Deep Inside Visual-Semantic Embeddings

Martin Engilberge

THÈSE DE DOCTORAT DE
SORBONNE UNIVERSITÉ

Spécialité
Informatique
École Doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Martin Engilberge

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITÉ

Sujet de la thèse :

Deep Inside Visual-Semantic Embeddings

Devant le jury composé de :

M. Yannis Avrithis	INRIA Rennes	Rapporteur
M. Nicolas Thome	CNAM – CEDRIC	Rapporteur
M. Patrick Gallinari	Sorbonne Université – LIP6	Examinateur
Mme. Diane Larlus	NAVER Labs Europe	Examinateur
M. Jean Ponce	INRIA Paris, ENS – WILLOW	Examinateur
M. Louis Chevallier	Interdigital	Encadrant (Invité)
M. Patrick Pérez	Valeo.ai	Co-directeur de thèse
M. Matthieu Cord	Sorbonne Université – LIP6	Directeur de thèse

Science isn't about WHY, it's about WHY NOT!

— Cave Johnson, 1958

ACKNOWLEDGMENTS

These past 3 years have been the most exciting years of my life so far. I'd like to think I grew a lot both as a person and as a researcher. While some moment were more intense than others I could always count on the support of my advisors, my friends and my family and I would like to thank all of them here.

I'm grateful to my thesis advisors who shared with me their unique view of research. Patrick, for his open mind as well as his thoroughness when it comes to scientific writing, that I hope, made me a better writer. Matthieu, for his constructive ideas and comments and its uncomparable skill when it comes to naming papers. Finally, Louis, for all the late evening discussion and for being an endless source of creative ideas.

I thank all the people I met at Technicolor/Interdigital as well as all the people from the MLIA research team. It was a pleasure to work alongside all of you.

I would like to thank all the friends I met during my time in Rennes: Dmitry^2, Juan, Charlotte, Jean and Matthieu, for all of our adventures and for easing my first contact with research. You played a huge part in my choice to do a PhD, and I'm grateful for that. Eloïse, for our weekly squash session/moral support. Xu, Lucas, Cédric, Claire-Helene, Frédéric, Gilles and Romain, for all the intense lunch discussions and banter.

Finally, I want to thank my family for their endless support. I wouldn't be writing these lines if it weren't for them.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AI	Artificial Intelligence
CAM	Class Activation Mapping
CNN	Convolutionnal Neural Network
DNN	Deep Neural Network
GAP	Global Average Pooling
GPU	Graphic Processing Unit
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
mAP	mean Average Precision
MLP	MultiLayer Perceptron
MSE	Mean Square Error
NLP	Natural Language Processing
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SRU	Simple Recurrent Unit
SVM	Support Vector Machine
VQA	Visual Question Answering
VSE	Visual Semantic Embedding
VSRN	Visual Semantic Reasoning Network
W2v	Word2vec

1

INTRODUCTION

Contents

1.1 Context

In today's society, Artificial Intelligence (AI) is ubiquitous. It is both used to completely automate low-level tasks like robotic industrial applications, as well as to provide assistance for higher-level tasks such as collision detection or health diagnosis. With a steady interest in AI, today's research aims at automating more and more these high-level tasks.

A huge part of this automation relies on the ability of AI agents to perceive, understand and reason on their environment. Computer vision focuses on this particular problem for the visual domain. While cameras produce very precise representation, often more accurate than the human eye, reducing these pixel representations from a large dimensional space to a high-level representation is a challenge.

Since 2012 and the breakthrough of deep learning on the ImageNet classification challenge, deep neural networks settled as the new standard for image representation. With the increasing amount of research brought by this breakthrough, further improvement on object detection and semantic segmentation quickly followed. This im-

provement of image representation also served as a starting point for more challenging applications involving semantic and reasoning such as image captioning, multimodal retrieval or visual question answering.

This progress was also backed up by the growth of the internet and the large amount of data created and uploaded online every day. Sorting this amount of data requires automated processes. It is also an opportunity to collect large quantity of data needed to train larger and larger machine learning models.

1.2 Computer vision, image and video understanding

With the goal of improving image and video understanding, the computer vision field is using increasingly difficult proxy tasks to challenge itself. Solving these tasks pushes forward visual representation.

In the past two decades tremendous progress has been made. Object recognition is one of the tasks that helped the computer vision field grow. It aims at identifying objects in an image. The main difficulty of this task lies in the diversity of objects. They vary in scales, shapes, and colors depending on the illumination and the viewpoint of the scene. For a given object, for example a chair, multiple types of chairs exist with large variations in their appearance. Creating a model able to generalize and detect any type of chairs requires to have a semantic understanding of the concept of chair *i.e.* an object, you can sit on. However since the only goal of classification is to classify an image for the presence or absence of an object it is possible to mostly rely on context and textures instead of truly grasping the concept behind the object. A wooden object in the living room is more likely to be a chair than a car; nonetheless we would like to be able to detect a car even if it is in the middle of a living room.

Nevertheless promising results were obtained on the classification task and the field moved toward more challenging tasks, extending simple classification with localization. For example, the goal of object detection is to detect and localize an object with a bounding box. Finer localization was also tackled trying to segment the object or even

its instance. Focusing on localization tasks helped improve visual representation with the need to capture high-level concepts while preserving low-level accuracy (pixel-wise).

1.3 Extending computer vision using semantic

With the growing performance of models on pure computer vision tasks and a first glimpse of their ability to capture semantics, it is a natural next step to strengthen the connection between the visual and semantic domains. This can be done by bridging the gap between visual and natural language tasks. The most direct example of such a task would be captioning, which aims at producing a short textual description corresponding to a given visual scene. In that form the textual caption can be seen as a high-level representation of the image, using natural language as representation encoding. The production of such captions requires a high understanding of the visual modality, including the detection of the objects in the scene, their attributes, and their relations with each other.

Another task connecting the visual and textual modality is Visual Question Answering (VQA), its goal is to answer any natural language question about any type of images. It requires high-level understanding of both visual and textual modality as well as advanced reasoning. Answering a question might require multiple steps of reasoning each involving different parts of the image as well as high-level concepts, object relationship and even common sense.

Lastly, and the approach of interest in this thesis, we will focus our attention on joint semantic and visual representation. With the availability of multimodal data, it is now possible to directly align the visual domain with the semantic domain. Given a set of visual data where each image is paired to a caption describing its content, joint semantic and visual representation aims at creating a common representation space, where images and texts are comparable. Using the properties of such space to measure similarity between elements of different modalities open up multimodal retrieval: using a natural language query describing a scene to retrieve the picture of the corresponding scene from a large database. While retrieval is the most obvious application using joint multimodal space, it can also serve as a basis for any application requiring visual and

semantic information such as captioning, visual grounding of text, textual guided image generation, object relationship detection.

1.4 Contribution

In this thesis, we aim to further advance image representation and understanding. Revolving around Visual Semantic Embedding (VSE) approaches, we explore different directions: First, we present relevant background in Chapter 2, covering images and textual representation and existing multimodal approaches. Then in Chapter 3 we propose novel architectures further improving retrieval capability of VSE. In Chapter 4 we extend VSE models to novel applications and leverage embedding models to visually ground semantic concept. Finally, in Chapter 5 we delve into the learning process and in particular the loss function.

This thesis is based on the work published in the following articles:

- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Finding beans in burgers: Deep semantic-visual embedding with localization." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3984–3993
- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "SoDeep: a Sorting Deep net to learn ranking loss surrogates." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10792–10801
- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Semantic-Visual Embedding with Distributed Self-Attention." In: *arXiv preprint*. 2020

1.5 Industrial context

This thesis has been completed within Technicolor and Interdigital. Technicolor is a French corporation that provides services and products for the communication, media and entertainment industries. Interdigital is a mobile technology research and development company which acquired Technicolor Research & Innovation Activity in 2019.

In the context of movie production, a tremendous amount of multimedia content is produced daily. One motivation behind this thesis is being able to process this content so it can be easily retrieved later. An effective retrieval system requires a fine-grained visual representation capturing discriminative features, it also needs to be intuitive to use. Recent Deep Learning systems based on visual-semantic embedding combine both of these properties. For Technicolor/Interdigital, further exploring multi-modality interaction is of high interest and has been studied in this thesis.

2

LITERATURE REVIEW

Contents

In this chapter we present a brief literature review of the machine learning field centered around Visual Semantic Embedding (VSE). After introducing the concept of statistical learning in Section 2.1, we will mostly focus on recent deep learning approaches.

VSE aims at building a vector space where visual and textual representation are comparable. With that in mind, we will first introduce existing strategies to represent visual and textual data. Then we will delve into existing multimodal approaches. Finally, we will explore existing localization methods within the multimodal context.

2.1 Statistical learning

In this section, we briefly present the concept of statistical learning. Starting from supervised learning, we will then introduce more complex objective functions and finish by introducing neural networks.

2.1.1 Supervised learning

Supervised learning aims at learning an approximation f_w parametrized by a set of coefficients w minimizing an empirical loss $\mathcal{L}$ with respect to a dataset $\mathcal{D} = \left\{(\mathbf{x}_i, \mathbf{y}_i)\right\}_{i=1}^{N}$ of aligned (input data, target annotation) pairs. The empirical loss $\mathcal{L}$ is often combined with a regularization penalty $\mathcal{R}$ to form the full objective function $\mathcal{J}$.

$$\begin{aligned} \widehat{w} &= \arg\min_{w} \mathcal{J}(w), \\ \mathcal{J}(w, \mathcal{D}) &= \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(\mathbf{y}_i, f_w\left(\mathbf{x}_i\right)\right) + \mathcal{R}(w). \end{aligned} \tag{2.1}$$

How to find the optimal $\widehat{w}$ largely depends on the type of function f used. For example, if f is a linear regression $\widehat{w}$ can be computed through a closed-form solution using the least square method. Sometimes there is no closed-form solution or it is intractable, for example neural networks are nonlinear functions. Nonetheless $\widehat{w}$ can be approximated using iterative gradient descent methods.

2.1.2 Loss functions

As shown in Equation 2.1 statistical learning aims at minimizing a loss function. The loss function formalizes the task we want to solve, it also measures the efficiency of the approximation. Multiple loss functions have been proposed for various types of tasks.

For example, when the target annotation $\mathbf{y}_i$ is quantitative (*e.g.* 2D coordinates) it is a regression problem. A common loss for this type of task is the Mean Square Error (MSE) defined as follows:

$$\mathcal{L}_{MSE}(w, \mathbf{x}_i, \mathbf{y}_i) = \left(f_w\left(\mathbf{x}_i\right) - \mathbf{y}_i\right)^2. \tag{2.2}$$

When the target annotation is categorical (*e.g.* presence of an object in an image) it is a classification problem. For classification the Cross-Entropy error is a common objective to minimize. In this context $f_w(\mathbf{x}_i)$ approximates a probability distribution over the C classes.

$$\mathcal{L}_{CE}(w, \mathbf{x}_i, y_i) = -\sum_{c=1}^{C} \mathbb{1}_{y_i=c} \left[\log\left(f_w(\mathbf{x}_i)\right)\right]. \tag{2.3}$$

Statistical learning is not always fully supervised, it can also be unsupervised or weakly supervised.

In unsupervised learning, the training data doesn't contain annotations $\mathbf{y}$ and the loss $\mathcal{L}$ can only be expressed with respect to $\mathbf{x}$.

In weakly supervised learning the provided annotation isn't exactly what we want the model to predict. For example, in weakly supervised localization $\mathbf{y}_i$ would be the class labels of objects present in the image $\mathbf{x}_i$ and the goals would be to learn a function f_w predicting the position of the object.

In this thesis we aim at learning VSE models and the supervision comes as aligned pairs of images and their textual descriptions. One of the goals is to solve a retrieval task. In other words given a query and a dataset VSE needs to produce a ranking while being trained only with pairs, making it close to being weakly supervised.

In that sense it shares similarities with metric and representation learning in particular with respect to objective function. Several methods have been proposed to learn such metrics. In pairwise approaches, [95] minimizes the distance within pairs of similar training examples with a constraint on the distance between dissimilar ones. This learning process has been extended to kernel functions as in [65]. Other methods consider triplets or quadruplets of images, which are easy to generate from classification training datasets, to express richer relative constraints among groups of similar and dissimilar examples [8, 31, 91].

2.1.3 Neural networks

In feedforward neural network, the input data is progressively transformed by going through an alternating sequence of projection layers and non-linear activation function

until a final projection is produced. A feedforward network with n layers can be formally represented as follows:

$$f_w(\boldsymbol{x}) = f_{w_n}\left(f_{w_{n-1}}\left(\dots f_{w_2}\left(f_{w_1}(\boldsymbol{x})\right)\right)\right). \tag{2.4}$$

Stacking a large number of such layers produces "deep" architectures which is the origin of the name Deep Neural Network (DNN) and deep learning. A DNN being differentiable, its parameters can be learned using gradient-based methods. Each parameters are updated using the gradient $\frac{\partial \mathcal{J}(\mathbf{w})}{\partial \mathbf{w}}$ of the objective function with respect to the parameters. The gradient for each parameter is computed using the back propagation algorithm [78]. The strength of each step taken in the gradient direction is weighted by a learning rate η resulting in the following update of the parameters:

$$\boldsymbol{w} = \boldsymbol{w} - \eta \frac{\partial \mathcal{J}(\boldsymbol{w})}{\partial \boldsymbol{w}}. \tag{2.5}$$

In practice computing the exact gradient is too expensive because it requires evaluating the model on every example of the dataset. To overcome this problem, methods such as Stochastic Gradient Descent (SGD) [73, 84] or Adam [50] estimate the gradient from randomly sampled subset of data called minibatch.

Learning deep learning models from scratch is a challenging task. To obtain a model generalizing well to unseen data the ratio between the number of parameters $|\boldsymbol{w}|$ and the amount of data samples N need to be balanced. Training very large neural networks such as VGG-19 (144 million parameters resulting in 19.6 billion FLOP) or ResNet-152 (60 million parameters resulting in 11.3 billion FLOP) requires a large amount of data.

Fortunately, it has been shown that the features learned by such networks can easily be reused on different tasks [99]. Making it common to use a neural network pretrained on a large dataset and to transfer it to a new task or dataset. It often only requires to train additional adaptation layers.

Pushing the adaptation one step further it is now quite common to "finetune" the parameters of pretrained network. In other word when using a large pretrained network in a new model, the weights of the large neural network can be updated in and end-to-end manner, usually with a smaller learning rate η in order to preserve the low-level processing already learned.

UNIVERSAL APPROXIMATOR While it is proven that multilayer feedforward networks are universal approximator [14, 40], architecture design is still empirically driven. Indeed the Universal Approximation capability of neural network doesn't consider the learnability properties of these networks. While multiple architectures could potentially reach the same optimal approximation given a proper combination of parameters, converging to this optimal solution in a very large parameter space is still a challenge.

Deep learning mostly relies on gradient descent optimization, since it is applied to non-convex functions, convergence to the global minima is not guaranteed. That is where the importance of neural network architecture comes in. Different architectures lead to different landscapes of the parameter space, and can improve the learnability in multiple ways: it can change convergence speed, stability, robustness to initialization, or help to converge to a better local minimum.

However using theory to model this characteristic is still an open problem and most of the approaches are only empirically evaluated. For this reason the deep learning field with its fast pace progress mostly adopted an empirical paradigm.

2.2 Mono-modal representation

Visual Semantic Embedding models aim to learn a joint visual and semantic representation. Before exploring multimodal interaction, let's first look at existing approaches to generate visual and textual representations.

2.2.1 Computer vision and image representation

The goal of computer vision is to capture high-level understanding from digital images. In this section we will quickly introduce traditional computer vision approaches and cover more extensively modern deep learning equivalent.

TRADITIONAL COMPUTER VISION Most computer vision tasks require to transform the low-level information of the visual domain to compact high-level vector representation. The main challenge resides in projecting the pixel space of images to a smaller vector space while preserving the content information. Multiple handcrafted methods

exist, such as appearance based methods representing an image using its edge, gradient or color histogram.

In the beginning of the 2000s dictionary learning method such as the bag of visual word models [64] were popular. Local features such as SIFT [62] were first extracted, then they were encoded as a function of a dictionary of visual words, and finally the visual code was aggregated to obtain a single vector representation. In the context of object classification one would typically use this single vector representation to separately train a classifier, be it a linear regression, a Support Vector Machine (SVM) [13] or a MultiLayer Perceptron (MLP).

In this framework classification is a two-step process: features are first extracted either by handcrafted method or learned with an unsupervised method and then a classifier is trained on the features.

COMPUTER VISION WITH DEEP LEARNING With deep learning models, it is now common to learn visual representation from scratch in an end-to-end manner. Low-level processing is learned at the same time as the higher-level reasoning. For example, in a Convolutionnal Neural Network (CNN) the first few layers are learned filters resembling Gabor filters and color blobs, while deeper layers capture more specialized features such as shapes or objects [54].

The convolution layer [56] was introduced to handle visual data, sharing the learnable weights by sliding a single kernel over the visual space, thus greatly reducing the total number of parameters, allowing the processing of images of any size, and making the model invariant to translation.

AlexNet [54] is a CNN that consists of 5 convolutional layers. It uses Rectified Linear Unit (ReLU) for non-linear activation function, and its final layer is a linear projection outputting the scores for the 1000 classes of the ImageNet classification challenge.

When it was proposed in 2012, it reduced the top-5 classification error by half with an accuracy of 15.3% percent compared to the previous 26% of non-deep approaches. The top-5 error drops down to 7.3% percent in the next year's challenge with the introduction of VGG [82] and its deeper architecture of 13 and 16 convolutional layers followed by 3 fully connected layers.

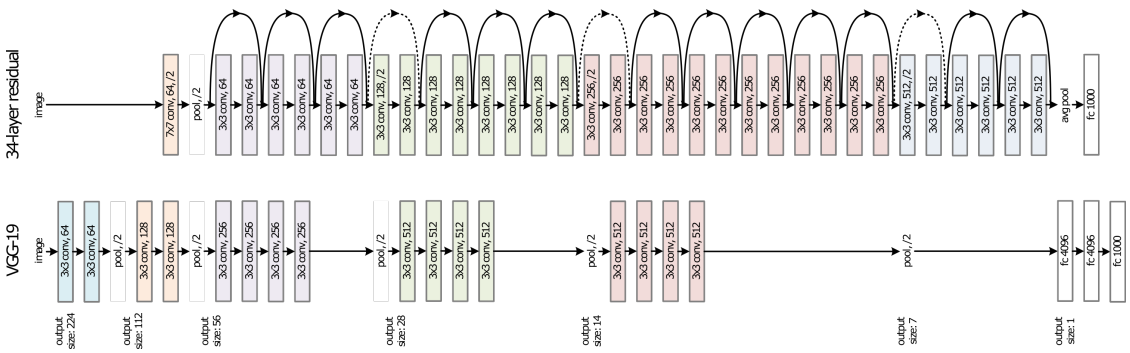


Figure 2.1: Resnet and VGG architecture. The illustration was taken from [37].

Finally, the next big improvement in the `ImageNet` challenge came from the introduction of the ResNet architecture by He et al. [37]: Assuming that it is easier to learn a residual mapping than an unreferenced mapping, this approach uses skip connections to force the network to learn a residual. It allowed for training of much deeper architectures with neural networks up to 200 layers. ResNet achieves a top-5 classification error of 3.6% on `ImageNet`. Diagram representation of VGG-19 and ResNet-34 architecture can be found in Figure 2.1.

In a residual network, the output of the n-th residual layer can be expressed as follows:

$$\mathbf{x}^{n+1} = f_{w_n}(\mathbf{x}^n) + \mathbf{x}^n, \tag{2.6}$$

with $\mathbf{x}^n$ and $\mathbf{x}^{n+1}$ the input and the output of the layer n, the layer only needs to learn the residual mapping $f_{w_n}(\mathbf{x}^n)$.

With the top-5 error on classification being extremely low, the field focus moved toward more difficult localization tasks. With respect to those tasks one key architecture innovation was the use of fully convolutional networks [61] by getting rid of the final linear layers. Fully convolutional networks are completely independent of the resolution of the input image, and preserve spatial information all the way until the final output.

Localization approaches will be more detailed in Section 2.5. We will only mention here a recent popular visual representation closely linked to the localization model, called bottom-up features [1]. They are produced by a faster R-CNN [75] trained on the object recognition task. Instead of producing features straight from a convolution network with a regular "grid-like" coverage of the entire image, they rely on a region proposal network extracting bounding boxes containing potential objects. Convolutional

features are then extracted for each of these bounding boxes to form the final image representation. It helps the image representation to focus on every important parts of the image and capture more details by using higher resolution on individual bounding boxes.

As architecture design is still a very active field, it would be impossible to make an exhaustive list of architectures. We chose to present the architecture most commonly used for feature extraction and image representation. In this thesis we will build on top of these approaches: we will mostly use ResNet and Faster R-CNN as the backbone of our image representation pipeline.

2.2.2 Computer vision datasets

While important progress has been made on deep learning architectures, it was only made possible by the apparition of large-scale databases of richly annotated images. We will introduce some of the major computer vision datasets. Figure 2.2 contains example images of three of them.

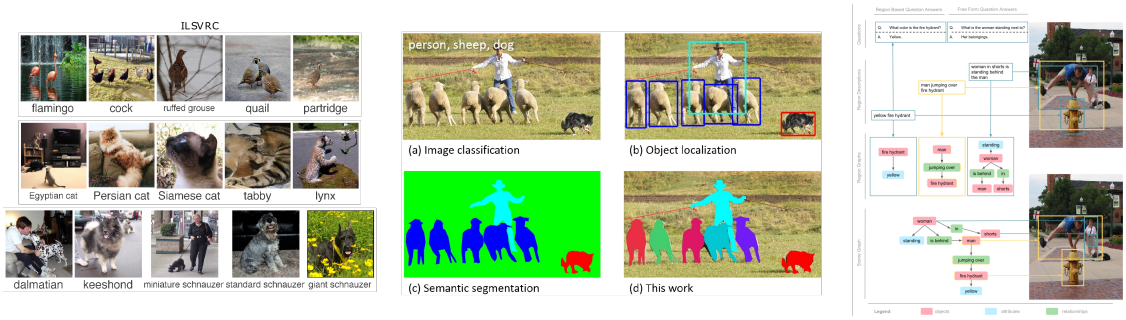


Figure 2.2: Examples of image and annotation from different datasets. left: `ImageNet`, center: `MS-COCO`, right: `Visual Genome`. The illustrations are taken from [79], [60], and [53].

- `ImageNet` propose by [16] consists of 14.2 million images, and more than 20000 classes. However the ImageNet Large-Scale Visual Recognition Challenge only uses a subset of 1.2 million images and 1000 classes. The annotation consists of one multi-class label per image for image classification and also contains bounding boxes for object detection. It consists mainly of simple scenes with usually a focus on a single object.

- `MS-COCO`[60] contains 120 000 images, of various types of scene with a higher complexity than `ImageNet`. The annotation is also richer but covers a smaller number of classes. It has 80 classes, which are annotated by bounding box, segmentation masks, and instance segmentation mask. Every image is also annotated with 5 short textual captions describing the scene.
- `Flickr-30K`[100] contains 30 000 images. Similarly to `MS-COCO` the annotation consists of 5 short textual captions describing the scene.
- `Visual Genome`[53] contains 108, 000 images. It shares some images with `MS-COCO` and extends its the annotation with visual questions and answers, object attributes and relationships between objects.

Our work leverages all of the datasets listed above. `ImageNet` is used for the pre-training of the visual representation while `Flickr-30K` and `MS-COCO` are used as direct supervision to align the visual and textual domain. Finally, `Visual Genome` is only used for the evaluation of the localization tasks.

