensionality as the text embedding. Cross Entropy Loss is not usually used for regression problems, where Mean Square Error loss is often used. We chose Cross Entropy Loss empirically, since it was the one providing an stable training and better performance. Although Cross Entropy Loss tends to be considered a loss for classification, it is also suitable for regression problems: despite this loss will not be zero when the regression solution matches the groundtruth, it will always be minimum compared to other solutions.

4.3.2 Datasets

We present the datasets used in this work and show some examples of their images and their associated text.

InstaCities1M

A dataset formed by Instagram images associated with one of the 10 most populated English speaking cities all over the world (in the images captions one of the names of these cities appears). It contains 100K images for each city, which makes a total of 1M

images, split in 800K training images, 50K validation images and 150K test images. The interest of this dataset is that is formed by recent Social Media data. The text associated with the images is the description and the hashtags written by the photo up-loaders, so it is the kind of free available data that would be very interesting to be able to learn from. The *InstaCities1M* dataset is available on ².

WebVision

The Webvision dataset [51] contains more than 2.4 million images crawled from the Flickr Website and Google Images search. The same 1,000 concepts as the ILSVRC 2012 dataset [37] are used for querying images. The textual information accompanying those images (caption, user tags and description) is provided. The validation set, which is used as test in this work, contains 50K images.

4.3.3 Results

To evaluate the learnt joint embeddings, we define a set of textual queries and check visually if the TOP-5 retrieved images contain the querying concept. We define 24 different queries. Half of them are single word queries and the other half two word queries. They have been selected to cover a wide area of semantic concepts that are usually present in Web and Social Media data. Both simple and complex queries are divided in four different categories: Urban, weather, food and people.

Tables 4.1 and 4.2 show the mean Precision at 5 for *InstaCities1M* and WebVision datasets and transfer learning between those datasets. To compute transfer learning results, we train the model with one dataset and test with the other. Table 4.3 shows the mean precision at 5 for *InstaCities1M* with introduced additional noise and of a model trained with Mean Square Error loss. The noise is introduced by changing the indicated % of captions to random captions from the training set. Figures 4.2, 4.3, 4.6, 4.4, 4.5 show the first retrieved images for some complex textual queries. Figure 4.6 also shows results for non-object queries, proving that our pipeline works beyond traditional instance-level retrieval. Figures 4.7 and 4.8 show that retrieval also works with multimodal queries combining an image and text.

For complex queries, where we demand two concepts to appear in the retrieved images, we obtain good results for those queries where the concepts tend to appear together. For instance, we generally retrieve correct images for “skyline + night” and for “bike + park”, but we do not retrieve images for “dog + kid”. When failing with this complex queries, usually images where only one of the two querying concepts appears are retrieved. Figure 4.9 shows that in some cases images corresponding to semantic concepts between the two querying concepts are retrieved. That proves that the common embedding space that has been learnt has a semantic structure. The performance is generally better in *InstaCities1M* than in WebVision. The reason is that

²InstaCities1M Dataset: <https://gomburu.github.io/2018/08/01/InstaCities1M/>

Table 4.1: Performance on InstaCities1M and WebVision. First column shows the mean P@5 for all the queries, second for the simple queries and third for complex queries.

Text embedding	InstaCities1M			WebVision		
	All	S	C	All	S	C
LDA 200	0.40	0.73	0.07	0.11	0.18	0.03
LDA 400	0.37	0.68	0.05	0.14	0.18	0.10
Word2Vec mean	0.46	0.71	0.20	0.37	0.57	0.17
Word2Vec tf-idf	0.41	0.63	0.18	0.41	0.58	0.23
Doc2Vec	0.22	0.25	0.18	0.22	0.17	0.27
GloVe	0.41	0.72	0.10	0.36	0.60	0.12
GloVe tf-idf	0.47	0.82	0.12	0.39	0.57	0.22
FastText tf-idf	0.31	0.50	0.12	0.37	0.60	0.13

Table 4.2: Performance on transfer learning. First column shows the mean P@5 for all the queries, second for the simple queries and third for complex queries.

Text embedding	Train: WebVision Test: InstaCities			Train: InstaCities Test: WebVision		
	All	S	C	All	S	C
LDA 200	0.14	0.25	0.03	0.33	0.55	0.12
LDA 400	0.17	0.25	0.08	0.24	0.39	0.10
Word2Vec mean	0.41	0.63	0.18	0.33	0.52	0.15
Word2Vec tf-idf	0.42	0.57	0.27	0.32	0.50	0.13
Doc2Vec	0.27	0.40	0.15	0.24	0.33	0.15
GloVe	0.36	0.58	0.15	0.29	0.53	0.05
GloVe tf-idf	0.39	0.57	0.22	0.51	0.75	0.27
FastText tf-idf	0.39	0.57	0.22	0.18	0.33	0.03

Table 4.3: Performance on InstaCities1M using GloVe tf-idf introducing noise by changing the indicated % of captions by random captions from the training set.

Experiment	InstaCities1M		
	All	S	C
Without introduced noise	0.47	0.82	0.12
10% introduced noise	0.25	0.43	0.07
20% introduced noise	0.18	0.32	0.05
30% introduced noise	0.15	0.25	0.05

the queries are closer to the kind of images people tend to post in Instagram than to the ImageNet classes. However, the results on transfer learning show that WebVision is a better dataset to train than *InstaCities1M*. That’s because WebVision has more images than *InstaCities1M* (2.4M training images vs 800k training images) and shows that the learned models are robust, general and scalable: Having more data, even if it’s not specifically related with the target task, allows learning embedding models that perform better in that task. Results show that all the tested text embeddings methods work quite well for simple queries. Though, LDA fails when is trained in WebVision. That is because LDA learns latent topics with semantic sense from the training data. Every WebVision image is associated to one of the 1,000 ImageNet classes, which influences a lot the topics learning. As a result, the embedding fails when the queries are not related to those classes.

The overall conclusion of the performance comparison between text embeddings in this experiment is that word level text embeddings such as Word2Vec and GloVe perform better than document level text embeddings (LDA, Doc2Vec) and character ngrams level text embeddings (FastText). The reason is that captions associated to images in Social Media tend to be quite concise, so averaging the word-level embeddings of a caption still gives us an informative representation that allows us to take profit of the rich semantic space learnt by this kind of embeddings. The fact that this semantic space is quite sparse allows us to perform arithmetic between embeddings in it, and also to be able to learn from those representations averaged over caption’s words.

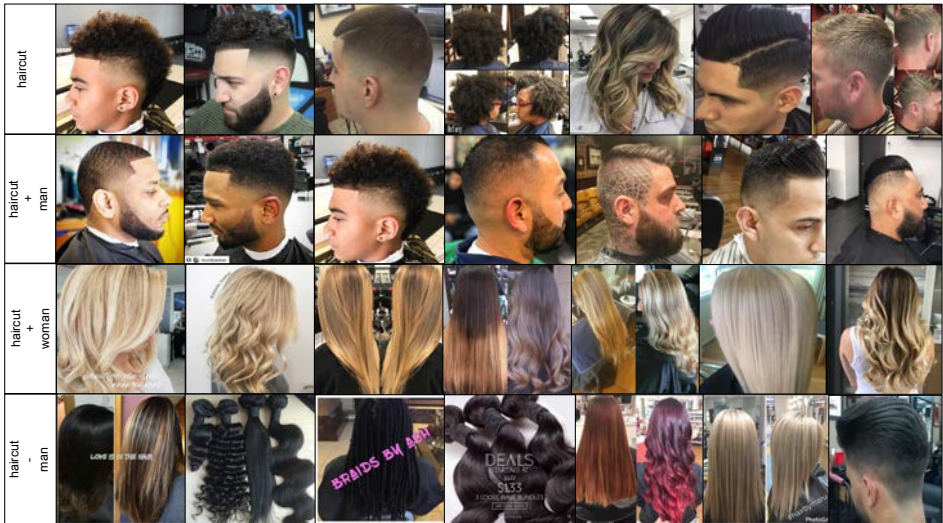


Figure 4.2: Top-ranked results of combined text queries related with haircuts by our semantic image retrieval model. The learnt joint image-text embedding permits to learn a rich semantic manifold even for previously unseen concepts even though they might not be explicitly present in the training set.

Error Analysis

Remarkable sources of errors are listed and explained below:

- Visual Features Confusion:** Errors due to the confusion between visually similar objects. For instance retrieving images of a quiche when querying “pizza”. Those errors could be avoided using more data and a higher dimensional representations, since the problem is the lack of training data to learn visual features that generalize to unseen samples.
- Errors from the Dataset Statistics:** An important source of errors is due to dataset statistics. As an example, the WebVision dataset contains a class which is “snow leopard” and it has many images of that concept. The word “snow” appears frequently in the images correlated descriptions, so the net learns to embed together the word “snow” and the visual features of a “snow leopard”. There are many more images of “snow leopard” than of “snow”, therefore, when we query “snow” we get snow leopard images. Figure 4.10 shows this error and how we can use complex multimodal queries to bias the results.
- Words with Different Meanings or Uses:** Words with different meanings or words that people use in different scenarios introduce unexpected behaviors. For instance when we query "woman + bag" in the *InstaCities1M* dataset we usually

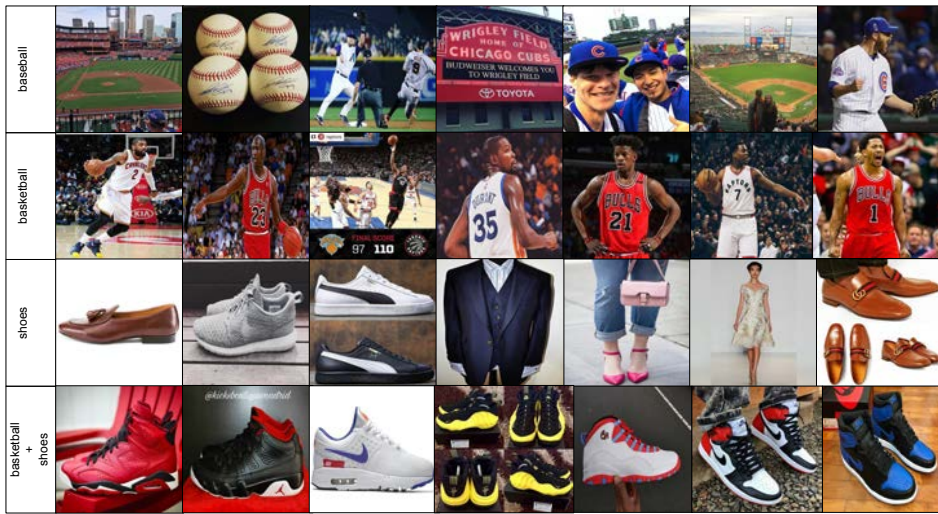


Figure 4.3: First retrieved images for complex queries related with basketball with Word2Vec on InstaCites1M.

retrieve images of pink bags. The reason is that people tend to write "woman" in an image caption when pink stuff appears. Those are considered errors in our evaluation, but inferring which images people relate with certain words in Social Media can be a very interesting research.

CNN Activation Maps Visualization

We have proved that, using only Social Media data, state of the art CNNs can be trained in a self-supervised way to learn powerful visual features, capable to discriminate among a huge variety of scenes: from objects to outdoor scenes, abstract concepts or specific buildings. In this experiment we visualize the images from the *InstaCites1M* retrieval set that generated the highest activations in some CNN units, using the GoogleNet trained from scratch with InstaCites1M and GloVe tf-idf text embedding as self-supervision. We also show the regions of the images that activated most the selected units. Figure 4.11 shows the results of a selection of neurons in the *pool5* layer of our model. We can notice that network units are selective to specific buildings, such as Golden Gate Bridge, objects such as guitars, drums or lights to identify concert scenes, or even basketball t-shirts.

Visualizing the Learned Semantic Space with t-SNE

We use the t-SNE dimensionality reduction method to reduce the dimensionality of the joint embedding space to 2 dimensions and we show images in that space to visualize

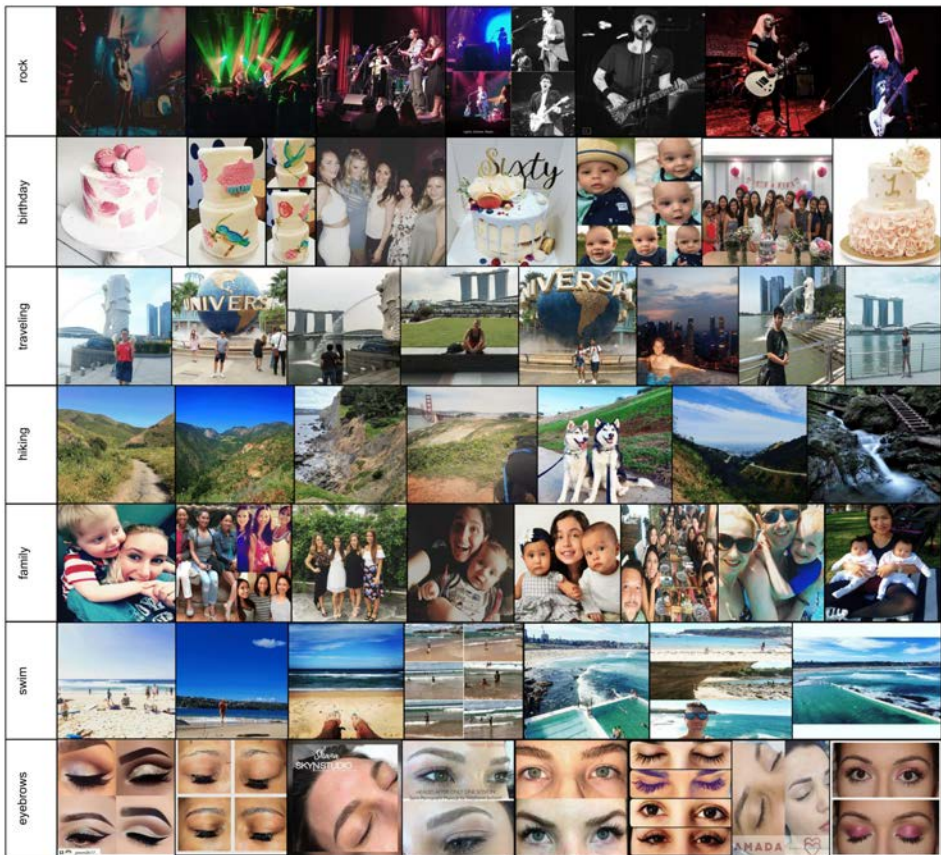


Figure 4.4: First retrieved images for textual queries with Word2Vec on InstaCites1M.

its semantic structure. t-SNE is a non-linear dimensionality reduction method, which we use on our 400 dimensional embeddings to produce 2 dimensional embeddings. For each one of the given 400 dimensional visual or textual embeddings, t-SNE computes a 2 dimensional embedding arranging elements that have similar representations nearby, providing a way to visualize the learnt joint image-text space and analyze qualitatively its semantic structure. As we have learnt a joint image and text embedding space, we can apply t-SNE to both modalities of embeddings at once. We apply t-SNE to a set formed by the visual embeddings of the images in test set of *InstaCites1M* and the text embeddings of the selected querying terms. In this experiment, we use the Word2Vec model trained on *InstaCites1M* dataset. For text embeddings, we use an image containing its words as their representations in the canvas. To get an interpretable visualization avoiding images overlaps, if two images share any pixel in the output Figure we omit one of them (prioritizing word images). Therefore, images surrounding word images are not necessary top retrieval results for that word, but they

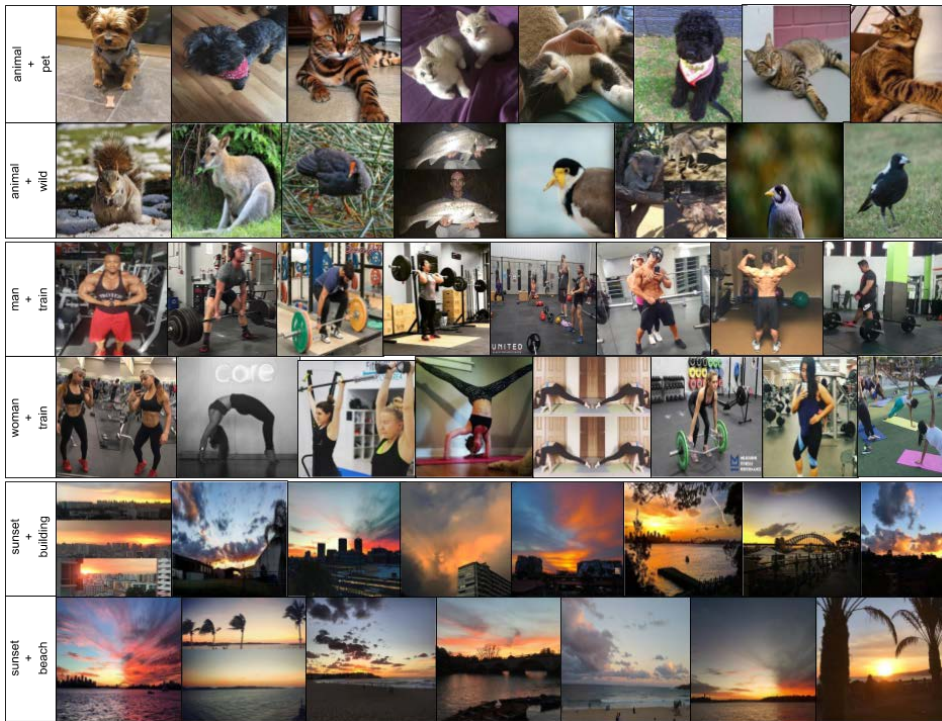


Figure 4.5: First retrieved images for complex textual queries with Word2Vec on InstaCites1M.

are the nearest images of the ones being represented in the figure.

