THÈSE DE DOCTORAT DE
l'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Thibaut DURAND

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Weakly Supervised Learning for Visual Recognition

Apprentissage faiblement supervisé pour la reconnaissance visuelle

soutenue le 20 septembre 2017

devant le jury composé de :

M. Patrick Pérez	Technicolor	Rapporteur
M. Alain Rakotomamonjy	INSA de Rouen - LITIS	Rapporteur
M. Francis Bach	INRIA Ecole Normale Superieure	Examinateur
Mme Cordelia Schmid	INRIA - THOTH	Examinatrice
Mme Véronique Serfaty	DGA	Invité
M. Matthieu Cord	UPMC - LIP6	Directeur de thèse
M. Nicolas Thome	CNAM - CEDRIC	Co-directeur de thèse

ABSTRACT

Today, with the massive use of smartphones and social networks, images are ubiquitous in our daily lives. To process and exploit this mass of data, it is important to have recognition systems to analyze and interpret the visual content of the images. This thesis studies the problem of image classification, where the goal is to predict if a semantic category – e.g. car, cat – is present in the image according to its visual content.

Several preliminary works have shown that to analyze images of complex scenes, it is important to learn localized representations. To this end, classical approaches use rich annotations – e.g. bounding boxes, segmentation masks – which are expensive to obtain, whereas in the standard protocol of the image classification these annotations are not available. To avoid collecting and using rich annotations during training, we have been interested in weakly supervised learning methods. The main idea is to model the missing information in the training data with latent variables. In this thesis, we propose models that simultaneously classify and localize objects, using only global image labels during learning. The contributions of this thesis are threefold: architecture, optimization and experimental results.

The weak supervision significantly reduces the cost of full annotation in many recognition tasks, but it makes learning and recognition more challenging. The key issue is how to aggregate local scores – e.g. regions – into a global score – e.g. image. This problem can be seen as a pooling problem. The main contribution of this thesis is the design of new pooling functions for weakly supervised learning. In particular, we propose a pooling function "max + min" which unifies many pooling functions, including several pooling functions introduced in this thesis – SyMIL, MANTRA, WELDON. We prove that the "min" regions provide complementary information to the "max" regions, and can be seen as *negative evidence* of the class in the case of multi-class classification. We describe how to use this pooling in a Latent Structured SVM framework as well as in convolutional networks. Finally, we present a new transfer layer that captures several modalities per class, to enrich the model and to have better predictions.

Our contributions about the optimization are multiple. We show that the objective function of SyMIL can be written as a difference of convex functions, and we present two solvers: one to solve the optimization problem in the primal, and another to solve it in the dual, and which allows to use the kernel trick. For the MANTRA model, we propose a non-convex cutting-plane algorithm to solve the optimization problem. We also show that MANTRA allows to optimize a ranking metric such as Average Precision during training, if there is a method to optimize this metric in the fully-supervised case.

We experimentally show the interest of our models with respect to state-of-the-art methods, on ten standard image classification datasets, including the large-scale dataset ImageNet. In particular, we show that, with the same representations, our models that focus on discriminating regions are better than state-of-the-art models using representations extracted over whole images. We also note that optimizing Average Precision during training is relevant to the datasets evaluated with this metric. To compare and analyze the strengths and the weaknesses of commonly used pooling functions, we propose an experimental study on six datasets. Finally, we show that our final model, developed for classification, is also competitive for localization and segmentation.

RÉSUMÉ

Aujourd'hui, avec l'utilisation massive des smartphones et des réseaux sociaux, les images sont omniprésentes dans notre vie quotidienne. Pour traiter et exploiter cette masse de données, il est important d'avoir des systèmes de reconnaissance visuelle, pour analyser et interpréter le contenu visuel des images. Cette thèse s'intéresse au problème de la classification d'images, où l'objectif est de prédire si une catégorie sémantique (e.g. voiture, chat) est présente dans l'image, à partir de son contenu visuel.

Plusieurs travaux préliminaires ont montré que pour analyser des images de scènes complexes, il est important d'apprendre des représentations localisées. Pour apprendre des représentations localisées, les approches classiques utilisent des annotations riches (e.g. boites englobantes, masques de segmentation) qui sont couteuses à obtenir, alors que dans le protocole standard de la classification d'images, ces annotations ne sont pas disponibles. Pour ne pas avoir à collecter et utiliser des annotations riches pendant l'apprentissage, nous nous sommes intéressé aux modèles d'apprentissage faiblement supervisé. L'idée est de modéliser les informations manquantes dans les données d'apprentissage avec des variables cachées. Dans cette thèse, nous proposons des modèles qui simultanément classifient et localisent les objets, en utilisant uniquement des labels globaux des images pendant l'apprentissage. Les contributions de cette thèse peuvent être décomposées en trois parties : l'architecture, l'optimisation et les résultats expérimentaux.

L'apprentissage faiblement supervisé permet de réduire le cout d'annotation, mais en contrepartie l'apprentissage est plus difficile. Le problème principal est comment agréger les informations locales (e.g. régions) en une information globale (e.g. image). Ce problème peut être vu comme un problème d'agrégation (pooling). La contribution principale de cette thèse est la conception de nouvelles fonctions de pooling pour l'apprentissage faiblement supervisé. En particulier, nous proposons une fonction de pooling "max+min", qui unifie de nombreuses fonctions de pooling, incluant plusieurs fonctions de pooling introduites dans cette thèse (SyMIL, MANTRA, WELDON). Nous prouvons que les régions "min" apportent une information complémentaire aux régions "max", et peuvent être vues comme des *negative evidence* de la classe, dans le cas de la classification multi-classes. Nous décrivons comment utiliser ce pooling dans le framework Latent Structured SVM ainsi que dans un réseau de neurones convolutifs. Finalement, nous présentons une nouvelle couche de transfert, qui permet de capturer plusieurs modalités par classe, pour enrichir le modèle et avoir une meilleure prédiction.

Du point de vue de l'optimisation, nos contributions sont multiples. Nous montrons que la fonction objective de SyMIL peut s'écrire comme une différence de fonctions convexes, et nous présentons deux solveurs : un pour résoudre le problème d'optimisation dans le primal, et un autre pour le résoudre dans le dual, et qui permet d'utiliser le kernel trick. Pour le modèle MANTRA, nous proposons un algorithme cutting-plane non-convexe pour résoudre le problème d'optimisation. Nous montrons aussi que MANTRA permet d'optimiser une métrique d'ordonnancement (ranking) comme l'Average Precision pendant l'apprentissage, s'il existe une méthode pour optimiser cette métrique dans le cas supervisé.

Expérimentalement, nous montrons l'intérêt nos modèles par rapport aux méthodes de l'état de l'art, sur dix bases de données standard de classification d'images, incluant ImageNet. En particulier, nous montrons que, à représentations égales, nos modèles qui se focalisent sur les régions discriminantes, sont meilleurs que des modèles état de l'art utilisant une représentation extraite sur toute l'image. Nous notons aussi qu'optimiser l'Average Precision durant l'apprentissage est pertinent sur les bases de données évaluées avec cette métrique. Pour comparer et analyser les forces et les faiblesses des fonctions de pooling couramment utilisées, nous proposons une étude expérimentale sur six bases de données. Finalement, nous montrons que notre modèle final, développé pour la classification, est aussi compétitif pour la localisation et la segmentation.

PUBLICATIONS

The material reported in this thesis was the subject of the following publications:

- Thibaut Durand, Nicolas Thome, Matthieu Cord, and Sandra Avila (2013). "Image Classification using Object Detectors". In: *IEEE International Conference on Image Processing (ICIP).*

- Thibaut Durand, David Picard, Nicolas Thome, and Matthieu Cord (2014). "Semantic Pooling for Image Categorization using Multiple Kernel Learning". In: *IEEE International Conference on Image Processing (ICIP).*

- Thibaut Durand, Nicolas Thome, Matthieu Cord, and David Picard (2014). "Incremental Learning of Latent Structural SVM for Weakly Supervised Image Classification". In: *IEEE International Conference on Image Processing (ICIP).*

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2015). "MANTRA: Minimum Maximum Latent Structural SVM for Image Classification and Ranking". In: *IEEE International Conference on Computer Vision (ICCV).*

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2016). "WELDON: Weakly Supervised Learning of Deep Convolutional Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

- Thibaut Durand*, Taylor Mordan*, Nicolas Thome, and Matthieu Cord (2017). "WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

The following publications are under review:

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2017b). "SyMIL: MinMax Latent SVM for Weakly Labeled Data". In: *IEEE Transactions on Neural Networks and Learning Systems (TNNLS) [Submission].*

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2017a). "Negative Evidence for Weakly Supervised Learning of Deep Structured Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence [Submission].*

CONTENTS

LIST OF FIGURES

LIST OF TABLES

2 RELATED WORK 9

3 SYMIL: MINMAX LATENT SVM FOR WEAKLY LABELED DATA 31

4 MANTRA: MINIMUM MAXIMUM LATENT STRUCTURAL SVM 51

5 WELDON: NEGATIVE EVIDENCE FOR WSL OF DEEP STRUCTURED MODELS 73

Chapter 1

INTRODUCTION

Contents

1.1 Context

Today, the field of computer vision has become ubiquitous in our society, with applications in image understanding, image search, medicine, drones, and self-driving cars. In particular, visual recognition is a central problem to computer vision research, and its goal is to automatically understand the contents of images. From robotics to information retrieval, many desired applications demand the ability to recognize objects, people, scenes, and activities. To train machines that are able to interpret the visual content of an image, the community has developed many algorithms and representations. The main tasks of visual recognition are image classification, detection and segmentation. In this thesis, we mainly focus on the image classification problem. The goal is to predict if a semantic category is present in the image according to its visual content. This is one of the fundamental problems in computer vision that has a large variety of practical applications.

The image classification problem becomes crucial because image and video data are one of the largest and fastest growing sources of information due to the popularization of digital photography (smartphones, digital cameras, etc.) coupled with the expansion of many social networks and mobile Internet access. For instance, there are 350 million photos uploads per day to Facebook[1] and 80 million to Instagram[2]. There are also 300 hours of video uploaded to YouTube every minute. In 2020, Cisco estimates that 82% of all the web traffic will be video, and that every second, a million minutes of video content will cross the network[3]. To exploit that immense and increasing collection of visual data, we need to annotate each image with semantically rich terms, which is the purpose of image classification. Most of the major technology companies, including Google, Facebook, Microsoft, IBM, Yahoo!, Twitter and Adobe, as well as a quickly growing number of

[1] http://www.businessinsider.com/facebook-350-million-photos-each-day-2013-9?IR=T

[2] https://maximizesocialbusiness.com/definitive-instagram-statistics-23286/

[3] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

start-ups, initiate research and development projects to deploy image understanding products and services.

For a human, the image classification (or image annotation) problem is easy. Unfortunately for a machine it is challenging, because the machine only "sees" numbers, without semantic meaning. The goal is to map all these numbers (i.e. the digital image) into one or several labels (Figure 1.1). This implies understanding complex semantic meanings based on an image's visual content. The main challenge is that low-level image representations (i.e. the pixels) are not discriminative enough to directly predict semantic-level concepts. (Smeulders et al. 2000) calls this problem *semantic gap*. Bridging the semantic gap requires an image classification model which is able to extract high-level representations from raw image pixels. Moreover a good image classification model must be invariant to the intra-class variations (i.e. appearance variations, see Figure 1.2), while simultaneously retaining sensitivity to the inter-class variations.

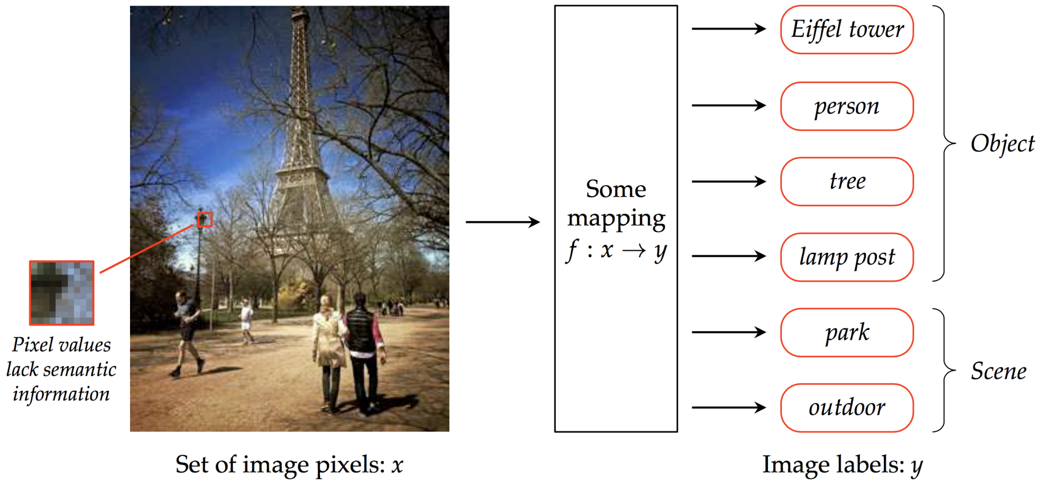


Figure 1.1.: The image annotation problem. The challenge of image annotation is to find a mapping that bridges the semantic gap between raw image pixels and semantic concepts, such as objects and scene categories. (Credit Hanlin Goh)

In the 2000s, most of the image representation models were based on handcrafted features. This approach requires careful engineering and considerable domain expertise to design a feature extractor that transforms the raw image pixels into a feature vector. The most popular model was the Bag of Words (BoW) approach. The BoW model is inspired from textual information retrieval (Salton et al. 1986). The intuition is to represent a document as a histogram of the occurrence rate of words in a dictionary. (Ma et al. 1999) was the first to adapt BoW for visual recognition in the *NeTra toolbox*, to represent an image as a bag of visual words based on color descriptors. This method was popularized by (Sivic et al. 2003), which employs Scale-Invariant Feature Transform (SIFT) local features.

Since AlexNet (Krizhevsky et al. 2012) won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky, Deng, et al. 2015) in 2012, deep Convolutional Networks (ConvNets) have become the state-of-the-art models for visual recognition. The ConvNets are widely used since 2012, but there were developed a long time ago. The first ConvNet is the *Neocognitron* (Fukushima 1980), which has analogies with brain, and was inspired

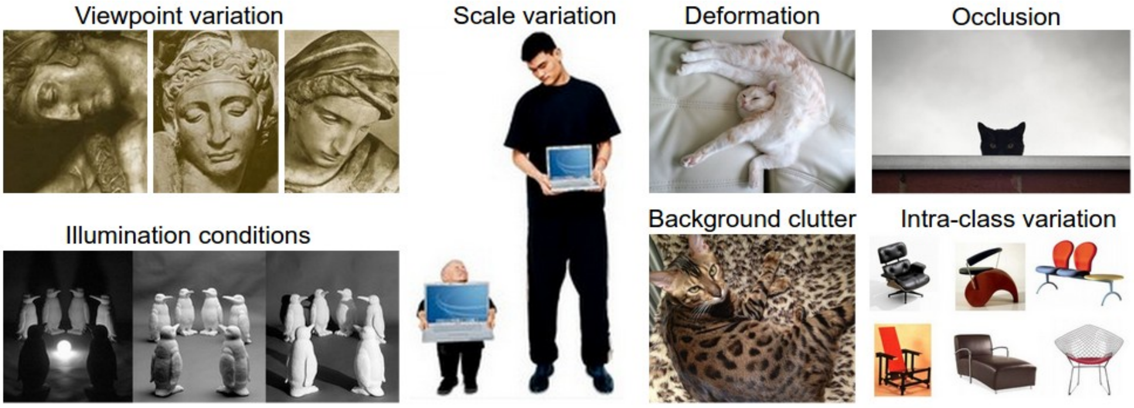


Figure 1.2.: Challenges in image classification. (Credit http://cs231n.github.io/)

by the receptive field representation proposed by (Hubel et al. 1962). Then the LeNet architecture (LeCun, B. Boser, et al. 1989; LeCun, L. Bottou, et al. 1998) developed in the 1980s and 1990s was the first successful application of ConvNets for digit recogniton. But the big ConvNet success was only possible a few decades later, and can be explained by two factors:

- a large amount of available labeled data to avoid over fitting. The ILSVRC classification dataset has 1.2M training images distributed in 1,000 classes.
- the use of Graphics Processing Units (GPUs), which enables training networks 10 or 20 times faster than with Central Processing Units (CPUs).

A deep ConvNet applies several stages of non-linear transformations to an input image, where each stage transforms its input representation into a higher-level one. It learns a hierarchy of representations with an increasing level of abstraction. Most of the stages are composed of convolutions and non-linear activation functions. Contrarily to handcrafted representations, the parameters of these layers are learned from labeled data by optimizing a task-specific objective function.

1.2 Motivations

Designing a robust image classification model requires techniques from both computer vision and machine learning fields. While computer vision focuses on representing the image content, machine learning creates statistical models that relate the image content to the semantic annotations.

In this manuscript, we are interested in learning localized representations for image classification. An important drawback of BoW with Spatial Pyramid Matching (SPM) and ConvNet representations is that they are limited by the lack of ability to be spatially invariant to the input image. Because of their design, these representations are robust to local deformations, but not to strong deformations. Moreover, they encode the whole image, so the final representation contains both discriminative (e.g. objects) and non-discriminative (background) information. Encoding background regions in the representation decreases

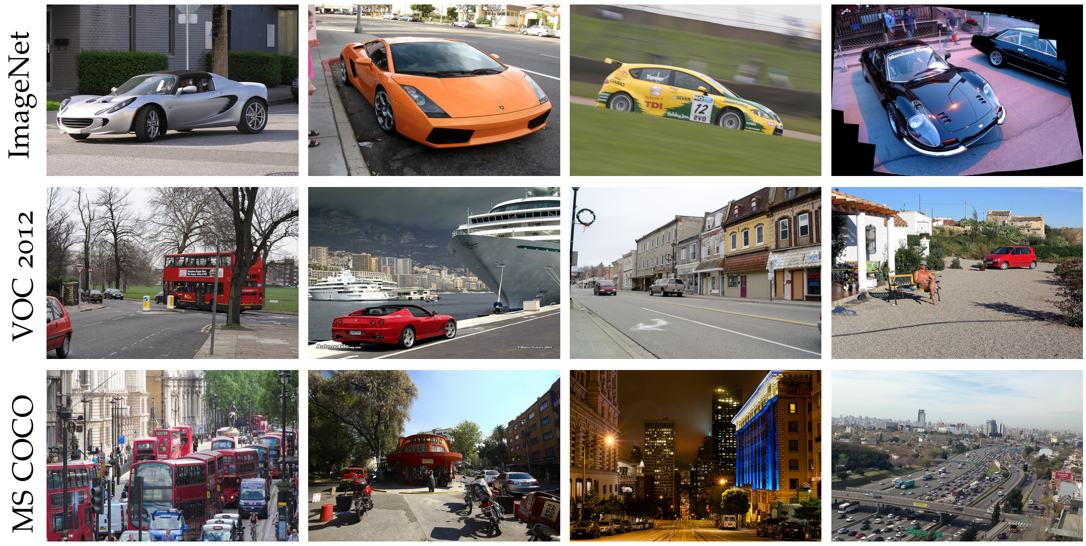


Figure 1.3.: Examples of *car* images from ImageNet, PASCAL VOC 2012 and MS COCO.

its discriminative power, because background regions introduce "noise". An approach to build translation and scale invariant representations is to use only the object regions to compute the final representation. The intuition is that if we know where to look, recognizing the objects should be easier. (Russakovsky, Y. Lin, et al. 2012) reports a proof of concept, where the authors observe a large improvement of classification performances when the representations are computed on the ground truth object bounding boxes. This validates the fact that learning localized representations, which are spatially invariant, is relevant for image classification.

Similarly, (Oquab et al. 2014) shows that using bounding box supervision is highly beneficial for object classification in cluttered and complex scenes. Indeed, most deep ConvNet architectures are learned on ImageNet, where the objects are centered in the image (first row of Figure 1.3). To use pre-trained deep ConvNet architecture on complex images with non-centered objects as in PASCAL VOC and MS COCO datasets (second and third rows of Figure 1.3), (Oquab et al. 2014) uses bounding box supervision to train object-centric classifiers, and applies the classifiers by searching over different locations in the images. Another alternative is to use pre-trained object detectors, e.g. (N. Zhang et al. 2014) aligns parts with poselet detectors to make human attribute recognition much more efficient. We now present two of our preliminary works (Durand, Thome, Cord, and Avila 2013; Durand, Picard, et al. 2014) based on object detectors, which motivate the approach developed in this thesis.

In (Durand, Thome, Cord, and Avila 2013), we use a set of pre-trained object detectors to build a discriminative and compact image representation. As in Object Bank (OB) (L.-J. Li, Su, et al. 2014), we represent an image based on its response to a large number of pre-trained object detectors. The whole pipeline of our approach is shown in Figure 1.4. The first step uses object detectors to generate heatmaps, where each score indicates whether or not there is an object of interest at the given position and scale in the image.

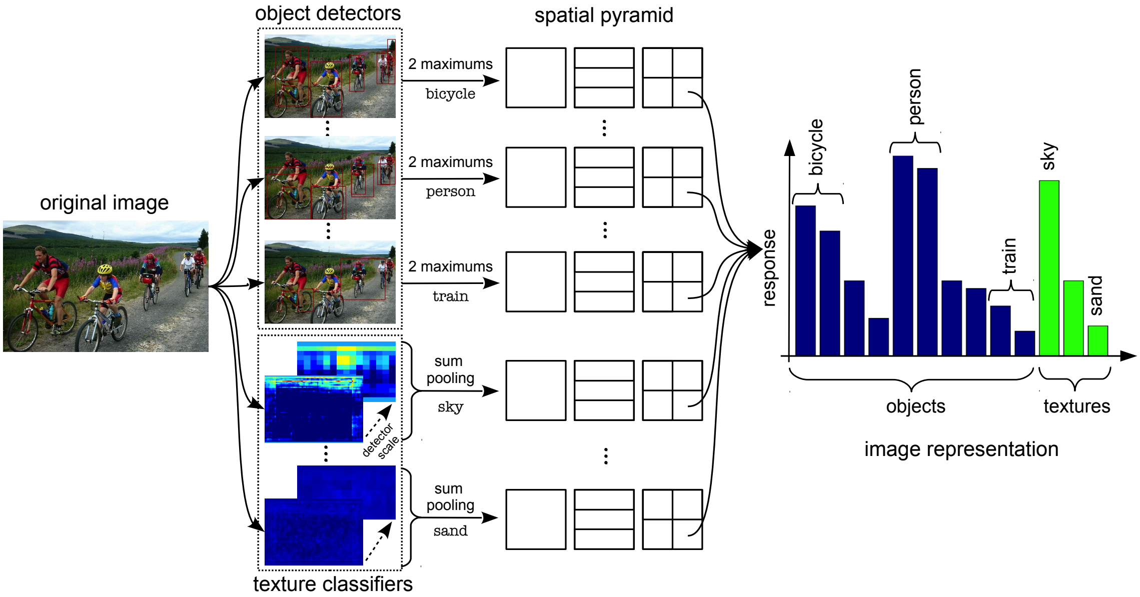


Figure 1.4.: Pipeline of our image representation using object detectors.

We then pool the heatmaps to build a vector representation. This pooling is applied to the different regions of the spatial pyramid, to keep spatial information.

The previous model uses a set of pre-trained object detectors to represent the content of an image, but the final representation is not invariant to the layout of the objects in the image. To address this problem, we use in (Durand, Picard, et al. 2014) a set of pre-trained object detectors for taking into account the spatial layout, and to align similar semantic regions. The whole pipeline is shown in Figure 1.5. For each category, we use an object detector to predict the region with the maximum likelihood of containing the object. As many regions represent "noise" (because only a few objects are present in an image), we select relevant regions by using a ℓ_1-Multiple Kernel Learning (MKL) (Rakotomamonjy et al. 2008).

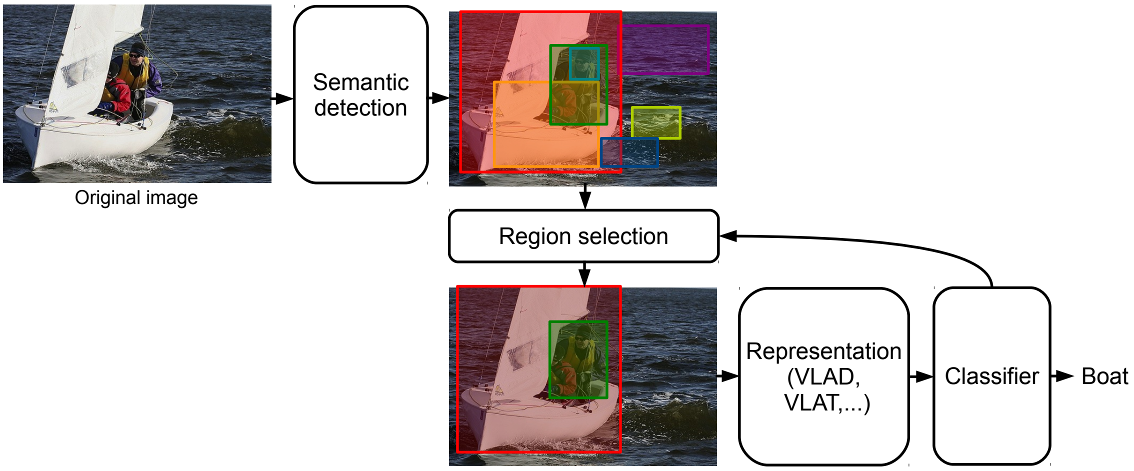


Figure 1.5.: Semantic pooling pipeline for image classification.

Discussion The methods based on bounding box supervision or object detectors previously presented give better performances than standard classification models based on global representations. This also validates that localized representations are relevant for image classification. Training object detectors requires bounding box annotations, which are not available in the standard image classification setting. The bounding box annotation gives information about the size and the location of the object, whereas image-level annotation only indicates if at least one object is present. Moreover, the annotation process for object bounding boxes is usually resource intensive and difficult to scale up (Bearman et al. 2016). An alternative is to use Weakly Supervised Learning (WSL).

1.3 Contributions and Outline

The Weakly Supervised Learning (WSL) is a framework where the model learns to capture aspects of the data that are not labeled in the training data. Learning from weakly labeled data is a very important problem that covers several theoretical and practical aspects towards the development of powerful learning machines. On the one hand, relaxing the requirement of expensive manual and accurate annotations of training data offers the possibility to build large scale databases at reasonable cost. For example, in the computer vision field, annotating images with a global label makes it possible to build databases containing several millions of examples, whereas annotations at the pixel level (e.g. segmentation mask) are much more expensive, which explains why only moderate-size datasets are available (Bearman et al. 2016). On the other hand, handling weakly labeled data generally requires to expand the representation space with latent variables to model hidden factors and compensate for the weak supervision. In this manuscript, we focus on the WSL setting where the model simultaneously classifies and locates objects given only image-level annotations for training.

In brief, the weak supervision significantly reduces the cost of full annotation in many recognition tasks, but it makes learning and recognition more challenging. The key issue of WSL is to find how to pool the local (i.e. region) scores into global (i.e. image) score. The central contribution of this thesis is the design of new pooling functions for WSL. Based on these pooling functions, we propose deep ConvNet architectures to perform image classification and learn discriminative regions from images annotated with a global label.

- SYMIL: MINMAX LATENT SVM FOR WEAKLY LABELED DATA. Chapter 3 introduces a new model SyMIL for binary bag classification, based on a symmetric pooling. Unlike Multiple-Instance Learning (MIL) approaches, the SyMIL model seeks discriminative instances in both positive and negative bags. We validate our SyMIL model on different kind of data: image, text and molecule. We also analyze the selected instances of both symmetric and asymmetric approaches.

- MANTRA: MINIMUM MAXIMUM LATENT STRUCTURAL SVM. Chapter 4 introduces a novel WSL framework which extends SyMIL (Chapter 3) to structured output prediction. Our new structured output latent variable model, called MANTRA, is based on negative evidence pooling: the prediction function relies on a pair of latent variables that provides positive (resp. negative) evidence for a given category. We propose two instantiations of our model: multi-class classification and ranking. The content of this chapter is based on (Durand, Thome, and Cord 2015).

- WELDON: NEGATIVE EVIDENCE FOR WSL OF DEEP STRUCTURED MODELS. Chapter 5 describes how to integrate the negative evidence pooling of Chapter 4 in a deep architecture, to have an end-to-end training model. The architecture design enables an efficient transfer learning and fine-tuning. This chapter also introduces a spatial pooling function, based on multiple regions, which generalizes the MANTRA pooling. The content of this chapter is based on (Durand, Thome, and Cord 2016).
- WILDCAT: SPATIAL AND CLASS-WISE POOLING. Chapter 6 extends the work presented in Chapter 5, this time focusing on the network architecture and the spatial pooling. We propose a new multi-map transfer to learn several modalities per class, and a new spatial pooling function, which generalizes the WELDON pooling of Chapter 5. The content of this chapter is based on (Durand* et al. 2017), which is a joint work with Taylor Mordan.

Before presenting our contributions, we provide relevant background for both image representation with deep ConvNets and WSL in Chapter 2. In Chapter 7, we finally summarize our contributions and suggest directions for future work.

Chapter 2

RELATED WORK

Contents

In this manuscript, we are interested in Weakly Supervised Learning (WSL) for visual recognition. We first present different strategies to extract image representation. We then introduce standard WSL methods, which automatically learn the locations of the objects with image-level labels only. Finally, we give an overview of deep convolutional network architectures for WSL.