2.2.3 Natural language processing and text representation

Natural language processing shares a similar goal with computer vision: Capturing high-level understanding of digital text. The early days of Natural Language Processing (NLP) adopted a rule-based paradigm, using hand-written grammar and heuristic rules [93]. In this section we will focus on the recent statistical learning paradigm, and the use of deep learning model to derive meaningful vector representations of any given text.

As opposed to computer vision, in language, a single word already encodes high-level semantic while a single pixel is meaningless. Text also has a sequential structure: in a sentence the order of the words carries important information. Therefore the methods to compute textual representation need to take these differences into account and change from the ones used in computer vision.

WORD AND SENTENCE REPRESENTATION Learning textual representation is often unsupervised, and only rely on large corpora of texts with no extra annotation. Word

and sentence representations are learned using sequence modeling models trained to predict the context of a given word/sentence.

For example, Word2vec (W2V) word representation proposed by Mikolov et al. [66] uses the skip-gram architecture. With the intuition that words used in similar contexts should have similar representations, given the representation of a word the model will aim at predicting the surrounding words in a text. After seeing the same word in many different contexts, the model will converge toward a representation of the word that captures some of its original semantic.

Similarly, the sentence representation model skip-thought was introduced by Kiros et al. [52]. Using an encoder decoder architecture: A sentence is encoded to a representation space and then two decoders will try to predict the previous and following sentence of the original text.

Text processing pipelines often consist of multiple blocks that can be pre-trained independently. Word representations are used as the basis for sentence representations, which can later be used to derive large document representations. This hierarchical structure allows easier transfer learning with the possibility to reuse the each block and avoid the need for very large corpora of annotated data for every task. Often the pre-training of this block is done in an unsupervised manner only using context in which a word or a sentence is present.

SEQUENCE MODELING MODELS Most modern language representations are learned using sequence modeling models such as Recurrent Neural Network (RNN). An RNN is a form of neural network design to process sequences. Using an internal state, it can iterate over a sequence and update its internal representation to capture sequential patterns and sequence representations.

Given a sentence represented as a sequence of words x of length T, x^t corresponds to the vector representation of t-th word of x. The RNN computes the representation of the sentence as follows:

$$\mathbf{h}^T = f_w(\mathbf{x}^T, \mathbf{h}^{T-1}), \tag{2.7}$$

where $\mathbf{h}^T$ coresponds to the hidden state of the RNN at time step T. To compute it, we first need to iteratively compute the hidden states at the previous time step $(\mathbf{h}^{T-1}, \mathbf{h}^{T-2}, ..., \mathbf{h}^1, \mathbf{h}^0)$. To compute $\mathbf{h}^0$ we will need the initial hidden state $\mathbf{h}^{-1}$ which can be randomly sampled or initialized to a specific value depending on the task.

Long Short-Term Memory (LSTM) is a type of RNN introduced by Hochreiter and Schmidhuber [39] and is largely adopted to process sequential data. LSTM relies on an internal state to preserve the current representation of a sequence. Using three sigmoid gates, LSTM is able to control the flow of information inside the network. Using this combination of gate the model decides how its internal state is updated with respect to its current input and its previous state.

Later a simplified version of LSTM called Gated Recurrent Units (GRUs) was proposed by Cho et al. [9]. GRUs has fewer parameters since it does not have an output gate. They have been found to have similar performances as LSTM on certain tasks. Figure 2.3 detail the inner working of GRU with its two sigmoid gates z and r. The computation of the gates and the internal state is defined as follows:

$$\begin{aligned}
\mathbf{r}^t &= \sigma\left(\mathbf{W}_r\mathbf{x}^t + \mathbf{U}_r\mathbf{h}^{t-1}\right), \\
\mathbf{z}^t &= \sigma\left(\mathbf{W}_z\mathbf{x}^t + \mathbf{U}_z\mathbf{h}^{t-1}\right), \\
\tilde{\mathbf{h}}^t &= \phi\left(\mathbf{W}\mathbf{x}^t + \mathbf{U}\left(\mathbf{r}^t \odot \mathbf{h}^{t-1}\right)\right), \\
\mathbf{h}^t &= \mathbf{z}^t\mathbf{h}^{t-1} + \left(1 - \mathbf{z}^t\right)\tilde{\mathbf{h}}^t,
\end{aligned} \tag{2.8}$$

with $\mathbf{x}^t$ and $\mathbf{h}^{t-1}$ the input and the previous hidden state, respectively. $\boldsymbol{W}_r$ and $\boldsymbol{U}_r$ are weight matrices which are learned.

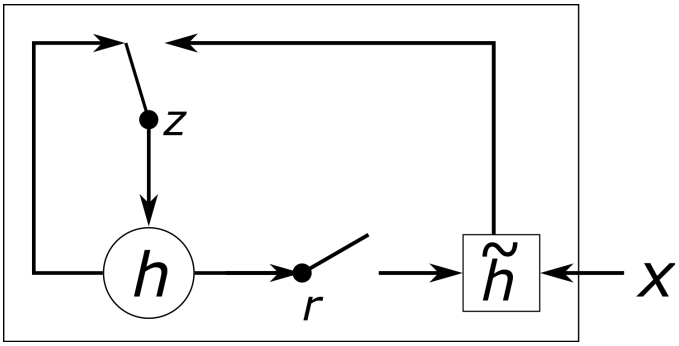


Figure 2.3: Illustration of the GRU showing the flow of information between input/output and the control gates. $\mathbf{h}$ is the hidden state, $\mathbf{r}$ the reset gate, $\mathbf{z}$ the memory gate and $\mathbf{h}^t$ the candidate hidden state. The illustration is taken from [9].

Finally Simple Recurrent Unit (SRU) further simplifies the GRU architecture by removing time dependency to allow for a better parallelization of multi-layer RNN. SRU is defined as follows:

$$\begin{aligned}
\mathbf{r}^t &= \sigma\left(\mathbf{W}_r\mathbf{x}^t + \mathbf{U}_r \odot \tilde{\mathbf{h}}^{t-1}\right), \\
\mathbf{z}^t &= \sigma\left(\mathbf{W}_z\mathbf{x}^t + \mathbf{U}_z \odot \tilde{\mathbf{h}}^{t-1}\right), \\
\tilde{\mathbf{h}}^t &= \mathbf{r}^t \odot \tilde{\mathbf{h}}^{t-1} + \left(1 - \mathbf{r}^t\right) \odot \left(\mathbf{W}\mathbf{x}^t\right), \\
\mathbf{h}^t &= \mathbf{z}^t \odot \tilde{\mathbf{h}}^t + \left(1 - \mathbf{z}^t\right) \odot \mathbf{x}^t.
\end{aligned} \tag{2.9}$$

When comparing Equation 2.8 of GRU and Equation 2.9 of SRU, both are similar and only use two sigmoid gates. The main benefit of the SRU is the use of pointwise multiplication in the computation of $\tilde{\mathbf{h}}^t$ making the dimension of the state vector independent therefore more easily parallelizable. Indeed it allows for the parallelization of the computation across the dimension without having to wait for the computation of the entire $\tilde{\mathbf{h}}^{t-1}$.

Recently the introduction of the transformers architecture by Vaswani et al. [86] opened the possibility to use feedforward networks for textual processing. Transformers heavily rely on self-attention mechanisms. In an effort to preserve sequential information, positional encoding is summed to the input textual sequence.

2.3 Multi-modal representation

With visual and textual representation having a similar goal of finding a rich representation, we have seen the very different approaches required to represent the specificity of each modality efficiently. The textual modality has a strong structure and limited vocabulary, while the visual modality with its very large input space is extremely diverse.

Combining visual and textual modality is key to solve certain tasks and can generally help create better mono-modal representation. Visual information can help disambiguate textual representation, while the textual representation can add semantic and common sense knowledge to the visual representation.

Indeed by creating interactions between visual and textual concept when learning both representation it can help CNN reduce their bias toward texture [32] and help

them take advantage of context and relation between objects. On the other hand, grounding language in the visual world might help language representation with ambiguities and world modeling: Associating objects more easily with its common attributes disambiguating relationships and references.

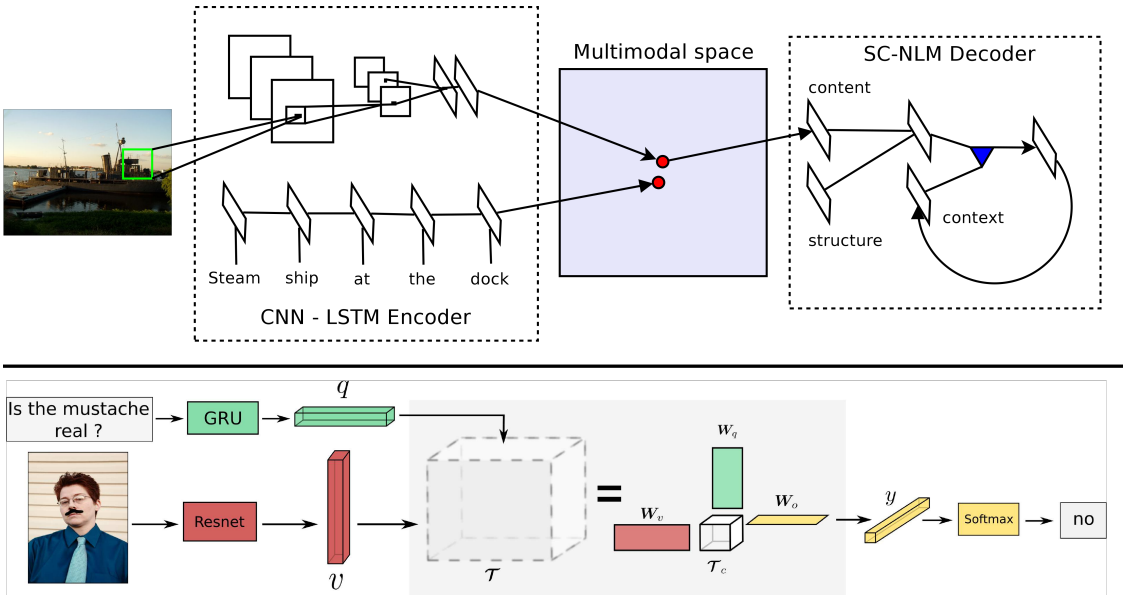


Figure 2.4: The top diagram illustrates the multimodal VSE with a visual and textual encoder projecting to a common space, a textual decoder starting from that space is then used for the captioning task. The bottom diagram represents a VQA model relying on multimodal fusion to create a visual-semantic representation used for the classification of the answers. The illustrations are taken from [51] and [4].

Following the notation from Section 2.1, visual and textual modality dataset could be represented as $\mathcal{D} = \{(\boldsymbol{I}_i, \boldsymbol{S}_i, \mathbf{y}_i)\}_{i=1}^{N}$ consisting of triplets of images, sentences and annotations. Depending on the application and dataset the annotation $\mathbf{y}$ is not always present. For example, in VQA the triplet would consists of an image $\boldsymbol{I}$, a question $\boldsymbol{S}$ in natural language and the class of the answer $\mathbf{y} \in \mathbb{R}^{d_o}$. In cross-modal retrieval, however, the data would only consists of pairs $(\boldsymbol{I}, \boldsymbol{S})$ of images and their textual descriptions.

Using models introduced in Section 2.2 the image $\boldsymbol{I}$ and the sentence $\boldsymbol{S}$ will be transformed to a vector representation $\mathbf{s} = f_{w_t}(\boldsymbol{S}) \in \mathbb{R}^{d_t}$ and $\mathbf{i} = f_{w_v}(\boldsymbol{i}) \in \mathbb{R}^{d_v}$ respectively.

Multiple ways to leverage multimodality have been proposed, we will present two of them in this section. The first one is modality fusion where both modalities are merged

to form a single vector representation, this approach is often used in VQA. The second approach is VSE where each modality is kept separate and is projected to a common embedding space where an alignment between the two modalities is enforced. Most common applications of VSE are image captioning and cross-modal retrieval. The fusion and embedding approaches are illustrated in Figure 2.4.

2.3.1 Multimodal fusion

Concatenation is one of the simplest forms of multimodal fusion. It was first used in Ren, Kiros, and Zemel [74]. After the concatenation $[\mathbf{i}, \mathbf{s}]$ of the image $\mathbf{i}$ and question $\mathbf{s}$ representations, an LSTM parametrized by weights $\mathbf{w}$ is used to predict the answer $\hat{\mathbf{y}} = f_w([\mathbf{i}, \mathbf{s}])$.

More recent VQA models are based on bilinear fusion. For example, in [4] the fusion is done with a bilinear model defined by a third-order tensor $\mathcal{T} \in \mathbb{R}^{d_t \times d_v \times d_o}$. The fusion is the expressed as:

$$\hat{\mathbf{y}} = f_{\mathcal{T}}(\mathbf{i}, \mathbf{s}) = (\mathcal{T} \times_1 \mathbf{s}) \times_2 \mathbf{i}, \tag{2.10}$$

where $\times_1$ denoted the i-mode product between tensors.

Bilinear fusion can capture stronger interactions between visual and textual features than concatenation. Indeed it computes all possible pairwise products for all the dimension pairs in $\mathbf{s}$ and $\mathbf{i}$.

2.3.2 Visual semantic embeddings

Another approach to leverage multi-modality is Visual Semantic Embedding, by aligning the representation of both modalities in a single space it opens multiple possibilities. The goal when training a VSE is to obtain a space where geometric distance can be interpreted as semantic similarity.

The Canonical Correlation Analysis (CCA) method is certainly one of the first techniques to align two views of heterogeneous data in a common space [41]. Linear projections defined on both sides are optimized in order to maximize the cross correlation. Recently, non-linear extensions using kernel (KCCA [55]) or deep net (DCCA [2]) have been proposed. [88] exploits DCCA strategies for image-text embeddings, while [98]

points out some limitations of this approach in terms of optimization complexity and overfitting and proposes ways to partially correct them. [22] proposes some CCA-based constraint regularization to jointly train two deep nets passing from one view to the other (text/image).

When considering the specific problem of embedding jointly images and labels (classification context), [30, 92] train models that combine a linear mapping of image features into the joint embedding space with an embedding vector for each possible class label. Approaches for the more advanced task of textual image description (captioning) often relies on an encoder/decoder architecture where the encoder consists of a joint embedding [47, 51]. Other works focus on the sole building of such a joint embedding, to perform image-text matching and cross-modal retrieval [27, 30, 63, 80].

Our work stems from this latter class. We aim at generating a joint embedding space that offers rich descriptors for both images and texts. We adopt the contrastive triplet loss that follows the margin-based principle to separate the positive pairs from the negative ones with at least a fixed margin [51]. The training strategy with stochastic gradient descent has to be carefully adapted to the cross-modality of the triplets. Following [27], we resort to batch-based hard mining.

When training a VSE, the visual and textual representation are projected to the common embedding space. Let's call $\hat{\mathbf{i}}$ and $\hat{\mathbf{s}}$ the representation in this new space. Both $\hat{\mathbf{i}}$ and $\hat{\mathbf{s}}$ belong to the same space of dimension $\mathbb{R}^d$. Multiple types of geometric distances can be used to measure similarity in this space, the most common one is cosine similarity define as follows:

$$c(\hat{\mathbf{s}}, \hat{\mathbf{i}}) = \frac{\langle \hat{\mathbf{s}}, \hat{\mathbf{i}} \rangle}{\|\hat{\mathbf{s}}\| \, \|\hat{\mathbf{i}}\|}. \tag{2.11}$$

Using cosine similarity has multiple benefits, first it is bounded in $[-1, 1]$. Secondly it has a low complexity: when working with unit vectors, the cosine similarity is equivalent to the dot product.

Once learned VSE can be used to measure similarity between any element of the embedding, this property is commonly used for retrieval: scoring every element of the database against a query only requires to measure the distance between them.

Taking advantage of the structured vector space, it is common to train decoders to solve various types of tasks. It is possible to learn decoders to do paraphrase, image captioning or image generation from text [35].

A multimodal space also displays interesting properties. Indeed geometric operations in the embedding can be interpreted as semantic operations. Similarly to word embeddings [67], it has been shown that the geometric translation in a VSE captures analogy [51]. For example, the representation of a red car $\mathbf{i}_{red-car}$ can be approximated using the representation of a blue car $\mathbf{i}_{blue-car}$ and the representations of the words "red" and "blue" $\mathbf{s}_{red}$, $\mathbf{s}_{blue}$ as follows:

$$\mathbf{i}_{red-car} \approx \mathbf{i}_{blue-car} - \mathbf{s}_{blue} + \mathbf{s}_{red}. \tag{2.12}$$

Our work will focus on VSE. Compared to other multimodal approaches, having a common embedding space leaves multiple opportunities to expand upon. In the context of cross-modal retrieval, VSE has one major benefit compared to multimodal fusion: its efficiency. Indeed once the representation of the images and sentences have been calculated computing the score between a query and a database can be done in linear time. Whereas with fusion, the fusion mechanism (often costly) needs to be recomputed for every pairs of query, element of the database.

2.4 Attention mechanism

Inspired from biological mechanisms [44], attention modules have become widespread across neural network architectures. With their ability to focus on the most relevant part of data, they are key in fully leveraging the large features produced by modern models.

Visual attention relies on scoring the different regions of an image. The attention weights can then be used to select the most relevant regions [97] or to generate a new representation taking into account the contribution of each region [1]. Figure 2.5

shows an example of a VQA architecture using visual attention guided using textual representation.

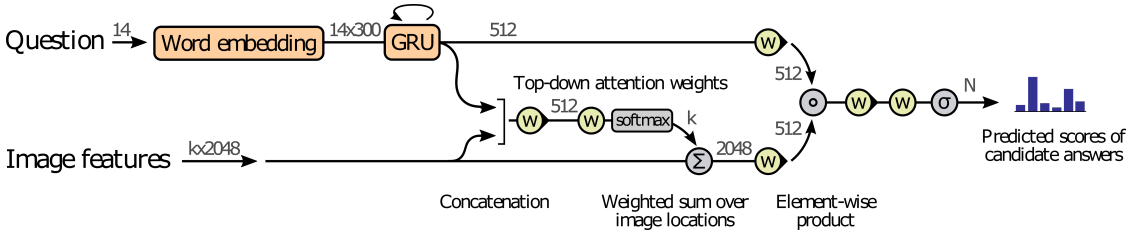


Figure 2.5: Example of visual attention in a VQA model. The text representation coming from a GRU is used as context to guide the visual attention. The illustration was taken from [1].

In the case of image-text matching, multiple forms of attention have been used. Some models only rely on self-attention [42, 59, 70], that is the attention scores for a modality are computed using only the context of that modality. For example, in the Dual Attention Network (DAN) [70], soft attention mechanisms relying on feed forward network and SoftMax function are learned for the visual and the textual pipeline respectively. Visual Semantic Reasoning Network (VSRN) [59] does region relationship reasoning using graph convolutions, which can be seen as a complex form of visual attention.

Other approaches bridge both modalities through the use of cross-modal attention [57]. These approaches tend to be more costly since the attention scores need to be computed for all the elements in the database when a new element is added.

2.5 Localization

In this thesis we aim to extend VSE toward localization. In that regards we will first present fully and weakly supervised localization models, then we will follow with existing multimodal approaches.

SUPERVISED LOCALIZATION Supervised localization methods aim at predicting the position and the class of objects in an image. Using ground truth localization from datasets such as `ImageNet`, `MS-COCO`, or `Visual Genome`, models can be trained to predict bounding boxes [34] or segmentation mask [61] precisely grounding predefined concepts.

Approaches producing bounding box localization, mostly rely on predicting candidate regions potentially containing an object and later classifying the presence of specific objects in these regions. One type of such approaches is called Faster R-CNN and was proposed by Ren et al. [75]. This model improves on previous approaches [33, 34] by incorporating the region proposal mechanism to the classification model, making it end-to-end trainable. The architecture diagram of Faster R-CNN can be seen in Figure 2.6.

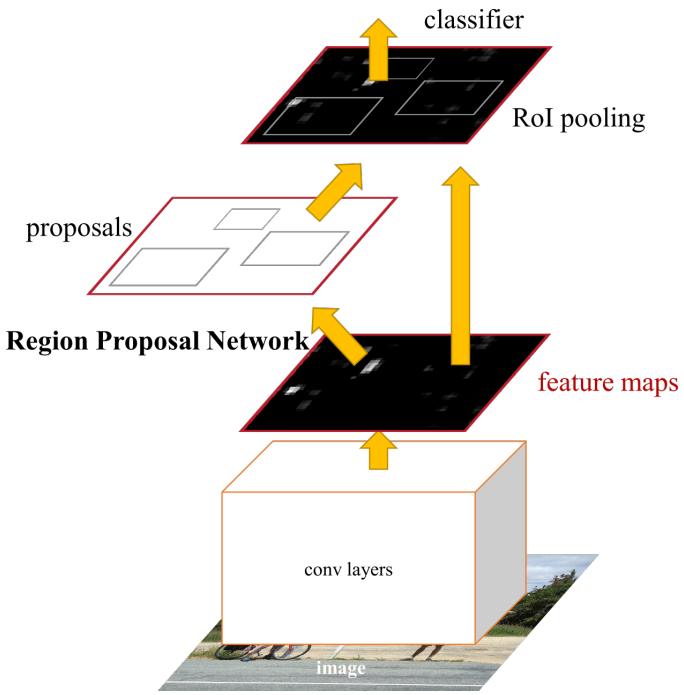


Figure 2.6: Faster R-CNN efficiently shares the convolutionnal feature maps between The Region Proposal Network and region classification. The illustration was taken from [75].

This model was later shown to produce rich visual features [1]. The spatialization of these features through bounding box paired with convolutional features makes them good candidate to transfer them to task that can leverage localization such as VQA or captionning. Overall, this type of feature performs better as global image representation than previous single convolutional representation.

WEAKLY SUPERVISED LOCALIZATION The rich annotations required for supervised learning rapidly become costly to obtain, making the development of weakly supervised approaches appealing. A number of weakly supervised object localization approaches extrapolate localization features while training an image classifier, *e.g.*[15, 20, 102]. The

main strategy consists in using a fully convolutional deep architecture that postpones the spatial aggregation (pooling) at the very last layer of the net. It can be used both for classification and for object detection. The Class Activation Mapping (CAM) weakly supervised approach [102] is illustrated in Figure 2.7.

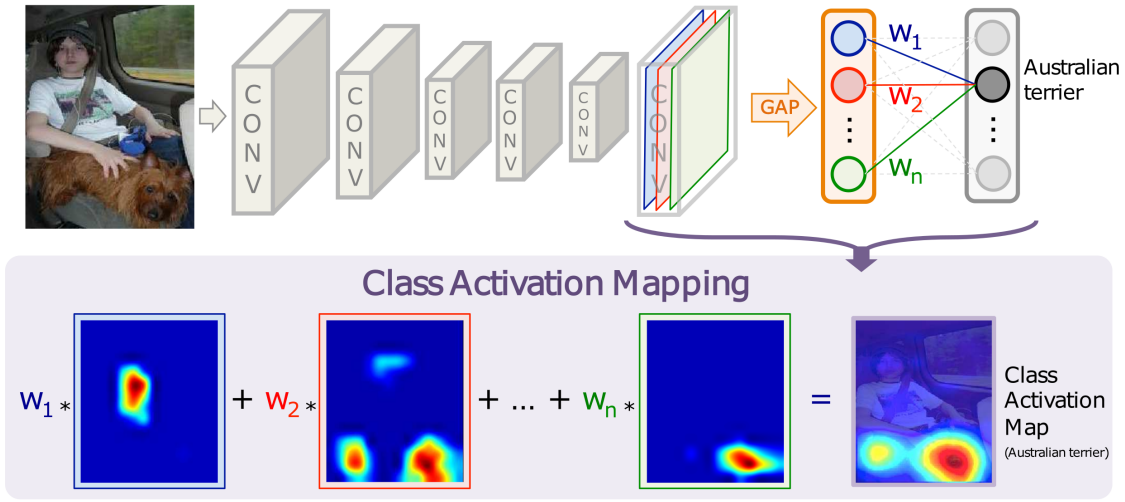


Figure 2.7: Global Average Pooling (GAP) aggregates 2d convolutionnal activation to a single value, making architecture fully convolutionnal and allowing weakly supervised localization through CAM. The illustration was taken from [102].