The joint embeddings 2 dimensional visualization in 4.12 shows the semantic structure of the learnt space. It shows semantic clusters that the joint embedding has learnt in a self-supervised way from the data distribution, that correspond to different kind of images people tend to post on Instagram. For instance, the Figure shows a cluster for food images, a cluster for sport images, a cluster for sunrise images, or a cluster for animal images. It also shows that images of people are very numerous, and that the joint embedding groups them correctly. It can also be appreciated how images we might consider noise, such as images with logos or text, are clustered together. The majority of those images are far from the semantic clusters, isolated and near the Figure edges. That is because the joint embedding hasn't been able to find semantic relations between these images and the rest, so it assigns to them embeddings that have not relation with the others. When computing t-SNE, as the objective is to place similar images nearby, this images without semantic relations are set far from the others. Therefore, we can conclude that the pipeline is quite robust to Social Media noise. More t-SNE visualizations of the learnt joint embeddings are available in ³.

³t-SNE visualizations: https://gombru.github.io/2018/08/01/learning_from_web_data

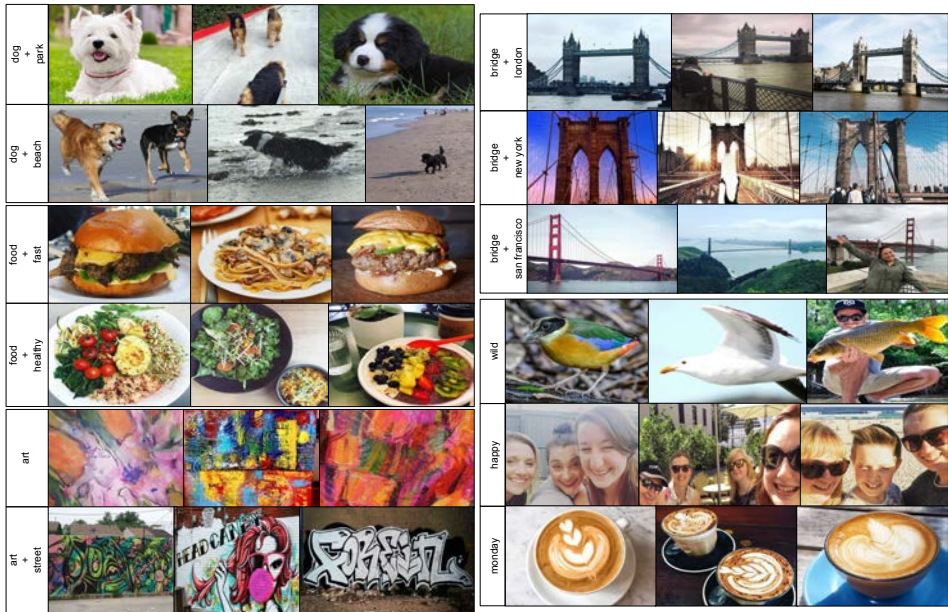


Figure 4.6: First retrieved images for complex queries (left), city related complex queries (top-right) and non-object queries (bottom-right) with Word2Vec on InstaCites1M.



Figure 4.7: First retrieved images for multimodal queries (concepts are added or removed to bias the results) with Word2Vec on WebVision.

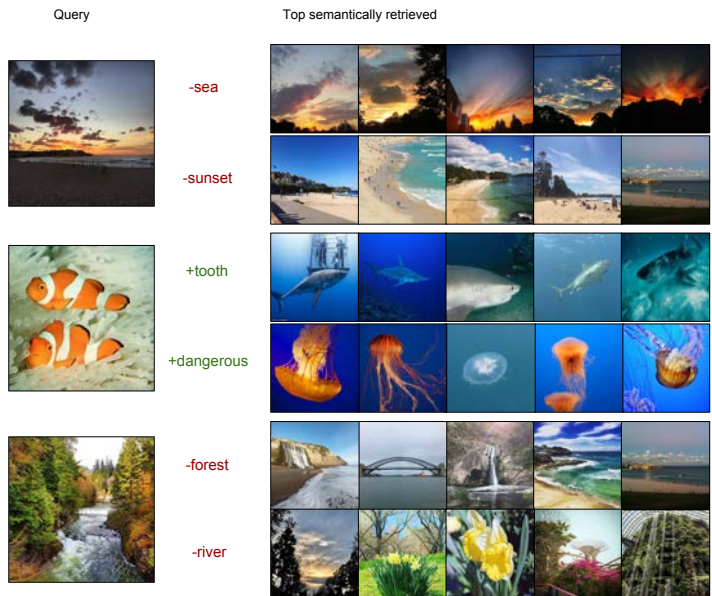


Figure 4.8: First retrieved images for multimodal complex queries with Word2Vec on WebVision.

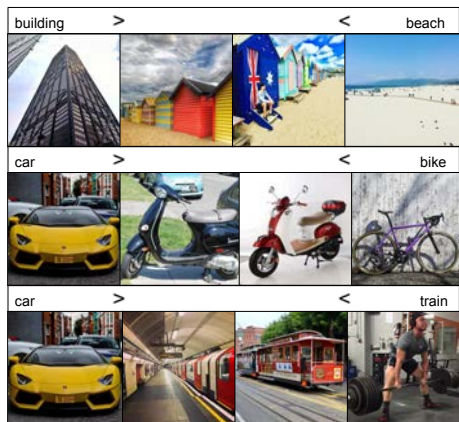


Figure 4.9: First retrieved images for simple (left and right columns) and complex weighted queries with Word2Vec on InstaCites1M.

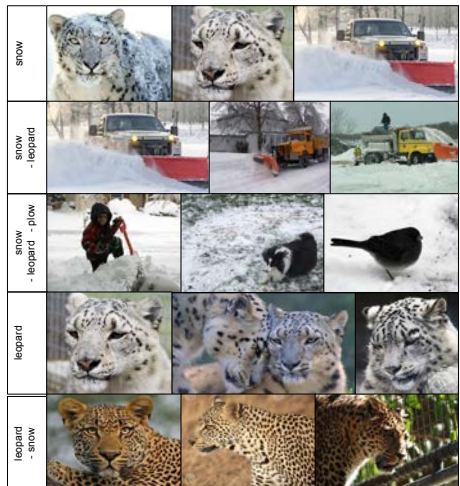


Figure 4.10: First retrieved images for text queries using Word2Vec on WebVision. Concepts are removed to bias the results.

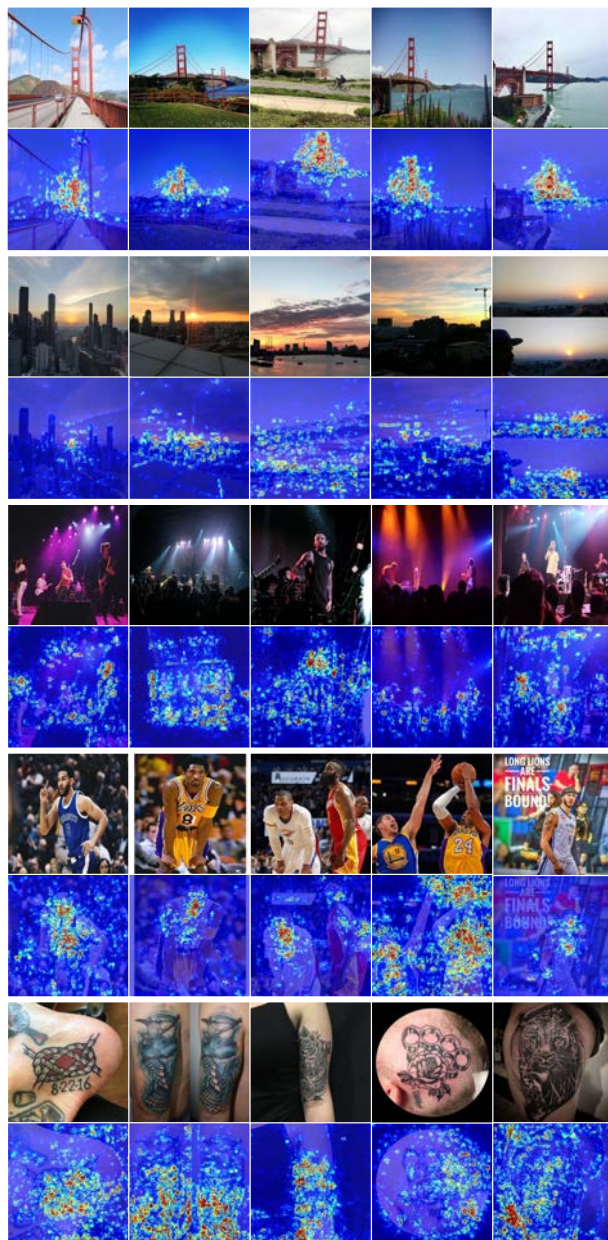


Figure 4.11: Top-5 activations for five units in pool5 layer of GoogleNet model trained from scratch with InstaCities1M using GloVe tf-idf as self-supervision and their activation maps.

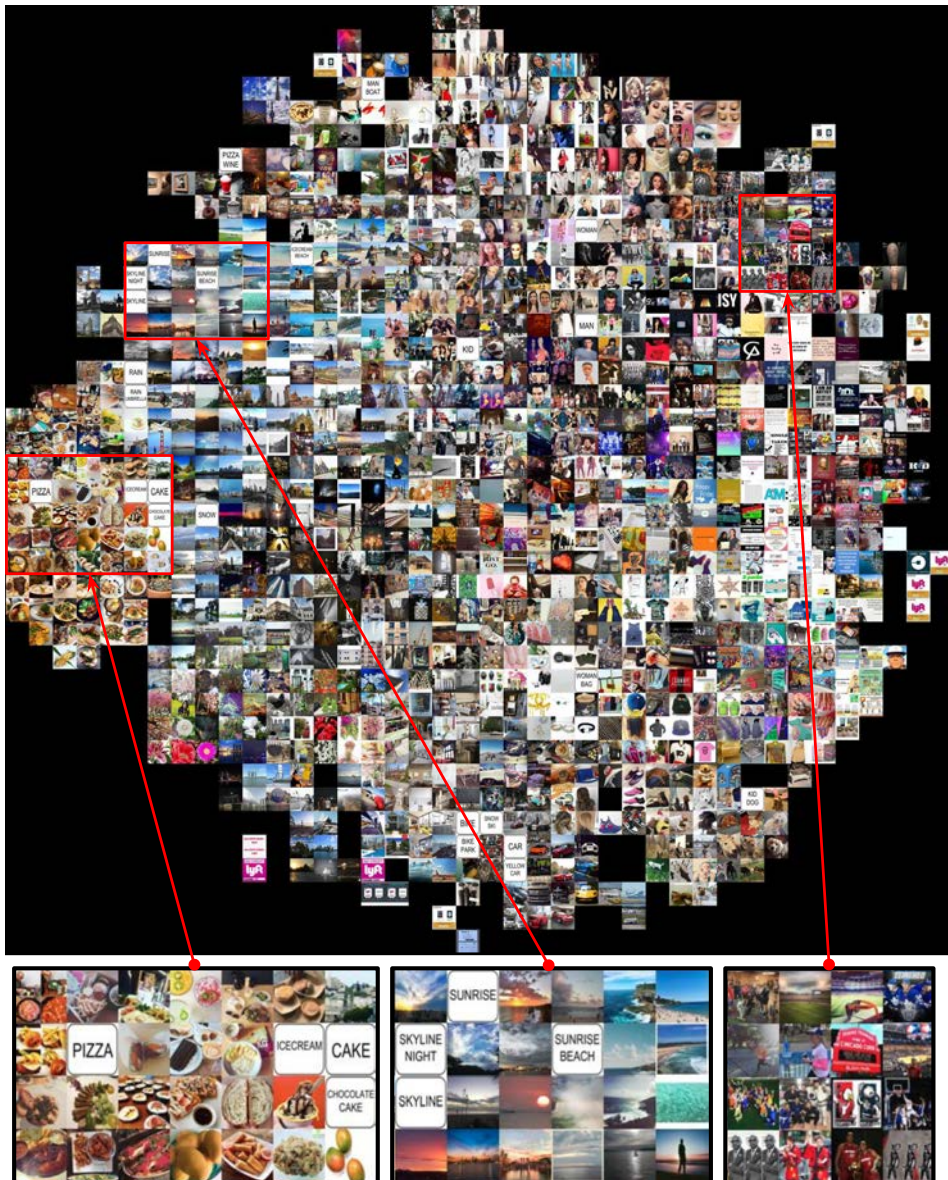


Figure 4.12: Visualization of the joint embedding with Word2Vec on InstaCities1M dataset.

4.3.4 Comparing the Image and Text Embeddings

We analyze the semantic quality of the learnt joint embedding spaces showing how the CNN has learnt to embed images in them.

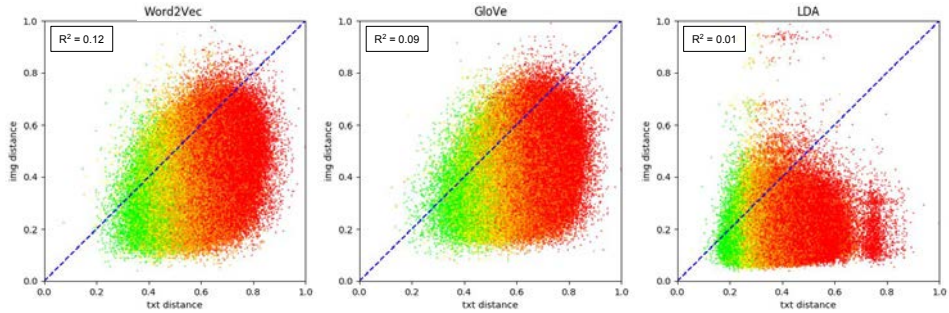


Figure 4.13: Text embeddings distance (X) vs the images embedding distance (Y) of different random image pairs for LDA, Word2Vec and GloVe embeddings trained with InstaCities1M. Distances have been normalized between [0,1]. Points are red if the pair does not share any tag, orange if it shares 1, light orange if it shares 2, yellow if it shares 3 and green if it shares more. R^2 is the coefficient of determination of images and texts distances.

Experiment Setup

To evaluate how the CNN has learnt to map images to the text embedding space and the semantic quality of that space, we perform the following experiment: We build random image pairs and we compute the cosine similarity between both their image and their text embeddings. In Figure 4.13 we plot the images embeddings distance vs the text embedding distance of 20,000 random image pairs. If the CNN has learnt correctly to map images to the text embedding space, the distances between the embeddings of the images and the texts of a pair should be similar, and points in the plot should fall around the identity line $y = x$. Also, if the learnt space has a semantic structure, both the distance between images embeddings and the distance between texts embeddings should be smaller for those pairs sharing more tags: The plot points' color reflects the number of common tags of the image pair, so pairs sharing more tags should be closer to the axis origin.

As an example, take a dog image with the tag "dog", a cat image with the tag "cat" and one of a scarab with the tag "scarab". If the text embedding has been learnt correctly, the distance between the projections of dog and scarab tags in the text embedding space should be bigger than the one between dog and cat tags, but smaller than the one between other pairs not related at all. If the CNN has correctly learnt to embed the images of those animals in the text embedding space, the distance between the dog and the cat image embeddings should be similar than the one between their tags

embeddings (and the same for any pair). So the point given by the pair should fall in the identity line. Furthermore, that distance should be nearer to the coordinates origin than the point given by the dog and scarab pair, which should also fall in the identity line and nearer to the coordinates origin than another pair that has no relation at all.

Results and Conclusions

The plots in Figure 4.13 for both the Word2Vec and the GloVe embeddings show a similar shape. The resulting blob is elongated along the $y = x$ direction, which proves that both image and text embeddings tend to provide similar distances for an image pair. The blob is thinner and closer to the identity line when the distances are smaller (so when the image pairs are related), which means that the embeddings can provide a valid distance for semantic concepts that are close enough (dog, cat), but fails inferring distances between weak related concepts (car, skateboard). The colors of the points in the plots show that the space learnt has a semantic structure. Points corresponding to pairs having more tags in common are closer to the coordinates origin and have smaller distances between the image and the text embedding. From the colors it can also be deduced that the CNN is good inferring distances for related images pairs: there are just a few images having more than 3 tags in common with image embedding distance bigger than 0.6, while there are many images with bigger distances that do not have tags in common. However, the visual embedding sometimes fails and infers small distances for image pairs that are not related, as those images pairs having no tags in common and an image embedding distance below 0.2.