Notation We note $\boldsymbol{x} \in \mathcal{X}$ an input and $\boldsymbol{y} \in \mathcal{Y}$ an output, where $\mathcal{X}$ is the *input space* and $\mathcal{Y}$ is the *output space*. We note $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\}_{i=1}^N \in (\mathcal{X} \times \mathcal{Y})^N$ the training dataset of N examples made up of independent and identically distributed (i.i.d.) samples from a data generating distribution p_{data}, i.e. $(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) \sim p_{data}$ for all i. We note $\boldsymbol{w}$ the vector of parameters.

2.1 Visual Representations

In this section, we present two strategies to extract image representations: Bag of Words (BoW) and deep architectures. The BoW approach was the state-of-the-art model for image classification in the 2000s. This is a handcrafted representation, i.e. it is manually designed and relies on expert knowledge. Since 2012, the Convolutional Network (ConvNet) is becoming the state-of-the-art model for image classification. Contrary to the BoW representations, the ConvNet representations are learned from training data.

2.1.1 Handcrafted Representation: Bag of Words

The Bag of Words (BoW) model is inspired from textual information retrieval (Salton et al. 1986). The concept is to represent a document as a histogram of occurrence rates of words from a dictionary. (Ma et al. 1999) was the first to adapt BoW for visual recognition in the *NeTra toolbox*, to represent an image as a bag of visual words. Then, (Fournier et al. 2001) extended this approach with Gabor filters. This method was popularized by (Sivic et al. 2003), which employs Scale-Invariant Feature Transform (SIFT) local features. To represent an image, the model performs a series of three consecutive steps:

1. *Local feature extraction*. The local descriptors are extracted uniformly across the image using a uniform grid, which may be overlapping and have multiple scales. The most popular local descriptor is the SIFT (Lowe 2004), because it is invariant to various image transformations, such as geometric and photometric transformations, which are essential when addressing image classification problems.

2. *Coding*. The coding step encodes the local descriptors as a function of the dictionary visual words, and outputs visual codes. To learn a visual dictionary, the most popular approach for image categorization is the *k-means* clustering algorithm (Lloyd 1982). The historical coding function is the hard assignment coding. To reduce the quantization errors and ambiguity resulting from the hard quantization, (Gemert et al. 2010) proposes the soft assignment. To keep more information, several methods propose to encode the distance in vectorial form: Fisher Vectors (FV) (Perronnin et al. 2007), Super-Vector Coding (SVC) (X. Zhou et al. 2010), Vector of Locally Aggregated Descriptors (VLAD) method (Jégou et al. 2010) and its generalization Vector of Locally Aggregated Tensors (VLAT) (Picard et al. 2011). The visual dictionary contains visual words which are used to project local descriptors into another feature space for the subsequent step in the BoW pipeline.

3. *Pooling*. The pooling step constructs a single vectorial representation (or signature) from the set of local visual codes across the whole image. The standard pooling function is the average pooling (or sum pooling) (Sivic et al. 2003). Another popular method is the max pooling. (Boureau et al. 2010) observes that the best pooling function may be an operation between average and max pooling. To incorporate spatial information, (Lazebnik et al. 2006) introduces the Spatial Pyramid Matching (SPM). In another way, (Avila et al. 2012) introduces BossaNova (BN), where the standard scalar pooling is replace by a vectorial pooling to capture higher-order statistics. The image representation has the same dimensionality across all the images of possibly different sizes. Then, the final representation is normalized.

The global aim is gaining invariance to nuisance factors (locations of the objects, changes in the background, small changes in appearance, etc.), while preserving the discriminating power of the local descriptors. To predict labels, the common approach is to use these representations to train a classifier with supervised learning algorithms. A classifier is a function that maps the representations to one of the possible categories. The most popular classifiers used in computer vision are k-Nearest Neighbors (k-NN) (Cover et al. 1967), Decision Trees (Breiman et al. 1984), Support Vector Machine (SVM) (B. E. Boser et al. 1992; Vapnik 1995) and neural networks, which are presented in the next section.

2.1.2 Learned Representation: Deep ConvNets

In 2012, AlexNet, a deep Convolutional Network (ConvNet) architecture proposed by Krizhevsky et al. (Krizhevsky et al. 2012), has emerged as a competitive method for classifying large-scale image datasets with huge amounts of training data, convincingly winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky, Deng, et al. 2015). In this section, we briefly present the ConvNet, then how to learn it and finally its applications to visual recognition.

2.1.2.1 Convolutional Networks

The *Convolutional Network (ConvNet)* (also known as Convolutional Neural Network or CNN) is a type of *feedforward network* that uses convolution in at least one of its layers. This model is called feedforward because information flows strictly in the forward direction, from the input units, through hidden units, if any, and finally to output units. When feedforward neural networks are extended to include feedback connections, they are called *Recurrent Neural Networks (RNNs)* (Rumelhart et al. 1986).

The convolution layer is the core layer of deep ConvNets. The convolution exploits spatially local correlation by enforcing a local connectivity pattern between units of adjacent layers. Standard ConvNet architectures are built by stacking convolutional layers followed by non-linearities, and possibly introducing pooling layers to control the computational complexity of the architecture.

ConvNets are typically represented by composing together many different functions or layers. For example, a feedforward network with n layers can be written

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = f_{\boldsymbol{w}_n}(f_{\boldsymbol{w}_{n-1}}(\dots f_{\boldsymbol{w}_2}(f_{\boldsymbol{w}_1}(\boldsymbol{x})))) \tag{2.1}$$

where $\boldsymbol{x}$ is the input, $\boldsymbol{w} = [\boldsymbol{w}_1, \dots, \boldsymbol{w}_n]$ is the vector of ConvNet parameters, $f_{\boldsymbol{w}_k}$ is the k-th layer and $\boldsymbol{w}_k$ is the vector of parameters of the k-th layer. The model is associated with a directed acyclic graph describing how the layers are composed together. We note $\boldsymbol{y}$ the output of the network $\boldsymbol{y} = f_{\boldsymbol{w}}(\boldsymbol{x})$, and $\boldsymbol{h}_k = f_{\boldsymbol{w}_k}(\boldsymbol{h}_{k-1})$ the output of the k-th layer, and $\boldsymbol{h}_0 = \boldsymbol{x}$.

The goal of a ConvNet is to approximate some function $f^\star$. For example, for a classification problem, the ConvNet maps an input to a category. In general, for an input-output pair $(\boldsymbol{x}, \boldsymbol{y})$, a ConvNet defines a mapping $\boldsymbol{y} = f_{\boldsymbol{w}}(\boldsymbol{x})$, and learns the value of the parameters $\boldsymbol{w}$ that results in the best function approximation of $f^\star$ with respect to a loss function $\mathcal{L}$ such that $\mathcal{L}(\boldsymbol{y}^\star, \boldsymbol{y}) > 0$ measures the disagreement between a ground-truth label $\boldsymbol{y}^\star$ and a output $\boldsymbol{y}$.

2.1.2.2 Learning ConvNets with SGD & Back-propagation

The problem of learning ConvNet reduces to an optimization problem of the general form $\widehat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \mathcal{J}(\boldsymbol{w})$ where $\mathcal{J}$ usually combines the empirical loss $\mathcal{L}$ (Vapnik 1991) of all training examples and a regularization penalty $\mathcal{R}$

$$\mathcal{J}(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{y}_i^\star, f_{\boldsymbol{w}}(\boldsymbol{x}_i)) + \mathcal{R}(\boldsymbol{w}) \tag{2.2}$$

The regularization term is used to control the complexity of the model and to prevent overfitting. It can also encode some *a priori* about the function $f_{\boldsymbol{w}}$. The ℓ_2 regularization ($\mathcal{R}(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$) is a common regularizer for ConvNets. We now explain how to solve this optimization problem with Stochastic Gradient Descent (SGD) and back-propagation. Solving this optimization problem is difficult, because the loss function is a non-convex function of the model parameters. The gradient descent requires to compute the gradient of the function $\mathcal{J}$, so in the following of this section, we assume that $f_{\boldsymbol{w}}$, $\mathcal{L}$ and $\mathcal{R}$ are differentiable with respect to the model parameters $\boldsymbol{w}$. We assume that we can calculate the Jacobian matrices $\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{h}_{k-1}}$ and $\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{w}_k}$ for each layer $k \in \{1, \dots, n\}$.

Stochastic gradient descent Stochastic Gradient Descent (SGD) and its variants are probably the most used optimization algorithms for machine learning in general and for deep learning in particular. The gradient descent method is a common way to minimize an objective function $\mathcal{J}$ of parameters $\boldsymbol{w} \in \mathbb{R}^d$ by updating the parameters in the opposite direction of the gradient of the objective function $\frac{\partial \mathcal{J}(\boldsymbol{w})}{\partial \boldsymbol{w}}$ w.r.t. the parameters. Computing the exact gradient $\frac{\partial \mathcal{J}(\boldsymbol{w})}{\partial \boldsymbol{w}}$ is very expensive because it requires evaluating the model on every example in the entire dataset. To alleviate this problem, the SGD method estimates the gradient by randomly sampling a minibatch of training examples. The SGD algorithm alternates between two steps:

1. estimating the gradient $\frac{\partial \mathcal{J}(\boldsymbol{w})}{\partial \boldsymbol{w}}$ on a minibatch of m training examples $\{(\boldsymbol{x}_{I(1)}, \boldsymbol{y}^\star_{I(1)}), \dots, (\boldsymbol{x}_{I(m)}, \boldsymbol{y}^\star_{I(m)})\}$

$$\frac{\partial \mathcal{J}(\boldsymbol{w})}{\partial \boldsymbol{w}} \approx \frac{\partial \widehat{\mathcal{J}}(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}}\left[\frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\boldsymbol{y}^\star_{I(i)}, f_{\boldsymbol{w}}(\boldsymbol{x}_{I(i)})) + \mathcal{R}(\boldsymbol{w})\right] \tag{2.3}$$

2. updating the parameters $\boldsymbol{w}$ in the direction of the negative gradient

$$\boldsymbol{w} = \boldsymbol{w} - \eta \frac{\partial \widehat{\mathcal{J}}(\boldsymbol{w})}{\partial \boldsymbol{w}} \tag{2.4}$$

   where η is the learning rate or step size.

The learning rate η is a critical hyper-parameter. If it is too high, the optimization may not converge or even diverge. If it is too low, learning will take too long. The most important property of SGD is that computation time per update does not grow with the number of training examples. This allows convergence even when the number of training examples becomes very large.

Several approaches have been proposed to accelerate the convergence of SGD. (Polyak 1964; Sutskever et al. 2013) introduce a momentum to accumulate an exponentially decaying moving average of past gradients and to keep moving in their directions. As the learning rate is a crucial hyper-parameter, different algorithms have been proposed to automatically adapt it e.g. AdaGrad (Duchi et al. 2011), Adadelta (M. D. Zeiler 2012), RMSProp (Tieleman et al. 2012) and Adam (Kingma et al. 2014).

Back-propagation To train a feedforward neural network $f_{\boldsymbol{w}}$ with SGD, we must compute the gradient $\frac{\partial \widehat{\mathcal{J}}(\boldsymbol{w})}{\partial \boldsymbol{w}}$ (Equation 2.3). The difficult step is the computation of the gradient $\frac{\partial \mathcal{L}(\boldsymbol{y}^\star, f_{\boldsymbol{w}}(\boldsymbol{x}))}{\partial \boldsymbol{w}}$ for a given example $\boldsymbol{x}$, and its output $\boldsymbol{y}^\star$. We need to compute the gradient w.r.t. the parameters in each layer. Using the chain rule, the final gradient is the matrix product of all the Jacobians

$$\frac{\partial \mathcal{L}(\boldsymbol{y}^\star, f_{\boldsymbol{w}}(\boldsymbol{x}))}{\partial \boldsymbol{w}_l} = \frac{\partial \mathcal{L}(\boldsymbol{y}^\star, f_{\boldsymbol{w}}(\boldsymbol{x}))}{\partial \boldsymbol{h}_n} \left(\prod_{k=l+1}^{n} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{h}_{k-1}} \right) \frac{\partial \boldsymbol{h}_l}{\partial \boldsymbol{w}_l} \tag{2.5}$$

The back-propagation (Rumelhart et al. 1986) is a recursive algorithm that computes the chain rule with a specific order of operations that is highly efficient.

2.1.2.3 Modern ConvNet Architectures for Image Classification

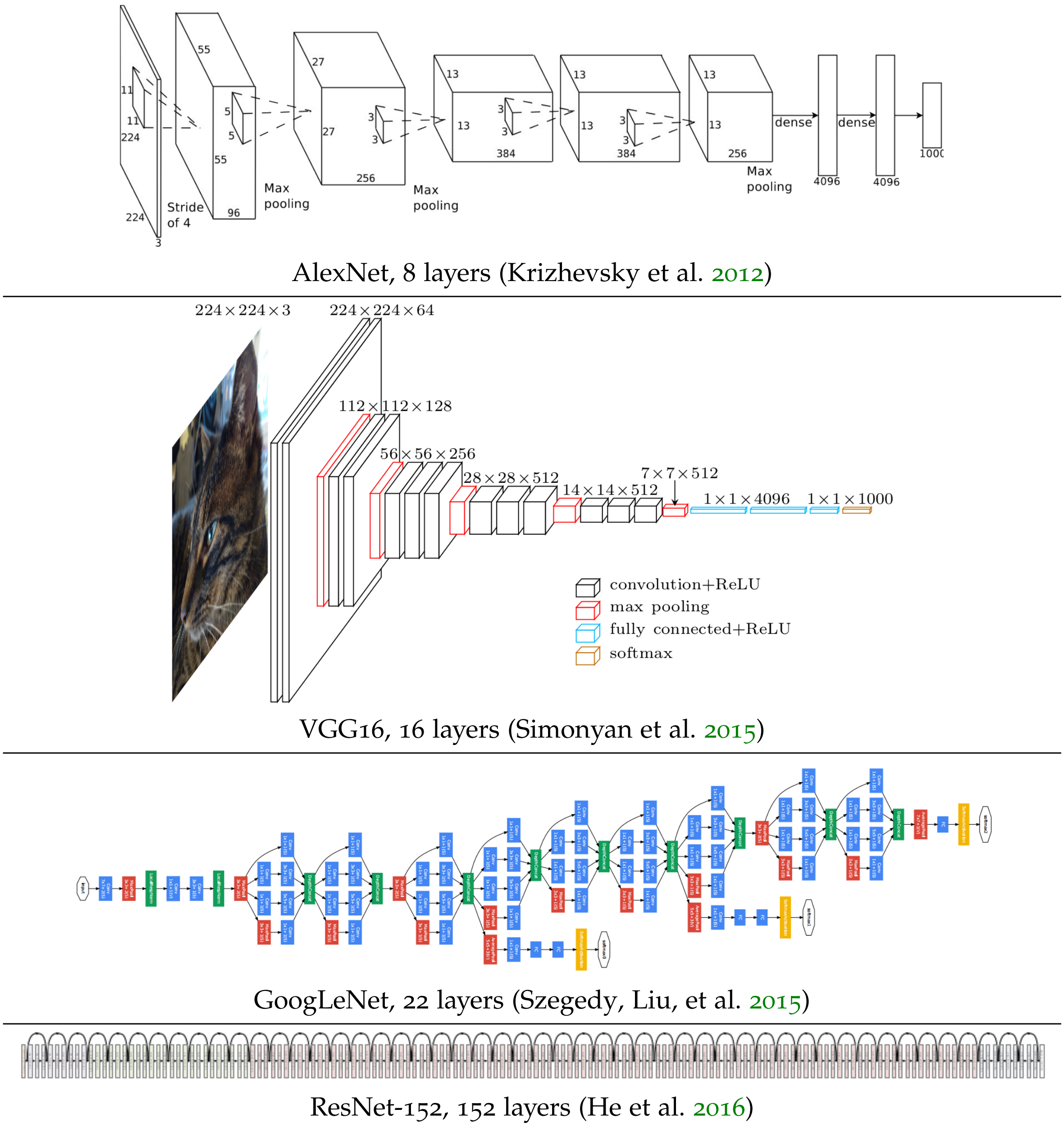
We now present the most important ConvNet architectures. Since 2012, ConvNets have become popular with the large win of AlexNet (Krizhevsky et al. 2012) to the competition ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012[1]. AlexNet significantly outperformed the second runner-up based on handcrafted representation (top 5 error of 16% compared to runner-up with 26% error). The network consists of eight learned layers (five convolutional and three fully-connected) that map image pixels to the semantic-level (see Figure 2.1). Then, similar architectures were proposed to improve AlexNet: ZF Net (M. D. Zeiler and Fergus 2014) (winner of the ILSVRC 2013 classification challenge), Overfeat (Sermanet et al. 2014), `vgg-s`, `vgg-m`, `vgg-f` (Chatfield et al. 2014) and CaffeNet (Jia et al. 2014).

In 2014, Simonyan and Zisserman show that the depth of the network is a critical component for good performance (Simonyan et al. 2015). They introduced the very deep VGG16, a 16 layers ConvNet architecture (see Figure 2.1). A downside of the VGG16 is that it is more expensive to evaluate and uses a lot more memory and parameters (138M).

To reduce the number of parameters, (Szegedy, Liu, et al. 2015) introduces the GoogLeNet, that won the ILSVRC 2014 classification challenge. The main contribution is the development of the *Inception* module that dramatically reduces the number of parameters in the network (7M, compared to AlexNet with 60M). GoogLeNet (22 layers) is deeper than VGG16 and uses a Global Average Pooling (GAP) instead of fully-connected layers at the top of the ConvNet, eliminating a large amount of parameters. Except the fully-connected of the last layer used for classification, all the learned layers are convolution layers Figure 2.1.

The next step is to learn deeper models than VGG16 and GoogLeNet by adding more layers. But this is not possible in practice because of the vanishing/exploding gradients

[1] http://www.image-net.org/challenges/LSVRC/2012/



AlexNet, 8 layers (Krizhevsky et al. 2012)

VGG16, 16 layers (Simonyan et al. 2015)

GoogLeNet, 22 layers (Szegedy, Liu, et al. 2015)

ResNet-152, 152 layers (He et al. 2016)

Figure 2.1.: Standard deep ConvNet architectures in computer vision.

problem (Bengio et al. 1994; Glorot et al. 2010). To overcome this problem and to learn ConvNets with several hundreds of layers, (He et al. 2016) builds *residual networks* (ResNet) by using residual blocks. The residual block uses shortcut connection that skips one or more layers. The authors also show that the residual blocks combined with batch normalization (Ioffe et al. 2015) make the training easier. Like GoogLeNet, the only learned layers are convolution layers. The ResNets won both ILSVRC & MS COCO[2] 2015 competitions.

[2] http://mscoco.org

Learning deep architecture The evolution of the deep ConvNet architectures shows that the network architecture is very important to perform accurate predictions. The previous architectures were manually designed with some *a priori*. We went from *feature engineering* (e.g. BoW) to *network engineering*. The next step is to automatically learn the network architecture. Note that learning deep network architecture is a challenging task because the number of possible network architectures increases exponentially with the number of layers in the network. Recently, some works propose different methods to learn automatically the network architecture. (Srivastava et al. 2015) introduces the *highway networks*, which utilize a learnable gating mechanism to automatically learn some connections between layers. (Kulkarni, Zepeda, et al. 2015) proposes a method to automatically select the number of units in fully-connected layers, by using a ℓ_1 regularization. So far, the most general approach is the *convolutional neural fabrics* (Saxena et al. 2016). The authors propose a fabric that embeds an exponentially large number of architectures. The fabric consists of a 3D trellis that connects response maps at different layers, scales, and channels with sparse local connectivity patterns.

2.1.2.4 Transfer Learning & Applications of Deep ConvNets

Training an entire ConvNet from scratch (with random initialization) on large dataset is difficult, because it requires a lot of labeled training images and computational resources (e.g. GPU). For instance, standard ConvNets take about two weeks to train on ImageNet, with only one GPU. To have models easily available, some people release pre-trained models. A key success of the ConvNets is that the learned representations (or features) on ImageNet are both discriminative and generic, so they can be efficiently *transfered* to other (small) datasets.

Transfer learning We now present the two common transfer learning strategies.

- *ConvNet as feature extractor*. This strategy is to use the representations of a ConvNet pre-trained on ImageNet to train a classifier for the new dataset. To achieve this, we take a ConvNet pre-trained on ImageNet, and we remove the last fully-connected layer (also called classification layer) because it represents the 1000 class scores of ImageNet. Then we treat the rest of the ConvNet as a fixed feature extractor for the new dataset. These features are called *deep features* (or CNN codes). It is important for performance that these features are extracted after their activation function. Once the deep features for all the images are extracted, we train a classifier (e.g. SVM) on the new dataset (Chatfield et al. 2014; A. Razavian et al. 2014).

- *Fine-tuning*. The intuition is to learn a ConvNet starting from a "good" initialization, i.e. initialized with the weights of a model pre-trained on ImageNet. The common way to achieve this, is to replace and retrain the classifier layer on top of the ConvNet on the new dataset, but to also fine-tune the weights of the pre-trained network by continuing the back-propagation. It is possible to fine-tune all the layers of the ConvNet, or it is possible to keep some of the earlier layers fixed (due to overfitting concerns) and only fine-tune some higher-level layers of the network (Azizpour, A. S. Razavian, et al. 2016).

Experimentally, we observe that the fine-tuning strategy has better performances than the feature extractor strategy. Unfortunately, the fine-tuning strategy requires computational resources because it needs to evaluate several forward and backward passes. On the contrary, the feature extractor strategy requires only one forward pass for each image.

Applications Since 2012, the ConvNets have become the state-of-the art model for large scale classification. They won the ILSVRC 2012, 2013, 2014, 2015, 2016 challenges (Krizhevsky et al. 2012; M. Zeiler et al. 2013; Szegedy, Liu, et al. 2015; He et al. 2016). They also show that they can be efficiently transfered to small and medium size datasets (Azizpour, A. S. Razavian, et al. 2016). The learned representation are generics and are state-of-the art methods for different classification problems such as object classification (Chatfield et al. 2014; Oquab et al. 2014; He et al. 2014; Oquab et al. 2015), scene classification (Gong et al. 2014; B. Zhou et al. 2014; Bolei Zhou, Khosla, et al. 2016), action recognition (Chéron et al. 2015; Georgia Gkioxari et al. 2015; Diba, Pazandeh, et al. 2016), fine-grained classification (Y. Wang et al. 2016; Huang et al. 2016; Reed et al. 2016), food recognition (Xin Wang et al. 2015; F. Zhou et al. 2016), etc.

The ConvNets have been successfully applied on other standard computer vision applications such as object detection, semantic segmentation. The goal of the object detection task is to predict a bounding box for each object present in the image. The most popular models are R-CNN (R. Girshick et al. 2014), Fast R-CNN (Ross Girshick 2015), Faster R-CNN (Ren et al. 2015), R-FCN (Dai, Y. Li, et al. 2016), MultiPathNet (Zagoruyko et al. 2016) and YOLO (Redmon et al. 2016). The ConvNets are also used for semantic segmentation, where the goal is to predict a label for each pixel of the image. Different architectures have been proposed to predict segmentation masks such as the Fully Convolutional Network (FCN) (Long et al. 2015), DeepLab (L. Chen et al. 2015) and CRFasRNN (Zheng et al. 2015). Similarly, (Pedro O Pinheiro et al. 2015) introduces DeepMask to predict class-agnostic segmentation masks and it can be coupled with SharpMask (Pedro O. Pinheiro, T.-Y. Lin, et al. 2016) to refine the masks.

Today, the ConvNet-based models yield state-of-the-art performances in many areas of computer vision: object tracking (L. Wang et al. 2015; H. Li et al. 2016), image retrieval (Paulin et al. 2015; Arandjelović et al. 2016), instance segmentation (Z. Zhang et al. 2016; Dai, He, et al. 2016), contour detection (J. Yang et al. 2016), pose estimation (Toshev et al. 2014; G. Gkioxari et al. 2016), optical flow estimation (Dosovitskiy et al. 2015), Visual Question Answering (VQA) (H. Xu et al. 2016; Fukui et al. 2016) etc.

2.2 Weakly Supervised Learning

Despite excellent performances for image classification, deep ConvNets carry limited invariance properties, i.e. a small shift invariance through pooling layers. This lack of spatial invariance hinders effective transfers on target datasets with strong variations w.r.t. the source datasets. For example, we note a large shift between ImageNet, which essentially contains centered objects, and other common datasets, e.g. PASCAL VOC or MS COCO, containing several objects and strong scale and translation variations (Figure 1.3).

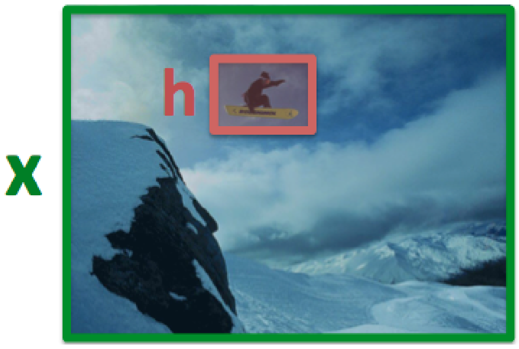


Figure 2.2.: Illustration of MIL for image classification: $\boldsymbol{x}$ is the image and $\boldsymbol{h}$ is a region of the image.

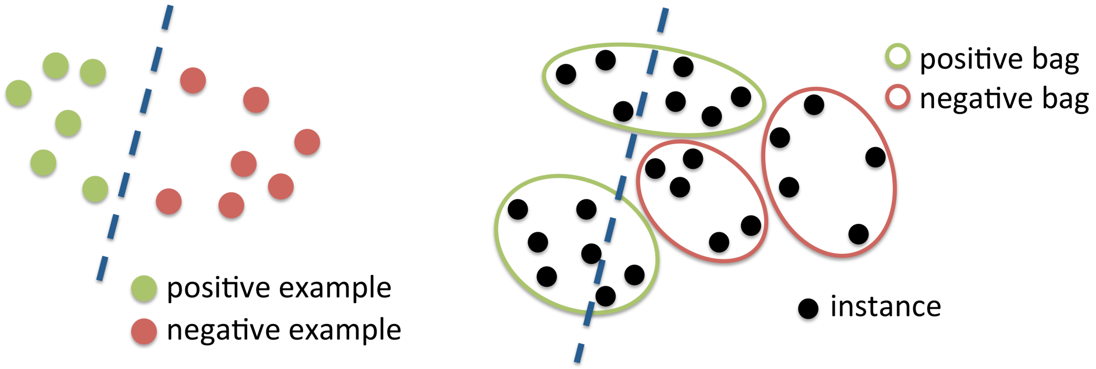
To optimally perform domain adaptation in this context, it becomes necessary to align informative image regions. To achieve region alignment, several methods use richer annotations (e.g. bounding boxes annotations (Oquab et al. 2014; R. Girshick et al. 2014; N. Zhang et al. 2014), segmentation masks (Long et al. 2015)) than image-level labels. However, these rich annotations rapidly become costly to obtain, making the development of Weakly Supervised Learning (WSL) models appealing. WSL methods allow to capture unobserved information about the data that is not labeled in the training data. In this thesis, we focus on WSL to learn spatially invariant representations, because we can explicitly align image regions. We now present Multiple-Instance Learning (MIL), which is the most popular WSL framework for computer vision, and WSL methods for structured outputs.

2.2.1 Multiple-Instance Learning & its Extensions

The Multiple-Instance Learning (MIL) is a popular framework for bag classification with binary labels, i.e. the output space is $\mathcal{Y} = \{-1, +1\}$. The input $\boldsymbol{x}$ is represented as a bag of instances $\boldsymbol{x} = \{\boldsymbol{x}^h\}_{h \in \mathcal{H}(\boldsymbol{x})}$, where $\boldsymbol{x}^h$ is the h-th instance, and $\mathcal{H}(\boldsymbol{x})$ is the set of instance indexes in bag $\boldsymbol{x}$. $\mathcal{H}(\boldsymbol{x})$ depends on the bag $\boldsymbol{x}$, because every bag has its own number of instances. To simplify the notation in the following, we write $\mathcal{H}$ without the bag dependency when there is no ambiguity. We note $\Phi(\boldsymbol{x}, h) \in \mathbb{R}^d$ the vectorial representation of the h-th instance of bag $\boldsymbol{x}$. For example in Figure 2.2, the bag is the image, and the instances are regions of the image. Because it may be expensive to assign a reliable label to each training instance, only the bag has a label, i.e. we know if an object is present or absent, but we do not know about its location. The important difference to standard supervised learning is that the label is given for the whole bag, but not for the individual instances in a bag (Figure 2.3). Given a training dataset of labeled bags $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\}_{i=1}^N \in (\mathcal{X} \times \mathcal{Y})^N$, the goal is to learn a bag classification function $f_{\boldsymbol{w}} : \mathcal{X} \to \mathcal{Y}$, where $\boldsymbol{w}$ are the model parameters. We now present the Multiple-Instance Learning (MIL) and its extensions.

2.2.1.1 Multiple-Instance Learning

The Multiple-Instance Learning (MIL) paradigm (Dietterich et al. 1997) is related to the relationship between bag and instance labels: a bag is positive if it contains at least one positive instance, and negative if it contains only negative instances (Figure 2.3). A classical toy example consists in viewing a bag as set of keys: a bag is labeled positive if it contains a key able to open the door, and negative if none of the keys can. Notice that the information provided by the label is asymmetric in the sense that a negative bag label induces a unique label for every pattern in a bag, while a positive label does not. MIL methods can be categorized into two main paradigms: bag-space methods and instance-space methods.