CROSS-MODAL EMBEDDING AND LOCALIZATION Existing works that combine localization and multimodal embedding rely on a two-step process. First, regions are extracted either by a dedicated model, *e.g.*, EdgeBox in [89], or by a module in the architecture. Then the embedding space is used to measure the similarity between these regions and textual data. [47, 72] use this approach on the dense captioning task to produce region annotations. It is also used for phrase localization by [89] where the region with the highest similarity with the phrase is picked. To address this specific problem of phrase grounding, Xiao *et al.* [94] proposed to learn jointly a similarity score and an attention mask. The model is trained using a structural loss, leveraging the syntactic structure of the textual data to enforce corresponding structure in the attention mask.

2.6 Positionning

This thesis revolves around Visual Semantic Embedding. We aim at extending existing approaches following three intertwined axes: Application, architecture and learning methods. A visual representation of our objectives can be found in Figure 2.8.

APPLICATION Extending existing approaches, we aim at improving VSE performance on cross-modal retrieval. In parallel, we would like to extend VSE localization ability, with a focus on visual grounding of text. While existing approaches use VSE to score regions and derive localization, [47, 87] we explore a more direct relation between the embedding space and spatial information coming from the visual representation avoiding the need of objects/region detectors trained using fully supervised annotation.

ARCHITECTURE We build on top of the two-branch architecture proposed by [51]. We first propose to improve the existing architecture by combining latest visual and textual representation such as ResNet, Faster R-CNN and SRU.

With localization in mind, we use selective spatial pooling [20] to learn a weakly supervised visual representation taking advantage of the new fully convolutional CNNs to preserve spatial information until the joint embedding space.

Departing from complex reasoning methods [57, 59], we propose to use distributed self-attention to fully take advantage of the spatialized region features at a very low computational cost.

LEARNING METHODS Finally, we aim at improving the learning mechanism used to train VSE. There is a strong connection between learning and architecture choice. With the current two-branch model, the cosine similarity is commonly used in a contrastive triplet loss to enforce alignment in the embedding space [51]. Building on this work, we explore hard negative mining strategies for the contrastive triplet loss.

Taking inspiration from the cross-modal retrieval use of VSE, we delve into a new method to learn a differentiable approximation of ranking metrics and use it to train not only VSE but any rank-based model such as ordinal regression.

Lastly, we will discuss the different learning strategies used along the way and their implication with respect to both application and architecture. It includes hyperparameter

tuning, training by part, pre-training, relation between image resolution, impact of batch size, training time, etc.

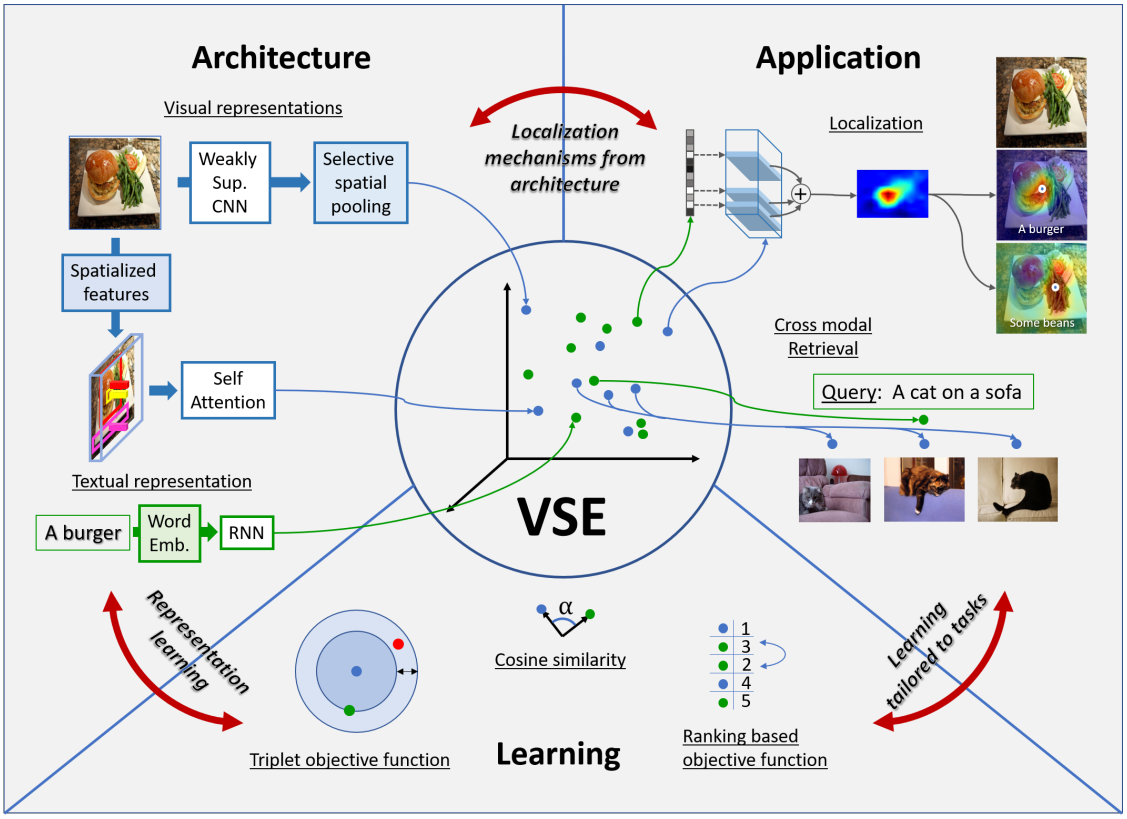


Figure 2.8: **Positioning and objectives** visualization of the three development axes: architecture in Chapter 3, application in Chapter 4 and learning in Chapter 5.

3

VISUAL SEMANTIC EMBEDDING

Contents

Chapter abstract

Several works have proposed to learn a two-path neural network that maps images and texts, respectively, to a same shared Euclidean space where geometry captures useful semantic relationships. Such a multimodal embedding can be trained and used for various tasks, notably image captioning. In the present chapter, we introduce two new architectures of this type. More specifically, we propose two different types of visual path tailored to take advantage of their respective visual features: The first one is based on ResNet leverages

space-aware pooling mechanisms; the second one relies on distributed self-attention to leverage at best localized bottom-up features. Once both are combined with a textual path which is jointly trained from scratch, our semantic-visual embeddings offer versatile models. Trained under the supervision of captioned images, they yield competitive performance on cross-modal retrieval.

The work in this chapter has led to two conference papers, one published and one under review:

- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Finding beans in burgers: Deep semantic-visual embedding with localization." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3984–3993
- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Semantic-Visual Embedding with Distributed Self-Attention." In: *arXiv preprint*. 2020

3.1 Introduction

Text and image understanding is progressing fast thanks to the ability of artificial neural nets to learn, with or without supervision, powerful distributed representations of input data. At runtime, such nets *embed* data into high-dimensional feature spaces where semantic relationships are geometrically captured and can be exploited to accomplish various tasks. Off-the-shelf already trained nets are now routinely used to extract versatile deep features from images which can be used for recognition or editing tasks, or to turn words and sentences into vectorial representations that can be mathematically analyzed and manipulated. Detailed presentation of the leading visual and textual representation models can be found in Section 2.2.

Recent works presented in Section 2.3 have demonstrated how such deep representations of images and texts can be jointly leveraged to build VSEs [30, 47, 51, 76]. The ability to map natural images and texts in a shared representation space where geometry (distances and directions) might be interpreted is a powerful unifying paradigm. Not only does it permit revisiting visual recognition and captioning tasks, but it also opens up new usages, such as cross-modal content search or generation and zero-shot

learning. One popular approach to semantic-visual joint embedding is to connect two mono-modal paths with one or multiple fully connected layers [3, 27, 47, 51, 89]: A visual path based on a pre-trained CNN and a text path based on a pre-trained RNN operating on a given word embedding. Using aligned text-image data, such as images with multiple captions from `MS-COCO` dataset [60], final mapping layers can be trained, along with the optional fine-tuning of the two branches.

Building on this line of research, we investigate two orthogonal methods to represent images.

In our first approach named BEAN[1] we introduce new pooling mechanisms in the visual path. Inspired by recent work on weakly supervised object localization [20, 102], we propose in particular to leverage selective spatial pooling with negative evidence proposed in [20] to improve visual feature extraction without resorting, *e.g.*, to expensive region proposal strategies. Without the use of spatialized supervision, this architecture can be trained and fine-tuned in an end-to-end manner using gradient-based optimization and can also produce weakly supervised localization. In this chapter we focus on cross-modal retrieval, localization mechanisms will be detailed later in Chapter 4.

In our second approach, we propose a visual pipeline based on a new visual self-attention mechanism, called SMILE for SeMantic embeddIng with seLf attEntion. SMILE aims at learning how the multi-object-region features of an image affect each other. We learn this dependency using a single neural layer that provides a simple yet powerful attention representation distributed across regions and feature channels. As opposed to BEAN, SMILE use pre-extracted features from a faster R-CNN model pretrained with spatial supervision. The feature extraction being costlier, it is done offline and prevents further fine-tuning of the feature extractor.

The proposed modifications to current approaches, along with additional design and training specifics, lead to two new systems whose performance is assessed on cross-modal matching, effectively composed of two symmetric sub-tasks: Retrieving captions from query images and vice versa.

1 Named BEAN after the title of the article where it was first introduced: "Finding beans in burgers: Deep semantic-visual embedding with localization" [23].

The rest of this chapter is organized as follows: Section 3.2 is dedicated to the presentation of our own systems, which couples selective spatial pooling and distributed self attention with recent architectures and which relies on a triplet ranking loss based on hard negatives. More details on the systems and their training are reported in Section 3.3, along with various experiments. Our systems obtain competitive results on the cross-modal retrieval task. Finally, extensive ablation studies are conducted in Section 3.4.

3.2 Visual Semantic Embedding

The overall structure of the proposed approach, shown in Figure 3.1, follows the dual path encoding architecture of Kiros, Salakhutdinov, and Zemel [51] visible in Figure 2.4. We first explain its specifics before turning to its training with a cross-modal triplet ranking loss.

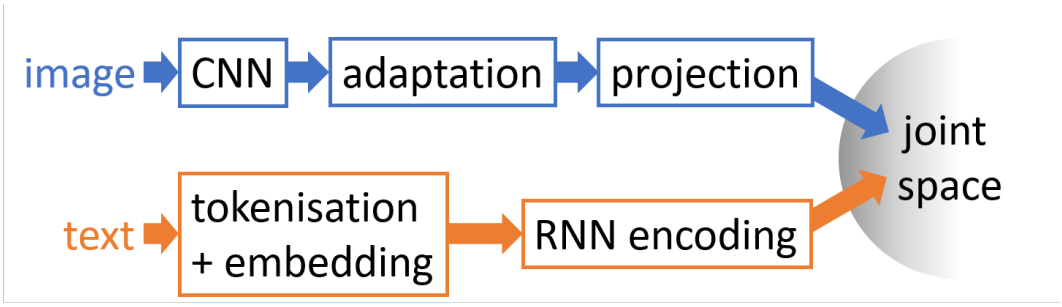


Figure 3.1: **Two-path multimodal embedding architecture**. Images of arbitrary size and texts of arbitrary length pass through dedicated neural networks to be mapped into a shared representation vector space. The visual path (blue) is composed of a convolutional neural network, followed by adaptation layers, which aggregates previous feature maps into a vector and a final projection to the final output space; The textual path (orange) is composed of a recurrent net running on sequences of text tokens individually embedded with an off-the-shelf map (word2vec in experiments).

In this section we introduce two new architectures. Both share the same textual pipeline and only differ in their visual path. One is spatially supervised: based on a faster R-CNN object detector (See Figure 2.6), and the second one only relies on a

ResNet (See Figure 2.1) pretrained on classification and can easily be trained in an end-to-end manner and preserves spatial information until the joint embedding.

3.2.1 Textual path

The inputs to this path are tokenized sentences (captions), *i.e.*, variable length sequences of tokens $S = (s_1 \cdots s_T)$. Each token s_t is turned into a vector representation $\mathbf{s}_t \in \mathbb{R}^K$ by the pre-trained word2vec embedding [66] of size $K = 620$ used in [52]. Several RNNs have been proposed in the literature to turn such variable length sequences of (vectorized) words into meaningful, fixed-sized representations. In the specific context of semantic-visual embedding, [27, 51] use for instance gated recurrent unit (GRU) [10] networks as text encoders. Based on experimental comparisons, we chose to encode sentences with the simple recurrent unit (SRU) architecture recently proposed in [58]. Since we train this network from scratch, we take its output, up to ℓ_2 normalization, as the final embedding of the input sentence. There is no need here for an additional trainable projection layer.

Formally, the textual path reads:

$$S \xmapsto{\text{w2v}} \mathbf{S} \xmapsto{\text{normSRU}_{\boldsymbol{\phi}}} \mathbf{v} \in \mathbb{R}^d, \tag{3.1}$$

where $\mathbf{S} = \text{w2v}(S) = \mathbb{R}^{K \times T}$ is an input sequence of text tokens vectorized with word2vec and $\mathbf{v}$ is the final sentence embedding in the joint semantic-visual space, obtained after ℓ_2-normalizing the output of SRU with parameters $\boldsymbol{\phi}$.

3.2.2 BEAN Visual path

The BEAN visual path is inspired by weakly supervised object detection. Taking advantage of fully convolutional architecture [61], and selective spatial pooling [20, 102] we aim at linking spatial localization to concepts from VSE. With that purpose in mind our visual path needs to preserve a connection between input image, spatialize features, and final image representation.

In order to accommodate variable size images and to benefit from the performance of very deep architectures, we rely on fully convolutional residual ResNet-152 [37] as

our base visual network. Its penultimate layer outputs a stack of $D = 2048$ feature maps of size $(w, h) = (\frac{W}{32}, \frac{H}{32})$, where (W, H) is the spatial size of the input image. These feature maps retain coarse spatial information that lends itself to spatial reasoning in subsequent layers. Following the weakly supervised learning framework proposed by Durand *et al*. [19, 20], we first transform this stack through a linear adaptation layer of 1×1 convolutions. While in WELDON [20] and in WILDCAT [19] the resulting maps are class-related (one map per class in the former, a fixed number of maps per class in the latter), we do not address classification or class detection here.

Hence we empirically set the number D' of these new maps to a large value, 2400 in our experiments. A pooling à la WELDON is then used, but again in the absence of classes, to turn these maps into vector representations of dimension D'. A linear projection with bias, followed by ℓ_2 normalization accomplishes the last step to the embedding space of dimension d.

More formally, the visual embedding path is defined as follows:

$$\mathbf{I} \overset{f_{\theta_0}}{\longmapsto} \mathbf{F} \overset{g_{\theta_1}}{\longmapsto} \mathbf{G} \overset{\text{sPool}}{\longmapsto} \mathbf{h} \in \mathbb{R}^{D'} \overset{p_{\theta_2}}{\longmapsto} \mathbf{x} \in \mathbb{R}^d, \tag{3.2}$$

where: $\mathbf{I} \in (0, 255)^{W \times H \times 3}$ is the input color image, $f_{\theta_0}(\mathbf{I}) \in \mathbb{R}_+^{w \times h \times D}$ is the output of ResNet's `conv5` parametrized by weights in $\boldsymbol{\theta}_0$, g_{θ_1} is a convolution layer with $|\boldsymbol{\theta}_1| = D \times D'$ weights and with activation in $\mathbb{R}^{w \times h \times D'}$, sPool is the selective spatial pooling with negative evidence defined in [20]:

$$\mathbf{h}[k] = \max \mathbf{G}[:, :, k] + \min \mathbf{G}[:, :, k], \; k = 1 \cdots D', \tag{3.3}$$

and p_{θ_2} is an ℓ_2-normalized affine function

$$p_{\theta_2}(\mathbf{h}) = \frac{A\mathbf{h} + \mathbf{b}}{\|A\mathbf{h} + \mathbf{b}\|_2}, \tag{3.4}$$

where $\boldsymbol{\theta}_2 = (A, \mathbf{b})$ is of size $(D' + 1) \times d$. We shall denote $\mathbf{x} = F(\mathbf{I}; \boldsymbol{\theta}_{0:2})$ for short this visual embedding.

3.2.3 SMILE visual path

Our second visual path aims at producing finer visual representation by exploring relations between image regions. Starting from a spatially pre-trained Faster R-CNN used to extract a selection of salient object regions, we propose a distributed self-attention mechanism merging the said regions by putting them in competition.

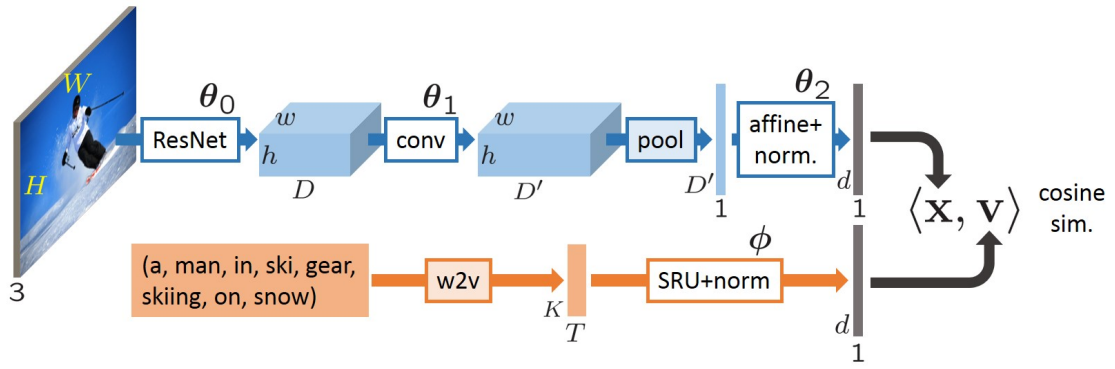


Figure 3.2: **Details of the proposed semantic-visual embedding architecture with BEAN visual path**. An image of size $W \times H \times 3$ is transformed into a unit norm representation $\mathbf{x} \in \mathbb{R}^d$; likewise, a sequence of T tokenized words is mapped to a normalized representation $\mathbf{v} \in \mathbb{R}^d$. Training will aim to learn parameters $(\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\phi})$ such that cross-modal semantic proximity translates into high cosine similarity $\langle \mathbf{x}, \mathbf{v} \rangle$ in the joint embedded space. Boxes with white background correspond to trainable modules, with parameters indicated on top. In our experiments, the dimensions are $K = 620$, $D = 2048$ and $D' = d = 2400$.

Faster R-CNN [75] proposes a variable-size selection of salient regions in the input image via its region pooling network and produces convolutional features for each of these regions. Given a color input image $\mathbf{I} \in (0, 255)^{w \times h \times 3}$, our visual embedding path is formally defined as follows:

$$\mathbf{I} \overset{\text{BU}}{\longmapsto} \mathbf{F} \overset{\text{Conv}_{\boldsymbol{\theta}_0}}{\longmapsto} \mathbf{G} \overset{\text{Att}_{\boldsymbol{\theta}_1}}{\longmapsto} \mathbf{x} \in \mathbb{R}^d, \tag{3.5}$$

where $\text{BU}(\mathbf{I}) \in \mathbb{R}_+^{R \times D}$ is the output of a pre-trained Faster R-CNN, that is the bottom-up features for the R proposed regions, $\text{Conv}_{\boldsymbol{\theta}_0}$ is a 1×1 convolution layer with $|\boldsymbol{\theta}_0| = D \times d$ weights and an activation G in $\mathbb{R}^{d \times R}$, and $\text{Att}_{\boldsymbol{\theta}_1}$ is the proposed attention mechanism that merges the region-wise activations into a final embedding vector $\mathbf{x}$ in $\mathbb{R}^d$.

SMILE embedding with distributed self-attention

Our main contribution consists in a vectorial, *distributed* self-attention mechanism in the visual path. Lightweight and efficient, it is designed to process multi-region features. By predicting multidimensional attention scores instead of just scalars, it allows for a finer re-weighting of the different regions of interest and, as a consequence, it produces a richer image embedding. The benefit of this novel attention mechanism first shows

in text-image matching, but also for visual grounding. As an additional, stand-alone improvement of our system, we introduce a simple, yet effective, asymmetric re-ranking (RR) technique that takes full advantage of the multi-modality for improved retrieval performance.

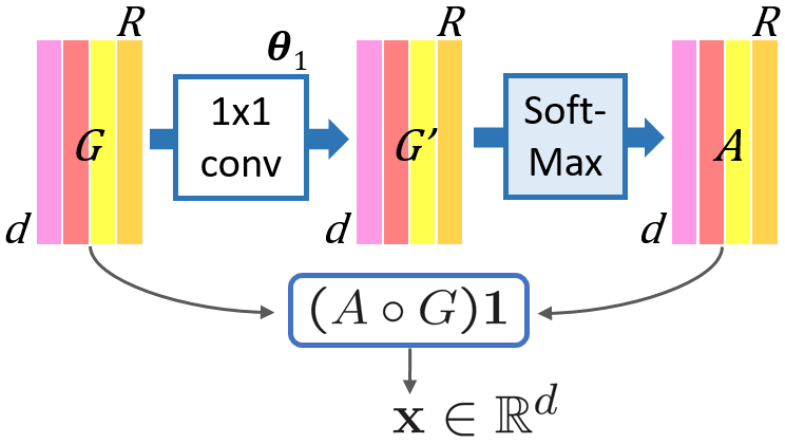


Figure 3.3: **From region features to global descriptors with distributed self-attention in SMILE visual path**. Given the bottom up region features G produced by a Faster R-CNN, our self-attention module learns to combine the features into a single image representation. The module first produces a distributed (channel, region) weighting A by learning a convolution on the original features and applying a softmax over the different regions. This weighting is applied to the original features to produce the final image representation.

DISTRIBUTED SELF-ATTENTION According to (3.5), an input image is turned into a collection $G \in \mathbb{R}^{d \times R}$ of R region-wise features, with R variable. These d-dimensional features are ℓ_2-normalized, *i.e.*, $\|G[:,r]\|_2 = 1,\ \forall r \in [\![1,R]\!]$. This variable-width matrix is fed to the attention module Att_{θ_1}, detailed in Figure 3.3, which produces an attention matrix $A \in \mathbb{R}^{d \times R}$ with a weight for each channel-region pair $(k,r) \in [\![1,d]\!] \times [\![1,R]\!]$. For each feature channel, a weighted average of all regions' contributions can thus be computed to produce the final d-dimensional image embedding $\mathbf{x}$. Formally:

$$\mathbf{x} = (A \circ G)\mathbf{1}, \tag{3.6}$$

where '$\circ$' denotes Hadamard entry-wise product. This yields for each feature channel $k \in [\![1,d]\!]$:

$$\mathbf{x}[k] = \sum_{r=1}^{R} a[k,r]G[k,r], \tag{3.7}$$

with positive weights required to sum to one. The attention matrix is defined as follows:

$$A = \sigma_{\text{row}}(WG), \tag{3.8}$$

where $W \in \mathbb{R}^{d \times d}$ is a trainable matrix and σ_{row} denotes the softmax operator applied row-wise. Note that the pre-multiplication by W amounts to a 1×1 convolution. The weights θ_1 of this layer are the entries of the matrix.