The plot of the LDA embedding shows that the learnt joint embedding is not so good in terms of the CNN images mapping to the text embedding space nor in terms of the space semantic structure. The blob does not follow the identity line direction that much which means that the CNN and the LDA are not inferring similar distances for images and texts of pairs. The points colors show that the CNN is inferring smaller distances for more similar image pairs only when the pairs are very related.

The coefficient of determination R^2 shown at each graph measures the proportion of the variance in a dependent variable that is predicted by linear regression and a predictor variable. In this case, it can be interpreted as a measure of how much image distances can be predicted from text distances and, therefore, of how well the visual embedding has learnt to map images to the joint image-text space. It ratifies our plots' visual inspection proving that visual embeddings trained with Word2Vec and GloVe representations have learnt a much more accurate mapping than LDA, and shows that Word2Vec is better in terms of that mapping.

4.4 Learning from Barcelona Instagram Data

4.4.1 Text Supervision in Social Media for Tourism Analysis

Instagram data can be exploited to obtain information about a city that has interesting social and commercial applications. Specifically, in this section we analyze images and captions related to Barcelona. Barcelona is a very touristic city which revives around 10 million tourists every year. That causes conflicts between tourists and locals and between the tourism industry and other local organizations, conflicts that are highly concentrated on neighborhoods with requested tourist attractions. Measuring the tourism overcrowding per neighborhood is not easy, since some areas receive high touristic interest but they don't have hotels or tourism installations. This work proposes a method to do that by exploiting Instagram data.

We perform a multimodal, language separate analysis using the text of the captions and its associated images, designing a pipeline that learns relations between words, images and neighborhoods in a self-supervised way. We focus on a per-neighborhood analysis, and analyze how the differences of tourism activity between Barcelona districts and neighborhoods are reflected on Instagram. This work has been published in [24]. The proposed method works as follows:

1. We split the data depending on whether it contains captions in a local language, Spanish and Catalan, or English, which we consider to be locals vs tourists publications.
2. We train a semantic word embedding model, Word2Vec [62], for each language and show the words that locals and tourists associate with different neighborhood names.
3. Using the semantic word embeddings as a supervisory signal, we train a CNN that learns relations between images and neighborhoods.
4. Using the trained models in a retrieval approach with unseen data we show, for each language, the most related images to different neighborhoods. Results show interesting differences between the visual elements that locals and tourists associate to each neighborhood.

4.4.2 Dataset

To perform the presented analysis we gathered a dataset of Instagram images related to Barcelona uploaded between September and December of 2017. That means images with a caption where the word "Barcelona" appears. We collected around 1.3 million images. The resulting dataset, *InstaBarcelona*, contains 597,766 image-captions pairs and is made publicly available for download. From those pairs 331,037 are En-

glish publications, 171,825 Spanish publications and 94,311 Catalan publications. The dataset is available on ⁴.

4.4.3 Textual Analysis

We train a Word2Vec model for each one of the analyzed languages: English, Spanish and Catalan. The objective is to learn the different contexts where the authors use words depending on their language. Using the Word2Vec learned models for each language, we can infer the words that users writing in English, Spanish or Catalan (tourist or locals) relate with each one of the Barcelona's neighborhoods.

Next, we show the closest words in the Word2Vec space to the four *Ciutat Vella* neighborhoods using the three Word2Vec models learned. Closest words of the **English** trained Word2Vec are shown in red, of the **Spanish** one in green, and of the **Catalan** one in blue. Spelling variants and synonyms have been removed from the results.

Barceloneta

hotelw, seaside, beachlife, beachview, port, bcnbeach
ramblademar, torremapfre, hotelvela, paseomaritimo
portolimpic, hotelwela, vilaolimpica, torremapfre, bogatell

Born

barcelonasspots, gothicdistrict, oldtown, catedraldelmar
passeigdelborn, portalnou, callejuelas, rinconesmagicos
mercatdelborn, ccm, banyorientals, cafedelborn, laribera

Gòtic

cathedral, history, gargoyles, churches, architecture
edadmedia, laribera, carrerdelbisbe, mercadodelborn
carrerdelbisbe, plaçadelrei, catedraldelmar, carrercomtal

Raval

cccb, macba, zeligbar, poblesec, elborn, grafity, gotico
rambladelraval, ravalcultural, fueradrogas, narcopisos
ravalcultural, somdebarri, ravalescultura, botigadecomics

This examples show the interests of the different language speakers in the query neighborhoods. Words related to *Barceloneta* and *El Gòtic* neighborhoods in the three languages are mostly tourist attractions. However, we can appreciate differences between languages. For instance, when mentioning *El Gòtic*, Spanish and Catalan speakers use along names of its streets and squares, while English speakers use more general words. Tourist publications mentioning *El Born* relate this district to Barcelona's old town, while locals publications mention its promenade, its market or its culture center (CCM). When mentioning *El Raval*, tourists publications mention its museums and other nearby districts. On the contrary, locals publications talk about its cultural activity, its promenade or its drug presence problem.

The trained Word2Vec models provide information that can be used beyond a district analysis. They can infer the words that Instagram users relate to Barcelona and any other word in the training vocabulary. For the following queries, the translation of the English query word to the corresponding language has been used, and the translation of the local languages results to English are shown.

⁴InstaBarcelona Dataset: <https://gomburu.github.io/2018/08/02/InstaBarcelona/>

<p><i>Beach</i></p> <p>summer, seaside, sand, sunset, sunny, whotel, seaview seaview, passeigmaritim, mediterráneo, novaicaria bogatell, novaicaria, mediterrani, lamarbella, voley</p>	<p><i>Tourism</i></p> <p>landscape, architecture, bluesky, spain, aroundtheworld catalanfood, adventure, crossing, eatanddrink, fair route, beautifulplaces, mycity, heritage, walking</p>
<p><i>Food</i></p> <p>tapas, fastfood, breakfast, sangria, seafood, blackrice sushi, ham, soup, chicken, hamburguer, fruit cannelloni, omelette, bread, soup, fruit, fries</p>	<p><i>Neighborhood</i></p> <p>quaint, hidden, restaurants, lively, locals, gracia, corners gotico, tourists, citizens, park, streets, shops, people ribera, citizen, ravalnotforsale, citizenfight, street, walking</p>

This experiments also show clear differences between the models trained with the different languages. For instance, when mentioning *Food* English speakers write along Spanish most characteristic dishes, while locals write about more daily meals. When mentioning *Neighborhood*, tourists talk about its restaurants or appearance, while locals talk more about its people.

4.4.4 Visual Analysis

Word2Vec allows us to find the words that authors relate neighborhoods when using different languages. That is possible because Word2Vec learns word embeddings to a vectorial space where semantic similar words (words appearing in similar contexts), are mapped nearby. *Img2NeighCtx* (Image to Neighborhood Context) is a Convolutional Neural Network that, learning from images and associated captions, allows us to find the images that authors relate to the different neighborhoods when using different languages.

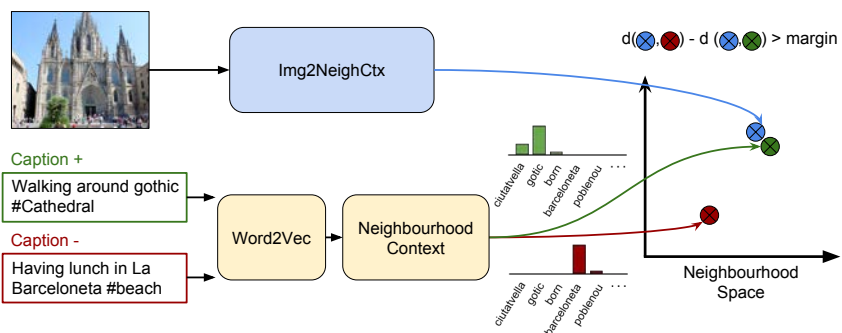


Figure 4.14: Training procedure of *Img2NeighCtx*. The CNN is trained to maximize the difference of distances between the image and the positive caption and the image and the negative caption in the *Neighborhood Space* space until a certain margin.

Img2NeighCtx

Word2Vec allows us to compute a similarity between two words. To compute a vector encoding the similarities of a caption with each of the 82 Barcelona’s districts and neighborhoods, we sum up the cosine similarities of all the caption words with each neighborhood name in the Word2Vec space and L2 normalize the vector. We call the resulting vector *Neighborhood Context (NC)*. Let W be the Word2Vec representations of all the words in a caption c and $N = \{n_j\}_{j=1:J}$ be all the neighborhoods names Word2Vec representations ($J = 82$). The neighborhood context of each word w in the caption is represented by

$$NC(w) = \left(\frac{\langle w, n_1 \rangle}{\|w\| \cdot \|n_1\|}, \frac{\langle w, n_2 \rangle}{\|w\| \cdot \|n_2\|}, \dots, \frac{\langle w, n_J \rangle}{\|w\| \cdot \|n_J\|} \right) \quad (4.3)$$

We eventually compute the *Neighborhood Context* of the caption c as:

$$NC(c) = \sum_{w \in W} NC(w) \quad (4.4)$$

which is L2 normalized.

Img2NeighCtx is a GoogleNet based CNN that learns to infer NC from images. The last classification layer is replaced by a fully connected layer with 82 outputs, which is the dimensionality of the *Neighborhood Space*, and uses a ranking loss to learn to embed images with similar captions *Neighborhood Contexts* nearby. Img2NeighCtx receives three inputs: the image (i), its caption embedding (NC^+), and a negative caption embedding (NC^-). The negative caption embedding is selected randomly from the 50% most distant batch caption embeddings. We define the loss by:

$$L(i, NC^+, NC^-) = \frac{1}{2} \max(0, m - \Phi_i^T NC^+ + \Phi_i^T NC^-) \quad (4.5)$$

where m is the margin and Φ is the function that embeds the image into the *Neighborhood Space*. Img2NeighCtx is trained to minimize this loss, which maximizes the difference between the distances of the image with the positive and negative captions upon a certain margin. The training pipeline of Img2NeighCtx is shown in Figure 4.14.

Images Associated to Districts

Once Img2NeighCtx has been trained to embed images in the *Neighborhood Space*, it can be used in a straightforward manner in an image by neighbourhood retrieval task. The CNN has learned from the images and the associated captions to extract visual features useful to relate images to the different neighborhoods. Using as a query

a neighborhood represented as a one hot vector in the *Neighborhood Space*, we can infer the kind of images that Instagram users writing in English, Spanish or Catalan relate to that neighborhood. To do that we retrieve the nearest images in the *Neighborhood Space*. Figure 4.15 show the first retrieved images for some of the neighborhoods. Images in the top row (red) correspond to the English trained model, in the second (green) to the Spanish one, and in the third (blue) to the Catalan one. When talking about *El Born (Sant Pere)* (Fig. 4.15), tourist tend to post photos of bikes, since there are many tourist oriented stores offering bike renting services there, while locals tend to post photos of its bars and streets. When posting about *El Poblesec* 4.15, tourist tend to post photos of the food they have in its popularity increasing restaurants, while locals tend to post photos of themselves, its bars or its art galleries. When posting about *El Poblenou* 4.15, the kind of images people post using the three languages are similar and related to design and art. This is because *El Poblenou* neighbourhood has been promoted as a technology and design hub in Barcelona, following the 22@ plan. This plan has attracted many foreign workers to live in the area. Therefore, and in contrast to other neighborhoods, the majority of English publications related to *El Poblenou* are not from tourists but from people that have settled here, and appear to have the same interests in *El Poblenou* as the Catalan and Spanish speakers.

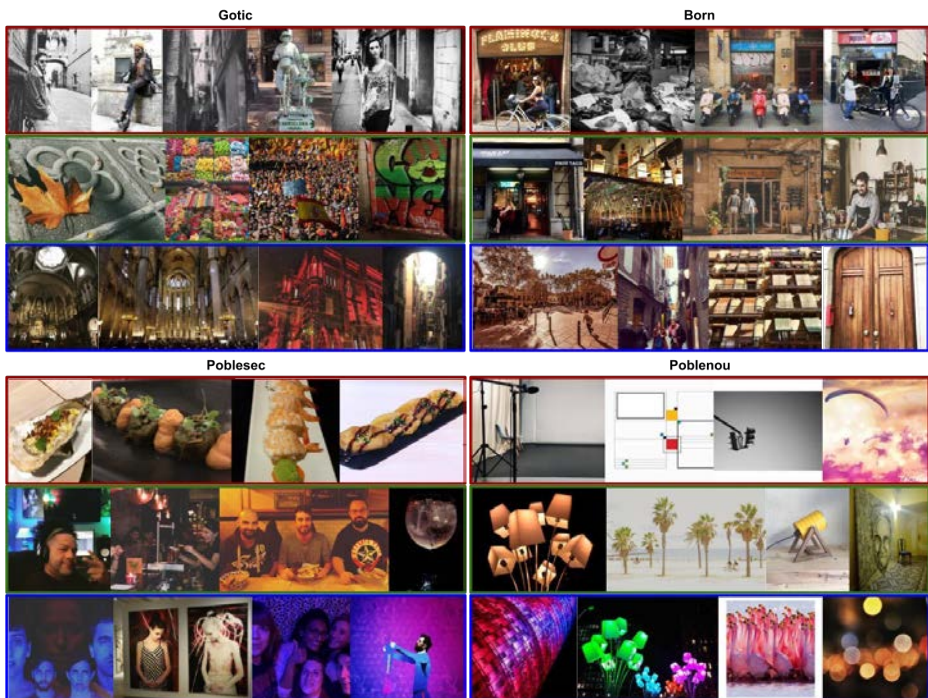


Figure 4.15: *Img2NeighCtx* image by neighborhood retrieval results for different neighborhoods in each of the languages.

Beyond Districts: Img2Word2Vec

Img2NeighCtx is very useful to retrieve images associated to each neighborhood in each one of the languages. In a similar way we trained Img2NeighCtx to predict *Neighborhood Contexts* from images, we can train a net to directly embed images in the Word2Vec space. We call that net Img2Word2Vec.

First, the embeddings of all the captions in the Word2Vec space are computed as the mean of its word embeddings and L2 normalized. Img2Word2Vec has the same structure as Img2NeighCtx, but the last fully connected layer has 300 outputs, which is the dimensionality of the Word2Vec space. It uses a ranking loss to learn to embed semantically similar images nearby. Img2Word2Vec receives 3 inputs: the image, its caption Word2Vec embedding, and a negative caption Word2Vec embedding. The negative caption embedding is selected randomly from the other batch captions. The training pipeline is similar to the Img2NeighCtx one (Figure 4.14) but leaving out the Neighborhood Context computation and applying the ranking loss directly to captions Word2Vec embeddings.

The trained Img2Word2Vec models can be used to relate text and images beyond districts and neighborhoods names, retrieving images related to any text concept present in the vocabulary. Figure 4.16 shows retrieval results for different query words. When using the word *food*, tourist tend to post photos of themselves in front of "healthy" and well presented dishes or seafood. As a contrast, locals tend to post photos where only the food appears, and it tends to be international and more diverse. For *friends* tourist tend to post photos of a group of friends in the beach, while locals tend to appear around a table, though they are more diverse. Associated with the word *views*, tourists post photos of Barcelona's views taken from popular places (Montjuic and Park Güell). As a contrast, locals photos are more diverse and include photos taken from houses and of other Barcelona areas, such as the port. When using the word *market*, tourist photos are mainly from Mercat de la Boqueria, an old market in Barcelona's old town that has turned into a very touristic place. Meanwhile, locals photos are more divers and include markets where people do their daily shopping.

4.4.5 Applications to Tourism Industry

This self-supervised setup to learn from Social Media images with associated text can have many applications: We can learn which visual features people associate with any term. This work, where I learn which visual features people in Barcelona associate with words, was interesting for the tourist industry and I presented it in ForumTurisTIC⁵, explaining the possibilities of this method and showing examples of what images people associate with words related with restaurants, hotels, or nightlife. The industry showed interest in this research and further developments.

⁵Forum TurisTIC presentation: https://gombru.github.io/2018/02/11/forumTurisTIC_presentation/



Figure 4.16: Img2Word2Vec image by text retrieval results for different queries in each of the languages.

4.4.6 #Barcelona Deep Dream

The Deep Dream algorithm by Google ⁶ magnifies the patterns that a given layer recognizes in an input image, and then generates a new image where those patterns are amplified. We can select any layer and ask the CNN to enhance what it sees. Lower layers will produce lower level patterns, since they are sensible to basic features, but higher layers will produce more sophisticated patterns or whole objects. Applying this algorithm iteratively (to its own output) will result in images where the detected patterns have been more amplified.

If we use Deep Dream to visualize what the Img2Word2Vec net has learnt from *InstaBarcelona* we get city recognizable patterns, as the one shown in Figure 4.17. We can clearly see the shapes of Sagrada Familia, and the lights and color of the Montjuic Fountain. Those are very popular tourist attractions in Barcelona, so there are many images of them in Instagram and the net learns consistent patterns for them. The image also clearly shows a dog face, which is probably because of the Imagenet initialization, and the fact that dog pictures in Instagram are also common. Figure 4.18 shows the results of amplifying another layer of Img2Word2Vec with Deep Dream. In this case

⁶<https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

the patterns shown are clearly related to the Football Club Barcelona, another tourist attraction from Barcelona. The image clearly shows the colors of the club, the field, and shapes of players. This visualisations show again how powerful can be self-supervised learning from Social Media images with associated text to learn visual features. Additional #Barcelona Deep Dream results are available in ⁷.