Supervised learning Multiple-Instance Learning (MIL)

Figure 2.3.: Supervised learning vs MIL: in supervised learning all the examples are labeled whereas in MIL only the bags are labeled, i.e. the instance labels are unknown. The blue dotted line shows the separator learned by the classifier.

Bag-space methods Bag-space methods treat each bag as a whole entity and train a classifier directly on the bags by making a global representation of bags or extracting discriminative bag-level information from them. *Embedded-space methods* correspond to methods that embed each bag into a feature space, where standard supervised learning techniques (e.g. SVM) can be applied. First, each bag is mapped to a single feature vector by a mapping function, then a single-instance classifier is trained in the embedded space. There are several methods to embed a bag: Simple MI (Dong 2006) maps each bag to the average of its instances, Multi-Instance Kernels (Gärtner et al. 2002) define kernels on the bags, and MIGraph and miGraph (Z.-H. Zhou et al. 2009) map a bag to an undirected graph and design a graph kernel. *Distance-based methods* use distance metrics to classify bags. For example, Citation k-NN (J. Wang et al. 2000) defines a bag-to-bag distance, and uses a k-NN approach to predict bag labels. M-C2B (H. Wang et al. 2012) learns a robust and discriminative class-to-bag (C2B) distance for MIL, where each class is a "super-bag" that includes all the instances in the bags of the same class.

Instance-space methods An instance-level classifier is trained to classify positive and negative instances in the instance space, and based on these classifiers a bag-level classifier is derived by aggregation. MIL approaches based on SVMs learn a hyperplane to separate positive and negative instances and perform bag label prediction through its max scoring instance

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \text{sign}\left[\max_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h)\rangle\right] \tag{2.6}$$

where $\boldsymbol{x}$ is the input bag, and $\boldsymbol{w} \in \mathbb{R}^d$ is a vector of model parameters (or hyperplane). If the maximum score is positive, then there is at least one positive instance. If the maximum score is negative, then all the instances are negatives. To learn the parameters $\boldsymbol{w}$, (Andrews et al. 2003) adapted SVMs to the MIL problem, and proposed mi-SVM and MI-SVM algorithms. Both these algorithms are max-margin algorithms, which are formulated as mixed integer optimization problems. The main difference between these two algorithms is how the margin is defined. The mi-SVM is based on an instance margin formulation, i.e. it maximizes the margin over all instances labeled with latent labels, while the MI-SVM is based on a bag margin and maximizes the margin over the most positive instance of each positive bag and all instances of the negative bags.

The MI-SVM inspired pioneer works for weakly labeled object detection in the computer vision community. Specifically, the Latent SVM (LSVM) (Felzenszwalb et al. 2010) solves a "MI-SVM-like" problem, where the instances correspond to sub-part positions of the putative object position. It is worth mentioning that LSVM slightly differs from MI-SVM in the optimization scheme, since only the maximum output latent variable are used for negative examples to solve LSVM optimization problem, whereas all negative instances are used for MI-SVM. Figure 2.4 shows the instances used during training, and the positions of the hyperplane for the standard MIL approaches. We note that the mi-SVM uses all the instances for each bag whereas the LSVM uses a single instance per bag and the MI-SVM uses a single instance for each positive bag, and all the instances for negative bags.

Interesting adaptations of these SVM-like MIL algorithms have been proposed: a solution dedicated to sparse positive bags (Bunescu et al. 2007), using deterministic annealing to continuously approximate the problem (Gehler et al. 2007), a convex relaxation with the soft-max loss function (Joulin et al. 2012), modeling instance dependencies as in MI-CRF (Deselaers et al. 2010), or using a multi-fold MIL procedure (Cinbis et al. 2016) to avoid poor local optima.

2.2.1.2 Multiple-Instance Learning Extensions

Recently, interesting MIL extensions have been introduced in (F. X. Yu et al. 2013; Lai et al. 2014; W. Li et al. 2015; Parizi, Andrea Vedaldi, et al. 2015). All these methods use a bag prediction strategy, which departs from the standard `max` scoring function in MIL, especially due to the relaxation of the common Negative Instances in Negative Bags (NINB) MIL assumption.

In (W. Li et al. 2015), the authors question the NINB assumption by claiming that it is often violated in practice during image annotation: human rather label images based on their dominant concept than on the actual presence of the concept in each sub-region. The standard formulation of MIL fails to account that negative bags can also have very noisy instance composition. (W. Li et al. 2015) introduces a new formulation, where both positive and negative bags are soft, in the sense that negative bags can also contain few positive instances. To support the dominant concept annotation, the authors introduce

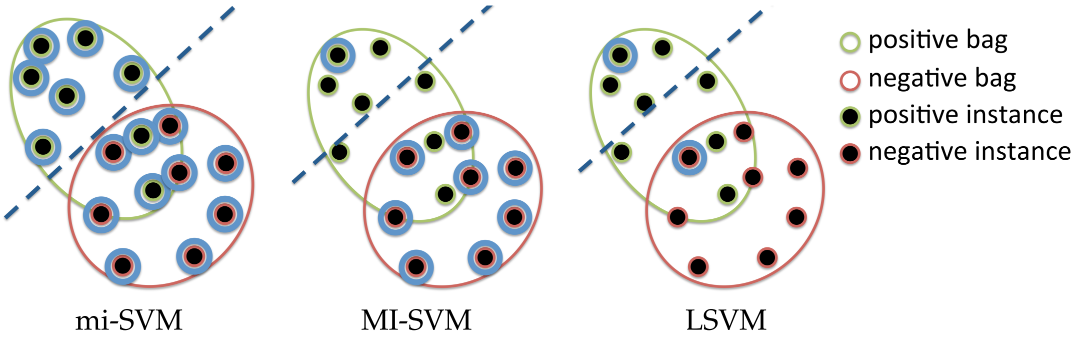


Figure 2.4.: Comparison of standard MIL models. The blue dotted lines show the hyperplanes learned by the models, and the blue circles show the instances used during training to find them.

the top instances prediction function, which selects the top scoring instances in each bag, generalizing the MIL prediction function. The model is trained with a LSVM approach. Similarly (Hajimirsadeghi et al. 2016) introduces the Ratio-constrained Multiple Instance Markov Network (RMIMN) which enforces each bag of class label y to contain at least a certain portion of instances from class y. The authors also propose a generalization of RMIMN to automatically learn the proportion.

In the Learning with Label Proportion (LLP) framework (F. X. Yu et al. 2013), only label ratios between positive and negative instances in the bags are provided during training. This problem often occurs in practice. For example, after election, the proportions of votes of each demographic area are released by the government. In (Lai et al. 2014), the LLP method of (F. X. Yu et al. 2013) is explicitly applied to MIL problems, in the context of video event detection. Similarly to top instances approach (W. Li et al. 2015), (Lai et al. 2014) assumes that each positive video contains "many" positive instances (i.e. frames), while each negative video contains few or no positive instances. They show that LLP outperforms baseline methods (mi/MI-SVM (Andrews et al. 2003)), especially by its capacity to relax the NINB assumption.

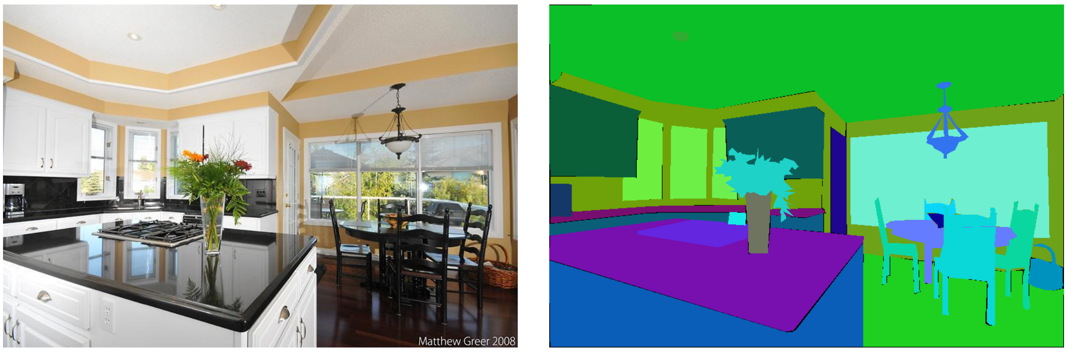
Other approaches depart from the NINB assumption by tracking the negative evidence of a class with regions (Azizpour, Arefiyan, et al. 2015; Parizi, Andrea Vedaldi, et al. 2015). The main idea is to learn mutual exclusion constraints, e.g. model scene subcategories where the positive object class is unlikely to be found, or to capture specific parts, which potentially indicate the presence of an object of a similar but distinct class. (Azizpour, Arefiyan, et al. 2015) proposes a generalization of LSVM by including negative latent variables. In (Parizi, Andrea Vedaldi, et al. 2015), the authors introduce a WSL formulation specific to multi-class classification, where negative evidence is explicitly encoded by augmenting the model parameters to represent the positive/negative contribution of a part to a class.

2.2.2 Weakly Supervised Learning with Structured Output

In this section, we present WSL models for structured output that extend the WSL models of Subsection 2.2.1 to more general outputs, e.g. multi-class labels, ranking matrices, segmentation masks etc.

2.2.2.1 Structured Prediction

In structured output prediction, the goal is to predict a set of *interdependent* output variables $\boldsymbol{y} \in \mathcal{Y}$ for a given input variable $\boldsymbol{x} \in \mathcal{X}$. To simplify the notations, we assume that all inputs have the same output domain $\mathcal{Y}$. In the case where the output domain is different for different examples, it depends on the input: $\mathcal{Y}(\boldsymbol{x})$. The output $\boldsymbol{y} = [y_1, y_2, \dots, y_M]$ is composed of M individual outputs, and $\mathcal{Y}$ is a discrete and finite output domain. The elements of $\mathcal{Y}$ are structured discrete objects such as sequences, trees, graphs, ranking matrices, bounding boxes, segmentation masks, etc. Figure 2.5 shows an example of semantic segmentation, where the goal is to predict the semantic label of each pixel. We aim at predicting for each pixel, whether that the label of a pixel is dependent on neighbor pixels. If we know that all of the neighbors of a particular pixel are labeled *table*, then the pixel itself is most likely also labeled *table*. The core problem in structured output prediction arises from the combinatorial explosion. For example on ADE20K dataset (Bolei Zhou, Zhao, et al. 2016), there are 150 classes and the average image size is about 600×800, therefore the number of possible segmentation masks is about 10^{10^6}.



Original image

Semantic segmentation mask

Figure 2.5.: Example of semantic segmentation mask (right) and the original image (left). The color of a pixel represent its category. The data is taken from the ADE20K dataset (http://groups.csail.mit.edu/vision/datasets/ADE20K/).

The goal of structured output prediction is to learn a function $f_{\boldsymbol{w}} : \mathcal{X} \to \mathcal{Y}$ that maps an input $\boldsymbol{x} \in \mathcal{X}$ to an output $\boldsymbol{y} \in \mathcal{Y}$. The common approach involves computing a *scoring* (or *compatibility*, or *discriminant*) *function* $s_{\boldsymbol{w}} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ for each output, and then selecting the output with the maximum score, as follows:

$$\hat{\boldsymbol{y}}(\boldsymbol{x}) = f_{\boldsymbol{w}}(\boldsymbol{x}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) \qquad (2.7)$$

VARIABLE	NOTATION	SPACE	TRAIN	TEST
Input	$\boldsymbol{x}$	$\mathcal{X}$	observed	observed
Output	$\boldsymbol{y}$	$\mathcal{Y}$	observed	unobserved
Latent	$\boldsymbol{h}$	$\mathcal{H}$	unobserved	unobserved

Table 2.1.: Details of the notations used in this thesis. For image classification, $\boldsymbol{x}$ is the image, $\boldsymbol{y}$ is the label and $\boldsymbol{h}$ is a region of the image.

where $\hat{\boldsymbol{y}}(\boldsymbol{x})$ is the predicted label. To simplify the notation, we note the predicted label $\hat{\boldsymbol{y}}$ (without the dependency of the input $\boldsymbol{x}$) in the rest of this manuscript. The scoring function $s_{\boldsymbol{w}}$ measures the compatibility between the input $\boldsymbol{x}$ and any output $\boldsymbol{y}$.

In many structured output applications (e.g. semantic segmentation), it is expensive to obtain a fully supervised training dataset. To address this problem, the WSL strategy proposes to label inputs with weak labels (e.g. image-level labels), and to model the missing annotations with latent variables. In the WSL setting, an input-output $(\boldsymbol{x}, \boldsymbol{y})$ pair depends on a set of unobserved latent variables $\boldsymbol{h} \in \mathcal{H}$. For example, the image is labeled with image-level labels indicating the presence/absence of a class, and the latent variable $\boldsymbol{h}$ models the segmentation mask. The notations are summarized in Table 2.1. We now present the Latent Structured SVM (LSSVM), and its connection with standard latent structured output models.

2.2.2.2 Latent Structured SVM (LSSVM)

The Latent Structured SVM (LSSVM) (C.-N. Yu et al. 2009) extends the Structured SVM (SSVM) framework (Taskar et al. 2003; Tsochantaridis et al. 2005) to include latent variables.

Model The scoring function depends on a latent variable $\boldsymbol{h} \in \mathcal{H}$, which is used to capture unobserved structures present in the input-output $(\boldsymbol{x}, \boldsymbol{y})$ pair. The latent variables are observed neither at training nor at evaluation time. We define a joint feature map $\Psi : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}^d$, which describes the relationship between the input, the output and the latent variable. Given an input $\boldsymbol{x}$ and output $\boldsymbol{y}$, the LSSVM scoring function searches the "best" latent variable value to measure the compatibility between the input $\boldsymbol{x}$ and the output $\boldsymbol{y}$. Formally, the LSSVM scoring function is given by

$$s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = \max_{\boldsymbol{h} \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) \rangle \tag{2.8}$$

where $\boldsymbol{w} \in \mathbb{R}^d$ is a parameter vector. The form of the feature map allows to learn models for problems as diverse as multi-class image classification (Bilen et al. 2013), natural language parsing (C.-N. Yu et al. 2009), motif finding in yeast DNA (C.-N. Yu et al. 2009), 3D scene understanding (Schwing et al. 2012) and semantic image segmentation (J. Xu et al. 2014).

The maximization operation of Equation 2.7 is known as the *prediction* (or *inference*) problem and requires to maximize over both output and latent variable. If the sets $\mathcal{Y}$ and $\mathcal{H}$ have low cardinalities, we can use an exhaustive search. When the cardinalities of $\mathcal{Y}$ and $\mathcal{H}$ are large, exhaustive search is impractical. We need to use "smart" inference procedures: graph cuts (Szummer et al. 2008), belief propagation (Schwing et al. 2011), etc.

Learning In LSSVM (C.-N. Yu et al. 2009), the optimization is defined as

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_{i=1}^{N} \Delta(\boldsymbol{y}_i^\star, f_{\boldsymbol{w}}(\boldsymbol{x}_i)) \tag{2.9}$$

where C is the trade-off parameter and $\Delta : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ is the loss function that penalizes mismatches between the ground truth output label $\boldsymbol{y}^\star$ and a predicted label $\boldsymbol{y}$. The form of the loss function depends on the nature of the problem. We assume the following properties of a loss function:

1. Non-negativity: $\Delta(\boldsymbol{y}^\star, \boldsymbol{y}) \geq 0 \qquad \forall \boldsymbol{y} \in \mathcal{Y}$
2. Zero for the ground-truth: $\Delta(\boldsymbol{y}^\star, \boldsymbol{y}^\star) = 0$
3. Upper bounded for every given target value, i.e. $\max_{\boldsymbol{y}\in\mathcal{Y}} \Delta(\boldsymbol{y}^\star, \boldsymbol{y})$ exists.

The loss function judges whether the prediction made for a training input is good, or similar enough to the ground truth output. However, the loss function Δ is typically not convex nor continuous and piecewise constant w.r.t $\boldsymbol{w}$, so optimizing it can be computationally expensive. To overcome this problem, the LSSVM replaces the loss function Δ with a surrogate loss $\mathcal{L}$. Formally, learning a LSSVM involves solving the following optimization problem

$$\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_{i=1}^{N} \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) \tag{2.10}$$

$$\text{with} \quad \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) = \max_{(\boldsymbol{y},\boldsymbol{h})\in\mathcal{Y}\times\mathcal{H}} [\Delta(\boldsymbol{y}_i^\star, \boldsymbol{y}) + \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h})\rangle] - \max_{\boldsymbol{h}^\star\in\mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}_i^\star, \boldsymbol{h}^\star)\rangle \tag{2.11}$$

Intuitively, the above problem introduces a margin between the score of the ground-truth output together with the best value of the latent variable and any other pair of output and latent variables

$$\max_{\boldsymbol{h}\in\mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, \boldsymbol{y}^\star, \boldsymbol{h})\rangle \geq \Delta(\boldsymbol{y}^\star, \boldsymbol{y}) + \max_{\boldsymbol{h}\in\mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})\rangle \qquad \forall \boldsymbol{y} \in \mathcal{Y} \tag{2.12}$$

The model is trained such that it maximizes the margin between the ground-truth and any other output. The desired margin is proportional to the loss between the ground-truth and the corresponding output.

Optimization The LSSVM optimization problem is not a convex optimization problem, because of the max operation in a term with negative sign within Equation 2.11. However, (C.-N. Yu et al. 2009) shows that the objective function (Equation 2.10) can be decomposed into a concave and convex function and efficiently be solved by the Concave-Convex Procedure (CCCP) (Yuille et al. 2003). The CCCP is a simple iterative procedure that guarantees to converge to a local minimum or stationary point of the objective. The CCCP

Algorithm 2.1 CCCP algorithm for learning LSSVM

```
Input: Initial model $\boldsymbol{w}$, training dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\}_{i=1}^N$
repeat
    Update $\boldsymbol{h}_i^\star = \arg\max_{\boldsymbol{h} \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}_i^\star, \boldsymbol{h}) \rangle \quad \forall i \in \{1, \dots, N\}$
    Update $\boldsymbol{w}$ by fixing the latent variables of the concave part
        $\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_{i=1}^{N} \max_{(\boldsymbol{y},\boldsymbol{h}) \in \mathcal{Y}\times\mathcal{H}} [\Delta(\boldsymbol{y}_i^\star, \boldsymbol{y}) + \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}) \rangle] - \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}_i^\star, \boldsymbol{h}_i^\star) \rangle \quad (2.14)$
until convergence
return $\boldsymbol{w}$
```

algorithm (Algorithm 2.1) has two main steps. First, it approximates the concave part by upper bounding linear functions. Second, it updates the model parameters by solving the modified optimization problem (Equation 2.14), which is now convex, because the concave part has been linearized. Note that this optimization problem is equivalent to the SSVM optimization problem with $\boldsymbol{y}^\star = (\boldsymbol{y}^\star, \boldsymbol{h}^\star)$. The SSVM optimization problem can be solved with gradient based approaches (Shalev-Shwartz et al. 2011; Lacoste-Julien et al. 2013) or cutting plane methods (Tsochantaridis et al. 2005; Joachims et al. 2009; Shah et al. 2015). These solvers require to solve the Loss-Augmented Inference (LAI) problem, which is a maximization of the sum of the score of the output and the loss function

$$\arg\max_{(\boldsymbol{y},\boldsymbol{h}) \in \mathcal{Y}\times\mathcal{H}} \; \Delta(\boldsymbol{y}^\star, \boldsymbol{y}) + \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) \rangle \tag{2.13}$$

Intuitively, the LAI finds the most violated constraint. Solving efficiently the LAI problem is crucial to learn a SSVM, because it is the most time-consuming step.

Note that we can still use CCCP with an arbitrary function f, by using the generic Difference of Convex functions (DC) decomposition proposed in (Yuille et al. 2003) (Theorem 1). Given a convex function g, there exists a positive constant λ such that $f_{vex}(\boldsymbol{w}) = f(\boldsymbol{w}) + \lambda g(\boldsymbol{w})$ is convex, and $f_{cave}(\boldsymbol{w}) = -\lambda g(\boldsymbol{w})$ is concave. So we can write an arbitrary function f as a difference of convex functions: $f(\boldsymbol{w}) = f_{vex}(\boldsymbol{w}) + f_{cave}(\boldsymbol{w}) = f(\boldsymbol{w}) + \lambda g(\boldsymbol{w}) - \lambda g(\boldsymbol{w})$. This property is used in Subsection 4.4.1.1.

2.2.2.3 Connection with Latent Structured Output Models

As presented in the introduction, the pooling over latent variables is a key issue for WSL models. We have previously presented the LSSVM model, which uses a `max` operator (or pooling) over both output and latent variables during training. A drawback of max pooling is that is not robust to the inherent uncertainty on the variables, because it uses only the information of the maximum values. We now present some models with alternative pooling functions for both output and latent variables during training.

A similar model to LSSVM is the Marginal Structured SVM (MSSVM) (Ping et al. 2014), which proposes to take into account the uncertainty on the latent variables by marginalizing them. But marginalizing over the latent variables usually requires more computation

	$\epsilon_h \to 0^+$ ($\max_{\boldsymbol{h}}$)	$\epsilon_h = 1$ ($\log \sum_{\boldsymbol{h}} \exp$)
$\epsilon_y \to 0^+$ ($\max_{\boldsymbol{y}}$)	LSSVM	MSSVM
$\epsilon_y = \epsilon_h \in (0,1)$	ϵ-framework	
$\epsilon_y = 1$ ($\log \sum_{\boldsymbol{y}} \exp$)	MLLR	HCRF

Table 2.2.: Comparison of pooling strategies for output and latent variables with the unified framework (Equation 2.16).

than maximizing. This MSSVM objective is similar to LSSVM objective (Equation 2.10), except replacing the `max` operator of $\boldsymbol{h}$ ($\max_{\boldsymbol{h}} g(\boldsymbol{y}, \boldsymbol{h})$ where g is a function of both $\boldsymbol{y}$ and $\boldsymbol{h}$) with the Log-Sum-Exp (LSE) function ($\log \sum_{\boldsymbol{h}} \exp g(\boldsymbol{y}, \boldsymbol{h})$). On the contrary, Multinomial Latent Logistic Regression (MLLR) (Z. Xu et al. 2014) marginalizes over the output variables and maximizes over the latent variables. Another popular model for structured output prediction with latent variables is the Hidden Conditional Random Field (HCRF) (Quattoni, S. B. Wang, et al. 2007), which naturally extends the Conditional Random Field (CRF) (Lafferty et al. 2001) to include hidden variables. Unlike MSSVM which marginalizes only over the latent variables, the HCRF marginalizes over both output and latent variables. The `max` operator of $\boldsymbol{y}$ in the MSSVM objective function is replaced by the LSE function. To control the uncertainty in the HCRF model, the ϵ-framework (Pletscher et al. 2010; Schwing et al. 2012) introduces a temperature parameter $\epsilon > 0$, by replacing $\log \sum_{\boldsymbol{y},\boldsymbol{h}} \exp g(\boldsymbol{y}, \boldsymbol{h})$ with $\epsilon \log \sum_{\boldsymbol{y},\boldsymbol{h}} \exp \frac{g(\boldsymbol{y},\boldsymbol{h})}{\epsilon}$. The temperature parameter enables to smooth between `max` ($\epsilon \to 0$) and `average` ($\epsilon \to +\infty$) pooling. The conditional probability of output $\boldsymbol{y}$ and latent variable $\boldsymbol{h}$ given an input $\boldsymbol{x}$ can be defined as follows:

$$P(\boldsymbol{y}, \boldsymbol{h} | \boldsymbol{x}) = \frac{\exp\left(\frac{\langle \boldsymbol{w}, \Psi(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})\rangle}{\epsilon}\right)}{\sum_{\boldsymbol{y}' \in \mathcal{Y}, \boldsymbol{h}' \in \mathcal{H}} \exp\left(\frac{\langle \boldsymbol{w}, \Psi(\boldsymbol{x},\boldsymbol{y}',\boldsymbol{h}')\rangle}{\epsilon}\right)} \tag{2.15}$$

To compare the different methods, (Ping et al. 2014) introduces a unified framework. The general objective function is

$$\begin{aligned} \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N} \sum_{i=1}^{N} \epsilon_y \log \sum_{\boldsymbol{y} \in \mathcal{Y}} \exp \left(\frac{\Delta(\boldsymbol{y}_i^*, \boldsymbol{y}) + \epsilon_h \log \sum_{\boldsymbol{h} \in \mathcal{H}} \exp \frac{\langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h})\rangle}{\epsilon_h}}{\epsilon_y} \right) \\ - \frac{C}{N} \sum_{i=1}^{N} \epsilon_h \log \sum_{\boldsymbol{h} \in \mathcal{H}} \exp \left(\frac{\langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}_i^*, \boldsymbol{h})\rangle}{\epsilon_h} \right) \end{aligned} \tag{2.16}$$

where ϵ_y and ϵ_h are temperature parameters that control how much uncertainty we want to account for in $\boldsymbol{y}$ and $\boldsymbol{h}$, respectively. The Table 2.2 summarizes the different models w.r.t ϵ_y and ϵ_h. We can note that both LSSVM and MLLR maximize over the latent variables, whereas both MSSVM and HCRF marginalize over the latent variables. ϵ-framework is a more general model, where HCRF and LSSVM are special cases.

Given a posterior distribution $P(\boldsymbol{y}, \boldsymbol{h}|\boldsymbol{x})$ (Equation 2.15) and a loss function Δ, an open question is what is the optimal way to predict output and latent variables for a given input variable x? From a theoretical point of view, if the true posterior distribution is known, marginalizing over variables is the optimal predictor according to Bayesian decision theory (Pletscher et al. 2010), but maximizing over variables is also optimal for classification losses (e.g. zero-one loss). However, the optimality may not guaranteed because the true posterior is unknown in practice – it is estimated from labeled training data. Note that this problem of feature selection is a common and open question in machine learning. For example, if we consider supervised learning in the presence of many irrelevant features, a ℓ_1 regularization is often used to perform feature selection.

2.3 Weakly Supervised Learning of Deep ConvNets

In this section, we present deep ConvNet architectures that use WSL methods to learn spatially invariant representations with image-level labels only. A well-known model is the Spatial Transformer Network (STN) (Jaderberg et al. 2015), that learns spatial transformation from data in a deep learning framework. It warps the feature map via a global parametric transformation such as affine transformation. Unfortunately, STN fails to learn the parameters of the transformation on complex images with cluttered background. Recently, several deep ConvNet architectures have been proposed to learn localized features for object localization or semantic segmentation with image-level labels only. We assume that we have a feature extraction network that generates feature maps (Figure 2.6). As we only have image-level labels during training, we use a classification loss to learn the model parameters. The challenging problem is how to transform the features maps into one score per class to use a classification loss (Figure 2.6).

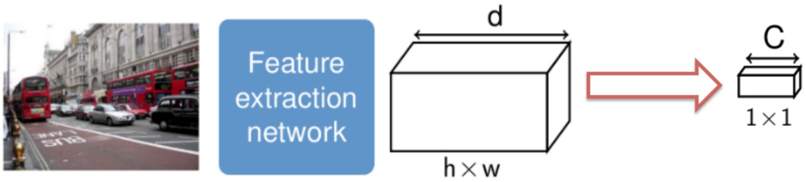


Figure 2.6.: Deep ConvNet architecture for WSL with image classification labels. The d feature maps (spatial resolution $h \times w$) of the feature extraction network are aggregated to generate one score per class, where C is the number of classes.

Global spatial pooling The key issue is to find how to pool the regions to have a score per map. The most popular approach is the `max` pooling (Oquab et al. 2015; Papandreou, Kokkinos, et al. 2015; Pathak, Shelhamer, et al. 2015), which selects the best region to perform prediction. In the case of binary classification, this pooling is an instantiation of the MIL paradigm (Subsection 2.2.1). As mentioned earlier, a limitation of the `max` pooling is related to its sensitivity to noise in the region scores, because it only uses the most discriminative region (Pedro O. Pinheiro and Collobert 2015). (Pedro O. Pinheiro and Collobert 2015) also observes that the training is slow, because only one region (selected by the max) is updated. To increase robustness, several approaches propose to use several regions. The authors of (Bolei Zhou, Khosla, et al. 2016) use the Global Average Pooling (GAP), and show that this pooling can find all the discriminative regions of a category. (Pedro O. Pinheiro and Collobert 2015; C. Sun et al. 2016; Kolesnikov et al. 2016) observe that this pooling have problems identifying the extent of the object: the models trained with max pooling tend to underestimate object sizes, while the models trained with GAP overestimate them. (Pedro O. Pinheiro and Collobert 2015; C. Sun et al. 2016) propose a trade-off between max and average pooling by using a Log-Sum-Exp (LSE) pooling, which is a type of soft-max. Similarly, (Kolesnikov et al. 2016) introduces the Global Weighted Rank-Pooling (GWRP) which has a soft-max behavior. Recently, several approaches (Teh et al. 2016; H. Xu et al. 2016) learn attention model to keep discriminative feature maps only.

Cascade architecture To refine the predicted regions, some methods based on cascade architecture (C. Sun et al. 2016; Diba, V. Sharma, et al. 2016) have been proposed. These architectures have two stages (Figure 2.7). The first stage (localization network) proposes a set of promising boxes that are likely to contain objects. Then, the second stage (classification network) classifies the proposed regions. ProNet (C. Sun et al. 2016) uses a cascade architecture to zoom into those promising boxes, and train new classifiers to verify them. In ProNet, the localization and classification networks are independents and are trained iteratively. This two-stage architecture can be repeated several times to progressively zoom into objects. To improve ProNet, (Diba, V. Sharma, et al. 2016) proposes an end-to-end architecture, where the networks of the two stages share the same convolution layers.