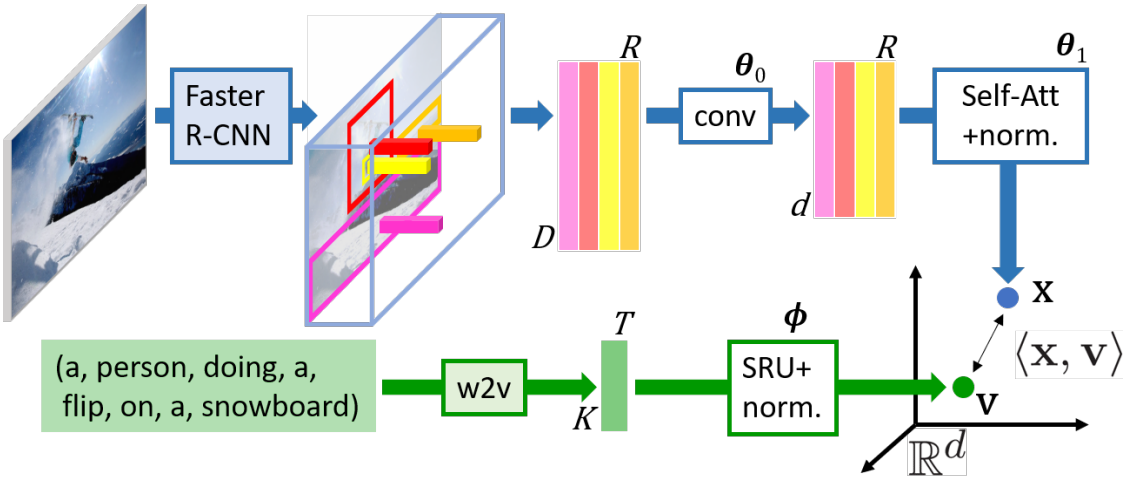


Figure 3.4: **Proposed semantic-visual embedding architecture with SMILE visual path**. (top) An input image is transformed into a representation $\mathbf{x} \in \mathbb{R}^d$ by the proposed visual path with distributed self-attention mechanism detailed in Figure 3.3; (bottom) An input sentence in the form of tokenized words is mapped as well to a representation $\mathbf{v} \in \mathbb{R}^d$. The two-stream architecture with parameters $(\theta_0, \theta_1, \phi)$ is trained with a triplet loss such that closeness in embedded space captures multimodal semantic proximity. White background boxes are trainable modules (with parameters indicated on top). As in BEAN model, the dimensions are set to $K = 620$, $D = 2048$ and $d = 2400$ in the experiments.

3.2.4 Learning and loss function

Both architectures are summarized in Figure 3.2 and Figure 3.4. The aim of training them is to learn the parameters θ of the visual path, as well as all parameters ϕ of the SRU text encoder. The goal is to create a joint embedding space for images and sentences such that closeness in this space can be interpreted as semantic similarity. This requires cross-modal supervision such that image-to-text semantic similarities are

indeed enforced. Note that mono-modal supervision can also be useful and relatively easier to get in the form, *e.g.*, of categorized images or of categorized sentences. Both are indeed used implicitly when relying on pre-trained CNNs and pre-trained text encoders. It is our case as well as far as the visual path is concerned. However, since our text encoder is trained from scratch, the only pure text (self-)supervision we implicitly use lies in the pre-training of word2vec.

CONTRASTIVE TRIPLET RANKING LOSS Following [51], we resort to a contrastive triplet ranking loss. Given a training set $\mathcal{T} = \{(\mathbf{I}_n, S_n)\}_{n=1}^N$ of aligned image-sentence pairs – the sentence describes (part of) the visual scene – the empirical loss to be minimized takes the form:

$$\mathcal{L}(\boldsymbol{\Theta}; \mathcal{T}) = \frac{1}{N} \sum_{n=1}^{N} \Big(\sum_{m \in C_n} \text{loss}(\mathbf{x}_n, \mathbf{v}_n, \mathbf{v}_m) + \sum_{m \in D_n} \text{loss}(\mathbf{v}_n, \mathbf{x}_n, \mathbf{x}_m) \Big), \tag{3.9}$$

where $\boldsymbol{\Theta} = (\boldsymbol{\theta}, \boldsymbol{\phi})$ are the parameters to learn, $\mathbf{x}_n = F(\mathbf{I}_n; \boldsymbol{\theta})$ is the embedding of image n, $\mathbf{v}_n = \text{normSRU}_{\boldsymbol{\phi}}(\text{w2v}(S_n))$ is the embedding of sentence n, $\{S_m\}_{m \in C_n}$ is a set of sentences unrelated to n-th image, $\{\mathbf{I}_m\}_{m \in D_n}$ is a set of images unrelated to n-th sentence. The two latter sets are composed of negative ("constrastive") examples. The triplet loss is defined as:

$$\text{loss}(\mathbf{y}, \mathbf{z}, \mathbf{z}') = \max\{0, \alpha - \langle \mathbf{y}, \mathbf{z} \rangle + \langle \mathbf{y}, \mathbf{z}' \rangle\}, \tag{3.10}$$

with $\alpha > 0$ a margin. It derives from triplet ranking losses used to learn metrics and to train retrieval/ranking systems. The first argument is a "query", while the second and third ones stand respectively for a relevant (positive) answer and an irrelevant (negative) one. The loss is used here in a similar way, but with a multimodal triplet. In the first sum of Equation 3.9, this loss encourages the similarity, in the embedding space, of an image with a related sentence to be larger by a margin to its similarity with irrelevant sentences. The second sum is analogous, but centered on sentences.

MINING HARD NEGATIVES In [47, 51], contrastive examples are sampled at random among all images (resp. sentences) in the mini-batch that are unrelated to the query sentence (resp. image). Faghri *et al.* [27] propose instead to focus only on the hardest negatives. We follow the same strategy: For each positive pair in the batch, a single

contrastive example is selected in this batch as the one that has the highest similarity with the query image/sentence while not being associated with it. This amounts to considering the following loss for the current batch $\mathcal{B} = \{(\mathbf{I}_n, S_n)\}_{n \in B}$:

$$\mathcal{L}(\Theta; \mathcal{B}) = \frac{1}{|B|} \sum_{n \in B} \Big(\max_{m \in C_n \cap B} \text{loss}(\mathbf{x}_n, \mathbf{v}_n, \mathbf{v}_m) + \max_{m \in D_n \cap B} \text{loss}(\mathbf{v}_n, \mathbf{x}_n, \mathbf{x}_m) \Big). \tag{3.11}$$

Beyond its practical interest, this mining strategy limits the amount of gradient averaging, making the training more discerning. Figure 3.5 contains a 2D visualization showing the difference between the contrastive loss, its hard negative variant and a solved situation where the loss is minimal.

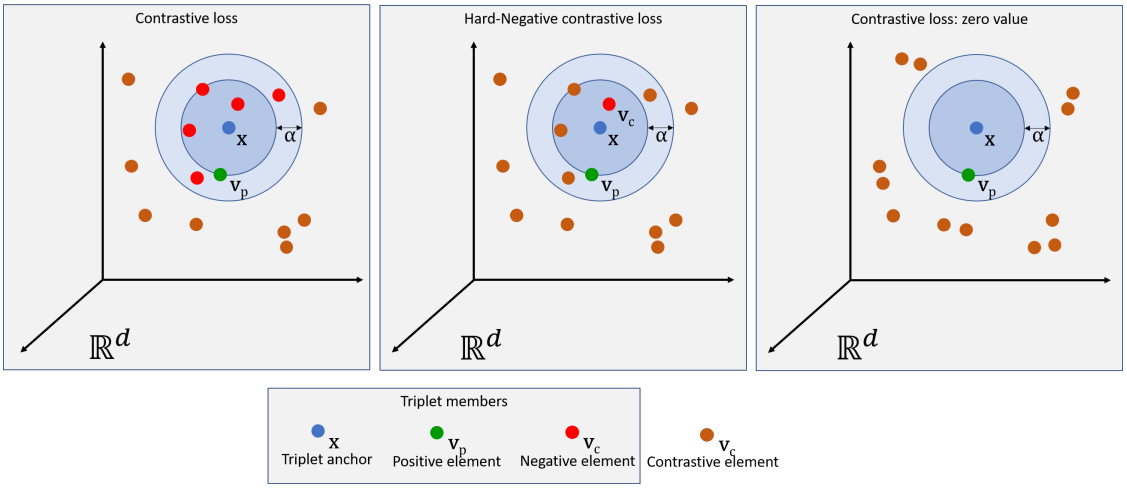


Figure 3.5: **2D visualization of contrastive losses**. On the standard contrastive loss, triplets are formed using all the contrastive examples that violate the α margin around the positive pair. On the hard negative variant, only one triplet is formed using the contrastive element that maximizes the violation.

3.2.5 Re-ranking

When used for cross-modal retrieval, the trained model turns database images and texts (captions) into collections $\{\mathbf{x}_i\}_{i=1}^M$ and $\{\mathbf{v}_j\}_{j=1}^N$ of representations. Cross-modal retrieval then amounts to finding the most relevant text (resp. image) for a query image (resp. text). Denoting $S = [\langle \mathbf{x}_i, \mathbf{v}_j \rangle] \in \mathbb{R}^{M \times N}$, the cross-modal similarity matrix in the embedded space, caption retrieval for the i-th image for instance reads $j^* = \arg\max_j S[i, j]$.

However, this similarity search ignores the relation that candidate captions might have with the other images of the database [90]. In other words, the score of caption j for query image i should somewhat take into account the similarity $S[:, j]$ of this caption with all images. In particular, if $\mathbf{v}_{j^*}$ turns out to be more similar to another image, *i.e.*, $S[i, j^*] \neq \max S[:, j^*]$, then the initial score of j^* should be downgraded. This calls for a normalization of the scores[2] yielding a re-ranking of retrieved results. To leverage this cross-modal information, we propose a simple score normalization as follows. For image retrieval, the new score matrix is defined as:

$$S_{\text{IR}}[i, j] = S[i, j] + \frac{S[i, j]}{\max(S[i, :])}, \ \forall (i, j) \in [\![1, M]\!] \times [\![1, N]\!]. \tag{3.12}$$

Likewise, for caption retrieval:

$$S_{\text{CR}}[i, j] = S[i, j] + \frac{S[i, j]}{\max(S[:, j])}, \ \forall (i, j) \in [\![1, M]\!] \times [\![1, N]\!]. \tag{3.13}$$

This way, when matching an image (resp. a sentence) to a sentence (resp. an image) we take into consideration the relative ranking of both the sentence and the image with respect to the other images and other sentences. If a pair (i, j) was already top scoring on both dimensions of S, meaning $S[i, j]$ is larger than all elements of $S[:, j]$ and $S[i, :]$, its rank will not be changed. However, if $S[i, j]$ is ranked differently in $S[:, j]$ and $S[i, :]$ the final score will be updated to form a more consistent ranking. An example can be seen in Figure 3.6.

2 This problem is reminiscent of collaborative filtering, where available user-item ratings can be normalized in a number of ways before matrix completion is performed.

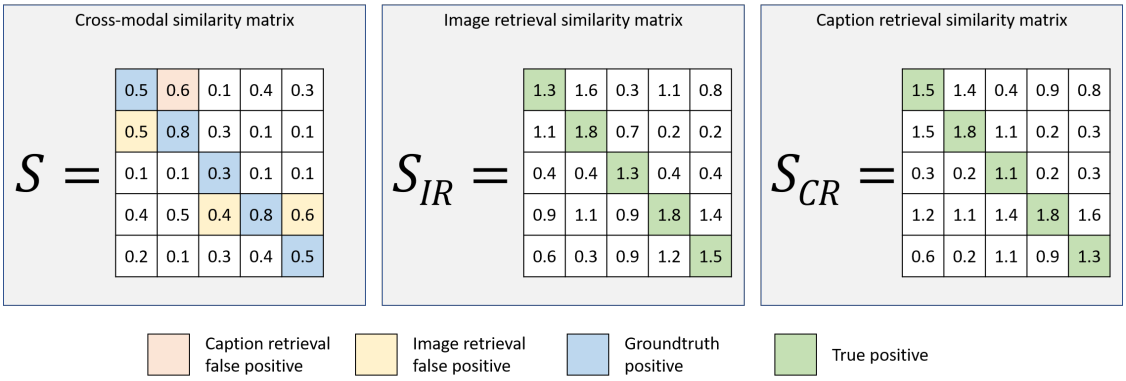


Figure 3.6: **Re-ranking example**. In the left matrix, the original score are displayed and the wrongly retrieved images (respectively caption) are colored in yellow (respectively red), the two right matrices contain the scores after being updated using Equation 3.12 and Equation 3.13.

3.3 Retrieval experiments

Starting from images annotated with text, we aim at producing rich descriptors for both image and text that live in the same embedding space. Our model is trained on the `MS-COCO` and `Flickr-30K` datasets. To evaluate the overall quality of the model we use cross-modal retrieval.

3.3.1 Training

DATASETS To train our model, we used the `MS-COCO` dataset [60][3]. This dataset contains 123,287 images (train+val), each of them annotated with 5 captions. It is originally split into a training set of 82,783 images and a validation set of 40,504 images. The authors of [47] proposed another split (called rVal in the rest of the paper) keeping from the original validation set 5,000 images for validation and 5,000 for testing and using the remaining 30,504 as additional training data. To make our results comparable, we trained a model using each split. We also trained one of our models on the `Flickr-30K` dataset [100]; this dataset contains 31,000 images, 1000 images are kept

3 http://cocodataset.org

for validation and 1000 for testing, the rest is used for training. For evaluation, we use the `MS-COCO` and `Flickr-30K` datasets.

TEXT PIPELINE To represent individual word tokens as vectors, we used pre-trained word2vec of size $K = 620$ from [52] with no further fine-tuning. The SRU text encoder [58] is trained from scratch jointly with the image pipeline. It has four stacked hidden layers of dimension 2400. Following [58], 0.25-probability dropout is applied on the linear transformation from input to hidden state and between the layers.

BEAN IMAGE PIPELINE AND MODEL The BEAN image pipeline is pre-trained on its own in two stages. We start from original ResNet-152 [37] pre-trained on ImageNet classification task. Then, to initialize the convolutional adaptation layer g_{θ_1}, we consider temporarily that the post-pooling projection is of size 1000 such that we can train both on ImageNet as well. Once this pre-training is complete, the actual projection layer p_{θ_2} onto the joint space is put in place with random initialization, and combined with a 0.5-probability dropout layer. As done in [27], random rectangular crops are taken from training images and resized to a fixed-size square (of size 256×256).

Visual and textual pipelines are trained together with pairs of images and captions, using Adam optimizer [50]. Not every part of the model is updated from the beginning. For the first 8 epochs only the SRU (parameters $\boldsymbol{\phi}$) and the last linear layer of the image pipeline ($\boldsymbol{\theta}_2$) are updated. After that, the rest of the image pipeline ($\boldsymbol{\theta}_{0:1}$) is also fine-tuned. The training starts with a learning rate of 0.001 which is then divided by two at every epoch until the seventh and kept fixed after that. Regarding mini-batches, we found in contrast to [27] that their size has an important impact on the performance of our system. After parameter searching, we set this size to 160. Smaller batches result in weaker performance while too large ones prevent the model from converging.

SMILE IMAGE PIPELINE AND MODEL The SMILE image pipeline uses bottom-up features extracted using a pre-trained Faster R-CNN on `Visual Genome`: For `MS-COCO` we use the features provided by [1], and the ones provided by [57] for `Flickr-30K`. The *bottom-up* features coming from Faster R-CNN are of dimension 36×2048 and are used with no further fine-tuning.

The model is trained in two stages, a warm-up stage for the first epochs where the visual pipeline is initialized (parameters θ_0 and θ_1), while the textual path is kept frozen. After 5 epochs the SRU (parameters ϕ) is also updated. We use the Adam optimizer [50] with a learning rate of 0.001 and halve it after epochs 6, 15, 25, 40 and 60. The batch size is empirically set to 192. The model is trained for 100 epochs, we select the best epoch using the validation score on the cross-modal retrieval task.

3.3.2 Cross-modal retrieval

The BEAN and SMILE models are both compared to similar existing architectures. Our models are quantitatively evaluated on a cross-modal retrieval task. Given a query image (resp. a caption), the aim is to retrieve the corresponding captions (resp. image). Since `MS-COCO` and `Flickr-30K` contain 5 captions per image, recall at r ("R@r") for caption retrieval is computed based on whether at least one of the correct captions is among the first r retrieved ones. For `MS-COCO`, the evaluation is performed 5 times on 1000-image subsets of the test set and the results are averaged (5-fold 1k evaluation protocol).

Results for BEAN are reported in Table 3.1. We compare our model with similar methods. For caption retrieval, we surpass VSE++ [27] by (5.2%,0.9%) on (R@1,R@10) in absolute, and by (3.9%,2.0%) for image retrieval. Three other methods are also available online, 2-Way Net [21], LayerNorm [3] and Embedding network [89]. The three methods are based on VGG network while VSE++ which uses a ResNet reports much stronger performance. We consistently outperforms all similar models, especially in terms of R@1.

The most significant improvement comes from the use of hard negatives in the loss, without them recall scores are significantly lower (R@1 - caption retrieval: -20,3%, image retrieval: -16.3%).

Note that in [27], the test images are scaled such that the smaller dimension is 256 and centrally cropped to 224×224. Our best results are obtained with a different strategy: Images are resized to 400×400 at inference irrespective of their size and aspect ratio, which our fully convolutional visual pipeline allows. When using the scale-and-crop protocol instead, the recalls of our system are reduced by approximately 1.4% in average

on the two tasks, remaining above VSE++ but less so. For completeness we tried our strategy with VSE++, but it proved counterproductive in this case.

		Caption retrieval			Image retrieval		
Model	CNN	R@1	R@5	R@10	R@1	R@5	R@10
Embedding network [89]	VGG	50.4	79.3	89.4	39.8	75.3	86.6
2-Way Net [21]	VGG	55.8	75.2	-	39.7	63.3	-
LayerNorm [3]	VGG	48.5	80.6	89.8	38.9	74.3	86.3
VSE++ [27]	R152	64.6	-	95.7	52.0	-	92.0
BEAN	R152	**69.8**	**91.9**	**96.6**	**55.9**	**86.9**	**94.0**

Table 3.1: **Comparison of BEAN with related approaches for cross-modal retrieval on `MS-COCO`.** On both caption retrieval from images and image retrieval from captions, the proposed architecture outperforms the similar systems. It yields an R@1 relative gain of 38% (resp. 40%) with respect to best published results [89] on cross-modal caption retrieval (resp. image retrieval), and 8% (resp 7.5%) with respect to best online results [27].

SMILE RESULTS As far as we know, the best comparable results on this task are obtained by the Visual Semantic Reasoning Network (VSRN) [59]. On `MS-COCO` 1K and for caption retrieval, we surpass it by (5.8%, 2.6%, 0.9%) on (R@1, R@5, R@10) in absolute, and by (3.6%, 2.1%, 1.3%) for image retrieval. On the `Flickr-30K` dataset and for caption retrieval we surpass it by (11.0%, 6.6%, 2.6%) on (R@1, R@5, R@10) in absolute, and on image retrieval by (7.6%, 4.9%, 3.6%).

SMILE also surpasses the BEAN model on `MS-COCO` 1K for caption retrieval by (10.4%, 4.5%, 2.5%) on (R@1, R@5, R@10) in absolute, and on image retrieval by (7.8%, 4.0%, 1.8%). This large gap in performance can be explained by the use of spatially supervised "bottom-up" features instead of simple ResNet features.

Overall, we obtain competitive results on all metrics and the simplicity of our model particularly shines on a smaller dataset like `Flickr-30K`, where the performance gap with more complex models (reasoning, cross attention) is the largest. Our architecture fully takes advantage of the richness of bottom-up features and is able to transfer their good properties to the final image representation.

	MS-COCO dataset					
	caption retrieval			image retrieval		
model	R@1	R@5	R@10	R@1	R@5	R@10
CAN [57]	70.9	94.5	97.8	56.4	87.0	93.9
MTFN [90]	94.9	97.9	60.1	89.1	95.0	
VSRN [59]	76.2	94.8	98.2	62.8	89.7	95.1
SMILE w/o RR	79.4	96.4	**99.1**	63.7	90.9	95.8
SMILE	82.0	97.4	**99.1**	66.4	91.8	96.4
	Flickr-30K dataset					
	caption retrieval			image retrieval		
model	R@1	R@5	R@10	R@1	R@5	R@10
CAN [57]	67.9	89.0	94.4	43.9	74.2	82.8
MTFN [90]	65.3	88.3	93.3	52.0	80.1	86.1
VSRN [59]	71.3	90.6	96.0	54.7	81.8	88.2
SMILE w/o RR	80.0	95.3	97.6	59.3	84.9	90.4
SMILE	**82.3**	**97.2**	**98.6**	**62.3**	**86.7**	**91.8**

Table 3.2: **Comparative performance of SMILE for Cross-modal retrieval.** The proposed model outperforms state-of-the-art system based on the bottom-up features on the Flickr-30K dataset, showing the effectiveness of the proposed self-attention mechanism and its rich visual representation even when trained on a small dataset. SMILE corresponds to the model described in Equation 3.2.3, with its distributed self-attention module and re-ranking during testing. For the sake of comparison, we also provide the test scores without the re-ranking (w/o RR).

3.3.3 Discussion

MINING HARD NEGATIVE AND BATCH SIZE The hard negative contrastive loss should benefit from using larger minibatch. Indeed the larger the minibacth the more likely it is to select a good hard negative when making triplets, resulting in better parameters update and faster convergence. However in practice we have observed a performance decline when batch size becomes too large. A possible explanation for this phenomenon is the presence of false negatives in the dataset. Actually, there are multiple pairs of image caption in `MS-COCO` that are similar, the caption of one image could be used to describe another image. However this alternate pairing is not annotated and is considered as a negative pair. If such false negative pairs are selected during the making of triplets, it would result in incorrect parameters update hurting the overall performance of the model.

	caption retrieval			image retrieval		
model	R@1	R@5	R@10	R@1	R@5	R@10
Emb. network [89]	40.7	69.7	79.2	29.2	59.6	71.7
2-Way Net [21]	49.8	67.5	-	36.0	55.6	-
VSE++ [27]	52.9	-	87.2	**39.6**	-	79.5
DAN [71]	**55.0**	**81.8**	**89.0**	39.4	69.2	79.1
BEAN (`MS-COCO` only)	46.5	72.0	82.2	34.9	62.4	73.5

Table 3.3: **Direct transfer to `Flickr-30K` with BEAN**. Although cross-validated and trained on `MS-COCO` only, our system delivers good cross-modal retrieval performance on `Flickr-30K`, compared to recent approaches trained on `Flickr-30K`: It is under the two best performing approaches, but above the two others on most performance measures.

DATASET SIZE AND MODEL ARCHITECTURE Training large deep learning model using small datasets is challenging, large network quickly overfit the dataset obtaining minimal loss on the training set but generalizing poorly to validation set. The BEAN architecture with the fine-tuning of its large Resnet-152 was prone to overfitting.

We propose to investigate how the BEAN model trained on MS-COCO may be transferred as such to other datasets, namely Flickr-30K here. We report the results in Table 3.3. Not surprisingly, our performance is below the best systems [27, 71] trained on Flickr-30K. Yet, while not being trained at all on Flickr-30K, it outperforms on almost all measures two other recent approaches trained on Flickr-30K [21, 89]. Note that *fine-tuning* our system on Flickr-30K makes it outperform all, including [27, 71], by a large margin (not reported in Table for the sake of fairness).

We chose to keep the architecture used on MS-COCO as it is and to experiment with transfer and fine-tuning. An actual evaluation on Flickr-30K would require cross-validation of the various hyper-parameters. This dataset being substantially smaller than MS-COCO, such a task is challenging given the size of our architecture with its 2400 new feature maps and its large final embedding dimension of 2400.