Figure 4.17: Deep Dream applied to Img2Word2Vec.



Figure 4.18: Deep Dream applied to another layer of Img2Word2Vec.

⁷#Barcelona Deep Dream Results: https://gombru.github.io/2018/10/10/barcelona_deepdream

4.5 Conclusions

In this chapter we have researched how Social Media data can be exploited to learn a joint visual and textual embedding by learning images embeddings on a pre-learned text representation space. We have shown how that methodology exploits the semantic structure of the text representations to learn semantic visual embeddings, and allows to perform semantic image retrieval, going beyond instance-level retrieval. The retrieval setup can handle multiple concepts queries and also multimodal queries, composed by a visual query and a text modifier to bias the results.

Also, we have benchmarked state of the art text embeddings in the image retrieval by text task, concluding that GloVe and Word2Vec have a similar performance and are the best ones for this data, consisting on images with associated captions. Note that, related works were I collaborated [67] concluded that LDA works better when we want to learn from images with longer associated texts, such as Wikipedia articles.

Additionally we showed an application of self-supervised learning from Social Media data using images and associated text where we learned which visual features people associate with a given concept and city. Next, we proposed a method to learn which images people associate with the different neighbourhoods of a city and performed a language-separate analysis that showed how it can be useful to analyze the tourism behaviour of a city.

Chapter 5

Multimodal Learning with Images, Text and Geolocations

Research Question 5: *Which is the best method to learn from images with associated tags?*

Research Question 6: *Can location information be exploited to boost image tagging performance?*

Research Question 7: *Can we train a retrieval system to find images associated to a given tag and near to a given location?*

5.1 Introduction

Besides text and images, location is a data modality widely present in contemporary data collections. Many cameras and mobile phones with built-in GPS systems store the location information in the corresponding *Exif* metadata header when a picture is taken. Moreover, most of the web and social media platforms add this information to generated content or use it in their offered services. In this chapter we leverage this third data modality: using location information can be useful in an image tagging task since location-related tagging can provide better contextual results. For instance, an image of a skier in France could have the tags “*ski, alps, les2alpes, neige*”, while an image of a skier in Canada could have the tags “*ski, montremblant, canada, snow*”. More importantly, location can also be very useful in an image retrieval setup where we want to find images related to a word in a specific location: the retrieved images related to the query tag *temple* in Italy should be different from those in China.

We propose a new architecture for modeling the joint distribution of images, hashtags, and geographic locations and demonstrate its ability to retrieve relevant images given a query composed by a hashtag and a location. In this task, which we call loca-

tion sensitive tag-based image retrieval, a retrieved image is considered relevant if the query hashtag is within its ground-truth hashtags and the distance between its location and the query location is smaller than a given threshold.

5.2 Background on Location Sensitive Image Retrieval and Tagging

A common approach to address both image by text retrieval and image tagging is to learn a joint embedding space for images and words [45, 95, 72, 22], as we have also seen in Chapter 4. In such a space, images are embedded near to the words with which they share semantics. Consequently, semantically *similar* images are also embedded together. Another approach to handle multiple modalities of data is by scoring tuples of multimodal samples aiming to get high scores on positive cases and low scores on negative ones [90, 97, 73, 35]. In this setup, instead of strict similarities between modalities, the model learns more relaxed compatibility scores between them. The work presented in this Chapter fits under this paradigm. Specifically, we train a model that produces scores for image-hashtag-coordinates triplets, and we use these scores in a ranking loss in order to learn parameters that discriminate between observed and unobserved triplets.

The task of location sensitive retrieval as defined before, has not yet been addressed. O’Hare *et al.* [66] presented the need of conditioning image retrieval to location information, and targeted it by using location to filter out distant photos and then performing a visual search for ranking. Similar location-based filtering strategies have been also used for landmark identification [10] and to speed-up loop closure in visual SLAM [47]. The obvious limitation of such systems compared to *LocSens* is that they require geolocation annotations in the entire retrieval set. Kennedy *et al.* [43, 42] and Rattenbury *et al.* [71] used location-based clustering to get the most representative tags and images for each cluster, and presented limited image retrieval results for a subset of tags associated to a given location (landmark tags). They did not learn, however, location-dependent visual representations for tags as we do here, and their system is limited to the use of landmark tags as queries. On the other hand, Zhang *et al.* [105] proposed a location-aware method for image tagging and tag-based retrieval that first identifies points of interest, clustering images by their locations, and then represents the image-tag relations in each of the clusters with an individual image-tag matrix [101]. Their study is limited to datasets on single city scale and small number of tags (1000). Their retrieval method is constrained to use location to improve results for tags with location semantics, and cannot retrieve location-dependent results (i.e. only the tag is used as query). Again, contrary to *LocSens*, this method requires geolocation annotations over the entire retrieval set. Other existing location-aware tagging methods [63, 54] have also addressed constrained or small scale setups (e.g. a fixed number of cities) and small-size tag vocabularies, while in this paper we target a worldwide scale unconstrained scenario.

5.3 Methodology

Given a large set of images, tags and geographical coordinates, our objective is to train a model to score triplets of image-hashtag-coordinates and rank them to perform two tasks: (1) image retrieval querying with a hashtag and a location, and (2) image tagging when both the image and the location are available. We address the problem in two stages: first, we train a location-agnostic CNN to learn image representations using tags as supervision. We propose different training methodologies and evaluate their performance on image tagging and retrieval. These serve as benchmark and provide compact image representations to be later used within the location sensitive models. Second, using the learnt image and hashtags best performing representations and the locations, we train multimodal models to score triplets of these three modalities. We finally evaluate them on image retrieval and tagging and analyze how these models benefit from the location information.

5.3.1 Learning with Tags Supervision

Three procedures for training location-agnostic visual recognition models using hashtag supervision are considered: (1) multi-label classification, (2) softmax multi-class classification, and (3) hashtag embedding regression. In the following, let \mathbb{H} be the set of H considered hashtags. \mathbf{I}_x will stand for a training image and $\mathbf{H}_x \subseteq \mathbb{H}$ for the set of its groundtruth hashtags. The image model $f(\cdot; \theta)$ used is a ResNet-50 [30] with parameters θ . The three approaches eventually produce a vector representation for an image \mathbf{I}_x , which we denote by \mathbf{r}_x . For a given hashtag $h^i \in \mathbb{H}$, its representation—denoted \mathbf{v}_i —is either learnt externally or jointly with those of the images.

Multi-Label Classification

We set the problem as a standard multi-label classification setup over H classes corresponding to the hashtags in the vocabulary \mathbb{H} . The last ResNet-50 layer is replaced by a linear layer with H outputs, and each one of the H binary classification problems is addressed with a cross-entropy loss with sigmoid activation, which is sometimes called Binary Cross-Entropy Loss. Let $\mathbf{y}_x = (y_x^1, \dots, y_x^H)$ be the multi-hot vector encoding the groundtruth hashtags of \mathbf{I}_x and $\mathbf{f}_x = \sigma(f(\mathbf{I}_x; \theta))$, where σ is the element-wise sigmoid function. The loss for image \mathbf{I}_x is written as:

$$L = -\frac{1}{H} \sum_{h=1}^H [y_x^h \log f_x^h + (1 - y_x^h) \log(1 - f_x^h)] \quad (5.1)$$

Multi-Class Classification

Despite being counter-intuitive, several prior studies [59, 90] demonstrate the effectiveness of formulating multi-label problems with large numbers of classes as multi-class problems. At training time a random target class from the groundtruth set \mathbf{H}_x is selected, and softmax activation with a cross-entropy loss is used. This setup is commonly known as softmax classification. Notice that during training we simulate that each image has only one positive class at each iteration, despite the ground-truth is multi-label.

Let $h_x^i \in \mathbf{H}_x$ be a randomly selected class (hashtag) for \mathbf{I}_x . Let also f_x^i be the coordinate of $\mathbf{f}_x = f(\mathbf{I}_x; \theta)$ corresponding to h_x^i . The loss for image \mathbf{I}_x is set to be:

$$L = -\log \left(\frac{e^{f_x^i}}{\sum_{j=1}^H e^{f_x^j}} \right) \quad (5.2)$$

In this setup we redefine ResNet-50 by adding a linear layer with D outputs just before the last classification layer with H outputs. This allows getting compact image D -dimensional representations \mathbf{r}_x as their activations in such layer. Since we are in a multi-class setup where the groundtruth is a one-hot vector, we are also implicitly learning hashtag embeddings: the weights of the last classification layer with input \mathbf{r}_x and output \mathbf{f}_x is an $H \times D$ matrix whose rows can be understood as D -dimensional representations of the hashtags in \mathbb{H} . Consequently, this approach learns at once D -dimensional embeddings for both images and hashtags. In our experiments, the dimensionality is set to $D = 300$ to match that of the word embeddings used in the next and last approach. This procedure does not apply to MLC for which groundtruth is multi-hot encoded.

Hashtag Embedding Regression

This approach is similar to the one used in Chapter 4. We use pretrained GloVe [69] embeddings for hashtags, which are D -dimensional with $D = 300$. For each image \mathbf{I}_x , we sum the GloVe embeddings of its groundtruth hashtags \mathbf{H}_x , which we denote as \mathbf{t}_x . Then we replace the last layer of the ResNet-50 by a D -dimensional linear layer, and we learn the parameters of the image model by minimizing a cosine embedding loss. If, $\mathbf{f}_x = f(\mathbf{I}_x; \theta)$ is the output of the vision model, the loss is defined by:

$$L = 1 - \left(\frac{\mathbf{t}_x \cdot \mathbf{f}_x}{\|\mathbf{t}_x\| \|\mathbf{f}_x\|} \right) \quad (5.3)$$

This method has benefits as we have seen in Chapter 4. As already stated by [90], because of the nature of the GloVe semantic space, this methodology has the potential

advantage of not penalizing predicting hashtags with close meanings to those in the groundtruth but that a user might not have used in the image description. Moreover, as shown in [17] and due to the semantics structure of the embedding space, the resulting image model will be less prone to drastic errors.

5.3.2 Location Sensitive Model (*LocSens*)

We design a location sensitive model that learns to score triplets formed by an image, a hashtag and a location. We use a siamese-like architecture and a ranking loss to optimize the model to score positive triplets (existing in the training set) higher than negative triplets (which we create). Given an image \mathbf{I}_x , we get its embedding \mathbf{r}_x computed by the image model, the embedding \mathbf{v}_{x_i} of a random hashtag h_x^i from its groundtruth set \mathbf{H}_x and its groundtruth latitude and longitude $\mathbf{g}_x = [\varphi_x, \lambda_x]$, which constitute a positive triplet. Both \mathbf{r}_x and \mathbf{v}_{x_i} are L2 normalized and latitude and longitude are both normalized to range in $[0, 1]$. Note that 0 and 1 latitudes fall on the poles while 0 and 1 represent the same longitude because of its circular nature and falls on the Pacific.

The three modalities are then mapped by linear layers with ReLU activations to 300 dimensions each, and L2 normalized again. This normalization guarantees that the magnitudes of the representations of the different modalities are equal when processed by subsequent layers in the multimodal network. Then the three vectors are concatenated. Although sophisticated multimodal data fusion strategies have been proposed, simple feature concatenation has also been proven to be an effective technique [92, 90]. We opted for a simple concatenation as it streamlines the strategy. The concatenated representations are then forwarded through 5 linear layers with normalization and ReLU activations with 2048, 2048, 2048, 1024, 512 neurons respectively. At the end, a linear layer with a single output calculates the score of the triplet. We have experimentally found that Batch Normalization [34] hampers learning, producing highly irregular gradients. We conjecture that all GPU-allowable batch size is in fact a small batch size for the problem at hand, since the number of triplets is potentially massive and the batch statistics estimation will always be erratic across batches. Group normalization [102] is used instead, which is independent of the batch size and permits learning of the models.

To create a negative triplet, we randomly replace the image or the tag of the positive triplet. The image is replaced by a random one not associated with the tag h_x^i , and the tag by a random one not in \mathbf{H}_x . We have found that the performance in image retrieval is significantly better when all negative triplets are created replacing the image. This is because the frequency of tags is preserved in both the positive and negative triplets, while in the tagging configuration less common tags are more frequently seen in negative triplets.

We train with a Ranking loss, with a margin set empirically to $m = 0.1$, use 6 negative triplets per positive triplet averaging the loss over them, and a batch size of 1024. If s_x is the score of the positive triplet and s_n the score of the negative triplet, the loss is

written as:

$$L = \max(0, s_n - s_x + m) \quad (5.4)$$

Figure 5.1 shows the model architecture and also the training strategies to balance location influence, which are explained next.

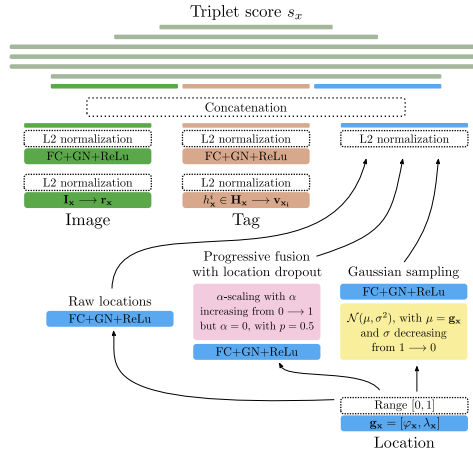


Figure 5.1: The proposed *LocSens* multimodal scoring model trained by triplet ranking (bars after concatenation indicate fully connected + group normalization + ReLU activation layers). During training, location information is processed and inputted to the model with different strategies.

Balancing Location Influence on Ranking.

One important challenge in multimodal learning is balancing the influence of the different data modalities. We started by introducing the raw location values into the *LocSens* model, but immediately observed that the learning tends to use the location information to discriminate between triplets much more than the other two modalities, forgetting previously learnt relations between images and tags. This effect is especially severe in the image retrieval scenario, where the model ends up retrieving images close to the query locations but less related to the query tag. This suggests that the location information needs to be gradually incorporated into the scoring model for location sensitive image retrieval. For that, we propose the following two strategies, also depicted in Figure 5.1.

Progressive Fusion with Location Dropout. We first train a model with *LocSens* architecture but silencing the location modality hence forcing it to learn to discriminate

triplets without using location information. To do that, we multiply by $\alpha = 0$ the location representation before its concatenation. Once the training has converged we start introducing locations progressively, by slowly increasing α until $\alpha = 1$. This strategy avoids new gradients caused by locations to ruin the image-hashtags relations *LocSens* has learned in the first training phase. In order to force the model to sustain the capability to discriminate between triplets without using location information we permanently zero the location representations with a 0.5 probability. We call this *location dropout* in a clear abuse of notation but because of its resemblance to zeroing random neurons in the well-known regularization strategy [83]. For the sake of comparison, we report results for the *LocSens* model with zeroed locations, which is in fact a location agnostic model.

Location Sampling. Exact locations are particularly narrow with respect to global coordinates and such a fine-grained degree of granularity makes learning troublesome. We propose to progressively present locations from rough precision to more accurate values while training advances. For each triplet, we randomly sample the training location coordinates at each iteration from a $2D$ normal distribution with mean at the image real coordinates ($\mu = \mathbf{g}_x$) and with standard deviation σ decreasing progressively. We constrain the sampling between $[0, 1]$ by taking modulo 1 on the sampled values. We start training with $\sigma = 1$, which makes the training locations indeed random and so not informative at all. At this stage, the *LocSens* model will learn to rank triplets without using the location information. Then, we progressively decrease σ , which makes the sampled coordinates be more accurate and useful for triplet discrimination. Note that σ has a direct relation with geographical distance, so location data is introduced during the training to be first only useful to discriminate between very distant triplets, and progressively between more fine-grained distances. Therefore, this strategy allows training models sensitive to different location levels of detail.

5.4 Experiments

5.4.1 Dataset

We conduct experiments on the YFCC100M dataset [86] which contains nearly 100 million photos from Flickr with associated hashtags and GPS coordinates among other metadata. We create the hashtag vocabulary following [90]: we remove numerical hashtags and the 10 most frequent hashtags since they are not informative. The hashtag set \mathbb{H} is defined as the set of the next 100,000 most frequent hashtags. Then we select photos with at least one hashtag from \mathbb{H} from which we filter out photos with more than 15 hashtags. Finally, we remove photos without location information. This results in a dataset of 24.8M images, from which we separate a validation set of 250K and a test set of 500K. Images have an average of 4.25 hashtags.

Notice the differences between this data and the data used in Chapter 4. In that previous Chapter, we learned from images with associated captions, which can be com-

plete sentences, and which have semantic meanings generally with synergetic relations with the image content, but not always. In here, we learn from images with associated tags, so the text is much more constrained in shape. But also in meaning, since tags are used to make the images searchable and they have a more strict synergetic relation with the images. That’s why, in this setup, it makes sense to define a limited vocabulary of tags and address the problem as a Multi-Class or multi-label classification problem, being the tags the categories. Also, this setup is closer to supervised learning, and tags can easily be called weak supervision.