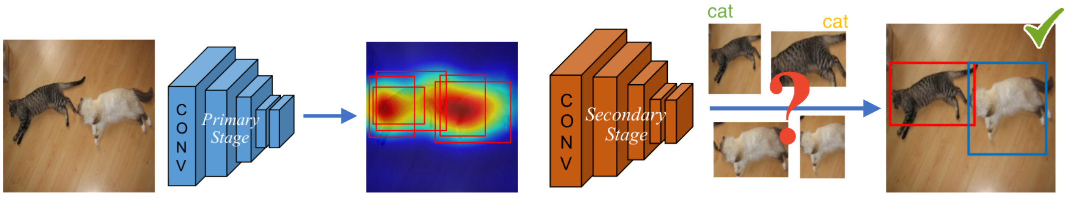


Figure 2.7.: Cascade architecture for WSL localization. (Credit (Diba, V. Sharma, et al. 2016)).

2.4 Image Classification Datasets

In this section, we present standard image classification datasets used in this thesis. We evaluate our models in very different recognition contexts:

- *Object recognition*. The PASCAL VOC 2007 (Everingham, Van Gool, et al. 2007) and PASCAL VOC 2012 (Everingham, Van Gool, et al. 2012) are standard benchmarks for object recognition, and contain twenty classes: *aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, sofa, train* and *TV/monitor*. Each image is a realistic scene (i.e. non-iconic image) from Flickr and can contained several objects. The Microsoft Common Objects in Context (MS COCO) (T.-Y. Lin, Maire, et al. 2014) is similar to PASCAL VOC but contains more classes (80) and considerably more object instances per image. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset (Russakovsky, Deng, et al. 2015) is a popular large scale dataset which contains 1.2M training images and 1,000 categories. Unlike PASCAL VOC and MS COCO, there is a single label per image.

- *Scene categorization*. The 15 Scene dataset (Lazebnik et al. 2006) is a single-label image classification dataset, which contains fifteen natural scene categories: *office, kitchen, living room, bedroom, store, industrial, tall building, inside cite, street, highway, coast, open country, mountain, forest*, and *suburb*. The MIT67 (Quattoni and Torralba 2009) is a standard dataset for indoor scene classification, with 67 categories. As in 15 Scene dataset, each image has a single label.

- *Action recognition*. The UIUC-Sports (L.-J. Li and F.-F. Li 2007) is a single-label image classification dataset which contains eight sport event categories: *rowing, badminton, polo, bocce, snowboarding, croquet, sailing*, and *rock climbing*. The PASCAL VOC 2012 Action (Everingham, Van Gool, et al. 2012) contains ten categories: *jumping, phoning, playing a musical instrument, reading, riding a bicycle or motorcycle, riding a horse, running, taking a photograph, using a computer*, and *walking*. The VOC 2012 Action classification task is harder than UIUC-Sports, because the actions are not mutually exclusive, e.g. a person may simultaneously be walking and phoning.

- *Fine-grained recognition*. The People Playing Musical Instrument (PPMI) dataset (Yao et al. 2010) contains images of humans interacting with twelve different musical instruments: *bassoon, cello, clarinet, erhu, flute, French horn, guitar, harp, recorder, saxophone, trumpet*, and *violin*. For each instrument, there are images that contain a person playing the instrument, as well as images that contain a person holding the instrument without playing. Caltech-UCSD Birds 200 (CUB-200) dataset (Wah et al. 2011) is an image dataset with photos of 200 bird species, and each image contains a single species.

The multi-class datasets (i.e. one single class per image) are evaluated with multi-class accuracy, and the multi-label datasets (i.e. several classes per image) are evaluated with Mean Average Precision (MAP). For all datasets except MS COCO and PASCAL VOC 2012 Action, the performances are evaluated following the standard protocol. On MS COCO dataset, we follow the protocol in (Oquab et al. 2015). On PASCAL VOC 2012 Action, we use the same classification protocol than PASCAL VOC 2007 object classification, with evaluation on the *val* set. Table 2.3 summarizes dataset information.

2.5 Summary & Discussion

In Section 2.1, we have presented two approaches to compute visual representations for image classification: handcrafted (BoW) and learned (ConvNet) representations. Since few years, the ConvNets have become the state-of-the-art models for visual representation. A key success of the ConvNets is that the learned representations on ImageNet are both discriminative and generic, so they can be efficiently transfered to other datasets (Subsection 2.1.2.4). They have also been successfully applied on other standard computer vision applications such as object detection, semantic segmentation, etc. Despite their

Dataset	#Train	#Test	#Classes	Evaluation	Topic
VOC 2007	5,011	4,952	20	MAP	object
VOC 2012	11,540	10,991	20	MAP	object
VOC 2012 Action	2,296	2,292	10	MAP	action
MS COCO	82,783	40,504	80	MAP	object
MIT67	5,360	1,340	67	accuracy	scene
15 Scene	1,500	2,985	15	accuracy	scene
PPMI	2,400	2,400	24	accuracy	fine-grained
UIUC-Sports	640	580	8	accuracy	action
CUB-200	5,994	5,794	200	accuracy	fine-grained
ILSVRC 2012	1,281,167	50,000	1000	accuracy	object

Table 2.3.: Dataset information: number of train and test images, number of classes and evaluation measures.

excellent performances, standard ConvNet architectures only carry limited invariance properties: although a small amount of shift invariance is built into the models through pooling layers, strong invariance is generally not dealt with. To overcome this limitation, we propose in this thesis to use Weakly Supervised Learning (WSL) models to explicitly align image regions to be robust to strong variations.

The most popular approach for WSL in computer vision is Multiple-Instance Learning (MIL) (Subsection 2.2.1.1). MIL is a binary classification problem where a class label is assigned only to a bag of instances, indicating the presence/absence of positive instances. The standard MIL assumption is: a bag is positive if it contains at least one positive instance, and negative if it contains only negative instances. But, several methods show that MIL assumptions are not adapted for real data applications, and propose to relax the assumption that all instances in negative bags are negatives (Subsection 2.2.1.2). In Chapter 3, we introduce a new model SyMIL, which departs from standard negative instances in negative bags assumption and models positive and negative bags in a symmetric manner. Another important difference with existing approaches is that SyMIL model seeks discriminative instances in both positive and negative bags, whereas MIL models and their extensions seek discriminative instances in positive bags only.

Unfortunately, MIL is limited to binary labels. To deal with more complex outputs, a solution is to use the WSL models for structured output prediction. In Subsection 2.2.2.2, we present Latent Structured SVM (LSSVM), which generalizes Structured SVM (SSVM) by incorporating latent variables. As MIL models, LSSVM uses a max pooling over latent variables to perform prediction, but this pooling is not robust to the inherent uncertainty on the latent variable. As detailed in Subsection 2.2.2.3, several methods propose to take into account the uncertainty of the latent variables, by marginalizing. But, marginalizing over latent variables is usually slower than maximizing, and does not allow to explicitly select relevant regions. In Chapter 4, we introduce a new pooling strategy based on pairs of latent variables, which takes into account the uncertainty. While SyMIL seeks discriminative instances for positive and negative classes, the new prediction selects a pair of latent variables which provides positive and negative evidence for a given output. Contrary to standard pooling functions, this pooling function explicitly models negative evidence, e.g. a cow detector should strongly penalize the prediction of the bedroom class. As shown in Subsection 2.2.1.2 for MIL models, tracking the negative evidence of a class is important to have accurate prediction. Based on this pooling, we propose a new structured output latent variable model, called Minimum mAximum lateNt sTRucturAl SVM (MANTRA), which extends SyMIL to structured outputs.

Finally, we present deep ConvNet architectures for WSL with image-level labels. As Section 2.3 shows, the key issue is how to pool the regions to have a score per class. Most of the methods use a `max`, average or soft-max pooling. An analysis of these pooling functions is done in Section 6.5. In Chapter 5, we propose the WELDON pooling, which extends MANTRA to multiple regions to be more robust. We design a fully convolutional network architecture which enables fast region feature computation by convolutional sharing. In Chapter 6, we propose a new pooling function which extends WELDON pooling by incorporating a trade-off parameter to weight the positive and negative evidence terms. To enrich the model, we also introduce a class-wise pooling to learn several modalities per class.

Chapter 3

SYMIL: MINMAX LATENT SVM FOR WEAKLY LABELED DATA

Contents

Chapter abstract

This chapter proposes a new Multiple-Instance Learning (MIL) framework for bag classification. Examples are represented as bags of instances, but we depart from standard MIL assumptions by introducing a symmetric strategy (SyMIL) that models positive and negative bags in a symmetric manner. SyMIL is represented with a latent variable model seeking the most discriminative instances. We derive a large margin formulation of our problem, which is cast as a Difference of Convex functions (DC), and optimized using the Concave-Convex Procedure (CCCP). Finally, we evaluate our model on text, musk and image MIL datasets and we analyze the selected instances for two applications.

The work in this chapter is under review:

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2017b). “SyMIL: MinMax Latent SVM for Weakly Labeled Data”. In: *IEEE Transactions on Neural Networks and Learning Systems (TNNLS) [Submission]*

3.1 Introduction

Designing powerful models able to handle weakly labeled data is a crucial problem in machine learning. This issue has been extensively studied during the last 20 years in several contexts: drug activity recognition, text classification, content-based image retrieval, etc. We presented leading techniques to model this problem as a Multiple-Instance Learning (MIL) problem in Subsection 2.2.1. The main MIL assumption is related to the relationship between bag and instance labels: a bag is positive if it contains at least one positive instance, and negative if it contains only negative instances. The standard MIL prediction used the instance with the maximum score to classify the bag. However, the Negative Instances in Negative Bags (NINB) assumption seems to be too restrictive for some applications (Subsection 2.2.1.2). To address this problem, several methods propose to relax the NINB assumption: a large portion of instances in a positive bag should be positive, whereas few instances in the negative bags may be positive.

In the same spirit, we propose SyMIL model, which models positive and negative bags in a symmetric manner. The main novelty of the SyMIL model is the introduction of an additional latent variable, to seek discriminative instances of the negative class (Subsection 3.2.1). To train the model, we propose a learning formulation (Subsection 3.2.2), and we show that this optimization problem can be cast as a Difference of Convex functions (DC), and optimized using the Concave-Convex Procedure (CCCP) (Section 3.3). In Subsection 3.4.1, we validate the intuition of SyMIL on toy datasets composed of synthetic or real data. Finally we evaluate SyMIL on standard MIL datasets (Subsection 3.4.2), and weakly supervised object detection datasets (Subsection 3.4.3).

3.2 SyMIL Model

We consider the problem of learning with weak supervision in a binary classification context. Training data are composed of N labeled bags $\mathcal{D} = \{(\boldsymbol{x}_1, y_1^\star), \dots, (\boldsymbol{x}_N, y_N^\star)\} \in (\mathcal{X} \times \mathcal{Y})^N$ with binary labels ($\mathcal{Y} = \{-1, +1\}$). Let us denote the set of positive bags as $\mathcal{P} = \{(\boldsymbol{x}_i, y_i^\star), y_i^\star = +1\}$, $N^+ = |\mathcal{P}|$, and the set of negative bags as $\mathcal{N} = \{(\boldsymbol{x}_i, y_i^\star), y_i^\star = -1\}$, $N^- = |\mathcal{N}|$. Each bag $\boldsymbol{x} = \{\boldsymbol{x}^h\}_{h \in \mathcal{H}}$ is itself a set of $|\mathcal{H}|$ instances, and $\Phi(\boldsymbol{x}, h)$ is the vectorial representation of the h-th instance.

3.2.1 Prediction Function

Given an unlabeled bag $\boldsymbol{x}$, we want to design a discriminant function $s_{\boldsymbol{w}} : \mathcal{X} \to \mathbb{R}$, parametrized by $\boldsymbol{w}$, such that $f_{\boldsymbol{w}}(\boldsymbol{x}) = \text{sign}\left[s_{\boldsymbol{w}}(\boldsymbol{x})\right]$ gives predicted label of $\boldsymbol{x}$: $s_{\boldsymbol{w}}(\boldsymbol{x}) > 0$ classifies the example as positive, and negative otherwise. We define two latent variables:

$$h^+ = \arg\max_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h) \rangle \qquad h^- = \arg\min_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h) \rangle \tag{3.1}$$

where h^+ (resp. h^-) is the maximum (resp. minimum) scoring latent value for the linear model $\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h) \rangle$. Using h^+ and h^-, we propose the following scoring function:

$$s_{\boldsymbol{w}}(\boldsymbol{x}) = \begin{cases} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^+) \rangle & \text{if } \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^+) \rangle \geq -\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^-) \rangle \\ \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^-) \rangle & \text{otherwise} \end{cases} \tag{3.2}$$

We note that this prediction function can be written as

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \text{sign}[\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^+) + \Phi(\boldsymbol{x}, h^-) \rangle] \tag{3.3}$$

To predict the bag label, Equation 3.3 sum the score of the instance with the maximum score, and the score of the instance with the minimum score. We now present the intuition of our prediction function.

Model intuition & discussion. The main novelty of the SyMIL model is the introduction of the latent variable h^-: The rationale of the function $s_{\boldsymbol{w}}(\boldsymbol{x})$ in Equation 3.2 is to compare the score of the most '$\oplus$-like' instance (i.e. $\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^+) \rangle$) to the score of the most '$\ominus$-like' instance (i.e. $-\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^-) \rangle$). h^+ (resp. h^-) represents the most discriminative latent value for class $\oplus$ (resp. $\ominus$). During training, we aim at using h^+ (resp. h^-) for positive (resp. negative) bags, and so learning the most discriminative model.

The SyMIL model is connected to the Learning with Label Proportion (LLP) model presented in Subsection 2.2.1.2. From the LLP perspective, SyMIL corresponds to a symmetric prior for the label proportion: for a positive bag $(\boldsymbol{x}, +1)$ (resp. negative bag $(\boldsymbol{x}, -1)$), the proportion of positive (resp. negative) instances is $p_\oplus(\boldsymbol{x}, +1) \geq \frac{1}{|\mathcal{H}|}$ (resp. $p_\ominus(\boldsymbol{x}, -1) \geq \frac{1}{|\mathcal{H}|}$). This departs from state-of-the-art SVM-like MIL algorithms, e.g. mi/MI-SVM (Andrews et al. 2003) and LSVM (Felzenszwalb et al. 2010)), where the prediction function takes the form $s_{\boldsymbol{w}}(\boldsymbol{x}) = \max_h \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h) \rangle$, corresponding to $p_\oplus(\boldsymbol{x}, +1) \geq \frac{1}{|\mathcal{H}|}$ but $p_\ominus(\boldsymbol{x}, -1) = 1$. In this asymmetric modeling, $\oplus$ instances represent patterns that are discriminative for the $\oplus$ class, whereas $\ominus$ instances are implicitly regarded as background (i.e. everything different from $\oplus$ instances in the feature space). In contrast, SyMIL assumes that the negative class presents a structure from which discriminative instances can be extracted. This is particularly relevant when the negative class is composed of examples from other categories, as in a multi-class context.

An illustrative comparison between symmetric and asymmetric MIL modeling is provided in Figure 3.1, for an image classification task. Here, bags represent images, and instances are rectangular image regions, in a simple two-class case (bison vs lama). Basically, asymmetric models tend to learn a function discriminating bison patches from the most difficult patches in negative images, i.e. background patches. In contrast, the symmetric SyMIL model seeks regions that are statistically discriminant for both positive and negative images (hopefully bison and lama patches), i.e. the instance the most distant from the hyperplane. SyMIL model tends to ignore background regions, i.e. those shared between $\oplus$ and $\ominus$ images. This more semantic instance selection, and its translation to an improved predictive accuracy, is extensively studied in the experiments (section Section 3.4).

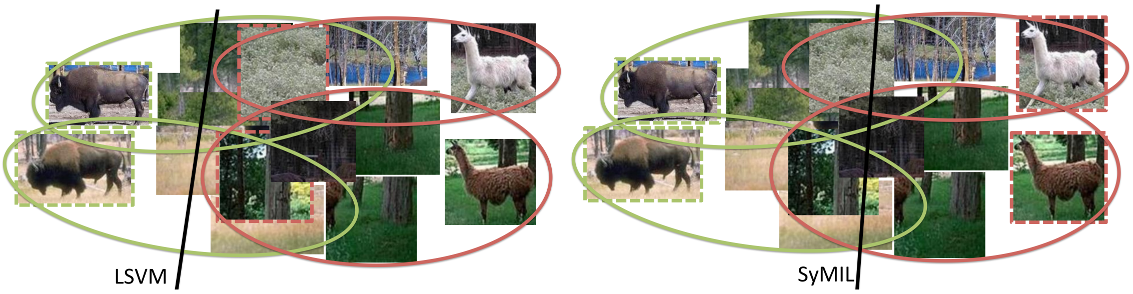


Figure 3.1.: SyMIL model motivation: symmetric (right) *vs* asymmetric (left) modeling between bison $\oplus$ (green) / lama $\ominus$ (red) bags. Asymmetric model seeks discriminative regions for positive bags only, whereas symmetric model seeks discriminative regions for both positive and negative bags. We show the hyperplane in black and the selected instances for each bag in dotted lines.

3.2.2 Learning

For learning, we define three constraints:

1. The first constraint

$$\forall i \in \mathcal{P} \quad \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) \rangle \geq 1 \tag{3.4}$$

enforces that the bag $\boldsymbol{x}_i \in \mathcal{P}$ is properly classified in the class $\oplus$, using the latent value h_i^+, with a safety margin of 1. This is satisfied for the positive bag in Figure 3.2.

2. The second constraint

$$\forall i \in \mathcal{N} \quad \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-) \rangle \leq -1 \tag{3.5}$$

enforces that the bag $\boldsymbol{x}_i \in \mathcal{N}$ is properly classified in the class $\ominus$, using the latent value h_i^-, with a safety margin of 1. This is satisfied for the negative bag in Figure 3.2.

3. The third constraint

$$\forall i \in \mathcal{D} \quad y_i^\star \left[\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) + \Phi(\boldsymbol{x}_i, h_i^-) \rangle \right] \geq 1 \tag{3.6}$$

enforces that each positive (resp. negative) bag is represented by h_i^+ (resp. h_i^-). For example, for $y_i = 1$, it translates into $\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) \rangle \geq -\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-) \rangle + 1$, so that h_i^+ is preferred over h_i^- to represent $\boldsymbol{x}_i$ with a safety margin of 1, and $s_{\boldsymbol{w}}(\boldsymbol{x}_i) = \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) \rangle$. In Figure 3.2, this constraint is satisfied for the positive bag since $\Delta = \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) + \Phi(\boldsymbol{x}_i, h_i^-) \rangle \geq 1$. In a similar fashion, this constraint is satisfied in Figure 3.2 for the negative bag with $\Delta \leq -1$.

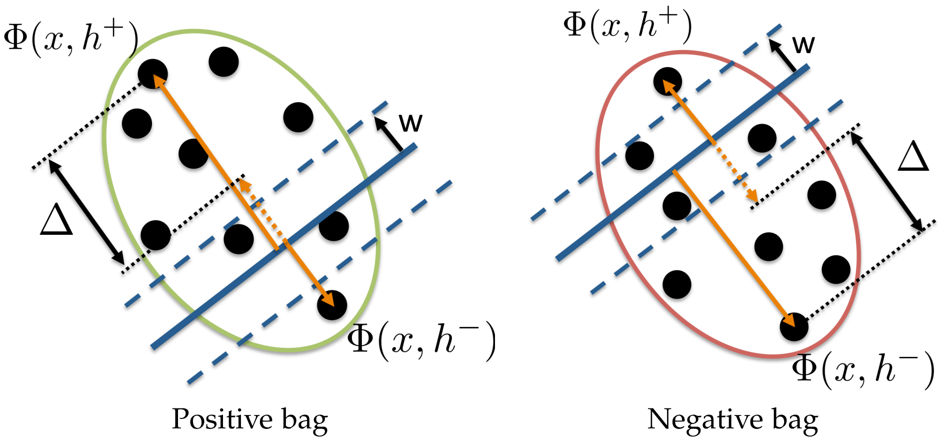


Figure 3.2.: Illustration of the three constraints enforced during training for a positive (green) and a negative (red) bag. Dashed lines represent the safety margin of 1. $\Delta = \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^+) + \Phi(\boldsymbol{x}, h^-) \rangle$.

To optimize $\boldsymbol{w}$ over all the constraints, the following primal regularized loss $\mathcal{P}(\boldsymbol{w})$ is minimized:

$$\mathcal{P}(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\mathcal{L}(\boldsymbol{w}, \mathcal{D}) \tag{3.7}$$

$$\mathcal{L}(\boldsymbol{w}, \mathcal{D}) = \frac{1}{N}\left(\frac{N}{N^+} \sum_{i \in \mathcal{P}} \left[1 - \max_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h) \rangle \right]_+ + \frac{N}{N^-} \sum_{i \in \mathcal{N}} \left[1 + \min_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h) \rangle \right]_+ \right.$$
$$\left. + \lambda \sum_{i \in \mathcal{D}} \left[1 - y_i^\star \left(\max_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h) \rangle + \min_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h) \rangle \right) \right]_+ \right) \tag{3.8}$$

where C is a hyper-parameter that balances the regularization penalty and the empirical loss. $\mathcal{P}(\boldsymbol{w})$ contains the standard max margin regularization term and a data-dependent term $\mathcal{L}(\boldsymbol{w}, \mathcal{D})$ penalizing the violation of the constraints using a hinge loss function $[b]_+ = \max(0, b)$. Note that it is easy to verify that the loss function $\mathcal{L}(\boldsymbol{w}, \mathcal{D})$ in Equation 3.7 defined over the three constraints is a surrogate of the $0/1$ loss on the prediction function $f_{\boldsymbol{w}}$.

Our prediction function $f_{\boldsymbol{w}}(\boldsymbol{x}) = \text{sign}[\langle \boldsymbol{w}, \Phi(\boldsymbol{x}, h^+) + \Phi(\boldsymbol{x}, h^-) \rangle]$ may be seen as an instantiation of the LSSVM prediction function (Equation 2.7). Indeed, with $\mathcal{Y} = \{-1, 1\}$ and $\Psi(\boldsymbol{x}, y, h) = y \cdot \Phi(\boldsymbol{x}, h)$, we have $f_{\boldsymbol{w}}(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} \max_{h \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, y, h) \rangle$. Interestingly, our prediction function $f_{\boldsymbol{w}}$ is actually the natural instantiation of LSSVM to the binary classification case, which is not the case for competitive algorithms, e.g. mi/MI-SVM or LSVM. However, regarding learning formulation, $\mathcal{P}(\boldsymbol{w})$ in Equation 3.7 differs from the LSSVM objective with the previous instantiation of $f_{\boldsymbol{w}}$, which would correspond to only incorporating the third constraint (Equation 3.6). We add the constraints 1 (Equation 3.4) & 2 (Equation 3.5), because they correspond to the ultimate goal of the weakly supervised classifier: properly classifying (beyond the margin) training bags. We analyze the impact of these constraints in Subsection 3.4.2.

Theoretical Analysis We provide a bound of the average Rademacher complexity ($\mathcal{R}_N$) of SyMIL model. We note $\mathcal{F}$ the hypothesis class for instances and $\bar{\mathcal{F}}$ the hypothesis class for bags and we assume that the instances are in the hyper-sphere of radius B. To bound the average Rademacher complexity, we use the Theorem 20 of (Sabato et al. 2012). The bound in the general case is:

$$\mathcal{R}_N(\bar{\mathcal{F}},\mathcal{D}) \leq \frac{4 + 10\log(4ea_1^2a_2^2B^2rN^2)(A + \frac{a_1a_2}{\beta+1}K\ln^{\beta+1}(16a_1^2a_2^2N))}{\sqrt{N}} \tag{3.9}$$

where r is the average bag size, N is the number of training examples, a_1 (resp. a_2) is the Lipschitz constant of bag-labeling (resp. loss) function, and A is a constant. The constant K and β must satisfy an inequality which depends on the worst-case Rademacher complexity over instances: $\mathcal{R}_N^{sup}(\mathcal{F}) \leq \frac{K\ln^{\beta}(N)}{\sqrt{N}}$. SyMIL learns a classification function in the instance space, so that the worst-case Rademacher complexity over instances is the same than SVM, i.e. $\mathcal{R}_N^{sup}(\mathcal{F}) \leq \frac{W}{\sqrt{N}}$[1] (proof in (Bartlett et al. 2003)), corresponding to $\beta = 0$ and $K = W$. Note that this bound is the same for LSVM, since both models use the same classification model over instances.

As mentioned when drawing the connection with LSSVM, SyMIL prediction function in Equation 3.2 is equivalent to $f_{\boldsymbol{w}}(\boldsymbol{x}) = \text{sign}[\langle \boldsymbol{w}, \Phi(\boldsymbol{x},h^+) + \Phi(\boldsymbol{x},h^-)\rangle]$. Therefore, the SyMIL bag-labeling function is 2-Lipschitz with respect to the infinity norm, because max and min functions are 1-Lipschitz with respect to the infinity norm. The loss function in Equation 3.7 is $(1+2\lambda)$-Lipschitz. Therefore, by substituting a_1, a_2, β and K values, we get following bound of the average Rademacher complexity:

$$\mathcal{R}_N(\bar{\mathcal{F}},\mathcal{D}) \leq \frac{4 + 10\log(16eB^2rN^2(1+2\lambda)^2)(A + 2(1+2\lambda)W\ln(64(1+2\lambda)^2N))}{\sqrt{N}} \tag{3.10}$$

The resulting bound indicates that there is a poly-logarithmic dependence of the sample complexity on the average bag size. By comparing SyMIL bound with the LSVM one provided by (Sabato et al. 2012), both bounds are similar and have the same order of magnitude[2]. Despite selecting the maximum or minimum instance, which introduces a non-linearity to the hypothesis class, this bound enables a control of the model complexity. We can note that SyMIL and LSSVM bounds have the same asymptotic behavior in $\frac{\ln(N)}{\sqrt{N}}$.

3.3 Solving the Optimization Problem

Like competitive MIL algorithms (mi-SVM, MI-SVM, LSVM), the objective function $\mathcal{P}(\boldsymbol{w})$ (Equation 3.7) is not a convex function of $\boldsymbol{w}$. In this section, we introduce our own solver to optimize $\mathcal{P}(\boldsymbol{w})$.

[1] In the SVM case, the class of functions is the set of linear separators with a bounded norm $\{x \mapsto \langle \boldsymbol{w}, \Phi(\boldsymbol{x})\rangle : \|\boldsymbol{w}\| \leq W\}$, for some $W > 0$.

[2] The difference with LSVM is that bag and loss functions are 1-Lipschitz.

3.3.1 Difference of Convex Functions

First, we show that primal regularized loss (Equation 3.7) can be written as $\mathcal{P}(\boldsymbol{w}) = u(\boldsymbol{w}) - v(\boldsymbol{w})$, where u and v are convex functions. Rewriting $\mathcal{P}(\boldsymbol{w})$ as a difference of convex functions is not straightforward given the form of Equation 3.7. For this purpose, we use the property $\max(0, a-b) = \max(a,b) - b$ where a, b are convex functions. We also use the properties that the maximum of a linear functions is a convex function, and the minimum of a linear functions is a concave function. Next, we show that each hinge loss can be rewritten as a difference of convex functions. It is not straight-forward because each loss is neither a concave nor a convex function. For example, the first loss is the maximum of a concave function and a constant function, so it is neither a concave nor a convex function. But it can be rewritten as a difference of convex function with $a = 0$ and $b = -(1 - \max\limits_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle)$:

$$\max(0, \underbrace{1 - \max_{h\in\mathcal{H}}\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle)}_{concave} = \underbrace{\max(0, \max_{h\in\mathcal{H}}\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle - 1)}_{convex} - \underbrace{(\max_{h\in\mathcal{H}}\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle - 1)}_{convex} \tag{3.11}$$

Similarly, we can rewrite the global optimization problem $\mathcal{P}(\boldsymbol{w})$ as a difference of convex functions: $\mathcal{P}(\boldsymbol{w}) = u(\boldsymbol{w}) - v(\boldsymbol{w})$ where:

$$\begin{aligned} u(\boldsymbol{w}) =& \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\Bigg(\sum_{i\in\mathcal{P}} \Big[\frac{N}{N^+} \max\Big(0, \max_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle - 1\Big) \\ &+ \lambda \max\Big(1 - \min_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle, \max_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle\Big)\Big] \\ &+ \sum_{i\in\mathcal{N}} \Big[\frac{N}{N^-} \max\Big(0, -\min_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle - 1\Big) \\ &+ \lambda \max\Big(1 + \max_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle, -\min_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle\Big)\Big]\Bigg) \end{aligned} \tag{3.12}$$

$$\begin{aligned} v(\boldsymbol{w}) =& \frac{C}{N}\Bigg(\sum_{i\in\mathcal{P}} \Big[\Big(\frac{N}{N^+} + \lambda\Big) \max_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle - \frac{N}{N^+} \Big] \\ &+ \sum_{i\in\mathcal{N}} \Big[-\Big(\frac{N}{N^-} + \lambda\Big) \min_{h\in\mathcal{H}} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h)\rangle + \frac{N}{N^-} \Big]\Bigg) \end{aligned} \tag{3.13}$$

u and v are convex on $\boldsymbol{w}$ as a sum of convex functions.