On the other hand, the SMILE architecture can easily be trained on Flickr-30K and outperforms all existing approaches. The large Faster R-CNN network is used to extract features offline, leaving a minimal amount of learnable parameters in the trained model, making it resilient to overfitting even on small datasets.

IMAGE REPRESENTATION AND RESOLUTION In the BEAN architecture, we noticed that the performance was directly linked to the resolution of the input images. Higher resolution tended to improve performance, however, due to hardware limitations we were not able to train a model with image larger than 256×256. Interestingly, increasing resolution at testing time further improve results and showed the potential of a fully convolutional network. Resolution might also explain the good performance of *bottom-up* features, since they are extracted with high-resolution images of dimension up to 1000×1000 pixels.

3.4 Ablation and model understanding

In order to get a better understanding of both proposed models, we conduct additional experiments and ablation study to identify which part of the models contribute to the overall performance.

3.4.1 BEAN: Changing pooling

One of the key elements of the proposed BEAN architecture is the final pooling layer, adapted from WELDON [20]. To see how much this choice contributes to the performance of the model, we tried instead the Global Average Pooling (GAP) [102] approach. With this single modification, the model is trained following the exact same procedure as the original one. This results in less good results: For caption retrieval (resp. image retrieval), it incurs a loss of 5.3% for R@1 (resp. 4.7%) for instance, and a loss of 1.1% in accuracy in the pointing game.

3.4.2 SMILE: Impact of self-attention

To measure the impact of the self-attention in SMILE, we evaluate our model against two baselines. First, we visualize its effect using the visual grounding capability of the model, then we use cross-modal retrieval to measure performance differences.

Figure 3.7: **Self-attention visualization in SMILE**. (Orange) "Mean" attention-less model baseline; (Green) Proposed attention mechanism. With its distributed self-attention module, SMILE is able to represent the image more globally, focusing less on specific details. As indicated by the transparent overlay, its attention is indeed more spread out, highlighting more accurately the salient parts of the scene.

Using the localization method described in Section 4.2, we can visualize the effect of the self-attention module. Figure 3.7 shows the similarity between each region and the

overall image representation for the "Mean" baseline and for our best model. In most cases, the attention mechanism helps the model distribute its focus among the salient objects of the scene instead of focusing only on small regions. In the first column of Figure 3.7 the model is for instance able to focus on the salient giraffe. In the second column, the attention helps focus on the cooking person rather than on the bottles in the background. This re-weighting of the regions is key to obtain a rich and complete representation of the depicted scene.

	Caption Retrieval		Image Retrieval	
Model	R@1	R@10	R@1	R@10
Mean	71.8	98.0	55.4	93.6
Mean + RR	74.3	98.7	58.7	94.4
1-Att	76.8	98.7	60.4	95.2
SMILE w/o RR	80.2	99.1	63.8	96.1
SMILE	82.2	99.5	66.6	96.6

Table 3.4: **Self-attention with SMILE on cross-modal retrieval**. Results of the comparison of the proposed model against two baselines using a simpler attention mechanism. The comparison is made on the cross-modal retrieval task on the validation set of `MS-COCO`.

Table 3.4 reports quantitative comparisons between our models and baselines on cross-modal retrieval. In addition to "Mean", we introduce the "1-Att" baseline where the number of filters in the 1×1 convolutional layer of the attention mechanism is reduced to one $W \in \mathbb{R}^{d \times 1}$. In this version, a single scalar attention weight is used per region to compute the final image representation. First, we can see that any type of visual attention mechanism (scalar or vectorial) is beneficial to image representation. Then, the difference between a scalar attention model with one attention weight per region (instead of one per region-channel pair) and our distributed approach is large, with an absolute gain on caption retrieval of (3.4%,0.4%) on (R@1,R@10), and of (3.4%,0.9%) for image retrieval. Finally, Table 3.4 also contains results showing the impact of the re-ranking technique. Whether on the "Mean" baseline or as part of SMILE, it improves retrieval results on all metrics.

SMILE: UNDERSTANDING THE ATTENTION MECHANISM To get a better understanding of the attention mechanism, we now perform an ablation study, removing either the convolutional layer or the SoftMax function (Table 3.5). The results show that both components of the attention contribute to the overall performance. They confirm the importance of feature re-weighting before fusing them to a single representation. Results without the convolutional layer show that making regions compete with each other using only SoftMax is already beneficial compared to a simple averaging ("Mean" baseline). On the other hand, when SoftMax is removed, the convolution layer is free to learn any linear re-weighting of the features: While being better than the "Mean" baseline, its overall performance remains lower than the full-fledged SMILE.

	Capt. Retrieval		Im. Retrieval	
Model	R@1	R@10	R@1	R@10
Mean	71.8	98.0	55.4	93.6
SMILE w/o Conv.	72.1	98.0	55.1	93.6
SMILE w/o Softmax	75.1	98.6	59.5	95.0
SMILE	80.2	99.1	63.8	96.1

Table 3.5: **Self attention mechanism ablation study**. Results of the ablation for the internal components of the self-attention mechanism in SMILE. The comparison is made on the cross-modal retrieval task on the validation set of `MS-COCO`.

To conclude the ablation experiments, we note that results in both Table 3.4 and Table 3.5 demonstrate the strength of the distributed self-attention mechanism and the need for a combination of convolution and softmax function.

3.4.3 Further analysis

With SMILE, we also obtain results that are similar to the very recently proposed Saliency-guided Attention Network (SAN) [45]. Since their visual pipeline is not based on bottom-up features but relies on an ensemble of networks (ResNext and Resnet) and uses a 10-crop strategy to produce initial visual features, a complete comparison

is difficult. Nonetheless, the main benefit of the SAN approach comes from its cross-modal textual attention (Saliency guided Text Attention), while our approach only relies on visual self-attention. For this reason we report in Table 3.6 a comparison between our model and SAN without cross-modal attention. We note that our visual representation outperforms SAN representation by a large margin. The two approaches being orthogonal in their proposed innovation, we hope that they can be combined toward new state-of-the-art results once SAN model is made available.

	Cap. Retrieval		Im. Retrieval	
Model	R@1	R@10	R@1	R@10
SAN w/o STA [45]	67.1	96.6	56.6	93.5
SAN [45]	85.4	99.0	69.1	97.2
SMILE w/o RR	79.4	99.1	63.7	95.8

Table 3.6: **Comparing SMILE and SAN visual pipelines**. Comparison of the proposed model with a version of SAN devoid of its Saliency-guided Textual Attention (STA), this architecture being similar to ours with its independent visual and textual pipelines. SMILE outperforms the ablated SAN by a large margin on the cross-modal retrieval task on MS-COCO test set, showing the strength of proposed visual representation.

MODEL EFFICIENCY To quantify the efficiency of our SMILE architecture, we finally benchmark the runtimes of existing approaches on cross-modal retrieval. Using the evaluation protocol of MS-COCO dataset, we measure the inference time of the model to encode images and texts and produce a final retrieval ranking on the whole validation set. Model initialization and features loading time is ignored, every model is tested with the same batch size of 128. For previous approaches, we use the official code available online. The results in Table 3.7 show that our architecture outperforms existing ones by a large margin, with a speed up of 6 times compared to VSRN [59] with its complex visual reasoning module. Compared to SCAN [57], our model is 35 times faster, which can be explained by the quadratic cost in the number of database images that SCAN incurs for textual encoding. Note that, as of today, the code for the SAN [45] is not

publicly available. However, since SAN relies on cross-modal attention, we expect a complexity close to the one of SCAN [57].

Model	Eval. Time
SCAN [57]	1606s
MTFN [90]	640s
VSRN [59]	266s
SMILE	**45s**

Table 3.7: **Cross-modal retrieval speed for SMILE and other related systems**. Evaluation times in seconds on `MS-COCO` validation dataset for the 5 folds of 1k images and caption retrieval. Our model is the fastest with a 6-time speedup w.r.t. the second fastest (VSRN).

3.5 Conclusion

We have presented two novel VSE models that leverage recent architectures to produce rich descriptors for both images and texts. When compared to similar architectures, they both achieve high performance on cross-modal retrieval.

Extensive ablation experiments show the contribution of each architecture element, and provide insights into the proposed attention mechanism of SMILE and the selective spatial pooling of BEAN.

Future improvement could come from the concept of attention. While we already introduced it in the visual pipeline with SMILE, it could also benefit the textual representation. Indeed a textual pipeline based on the recently introduced transformer [86] might outperform the current one only using RNN. One caveat is that training very large attention based model might require larger multimodal datasets to avoid overfitting.

Cross-modal attention is another form of attention that could also improve VSE. Using cross-modal attention, meaning computing the representation of a sentence with respect to an image (or vice versa) would intuitively produce a more accurate representation and would help with disambiguation. While cross-modal attention was already proposed

for VSE [57] it remains limited and costly to compute, the development of hardware (faster Graphic Processing Unit (GPU) with larger memory) might allow the creation of a more powerful mechanism.

Finally, both architectures are designed with localization in mind, one is weakly supervised while the other one is spatially pre-trained. In the following chapter, we will explore and evaluate their localization ability.

4

APPLICATION TO LOCALIZATION

Contents

Chapter abstract

In this chapter we explore the localization capability of multimodal embedding spaces. While VSEs are most commonly used for image-text matching, it is possible to use the aligned space to visually ground textual concept for free. Making the most of previously introduced architecture in Chapter 3 we tackle visual grounding of text in two different ways. First using the visual path based on space aware pooling mechanisms, we are able to produce weakly supervised localization of concepts. Then, with the visual path using self attention on bottom-up features, we output higher precision localization taking advantage of the spatial supervision from pre-training. Both methods obtain competitive results on the pointing game, and open interesting perspective for localization of unseen concepts.

The work in this chapter has led to two conference papers (one under review):

- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Finding beans in burgers: Deep semantic-visual embedding with localization." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3984–3993

- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Semantic-Visual Embedding with Distributed Self-Attention." In: *arXiv preprint*. 2020

4.1 Introduction

In this chapter we aim at extending the application of VSE. In Chapter 3 we proposed two new architectures and showed their good performance on image text matching. We will now introduce a localization method for each of the architecture.

With the strong prevalence of deep learning in computer vision, most visual scene understanding methods now rely on deep features extracted using off-the-shelf components. BEAN and SMILE are no exception:

- BEAN uses a visual pipeline based on a ResNet, it can be trained and fine-tuned in an end-to-end manner and doesn't require spatial supervision.
- SMILE is using spatialized *bottom-up* features, pretrained using ground truth localization of known objects.

We have shown that both architectures are competitive in the retrieval task, we will now exploit the specificity of the two pipelines and propose localization modules for each of them. Building on top of existing localization methods presented in Section 2.5 and taking advantage of the selective spatial pooling of BEAN, we will first introduce a weakly supervised visual grounding of text. Leveraging the object regions of SMILE and its efficient attention mechanism, we will then propose a partially supervised version of visual grounding of text.

Section 4.2 will detail the inner working of both localization mechanisms. Then Section 4.3 will contain qualitative visualization of the generated localization. Finally, we will quantitatively evaluate the localization mechanisms using the "pointing game" framework which has been recently introduced.

4.2 Localization from visual semantic embedding

In this section we will explain the mechanisms used to derive localization from the two models introduced in Chapter 3. Our first approach is weakly supervised and takes advantage of space aware pooling mechanism to link spatial information to the embedding space. The second mechanism leverages the supervised pre-training and uses the embedding space to link pre-extracted regions to concepts.

4.2.1 BEAN: Weakly supervised localization

Given an image and the embedding of a text (or any point of the embedding space), we propose a mechanism to compute a localization map, as demonstrated in Figure 4.2.

As described in Section 2.5, several works on weak supervised localization [20, 102] combine fully convolutional architectures with specific pooling mechanisms such that the unknown object positions in the training images can be hypothesized. This localization ability derives from the activation maps of the last convolutional layer. Suitable linear combinations of these maps can indeed provide one heatmap per class.

Based on the pooling architecture of [20] which is included in our system (described in Section 3.2.2) and without relying on additional training procedures, we derive the localization mechanism for our semantic-visual embedding. Let's remind that in our case, the number of feature maps is arbitrary since we are not training on a classification task but on a cross-modal matching one. Yet, one can imagine several ways to leverage these maps to try and map an arbitrary vector of the joint embedding space into an arbitrary input image. When this vector is the actual embedding of a word or sentence, this spatial mapping should allow localizing the associated concept(s) in the image, if present. Ideally, a well-trained joint embedding should allow such localization even for concepts that are absent from the training captions.

To this end, we propose the following localization process (Figure 4.1). Let $\mathbf{I}$ be an image and $\mathbf{G}$ its associated D' feature maps (Equation 3.5). This stack is turned into a stack $\mathbf{G}' \in \mathbb{R}^{w \times h \times d}$ of d heatmaps using the linear part of the projection layer p_{θ_2}:

$$\mathbf{G}'[i, j, :] = A\mathbf{G}[i, j, :], \ \forall (i, j) \in [\![1, w]\!] \times [\![1, h]\!], \tag{4.1}$$

which is a 1×1 convolution.

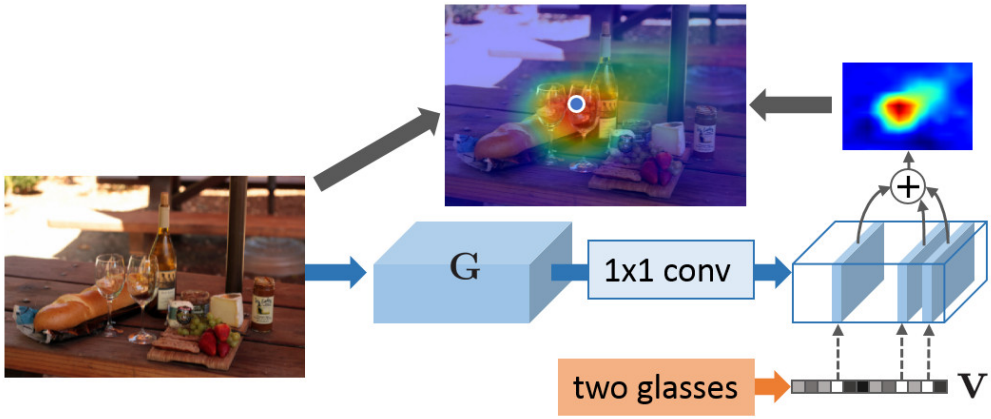


Figure 4.1: **From text embedding to visual localization with BEAN**. Given the feature maps $\mathbf{G}$ associated to an image by our semantic-visual architecture and the embedding of a sentence, a heatmap can be constructed: Learned projection matrix A serves as a 1×1 convolution; Among the d maps thus generated, the k ones associated with the largest among the d entries of $\mathbf{v}$ are linearly combined. If the sentence relates to a part of the visual scene, like "two glasses" in this example, the constructed heatmap should highlight the corresponding location. Circled blue dot indicates the heat maximum.

In other words, the pooling is removed. Bias and normalization being of no incidence on the location of the peaks, they are ignored.

Given $\mathbf{v} \in \mathbb{R}^d$ the embedding of a word or sentence (or any unit vector in the embedded space) and $K(\mathbf{v})$ the set of the indices of its k largest entries, the 2D heatmap $\mathbf{H} \in \mathbb{R}^{w \times h}$ associated with the embedded text $\mathbf{v}$ in image $\mathbf{I}$ is defined as:

$$\mathbf{H} = \sum_{u \in K(\mathbf{v})} \left|\mathbf{v}[u]\right| \times \mathbf{G}'[:, :, u]. \tag{4.2}$$

The number of feature maps from $\mathbf{G}'$ that are used to produce the localization map was set through cross-validation to $k = 180$ (out of 2400). We keep this parameter fixed for all presented results.

In this thesis, the heatmaps from the BEAN model will be shown in false colors, overlaid on the input image after suitable resizing, as illustrated in Figure 4.1 and Figure 4.2.

Figure 4.2: **Concept localization with BEAN semantic-visual embedding**. Not only does our deep embedding allows cross-modal retrieval, but it can also associate to an image, *e.g.*, the hamburger plate on the left, a localization heatmap for any text query, as shown with overlays for three text examples. The circled blue dot indicates the highest peak in the heatmap.

4.2.2 SMILE: Object region to localization using VSE

As described in Section 2.5, several works already proposed to use VSE to ground textual concepts in the visual space [89]. In this section we present a localization mechanism using the object region coming from the proposed model based on faster R-CNN described in Section 3.2.3.

Grounding semantic concepts in the visual domain is a common application of joint embedding models. Being based on spatialized bottom-up features, our SMILE visual pipeline lends itself naturally to this usage. Indeed, the proposed self-attention mechanism allows us to retain the localized information until the joint embedding space is reached.

Leveraging the region features in G, we use the joint embedding to score the relevance of each region with respect to a given concept. Given an arbitrary concept (*e.g.*, a word or a sentence) with associated representation $\mathbf{z}$ in $\mathbb{R}^d$ in the joint embedding space, the relevance of the R regions of interest in the input image is computed as follows:

$$L[r] = \langle \mathbf{z}, G[:, r] \rangle, \ \forall r \in [\![1, R]\!]. \tag{4.3}$$

As illustrated in Figure 4.3 the final visual grounding of concept $\mathbf{z}$ in image $\mathbf{I}$ can take two forms: The top-scoring region according to L (represented by a red bounding box) and a dense map obtained by associating each pixel with the top score among regions it belongs to.

In this thesis, such localization maps will be represented as a transparency layer (the more transparent the more relevant) overlaid on the input image as illustrated in Figure 4.3 and Figure 4.6.

Figure 4.3: **Free-form language localization using the SMILE semantic-visual embedding**. The rich visual representation produced by our distributed self-attention mechanism retains fine-grained details. This is useful for instance for visual grounding of text. In this example our model is able to distinguish the car from the truck and, when the "security" adjective is used, the correct vehicle is localized. Relying on multiple overlapping regions, the model is capable of subtle grounding, producing for instance two distinct localizations for the woman and for the couple. The red bounding box shows the top-scoring region for the overlaid text. The transparency map indicates pixel-level attention built out of all regions' scores.

4.3 Experiments

In the following section, we first describe the datasets used to evaluate the two previously introduced localization mechanisms, we then report qualitative and quantitative results on visual grounding. Finally, we further test our model with comparison to simpler baselines and ablation studies.

For the localization experiments, the model is trained only on the `MS-COCO` dataset [60]. We used the train validation split proposed in [47]. We complemented the `MS-COCO` dataset with the annotations from `Visual Genome` dataset [53] to get the localization ground truth that is needed for evaluation of the visual grounding.

We are evaluating two localization mechanisms leveraging the two different models proposed in Section 4.2. To easily distinguish both, all visualization followed the same

color code: Colored heatmaps are produced by the weekly supervised BEAN model. All the white heatmap with red top scoring regions are coming from the SMILE model.

4.3.1 The pointing game

Figure 4.4: **Pointing game illustration**. The left box contains an example images from the `Visual Genome` dataset, overlaid and underneath are the sentences associated with regions of the image. The two right images illustrate correct and incorrect localization example in the pointing game. The red dot corresponds to the predicted localization. Localization is considered correct if the pointwise localization for a query belongs to the ground truth bounding box associated.

We evaluate quantitatively our localization modules with the pointing game defined by [94]. This task relies on images that are present both in `MS-COCO val 2014` dataset and in `Visual Genome` dataset. The data contains 17,471 images with 86,5582 text region annotations (a bounding box associated with a caption). The task consists in "pointing" the region annotation in the associated image. If the returned location lies inside the ground truth bounding box, it is considered as a correct detection, a negative one otherwise. Figure 4.4 shows an example of the pointing game, the annotation used and a correct and incorrect localization. When our system produces a localization map, the location of its maximum is used as output for the evaluation. When our system produces region scores, the center of the maximum scoring region is used as output for the evaluation.

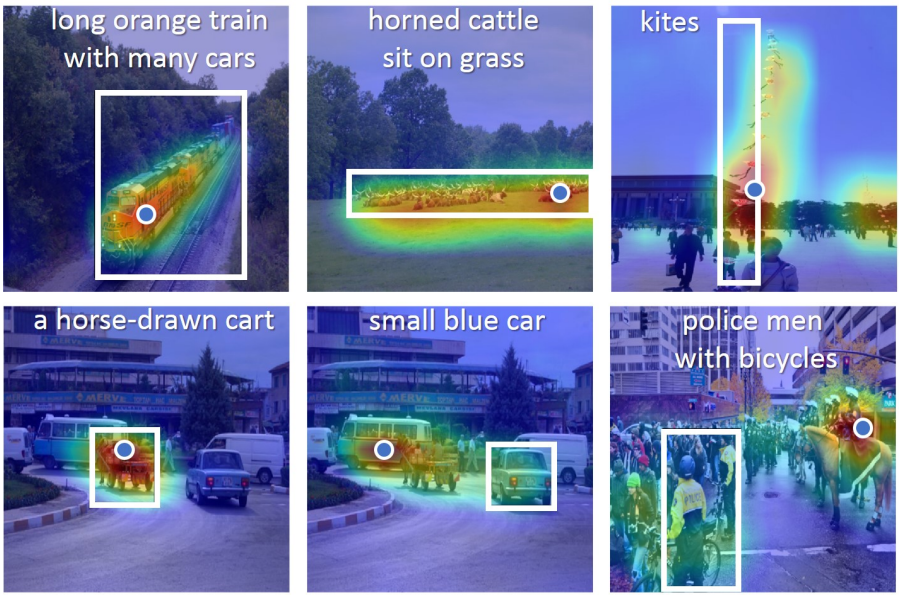


Figure 4.5: **Pointing game examples using BEAN**. Images from the `Visual Genome` dataset overlaid with the heatmap localizing the input text according to our weakly supervised system. The white box is the ground-truth localization of the text and the cirled blue dot marks the location predicted by our model for this text. The first four predictions are correct, unlike the last two ones. In the last ones, the heatmap is nonetheless active inside the ground-truth box.

The quantitative results are reported in Table 4.1 and some visual examples are shown in Figure 4.5. We add to the comparison a baseline that always outputs the center of the image as localization, leading to a surprisingly high accuracy of 19.5%. Our weakly supervised model, with an accuracy of 33.8%, offers absolute (resp. relative) gains of 9.4% (resp. 38%) over [94] and of 14% (resp. 73%) over the trivial baseline.

Our SMILE model, with an accuracy of 48.5%, offers absolute gains of 14.7% over the weakly supervised one and of 24.1% over [94]. Note that these two methods are not spatially supervised like SMILE (Faster R-CNN being pretrained using supervised annotations). For the sake of fairness, we also provide a baseline using the bottom-up features. In this baseline, "Mean", the attention mechanism is removed and the image representation is computed by averaging G as follows $\mathbf{x} = \frac{G\mathbf{1}}{R}$. As shown in Table 4.1, our model outperforms the "Mean" baseline by an absolute 6.2%.

Model	Accuracy
Unsupervised model	
center baseline [23]	19.5
Linguistic structure [94]	24.4
BEAN	33.8
Bottom-up features	
Mean	42.3
SMILE	**48.6**

Table 4.1: **Pointing game results**. Both our architectures obtain competitive results and outperform previous non-supervised approaches. Our SMILE model also outperforms a supervised baseline ('Mean') using the same bottom-up features.

Qualitative localization examples are also provided in Figure 4.6 and Figure 4.7. The model is able to localize any sort of textual query, from single words such as "elephant" or "cake" to more complex sentences such as "People on the station platform". In addition, the model can handle multiple instances of the same object class. For instance, it localizes perfectly the three candles and the trees in the second and third rows of Figure 4.6.

4.3.2 Further analysis

TOWARDS ZERO-SHOT LOCALIZATION The good performance we obtain in the pointing game highlights the ability of our system to localize visual concepts based on their embedding in the learned joint space. Going one step further, we conducted similar experiments with images from the web and concepts that were checked *not to appear in any of the training captions*, see Figure 4.8.