5.4.2 Image by Tag Retrieval

We first study hashtag based image retrieval, which is the ability of our models to retrieve relevant images given a hashtag query. We define the set of querying hashtags \mathbb{H}^q as the hashtags in \mathbb{H} appearing at least 10 times in the testing set. The number of querying hashtags is 19,911. The precision@10 of the different location agnostic methods is as follows: MLC: 1.01, MCC: **14.07**, HER (GloVe): 7.02. The multi-class classification (MCC) model has the best performance in the hashtag based image retrieval task.

5.4.3 Location Sensitive Image by Tag Retrieval

We evaluate the ability of the models to retrieve relevant images given a query composed by a hashtag and a location (Figure 5.2, 5.3). A retrieved image is considered relevant if the query hashtag is within its groundtruth hashtags and the distance between its location and the query location is smaller than a given threshold. Inspired by [91], we use different distance thresholds to evaluate the models’ location precision at different levels of granularity.

We define our query set of hashtag-location pairs by selecting the location and a random hashtag of 200,000 images from the testing set. In this query set there will be repeated hashtags with different locations, and more frequent hashtags over all the dataset will also be more frequent in the query set (unlike in the location agnostic retrieval experiment). This query set guarantees that the ability of the system to retrieve images related to the same hashtag but different locations is evaluated. To retrieve images for a given hashtag-location query with *LocSens*, we compute triplet plausibility scores with all test images and rank them.

Table 5.1 shows the performance of the different methods in location agnostic image retrieval and in different location sensitive levels of granularity. In location agnostic retrieval (first column) the geographic distance between the query and the results is not evaluated (infinite distance threshold). The performances are higher because in this case the query sets contains more instances of the most frequent hashtags. The upper bound ranks the retrieval images containing the query hashtag by proximity to the query location, showcasing the optimal performance of any method in this evaluation. In location sensitive evaluations the optimal performance is less than 100%

because we do not always have 10 or more relevant images in the test set.

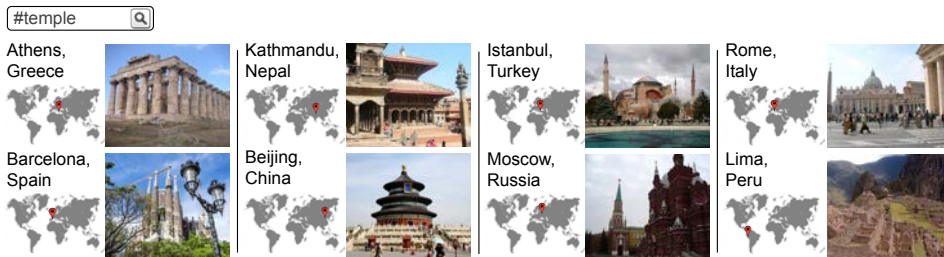


Figure 5.2: Top retrieved image by *LocSens*, our location sensitive model, for the query hashtag “*temple*” at different locations.

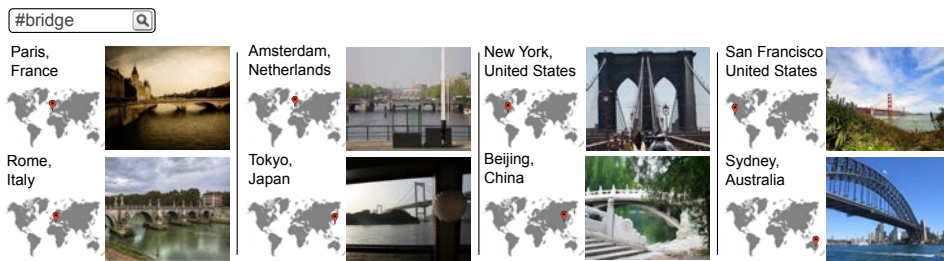


Figure 5.3: Top retrieved image by *LocSens*, our location sensitive model, for the query hashtag “*bridge*” at different locations.

Results show how the zeroed locations version of *LocSens* gets comparable results as MCC. By using raw locations in the *LocSens* model, we get the best results at fine level of location detail at the expense of a big drop in location agnostic retrieval. The reason is that it is relying heavily on locations to rank triplets decreasing its capability to predict relations between images and tags. As a result, it tends to retrieve images close to the query location, but less related to the query tag. The proposed dropout training strategy reduces the deterioration in location agnostic retrieval performance at a cost of a small drop in the fine levels of granularity. Also, it outperforms the former models in the coarse continent and country levels, due to its better balancing between using the query tag and location to retrieve related images. In its turn, the location sampling proposed approach with $\sigma = 1$ gets similar results as *LocSens* with zeroed locations because the locations are as irrelevant in both cases. When σ is decreased, the model improves its location sensitive retrieval performance while maintaining a high location agnostic performance. This is achieved because informative locations are introduced to the model in a progressive way, from coarse to fine, and always maintaining triplets where the location is not informative, forcing the network to retain its capacity to rank triplets using only the image and the tag.

Figure 5.4 shows the absolute and relative performances at different levels of granularity while σ is decreased. At $\sigma = 0.05$, it can be seen that the location sensitive per-

Table 5.1: Location sensitive hashtag based image retrieval: $P@10$. A retrieved image is considered correct if its groundtruth hashtags contain the queried hashtag and the distance between its location and the queried one is smaller than a given threshold (in kilometers in the table)

		$P@10$					
	Method	Location Agnostic	Continent (2500 km)	Country (750 km)	Region (200 km)	City (25 km)	Street (1 km)
	Upper Bound	100	96.08	90.51	80.31	64.52	42.46
Img + Tag	MLC	5.28	2.54	1.65	1.00	0.62	0.17
	MCC	42.18	29.23	24.2	18.34	13.25	4.66
	HER (GloVe)	37.36	25.03	20.27	15.51	11.23	3.65
	LocSens - Zeroed locations	40.05	28.32	24.34	18.44	12.79	3.74
Loc + Img + Tag	LocSens - Raw locations	32.74	28.42	25.52	21.83	15.53	4.83
	LocSens - Dropout	36.95	30.42	26.14	20.46	14.28	4.64
	LocSens - Sampling $\sigma = 1$	40.60	28.40	23.84	18.16	13.04	4.13
	LocSens - Sampling $\sigma = 0.1$	40.03	29.30	24.36	18.83	13.46	4.22
	LocSens - Sampling $\sigma = 0.05$	39.80	31.25	25.76	19.58	13.78	4.30
	LocSens - Sampling $\sigma = 0.01$	37.05	31.27	26.65	20.14	14.15	4.44
	LocSens - Sampling $\sigma = 0$	35.95	30.61	27.00	21.39	14.75	4.83

performances at all granularities have improved with a marginal drop on location agnostic performance. When σ is further decreased, performances at finer locations keep increasing, while the location agnostic performance decreases. When $\sigma = 0$, the training scenario is the same as in the raw locations one, but the training schedule allows this model to reduce the drop in location agnostic performance and at coarse levels of location granularity. The location sampling technique provides *LocSens* with a better balancing between retrieving images related to the query tag and their location. Furthermore, given that σ has a direct geographical distance interpretation, it permits to tune the granularity to which we want our model to be sensitive. Note that *LocSens* enables to retrieve images related to a tag and near to a given location, which location agnostic models cannot do. The performance improvements in Table 5.1 at the different levels of location granularity are indeed significant since for many triplets the geographic location is not informative at all.

Figures 5.2, 5.3 and 5.5 show qualitative retrieval results of several hashtags at different locations. They demonstrate that the model successfully fuses textual and location information to retrieve images related to the joint interpretation of the two query modalities, being able to retrieve images related to the same concept across a wide range of locations with different geographical distances between them. *LocSens* is able to distinguish between images related to the same concept across a wide range of cities with different geographical distances between them. Note that, despite some specific bridges might have a huge amount of images tagged with *bridge* in the dataset, as the San Francisco bridge or the Brooklyn bridge in New York, the system manages to retrieve images of other less represented bridges around the world. So, first and despite the bridges samples unbalance, it is learning to extract visual patterns that general-

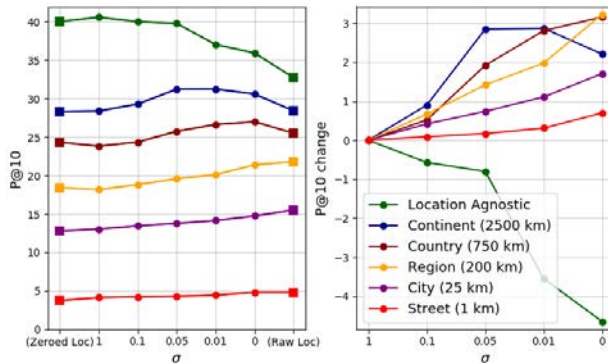


Figure 5.4: Left: $P@10$ of the location sampling strategy for different σ and models with zeroed and raw locations. Right: $P@10$ difference respect to $\sigma = 1$.

ize to many different bridges around the world and, second, it is correctly balancing the tag query and location query influence in the final score. Paper's Figure 5 shows *LocSens* results for hashtag queries in different locations. The model is able to retrieve images related to a wide range of tags, from tags referring to objects, such as *car*, to tags referring to more abstract concepts, such as *hiking*, from the 100.000 tags vocabulary. It goes beyond learning the most common images from each geographical location, as it is demonstrated by the *hiking* results in El Cairo or the *car* results in Paris, which are concepts that do not prevail in images in those locations, but the system is still able to accurately retrieve them.

Challenging queries

Figure 5.6 shows *LocSens* results for hashtag queries in different locations where some queries are incompatible because the hashtag refers to a concept which does not occur in the given location. When querying with the *beach* hashtag in a coastal location such as Auckland, *LocSens* retrieves images of close-by beaches. But when we query for *beach* images from Madrid, which is far away from the coast, we get bullfighting and beach volley images, because the sand of both arenas makes them visually similar to beach images. If we try to retrieve beach images near Moscow, we get scenes of people sunbathing. Similarly, if we query for *ski* images in El Cairo and Sydney, we get images of the dessert and water sports respectively, which have visual similarities with ski images.

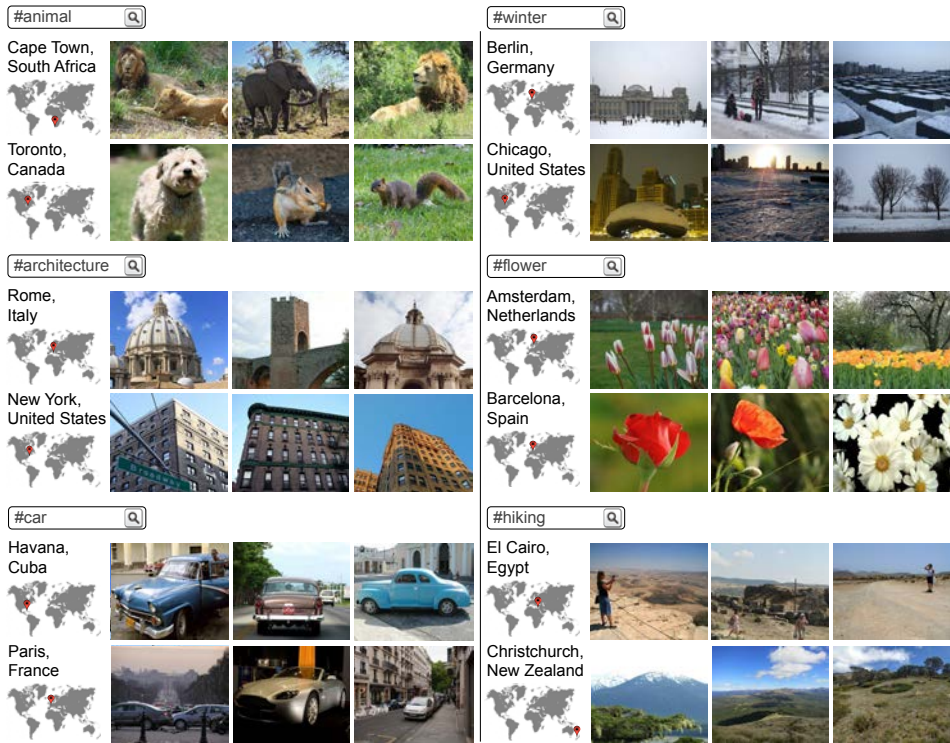


Figure 5.5: Query hashtags with different locations and top 3 retrieved images.

5.4.4 Image Tagging

In this section we evaluate the ability of the models to predict hashtags for images in terms of $A@k$ (accuracy at k). We evaluate accuracy at $k = 1$ and $k = 10$, which measure how often the first ranked hashtag is in the groundtruth and how often at least one of the 10 highest ranked hashtags is in the groundtruth respectively. A desired feature of a tagging system is the ability to infer diverse and distinct tags [100, 99]. In order to measure the variety of tags predicted by the models, we measure the percentage of all the test tags predicted at least once in the whole test set (%pred) and the percentage of all the test tags correctly predicted at least once (%cpred), considering the top 10 tags predicted for each image.

Table 5.2 shows the performance of the different methods. Global Frequency ranks the tags according to the training dataset frequency. Among the location agnostic methods, MCC is the best one. This finding corroborates the experiments in [59, 90] verifying that this simple training strategy outperforms others when having a large number of classes. To train the *LocSens* model we used the image and tag representations inferred by the MCC model, since it is the one providing the best results.

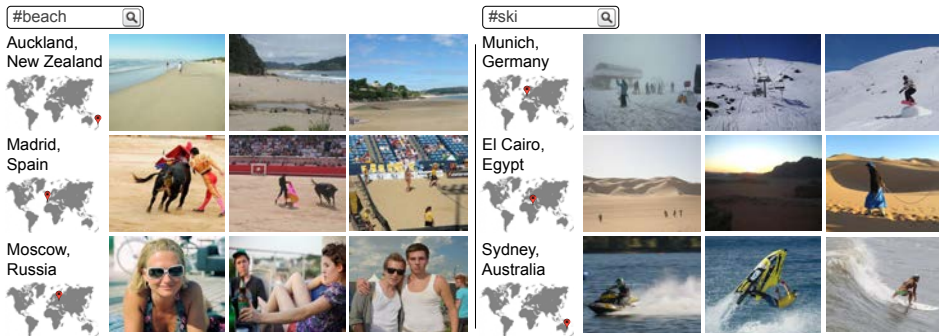


Figure 5.6: Query hashtags with different locations where some queries are incompatible because the hashtag refers to a concept which does not occur in the query location.

Table 5.2: Image tagging: $A@1$, $A@10$, %pred and %cpred of the frequency baseline, location agnostic prediction and the location sensitive model

Method	$A@1$	$A@10$	%pred	%cpred
Global Frequency	1.82	13.45	0.01	0.01
MLC	8.86	30.59	8.04	4.5
MCC	20.32	47.64	29.11	15.15
HER (GloVe)	15.83	31.24	18.63	8.74
LocSens - Zeroed locations	15.92	46.60	26.98	13.31
LocSens - Raw locations	28.10	68.21	44.00	24.04

To get the highest scoring tags for an image with location with *LocSens*, we compute triplet plausibility scores with all the tags and rank them. *LocSens - Zeroed locations* stands for a model where the location representations are zeroed, so it only learns to rank image and tag pairs. The aim of training this model is to check whether *LocSens* additional capacity and training strategy are providing a boost on location agnostic tagging. Results confirm they are not, since $A@10$ is comparable for both measures and $A@1$ drops significantly. This later deterioration is due to the softmax activations used in MCC, which foster highly frequent tags and penalize infrequent ones. Moreover, the training strategy of the *LocSens* model does not penalize infrequent tags that much but suffers greatly from the missing labels problem.