3.3.2 Optimization

Once we exhibit the decomposition in difference of convex functions, we solve the resulting difference of convex functions using Concave-Convex Procedure (CCCP) (Yuille et al. 2003). In addition to the CCCP convergence properties (Sriperumbudur et al. 2009), this solution offers the possibility to jointly optimize some latent variables with the classifier parameters for each convex sub-problem, resulting in different (and better) local

Algorithm 3.1 for training SyMIL with CCCP

```
Input: training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,n}$
1: Set $t = 0$, randomly initialize $\{h^+_{i,0}, h^-_{i,0}\}_{i=1,\dots,n}$ and linearize the concave part
2: repeat
3:     Solve convex problem
           $\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w}} \mathcal{P}_t^{CCCP}(\boldsymbol{w}) \quad \text{or} \quad \boldsymbol{\alpha}_{t+1} = \arg\max_{\boldsymbol{\alpha}} \mathcal{D}_t^{CCCP}(\boldsymbol{\alpha}) \qquad (3.14)$
4:     $t \leftarrow t + 1$
5:     Linearize the concave part $-v$ at the current solution $\boldsymbol{w}_t$ / $\boldsymbol{\alpha}_t$
6: until stopping criteria reach
7: return $\boldsymbol{w}_t / \boldsymbol{\alpha}_t$
```

minima. The overall scheme of our CCCP-based optimization is presented in Algorithm 3.1: we alternate between linearizing the concave part $(-v)$ at the current solution (Line 5) and solving the resulting convexified problem (Line 3). We now detail how the problem is solved in the primal and the dual.

3.3.2.1 Primal

The overall algorithm to train SyMIL with CCCP in the primal is given in Algorithm 3.2. CCCP is an iterative algorithm that alternates between linearizing the concave part $(-v)$ at the current solution $\boldsymbol{w}_t$ (Line 5 of Algorithm 3.2) and solving the resulting convex problem (Line 3 of Algorithm 3.2). The linearization of the concave part $-v(\boldsymbol{w})$ consists in upper bounding it by its tangent hyperplane: $-v(\boldsymbol{w}) \leq -\langle \boldsymbol{w}, \nabla_{\boldsymbol{w}} v(\boldsymbol{w}_t)\rangle$, with:

$$\nabla_{\boldsymbol{w}} v(\boldsymbol{w}_t) = \left(\sum_{i \in \mathcal{P}} (\frac{N}{N^+} + \lambda) \Phi(\boldsymbol{x}_i, h^+_{i,t}) - \sum_{i \in \mathcal{N}} (\frac{N}{N^-} + \lambda) \Phi(\boldsymbol{x}_i, h^-_{i,t}) \right) \tag{3.15}$$

where $h^+_{i,t} = \arg\max_{h \in \mathcal{H}} \langle \boldsymbol{w}_t, \Phi(\boldsymbol{x}_i, h)\rangle$ and $h^-_{i,t} = \arg\min_{h \in \mathcal{H}} \langle \boldsymbol{w}_t, \Phi(\boldsymbol{x}_i, h)\rangle$. After linearization, the resulting optimization problem is:

$$\begin{aligned} \mathcal{P}_t^{CCCP}(\boldsymbol{w}) &= u(\boldsymbol{w}) - \langle \boldsymbol{w}, \nabla_{\boldsymbol{w}} v(\boldsymbol{w}_t)\rangle \qquad (3.16) \\ &= u(\boldsymbol{w}) - \frac{C}{N} \left(\sum_{i \in \mathcal{P}} \left(\frac{N}{N^+} + \lambda \right) \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h^+_{i,t})\rangle - \sum_{i \in \mathcal{N}} \left(\frac{N}{N^-} + \lambda \right) \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h^-_{i,t})\rangle \right) \end{aligned}$$

At iteration t, to solve the convexified optimization problem $\min_w \mathcal{P}_t^{CCCP}(\boldsymbol{w})$ in the primal, we use a SGD strategy (Léon Bottou 2010) that proves to be simple and achieves fast convergence. Although we could use more efficient techniques, such as Stochastic Average Gradient (SAG) (Le Roux et al. 2012), we find SGD sufficient in our experiments (Section 3.4). The gradient computation is given in Equation 3.17.

$$\nabla_w \mathcal{P}_t^{CCCP}(\boldsymbol{w}) = \begin{cases} \boldsymbol{w} + \frac{C}{N}(D + E - (\frac{N}{N^+} + \lambda)\Phi(\boldsymbol{x}_i, h^+_{i,t})) & \text{if } y_i^\star = +1 \\ \boldsymbol{w} + \frac{C}{N}(F + G + (\frac{N}{N^-} + \lambda)\Phi(\boldsymbol{x}_i, h^-_{i,t})) & \text{otherwise} \end{cases} \tag{3.17}$$

Algorithm 3.2 for training SyMIL with CCCP (Primal)

```
Input: training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,N}$
Set $t = 0$, randomly initialize $h^+_{i,0}$, $h^-_{i,0}$ and
    $$\boldsymbol{g}_0 = \frac{C}{N}\left(\sum_{i\in\mathcal{P}}\left(\frac{N}{N^+}+\lambda\right)\Phi(\boldsymbol{x}_i, h^+_{i,0}) - \sum_{i\in\mathcal{N}}\left(\frac{N}{N^-}+\lambda\right)\Phi(\boldsymbol{x}_i, h^-_{i,0})\right) \quad (3.22)$$
repeat
    Solve $\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w}} [u(\boldsymbol{w}) - \langle \boldsymbol{w}, \boldsymbol{g}_t\rangle]$
    $t \leftarrow t+1$
    Compute $\boldsymbol{g}_t = \nabla_{\boldsymbol{w}} v(\boldsymbol{w}_t)$
until $[u(\boldsymbol{w}_t) - v(\boldsymbol{w}_t)] - [u(\boldsymbol{w}_{t-1}) - v(\boldsymbol{w}_{t-1})] < \varepsilon$
return $\boldsymbol{w}_t$
```

$$D = \begin{cases} \frac{N}{N^+}\Phi(\boldsymbol{x}_i, h_i^+) & \text{if } \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+)\rangle - 1 > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.18}$$

$$E = \begin{cases} -\lambda\Phi(\boldsymbol{x}_i, h_i^-) & \text{if } 1 - \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-)\rangle > \langle w, \Phi(\boldsymbol{x}_i, h_i^+)\rangle \\ \lambda\Phi(\boldsymbol{x}_i, h_i^+) & \text{otherwise} \end{cases} \tag{3.19}$$

$$F = \begin{cases} -\frac{N}{N^-}\Phi(\boldsymbol{x}_i, h_i^-) & \text{if } -\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-)\rangle - 1 > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.20}$$

$$G = \begin{cases} \lambda\Phi(\boldsymbol{x}_i, h_i^+) & \text{if } 1 + \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+)\rangle > -\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-)\rangle \\ -\lambda\Phi(\boldsymbol{x}_i, h_i^-) & \text{otherwise} \end{cases} \tag{3.21}$$

3.3.2.2 Dual

For many applications, nonlinear models are required to achieve good performances. We propose here a kernelized version of our SyMIL scheme. First, we detail the linearization of the concave part (Line 5 in Algorithm 3.1) in the dual, and then the solving of the convexified problem with cutting-plane.

Linearizing the concave part. To linearize the concave part at iteration $t+1$, we have to fix the latent variables. For a bag $\boldsymbol{x}_j$, with current solution $\boldsymbol{\alpha}^{(t)}$, the inference of the new latent variable value is:

$$h^+_{j,t+1} = \arg\max_{h\in\mathcal{H}} \left\langle \sum_k \alpha_k^{(t)} \frac{1}{N} \sum_{i\in\mathcal{D}} (\boldsymbol{g}_{cave}(\boldsymbol{x}_i, h^+_{i,t}, h^-_{i,t}) - \boldsymbol{g}_{vex}(\boldsymbol{x}_i, h_i^+, h_i^-)), \Phi(\boldsymbol{x}_j, h)\right\rangle \tag{3.23}$$

$$h^-_{j,t+1} = \arg\min_{h\in\mathcal{H}} \left\langle \sum_k \alpha_k^{(t)} \frac{1}{N} \sum_{i\in\mathcal{D}} (\boldsymbol{g}_{cave}(\boldsymbol{x}_i, h^+_{i,t}, h^-_{i,t}) - \boldsymbol{g}_{vex}(\boldsymbol{x}_i, h_i^+, h_i^-)), \Phi(\boldsymbol{x}_j, h)\right\rangle \tag{3.24}$$

where the gradient of the convex and concave terms are:

$$\boldsymbol{g}_{cave}(\boldsymbol{x}_i, h^+_{i,t}, h^-_{i,t}) = \begin{cases} (\frac{n}{n^+}+\lambda)\Phi(\boldsymbol{x}_i, h^+_{i,t}) & \text{if } i \in \mathcal{P} \\ (\frac{n}{n^-}+\lambda)\Phi(\boldsymbol{x}_i, h^-_{i,t}) & \text{if } i \in \mathcal{N} \end{cases} \tag{3.25}$$

Algorithm 3.3 Cutting plane algorithm with 1-slack formulation at iteration t

```
Input: training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1,\dots,n}$, $\{(h^+_{i,t}, h^-_{i,t})\}_{i=1,\dots,n}$ , precision $\varepsilon$
 1: Set $T = 0$, $\boldsymbol{c} \leftarrow 0$, $\boldsymbol{H} \leftarrow 0$
 2: repeat
 3:     $\boldsymbol{H} \leftarrow (H_{ij})_{1 \le i,j \le T}$ where $H_{ij} = \boldsymbol{g}_{(i)}^T \boldsymbol{g}_{(j)}$
 4:     $\boldsymbol{\alpha} \leftarrow \arg\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T \boldsymbol{c} - \boldsymbol{\alpha}^T \boldsymbol{H} \boldsymbol{\alpha} \quad \text{s.t. } 0 \le \mathbf{1}^T \boldsymbol{\alpha} \le C$
 5:     $\xi \leftarrow \frac{1}{C}(\boldsymbol{\alpha}^T \boldsymbol{c} - \boldsymbol{\alpha}^T \boldsymbol{H} \boldsymbol{\alpha})$
 6:     $T \leftarrow T + 1$
 7:     for $i = 1, \dots, N$ do
 8:         $h_i^+ \leftarrow \arg\max_{h \in \mathcal{H}} \langle \sum_{i=1}^{T-1} \alpha_i \boldsymbol{g}_{(i)}, \Phi(\boldsymbol{x}_i, h) \rangle \qquad h_i^- \leftarrow \arg\min_{h \in \mathcal{H}} \langle \sum_{i=1}^{T-1} \alpha_i \boldsymbol{g}_{(i)}, \Phi(\boldsymbol{x}_i, h) \rangle$
 9:     end for
10:     $\boldsymbol{g}_{(T)} \leftarrow \frac{1}{N} \sum_{i \in \mathcal{D}} \boldsymbol{g}_{cave}(\boldsymbol{x}_i, h^+_{i,t}, h^-_{i,t}) - \boldsymbol{g}_{vex}(\boldsymbol{x}_i, h_i^+, h_i^-)$
11:     $c_T \leftarrow \frac{1}{N} \sum_{i \in \mathcal{D}} \Big( vex(\boldsymbol{x}_i, h_i^+, h_i^-) - \langle \boldsymbol{g}_{vex}(\boldsymbol{x}_i, h_i^+, h_i^-), \sum_{i=1}^{T-1} \alpha_i \boldsymbol{g}_{(i)} \rangle \Big)$
12: until $\langle \sum_{i=1}^{T-1} \alpha_i \boldsymbol{g}_{(i)}, \boldsymbol{g}_{(T)} \rangle \ge c_T - \xi - \varepsilon$
13: return $\boldsymbol{\alpha}$
```

$$\boldsymbol{g}_{vex}(\boldsymbol{x}_i, h_i^+, h_i^-) = \begin{cases} D + E & \text{if } i \in \mathcal{P} \\ F + G & \text{if } i \in \mathcal{N} \end{cases} \tag{3.26}$$

D, E, F, G are defined in equations 3.18 - 3.21. $(h^+_{i,t}, h^-_{i,t})$ are the predicted latent variable for linearizing the concave part at iteration t.

Solving the convexified problem. A direct resolution of the convexified problem in the dual would be intractable, as for many other kernelized (latent) structured output problems. We adopt a cutting-plane strategy to train our SyMIL model, using the 1-slack formulation (Joachims et al. 2009). The learning algorithm is given in Algorithm 3.3. Cutting-plane training searches the optimal solution and the set of active constraints simultaneously in an iterative manner. This algorithm is guaranteed to converge to an approximate solution with a reasonable number of outer loops. Starting from an empty working set of constraints, in each iteration it solves the optimization problem (Line 4) with only the constraints of the working set. Then it finds the most violated constraint (Line 7-11) and adds it to the working set. $vex(\boldsymbol{x}_i, h_i^+, h_i^-)$ is the convex term for bag $\boldsymbol{x}_i$ and the equation is given in Equation 3.27.

$$vex(\boldsymbol{x}_i, h_i^+, h_i^-) = \begin{cases} \frac{N}{N^+} \max(0, \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) \rangle - 1) & \text{if } i \in \mathcal{P} \\ \quad +\lambda \max(1 - \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-) \rangle, \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) \rangle) & \\ \frac{N}{N^-} \max(0, -\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-) \rangle - 1) & \text{if } i \in \mathcal{N} \\ \quad +\lambda \max(1 + \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^+) \rangle, -\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, h_i^-) \rangle) & \end{cases} \tag{3.27}$$

The algorithm stops once no constraint can be found that is violated by more than the desired precision ε (Line 12). In our implementation, we use a stopping criterion defined by a fix number of iterations. During each iteration, we use MOSEK[3] to solve the quadratic problem with the given set of active constraints (Line 4).

3.4 Evaluation

This section describes experiments performed and provides both qualitative and quantitative analysis on SyMIL models. We first perform experiments on toy datasets composed of synthetic or real data, and then evaluate SyMIL on standard MIL datasets and on weakly supervised object detection datasets.

3.4.1 Toy Experiments

In this section, we perform toy experiments on synthetic and real (image and text) data to validate the intuition of our model.

3.4.1.1 Synthetic data

We first show that SyMIL better separates classes than LSVM on simple datasets where both classes have similar distributions. We design synthetic toy datasets where positive and negative bags are modeled in a symmetric manner: instances are generated from two different Gaussian distributions, with a parameter α controlling the distance between them (see Figure 3.3). The smaller the α, the more instances are shared between positives and negatives bags: the overlapping region contains "non-discriminative" instances for the classification task, because they are shared by both classes. We generate 2D Gaussian distributions and sample 500 bags with 20 instances for each class, for 10 values $\alpha \in [0.1, 1]$. The performances are evaluated using a 5-fold cross-validation: we train a linear model with 400 positive and 400 negative bags, evaluate the performance (accuracy) on other 100 test positive and negative bags, and average the results over the five folds.

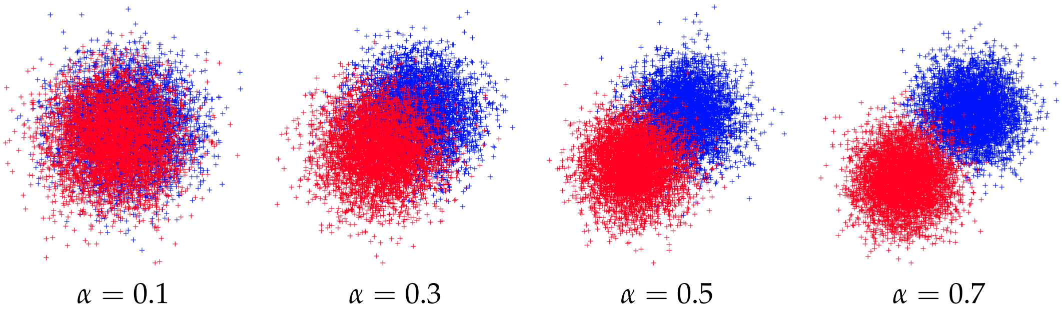
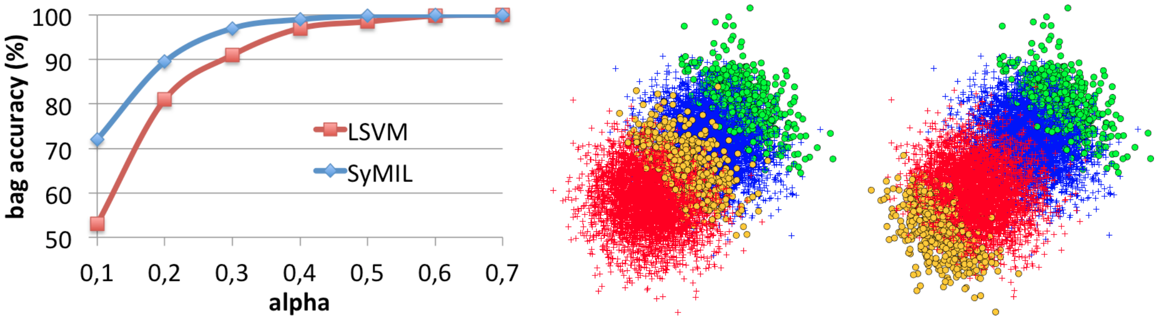


Figure 3.3.: Toy datasets generated with different α values. The blue points (resp. red) are the instances of the positive bags (resp. negative bags)

Figure 3.4 a) shows the performance evolution when varying α for SyMIL and LSVM. For large overlap values ($\alpha \geq 0.6$), the classification task is easy and both models have similar

[3] www.mosek.com

performances. When the task becomes more challenging, ($\alpha \leq 0.3$), SyMIL outperforms by more than 5 pts LSVM. To analyze the performance gain and interpret the latent representation learned by the different models, we visualize the instances selected by SyMIL and LSVM during training in Figure 3.4 b) and Figure 3.4 c), respectively. We can notice that SyMIL selected instances are discriminative for the positive and negative class, i.e. they belong to a region in the feature space that is not shared between the two gaussians. On the contrary, the instances selected by LSVM for the negative class essentially belong to the background area, and are shared by the two classes. This experiment validates the relevance of our model for bag classification.



a) Test accuracy w.r.t. α b) LSVM c) SyMIL

Figure 3.4.: Toy dataset: test accuracy with respect to α, and visualization of the selected instances during training ($\alpha = 0.5$): green over positive instances (in blue), orange over negative ones (in red).

3.4.1.2 Real image data

We now validate our intuition on real data. We use Mammal dataset (Heitz et al. 2009), which is a multi-class dataset containing 6 categories: *bison, deer, elephant, giraffe, llama* and *rhino*. We use the same protocol as in (Miller et al. 2012): the latent space $\mathcal{H}$ is composed of constant-size rectangular regions (which is a reasonable assumption for this dataset), and Histogram of Oriented Gradients (HOG) descriptors (Dalal et al. 2005) are used as features $\Phi(\boldsymbol{x}, h)$ for each region h. To have a similar setting as in Figure 3.1, we use only the classes bison and llama. We report in Table 3.1 the classification results for bison vs llama (bison (resp. llama) is the positive (resp. negative) class) and llama vs bison (llama (resp. bison) is the positive (resp. negative) class). The performances are evaluated using a 10-fold cross-validation.

We note that our model has better results than LSVM. For bison vs llama experiment, SyMIL seeks discriminative regions for both bison and llama, while LSVM seeks discriminative regions only for bison. We note that reverse the positive and negative class gives different results for LSVM, because it use an asymmetric strategy, whereas our model gives the same results, because it use a symmetric strategy.

Method	BISON VS LLAMA	LLAMA VS BISON
LSVM	90.3	87.7
SyMIL	95.7	95.7

Table 3.1.: Classification performances (accuracy) on Mammal dataset

3.4.1.3 Real text data

We finally analyze our model on a dataset where the negative class is composed of several concepts. We perform experiments on a text dataset from Reuters21578[4]. We choose the category *money* as positive examples and *ship*, *crude* as negative. 100 documents from the 3 categories are randomly selected. Each document is a bag, and each paragraph is an instance. To represent each paragraph, we use `tf-idf` feature with vocabulary of size 18933. Performances are evaluated using a 10-times 5-fold cross-validation.

Results given in Table 3.2 a) show that SyMIL outperforms LSVM in terms of predictive accuracy (97.6% *vs* 96.3%). To analyze the instances selected by the two models, we compute the semantic similarity between the words in the selected instances and the related category, using Wu and Palmer (WP) similarity measure (Palmer et al. 1995) on WordNet[5]. More precisely, the similarity is determined by computing the ratio of words that have a WP similarity with respect to the category larger than a threshold (set here to 0.2). For negative bags, we use the maximum similarity between *ship* or *crude*. In Table 3.2 b), we notice that LSVM and SyMIL perform similarly (74% vs 73 %) for positive bags, whereas SyMIL is much better than LSVM for negative bags (78% vs 67 %). This highlights the superiority of the symmetric modeling to select instances which are representative of the negative class. Finally, Table 3.2 c) shows an example of the 5 words that mostly contribute to the decision function. The top 5 selected words are generated as follows: for each selected instance (i.e. paragraph) we compute the top 5 words (i.e. dimensions in the instance space) that mostly contribute to the classification score (largest components of $|w|$), and average over all positive/negative bags. More precisely, for word k, we compute a histogram of contribution $w_k \times \Phi(\boldsymbol{x}, h^{predict})[k]$. We can point out that SyMIL extracts words that are semantically in touch with the negative class, e.g. (oil, OPEC) for crude and (port, shipping) for ship. On the contrary, LSVM selected words are not always semantically meaningful for the negative class, and are even more related to the positive class (*money*).

3.4.2 Standard MIL Dataset Results

We demonstrate the efficiency of SyMIL on standard MIL datasets[6] with three different applications: molecule categorization, automatic image annotation, and text categorization. We start by giving details of datasets:

[4] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

[5] http://wordnet.princeton.edu

[6] The datasets are available at http://www.cs.columbia.edu/andrews/mil/datasets.html

	LSVM	**SyMIL**
a) Predictive accuracy		
	96.3%	97.6%
b) Similarity between instances and category		
	Bag ⊕ = 74% Bag ⊖ = 67%	Bag ⊕ = 73% Bag ⊖ = 78%
c) Examples		
Bag ⊕	bank, currency, money, exchange, treasury	bank, exchange, rate, currency, monetary
Bag ⊖	west, finance, bank, british, money	oil, opec, shipping, port, union

Table 3.2.: Instance selection for text classification: a) predictive accuracy b) words in selected instances which are a semantically correlated to the category and c) example of top 5 selected words. The positive (resp. negative) bags are texts of the category *money* (resp. categories *ship* and *crude*).

- *Musk datasets*: consists of descriptions of molecules using multiple low-energy conformations. Each conformation is represented by a 166-dimensional feature vector derived from surface properties.
- *Image datasets*: an image consists of a set of segments, each characterized by color, texture and shape descriptors. There are three different datasets ("elephant", "fox", "tiger"). In each case, the dataset has 100 positive and 100 negative example images. The latter have been randomly drawn from a pool of photos of other animals.
- *Text datasets*: starting from the publicly available TREC9 data set, each document is split into passages using overlapping windows of maximal 50 words each. Then, documents are annotated with MeSH terms (Medical Subject Headings), each defining a binary concept.

Table 3.3 provides information about the number of training examples, the average number of instances per bag for each dataset, and the dimension of the features.

Dataset	Image	Musk1	Musk 2	Text
pos/neg bags	100/100	47/45	39/63	200/200
instances/bag	~ 6.5	5.17	64.69	~ 8
feature dimension	230	166	166	$\sim 66\,500$

Table 3.3.: Dataset Statistics. The features of text datasets are sparses.

The parametrization for our method is the following. Regarding hyper-parameters (Equation 3.7), C is fixed to a large value[7] (10^4). λ is chosen by cross-validation on the training set, on the range $\{0.1, 0.2, 0.5, 1\}$. We evaluate our method with linear and Radial Basis Function (RBF) kernels ($k(\boldsymbol{a}, \boldsymbol{b}) = \exp(-\gamma\|\boldsymbol{a} - \boldsymbol{b}\|_2^2)$). The scale parameter γ for RBF kernels is determined as the mean pairwise instance distance on the training set. Note that we could expect further improving performances by cross-validating it more carefully. For all methods, the initial latent variables are randomly selected. We follow

[7] We notice it has a small impact if it is sufficiently large.

Method	Image	Musk	Text	Avg.
a) Re-implemented method				
mi-SVM	73.4	84.5	81.6	
MI-SVM	75.5	81.7	80.3	
LSVM	74.4	82.7	80.0	
SyMIL linear	79.1	88.2	**84.8**	
RBF	**80.2**	89.2	-	
Without constraints 1 & 2 linear	78.1	86.9	83.7	
RBF	78.7	87.5	-	
b) State-of-the art results				
SyMIL	**80.2**	89.2	**84.8**	**84.7**
mi-SVM (Andrews et al. 2003)	72.9	85.5	81.6	80.0
MI-SVM (Andrews et al. 2003)	74.4	81.1	81.4	79.0
ALP-SVM (Gehler et al. 2007)	77.9	86.3	-	-
MICA (Mangasarian et al. 2008)	73.9	87.5	82.3	80.1
MIGraph (Z.-H. Zhou et al. 2009)	76.1	**90.0**	-	-
MiGraph (Z.-H. Zhou et al. 2009)	78.1	89.6	-	-
MI-CRF (Deselaers et al. 2010)	78.5	86.7	-	-
Convex relaxation (Joulin et al. 2012)	75.8	-	-	-
GP-WDA (Kim et al. 2013)	79.0	88.4	83.2	83.5
eMIL (Krummenacher et al. 2013)	77.0	85.3	82.7	81.7
MILEAGE (D. Zhang et al. 2013)	77.7	-	-	-

Table 3.4.: Bag classification accuracy (%) on the three datasets. Boldfaced numbers indicate best results.

the standard protocol to evaluate performances, as described in (Andrews et al. 2003): the performances are evaluated using a 10-times 10-fold cross-validation.

The overall results for the three kind of datasets (image, text, molecule) are gathered in Table 3.4. A first comparison is given in Table 3.4 a) with methods the most closely connected to ours: mi-SVM/MI-SVM (Andrews et al. 2003) and LSVM (Felzenszwalb et al. 2010). From a modeling point of view, these approaches basically differ from ours by the way instances are selected in positive and negative bags during training. We re-implement the three methods in order to compare the methods on the same splits. For mi-SVM and MI-SVM, we use linear kernels, that were reported to achieve optimal performances[8] (Andrews et al. 2003). One can notice SyMIL with linear kernel significantly outperforms mi-SVM, MI-SVM and LSVM: on average in the three types of data, there is a gain of about 4 pt over the best baseline. We perform paired t-test to assess the statistical significance of the difference in each dataset. It turns out that SyMIL is statistically better than its competitors with a risk of 1% for all image and molecule datasets, and for 5 of the 7 text

[8] Note that our re-implementation matches the results in reported in (Andrews et al. 2003).

datasets. These results clearly highlight the relevance of our model, i.e. the importance of seeking discriminative instances in both positive and negative bags.

Using non-linear kernels can further improve performances: about 1 pt increase in the image and molecule datasets. However, for the text datasets, the linear model outperforms RBF kernels. Note that this trend is conform to the results reported in GP-WDA (Kim et al. 2013). We also evaluate the performance reached when using the LSSVM (Subsection 2.2.2.2) instantiation corresponding to our prediction function, i.e. $\Psi(\boldsymbol{x}, y, h) = y \cdot \Phi(\boldsymbol{x}, h)$. As explained in Subsection 3.2.2, the SyMIL learning scheme is different from this LSSVM instantiation, which translates in ignoring constraints 1 (Equation 3.4) & 2 (Equation 3.5). We observe a performance drop between 1 and 3 pt depending on the dataset (Image-Molecule), and on the kernel type (linear vs RBF). For example, the superiority of our method is largely significant on Elephant (t-test validation with a risk of 5%). It confirms that enforcing constraints 1 & 2 is relevant and favorably impacts classification performances.

An absolute performance comparison with recent state-of-the-art works is provided in Table 3.4 b). On average on the three datasets, our method outperforms all reported results[9]. Competitive approaches in these datasets include recent works such as MILEAGE (D. Zhang et al. 2013), GP-WDA (Kim et al. 2013) which solves the MIL problem using Gaussian Processes, eMIL (Krummenacher et al. 2013) or MI-CRF (Deselaers et al. 2010) or MIGraph (Z.-H. Zhou et al. 2009). Despite the complex models used by these strong competitors, SyMIL outperforms them in the image and text databases. Although our method remains very competitive on the Musk datasets, it is slightly outperformed by MIGraph. One explanation may be that MIL assumptions are better satisfied on this historical dataset. Note, however, that MIGraph performs poorly on the image dataset. To summarize, the excellent results for the three applications exhibit the capacity of our method to successfully handle various types of data. Note that the local information in SyMIL can be combined with a global bag feature, as done in MILEAGE (D. Zhang et al. 2013) or MI-CRF.

Analysis of hyper-parameter λ We also study the performances with respect to the parameter λ, which is an important hyper-parameter for SyMIL model. This hyper-parameter adjusts the trade-off between constraints 1 & 2 and constraint 3 during training. A large λ is similar to LSSVM (see Subsection 3.2.2) because the constraints 1 & 2 are negligible with respect to the constraint 3. Figure 3.5 shows the results on Musk2 and Elephant datasets, for a λ on the range [0.01, 1000]. We observe that the best results are for a lambda around 1 on Musk2, and 0.1 on Elephant. The optimal λ changes for each dataset. Note that using a small λ leads to bad results because the model is not able to predict the relevant instance.