Figure 4.6: **Concept localization with SMILE**. Given a text query (overlay), the top-scoring region of the image is identified (red box) and pixel-wise scores are derived from all regions' scores (transparency layer).

4.4 Conclusion

In this chapter we have shown how BEAN and SMILE can be used to localize any concept represented in the join embedding space. The localization based on BEAN is weakly supervised and take advantage of the spatial information preserved by the fully convolutional architecture. The localization from SMILE is more accurate but requires a spatially supervised object detector.

Overall the results obtained by both models confirmed the benefit of using a VSE for localization. Indeed the possibility to visually ground any concept opens multiple opportunity for few or zero shot learning. In particular results in Figure 4.8 which

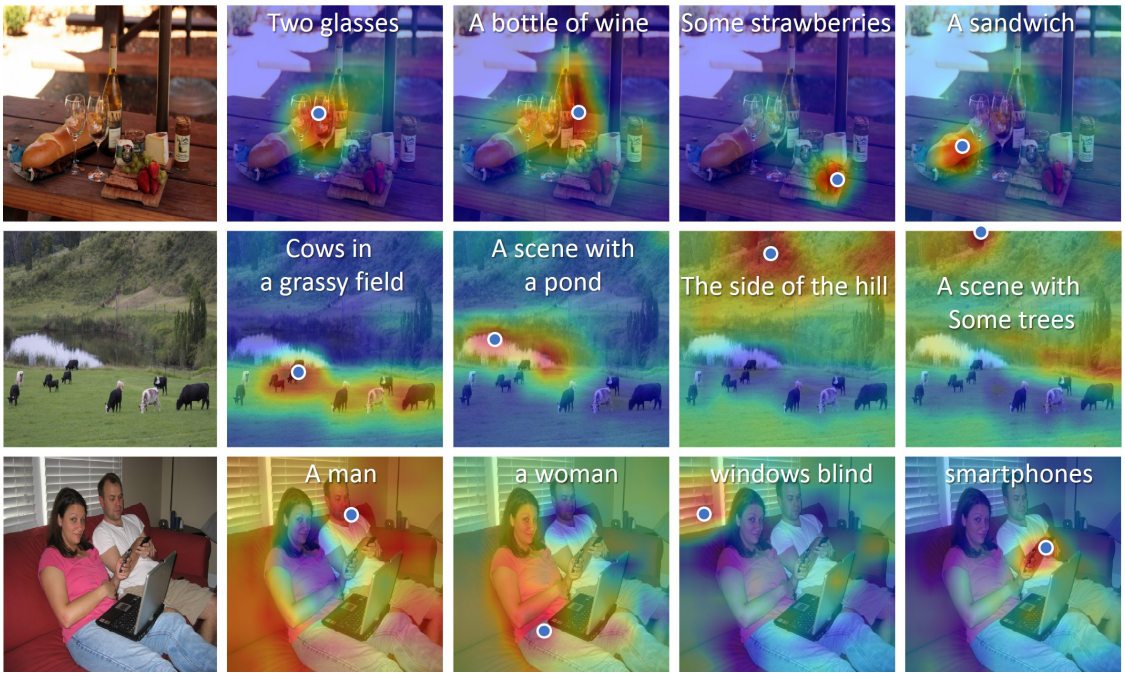


Figure 4.7: **Localization examples with BEAN**. The first column contains the original image, the next columns show as overlays the heatmaps provided by the localization module of our system for different captions (superimposed). In each image the circled blue dot marks the maximum value of the heatmap.

showed the ability of the localization mechanism to generalize to unseen concepts truly show the benefit of the combination of the textual and the visual modality.

With the increase in computer power, we could hope for a combination of both SMILE and BEAN architecture, the object detector could serve as a first rough localization while the weakly supervised localization module of BEAN could refine the localization using the convolutional activation map of the selected region.

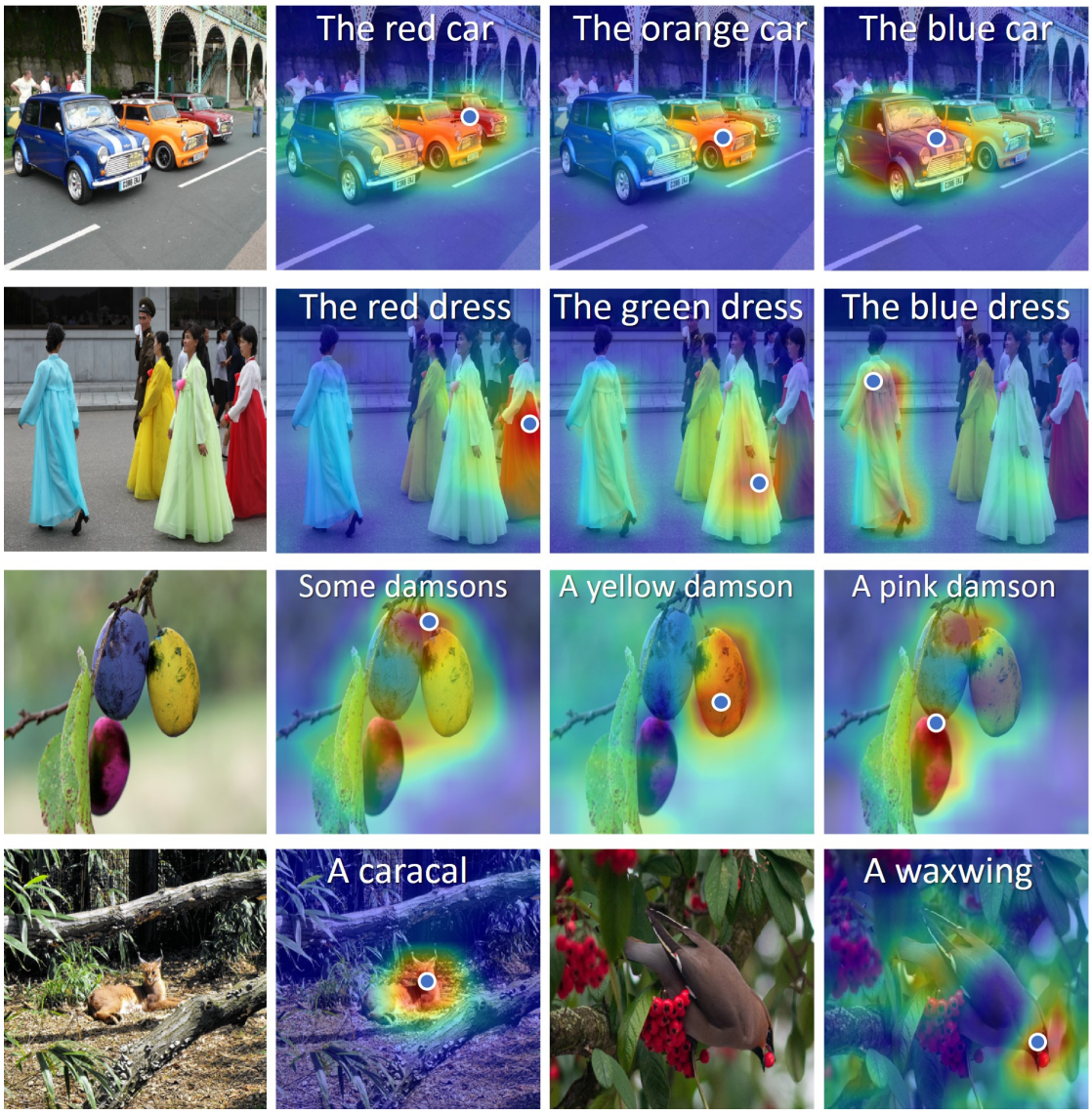


Figure 4.8: **Toward zero-shot localization using BEAN**. The first three rows show the ability to differentiate items according to their colors, even if, as in third example, the colors are unnatural and the concept has not been seen at training. This example, and the two last ones could qualify as "zero-shot localization" as `damson`, `caracal`, and `waxwing` are not present in `MS-COCO` train set.

5

RANKING LOSS FUNCTION

Contents

Chapter abstract

We have seen that VSEs are most commonly trained with a contrastive loss. In the retrieval context, the triplet loss is a good surrogate for the recall metrics. We are interested in finding a better one, not only for recall but for all ranking based metric e.g. *mean Average Precision (mAP) and Spearman correlation.*

In this chapter, we introduce a new method to learn approximations of non-differentiable ranking objective functions. Our approach is based on a deep architecture that approximates the sorting of arbitrary sets of scores. It is trained virtually for free using synthetic data. This

sorting deep (SoDeep) net can then be combined in a plug-and-play manner with existing deep architectures. We demonstrate the interest of our approach in three different tasks that require ranking: Cross-modal text-image retrieval, multi-label image classification and visual memorability rankings. Our approach yields very competitive results on these three tasks, which validates the merit and the flexibility of SoDeep as a proxy for sorting operation in ranking-based losses.

The work in this chapter has led to the publication of a conference paper:

- Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "SoDeep: a Sorting Deep net to learn ranking loss surrogates." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10792–10801

5.1 Introduction

The proposed VSE models in Chapter 3 were trained with a hard negative contrastive loss (See Section 3.2.4). In this last chapter, we take a step back and explore other possibilities of loss function for VSE and more generally for any ranking based loss function.

In Section 2.1 we explained how the loss function is critical to formalize a task during the training of a model. However machine learning tasks are often evaluated and compared using metrics which differ from the objective function used during training. The choice of an evaluation metric is intimately related to the definition of the task at hand, even sometimes to the benchmark itself. For example, accuracy seems to be the natural choice to evaluate classification methods, whereas the choice of the objective function is also influenced by the mathematical properties that allow a proper optimization of the model. For classification, one would typically choose the cross entropy loss – a differentiable function – over the non-differentiable accuracy. Ideally, the objective function used during training would be identical to the evaluation metric. However, standard evaluation metrics are often not suitable as training objectives for lack of differentiability to start with. This results in the use of surrogate loss functions that are better behaved (smooth, possibly convex). Unfortunately, coming up with good surrogate functions is not an easy task.

In this chapter, we focus on the non-differentiability of the evaluation metrics used in ranking-based tasks such as recall, mean average precision and Spearman correlation. Departing from prior art on building surrogate losses for such tasks, we adopt a simple, yet effective, learning approach: Our main idea is to approximate the non-differentiable part of such ranking-based metrics by an all-purpose learnable deep neural network. In effect, this architecture is designed and trained to mimic sorting operations. We call it SoDeep. SoDeep can be added in a plug-and-play manner on top of any deep network trained for tasks whose final evaluation metric is rank-based, hence not differentiable. The resulting combined architecture is end-to-end learnable with a loss that relates closely to the final metric.

We discuss in Section 5.2 the related works on direct and indirect optimization of ranking-based metrics, and position our work accordingly. In Section 5.3 we present the SoDeep approach. We show in particular how a "universal" sorting proxy suffices to tackle standard rank-based metrics, and present different architectures to this end. More details on the system and its training are reported in Section 5.5, along with various experiments. Finally, in Section 5.6 we discuss the benefit and limitation of our approach and future perspectives.

5.2 Related works

Many data processing systems rely on sorting operations at some stage of their pipeline. It is the case also in machine learning, where handling such non-differentiable, non-local operations can be a real challenge [69]. For example, retrieval systems require to rank a set of database items according to their relevance to a query. For sake of training, simple loss functions that are decomposable over each training sample have been proposed as for instance in [38] for the area under the ROC curve. Recently, some more complex non-decomposable losses (such as the Average Precision (AP), Spearman coefficient, and normalized discounted cumulative gain (nDCG) [7]), which present hard computational challenges, have been proposed [68].

MEAN AVERAGE PRECISION OPTIMIZATION Our work shares the high level goal of using ranking metrics as training objective function with many works before us. Several works studied the problem of optimizing average precision with support vector machines [46, 101] and other works extended these approaches to neural networks [5, 18, 68]. To learn rank, the seminal work [46] relies on a structured hinge upper bound to the loss. Further works reduce the computational complexity [68] or rely on asymptotic methods [83]. The focus of these works is mainly on the relaxation of the mean average precision, while our focus is on learning a surrogate for the ranking operation itself such that it can be combined with multiple ranking metrics. In contrast to most ranking-based techniques, which have to face the high computational complexity of the loss augmented inference [46, 68, 83], we propose a fast, generic, deep sorting architecture that can be used in gradient-based training for rank-based tasks.

APPLICATION OF RANKING BASED METRICS Ranking is commonly used in evaluation metrics. In this thesis, our focus is on cross-modal retrieval [27, 30, 47, 51, 63], where recall is the standard evaluation. We also explore other ranking problems such as image classification [19, 26] and object recognition which are both evaluated with mean average precision in the multi-label case. Ordinal regression [12] is evaluated using Spearman correlation.

EXISTING SURROGATE FUNCTIONS Multiple surrogates for ranking exist. Using metric learning to do retrieval is one of them. This popular approach avoids the use of the ranking function altogether. Instead, pairwise [96], triplet-wise [8, 91] and list-wise [6, 28] losses are used to optimize distances in a latent space. The cross-entropy loss is typically used for multi-label and multi-class classification tasks.

5.3 SoDeep approach

Rank-based metrics such as recall, Spearman correlation and mean average precision can be expressed as a function of the rank of the output scores. The computation of the rank being the only non-differentiable part of these metrics, we propose to learn a surrogate network that approximates directly this sorting operation. Using the approximation of the rank, we can then express any rank based loss function to train a DNN. Figure 5.1 shows an example of the SoDeep loss in application.

5.3.1 Learning a sorting proxy

Let $\mathbf{y} \in \mathbb{R}^d$ be a vector of d real values and $\mathbf{rk}$ the ranking function so that $\mathbf{rk}(\mathbf{y}) \in \{1 \cdots d\}^d$ is the d-permutation vector containing the rank for each variable in $\mathbf{y}$, *i.e.* $\mathbf{rk}(\mathbf{y})_i$ is the rank of $\mathbf{y}_i$ among the $\mathbf{y}_j$'s. We want to design a deep architecture $f_{\mathbf{\Theta}_B}$ that is able to mimic this sorting operator. The training procedure of this DNN is summarized in Figure 5.2. The aim is to learn its parameters, $\mathbf{\Theta}_B$, so that the output of the network is as close as possible to the output of the exact sorting.

Before discussing possible architectures, let's consider the training of this network, independent of its future use. We first generate a training set by randomly sampling N

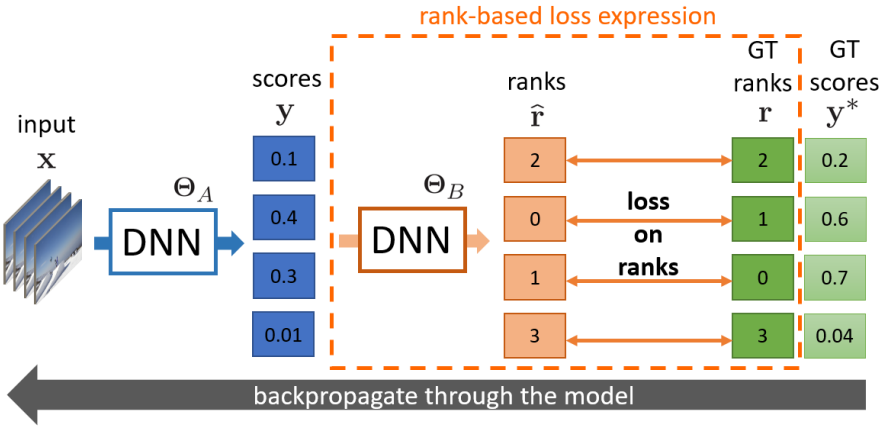


Figure 5.1: **Overview of SoDeep, the proposed end-to-end trainable deep architecture to approximate non-differentiable ranking metrics**. A pre-trained differentiable sorter (DNN Θ_B) is used to convert into ranks the raw scores $\mathbf{y}$ given by the model (DNN Θ_A) being trained with a collection of inputs $\mathbf{x}$. A loss is then applied to the predicted rank $\hat{\mathbf{r}}$ and the error can be back propagated through the differentiable sorter and used to update the weights Θ_A.

input vectors $\mathbf{y}^{(n)}$ and we compute through exact sorting the associated ground-truth rank vectors $\mathbf{r}^{(n)} = \mathbf{rk}(\mathbf{y}^{(n)})$. We then classically learn the DNN f_{Θ_B} by minimizing a L_1 loss between the predicted ranking vector $\hat{\mathbf{r}} = f_{\Theta_B}(\mathbf{y})$ and the ground-truth rank $\mathbf{r}$ over the training set:

$$\min_{\Theta_B} \sum_{n=1}^{N} \left\| \mathbf{rk}(\mathbf{y}^{(n)}) - f_{\Theta_B}(\mathbf{y}^{(n)}) \right\|_1 . \tag{5.1}$$

We explore in the following different network architectures and we explain how the training data is generated.

SORTER ARCHITECTURES We investigate two types of architectures for our differentiable sorter f_{Θ_B}. One is a recurrent network and the other one a convolutional network, each capturing interesting aspects of standard sorting algorithms:

- The recurrent architecture in Figure 5.3a consists of a bi-directional LSTM [81] followed by a linear projection. The bi-directional recurrent network creates a connection between the output of the network and every input, which is critical for ranking computation. Knowledge about the whole sequence is needed to compute the true rank of any element.

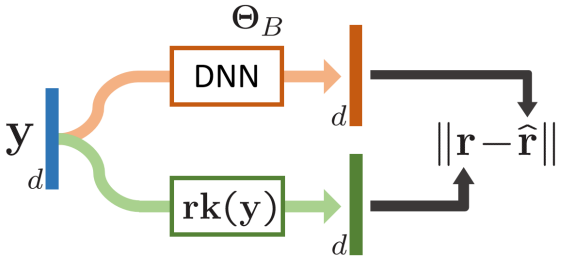
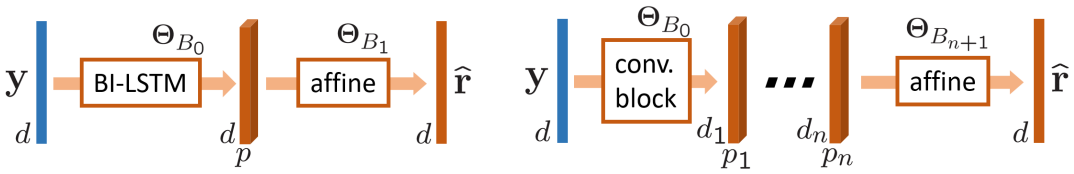


Figure 5.2: **Training a differentiable sorter**. Given a score vector $\mathbf{y}$ we learn the parameters $\mathbf{\Theta}_B$ of a DNN such that its output $\hat{\mathbf{r}}$ approximates the true rank vector $\mathbf{rk}(\mathbf{y})$. The model is trained using gradient descent and an L_1 loss. Once trained, $f_{\mathbf{\Theta}_B}$ can be used as a differentiable surrogate of the ranking function.



(a) **Architecture of LSTM sorter**.

(b) **Architecture of CNN sorter**.

Figure 5.3: **SoDeep architectures.** The sorter takes a vector of raw score $\mathbf{y} \in \mathbb{R}^d$ as input and outputs a vector $\hat{\mathbf{r}} \in \mathbb{R}^d$. Two architectures are explored, one recurrent (a), the other one, convolutional (b). Both architectures present a last affine layer to get a final projection to a vector $\hat{\mathbf{r}}$ in $\mathbb{R}^d$. Note that even if it is not explicitly enforced, $\hat{\mathbf{r}}$ will try to mimic as close as possible the vector of the ranks of the $\mathbf{y}$ variables.

- The convolutional architecture in Figure 5.3b consists of 8 convolutional blocks, each of these blocks being a one-dimensional convolution followed by a batch normalization layer [43] and a ReLU activation function. The sizes of the convolutional filters are chosen such that the output of the network contains as many channels as the length of the input sequence. Convolutions are used for their local property: indeed, sorting algorithms such as bubble sort [29] only rely on a sequence of local operations. The intuition is that a deep enough convolutional network, with its cascaded local operations, should be able to mimic recursive sorting algorithms and thus to provide an efficient approximation of ranks.

We will further discuss the interest of both types of SoDeep block architectures in the experiments.

TRAINING DATA SoDeep module can be easily (pre)trained with supervision on synthetic data. Indeed, while being non-differentiable, the ranking function **rk** can be computed with classic sorting algorithms. The training data consists of vectors of randomly generated scalars, associated with their ground-truth rank vectors. In our experiments, the numbers are sampled from different types of distributions:

- Uniform distribution over $[-1, 1]$;
- Normal distribution with $\mu = 0$ and $\sigma = 1$;
- Sequence of evenly spaced numbers in a uniformly drawn random sub-range of $[-1, 1]$;
- Random mixtures of the previous distributions.

While the differentiable sorter can be trained ahead of time on a variety of input distributions, as explained above, there might be a shift with the actual score distribution that the main network f_{Θ_A} will output for the task at hand. This shift can reduce naturally during training, or an alignment can be explicitly enforced. For example, f_{Θ_A} can be designed to output data in the interval used to learn the sorter, with the help of bounded functions such as cosine similarity.

5.3.2 SoDeep Training and Analysis

In this section we detail the way we train our differentiable sorter deep block using only synthetic data. We also present a comparison between the different models based on CNNs and on LSTM recurrent nets and with our baseline inspired from pairwise comparisons.

The proposed SoDeep models based on BI-LSTM and CNNs are trained on synthetic pairs of scores and ranks generated on the fly according to the distributions defined above.

For convenience we call an epoch as going through 100 000 pairs. The training is done using the Adam optimizer [49] with a learning rate of 0.001 which is halved every 100 epochs. Mini-batches of size 512 are used. The model is trained until the loss values stop decreasing and are stable.

A HANDCRAFTED SORTING BASELINE We add to our trainable SoDeep blocks a baseline that does not require any training. Inspired by the representation of the ranking problem as a matrix of pairwise ordering in [101], we build a handcrafted differentiable sorter f_h using pairwise comparisons.

A *sigmoid* function parametrized with λ scalar is used as a binary comparison function between two scalars a and b as:

$$\sigma_{comp}(a,b) = \frac{1}{1+e^{-\lambda(b-a)}}. \tag{5.2}$$

Indeed, if a and b are separated by a sufficient margin, $\sigma_{comp}(a,b)$ will be either 0 or 1. The parameter λ is used to control the precision of the comparator.

This function may be used to approximate the relative rank of two components $\mathbf{y}_i$ and $\mathbf{y}_j$ in a vector $\mathbf{y}$: $\sigma_{comp}(\mathbf{y}_i, \mathbf{y}_j)$ will be close to 1 if $\mathbf{y}_i$ is (significantly) smaller than $\mathbf{y}_j$, 0 otherwise. By summing up the result of the comparison between $\mathbf{y}_i$ and all the other elements of the vector $\mathbf{y}$, we form our ranking function f_h. More precisely, the rank $f_h(\mathbf{y}, i)$ for the i-est element of $\mathbf{y}$ is expressed as follows:

$$f_h(\mathbf{y}, i) = \sum_{j:j\neq i} \sigma_{comp}(\mathbf{y}_i, \mathbf{y}_j). \tag{5.3}$$

The overall precision of the handcrafted sorter can be controlled by the hyper parameter λ. The value of lambda is a trade off between the precision of the predicted rank and the efficiency when back-propagating through the sorter. This trade off is illustrated in Figure 5.4, when the sigmoid function takes value close to one or zero, its derivative is almost zero. We experimented with variable λ depending on the standard deviation of $\mathbf{y}$ but didn't obtained noticeable improvement compared to using a fixed value. Further experiments will use $\lambda = 10$.

SORTER PERFORMANCE COMPARISON Table 5.1 contains the loss values of the two different trained sorters and the handcrafted one on a generated test set of 10 000 samples. The LSTM based sorter is the most efficient, outperforming the CNN and the handcrafted sorters.