LocSens - Raw locations stands for the model where the raw triplets locations are always inputted both at train and test time. It outperforms the location agnostic methods in accuracy, successfully using location information to improve the tagging results. Moreover, it produces more diverse tags than location agnostic models, demonstrating that using location is effective for augmenting the hashtag prediction diversity. Figure 5.7 shows some tagging examples of a location agnostic model (MCC) compared to *LocSens*, that demonstrate how the later successfully processes jointly visual and lo-

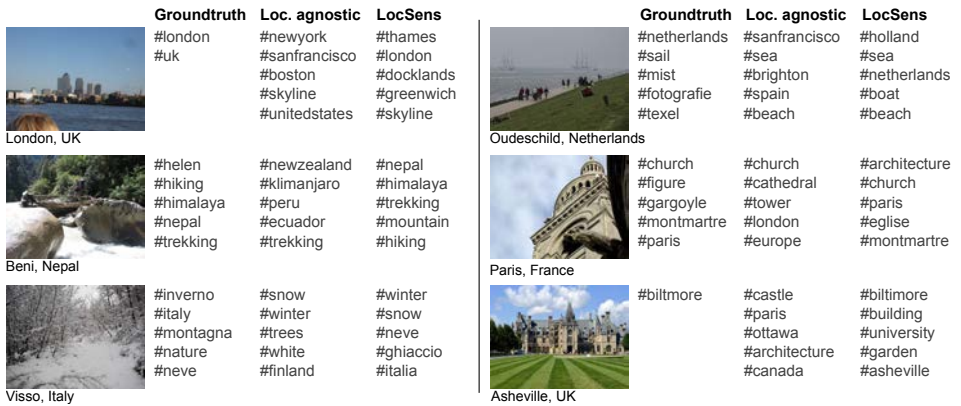


Figure 5.7: Images with their locations and groundtruth hashtags and the correspond top 5 hashtags predicted by the location agnostic MCC model and *LocSens*.

cation information to assign tags referring to the concurrence of both data modalities. As seen in the first example, besides assigning tags directly related to the given location (*london*) and discarding tags related to locations far from the given one (*newyork*), *LocSens* predicts tags that need the joint interpretation of visual and location information (*thames*). Figure 5.8 shows *LocSens* tagging results on images with different faked locations, and demonstrates that *LocSens* jointly interprets the image and the location to assign better contextualized tags, such as *caribbean* if a sailing image is from Cuba, and *lake* if it is from Toronto. Similarly, Figure 5.9 demonstrate that *LocSens* is able to exploit locations to assign better contextualized tags, jointly interpreting both query visual and location modalities. For instance, it assigns to the river image *lake* and *west-lake* if it is from Los Angeles, since Westlake is the nearest important water geographic accident, while if the image is from Rio de Janeiro it tags it with *amazonia* and *rainforest*, and with *nile* if it is from El Cairo. In the example of an image of a road, it predicts as one of the most probable tags *carretera* (which means *road* in spanish) if the image is from Costa Rica, while it predicts *hills*, *Cumbria* and *Scotland* if the image is from Edinburgh, referring to the geography and the regions names around. If the image is from Chicago, it predicts *interstate*, since the road in it may be from the United States interstate highway system. These examples prove the joint interpretation of the visual and the location modalities to infer the most probable tags, since predicted tags are generally related to the image content while clearly conditioned by the image location.

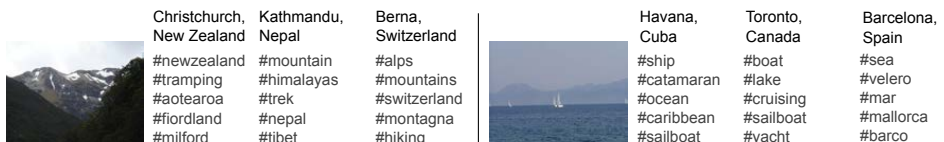


Figure 5.8: *LocSens* top predicted hashtags for images with different faked locations.

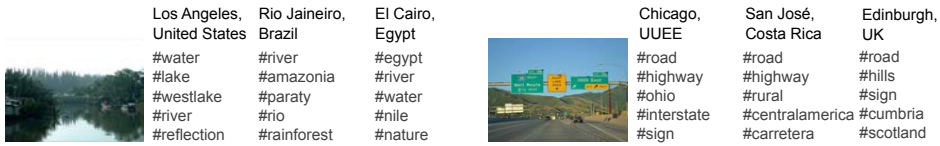


Figure 5.9: *LocSens* top predicted hashtags for images with different faked locations.

5.4.5 Conclusions

In Chapter 4 we saw the benefits of exploiting pre-learned semantic text representations to learn image embeddings using text supervision. However, in this Chapter we have seen that in a tags supervision setup a simpler approach delimiting a vocabulary and setting the task as a classification problem results in better performance. Particularly, we have seen that a multi-class classification setup is the best method to learn image and tag representations using tags supervision when a large number of classes is available.

Using the learned image and tags embeddings, we have trained *LocSens* to rank image-tag-coordinates triplets by plausibility. We have shown how it is able to perform image by tag retrieval conditioned to a given location by learning location-dependent visual representations, and have demonstrated how it successfully utilizes location information for image tagging, providing better contextual results. We have identified a problem in the multimodal setup, especially acute in the retrieval scenario: *LocSens* heavily relies on location for triplet ranking and tends to return images close to the query location and less related to the query tag. To address this issue we have proposed two novel training strategies: progressive fusion with location dropout, which allows training with a better balance between the modalities influence on the ranking, and location sampling, which results in a better overall performance and enables to tune the model at different levels of distance granularity.

Chapter 6

Multimodal Hate Speech Detection

Research Question 8: *Can we exploit the joint context provided by textual and image data for hate speech classification?*

6.1 Introduction

In Chapter 4 we exploited captions associated to images in Social Media data and pre-learned text representations as supervision to learn visual features. In Chapter 5 we learned visual and textual representations from images with associated tags using a multi-class classification setup where each tag is understood as a category. Also, we learnt a model to score triplets formed by an image, tag and a location by plausibility. In this Chapter we also work with images with associated text, but addressing a multimodal data classification task. Specifically, we want aim to classify multimodal data formed by an image, an associated text, and text we might find in the image as hate speech or not hate speech. Differently from previous chapters, to solve this task we'll join visual and textual representations, set the problem as a binary classification task and address it with a Binary Cross-Entropy Loss.

Social Media platforms such as Facebook, Twitter or Reddit have empowered individuals' voices and facilitated freedom of expression. However they have also been a breeding ground for hate speech and other types of online harassment. Hate speech is defined in legal literature as speech (or any form of expression) that expresses (or seeks to promote, or has the capacity to increase) hatred against a person or a group of people because of a characteristic they share, or a group to which they belong [31]. Twitter develops this definition in its hateful conduct policy¹ as *violence against or directly attack or threaten other people on the basis of race, ethnicity, national origin,*

¹Twitter hateful conduct policy : <https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy>

sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease.

Modern social media content usually include images and text. Some of these multimodal publications are only hate speech because of the combination of the text with a certain image. That is because the presence of offensive terms does not itself signify hate speech, and the presence of hate speech is often determined by the context of a publication. Moreover, users authoring hate speech tend to intentionally construct publications where the text is not enough to determine they are hate speech. This happens especially in Twitter, where multimodal tweets are formed by an image and a short text, which in many cases is not enough to judge them. In those cases, the image might give extra context to make a proper judgement. The research presented in this Chapter was also published in [23].

In this Chapter:

- We present the novel task of hate speech detection in multimodal publications, collect, annotate and publish a large scale dataset.
- We evaluate state of the art multimodal models on this specific task and compare their performance with unimodal detection. Even though images are proved to be useful for hate speech detection, the proposed multimodal models do not outperform unimodal textual models.
- We study the challenges of the proposed task, and open the field for future research.

6.2 Background

6.2.1 Background on Multimodal Hate Speech Detection

The literature on detecting hate speech on online textual publications is extensive [80, 78, 12, 16, 106]. However, although often modern social media publications include images, not too many contributions exist that exploit visual information. Zhong *et al.* [108] worked on classifying Instagram images as potential cyberbullying targets, exploiting both the image content, the image caption and the comments. However, their visual information processing is limited to the use of features extracted by a pre-trained CNN, the use of which does not achieve any improvement. Hosseinmardi *et al.* [33] also address the problem of detecting cyberbullying incidents on Instagram exploiting both textual and image content. But, again, their visual information processing is limited to use the features of a pre-trained CNN, and the improvement when using visual features on cyberbullying classification is only of 0.01%.

Parallel to this work, Yang *et al.* [103], working at Facebook, addressed the same problem as us but with a dataset made by user-reported publications from their social

network with further platform moderation, which has not been published. They explore different multimodal feature fusion strategies which provide small performance boosts compared to a simple feature concatenation. Also, Sabat *et al.* [77] worked on the similar problem of detecting hate speech memes using both images and the text extracted from them. Recently Facebook [44] published a reduced but accurately annotated dataset for hate speech detection in multimodal memes, and opened a challenge to push forward the research in the task². We have seen that, since we started working on multimodal hate speech detection, it has become a hot topic which will probably lead to a big research effort in the following years.

6.2.2 Background on Visual and Textual Data Fusion

A typical task in multimodal visual and textual analysis is to learn an alignment between feature spaces. This approach is applied in tasks such as image captioning [41] and multimodal image retrieval [29]. On the other hand, instead of explicitly learning an alignment between two spaces, the goal of Visual Question Answering (VQA) is to merge both data modalities in order to decide which answer is correct. This problem requires modeling very precise correlations between the image and the question representations. The VQA task requirements are similar to our hate speech detection problem in multimodal publications, where we have a visual and a textual input and we need to combine both sources of information to understand the global context and make a decision. We thus take inspiration from the VQA literature for the tested models. Early VQA methods [109] fuse textual and visual information by feature concatenation. In a recent work, Gao *et al.* [19] propose a feature fusion scheme to overcome these limitations. They learn convolution kernels from the textual information –which they call question-guided kernels– and convolve them with the visual information in an earlier stage to get the multimodal features. Margffoy-Tuay *et al.* [60] use a similar approach to combine visual and textual information, but they address a different task: instance segmentation guided by natural language queries. We inspire in these latest feature fusion works to build the models for hate speech detection.

6.3 The MMHS150K Dataset

Existing hate speech datasets contain only textual data. We create a new manually annotated multimodal hate speech dataset formed by 150,000 tweets, each one of them containing text and an image. We call the dataset *MMHS150K*, and made it available online³.

We used the Twitter API to gather real-time tweets from September 2018 until February 2019, selecting the ones containing any of the 51 Hatebase terms that are more common in hate speech tweets, as studied in [16]. From that selection, we kept the

²<https://www.drivendata.org/competitions/64/hateful-memes/>

³MMHS150K Dataset: <https://gomburu.github.io/2019/10/09/MMHS/>

ones that included images and downloaded them. We aim to create a multimodal hate speech database where all the instances contain visual and textual information that we can later process to determine if a tweet is hate speech or not. But a considerable amount of the images of the selected tweets contain only textual information, such as screenshots of other tweets. To ensure that all the dataset instances contain both visual and textual information, we remove those tweets. To do that we use TextFCN [6, 4], explained in Chapter 3, a Fully Convolutional Network that produces a pixel wise text probability map of an image. We set empirical thresholds on the image text probability sum to discard images that have a substantial text probability.

We annotate the gathered tweets using the crowdsourcing platform Amazon Mechanical Turk. There, we give the workers the definition of hate speech and show some examples to make the task clearer. We then show the tweet text and image and we ask them to classify it in one of 6 categories: *No attacks to any community*, *racist*, *sexist*, *homophobic*, *religion based attacks* or *attacks to other communities*. Each one of the 150,000 tweets is labeled by 3 different workers to palliate discrepancies among workers.

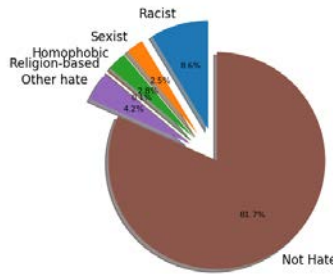


Figure 6.1: Percentage of tweets per class in MMHS150K.

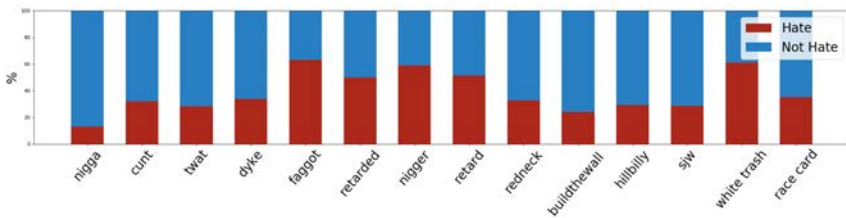


Figure 6.2: Percentage of hate and not hate tweets for top keywords of MMHS150K.

We do a majority voting between the three annotations to get the tweets category. At the end, we obtain 112,845 not hate tweets and 36,978 hate tweets. The latest are divided in 11,925 racist, 3,495 sexist, 3,870 homophobic, 163 religion-based hate and 5,811 other hate tweets (Figure 6.1). In this work, we do not use hate sub-categories,

and stick to the hate / not hate split. We separate balanced validation (5,000) and test (10,000) sets. The remaining tweets are used for training. As far as we know, this dataset is the biggest hate speech dataset to date, and the first multimodal hate speech dataset. One of its challenges is to distinguish between tweets using the same key offensive words that constitute or not an attack to a community (hate speech). Figure 6.2 shows the percentage of hate and not hate tweets of the top keywords.

The images and texts of the tweets in this dataset, can have different semantic relations. Some of them, where the text and the image carry the same information, will be synergetic. In those cases hate speech could be detected by unimodal models. Others will be complementary, and in those the images and the text might have to be interpreted together to detect hate speech. Those tweets are undetectable by an unimodal hate speech detection system, and detecting them is the objective of this research. Other tweets won't have a relevant semantic relation between images and text, and in those the joint context provided by both modalities won't help on the detection.

6.4 Methodology

6.4.1 Unimodal Treatment

Images

We use a CNN as the image features extractor which is an Imagenet [37] pre-trained Google Inception v3 architecture [85]. The fine-tuning process of the Inception v3 layers aims to modify its weights to extract the features that, combined with the textual information, are optimal for hate speech detection.

Tweet Text

We train a single layer LSTM with a 150-dimensional hidden state for hate / not hate classification. GloVe [69] embeddings are used as word input representations. Since our dataset is not big enough to train a GloVe word embedding model, we used a pre-trained model that has been trained in two billion tweets⁴. This ensures that the model will be able to produce word embeddings for slang and other words typically used in Twitter. To encode the tweet text and input it later to multimodal models, we use the LSTM hidden state after processing the last tweet word. Since the LSTM has been trained for hate speech classification, it extracts the most useful information for this task from the text, which is encoded in the hidden state after inputting the last tweet word.

⁴Pretrained GloVe word embedding model: <https://nlp.stanford.edu/projects/glove/>

Image Text

The text in the image can also contain important information to decide if a publication is hate speech or not, so we extract it and also input it to our model. To do so, we use Google Vision API Text Detection module. We input the tweet text and the text from the image separately to the multimodal models, so it might learn different relations between them and between them and the image. For instance, the model could learn to relate the image text with the area in the image where the text appears, so it could learn to interpret the text in a different way depending on the location where it is written in the image. The image text is also encoded by the LSTM as the hidden state after processing its last word.

6.4.2 Multimodal Architectures

The objective of this research is to build a hate speech detector that leverages both textual and visual data and detects hate speech publications based on the context given by both data modalities. To study how the multimodal context can boost the performance compared to an unimodal context we evaluate different models: a Feature Concatenation Model (FCM), a Spatial Concatenation Model (SCM) and a Textual Kernels Model (TKM). All of them are CNN+RNN models with three inputs: the tweet image, the tweet text and the text appearing in the image (if any).

Feature Concatenation Model (FCM)

The image is fed to the Inception v3 architecture and the 2048 dimensional feature vector after the last average pooling layer is used as the visual representation. This vector is then concatenated with the 150 dimension vectors of the LSTM last word hidden states of the image text and the tweet text. This vector is then processed by three fully connected layers until the dimensions are reduced to two, the number of classes, in the last classification layer. The FCM architecture is illustrated in Figure 6.3.

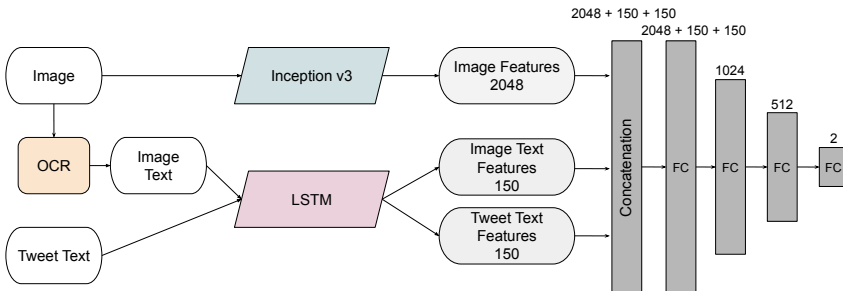


Figure 6.3: FCM architecture. Image and text representations are concatenated and processed by a set of fully connected layers.

Spatial Concatenation Model (SCM)

Instead of using the latest feature vector before classification of the Inception v3 as the visual representation, in the SCM we use the $8 \times 8 \times 2048$ feature map after the last Inception module. Then we concatenate the 150 dimension vectors encoding the tweet text and the tweet image text at each spatial location of that feature map. The resulting multimodal feature map is processed by two Inception-E blocks [84]. After that, dropout and average pooling are applied and, as in the FCM model, three fully connected layers are used to reduce the dimensionality until the classification layer.

Textual Kernels Model (TKM)

The TKM design, inspired by [19] and [60], aims to capture interactions between the two modalities more expressively than concatenation models. As in SCM we use the $8 \times 8 \times 2048$ feature map after the last Inception module as the visual representation. From the 150 dimension vector encoding the tweet text, we learn K_t text dependent kernels using independent fully connected layers that are trained together with the rest of the model. The resulting K_t text dependent kernels will have dimensionality of $1 \times 1 \times 2048$. We do the same with the feature vector encoding the image text, learning K_{it} kernels. The textual kernels are convolved with the visual feature map in the channel dimension at each spatial location, resulting in a $8 \times 8 \times (K_t + K_{it})$ multimodal feature map, and batch normalization is applied. Then, as in the SCM, the 150 dimension vectors encoding the tweet text and the tweet image text are concatenated at each spatial dimension. The rest of the architecture is the same as in SCM: two Inception-E blocks, dropout, average pooling and three fully connected layers until the classification layer. The number of tweet textual kernels K_t and tweet image textual kernels K_{it} is set to $K_t = 10$ and $K_{it} = 5$. The TKM architecture is illustrated in Figure 6.4.