[9] SyMIL results are reported for RBF kernels in the image and molecules datasets, but for linear kernels in the text datasets. This is similar to the setup in (Kim et al. 2013), since linear models generally lead to better performances.

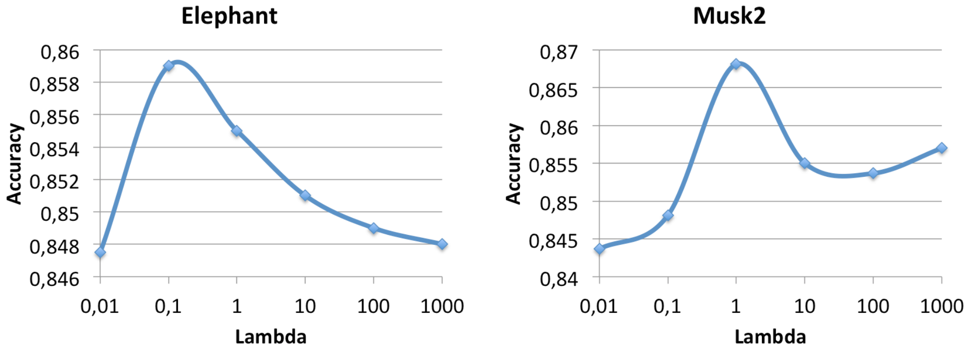


Figure 3.5.: Accuracy performance with respect to parameter λ (logarithmic scale) on Elephant and Musk2 datasets.

3.4.3 Weakly Supervised Object Detection

In weakly supervised object detection, the goal is to learn a model, which jointly classifies the image and localizes the object. Training data only have image-level labels indicating the presence/absence of each object category in an image. The exact object location in the image is unknown and is modeled as a latent variable h. We make experiments on two different datasets: Mammal dataset and PASCAL VOC 2007.

3.4.3.1 Classification results

Mammal dataset This dataset is presented in Subsection 3.4.1.2. To perform multi class classification, we use 1 vs All strategy. The performances are evaluated using a 10-fold cross-validation. We compare the proposed SyMIL RBF model to LSVM (Felzenszwalb et al. 2010) and its recently kernelized version (W. Yang et al. 2012), using a RBF kernel. In addition, we evaluate Max-Margin Min-Entropy (M3E) (Miller et al. 2012) by using the code available online[10]. The best performing M3E models use a small value of α, so we fix $\alpha = 5$ for our experiments. We measure prediction performances by using accuracy, and Mean Average Precision (MAP) to be robust to the $\oplus/\ominus$ unbalance.

Method	MAP (%)	Accuracy (%)
LSVM (Felzenszwalb et al. 2010)	67.9	89.3
KLSVM (W. Yang et al. 2012)	73.3	90.1
M3E 1vsAll (Miller et al. 2012)	71.9	91.1
SyMIL	**78.7**	**92.1**

Table 3.5.: Classification performances on Mammal dataset

The results are reported in Table 3.5. As we can see, our SyMIL model outperforms other approaches using both metrics. The trend is the same for both metrics (MAP & accuracy). In addition, all improvements are statistically significant (risk 5%). The prediction results again illustrate the superiority of the symmetric modeling, especially with respect to

[10] see M3E webpage.

Method	Classification MAP (%)	Train Overlap (%)	Test Overlap (%)
LSVM	76.21	36.38	41.99
SyMIL	78.37	42.71	43.42

Table 3.6.: Classification and localization performances on PASCAL VOC 2007.

Method	Train Ov.	Test Ov.	Train MAP	Test MAP
LSVM (Felzenszwalb et al. 2010)	59.8	61.3	40.2	40.7
KLSVM (W. Yang et al. 2012)	60.9	60.8	39.9	40.1
M3E 1vsAll (Miller et al. 2012)	62.5	60.9	44.3	42.7
SyMIL	**64.7**	**63.2**	**47.6**	**46.5**

Table 3.7.: Detection performances on Mammal dataset (Ov. = overlap)

Kernel Latent SVM (KLSVM) (W. Yang et al. 2012) where the comparison directly measures the impact of the min/max selection strategy. Our method also has an edge over M3E, which tackles the weakly supervised learning in a direction complementary to ours (modeling ambiguities between latent variables).

PASCAL VOC 2007 We perform another experiment on the PASCAL VOC 2007 dataset, presented in Section 2.4. For each image, we extract 25 regions of size 60% (of the image size) with a sliding window strategy. To have a vectorial representation of each region, we extract deep features pre-trained on ImageNet using MatConvNet library (A. Vedaldi et al. 2015) (model `vgg-m-2048`). Each region is represented by a 2048-dimensional vector (output of the `FC7` layer after the ReLU).

The classification results are shown in Table 3.6. We compare SyMIL and LSVM. As observed in previous experiments, SyMIL outperforms LSVM. It confirms that seek discriminative instances for both positive and negative class is relevant, even on challenging dataset like PASCAL VOC 2007.

3.4.3.2 Detection results

We analyze the predicted regions for weakly supervised object detection. We report localization performances to quantitatively evaluate the quality of the predicted latent values. We use the standard detection metric (Everingham, Van Gool, et al. 2007), measuring the overlap between the predicted and ground truth bounding boxes. We consider that a prediction is correct if the overlap is larger than 0.5.

Table 3.7 summarizes the average performances for both detection metrics on Mammal dataset. SyMIL outperforms asymmetric approaches for both metrics. In addition, all improvements are statistically significant (risk 5%). Detection results are connected to prediction performances. They quantitatively validate the motivation of the method, i.e. the fact that SyMIL is better able than asymmetric MIL models to track the structure of the negative class. In this dataset, we show that SyMIL successfully localizes regions containing object of the five categories composing the negative class. Visualizations

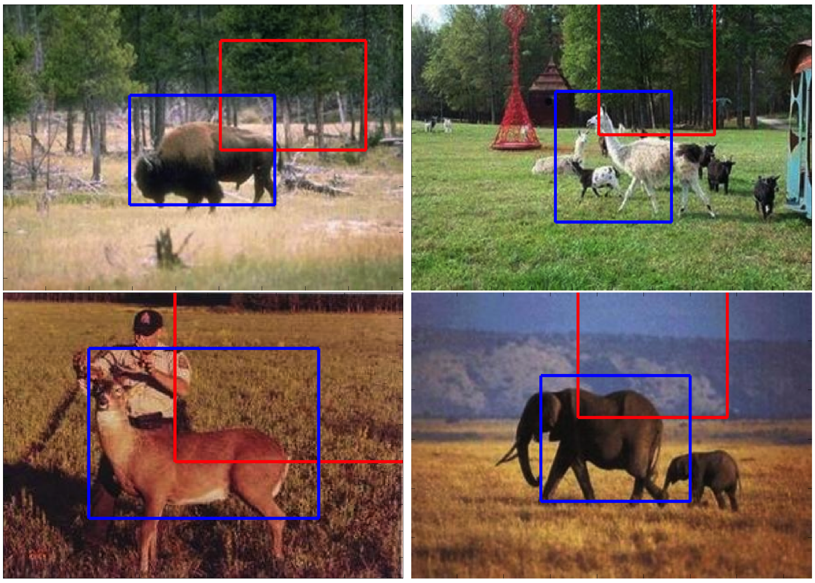


Figure 3.6.: Visualization of predicted latent variable for negative examples on Mammal Dataset: LSVM (red) and SyMIL (blue)

of weakly supervised objects detection are given in Figure 3.6. LSVM tends to predict background regions, because of the asymmetry of the model, whereas SyMIL predicts foreground regions.

For VOC 2007, we normalize the overlap by the area of the predicted bounding box. Used the intersection over union score is not adapted, because we have only one size of box. If the ground truth bounding box is smaller than the size of the bounding boxes, it is not possible to have a good score even if the ground truth is in the predicted region. We observe similar results as on Mammal Dataset. SyMIL achieves better results for classification (+2,1%) and detection (+6% on training set) than LSVM (Table 3.6).

3.5 Conclusion

In this chapter, we introduced SyMIL, a new model for learning from weakly labeled data. Following Learning with Label Proportion (LLP) ideas, SyMIL models positive and negative bags in a symmetric manner, using a pair of latent variables. SyMIL is trained by defining a regularized large margin objective function, with specific dedicated constraints. We provided a primal solver based on Stochastic Gradient Descent (SGD) and a dual solver based on one-slack cutting-plane. In addition, we derived a generalization error bound based on the Rademacher complexity, which shows that our model has the same asymptotic behavior that standard MIL models.

Successful experimental results are reported on standard MIL and weakly supervised object detection datasets: SyMIL significantly outperforms competitive methods (mi-SVM, MI-SVM, LSVM), and gives very competitive performances compared to state-of-the-art works. The analysis of the SyMIL instance selection strategy on weakly supervised object detection and text classification tasks revealed the capacity of the symmetric modeling to track the structure of the negative class.

A limitation of SyMIL relies on the prediction function which selects a single instance, making the method sensitive to false alarm outliers. To improve the prediction, it could be interesting to combine SyMIL approach with LLP (see Appendix A) or to extend the prediction function to multiple instances, i.e. to select the k most positive (resp. negative) instances instead of the most positive (resp. negative) instance (Chapter 5). Another interesting strategy is to model interactions between bag instances as is done in MI-CRF or MIGraph. Another limitation of SyMIL is that bags are labeled with binary labels, while a lot of datasets are labeled with multi-class labels. To address this problem, a solution is to use WSL models for structured output prediction, e.g. LSSVM. In the next chapter, we present a new model which generalizes SyMIL to structured output prediction, e.g. multi-class, ranking, etc.

Chapter 4

MANTRA: MINIMUM MAXIMUM LATENT STRUCTURAL SVM FOR IMAGE CLASSIFICATION AND RANKING

Contents

Chapter abstract

The SyMIL model introduced in the previous chapter is based on a pair of latent variables $(\boldsymbol{h}^+, \boldsymbol{h}^-)$ *for prediction. While the SyMIL model is limited to binary labels, we propose to extend this model to more general outputs. We instantiate our model for two different visual recognition tasks: multi-class classification and ranking. For ranking, we propose efficient solutions to exactly solve the inference and the loss-augmented problems. Finally, we validate the relevance of our model on six different datasets.*

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

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2015). “MANTRA: Minimum Maximum Latent Structural SVM for Image Classification and Ranking”. In: *IEEE International Conference on Computer Vision (ICCV)*

4.1 Introduction

Deep learning with ConvNets is becoming a key ingredient of visual recognition systems (Subsection 2.1.2). Internal ConvNet representations trained on large scale datasets currently provide state-of-the-art features for various tasks, e.g. image classification or object detection. To overcome the limited invariance capacity of ConvNets, bounding box annotations are often used. However, collecting full annotations for all images in a large dataset is an expensive task, and makes the development of Weakly Supervised Learning (WSL) models appealing (Section 1.2).

In Chapter 3, we introduced a `max+min` prediction function for WSL dedicated to binary classification. In this chapter, we introduce Minimum mAximum lateNt sTRucturAl SVM (MANTRA), a new WSL model for structured output that extends SyMIL to more general outputs (multi-class label, ranking). Our first contribution is the extension of the `max+min` prediction function to structured output (Section 4.2). We also show that the minimum latent variable explicitly models negative evidence, i.e. looks for "counter evidence" for the class. Our second contribution is that the MANTRA prediction function enables to efficiently tackle the important problem of learning to rank (Subsection 4.3.2), since many computer vision tasks are evaluated with ranking metrics, e.g. Average Precision (AP) in PASCAL VOC. Optimizing ranking models with AP is challenging because the AP does not decompose linearly in the examples. Finally, extensive experiments highlight the relevance of MANTRA for multi-class classification (Subsection 4.4.1) and ranking (Subsection 4.4.2).

4.2 MANTRA Model

We present here the proposed WSL model: Minimum mAximum lateNt sTRucturAl SVM (MANTRA). We follow the notations introduced in Subsection 2.2.2. We assume that a joint feature map $\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) \in \mathbb{R}^d$, describing the relation between input $\boldsymbol{x}$, output $\boldsymbol{y}$, and latent variable $\boldsymbol{h}$, is designed. For example for multi-class image classification, $\boldsymbol{x}$ is the image, $\boldsymbol{y}$ is a multi-class label, $\boldsymbol{h}$ is a region of image $\boldsymbol{x}$ and $\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$ is the vectorial representation of region $\boldsymbol{h}$ of image $\boldsymbol{x}$ for label $\boldsymbol{y}$. Our goal is to learn a prediction function $f_{\boldsymbol{w}}$, parametrized by $\boldsymbol{w} \in \mathbb{R}^d$, to predict the output. During training, we assume that we are given a set of N training pairs $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\}_{i=1}^N \in (\mathcal{X} \times \mathcal{Y})^N$. Our goal is to optimize $\boldsymbol{w}$ in order to minimize a user-supplied loss function $\Delta(\boldsymbol{y}_i^\star, \hat{\boldsymbol{y}})$ over the training set.

4.2.1 Prediction Function

As mentioned earlier, the main intuition of the proposed MANTRA model is to equip each possible output $\boldsymbol{y} \in \mathcal{Y}$ with a pair of latent variables $(\boldsymbol{h}_{i,\boldsymbol{y}}^+, \boldsymbol{h}_{i,\boldsymbol{y}}^-)$. $\boldsymbol{h}_{i,\boldsymbol{y}}^+$ (resp. $\boldsymbol{h}_{i,\boldsymbol{y}}^-$) corresponds to the max (resp. min) scoring latent value, for input $\boldsymbol{x}_i$ and output $\boldsymbol{y}$:

$$\boldsymbol{h}_{i,\boldsymbol{y}}^+ = \arg\max_{\boldsymbol{h} \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}) \rangle \qquad \boldsymbol{h}_{i,\boldsymbol{y}}^- = \arg\min_{\boldsymbol{h} \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}) \rangle \tag{4.1}$$

For an input/output pair $(\boldsymbol{x}_i, \boldsymbol{y})$, the scoring of the model, $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$, sums $\boldsymbol{h}^+_{i,\boldsymbol{y}}$ and $\boldsymbol{h}^-_{i,\boldsymbol{y}}$ scores, as follows:

$$s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) = \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}^+_{i,\boldsymbol{y}}) \rangle + \langle \boldsymbol{w}, \Psi(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}^-_{i,\boldsymbol{y}}) \rangle \tag{4.2}$$

Finally, MANTRA prediction outputs $\hat{\boldsymbol{y}}_i = f_{\boldsymbol{w}}(\boldsymbol{x}_i)$ which maximizes $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$ w.r.t. $\boldsymbol{y}$:

$$\hat{\boldsymbol{y}}_i = f_{\boldsymbol{w}}(\boldsymbol{x}_i) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) \tag{4.3}$$

Connection with SyMIL If we consider the feature map $\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) = \boldsymbol{y} \cdot \Phi(\boldsymbol{x}, \boldsymbol{h})$ and $\mathcal{Y} = \{-1, +1\}$, the MANTRA prediction function is equivalent to SyMIL prediction function (Equation 3.2).

Intuition Let us consider a multi-class classification instantiation of MANTRA, where latent variables $\boldsymbol{h}$ correspond to part localizations. To highlight the importance of the pair $(\boldsymbol{h}^+, \boldsymbol{h}^-)$, we show in Figure 4.1, for an image of the class *library*, classification scores for each latent location using (on the left) the *library* classifier s_l (the correct one) and (on the right) the *cloister* classifier s_c (a wrong one). $\boldsymbol{h}^+$ (resp. $\boldsymbol{h}^-$) regions are boxed in red (resp. blue). As we can see, the prediction score $s_l(\boldsymbol{h}^+) = 1.8$ for the correct *library* classifier is large, since the model finds strong local evidence $\boldsymbol{h}^+$ of its presence (bookcase), and no clear evidence of its absence (medium score $s_l(\boldsymbol{h}^-) = 0.1$). Contrarily, the prediction score for the *cloister* classifier s_c is substantially smaller: although the model heavily fires on the vault ($s_c(\boldsymbol{h}^+) = 1.5$), it also finds clear evidence of the absence of *cloister*, here books ($s_c(\boldsymbol{h}^-) = -0.8$). As a consequence, MANTRA correctly predicts the class *library*. Note that the vector of parameters $\boldsymbol{w}$ is common for both latent variables $\boldsymbol{h}^+$ and $\boldsymbol{h}^-$. The same model learns which regions are discriminative for the class and which regions indicate the absence of the class. The minimum region $\boldsymbol{h}^-$ can be seen as a regularizer on the latent space exploiting contextual information.

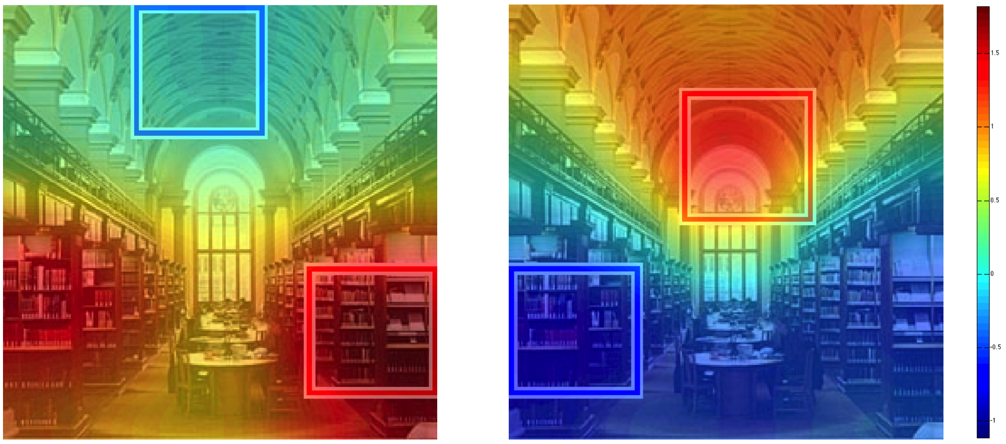
4.2.2 Learning Formulation

During training, we enforce the following constraints:

$$\forall \boldsymbol{y} \neq \boldsymbol{y}_i^\star, \quad s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) \geq \Delta(\boldsymbol{y}_i^\star, \boldsymbol{y}) + s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) \tag{4.4}$$

Each constraint in Equation 4.4 requires the scoring value $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)$ for the correct output $\boldsymbol{y}_i^\star$ to be larger than the scoring value $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$ for each incorrect output $\boldsymbol{y} \neq \boldsymbol{y}_i^\star$, plus a margin of $\Delta(\boldsymbol{y}_i^\star, \boldsymbol{y})$. $\Delta(\boldsymbol{y}_i^\star, \boldsymbol{y})$, a user-specified loss, makes it possible to incorporate domain knowledge into the penalization. To give some insights of how the model parameters can be adjusted to fulfill constraints in Equation 4.4, let us notice that:

- $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)$, i.e. the score for the correct output $\boldsymbol{y}_i^\star$, can be increased if we find statistically high scoring variables $\boldsymbol{h}^+_{i,\boldsymbol{y}_i^\star}$, which represent strong evidence for the presence of $\boldsymbol{y}_i^\star$, while enforcing $\boldsymbol{h}^-_{i,\boldsymbol{y}_i}$ variables not having large negative scores.
- $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$, i.e. the score for an incorrect output $\boldsymbol{y} \neq \boldsymbol{y}_i^\star$, can be decreased if we find low scoring variables $\boldsymbol{h}^+_{i,\boldsymbol{y}}$, limiting evidence of the presence of $\boldsymbol{y}$, while seeking $\boldsymbol{h}^-_{i,\boldsymbol{y}}$ variables with large negatives scores, supporting the absence of output $\boldsymbol{y}$.



a) $s_l(\boldsymbol{h}^+) = 1.8$; $s_l(\boldsymbol{h}^-) = 0.1$

b) $s_c(\boldsymbol{h}^+) = 1.5$; $s_c(\boldsymbol{h}^-) = -0.8$

Figure 4.1.: MANTRA prediction maps for *library* classifier s_l a) and *cloister* classifier s_c b), for an image of class *library*. For each class, MANTRA score is $s(\boldsymbol{h}^+) + s(\boldsymbol{h}^-)$: $\boldsymbol{h}^+$ (red) provides localized evidence for the class, whereas $\boldsymbol{h}^-$ (blue) reveals its absence.

To allow some constraints in Equation 4.4 to be violated, we introduce the following loss function:

$$\mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) = \max_{\boldsymbol{y} \in \mathcal{Y}} \left[\Delta(\boldsymbol{y}_i^\star, \boldsymbol{y}) + s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\right] \tag{4.5}$$

The loss function $\mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i)$ (Equation 4.5) is an upper bound of $\Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)$.

Proof. We show that the loss function $\mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)$ (Equation 4.5) is an upper bound of $\Delta(\boldsymbol{y}_i^\star, \hat{\boldsymbol{y}}_i)$, where $\boldsymbol{x}_i$ is the input, $\boldsymbol{y}_i^\star$ is the ground truth, and $\hat{\boldsymbol{y}}_i = \operatorname*{arg\,max}_{\boldsymbol{y} \in \mathcal{Y}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$ is the predicted output.

$$\Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) \leq \Delta(\boldsymbol{y}_i^\star, \hat{\boldsymbol{y}}_i) + \underbrace{s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)}_{\geq 0} \tag{4.6}$$

$$\leq \max_{\boldsymbol{y} \in \mathcal{Y}} \left[\Delta(\boldsymbol{y}_i^\star, \boldsymbol{y}) + s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\right] \tag{4.7}$$

$$\leq \mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) \tag{4.8}$$

This proves that $\mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)$ is an upper bound of $\Delta(\boldsymbol{y}_i^\star, \hat{\boldsymbol{y}}_i)$. □

Using the standard max margin regularization term $\|\boldsymbol{w}\|^2$, our primal objective function is defined as follows:

$$\mathcal{P}(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{C}{N}\sum_{i=1}^{N} \mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star) \tag{4.9}$$

4.2.3 Optimization

The problem in Equation 4.9 is not convex with respect to $\boldsymbol{w}$. To solve it, we propose an efficient optimization scheme based on a cutting plane algorithm with the one-slack formulation (Joachims et al. 2009). Our objective function in Equation 4.9 can thus be rewritten as follows:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\xi \quad s.t. \quad \forall(\hat{\boldsymbol{y}}_1,\ldots,\hat{\boldsymbol{y}}_N) \in \mathcal{Y}^N \tag{4.10}$$

$$\text{s.t.} \quad \frac{1}{N}\sum_{i=1}^{N} \Delta(\boldsymbol{y}_i^\star,\hat{\boldsymbol{y}}_i) + s_{\boldsymbol{w}}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}_i) - s_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}_i^\star) \le \xi$$

The key idea of the 1-slack formulation in Equation 4.10 is to replace the N slack variables (weak constraints) by a single shared slack variable ξ (strong constraint). It is shown in (Joachims et al. 2009) that this formulation helps speeding up the training of SSVMs, reducing the complexity from being super-linear to linear in the number of training examples.

Cutting Plane Algorithm Based on the 1-slack formulation, we use a cutting plane strategy to optimize Equation 4.10. Compared to sub-gradient methods, the cutting plane approach takes an optimal step in the current cutting plane model, leading to faster convergence (Tsochantaridis et al. 2005).

For convex optimization problems, the idea of the cutting plane method is to build an accurate approximation, under-estimating the objective function. However, it cannot be directly applied for solving non-convex optimization problems, because the cutting plane approximation might not be underestimating the objective at all points, with the risk of missing good local minima (Do et al. 2012). Based on (Do et al. 2012), we derive a non-convex cutting plane algorithm to solve Equation 4.10. In particular, we use a method to detect and solve conflicts when adding a new cutting plane, as in (Do et al. 2012), in order to avoid over-estimating the objective function. It is important to stress that the proposed approach consists in a direct optimization, contrarily to iterative methods, which usually solve a set of approximate problems, e.g. CCCP (Yuille et al. 2003).

The overall training scheme of MANTRA is shown in Algorithm 4.1. Starting from an initial cutting plane (Line 1), each cutting plane iteration consists in solving the resulting Quadratic Program (QP) problem with the working set of cutting planes $\boldsymbol{H}$ (Line 5). As in (Joachims et al. 2009), we solve the QP in the dual, because $|\boldsymbol{H}|$ is generally much smaller than the input dimension. The dual formulation of Equation 4.10 (Line 5) is derived in Appendix B. Then, the current $\boldsymbol{w}$ solution (Line 7) is used to find the most violated constraint $\hat{\boldsymbol{y}}$ for each example (Line 10). The $\hat{\boldsymbol{y}}$'s are used to compute $\boldsymbol{g}_t$ from the sub-gradient $\nabla_{\boldsymbol{w}}\mathcal{L}_{\boldsymbol{w}}$ (Line 13). The sub-gradient is:

$$\nabla_{\boldsymbol{w}}\mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}_i^\star) = \nabla_{\boldsymbol{w}}s_{\boldsymbol{w}}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}}s_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}_i^\star) \tag{4.11}$$

$$\text{where} \qquad \nabla_{\boldsymbol{w}}s_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}) = \Psi(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h}_{i,\boldsymbol{y}}^{+}) + \Psi(\boldsymbol{x}_i,\boldsymbol{y},\boldsymbol{h}_{i,\boldsymbol{y}}^{-}) \tag{4.12}$$

$\boldsymbol{g}_t$ serves to update the working set $\boldsymbol{H}$ at the next iteration. Finally, when adding a new cutting plane, we detect and solve conflicts (Line 15) using the method detailed in (Do et al. 2012). The algorithm stops as soon as no constraint can be found that is violated by more than the desired precision ε (Line 16).

Algorithm 4.1 Cutting Plane Algorithm for training MANTRA

```
Input: Training set $\{(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\}_{i=1,...,N}$, precision $\varepsilon$, $C$.
Initialize $t \leftarrow 1$, $\{\hat{\boldsymbol{y}}_i, \boldsymbol{h}^+_{i,\hat{\boldsymbol{y}}_i}, \boldsymbol{h}^-_{i,\hat{\boldsymbol{y}}_i}\}_{i=1,...,N}$ and compute initial cutting plane $(\boldsymbol{g}_1, c_1)$
repeat
    // Update working set and solve QP
    $\boldsymbol{H} \leftarrow (H_{ij})_{1 \le i,j \le t}$ where $H_{ij} = \langle \boldsymbol{g}_i, \boldsymbol{g}_j \rangle$
    $\boldsymbol{\alpha} \leftarrow \arg\max_{\boldsymbol{\alpha} \ge 0} \boldsymbol{\alpha}^T \boldsymbol{c} - \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{H} \boldsymbol{\alpha} \quad \text{s.t.} \quad \boldsymbol{\alpha}^T 1 \le C$
    $\xi \leftarrow \frac{1}{C}(\boldsymbol{\alpha}^T \boldsymbol{c} - \boldsymbol{\alpha}^T \boldsymbol{H} \boldsymbol{\alpha})$
    $\boldsymbol{w} \leftarrow \sum_{i=1}^t \alpha_i \boldsymbol{g}_i$
    $t \leftarrow t + 1$
    for i=1 to N do
        $\hat{\boldsymbol{y}}_i = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} \mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)$    // Loss-augmented inference
    end for
    // Compute new cutting plane and solve conflict
    $\boldsymbol{g}_t \leftarrow \frac{1}{N}\sum_{i=1}^N -\nabla_{\boldsymbol{w}} \mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)$
    $c_t \leftarrow \frac{1}{N}\sum_{i=1}^N \Delta(\boldsymbol{y}_i^\star, \hat{\boldsymbol{y}}_i)$
    $(\boldsymbol{g}_t, c_t) \leftarrow SolveConflict(\boldsymbol{w}, \boldsymbol{g}_t, c_t)$
until $\langle \boldsymbol{w}, \boldsymbol{g}_t \rangle \ge c_t - \xi - \varepsilon$
return $\boldsymbol{w}$
```

4.3 MANTRA Instantiation

MANTRA instantiation consists in specifying a particular joint feature Ψ and loss function Δ. For each instantiation, training the model requires solving two problems: inference (Equation 4.3), and loss-augmented inference (Equation 4.13):

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} \Delta(\boldsymbol{y}_i^\star, \boldsymbol{y}) + s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) \tag{4.13}$$

In this section, we instantiate MANTRA for two WSL detection tasks: multi-class classification and ranking.

4.3.1 Multi-class Instantiation

For multi-class classification, the input $\boldsymbol{x}$ is an image, and the latent variable $\boldsymbol{h}$ is the location of a region (bounding box) in the image. The output space is the set of classes $\mathcal{Y} = \{1, \dots, K\}$, where K is the number of classes. We use the standard joint feature map:

$$\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) = [I(\boldsymbol{y} = 1)\Phi(\boldsymbol{x}, \boldsymbol{h}), \dots, I(\boldsymbol{y} = K)\Phi(\boldsymbol{x}, \boldsymbol{h})] \tag{4.14}$$

where $\Phi(\boldsymbol{x}, \boldsymbol{h}) \in \mathbb{R}^d$ is a vectorial representation of image $\boldsymbol{x}$ at location $\boldsymbol{h}$, and $I(\boldsymbol{y} = k) = 1$ if $\boldsymbol{y} = k$ and $I(\boldsymbol{y} = k) = 0$ if $\boldsymbol{y} \neq k$. $\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$ is then a $(K \times d)$-dimensional vector. The loss function Δ is the $0/1$ loss. The inference and the loss-augmented inference are exhaustively solved.