The performance of the CNN sorter slightly below the LSTM-based one can be explained by local behaviour of the CNNs, requiring a more complex structure to be able to rank elements.

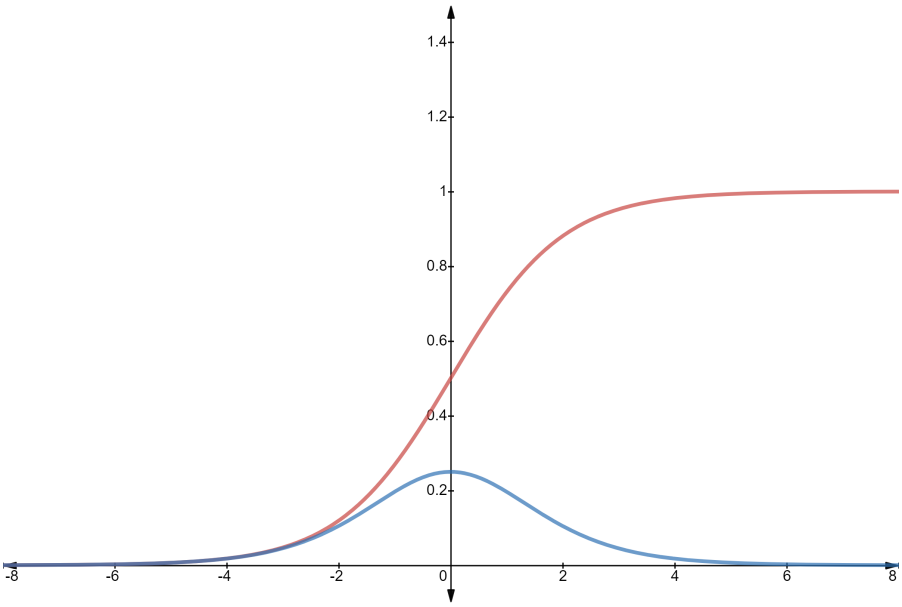


Figure 5.4: **Sigmoid function and its derivative**.The red curve corresponds to the sigmoid function used in Equation 5.2, the blue curve is its derivative. To use the sigmoid as a differentiable binary comparator the difference between the value a and b should be close to the range $\pm[4, 6]$ where the value of sigmoid function is almost 0 or 1 while its derivative is not 0.

In Figure 5.5 we compare CNN sorters with respect to their number of layers. From these results, we choose to use 8 layers in our CNN sorter since the performance seems to saturate once this depth has been reached. A possible explanation of this saturation is that the relation between the depth of the network and the input dimension ($d = 100$ here) is logarithmic.

FURTHER ANALYSIS The ranking function being non-continuous is non-differentiable, the rank value is jumping from one discrete value to another. We design an experiment to visualize how the different types of sorter behave at these discontinuities. Starting from a uniformly sampled vector $\mathbf{y}' \in \mathbb{R}^{100}$ of raw scores in the range $[-1, 1]$, we compute the ground truth rank $\mathbf{rk}(\mathbf{y}')_1$ and the predicted rank $f_{\Theta_B}(\mathbf{y}')_1$ of the first element y'_1 while varying this element y'_1 from -1 to 1 in increments of 0.001. The plot of the predicted ranks can be found in Fig. 5.6. The blue curve corresponds to the ground-truth rank where non-continuous steps are visible, whereas the curves for the learned sorters (orange and green) are a smooth approximation of the ground-truth curve.

Sorter model	L1 loss
Handcrafted sorter	0.0350
CNN sorter	0.0120
LSTM sorter loss	**0.0033**

Table 5.1: **Performance of the sorters on synthetic data**. Ranking performance of the sorter on the synthetic dataset. Among the learned sorters the LSTM one is the most efficient.

In Figure 5.7 we compare our SoDeep against previous approaches optimizing structured hinge upper bound to the mAP loss. We followed the protocol described in [83] for their synthetic data experiments. Our sorters using the loss $\mathcal{L}_{mAP}$ defined in Equation 5.9 are compared to a re-implementation of the Hinge-AP loss proposed in [46]. The results in Figure 5.7 show that our approach with the LSTM sorter (blue curve) gets mAP scores similar to [46] (purple curve) while being generic and less complex.

From the learned sorters, the LSTM architecture is the one performing best on synthetic data (Table 5.1). In addition, its simple design and small number of hyper-parameters make it straightforward to train. The CNN architecture while not being as efficient, uses a smaller number of weights and is 1.7 time faster. Further experiments will use the LSTM sorter unless specified otherwise.

5.4 Differentiable Sorter based loss functions

Rank-based metrics are used for evaluating and comparing learned models in a number of tasks. Recall is a standard metric for image and information retrieval, mean Average Prediction (mAP) for classification and recognition, and Spearman correlation for ordinal prediction. This type of rank-based metrics are non-differentiable because they require to transition from the continuous domain (score) toward the discrete domain (rank).

As presented in Figure 5.1, we propose to insert a pre-trained SoDeep proxy block f_{Θ_B} between the deep scoring function f_{Θ_A} and the chosen rank-based loss. We show in

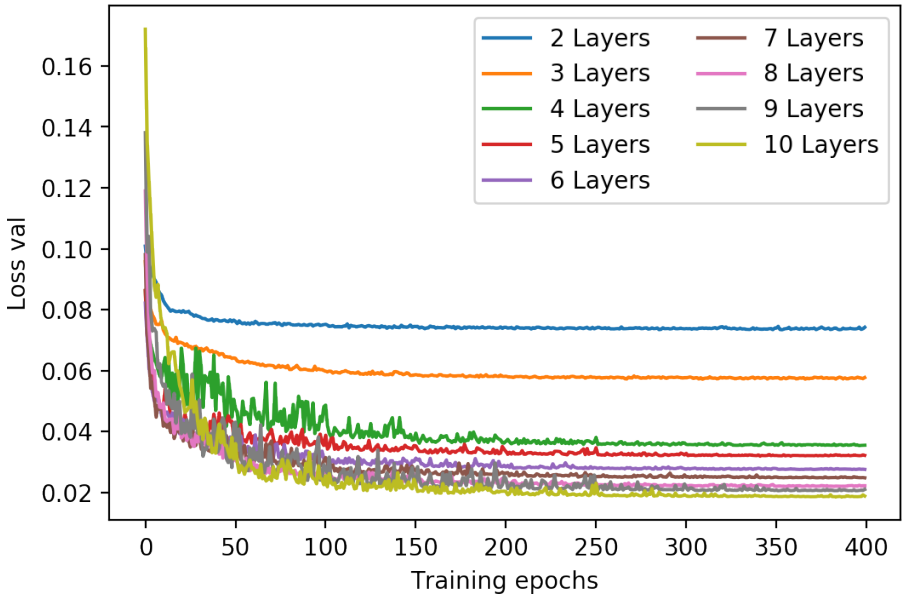


Figure 5.5: **Performance of the CNN sorter with respect to the depth of the CNN**. Value of the cost function during the training of multiple CNN sorters with a number of layers varying from 2 to 10. The model performances saturate for models with 8 layers or more.

the following how mAP, Spearman correlation and recall can be expressed as functions of the rank and combined with SoDeep accordingly.

In the following we assume a training set of annotated pairs $\{(\mathbf{x}_i, y_i^*)\}_{i=1}^M$ for the task at hand. A group $\mathcal{B}$ of d training examples among them yields a prediction vector $\mathbf{y}(\mathbf{\Theta}_A) = [f_{\mathbf{\Theta}_A}(\mathbf{x}_i)]_{i\in\mathcal{B}}$ and an associated ground-truth score vector $\mathbf{y}^* = [y_i^*]_{i\in\mathcal{B}}$ (Figure 5.1).

5.4.1 Spearman correlation

For two vectors $\mathbf{y}$ and $\mathbf{y}'$ of size d, corresponding to two sets of d observations, the Spearman correlation [17] is defined as:

$$r_s = 1 - \frac{6\|\mathbf{rk}(\mathbf{y}) - \mathbf{rk}(\mathbf{y}')\|_2^2}{d(d^2-1)}. \tag{5.4}$$

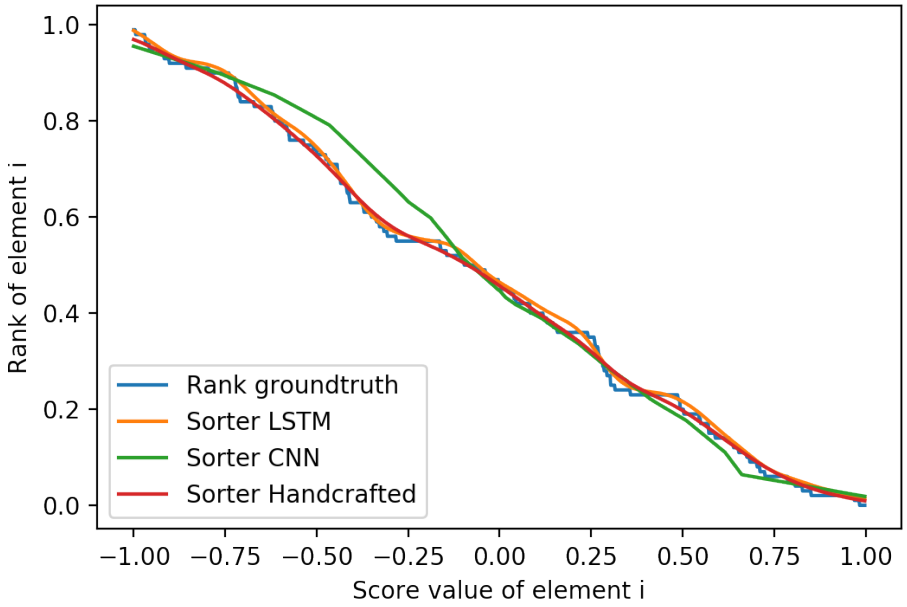


Figure 5.6: **Sorter behaviour analysis**. Given a synthetic vector $\mathbf{y}'$ of raw scores in the range $[-1, 1]$ of size 100, we plot the rank of its first element $\mathbf{y}'_1$ when the said value is linearly interpolated between -1 and 1. The x-axis represents the value $\mathbf{y}'_1$, and the y-axis is its corresponding rank.

Maximizing w.r.t. parameters $\mathbf{\Theta}_A$ the sum of Spearman correlations Equation 5.4 between ground truth and predicted score vectors over N subsets of training examples amounts to solving the minimization problem:

$$\min_{\mathbf{\Theta}_A} \sum_{n=1}^{N} \left\| \mathbf{rk}(\mathbf{y}^{(n)}) - \mathbf{rk}(\mathbf{y}^{*(n)}) \right\|_2^2, \tag{5.5}$$

with the loss not being differentiable.

Using now our differentiable proxy instead of the rank function, we can define the new Spearman loss for a group $\mathcal{B}$:

$$\mathcal{L}_{SPR}(\mathbf{\Theta}_A, \mathcal{B}) = \sum_{n=1}^{N} \left\| f_{\mathbf{\Theta}_B}(\mathbf{y}(\mathbf{\Theta}_A)^{(n)}) - \mathbf{rk}(\mathbf{y}^{*(n)}) \right\|_2^2. \tag{5.6}$$

Training will typically minimize it over a large set of groups. Note that here the optimization is done over $\mathbf{\Theta}_A$, knowing that SoDeep block $f_{\mathbf{\Theta}_B}$ has been trained independently on specific synthetic training data. Optionally, the block can be fine-tuned along the way, hence minimizing w.r.t. $\mathbf{\Theta}_B$ as well.

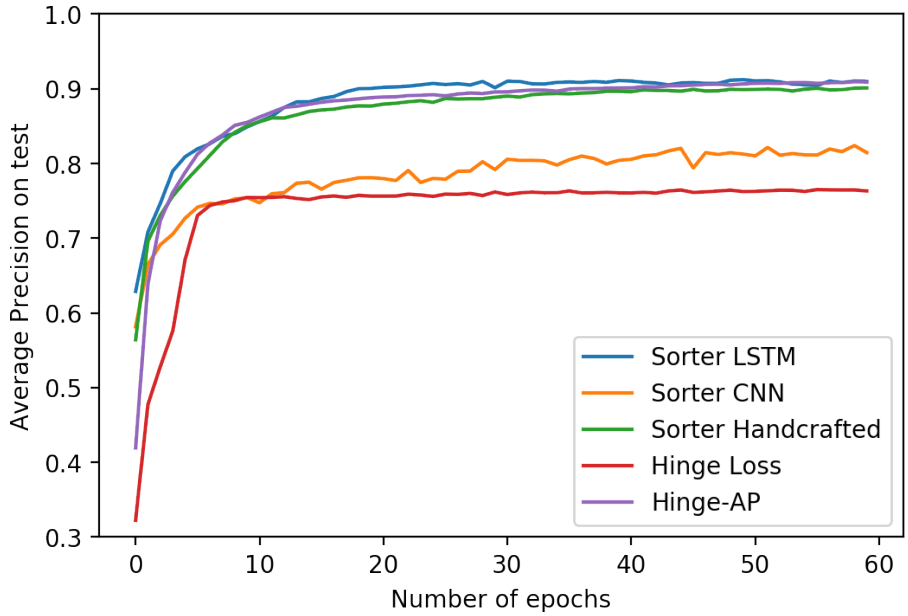


Figure 5.7: **Synthetic experiment on mAP optimization**. Comparison of the proposed sorter and the previous approaches.

5.4.2 Mean Average Precision (mAP)

Multilabel image classification is often evaluated using mAP, a metric from information retrieval. To define it, each of the C classes is considered as a query over the d elements of the datasets. For class c, denoting $\mathbf{y}_c^*$ the d-dimensional ground-truth binary vector and $\mathbf{y}_c$ the vector of scores for this class, the average precision (AP) for the class is defined as [101] :

$$AP(\mathbf{y}_c, \mathbf{y}_c^*) = \frac{1}{\text{rel}} \sum_{j:\mathbf{y}_c^*(j)=1} Prec(j), \tag{5.7}$$

where $\text{rel} = |j : \mathbf{y}_c^*(j) = 1|$ is the number of positive items for class c and precision for element j is defined as:

$$Prec(j) = \frac{|\{s \in \mathcal{S} \,:\, \mathbf{y}_c^*(s) = 1\}|}{\mathbf{rk}(\mathbf{y}_c)_j}, \tag{5.8}$$

with $\mathcal{S}$ the set of indices of the elements of $\mathbf{y}_c$ larger than $\mathbf{y}_c(j)$.

Minimizing $\mathbf{rk}(\mathbf{y})_j$ for all j from class c (i.e., those verifying $\mathbf{y}_c^*(j) = 1$) will be used as a surrogate of the maximization of the AP over predictor's parameters $\mathbf{\Theta}_A$.

The mAP is obtained by averaging AP over the C classes. Replacing the rank function by its differentiable proxy, the proposed mAP-based loss reads:

$$\mathcal{L}_{mAP}(\boldsymbol{\Theta}_A, \mathcal{B}) = \sum_{c=1}^{C} \langle f_{\boldsymbol{\Theta}_B}(\mathbf{y}_c), \mathbf{y}_c^* \rangle. \tag{5.9}$$

5.4.3 Recall at K

Recall at rank k is often used to evaluate retrieval tasks. In the following we assume a training set $\{\mathbf{x}_i\}_{i=1}^{M}$ for the task at hand. A group $\mathcal{B}$ of d training examples among them yields a $d \times d$ prediction matrix $\mathbf{Y}(\boldsymbol{\Theta}_A) = [f_{\boldsymbol{\Theta}_A}(\mathbf{x}_i)]_{i \in \mathcal{B}}$ representing the scores of all pairwise combinations of training examples in $\mathcal{B}$. In other words, the i-th column of this matrix, $\mathbf{Y}[i] = f_{\boldsymbol{\Theta}_A}(\mathbf{x}_i)$, provides the relevance of other vectors in the group w.r.t. to query $\mathbf{x}_i$.

This matrix being given, recall at K is defined as:

$$R@K(\mathbf{Y}) = \frac{1}{d} \sum_{i=1}^{d} \begin{cases} 1, & \text{if } \mathbf{rk}(\mathbf{Y}[i])_p < K \\ 0, & \text{otherwise,} \end{cases} \tag{5.10}$$

with p the index of the unique positive entry in $\mathbf{Y}[i]$, a single relevant item being assumed for query $\mathbf{x}_i$.

Once again, our sorter enables a differentiable implementation of this measure. However, we could not obtain conclusive results yet, possibly due to the batch size limiting the range of the summation. We found, however, an alternative way to leverage our sorting network. It is based on the use of the "triplet loss", a popular surrogate for recall. We propose to apply this loss on ranks instead of similarity scores, making it only dependent on the ordering of the retrieved elements. The triplet loss on the rank can be expressed as follows:

$$\text{loss}(\mathbf{Y}[i], p, c) = \max\left\{0, \alpha + f_{\boldsymbol{\Theta}_B}(\mathbf{Y}[i])_p - f_{\boldsymbol{\Theta}_B}(\mathbf{Y}[i])_c\right\}, \tag{5.11}$$

where p is defined as above (the positive example in the triplet, given anchor query $\mathbf{x}_i$) and c is the index of a negative (irrelevant) example for this query. The goal is to minimize the rank of the positive pair with score $\mathbf{Y}[i]_p$ such that its rank is lower than the rank of the negative pair with score $\mathbf{Y}[i]_c$ by a margin of α.

The complete loss is then expressed over all the elements of $\mathcal{B}$ in its *hard* negative version as:

$$\mathcal{L}_{REC}(\mathbf{\Theta}_A, \mathcal{B}) = \frac{1}{d} \sum_{i \in \mathcal{B}} \max_{c \neq p, c \neq i} \text{loss}(\mathbf{Y}[i], p, c). \tag{5.12}$$

In Equation 5.4, (5.7) and (5.10), the metrics are expressed in function of the non-differentiable rank function **rk**. Leveraging our differentiable surrogate allows us to design a differentiable loss function for each of these metrics, respectively Equation 5.6, (5.9) and (5.12).

5.5 Experimental Results

We present in this section several experiments to evaluate our approach. We evaluate the SoDeep combined with deep scoring functions $f_{\mathbf{\Theta}_B}$. The loss functions expressed in Equation 5.6, Equation 5.9 and Equation 5.12 are applied to three different tasks: memorability prediction, cross-modal retrieval, and object recognition.

Our method is benchmarked on three tasks. Each one of these tasks focuses on a different rank based loss function. Cross-modal retrieval will be used to test recall evaluation metrics, memorability prediction will be used for Spearman correlation and image classification will be used for mean average precision.

As explained in Figure 5.3.1, a shift in distribution might appear when using sorter-based loss. To prevent this, a parallel loss can be used to help domain alignment. This loss can be used only to stabilize the initialization or kept for the whole training.

5.5.1 Spearman Correlation: Predicting Media Memorability

The media memorability prediction task [12] is used to test the differentiable sorter with respect to the spearman correlation metrics. Examples of elements of the dataset can be found in Figure 5.8. Given a 7-second video the task consists in predicting the short term memorability score. The memorability score reflects the probability of a video being remembered.

Figure 5.8: **Media memorability dataset**. Frames with low and high memorability scores coming from 4 different videos of the memorability dataset [12]. The memorability scores are overlaid on top of the images.

The task is originally on video memorability. However the model used here is pre-trained on images, therefore 7 frames are extracted from each video and are associated with the memorability score of the source video. The training is done on pairs of frame and memorability score. During testing the predicted score of the 7 frames of a video are averaged to obtain the score per video. The dataset contains 8000 videos (56000 frames) for training and 2000 videos for testing. This training set is completed using LaMem dataset [48] adding 60 000 (image, memorability) pairs to the training data.

ARCHITECTURES AND TRAINING The regression model consists of a feature extractor combined with a two layers MLP [77] regressing features to a single memorability score. We use two pretrained nets to extract features: the Resnet-34 [37] and the BEAN visual path introduced in Section 3.2.

We use the loss $\mathcal{L}_{SPR}$ defined in Equation 5.6 to learn the memorability model. The training is done in two steps. First, for 15 epochs only the MLP layers are trained while the weights of the feature extractor are kept frozen. Second, the whole model is fine-tuned. The Adam optimizer [49] is used with a learning rate of 0.001 which is halved every 3 epochs. To help with domain adaptation, our loss is combined with an L1 loss for the first epoch.

RESULTS In Table 5.2, we compare the impact of the learned loss over two architectures. For both models we defined a baseline using a L2 loss. On both architectures

Single model	Spear. cor. test
Baseline [11]	46.0
Image only [36]	48.8
R34 + MSE loss	44.2
R34 + SoDeep loss	46.6
BEAN + MSE loss	48.6
BEAN + SoDeep loss	**49.4**

Table 5.2: **Media Memorability prediction results**. Our proposed loss function and architecture outperform the state-of-the-art system [36] by 0.6 pt. The last 4 lines show the benefit of our SoDeep loss function compared to a MSE loss on two different architectures: the first one uses ResNet 34 to extract visual features, while the second one uses the pre-trained BEAN visual path.

the proposed loss function achieves higher Spearman correlation by 2.4 points on the Resnet model and 0.8 points on the semantic embedding model. These are state of the art results on the task with an absolute gain of 0.6 pt. The model is almost on par (-0.3 pt) with an ensemble method proposed by [36] that is using additional textual data.

SORTER COMPARISON The memorability prediction is also used to compare the different types of sorters presented so far. Fixing the model and the hyper parameters, 4 models are trained with 4 different types of loss. The losses based on the LSTM sorter, the CNN sorter and the handcrafted sorter obtained respectively a Spearman correlation of 49.4, 46.6, 45.7, and the L1 loss gives a correlation of 46.2. These results are consistent with the result on synthetic data, with the LSTM sorter performing the best, followed by the CNN and handcrafted ones.

5.5.2 Mean Average precision: Image classification

The VOC 2007 [26] object recognition challenge is used to evaluate our sorter on a task using the mean average precision metric. We use an off-the-shelf model [19]. This model

is a fully convolutional network, combining a Resnet-101 [37] with advanced spatial aggregation mechanisms.

To evaluate the loss $\mathcal{L}_{mAP}$ defined in Equation 5.9 two versions of the model are trained: A baseline using only multi-label soft margin loss, and another model trained using the multi-label soft margin loss combined with $\mathcal{L}_{mAP}$.

Rows 3 and 4 of Table 5.3 show the results obtained by the two previously described models. Both models are below the state-of-the-art. However the use of the rank loss is beneficial and improves the mAP by 0.8 pt compared to the model using only the soft margin loss.

Loss	mAP
VGG 16 [82]	89.3
WILDCAT [19]	95.0
WILDCAT*	93.2
WILDCAT* + SoDeep loss	94.0

Table 5.3: **Object recognition results**. Model marked by (*) are obtained with code available online: https://github.com/durandtibo/wildcat.pytorch

5.5.3 Recall@K: Cross-modal Retrieval

The last benchmark used to evaluate the differentiable sorter is the cross-modal retrieval. Starting from images annotated with text, we train a model to produce rich features for both image and text that live in the same embedding space. Similarity in the embedding space is then used to evaluate the quality of the model on the cross-modal retrieval task.

Our approach is evaluated on the MS-COCO dataset [60] using the rVal split proposed in [47]. The dataset contains 110k images for training, 5k for validation and 5k for testing. Each image is annotated with 5 captions.

Given a query image (resp. a caption), the aim is to retrieve the corresponding captions (resp. image). Since MS-COCO contains 5 captions per image, recall at r ("R@r")

for caption retrieval is computed based on whether at least one of the correct captions is among the first r retrieved ones. The task is performed 5 times on 1000-image subsets of the test set and the results are averaged.