Training

We train the multimodal models with a Cross-Entropy Loss and an ADAM optimizer with an initial learning rate of $1e-4$. Our dataset suffers from a high class imbalance, so we weight the contribution to the loss of the samples to totally compensate for it. One of the goals of this research is to explore how every one of the inputs contributes to the classification and to prove that the proposed model can learn concurrences between visual and textual data useful to improve the hate speech classification results on multimodal data. To do that we train different models where all or only some inputs are available. When an input is not available, we set it to zeros, and we do the same when an image has no text.

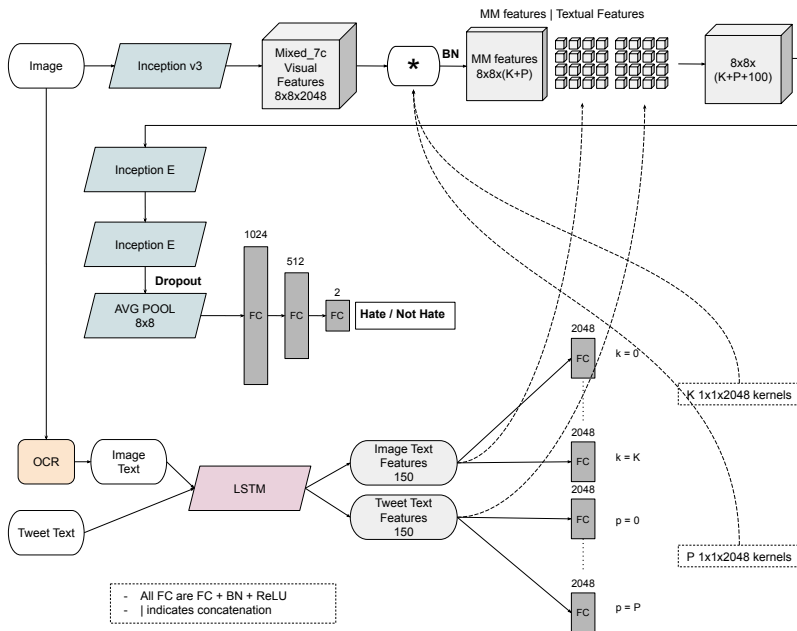


Figure 6.4: TKM architecture. Textual kernels are learnt from the text representations, and convolved with the image representation.

6.5 Results

Table 6.1 shows the F-score, the Area Under the ROC Curve (AUC) and the mean accuracy (ACC) of the proposed models when different inputs are available. *TT* refers to the tweet text, *IT* to the image text and *I* to the image. It also shows results for the LSTM, for the Davison method proposed in [12] trained with *MMHS150K*, and for random scores.

First, notice that given the subjectivity of the task and the discrepancies between annotators, getting optimal scores in the evaluation metrics is virtually impossible. However, a system with relatively low metric scores can still be very useful for hate speech detection in a real application: it will fire on publications for which most annotators agree they are hate, which are often the stronger attacks. The proposed LSTM to detect hate speech when only text is available, gets similar results as the method presented in [12], which we trained with *MMHS150K* and the same splits. However, more than substantially advancing the state of the art on hate speech detection in textual publications, our key purpose in this research is to introduce and work on its detection on multimodal publications. We use LSTM because it provides a strong representation of the tweet texts. The FCM trained only with images gets decent results, considering that in many publications the images might not give any useful information for the task.

Model	Inputs	F	AUC	ACC
Random	-	0.666	0.499	50.2
Davison [12]	<i>TT</i>	0.703	0.732	68.4
LSTM	<i>TT</i>	0.703	0.732	68.3
FCM	<i>TT</i>	0.697	0.727	67.8
FCM	<i>TT, IT</i>	0.697	0.722	67.9
FCM	<i>I</i>	0.667	0.589	56.8
FCM	<i>TT, IT, I</i>	0.704	0.734	68.4
SCM	<i>TT, IT, I</i>	0.702	0.732	68.5
TKM	<i>TT, IT, I</i>	0.701	0.731	68.2

Table 6.1: Performance of the proposed models, the LSTM and random scores. The Inputs column indicates which inputs are available at training and testing time.

Despite the model trained only with images proves that they are useful for hate speech detection, the proposed multimodal models are not able to improve the detection compared to the textual models. Besides the different architectures, we have tried different training strategies, such as initializing the CNN weights with a model already trained solely with *MMHS150K* images or using dropout to force the multimodal models to use the visual information. Eventually, though, these models end up using almost only the text input for the prediction and producing very similar results to those of the textual models. The proposed multimodal models, such as TKM, have shown good performance in other tasks, such as VQA. Next, we analyze why they do not perform well in this task and with this data:

- **Noisy data:** A major challenge of this task is the discrepancy between annotations due to subjective judgement. Although this affects also detection using only text, its repercussion is bigger in more complex tasks, such as detection using images or multimodal detection.
- **Complexity and diversity of multimodal relations:** Hate speech multimodal publications employ a lot of background knowledge which makes the relations between visual and textual elements they use very complex and diverse, and therefore difficult to learn by a neural network.
- **Small set of multimodal examples:** Although we have collected a big dataset of 150K tweets, the subset of multimodal hate there is still too small to learn the complex multimodal relations needed to identify multimodal hate.

6.6 Conclusions

We have explored the task of hate speech detection on multimodal publications. We have created *MMHS150K*, to our knowledge the biggest available hate speech dataset, and the first one composed of multimodal data, namely tweets formed by image and text. We have trained different textual, visual and multimodal models with that data, and found out that, despite the fact that images are useful for hate speech detection, the multimodal models do not outperform the textual models. Finally, we have analyzed the challenges of the proposed task and dataset. Given that most of the content in Social Media nowadays is multimodal, we truly believe on the importance of pushing forward this research.

Chapter 7

Conclusions

In this thesis we have researched on techniques for scene interpretation when both visual and textual data are available. In each Chapter we have worked with multimodal image and textual data that had different format relations and also potentially different semantic relations. Also, we have addressed different tasks that require dealing with the multimodal data in diverse ways, and we have investigated which techniques are the best ones for each one of the cases.

Text Detection and Style Transfer: Learning the Image Text Texture

In Chapter 3 we researched on detection of scene text appearing in images content. We presented a Fully Convolutional Network to segment text in images at pixel level, *TextFCN*. We analyzed the patterns the FCN is learning to detect text, and found that is learning generic texture patterns that are robust to detect text with high recall in any language, but that also detects as text other symbols with similar textures as text, but do not belong to any alphabet. Therefore, the text detection is based on the text texture, and not in recognizing its specific symbols. Also, we saw how *TextFCN* can be exploited to improve an objects proposals plus text recognizer pipeline for scene text detection by suppressing non-textual image areas in the object proposals stage.

Later in Chapter 3, we proposed a style transfer network able to learn text style patterns from a single image and to transfer them to other texts, and which works in handwritten text, machine printed text and scene text. The proposed style transfer is based on the text texture patterns *TextFCN* learns for text detection. Text fonts have generic styles for all characters, such as a given tilt, stroke width, color, or shadow. The proposed style transfer network learns this generic font patterns and, learning the same powerful texture patterns instead of character patterns, is able to apply them to another texts without recognizing the individual characters. This results in a powerful, simple and efficient text style transfer method compared to character-by-character

style transfer approaches. First, we start with a model that uses *TextFCN* to transfer the style only to textual areas. Then, we propose a model that learns to detect textual areas and perform the style transfer at once.

A further research line related to this chapter would be researching if neural style transfer is an effective data augmentation method for other tasks, such as generic object detection.

Learning from Social Media Images with Associated Captions

In Chapter 4 we explained our research on exploiting Social Media data consisting in images with associated captions to learn visual features using text supervision. Captions are very diverse, but generally have a synergetic semantic relation with images, and that is why they can be exploited as supervision. We proposed a pipeline to explode the semantic structure of pre-learned text representation methods to learn image embeddings in a space with semantic structure. We showed that applying that method to semantic multimodal image retrieval allows us to retrieve images semantically related to different queries, and to perform complex queries by combining concepts with arithmetics. We compared state of the art text representations methods in that setup, concluding that Word2Vec [62] and GloVe [69] are the ones providing better performance. However, later experiments showed that when using data consisting on long articles with associated images, such as Wikipedia articles, LDA works better.

Multimodal semantic image retrieval is a task which is addressed by many researchers but lacks a consistent benchmark and evaluation. Some of the works working on it use the COCO [52] dataset, which consist on images with associated captions, to evaluate image by text (by caption) retrieval. But that setup and evaluation is very limited since the dataset is small and limited in semantics, it does not consider image queries or multimodal queries, and queries are captions which are less appropriate to evaluate the semantics of the results than queries consisting on arithmetic between concepts. So future work for the community would be to create a consistent multimodal semantic image retrieval, or at least semantic image retrieval by text, benchmark.

Later in Chapter 4 we showed how learning from Social Media data by text supervision can have very interesting applications. We showed how, by training on Instagram data associated to a city, we can learn which visual features people associate with different concepts. Also how, by doing a language-separate analysis, we can learn which images different language speakers (and tourist and locals) associate with the different neighbourhoods of a city, and so learn how differently the neighbourhoods are perceived and lived.

This Chapter and other related works show that learning from images with associated text using the text as supervision is a powerful method to learn visual features for different tasks. That fact, together with the extensive literature on self-supervised learning and unsupervised learning, announces that traditional supervised learning might be expendable in a near future.

Learning from Social Media Images with Associated Tags and Locations

In Chapter 5 we learned from Social Media data consisting on images with tags and geolocation. First, we studied which one is the best technique to learn using the tags as supervision. Tags are words we use to make images searchable, so they have a syntergetic semantic relation with images. Despite the advantages of using pre-learned semantic text representations shown in Chapter 4, we found that when the data consists on images with associated tags is better to setup the learning as a simple classification problem. Particularly, we found that in this case where we have a huge number of tasks, setting the problem as a multi-class classification using Softmax activations (using a randomly selected tag from the positives) and a Cross-Entropy Loss results in better performance. Despite setting a multi-label problem, where we have multiple positive classes, as a multi-class problem with Softmax activations is counter-intuitive, both this work and the work by Veit [90] prove it results in the best performance.

After finding the best method to learn visual representations using tags supervision, we present the novel task of location-sensitive image retrieval, where we aim to retrieve images related to a given tag and near to a given location, and image tagging with location information. We propose *LocSens*, a model that learns to score image, tag and location triplets by plausibility. We show how it correctly exploits location information to improve image tagging and how, using proposed techniques to manage tag and location influence, is able to retrieve images related to a given tag and near to a given location at different granularities.

Deep learning techniques are evolving very fast, and are being applied to many data modalities with impressive results. The research panorama has been revolutionized by neural networks in many fields by improving previous methods results loudly. However, most of the published research deals only with one data modality, and at most with two. But the data on the Web and Social Media is multimodal, and further research on how to deal with different data modalities in deep learning models is required. Recently Wang *et al.* [98] analyzed the challenges of training a network with multimodal data and proposed a method to balance the gradients of the different modalities that achieve superior results on video recognition. Training neural networks with multimodal data is still an open problem and an interesting research line.

Multimodal Hate Speech Classification

In Chapter 6 we address a problem where we need to classify multimodal data consisting on an image, an associated text and text that might appear in the image. Specifically, we address multimodal hate speech classification of tweets. Differently from Chapters 4 and 5 in this case we have inputs of two different modalities to a neural network and we want to jointly analyze the information to make a decision. To do that, we first learn unimodal visual and textual representation, and then merge them inside a neural network. Several techniques to merge representations of different modalities in a neural network have been proposed in the literature, but none of them showed significant

improvements over a simple feature vectors concatenations.

Despite nowadays most of the Social Media content is multimodal, the problem of hate speech detection on multimodal publications has never been properly addressed. That is why we created the first multimodal hate speech dataset *MMHS150K*, which we publish for further research. We conduct experiments using different multimodal classification models, and concluded that, even though visual information is useful for hate speech detection, multimodal detection do not outperform unimodal textual detection. We analyze the challenges of this task which are related with the image and text semantic relations we find in multimodal hate speech. We find tweets where the images and the text have synergetic relations in which ones, despite the text solely might be sufficient to detect they are hate speech, the image can provide valuable information. We find tweets where the images and text have a complementary semantic relation, where both modalities should be interpreted together to detect they are hate speech. Those tweets are undetectable by unimodal models, and detecting them is the objective of our research. Finally, we find tweets where the image and the text have no relevant semantic relation. We suspect that the amount of tweets with complementary image-text semantic relations are not sufficient in *MMHS150K* to make the proposed multimodal models outperform the unimodal textual detection models.

Multimodal hate speech detection is a task with big scientific, social and political interests, and with the work presented in Chapter 6 we aim to open the field for further research. In future research on this task, the community could consider transfer learning from other domains (such as visual features relevant in hate images) in order be able to learn with less amount of labeled hate speech data. Multimodal hate speech detection has attracted a lot of attention lately. Facebook exploited Social Media user reports and interactions to create a dataset to train hate speech detection models [103], however, they do not publish the used dataset. Recently, also Facebook created a synthetic multimodal hate speech detection dataset [44], published it, and opened a detection challenge¹.

General Conclusions

In this PhD we have experimented how to use latest computer vision and natural language processing techniques, which shown a solid performance on controlled datasets, to learn from high scale and unconstrained multimodal Social Media data. We have benchmarked different techniques under diverse types of data, such as images from Instagram accompanied by a caption, or images from Flickr accompanied with tags. Finally, we have addressed specific tasks with direct social or industry applications learning from Social Media data: Analyze which visual perception users have from city neighbourhoods, retrieving images related to a given tag and near to a given location, or detecting hate speech in multimodal publications. We believe that, despite research under controlled scenarios and data is necessary to evaluate improvements, experimentation under uncontrolled scenarios and addressing applied tasks is also neces-

¹ai.facebook.com/blog/hateful-memes-challenge-and-data-set

sary to ascertain how that research can benefit or affect our society.

List of Publications

- *Improving Text Proposals for Scene Images with Fully Convolutional Networks*
Dena Bazazian, **Raul Gomez**, Angelos Nicolaou, Lluís Gomez, Dimosthenis Karatzas and Andrew Bagdanov
International Conference on Pattern Recognition (ICPR) DLPR workshop, 2016
- *FAST: Facilitated and Accurate Scene Text Proposals through FCN Guided Pruning*
Dena Bazazian, **Raul Gomez**, Angelos Nicolaou, Lluís Gomez, Dimosthenis Karatzas and Andrew Bagdanov
Pattern Recognition Letters, 2017
- *ICDAR2017 Robust Reading Challenge on COCO-Text*
Raul Gomez, Baoguang Shi, Lluís Gomez, Lukas Numann, Andreas Veit, Jiri Matas, Serge Belongie and Dimosthenis Karatzas
International Conference on Document Analysis and Recognition (ICDAR), 2017
- *TextTopicNet - Self-Supervised Learning of Visual Features Through Embedding Images on Semantic Text Spaces*
Yash Patel, Lluís Gomez, **Raul Gomez**, Marçal Rusiñol, Dimosthenis Karatzas and CV Jawahar
arXiv preprint, 2018
- *Learning to Learn from Web Data through Deep Semantic Embeddings*
Raul Gomez, Lluís Gomez, Jaume Gibert and Dimosthenis Karatzas
European Conference on Computer Vision (ECCV) MULA workshop (Oral), 2018
- *Learning from Barcelona Instagram data what Locals and Tourists post about its Neighbourhoods*
Raul Gomez, Lluís Gomez, Jaume Gibert and Dimosthenis Karatzas
European Conference on Computer Vision (ECCV) workshop, 2018
- *Self-Supervised Learning from Web Data for Multimodal Retrieval*
Raul Gomez, Lluís Gomez, Jaume Gibert and Dimosthenis Karatzas
Multi-Modal Scene Understanding, 2019
- *Selective Style Transfer for Text*
Raul Gomez*, Ali Furkan Biten*, Lluís Gomez, Jaume Gibert, Marçal Rusiñol and

Dismosthenis Karatzas

International Conference on Document Analysis and Recognition (ICDAR), 2019

- *ARSI: An Aerial Robot for Sewer Inspection*

François Chataigner, Pedro Cavestany, Marcel Soler, Carlos Rizzo, Jesus-Pablo Gonzalez, Carles Bosch, Jaume Gibert, Antonio Torrente, **Raul Gomez** and Daniel Serrano

Advances in Robotics Research: From Lab to Market, 2019

- *Exploring Hate Speech Detection in Multimodal Publications*

Raul Gomez, Jaume Gibert, Lluís Gomez and Dimosthenis Karatzas

Winter Conference on Applications of Computer Vision (WACV), 2020

- *Location Sensitive Image Tagging and Retrieval*

Raul Gomez, Jaume Gibert, Lluís Gomez and Dimosthenis Karatzas

European Conference on Computer Vision (ECCV), 2020

- *Unsupervised Retrieval-guided Image-to-image Translation*

Raul Gomez*, Yahui Liu*, Marco de Nadai, Bruno Lepri, Lluís Gomez, Jaume Gibert, Dimosthenis Karatzas and Nicu Sebe

Submitted to ACM Multimedia Conference (ACMMM), 2020

Publications are displayed in chronological order. The content of this dissertation is based on the publications where Raul Gomez is the first author.