4.3.2 AP Ranking Instantiation

Context The Average Precision (AP) metric is often used in information retrieval and many computer vision tasks, such as PASCAL challenges (Everingham, Eslami, et al. 2015). Optimizing ranking models with AP is challenging even in the fully supervised case. An elegant instantiation of SSVM is introduced in (Yue et al. 2007), making it possible to optimize a convex upper bound over AP. To optimize this surrogate, the key issue is how to solve the Loss-Augmented Inference (LAI) problem. (Yue et al. 2007) propose a greedy algorithm to find an optimal solution. Recently, (Mohapatra et al. 2014) proposes a more efficient method by taking in account the structure of the problem. Instead of optimizing a convex surrogate, (Song et al. 2016) proposes a direct loss minimization to optimize AP.

In the WSL setting, optimizing AP is, however, a very challenging problem: for example, no algorithm exists for solving the loss-augmented inference problem with LSSVM (Subsection 2.2.2.2). In (Behl et al. 2015), Latent AP-SVM (LAPSVM) is introduced, enabling a tractable optimization by defining an ad-hoc prediction rule dedicated to ranking: first the latent variables are fixed, and then an optimal ranking with fixed latent variables is found. In this section, we propose a MANTRA AP ranking instantiation, which allows an efficient solving of both inference and loss-augmented inference problems.

Contrary to multi-class classification, for AP ranking there is one example that contains all the examples, i.e. $\boldsymbol{x} = \{\boldsymbol{x}_i, i = 1, \dots, N\}$. This formalism can be explained by the fact that we need all the examples to evaluate the AP loss (Yue et al. 2007). The indexes for the positive and negative samples are denoted by $\mathcal{P}$ and $\mathcal{N}$ respectively. In other words, if $p \in \mathcal{P}$ and $n \in \mathcal{N}$ then $\boldsymbol{x}_p$ belongs to positive class and $\boldsymbol{x}_n$ belongs to the negative class. The structured output is a ranking matrix $\boldsymbol{y}$ of size $N \times N$ providing an ordering of the training examples, such that:

- $y_{ij} = 1$ if $\boldsymbol{x}_i \prec_{\boldsymbol{y}} \boldsymbol{x}_j$ i.e. $\boldsymbol{x}_i$ is is ranked ahead of $\boldsymbol{x}_j$;
- $y_{ij} = -1$ if $\boldsymbol{x}_j \prec_{\boldsymbol{y}} \boldsymbol{x}_i$ i.e. $\boldsymbol{x}_j$ is is ranked ahead of $\boldsymbol{x}_i$;
- $y_{ij} = 0$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are assigned the same rank.

$\boldsymbol{y}^*$ is the ground-truth ranking matrix, *i.e.* $y^*_{ij} = 1$ if $(i,j) \in \mathcal{P} \times \mathcal{N}$, $y^*_{ij} = 0$ if $(i,j) \in \mathcal{P} \times \mathcal{P}$ and $y^*_{ij} = 0$ if $(i,j) \in \mathcal{N} \times \mathcal{N}$. We now present the joint feature map and the loss function.

Joint Feature Map The joint feature map $\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$ is defined as follows:

$$\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) = \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} y_{pn} (\Phi(\boldsymbol{x}_p, \boldsymbol{h}_{p,n}) - \Phi(\boldsymbol{x}_n, \boldsymbol{h}_{n,p})) \tag{4.15}$$

The latent space $\mathcal{H}$ corresponds to the set of latent variables for each pair of positive-negative examples: $\boldsymbol{h} = \{(\boldsymbol{h}_{p,n}, \boldsymbol{h}_{n,p}) \in \mathcal{H}_p \times \mathcal{H}_n, (p,n) \in \mathcal{P} \times \mathcal{N}\}$, where $\mathcal{H}_p$ (resp. $\mathcal{H}_n$) is the set of locations in image $\boldsymbol{x}_p$ (resp. $\boldsymbol{x}_n$). $\Phi(\boldsymbol{x}_p, \boldsymbol{h}_{p,n}) \in \mathbb{R}^d$ (resp. $\Phi(\boldsymbol{x}_n, \boldsymbol{h}_{n,p})$) is thus a vectorial representation of image $\boldsymbol{x}_p$ (resp. $\boldsymbol{x}_n$) at location $\boldsymbol{h}_{p,n}$ (resp. $\boldsymbol{h}_{n,p}$). Note that $\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$ in Equation 4.15 is a generalization of the feature map used in (Behl et al. 2015), where the selection of bounding boxes is specific to each image pair.

Loss Function During training, the goal is to minimize a given ranking loss function. We especially focus on AP, with $\Delta_{AP}(\boldsymbol{y}^\star, \boldsymbol{y}) = 1 - AP(\boldsymbol{y}^\star, \boldsymbol{y})$, where $AP(\boldsymbol{y}^\star, \boldsymbol{y})$ is the AP score (Baeza-Yates et al. 1999) between rankings $\boldsymbol{y}^\star$ and $\boldsymbol{y}$. Optimizing Δ_{AP} is difficult, because Δ_{AP} does not decompose linearly in the examples (Yue et al. 2007). In the WSL setting, the problem is exacerbated: for example, no efficient algorithm currently exists for solving the loss-augmented inference problem in the LSSVM case (C.-N. Yu et al. 2009), as pointed out in (Behl et al. 2015).

We now show that inference and loss-augmented inference can be solved exactly and efficiently with MANTRA. Firstly, we show (Lemma 4.1) that in our ranking instantiation, $s_{\boldsymbol{w}}$ (Equation 4.2) can be computed a standard fully supervised feature map. This result has major consequences, which enables to decouple the optimization over $\boldsymbol{y}$ and $\boldsymbol{h}$.

Lemma 4.1. *$\forall(\boldsymbol{x}, \boldsymbol{y})$, $s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$ in Equation 4.2, for the ranking instantiation of Ψ given in Equation 4.15, rewrites as $\Theta(\boldsymbol{x}, \boldsymbol{y})$:*

$$\Theta(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} y_{pn} \left(\langle \boldsymbol{w}, \Phi_{-}^{+}(\boldsymbol{x}_p) \rangle - \langle \boldsymbol{w}, \Phi_{-}^{+}(\boldsymbol{x}_n) \rangle \right) \tag{4.16}$$

where $\langle \boldsymbol{w}, \Phi_{-}^{+}(\boldsymbol{x}_i) \rangle = \max_{\boldsymbol{h} \in \mathcal{H}_i} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, \boldsymbol{h}) \rangle + \min_{\boldsymbol{h} \in \mathcal{H}_i} \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i, \boldsymbol{h}) \rangle$

Proof. We prove that $s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$ can be re-written as a supervised feature map, where the score of each example $\boldsymbol{x}_i$ is $\langle \boldsymbol{w}, \Phi_{-}^{+}(\boldsymbol{x}_i) \rangle$. Given an input $\boldsymbol{x}$, and an output $\boldsymbol{y}$ and a weight vector $\boldsymbol{w}$, we have:

$$\begin{aligned}
s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) =& \max_{\boldsymbol{h} \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) \rangle + \min_{\boldsymbol{h}' \in \mathcal{H}} \langle \boldsymbol{w}, \Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}') \rangle && (4.17) \\
=& \max_{\boldsymbol{h} \in \mathcal{H}} \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} y_{pn} (\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_p, \boldsymbol{h}_{p,n}) \rangle - \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_n, \boldsymbol{h}_{n,p}) \rangle) && (4.18) \\
&+ \min_{\boldsymbol{h}' \in \mathcal{H}} \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} y_{pn} (\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_p, {\boldsymbol{h}'}_{p,n}) \rangle - \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_n, {\boldsymbol{h}'}_{n,p}) \rangle) && \\
=& \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} \max_{(\boldsymbol{h}_p, \boldsymbol{h}_n) \in \mathcal{H}_p \times \mathcal{H}_n} y_{pn} (\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_p, \boldsymbol{h}_p) \rangle - \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_n, \boldsymbol{h}_n) \rangle) && (4.19) \\
&+ \frac{1}{|\mathcal{P}||\mathcal{N}|} \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} \min_{(\boldsymbol{h}'_p, \boldsymbol{h}'_n) \in \mathcal{H}_p \times \mathcal{H}_n} y_{pn} (\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_p, \boldsymbol{h}'_p) \rangle - \langle \boldsymbol{w}, \Phi(\boldsymbol{x}_n, \boldsymbol{h}'_n) \rangle) &&
\end{aligned}$$

With the definition of the latent variable and the joint feature, the maximization (resp. minimization) over the latent variables can be decomposed for each term of the sum. So maximizing (resp. minimizing) the sum is equivalent to maximize (resp. minimize) each term of the sum independently, because the latent variable $\boldsymbol{h}$ can be decomposed for each term of the sum and each couple of latent variables $(\boldsymbol{h}_{p,n}, \boldsymbol{h}_{n,p})$ is independent.

Now, the two sums are grouped in a single sum:

$$\begin{aligned}
&\frac{1}{|\mathcal{P}||\mathcal{N}|}\sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}}\max_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n}\boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle) && (4.20)\\
&+\frac{1}{|\mathcal{P}||\mathcal{N}|}\sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}}\min_{(\boldsymbol{h}'_p,\boldsymbol{h}'_n)\in\mathcal{H}_p\times\mathcal{H}_n}\boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle)\\
=&\frac{1}{|\mathcal{P}||\mathcal{N}|}\sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}}\Big(\max_{(h_p,h_n)\in\mathcal{H}_p\times\mathcal{H}_n}\boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle) && (4.21)\\
&\qquad+\min_{(\boldsymbol{h}'_p,\boldsymbol{h}'_n)\in\mathcal{H}_p\times\mathcal{H}_n}\boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle)\Big)
\end{aligned}$$

We define

$$\begin{aligned}
\Theta(\boldsymbol{x},\boldsymbol{y})=&\frac{1}{|\mathcal{P}||\mathcal{N}|}\sum_{x_i\in\mathcal{P}}\sum_{x_j\in\mathcal{N}}\Big(\max_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n}\boldsymbol{y}_{ij}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle)\\
&+\min_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n}\boldsymbol{y}_{ij}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle)\Big) \qquad (4.22)
\end{aligned}$$

By construction, we have the equality $A(\boldsymbol{x},\boldsymbol{y})=s_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y})$. Now, we show that the latent variables can be fixed independently to the ranking matrix $\boldsymbol{y}$. For a couple of examples $(\boldsymbol{x}_p,\boldsymbol{x}_n)$, with $p\in\mathcal{P},n\in\mathcal{N}$, we analyze the value of the latent variables $\boldsymbol{h}_p,\boldsymbol{h}_n,\boldsymbol{h}'_p,\boldsymbol{h}'_n$ with respect to $\boldsymbol{y}_{pn}$. There are only two cases to analyze: $\boldsymbol{y}_{pn}=1$ and $\boldsymbol{y}_{pn}=-1$.

If $\boldsymbol{y}_{pn}=1$

$$\begin{aligned}
&\max_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle) && (4.23)\\
&\qquad+\min_{(\boldsymbol{h}'_p,\boldsymbol{h}'_n)\in\mathcal{H}_p\times\mathcal{H}_n}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle)\\
&=\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p^+)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n^-)\rangle+\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p^-)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n^+)\rangle && (4.24)\\
&=\langle\boldsymbol{w},\Phi_-^+(\boldsymbol{x}_p)\rangle-\langle\boldsymbol{w},\Phi_-^+(\boldsymbol{x}_n)\rangle && (4.25)\\
&\text{where}\quad\Phi_-^+(\boldsymbol{x}_i)=\Phi(\boldsymbol{x}_i,\boldsymbol{h}_i^+)+\Phi(\boldsymbol{x}_i,\boldsymbol{h}_i^-) && (4.26)\\
&\qquad\boldsymbol{h}_i^+=\arg\max_{\boldsymbol{h}\in\mathcal{H}_i}\langle\boldsymbol{w},\Phi(\boldsymbol{x}_i,\boldsymbol{h})\rangle\qquad\boldsymbol{h}_i^-=\arg\min_{\boldsymbol{h}\in\mathcal{H}_i}\langle\boldsymbol{w},\Phi(\boldsymbol{x}_i,\boldsymbol{h})\rangle && (4.27)
\end{aligned}$$

If $\boldsymbol{y}_{ij}=-1$

$$\begin{aligned}
&\max_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n}-(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle) && (4.28)\\
&\qquad+\min_{(\boldsymbol{h}'_p,\boldsymbol{h}'_n)\in\mathcal{H}_p\times\mathcal{H}_n}-(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle)\\
&=\max_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle) && (4.29)\\
&\qquad+\min_{(\boldsymbol{h}'_p,\boldsymbol{h}'_n)\in\mathcal{H}_p\times\mathcal{H}_n}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle)\\
&=\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n^+)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p^-)\rangle+\langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n^-)\rangle-\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p^+)\rangle && (4.30)\\
&=-(\langle\boldsymbol{w},\Phi_-^+(\boldsymbol{x}_p)\rangle-\langle\boldsymbol{w},\Phi_-^+(\boldsymbol{x}_n)\rangle) && (4.31)
\end{aligned}$$

We notice that the predicted latent variables are the same in the two cases. So the latent variables can be fixed independently to the value of $\boldsymbol{y}_{pn}$.

$$\begin{aligned}&\max_{(\boldsymbol{h}_p,\boldsymbol{h}_n)\in\mathcal{H}_p\times\mathcal{H}_n} \boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}_p)\rangle - \langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}_n)\rangle) &(4.32)\\ &\quad+\min_{(\boldsymbol{h}'_p,\boldsymbol{h}'_n)\in\mathcal{H}_p\times\mathcal{H}_n} \boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi(\boldsymbol{x}_p,\boldsymbol{h}'_p)\rangle - \langle\boldsymbol{w},\Phi(\boldsymbol{x}_n,\boldsymbol{h}'_n)\rangle)\\ &=\boldsymbol{y}_{pn}(\langle\boldsymbol{w},\Phi^{+}_{-}(\boldsymbol{x}_p)\rangle - \langle\boldsymbol{w},\Phi^{+}_{-}(\boldsymbol{x}_n)\rangle) &(4.33)\end{aligned}$$

When the latent variables are fixed, each example $\boldsymbol{x}_i$ can be represented by $\Phi^{+}_{-}(\boldsymbol{x}_i)$, and $\Theta(\boldsymbol{x},\boldsymbol{y})$ can be written as follow:

$$\Theta(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{|\mathcal{P}||\mathcal{N}|}\sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}}\boldsymbol{y}_{ij}(\langle\boldsymbol{w},\Phi^{+}_{-}(\boldsymbol{x}_p)\rangle - \langle\boldsymbol{w},\Phi^{+}_{-}(\boldsymbol{x}_n)\rangle) \tag{4.34}$$

So $s_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y})$ can be written as a supervised feature map, where the latent are fixed independently to the ranking matrix $\boldsymbol{y}$, and each example $\boldsymbol{x}_i$ is represented by $\Phi^{+}_{-}(\boldsymbol{x}_i)$. □

The proof of Lemma 4.1 comes from a symmetrization of the problem due to the max + min operation. The supervised feature map $\Phi^{+}_{-}(\boldsymbol{x}_i)$ is the solution of the optimization over $\boldsymbol{h}$, whatever $\boldsymbol{y}$ value. We now explain how inference and loss-augmented inference can be efficiently solved with MANTRA.

Proposition 4.1. *Inference for the MANTRA ranking instantiation is solved exactly by sorting the examples in descending order of score* $\langle\boldsymbol{w},\Phi^{+}_{-}(\boldsymbol{x}_i)\rangle$

Proof. Since the inference consists in solving $\max_{\boldsymbol{y}}\Theta(\boldsymbol{x},\boldsymbol{y})$, this is a direct consequence of Lemma 4.1: the problem reduces to solving a fully supervised ranking inference problem, where each example $\boldsymbol{x}_i$ is represented by $\Phi^{+}_{-}(\boldsymbol{x}_i)$. This is solved by sorting the example in descending order of score $\langle\boldsymbol{w},\Phi^{+}_{-}(\boldsymbol{x}_i)\rangle$ (Yue et al. 2007). □

Proposition 4.2. *Loss-augmented inference for MANTRA (Equation 4.13), with the instantiation of Eq. (4.15), is equivalent to:*

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}\in\mathcal{Y}}\ [\Delta(\boldsymbol{y}^*,\boldsymbol{y}) + \Theta(\boldsymbol{x},\boldsymbol{y})] \tag{4.35}$$

Proposition 4.2 directly follows from Lemma 4.1. This is a key result, since it allows to use MANTRA with different loss functions, as soon as there is an algorithm to solve the loss-augmented inference in the fully supervised setting. To solve it with Δ_{AP}, we use the greedy algorithm proposed by (Yue et al. 2007), which finds a globally optimal solution.

Time complexity Finally, we analyze the complexity of inference and loss-augmented inference for ranking instantiation. We define $\bar{h}$ as the average number of regions per images. The inference complexity with MANTRA is $O(N\bar{h}d + N\log N)$, where the first term is the complexity to infer exhaustively the latent variables and the second term is the complexity of the sort. With the algorithm proposed by (Yue et al. 2007), the complexity of loss-augmented inference for MANTRA is $O(N\bar{h}d + N\log N + |\mathcal{P}||\mathcal{N}|)$, where the third term is the complexity to rank negative examples.

Scale	90	80	70	60	50	40	30
Number of regions	4	9	16	25	36	49	64

Table 4.1.: Number of regions per image for each scale.

4.4 Experiments

In this section, we present an evaluation and analysis of MANTRA for multi-class classification and ranking tasks. In our implementation, we use MOSEK[1] to solve the Quadratic Program (QP) at each cutting plane iteration (Line 5 of Algorithm Algorithm 4.1 for MANTRA). By default, the regularization parameter C is fixed to a large value, e.g. 10^5. An analysis of this parameter is done in Subsection 4.4.2.

4.4.1 Multi-class Classification

In this section, we analyze our multi-class model (Subsection 4.3.1) on four datasets. We evaluate our multi-class model for four different visual recognition tasks: scene categorization (15 Scene), cluttered indoor scenes (MIT67), fine-grained recognition (PPMI) and complex event and activity images (UIUC-Sports). These datasets are presented in Section 2.4. We now present how the feature regions are extracted.

Features We extract region features for different bounding box scales: from 30% to 90% of the image size, with a step of 10%. The number of regions per image for each scale is given in Table 4.1. Each image region is described using deep features computed with Caffe CNN library (Jia et al. 2014). We use the output of the sixth layer (after the Rectified Linear Unit (ReLU)), so that each region is represented by a 4096-dimensional vector. For UIUC-Sports and PPMI (resp. 15 Scene and MIT67), we use deep features based on a model pre-trained on ImageNet (Russakovsky, Deng, et al. 2015) (resp. Places (B. Zhou et al. 2014)).

4.4.1.1 MANTRA Analysis

In this section, we provide an analysis of our method. We first study the accuracy and training time with respect to the bounding box scale, because the bounding box scale is an important hyper-parameter of our model. Then, we compare MANTRA to the popular LSSVM (C.-N. Yu et al. 2009), and we show visual results.

Mono-scale results We report MANTRA results with respect to the bounding box scale in Figure 4.2. It is worth pointing out that parts learned with a single region by MANTRA are able to improve performances over deep features computed on the whole image ($s = 100\%$), *e.g.* 5 pt for PPMI. It confirms that using regions allows to find more discriminant representations. We observe that the performances on small scales remain very good: for example, results for scale $s = 40\%$ are as good as for $s = 100\%$ in PPMI

[1] www.mosek.com

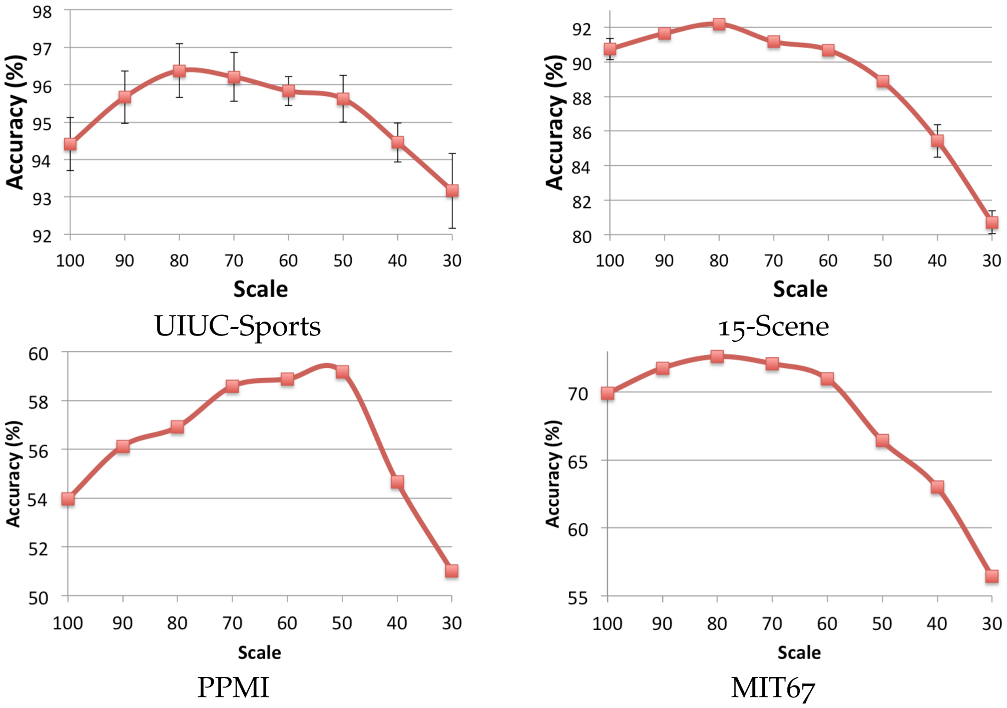
and UIUC; although performances slightly decrease for 15-Scene and MIT67, they remain very competitive (see Table 4.3).

Time analysis The Figure 4.3 shows the training time required on 1 CPU (2.7 Ghz, 32 Go RAM) to train models on UIUC-Sports, 15 Scene, PPMI and MIT67. Training MANTRA is fast: for example, it takes 1 minute at scale 30% of UIUC, where the training set is composed of 540 images and about 36 000 regions. The training time increases linearly with respect to the number of regions per image. It is the expected behavior, because the most time consuming step of Algorithm 4.1 is the loss-augmented inference, which is proportional to the size of the latent space when it is solved exhaustively. This confirms that the proposed 1-slack cutting plane strategy to solve the optimization problem (Subsection 4.2.3) is efficient.

Comparison to LSSVM As previously mentioned, most of state-of-the-art WSL works are based on Deformable Part Model (DPM) (Felzenszwalb et al. 2010) or LSSVM (C.-N. Yu et al. 2009). To highlight model differences between MANTRA and LSSVM, we carry out experiments with the same (deep) features, and evaluate performances on the same splits. For small scales, the choice of a proper region for classification is crucial. In Table 4.2, we report classification performances for both methods at scale 30%. Results clearly show the superiority of our model: MANTRA outperforms LSSVM by a very large margin, *e.g.* ~ 30 pt increase on PPMI and MIT67. In Table 4.2, we also report the training time. MANTRA training is much faster than LSSVM's: for example, MANTRA is 30 times faster for UIUC Sports. The significant speedup for training MANTRA can be explained by the fact that LSSVM uses CCCP (Yuille et al. 2003) to solve the non-convex optimization problem.

To further analyze the performance gain of MANTRA vs LSSVM, we isolate in Table 4.2 the impact of the new prediction function (Section 4.2) by training MANTRA with CCCP (MANTRA-C), and the Non-Convex Cutting-Plane (NCCP) optimization (Subsection 4.2.3), by training LSSVM with NCCP (LSSVM-N). CCCP leads to slightly better results for LSSVM, because the decomposition proposed by (C.-N. Yu et al. 2009) exploits the structure of the optimization problem. In contrast, MANTRA objective (Equation 4.9) does not directly rewrites as a Difference of Convex functions (DC). By using the generic DC decomposition detailed in Subsection 2.2.2.2, we can used CCCP for MANTRA: results in Table 4.2 show that both optimizations give similar performances, because the decomposition is not driven by the structure of the problem, while MANTRA CCCP being significantly slower. The conclusion of this study is that the superiority of MANTRA vs LSSVM is due to the new prediction function.

Visual results In Figure 4.4 (resp. Figure 4.5, Figure 4.6), we show visual results on UIUC-Sports (resp. MIT67 and 15 Scene) dataset. We show prediction maps and $(\boldsymbol{h}^+, \boldsymbol{h}^-)$ regions, for different classifiers. For instance, when classifying a snowboard image (Figure 4.4, last row), the *snowboard* classifier learns that the most discriminative region is the region with the snowboard and the feet of the person. It also learns that all the other regions are relevant and assign a high score to all regions, including the $\boldsymbol{h}^-$ region. The *bocce* classifier learns that the head of the person is quite discriminative, but it also learns that bocce is not practiced on the snow, and assign a very low score to the $\boldsymbol{h}^-$



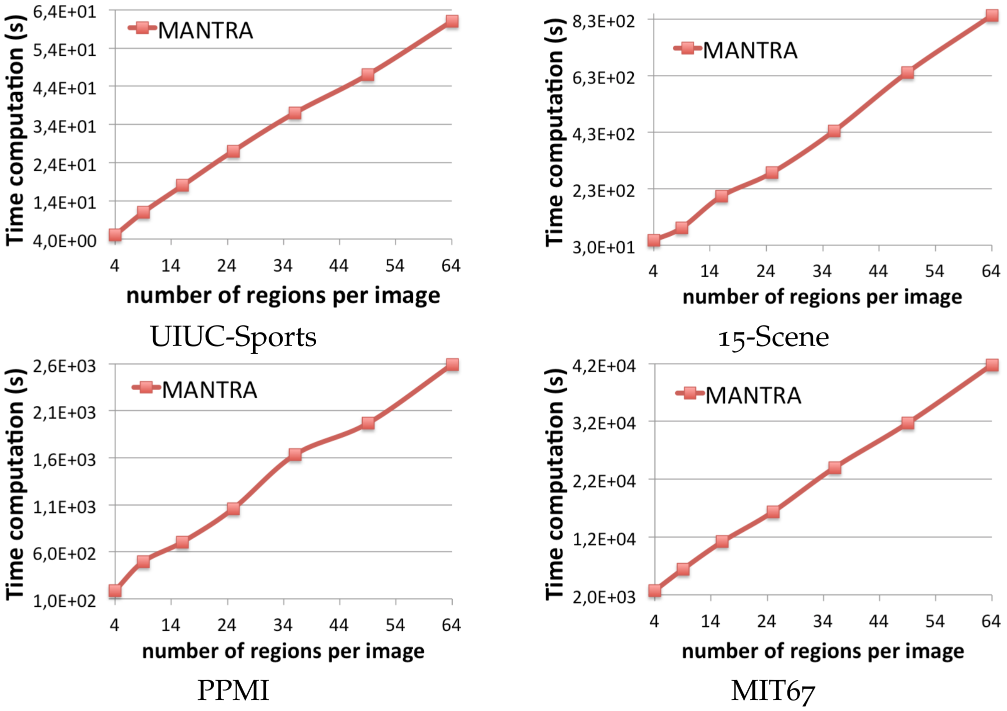
UIUC-Sports

15-Scene

PPMI

MIT67

Figure 4.2.: Multi-class accuracy (%) with respect to the scale.



UIUC-Sports

15-Scene

PPMI

MIT67

Figure 4.3.: MANTRA training time (seconds) w.r.t. the number of regions per image.

Method	UIUC	15 Scene	PPMI	MIT67
Multi-class accuracy (%)				
LSSVM	73.3 ± 0.3	65 ± 1.5	13.3	26.6
MANTRA	**93.2 ± 1.0**	**80.7 ± 0.7**	**51.0**	**56.4**
LSSVM-N	71.6 ± 1.3	64.3 ± 0.9	13.6	25.2
MANTRA-C	**93.2 ± 0.9**	80.4 ± 0.6	50.9	**56.5**
Average training time (seconds)				
LSSVM	1863	14179	21327	156360
MANTRA	**61**	**843**	**2593**	**41805**

Table 4.2.: Performances comparison and training time between MANTRA and LSSVM for scale 30%. MANTRA-C is MANTRA with CCCP, and LSSVM-N is LSSVM with NCCP.

region, which focuses on the snow. Similarly, the $\boldsymbol{h}^-$ region of *croquet* (resp. *polo*) classifier focuses on the snow, which supports the absence of *croquet* (resp. *polo*) class. For the correct class, the incorporation of $\boldsymbol{h}^-$ prevents from having large negative values for any (random) window. The fourth row of Figure 4.4 shows the prediction of *sailing* and *rowing* categories for a *sailing* image. For each classifier, $\boldsymbol{h}^+$ corresponds to discriminative parts, *i.e.* boat with sail and water. The $\boldsymbol{h}^-$ region for the *rowing* classifier focuses on the sail of the boat with a very low score. It suggests that if a sail is found, the image is very unlikely to belong to the class *rowing*. Another example is the *greenhouse* image of MIT67 (second row of Figure 4.5), where both *greenhouse* and *florist* classifiers have high scores and focus on plants. For the *greenhouse* classifier, $\boldsymbol{h}^-$ has a quite high score, so all regions are discriminative for it. This is in stark contrast to the *florist* classifier, for which $\boldsymbol{h}^-$ focuses on the roof of the greenhouse, and has a very low score because this structure with walls and roof made of transparent material is never present in florist shop. The roof of the greenhouse is a clear evidence of the absence of the *florist* category. For classifiers of uncorrelated categories like *laboratorywet*, all regions have very low scores because the roof structure and the plant are never present in laboratory wet. In Figure 4.4-4.6, we can point out other examples, where wrong classifiers can have large scores on local regions, which are, however, compensated by very strong evidence of the absence of the class: *croquet* vs *bocce* in Figure 4.4 (UIUC-Sports), *closet* vs *clothing store* and *book store* vs *library* in Figure 4.5 (MIT67), *street* vs *inside city* or *highway* and *tall building* vs *inside city* in Figure 4.6 (15 Scene). For example, the second row of Figure 4.6 shows results for a *street* image. The region $\boldsymbol{h}^+$ predicted by classifier *highway* has a high score because this region focuses on the road and is similar to highway. But the $\boldsymbol{h}^-$ region focuses on the front of the buildings with a low score, because there is not highway closed to buildings. On the contrary, the classifier *inside city* have high score on the front of the buildings, but low score on the road, because most of the *inside city* images show the front of the buildings without road. Only the *street* classifier has quite high score for the $\boldsymbol{h}^-$ region.