We use the BEAN model introduced in Chapter 3. It is a two-paths multimodal embedding approach that leverages the latest neural network architecture. The visual pipeline is based on a Resnet-152 and is fully convolutional. The textual pipeline is trained from scratch and uses a Simple Recurrent Unit (SRU) [58] to encode sentences. The model is trained using the loss $\mathcal{L}_{REC}$ defined in Equation 5.12 instead of the triplet based loss.

	caption retrieval				image retrieval			
model	R@1	R@5	R@10	Med. r	R@1	R@5	R@10	Med. r
Emb. network [87]	54.9	84.0	92.2	-	43.3	76.4	87.5	-
BEAN	69.8	91.9	96.6	1	55.9	86.9	94.0	1
GXN (i2t+t2i) [35]	68.5	-	**97.9**	1	**56.6**	-	**94.5**	1
BEAN + SoDeep loss	**71.5**	**92.8**	97.1	1	56.2	**87.0**	94.3	1

Table 5.4: **Cross-modal retrieval results on `MS-COCO`.** Using the proposed rank based loss function outperforms the hard negative triplet margin loss, outperforming similar approaches on the caption retrieval task.

Cross-modal retrieval results can be found in Table 5.4. The model trained using the proposed loss function (BEAN + SoDeep loss) outperforms the similar architecture BEAN trained with the triplet margin based loss by (1.7%,0.9%,0.5%) on (R@1,R@5,R@10) in absolute for caption retrieval, and by (0.3%,0.1%,0.3%) for image retrieval. It obtains state-of-the-art performance on caption retrieval and is very competitive on image retrieval being almost on par with the GXN [35] model, which has a much more complex architecture. It is important to note that the loss function proposed could be beneficial for any type of architecture.

5.6 Discussion

One could assume that the loss function for recall in Equation 5.11 could be simplified to:

$$\mathcal{L}_{REC}(\mathbf{\Theta}_A, \mathcal{B}) = \frac{1}{d} \sum_{i \in \mathcal{B}} f_{\mathbf{\Theta}_B}(\mathbf{Y}[i])_p. \tag{5.13}$$

With the idea that minimizing the rank of the positive pairs is the only thing needed. It would be a correct assumption, indeed backpropagation through the sorter should be enough to also update the negative elements without having to explicitly use them in the loss.

However, we found an interesting behavior when using Equation 5.13 to train our VSE model. When we started fine-tuning the parameters of the ResNet, the value of the loss kept decreasing while the evaluation performance stopped increasing accordingly.

Further analysis revealed that this behavior could be linked to a drop in accuracy of the sorting network $f_{\mathbf{\Theta}_B}$. With enough free parameters, the visual pipeline was able to "trick" the sorting network in outputting optimal rank, not corresponding to the input similarity scores but minimizing the overall loss.

In other words, the train model behaves in an adversarial mode to minimize the loss. Formulating the loss in terms of triplets made it possible to avoid the shortcoming of the sorting network and made training more robust.

This example reflects two connected limitations of the SoDeep approach:

- A transfer learning problem, since the sorter is trained on synthetic data there is no guarantee to have the training distribution matching the distribution on real applications.
- A robustness problem, the approximation learn by the sorter is not perfect and can be "tricked".

Multiple approaches might be used to mitigate these limitations, we implemented some of them. We increased the robustness of the sorter by training it on a wide range of distributions and using a powerful architecture.

To fight both limitations we could also imagine dynamic training where both the trained model and the sorter would be updated simultaneously. The sorter would adapt to the distribution produced by the model. This solution seems promising but raise other training stability problems and could be explored in future work.

5.7 Conclusion

In this chapter we introduced SoDeep, a novel method that leverages the expressivity of recent architectures to learn differentiable surrogate functions. Based on a direct deep network modeling of the sorting operation, such a surrogate allows us to train, in an end-to-end manner, models on a diversity of tasks that are traditionally evaluated with rank-based metrics. Remarkably, this deep proxy to estimate the rank comes at virtually no cost since it is easily trained on purely synthetic data.

Our experiments show that the proposed approach achieves good performance on cross-modal retrieval tasks as well as on media memorability prediction and multi-label image classification. These experiments demonstrate the potential and the versatility of SoDeep. This approach allows the design of training losses that are closer than before to metrics of interest, which opens up a wide range of other applications in the future. In particular it would be interesting to test this method on non-rank related function.

6

GENERAL CONCLUSION

Contents

6.1 Summary of contributions

In this thesis we extended Visual Semantic Embedding with the introduction of two models: SMILE and BEAN. Each of them with their specificity and their ability to localize concepts. In parallel we also explored training method for VSE and more generally for rank-based problems with the introduction of SoDeep, which learns approximation of non-differentiable ranking metrics.

BEAN We proposed BEAN a visual semantic embedding architecture that leverages the progress made both in visual and textual representation. The visual pipeline is fully convolutional and uses ResNet and selective spatial pooling while the textual pipeline is based on the efficient SRU. Trained with aligned paired of images and text from `MS-COCO` with the hard negative contrastive loss, it showed competitive performance again similar architecture on cross-modal retrieval. Most importantly it showed promising results when used for visual grounding. Trained without any spatial supervision, it was able to outperform other weakly supervised methods on the pointing game and to generalize to elements and concepts unseen during training.

SMILE SMILE is the second architecture introduced, it uses more sophisticated visual features coming from Faster R-CNN. Using the same textual representation as BEAN

and combined with the proposed distributed self-attention mechanism it produced a finer visual semantic alignment at the cost of a spatially supervised pre-training of Faster R-CNN. When tested on cross-modal retrieval, on `MS-COCO` and `Flickr-30K` it outperforms every existing method based on the same visual "Bottom-up" features. Transferring the spatial information learned during training to the joint embedding space, it also proved to be successful on visual grounding of phrases.

SODEEP Finally, we proposed a new method to design ranking loss functions with SoDeep. SoDeep is a new type of loss function; it is a differentiable proxy that is learned to mimic the non-differentiable ranking operation. Once learned we used it to derive approximation for the mAP, the Spearmann correlation and recall. We extensively tested these new loss functions using three different tasks and demonstrated the benefit of our approach. First, we showed it was a good alternative to the triplet loss to train VSE, then that it could also benefit ordinal regression on visual memorability and multi-label image classification on object recognition.

6.2 Perspectives and future work

The current performance of VSE on image-text matching might suggest that the task is mostly solved and only minimal progress can be made. I believe it is not the case and visual semantic representation has still a long way to go; at most we are starting to reach the limitation of the current datasets and are overfitting the existing tasks. I also believe there are many more tasks that could benefit from interaction between image and text; with our attempt at textual grounding of visual concepts we started to uncover the potential of VSE. In the future it might be a key element for the development of human and machine interaction.

In the short term, multiple improvements to existing architectures could be made to refine the current dual encoder pipelines, one of the more interesting concepts probably is the attention idea. While attention mechanisms have already been explored in the context of VSE, there is room for improvement. Overcoming the high computational cost of cross-modal attention would go a long way in order to permit stronger multimodal interactions.

In the long term, improving multimodal interaction might require to rethink visual and textual representation, helping models to learn finer representation by incorporating stronger structure. With the development of deep learning, came the development of differentiable programming frameworks such as Pytorch or Tensorflow; backed by these tools, new research started to combine deterministic differentiable modules with learnable networks in order to enforce structure into the representation which is sometimes denoted as differentiable programming. For example, 3d differentiable renderers are already used in combination with neural networks to extract faces' mesh from images or videos [85]. While it remains a challenge to generalize such ideas to generic images, it could greatly improve visual representation by enforcing a 3d representation of scenes. Improving model understanding and explainability would open up novel opportunities for multimodal interaction and might be a step toward perfect image-text matching.

BIBLIOGRAPHY

[1] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. "Bottom-up and top-down attention for image captioning and visual question answering." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 6077–6086 (cit. on pp. 31, 40–42, 60).

[2] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. "Deep canonical correlation analysis." In: *ICML*. 2013 (cit. on p. 38).

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey Hinton. "Layer normalization." In: *arXiv preprint arXiv:1607.06450* (2016) (cit. on pp. 49, 61, 62).

[4] Hedi Ben-Younes, Rémi Cadene, Nicolas Thome, and Matthieu Cord. "Block: Bilinear superdiagonal fusion for visual question answering and visual relationship detection." In: *arXiv preprint arXiv:1902.00038* (2019) (cit. on pp. 37, 38).

[5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. "Learning to rank using gradient descent." In: *ICML*. 2005 (cit. on p. 88).

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. "Learning to rank: From pairwise approach to listwise approach." In: *ICML*. 2007 (cit. on p. 89).

[7] Soumen Chakrabarti, Rajiv Khanna, Uma Sawant, and Chiru Bhattacharyya. "Structured learning for non-smooth ranking losses." In: *ACM SIGKDD*. 2008 (cit. on p. 88).

[8] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. "Large scale online learning of image similarity through ranking." In: *J. Machine Learning Research* 11 (2010), pp. 1109–1135 (cit. on pp. 27, 89).

[9] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." In: *arXiv preprint arXiv:1406.1078* (2014) (cit. on p. 35).

[10] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. "Empirical evaluation of gated recurrent neural networks on sequence modeling." In: *NeurIPS*. 2014 (cit. on p. 51).

[11] Romain Cohendet, Claire-Hélène Demarty, and Ngoc Q. K. Duong. "Transfer learning for video memorability prediction." In: *MediaEval Workshop*. 2018 (cit. on p. 102).

[12] Romain Cohendet, Claire-Hélène Demarty, Ngoc Duong, Mats Sjöberg, Bogdan Ionescu, and Thanh-Toan Do. "MediaEval 2018: Predicting Media Memorability Task." In: *arXiv preprint arXiv:1807.01052* (2018) (cit. on pp. 89, 100, 101).

[13] Corinna Cortes and Vladimir Vapnik. "Support-vector networks." In: *Machine learning* 20.3 (1995), pp. 273–297 (cit. on p. 30).

[14] Balázs Csanád Csáji. "Approximation with artificial neural networks." In: *Faculty of Sciences, Etvs Lornd University, Hungary* 24 (2001), p. 48 (cit. on p. 29).

[15] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. "R-FCN: Object detection via region-based fully convolutional networks." In: *NeurIPS*. 2016 (cit. on p. 42).

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In: *CVPR*. 2009 (cit. on p. 32).

[17] Yadolah Dodge. *The concise encyclopedia of statistics*. Springer Science & Business Media, 2008 (cit. on p. 96).

[18] Abhimanyu Dubey, Nikhil Naik, Devi Parikh, Ramesh Raskar, and César A Hidalgo. "Deep learning the city: Quantifying urban perception at a global scale." In: *ECCV*. 2016 (cit. on p. 88).

[19] Thibaut Durand, Taylor Mordan, Nicolas Thome, and Matthieu Cord. "Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on pp. 52, 89, 102, 103).

[20] Thibaut Durand, Nicolas Thome, and Matthieu Cord. "Weldon: Weakly supervised learning of deep convolutional neural networks." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on pp. 42, 44, 49, 51, 52, 66, 75).

[21] Aviv Eisenschtat and Lior Wolf. "Linking image and text with 2-way nets." In: *arXiv preprint arXiv:1608.07973* (2016) (cit. on pp. 61, 62, 64, 65).

[22] Aviv Eisenschtat and Lior Wolf. "Linking image and text with 2-way nets." In: *CVPR*. 2017 (cit. on p. 39).

[23] Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Finding beans in burgers: Deep semantic-visual embedding with localization." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3984–3993 (cit. on pp. 22, 48, 49, 73, 81).

[24] Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "SoDeep: a Sorting Deep net to learn ranking loss surrogates." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10792–10801 (cit. on pp. 22, 86).

[25] Martin Engilberge, Louis Chevallier, Patrick Pérez, and Matthieu Cord. "Semantic-Visual Embedding with Distributed Self-Attention." In: *arXiv preprint*. 2020 (cit. on pp. 22, 48, 74).

[26] Mark Everingham and J Winn. *The PASCAL visual object classes challenge 2007 development kit*. Tech. rep. 2007 (cit. on pp. 89, 102).

[27] Fartash Faghri, David Fleet, Jamie Ryan Kiros, and Sanja Fidler. "VSE++: Improved Visual-Semantic Embeddings." In: *arXiv preprint arXiv:1707.05612* (2017) (cit. on pp. 39, 49, 51, 56, 60–62, 64, 65, 89).

[28] Basura Fernando, Efstratios Gavves, Damien Muselet, and Tinne Tuytelaars. "Learning to rank based on subsequences." In: *ICCV*. 2015 (cit. on p. 89).

[29] Edward H Friend. "Sorting on electronic computer systems." In: *JACM* (1956) (cit. on p. 91).

[30] Andrea Frome, Greg Corrado, Jon Shlens, Samy Bengio, Jeff Dean, and Tomas Mikolov. "DeViSE: A deep visual-semantic embedding model." In: *NeurIPS*. 2013 (cit. on pp. 39, 48, 89).

[31] Andrea Frome, Yoram Singer, Fei Sha, and Jitendra Malik. "Learning globally-consistent local distance functions for shape-based image retrieval and classification." In: *ICCV*. 2007 (cit. on p. 27).

[32] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness." In: *arXiv preprint arXiv:1811.12231* (2018) (cit. on p. 36).

[33] Ross Girshick. "Fast r-cnn." In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448 (cit. on p. 42).

[34] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587 (cit. on pp. 41, 42).

[35] Jiuxiang Gu, Jianfei Cai, Shafiq Joty, Li Niu, and Gang Wang. "Look, Imagine and Match: Improving Textual-Visual Cross-Modal Retrieval with Generative Models." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018 (cit. on pp. 40, 104).

[36] Rohit Gupta and Kush Motwani. "Linear Models for Video Memorability Prediction Using Visual and Semantic Features." In: *MediaEval Workshop*. 2018 (cit. on p. 102).

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on pp. 31, 51, 60, 101, 103).

[38] Alan Herschtal and Bhavani Raskutti. "Optimising Area Under the ROC Curve Using Gradient Descent." In: *ICML*. 2004 (cit. on p. 88).

[39] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on p. 35).

[40] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks." In: *Neural networks* 4.2 (1991), pp. 251–257 (cit. on p. 29).

[41] Harold Hotelling. "Relations between two sets of variates." In: *Biometrika* (1936) (cit. on p. 38).

[42] Yan Huang, Wei Wang, and Liang Wang. "Instance-aware image and sentence matching with selective multimodal lstm." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2310–2318 (cit. on p. 41).

[43] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In: *arXiv preprint arXiv:1502.03167* (2015) (cit. on p. 91).

[44] Laurent Itti, Christof Koch, and Ernst Niebur. "A model of saliency-based visual attention for rapid scene analysis." In: *IEEE Trans. Pattern Recognition and Machine Intell.* 11 (1998), pp. 1254–1259 (cit. on p. 40).

[45] Zhong Ji, Haoran Wang, Jungong Han, and Yanwei Pang. "Saliency-Guided Attention Network for Image-Sentence Matching." In: *ICCV*. 2019 (cit. on pp. 68, 69).

[46] Thorsten Joachims. "Optimizing search engines using clickthrough data." In: *ACM SIGKDD*. 2002 (cit. on pp. 88, 95).

[47] Andrej Karpathy and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015 (cit. on pp. 39, 43, 44, 48, 49, 56, 59, 78, 89, 103).

[48] Aditya Khosla, Akhil S. Raju, Antonio Torralba, and Aude Oliva. "Understanding and Predicting Image Memorability at a Large Scale." In: *ICCV*. 2015 (cit. on p. 101).

[49] D Kinga and J Ba Adam. "A method for stochastic optimization." In: *ICLR*. 2015 (cit. on pp. 92, 101).

[50] Diederik Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." In: *ICLR*. 2014 (cit. on pp. 28, 60, 61).

[51] Ryan Kiros, Ruslan Salakhutdinov, and Richard Zemel. "Unifying visual-semantic embeddings with multimodal neural language models." In: *arXiv preprint arXiv:1411.2539* (2014) (cit. on pp. 37, 39, 40, 44, 48–51, 56, 89).

[52] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. "Skip-thought vectors." In: *NeurIPS*. 2015 (cit. on pp. 34, 51, 60).

[53] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. "Visual genome: Connecting language and vision using crowdsourced dense image annotations." In: *arXiv preprint arXiv:1602.07332* (2016) (cit. on pp. 32, 33, 78).

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks." In: *NeurIPS*. 2012 (cit. on p. 30).

[55] Pei Ling Lai and Colin Fyfe. "Kernel and nonlinear canonical correlation analysis." In: *Int. J. Neural Syst.* (2000) (cit. on p. 38).

[56] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. "Backpropagation applied to handwritten zip code recognition." In: *Neural computation* 1.4 (1989), pp. 541–551 (cit. on p. 30).

[57] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. "Stacked cross attention for image-text matching." In: *ECCV*. 2018, pp. 201–216 (cit. on pp. 41, 44, 60, 63, 69–71).

[58] Tao Lei and Yu Zhang. "Training RNNs as Fast as CNNs." In: *arXiv preprint arXiv:1709.02755* (2017) (cit. on pp. 51, 60, 104).

[59] Kunpeng Li, Yulun Zhang, Kai Li, Yuanyuan Li, and Yun Fu. "Visual Semantic Reasoning for Image-Text Matching." In: *ICCV*. 2019 (cit. on pp. 41, 44, 62, 63, 69, 70).

[60] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. "Microsoft COCO: Common objects in context." In: *ECCV*. 2014 (cit. on pp. 32, 33, 49, 59, 78, 103).

[61] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440 (cit. on pp. 31, 41, 51).

[62] David Lowe. "Object recognition from local scale-invariant features." In: *ICCV*. 1999 (cit. on p. 30).

[63] Lin Ma, Zhengdong Lu, Lifeng Shang, and Hang Li. "Multimodal convolutional neural networks for matching image and sentence." In: *ICCV*. 2015 (cit. on pp. 39, 89).

[64] Wei-Ying Ma and Bangalore S Manjunath. "Netra: A toolbox for navigating large image databases." In: *Multimedia systems* 7.3 (1999), pp. 184–198 (cit. on p. 30).

[65] Alexis Mignon and Frédéric Jurie. "PCCA: A New Approach for Distance Learning from Sparse Pairwise Constraints." In: *CVPR*. 2012 (cit. on p. 27).

[66] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." In: *arXiv preprint arXiv:1301.3781* (2013) (cit. on pp. 34, 51).

[67] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. "Linguistic regularities in continuous space word representations." In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2013, pp. 746–751 (cit. on p. 40).

[68] Pritish Mohapatra, Michal Rolinek, CV Jawahar, Vladimir Kolmogorov, and M Kumar. "Efficient optimization for rank-based loss functions." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018 (cit. on p. 88).

[69] Mehryrar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT Press, 2012 (cit. on p. 88).

[70] Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. "Dual attention networks for multimodal reasoning and matching." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 299–307 (cit. on p. 41).

[71] Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. "Dual attention networks for multimodal reasoning and matching." In: *arXiv preprint arXiv:1611.00471* (2017) (cit. on pp. 64, 65).

[72] Zhenxing Niu, Mo Zhou, Le Wang, Xinbo Gao, and Gang Hua. "Hierarchical Multimodal LSTM for Dense Visual-Semantic Embedding." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on p. 43).

[73] Boris T Polyak. "Some methods of speeding up the convergence of iteration methods." In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (1964), pp. 1–17 (cit. on p. 28).

[74] Mengye Ren, Ryan Kiros, and Richard Zemel. "Exploring models and data for image question answering." In: *Advances in neural information processing systems*. 2015, pp. 2953–2961 (cit. on p. 38).

[75] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In: *Advances in neural information processing systems*. 2015, pp. 91–99 (cit. on pp. 31, 42, 53).

[76] Zhou Ren, Hailin Jin, Zhe Lin, Chen Fang, and Alan Yuille. "Joint Image-Text Representation by Gaussian Visual-Semantic Embedding." In: *ACMMM*. 2016 (cit. on p. 48).

[77] Frank Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain." In: *Psychological review* (1958) (cit. on p. 101).

[78] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors." In: *nature* 323.6088 (1986), pp. 533–536 (cit. on p. 28).

[79] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. "Imagenet large scale visual recognition challenge." In: *International journal of computer vision* 115.3 (2015), pp. 211–252 (cit. on p. 32).

[80] 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." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on p. 39).

[81] Mike Schuster and Kuldip K Paliwal. "Bidirectional recurrent neural networks." In: *IEEE Trans. Signal Processing* (1997) (cit. on p. 90).

[82] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on pp. 30, 103).

[83] Yang Song, Alexander Schwing, and Raquel Urtasun. "Training deep neural networks via direct loss minimization." In: *ICML*. 2016 (cit. on pp. 88, 95).

[84] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. "On the importance of initialization and momentum in deep learning." In: *International conference on machine learning*. 2013, pp. 1139–1147 (cit. on p. 28).

[85] Ayush Tewari, Michael Zollhofer, Hyeongwoo Kim, Pablo Garrido, Florian Bernard, Patrick Perez, and Christian Theobalt. "Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction." In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1274–1283 (cit. on p. 109).

[86] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In: *Advances in neural information processing systems*. 2017, pp. 5998–6008 (cit. on pp. 36, 70).

[87] Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. "Learning two-branch neural networks for image-text matching tasks." In: *IEEE Trans. Pattern Recognition and Machine Intell.* 41.2 (2018), pp. 394–407 (cit. on pp. 44, 104).

[88] Liwei Wang, Yin Li, and Svetlana Lazebnik. "Learning Deep Structure-Preserving Image-Text Embeddings." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on p. 38).

[89] Liwei Wang, Yin Li, and Svetlana Lazebnik. "Learning Two-Branch Neural Networks for Image-Text Matching Tasks." In: *IEEE Trans. Pattern Recognition and Machine Intell.* (2017) (cit. on pp. 43, 49, 61, 62, 64, 65, 77).

[90] Tan Wang, Xing Xu, Yang Yang, Alan Hanjalic, Heng Tao Shen, and Jingkuan Song. "Matching Images and Text with Multi-modal Tensor Fusion and Re-ranking." In: *arXiv preprint arXiv:1908.04011* (2019) (cit. on pp. 58, 63, 70).

[91] Kilian Q Weinberger and Lawrence K Saul. "Distance metric learning for large margin nearest neighbor classification." In: *J. Machine Learning Research* (2009) (cit. on pp. 27, 89).

[92] Jason Weston, Samy Bengio, and Nicolas Usunier. "Wsabie: Scaling Up to Large Vocabulary Image Annotation." In: *IJCAI*. 2011 (cit. on p. 39).

[93] Terry Winograd. *Procedures as a representation for data in a computer program for understanding natural language*. Tech. rep. MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, 1971 (cit. on p. 33).

[94] Fanyi Xiao, Leonid Sigal, and Yong Jae Lee. "Weakly-Supervised Visual Grounding of Phrases With Linguistic Structures." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017 (cit. on pp. 43, 79–81).

[95] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. "Distance metric learning, with application to clustering with side-information." In: *NeurIPS*. 2002 (cit. on p. 27).

[96] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. "Distance metric learning with application to clustering with side-information." In: *NeurIPS*. 2003 (cit. on p. 89).

[97] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. "Show, attend and tell: Neural image caption generation with visual attention." In: *ICML*. 2015, pp. 2048–2057 (cit. on p. 40).

[98] Fei Yan and Krystian Mikolajczyk. "Deep correlation for matching images and text." In: *CVPR*. 2015 (cit. on p. 38).

[99] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. "How transferable are features in deep neural networks?" In: *Advances in neural information processing systems*. 2014, pp. 3320–3328 (cit. on p. 28).

[100] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions." In: *Trans. of the Association for Computational Linguistics* 2 (2014), pp. 67–78 (cit. on pp. 33, 59).

[101] Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. "A support vector method for optimizing average precision." In: *ACM SIGIR*. 2007 (cit. on pp. 88, 93, 98).

[102] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning deep features for discriminative localization." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016 (cit. on pp. 42, 43, 49, 51, 66, 75).