* Stands for equal contribution.

Bibliography

- [1] Kotaro Abe, Brian Kenji Iwana, Viktor Gösta Holmér, and Seiichi Uchida. Font Creation Using Class Discriminative Deep Convolutional Generative Adversarial Networks. *ACPR*, 2017.
- [2] Emre Aksan, Fabrizio Pece, and Otmar Hilliges. DeepWriting: Making Digital Ink Editable via Deep Generative Modeling. *Conference on Human Factors in Computing Systems*, 2018.
- [3] Samaneh Azadi, Matthew Fisher, Vladimir Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-Content GAN for Few-Shot Font Style Transfer. *CVPR*, 2018.
- [4] Dena Bazazian, Raúl Gómez, Anguelos Nicolaou, Lluís Gómez, Dimosthenis Karatzas, and Andrew D. Bagdanov. FAST: Facilitated and Accurate Scene Text Proposals through FCN Guided Pruning. *Pattern Recognition Letters*, 2017.
- [5] Dena Bazazian, Raul Gomez, Anguelos Nicolaou, Lluís Gomez, Dimosthenis Karatzas, and Andrew D. Bagdanov. Improving Text Proposals for Scene Images with Fully Convolutional Networks. *ICPR DLPR Workshop*, 2017.
- [6] Dena Bazazian, Raul Gomez, Anguelos Nicolaou, Lluís Gomez, Dimosthenis Karatzas, and Andrew D. Bagdanov. Improving Text Proposals for Scene Images with Fully Convolutional Networks. *DLPR ICPR workshop*, 2017.
- [7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003.
- [8] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching Word Vectors with Subword Information. *arXiv*, 2016.
- [9] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11218 LNCS, pages 139–156, 2018.

-
- [10] David M Chen, Georges Baatz, Kevin Köser, Sam S Tsai, Ramakrishna Vedantam, Timo Pylvänäinen, Kimmo Roimela, Xin Chen, Jeff Bach, Marc Pollefeys, et al. City-scale landmark identification on mobile devices. In *CVPR*, 2011.
- [11] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, 2005.
- [12] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated Hate Speech Detection and the Problem of Offensive Language. *ICWSM*, 2017.
- [13] R Devon Hjelm, Karan Grewal, Phil Bachman, Alex Fedorov, Adam Trischler, Samuel Lavoie-Marchildon, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.
- [14] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pages 1422–1430, 2015.
- [15] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A Learned Representation for Artistic Style. *ICLR*, 2016.
- [16] Mai ElSherief, Shirin Nilizadeh, Dana Nguyen, Giovanni Vigna, and Elizabeth Belding. Peer to Peer Hate: Hate Speech Instigators and Their Targets. *ICWSM*, 2018.
- [17] Andrea Frome, Greg S Corrado, Jonathon Shlens, Samy Bengio Jeffrey Dean, Aurelio Ranzato, and Tomas Mikolov. DeViSE: A Deep Visual-Semantic Embedding Model. *NIPS*, 2013.
- [18] Chuang Gan, Hang Zhao, Peihao Chen, David Cox, and Antonio Torralba. Self-supervised moving vehicle tracking with stereo sound. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2019-Octob, pages 7052–7061, 2019.
- [19] Peng Gao, Pan Lu, Hongsheng Li, Shuang Li, Yikang Li, Steven Hoi, and Xiaogang Wang. Question-Guided Hybrid Convolution for Visual Question Answering. *ECCV*, 2018.
- [20] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A Neural Algorithm of Artistic Style. *arXiv*, 2015.
- [21] Lluís Gómez and Dimosthenis Karatzas. TextProposals: A text-specific selective search algorithm for word spotting in the wild, 2017.
- [22] Lluís Gomez, Yash Patel, Marçal Rusiñol, Dimosthenis Karatzas, and C. V. Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. *CVPR*, 2017.

- [23] Raul Gomez, Jaume Gibert, Lluís Gomez, and Dimosthenis Karatzas. Exploring Hate Speech Detection in Multimodal Publications. *WACV*, 2020.
- [24] Raul Gomez, Lluís Gomez, Jaume Gibert, and Dimosthenis Karatzas. Learning from #Barcelona Instagram data what Locals and Tourists post about its Neighbourhoods. *ECCV MULA Workshop*, 2018.
- [25] Raul Gomez, Lluís Gomez, Jaume Gibert, and Dimosthenis Karatzas. Learning to Learn from Web Data through Deep Semantic Embeddings. *ECCVW*, 2018.
- [26] Raul Gomez, Lluís Gomez, Jaume Gibert, and Dimosthenis Karatzas. Self-Supervised Learning from Web Data for Multimodal Retrieval. *Multi-Modal Scene Understanding*, 2019.
- [27] Raul Gomez, Baoguang Shi, Lluís Gomez, Lukas Numann, Andreas Veit, Jiri Matas, Serge Belongie, and Dimosthenis Karatzas. ICDAR2017 Robust Reading Challenge on COCO-Text. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pages 1435–1443. IEEE, nov 2017.
- [28] Albert Gordo, Jon Almazan, Naila Murray, and Florent Perronin. LEWIS: Latent embeddings for word images and their semantics. *ICCV*, 2015.
- [29] Albert Gordo and Diane Larlus. Beyond Instance-Level Image Retrieval: Leveraging Captions to Learn a Global Visual Representation for Semantic Retrieval. *CVPR*, 2017.
- [30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016.
- [31] Michael Herz and Peter Molnar. *The content and context of hate speech: Rethinking regulation and responses*. Cambridge University Press, Cambridge, 2012.
- [32] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv*, 2015.
- [33] Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. Analyzing labeled cyberbullying incidents on the instagram social network. *Lecture Notes in Computer Science*, 9471:49–66, 2015.
- [34] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv*, 2015.
- [35] Allan Jabri, Armand Joulin, and Laurens Van Der Maaten. Revisiting Visual Question Answering Baselines. Technical report, Facebook AI Research, 2016.
- [36] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading Text in the Wild with Convolutional Neural Networks. *International Journal of Computer Vision*, 2014.

- [37] Deng Jia, Dong Wei, Socher R, Li Li-Jia, Li Kai, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [38] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. *ECCV*, 2016.
- [39] Dimosthenis Karatzas, Lluís Gomez-Bigorda, Angelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, Faisal Shafait, Seiichi Uchida, and Ernest Valveny. ICDAR 2015 Competition on Robust Reading. *ICDAR*, 2015.
- [40] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluís Gomez I Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazàn Almazàn, and Lluís Pere De Las Heras. ICDAR 2013 Robust Reading Competition. *ICDAR*, 2013.
- [41] Andrej Karpathy and Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):664–676, 2017.
- [42] Lyndon Kennedy and Mor Naaman. Generating Diverse and Representative Image Search Results for Landmarks. *International Conference on World Wide Web*, 2008.
- [43] Lyndon Kennedy, Mor Naaman, Shane Ahern, Rahul Nair, and Tye Rattenbury. How Flickr Helps us Make Sense of the World: Context and Content in Community-Contributed Media Collections. *ACM International Conference on Multimedia*, 2007.
- [44] Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes. *arXiv*, may 2020.
- [45] Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models. *arXiv*, 2014.
- [46] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pages 1–9, 2012.
- [47] Ankita Kumar, Jean-Philippe Tardif, Roy Anati, and Kostas Daniilidis. Experiments on visual loop closing using vocabulary trees. In *CVPR Workshops*, 2008.
- [48] Ankan Kumar Bhunia, Ayan Kumar Bhunia, Prithaj Banerjee, Aishik Konwer, Abir Bhowmick, Partha Pratim Roy, and Umapada Pal. Word Level Font-to-Font Image Translation using Convolutional Recurrent Generative Adversarial Networks. *ICPR*, 2018.
- [49] Quoc V. Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. *NIPS*, 2014.

- [50] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, Jesse Berent, Abhinav Gupta, Rahul Sukthankar, and Luc Van Gool. WebVision Challenge: Visual Learning and Understanding With Web Data. *arXiv*, 2017.
- [51] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. WebVision Database: Visual Learning and Understanding from Web Data. *arXiv*, 2017.
- [52] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C Lawrence Zitnick, and Piotr Doll. Microsoft COCO: Common Objects in Context. Technical report, Microsoft, 2014.
- [53] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. *Lecture Notes in Computer Science*, 2014.
- [54] Jing Liu, Zechao Li, Jinhui Tang, Yu Jiang, and Hanqing Lu. Personalized geo-specific tag recommendation for photos on social websites. *IEEE Transactions on Multimedia*, 2014.
- [55] Xialei Liu and Andrew D Bagdanov. Leveraging Unlabeled Data for Crowd Counting by Learning to Rank. *CVPR*, 2018.
- [56] Yang Liu, Zhaowen Wang, Hailin Jin, and Ian Wassell. Synthetically Supervised Feature Learning for Scene Text Recognition. *ECCV*, 2018.
- [57] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition - CVPR '15*, pages 3431–3440, 2015.
- [58] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. In *International Journal of Computer Vision*, 2004.
- [59] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, Laurens Van Der, and Maaten Facebook. Exploring the Limits of Weakly Supervised Pretraining. *ECCV*, 2018.
- [60] Edgar Margffoy-Tuay, Juan C. Pérez, Emilio Botero, and Pablo Arbeláez. Dynamic Multimodal Instance Segmentation guided by natural language queries. *ECCV*, 2018.
- [61] U.-V. Marti and H. Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 2002.
- [62] Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *ICLR*, 2013.
- [63] Emily Moxley, Jim Kleban, and BS Manjunath. Spirittagger: a geo-aware tag suggestion tool mined from flickr. In *Proc. ACM ICMIR*, 2008.

- [64] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9910 LNCS, pages 69–84, 2016.
- [65] Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S. Corrado, and Jeffrey Dean. Zero-Shot Learning by Convex Combination of Semantic Embeddings. *NIPS*, 2013.
- [66] Neil O’Hare, Cathal Gurrin, Gareth J.F. Jones, and Alan F. Smeaton. Combination of content analysis and context features for digital photograph retrieval. In *European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology*, 2005.
- [67] Yash Patel, Lluís Gomez, Raul Gomez, Marçal Rusiñol, Dimosthenis Karatzas, and C. V. Jawahar. TextTopicNet - Self-Supervised Learning of Visual Features Through Embedding Images on Semantic Text Spaces. *arXiv*, 2018.
- [68] Yash Patel, Lluís Gomez, Marçal Rusiñol, and Dimosthenis Karatzas. Dynamic Lexicon Generation for Natural Scene Images. *ECCV*, 2016.
- [69] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. *EMNLP*, 2014.
- [70] Princeton University. WordNet, 2010.
- [71] Tye Rattenbury, Nathaniel Good, and Mor Naaman. Towards Automatic Extraction of Event and Place Semantics from Flickr Tags. *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.
- [72] Zhou Ren, Hailin Jin, Zhe Lin, Chen Fang, and Alan Yuille. Joint Image-Text Representation by Gaussian Visual-Semantic Embedding. *ACM Multimedia*, 2016.
- [73] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of Textual Phrases in Images by Reconstruction. Technical report, Max Planck Institute for Informatics, 2015.
- [74] Andrew Rouditchenko, Hang Zhao, Chuang Gan, Josh Mcdermott, and Antonio Torralba. Self-Supervised Audio-Visual Co-Segmentation. *ICASSP*, 2019.
- [75] Andrew Rouditchenko, Hang Zhao, Chuang Gan, Josh Mcdermott, and Antonio Torralba. Self-Supervised Segmentation and Source Separation on Videos. *arXiv*, 2019.
- [76] David.E Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning Representations by Back-propagating Errors. In *Nature*, 1986.
- [77] Benet Oriol Sabat, Cristian Canton Ferrer, and Xavier Giro-I-Nieto. Hate Speech in Pixels: Detection of Offensive Memes towards Automatic Moderation. *NeurIPS AI for Social Good WS*, 2019.

- [78] Haji Mohammad Saleem, Kelly P Dillon, Susan Benesch, and Derek Ruths. A Web of Hate: Tackling Hateful Speech in Online Social Spaces. *First Workshop on Text Analytics for Cybersecurity and Online Safety at LREC*, 2017.
- [79] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning Cross-Modal Embeddings for Cooking Recipes and Food Images. *CVPR*, 2017.
- [80] Anna Schmidt and Michael Wiegand. A Survey on Hate Speech Detection using Natural Language Processing. *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, 2017.
- [81] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 815–823, 2015.
- [82] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations*, 2015.
- [83] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 2014.
- [84] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015.
- [85] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, and Jonathon Shlens. Rethinking the Inception Architecture for Computer Vision. *CVPR*, 2016.
- [86] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: The New Data in Multimedia Research. *Communications of the ACM*, 2015.
- [87] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. *ICML*, 2016.
- [88] Aaron van den Oord DeepMind, Yazhe Li DeepMind, and Oriol Vinyals DeepMind. Representation Learning with Contrastive Predictive Coding. *arXiv*, 2018.
- [89] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images. *arXiv*, 2016.
- [90] Andreas Veit, Maximilian Nickel, Serge Belongie, and Laurens Van Der Maaten. Separating Self-Expression and Visual Content in Hashtag Supervision. In *CVPR*, 2018.

-
- [91] Nam Vo, Nathan Jacobs, and James Hays. Revisiting IM2GPS in the Deep Learning Era. *ICCV*, 2017.
- [92] Nam Vo, Lu Jiang, Chen Sun, Kevin Murphy, Li-Jia Li, Li Fei-Fei, and James Hays. Composing Text and Image for Image Retrieval - An Empirical Odyssey. *CVPR*, 2019.
- [93] Nam Vo, Lu Jiang, Chen Sun, Kevin Murphy, Li Jia Li, Li Fei-Fei, and James Hays. Composing text and image for image retrieval-An empirical odyssey. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 6432–6441, 2019.
- [94] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.
- [95] Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. Learning Two-Branch Neural Networks for Image-Text Matching Tasks. *CVPR*, 2017.
- [96] Liwei Wang, Yin Li, and Svetlana Lazebnik. Learning deep structure-preserving image-text embeddings. *CVPR*, 2016.
- [97] Liwei Wang, Yin Li, Svetlana Lazebnik, and Slazebni@illinois Edu. Learning Deep Structure-Preserving Image-Text Embeddings. Technical report, Illinois, 2016.
- [98] Weiyao Wang, Du Tran, and Matt Feiszli. What Makes Training Multi-Modal Classification Networks Hard? *CVPR*, 2020.
- [99] Baoyuan Wu, Weidong Chen, Peng Sun, Wei Liu, Bernard Ghanem, and Siwei Lyu. Tagging like Humans: Diverse and Distinct Image Annotation. *CVPR*, 2018.
- [100] Baoyuan Wu, Fan Jia, Wei Liu, and Bernard Ghanem. Diverse Image Annotation. Technical report, Unknown, 2017.
- [101] Lei Wu, Rong Jin, and Anil K Jain. Tag completion for image retrieval. *IEEE TPAMI*, 2012.
- [102] Yuxin Wu and Kaiming He. Group Normalization. *ECCV*, 2018.
- [103] Fan Yang, Xiaochang Peng, Gargi Ghosh, Reshef Shilon, Hao Ma, Eider Moore, and Goran Predovic. Exploring Deep Multimodal Fusion of Text and Photo for Hate Speech Classification. *Workshop on Abusive Language Online*, 2019.
- [104] Qiangpeng Yang, Hongsheng Jin, Jun Huang, Wei Lin, and Alibaba Group. Swap-Text: Image Based Texts Transfer in Scenes. In *CVPR*, 2020.
- [105] Jiaming Zhang, Shuhui Wang, and Qingming Huang. Location-Based Parallel Tag Completion for Geo-tagged Social Image Retrieval General Terms. *ACM Transactions on Intelligent Systems and Technology*, 2017.

-
- [106] Ziqi Zhang, David Robinson, and Jonathan Tepper. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. *Lecture Notes in Computer Science*, 2018.
- [107] Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. The Sound of Pixels. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11205 LNCS, pages 587–604, 2018.
- [108] Haoti Zhong, Hao Li, Anna Squicciarini, Sarah Rajtmajer, Christopher Griffin, David Miller, and Cornelia Caragea. Content-driven detection of cyberbullying on the instagram social network. *JCAI International Joint Conference on Artificial Intelligence*, 2016.
- [109] Bolei Zhou, Yuandong Tian, Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Simple Baseline for Visual Question Answering. *arXiv*, dec 2015.
- [110] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. EAST: An Efficient and Accurate Scene Text Detector. *CVPR*, 2017.

This work has been supported by the Programa de Doctorands Industrials de la Generalitat de Catalunya, ajut 2016 DI 84.