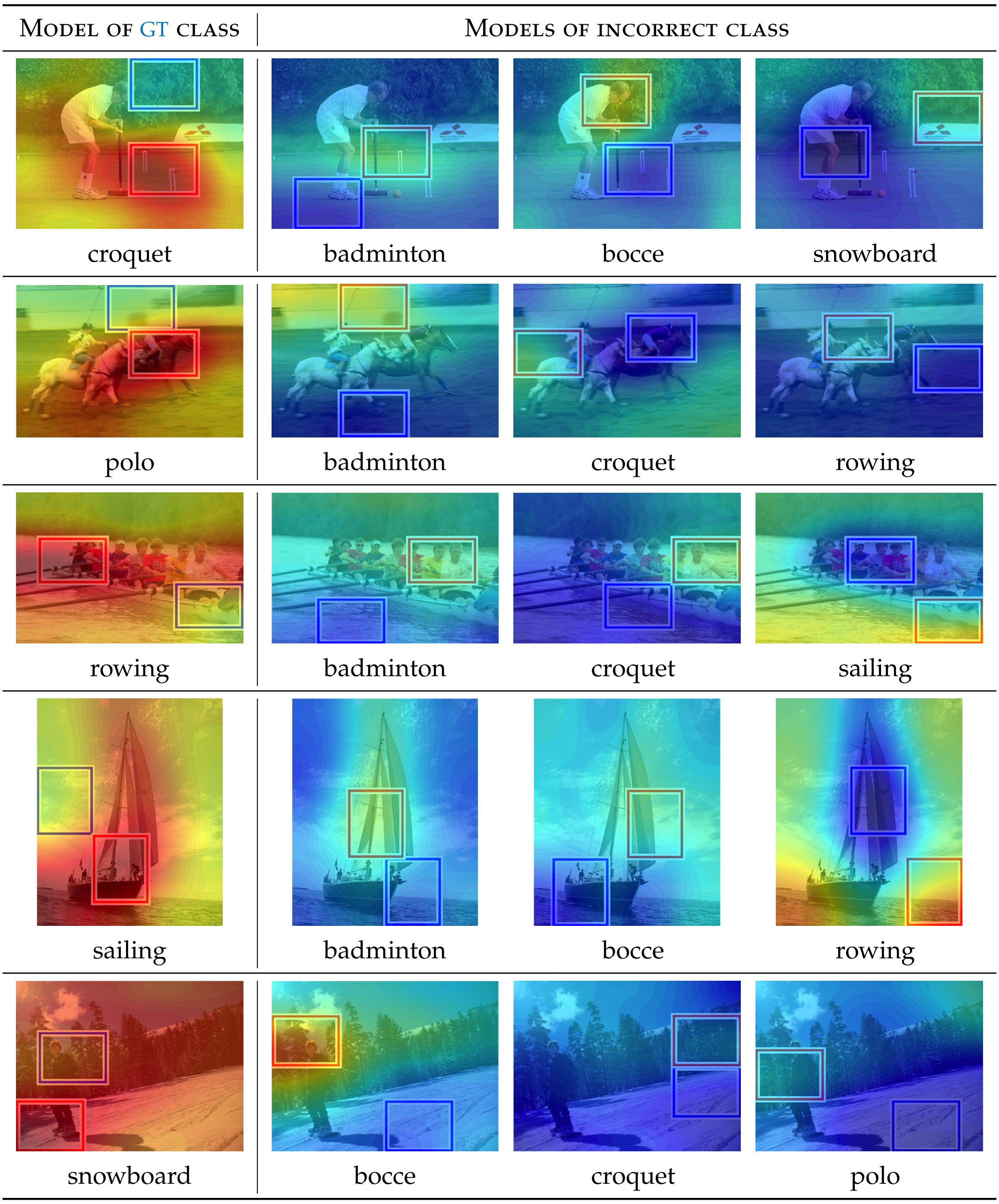


Figure 4.4.: Example of response map for UIUC-Sports images and for model of the correct class (left column) and models of incorrect class. For each model, the red (resp. blue) bounding box show the region with the maximum (resp. minimum) score.

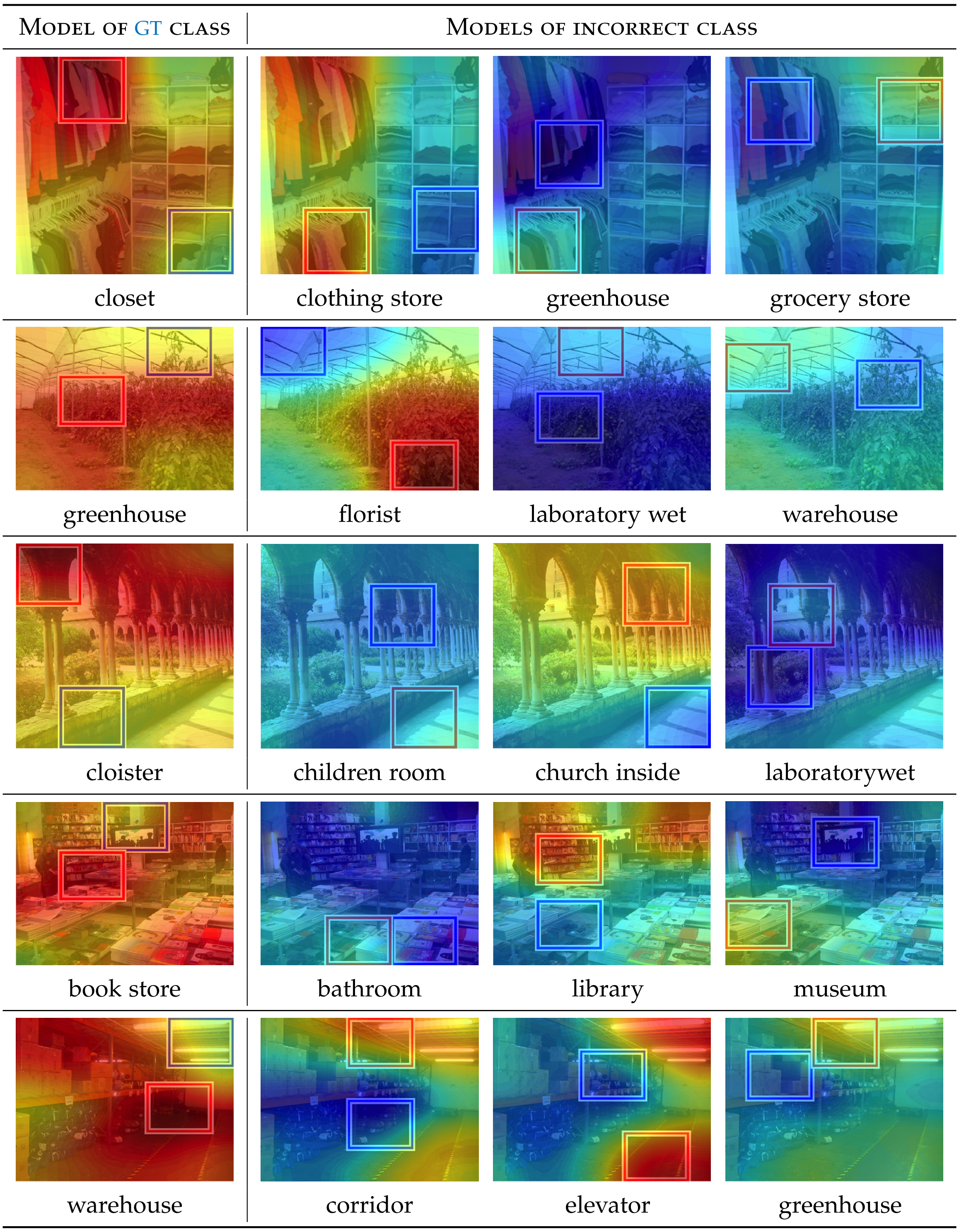


Figure 4.5.: Example of response map for MIT67 images and for model of the correct class (left column) and models of incorrect class. For each model, the red (resp. blue) bounding box show the region with the maximum (resp. minimum) score.

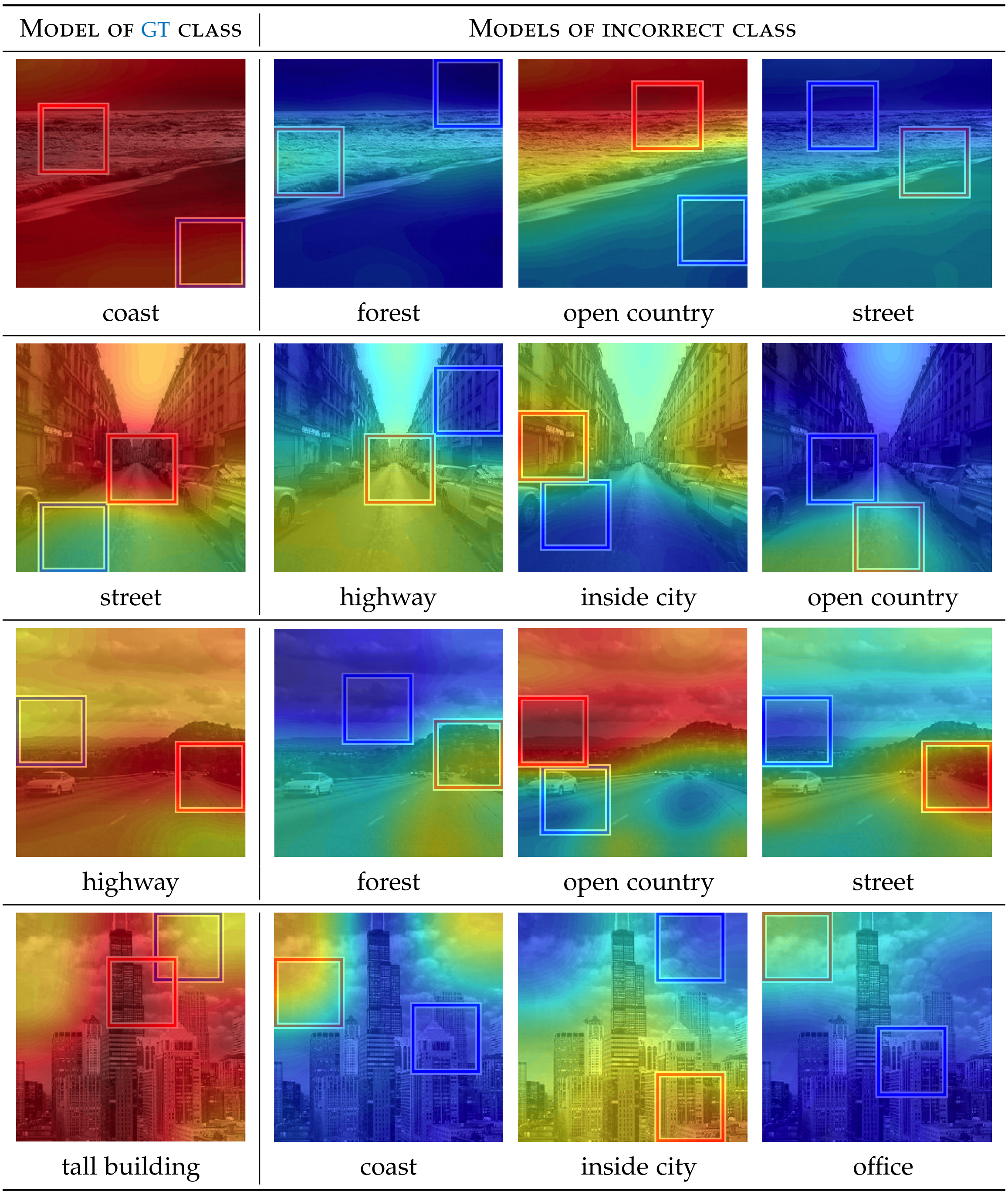


Figure 4.6.: Example of response map for 15 Scene images and for model of the correct class (left column) and models of incorrect class. For each model, the red (resp. blue) bounding box show the region with the maximum (resp. minimum) score.

Method	PPMI	UIUC	15Scene	MIT67
Deep features				
CaffeNet ImageNet (Jia et al. 2014)	54.5†	94	88	58.5
CaffeNet Places (B. Zhou et al. 2014)	38.6†	94.1	90.2	68.2
MOP-CNN (Gong et al. 2014)	-	-	-	68.9
Part-based				
SPM (Lazebnik et al. 2006)	39.1	71.6	81.4	34.4
Object Bank (L.-J. Li, Su, et al. 2014)	-	77.9	80.9	37.6
RBoW (Parizi, Oberlin, et al. 2012)	-	-	78.6	37.9
DSS (G. Sharma et al. 2012)	49.4	-	85.5	-
LPR (Sadeghi et al. 2012)	-	86.3	85.8	44.8
BoP+IFV (Juneja et al. 2013)	-	-	-	63.1
MLrep+IFV (Doersch et al. 2013)	-	-	-	66.9
MMDL (Xinggang Wang et al. 2013)	-	88.5	86.4	50.2
Hierarchical model (Lobel et al. 2013)	-	-	84.6	39.5
Discriminative parts (J. Sun et al. 2013)	-	86.4	86.0	51.4
RFDC (Bossard et al. 2014)	-	-	-	54.4
DSFL (Zuo et al. 2014)	-	86.5	84.2	52.2
MANTRA	**66.2**	**97.3**	**93.4**	**76.6**

Table 4.3.: MANTRA results and comparison to state-of-the art works († is our re-implementation).

4.4.1.2 Comparison with State-of-the-Art Methods

We now compare our model with state-of-the-art models. The mono-scale results (Subsection 4.4.1.1) suggest the idea of combining several scales, which are expected to convey complementary informations. To perform scale combination, we use an Object Bank (OB) (L.-J. Li, Su, et al. 2014) strategy, which is often used in WSL works (Sadeghi et al. 2012; Juneja et al. 2013; J. Sun et al. 2013). We aggregate the score of each class (i.e. sum of $\boldsymbol{h}^+$ and $\boldsymbol{h}^-$ regions) and each scale in a vector. Contrary to the original OB, we do not use SPM. Our final representation is thus compact: $S \times K$ where S is the number of scales, and K is the number of classes. Ultimately, we use a linear SVM classifier for classification, because it enables to automatically learned the importance of each scale, and the correlations between the classes.

Results for our multi-scale method are shown in Table 4.3. We first notice that performances improve compared to the best mono-scale results (Figure 4.2: 4 pt for MIT67, 7 pt for PPMI), validating the fact that taking into account different scales enable catching complementary and discriminative localized information. Then, we compare MANTRA to state-of-the-art works. We note that the improvement over part-based models, which use weaker features and essentially based on LSSVM (C.-N. Yu et al. 2009), e.g. HOG, is huge. We also provide comparisons to recent methods based on deep features: we report performances with models pre-trained on ImageNet, but also using Places, a large-scale scene dataset recently introduced in (B. Zhou et al. 2014). As we can verify, Places is better-suited for scene recognition (performance boost in 15-Scene and MIT67), whereas ImageNet has an edge over Places for object classification (PPMI). For UIUC, both models present similar performances. In Table 4.3, we can see that MANTRA can further improve performances over the best deep features (ImageNet or Places) by a large margin on the 4 databases, e.g. 8.5 pt on the challenging MIT67 dataset, or 11 pt on PPMI. As mentioned earlier, internal representations learned by ConvNets present limited invariance power: learning strong invariance is therefore challenging (N. Zhang et al. 2014). We show here that the proposed WSL scheme is able to efficiently learn strong invariance by aligning image regions, increasing performances when built upon strong deep features. MANTRA also significantly outperforms MOP-CNN (Gong et al. 2014) which uses VLAD pooling with deep features extracted at different scales. This shows the capacity of our model to seek discriminative part regions, whereas background and non-informative parts are incorporated into image representation in (Gong et al. 2014).

4.4.2 Ranking

We evaluate our ranking model (Subsection 4.3.2) for 2 different applications: action classification (PASCAL VOC 2011), and object recognition (PASCAL VOC 2007). The performances on the two datasets are evaluated with a ranking measure: MAP.

4.4.2.1 Action Classification

We use standard (∼2400-dim) poselets as region features (Behl et al. 2015), and there are about 20 regions per image. We compare WSL models optimizing accuracy, i.e. LSSVM-Acc and MANTRA-Acc, and models explicitly optimizing AP, i.e. MANTRA-AP and LAPSVM (Behl et al. 2015). Since the dataset contains bounding box annotations, we also evaluate detection performances to analyze the selected regions. We do not use the standard detection metric used in PASCAL VOC challenges, because it requires a good prediction of the object extend, which is not always possible with the poselets features of (Behl et al. 2015). We evaluate detection performances by measuring the averaged overlap between predicted and ground truth bounding box, which gives a coarse evaluation of the localization performances. Experiments are carried out on the *trainval* set in a weakly supervised setup, *i.e.* without bounding box for training and testing, for 5 random splits (with 80% for training, 20% for testing).

Results As shown in Table 4.4, MANTRA-Acc outperforms LSSVM-Acc by ∼ 6 pt, again validating the relevance of the new model introduced in this chapter. MANTRA-Acc also performs similarly to LAPSVM, which is, to the best of our knowledge, the only method that optimizes an AP-based loss function over weakly supervised data. MANTRA-AP can further improve performances over MANTRA-Acc by 7 pt, which confirms the relevance of optimizing AP during training. We can also see that detection results are strongly correlated to ranking performances: MANTRA-AP also outperforms LAPSVM in terms of detection metric. Detection performances also give a quantitative validation that MANTRA is able to localize semantic parts, here people performing the action. We can interpret the use of the max + min operation as a regularizer of the latent space, which exploits the capacity of $\boldsymbol{h}^-$ to witness the absence of a class to find more semantic part predictions $\boldsymbol{h}^+$.

Method	Ranking MAP (%)	Detection overlap (%)
LSSVM-Acc	29.5 ± 1.3	12.7 ± 0.3
MANTRA-Acc	35.2 ± 1.2	18.9 ± 0.9
LAPSVM	36.7 ± 0.8	20.1 ± 0.7
MANTRA-AP	$\mathbf{42.2} \pm \mathbf{1.3}$	$\mathbf{26.5} \pm \mathbf{1.4}$

Table 4.4.: Ranking and detection results on VOC 2011 Action.

Note that our protocol differs from (Behl et al. 2015), which evaluates on the test set and uses bounding box annotations. When using the same protocol as in (Behl et al. 2015), LAPSVM reaches 44.3% *vs* 47.1% for MANTRA-AP. Note that with this protocol, the prediction function used in test is the same for both models.

Impact of hyper-parameter C We show in Figure 4.7 performance variations vs the regularization parameter C. We observe that all methods reach optimal scores for large values: cross-validation on the train set always leads to $C = 10^4$ or 10^5 optimal values. We note a large drop of performances when C is small. We also note that model optimizing AP during training are more robust to C parameter.

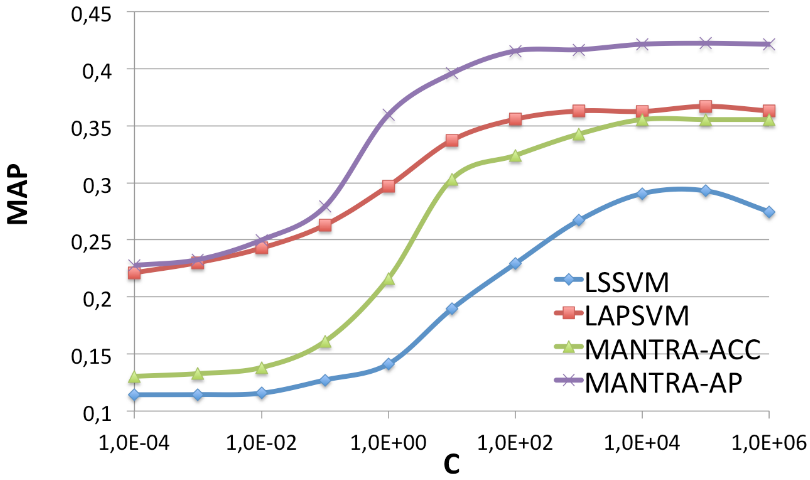


Figure 4.7.: Analysis of hyper-parameter C on ranking performances.

4.4.2.2 Object Recognition

Finally, we perform experiments on the PASCAL VOC 2007. We extract deep features pre-trained on ImageNet using MatConvNet library (Chatfield et al. 2014). As in (Chatfield et al. 2014), we take the output of the seventh layer of vgg-m-2048, after the ReLU. As done for the multi-class classification (Subsection 4.4.1.2), we extract deep features at different scales, and combine them with OB (L.-J. Li, Su, et al. 2014) to have a multi-scale model. We compare our model instantiated for multi-class classification (MANTRA-Acc) and ranking (MANTRA-AP) to state-of-the-art results.

Results The performance obtained with deep features computed on the whole image is 77%, which is conform to what is reported in (Chatfield et al. 2014). As shown in Table 4.5, MANTRA-Acc based on these features can improve performances by more than 5 pt, reaching 82.6%. MANTRA-AP further significantly improves performances by more than 3 pt, reaching 85.8%, again supporting the relevance of optimizing AP during training. Compared to recent methods, our model also outperforms (Oquab et al. 2014) and SPP-net (He et al. 2014), which used a spatial pyramid pooling layer. In (Chatfield et al. 2014), a fine-tuning with a ranking-based objective function is used. MANTRA-AP outperforms this method by more than 3 pt, without fine-tuning and data augmentation.

Method	MAP (%)
Deep transfer (Oquab et al. 2014)	77.7
SPP-net (He et al. 2014)	80.1
`vgg-m-2048` (Chatfield et al. 2014)	82.4
MANTRA-Acc	82.6
MANTRA-AP	**85.8**

Table 4.5.: Ranking performances on PASCAL VOC 2007.

4.5 Conclusion

This chapter introduced a new latent structured output model, named MANTRA, dedicated to WSL of discriminative regions from images annotated with global labels. The MANTRA prediction function is based on two latent variables $(\boldsymbol{h}^+, \boldsymbol{h}^-)$. The intuition behind $\boldsymbol{h}^-$ is as follows: for an incorrect output, it seeks negative evidence against it. For a correct output, it prevents from having large negative values for any region, thus $\boldsymbol{h}^-$ acts as a latent space regularizer exploiting contextual information. We proposed an efficient cutting plane algorithm to train the model. We instantiated MANTRA for two different visual recognition tasks: multi-class classification and AP ranking. Another important contribution is the MANTRA AP ranking instantiation, for which efficient solutions are introduced to solve the challenging (loss-augmented) inference problem.

We evaluated MANTRA on the image classification task, where each region is represented by a deep feature. The experiments, carried out on six different datasets, validated the relevance of the proposed approach: MANTRA has good performances and training is fast. We showed that MANTRA outperforms deep features extracted on whole images, and WSL models using the maximum regions. In particular, on datasets evaluated with AP, we note a large performance improvement when the AP loss is optimized during training.

Unfortunately, extracting deep features for each region of an image is time consuming and inefficient. To address this problem, we propose in the next chapter to include MANTRA model in a deep ConvNet architecture.

Chapter 5

WELDON: NEGATIVE EVIDENCE FOR WEAKLY SUPERVISED LEARNING OF DEEP STRUCTURED MODELS

Contents

Chapter abstract

This chapter explains how to integrate MANTRA (Chapter 4) in a deep ConvNet architecture. The core of the approach is a new pooling function which generalizes the MANTRA pooling function by selecting several instances. We introduce a specific architecture design which enables an efficient processing by sharing region feature computations, as well as an easy and effective transfer learning and fine-tuning. We show that our model can be trained end-to-end for different visual recognition tasks: multi-class and multi-label classification, and also structured Average Precision (AP) ranking. Finally, we evaluate classification performances of our model on six challenging datasets.

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

- Thibaut Durand, Nicolas Thome, and Matthieu Cord (2016). "WELDON: Weakly Supervised Learning of Deep Convolutional Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

5.1 Introduction

To address the limited spatial invariance of ConvNets, we proposed in Chapter 4 a new WSL model (MANTRA) to learn discriminative regions from images annotated with global labels, where each region is represented by a deep feature extracted with a pre-trained model. Unfortunately, this strategy is highly inefficient since feature computations in (close) neighbor regions are not shared. To overcome this problem, we propose WEakly supervised Learning of Deep cOnvolutional neural Network (WELDON).

We design a fully convolutional network architecture which enables fast region feature computation by convolutional sharing (Section 5.3). As explained in Section 2.3, the key issue to WSL of deep ConvNets is to find how to pool the regions to get one score per map. The output of the ConvNet is a detection map for each category, so to train it with standard classification loss – since only image-level labels are available –, it is necessary to aggregate the maps into a global prediction for each class. The most popular approach is the `max` pooling, which selects the best region to perform prediction. Similarly to MANTRA, we propose in Subsection 5.3.2 a new pooling strategy based on negative evidence to automatically learn localized features. This pooling function generalizes MANTRA pooling function by selecting several maximum and minimum regions to be more robust. Contrary to the MANTRA model, the WELDON model can be trained end-to-end, which enables to fine-tune the whole network on the target dataset. Finally, we evaluate our model on six challenging datasets (Section 5.5), and analyze the impact of each improvement.

5.2 Generalized Negative Evidence Model

We introduce a new prediction function, which generalizes the MANTRA prediction function (Equation 4.3). We define a scoring function $F_{\boldsymbol{w}} : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}$, with depends on the input data $\boldsymbol{x} \in \mathcal{X}$, the output $\boldsymbol{y} \in \mathcal{Y}$, the latent variable $\boldsymbol{h} \in \mathcal{H}$ and some parameters $\boldsymbol{w} \in \mathbb{R}^d$. Our goal is to learn a prediction function $f_{\boldsymbol{w}}$, parametrized by $\boldsymbol{w}$, so predicted output $\hat{\boldsymbol{y}}$ depends on $F_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$.

As mentioned earlier, the main intuition of our negative evidence model is to equip each possible output $\boldsymbol{y} \in \mathcal{Y}$ with a pair of latent variables $(\boldsymbol{h}^+_{i,\boldsymbol{y}}, \boldsymbol{h}^-_{i,\boldsymbol{y}})$. $\boldsymbol{h}^+_{i,\boldsymbol{y}}$ (resp. $\boldsymbol{h}^-_{i,\boldsymbol{y}}$) corresponding to the maximum (resp. minimum) scoring latent value, for input $\boldsymbol{x}_i$ and output $\boldsymbol{y}$:

$$\boldsymbol{h}^+_{i,\boldsymbol{y}} = \arg\max_{\boldsymbol{h} \in \mathcal{H}} F_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}) \qquad \boldsymbol{h}^-_{i,\boldsymbol{y}} = \arg\min_{\boldsymbol{h} \in \mathcal{H}} F_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}) \tag{5.1}$$

For an input/output pair $(\boldsymbol{x}_i, \boldsymbol{y})$, the scoring of the model, $s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$, sums $\boldsymbol{h}^+_{i,\boldsymbol{y}}$ and $\boldsymbol{h}^-_{i,\boldsymbol{y}}$ scores, as follows:

$$s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) = F_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}^+_{i,\boldsymbol{y}}) + F_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h}^-_{i,\boldsymbol{y}}) \tag{5.2}$$

Finally, our prediction is:

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{w}}(\boldsymbol{x}_i) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) \tag{5.3}$$

This maximization in Equation 5.3 is known as the inference problem. Regarding the soring function in Equation 5.2, we are here considering deep ConvNets models for $f_{\mathbf{w}}$. This generalizes the MANTRA model introduced in Chapter 4, using a log-linear scoring function: $f_{\mathbf{w}}(\mathbf{x}, \mathbf{y}, \mathbf{h}) = \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}, \mathbf{h}) \rangle$ where $\Psi(\mathbf{x}, \mathbf{y}, \mathbf{h})$ is a joint feature map that describes the relation between input $\mathbf{x}$, output $\mathbf{y}$, and latent variable $\mathbf{h}$.

5.3 WELDON Network Architecture

Based on the model presented in previous section, we propose WELDON, a new weakly supervised learning dedicated to learn localized visual features by using only image-level labels during training. The proposed network architecture is decomposed into two sub-networks: a deep feature extraction network based on VGG16 and a prediction network, as illustrated in Figure 5.1. The feature extraction net purpose is to extract a fixed-size deep descriptor for each region in the image, while the prediction network outputs a structured output.

Notation We note $F^{l}_{\mathbf{w}}(\mathbf{x}, \mathbf{y}, \mathbf{h})$ the output of the layer l at the location $\mathbf{h}$ of the feature map (or category) $\mathbf{y}$ for the input image $\mathbf{x}$. $\mathbf{w}$ are the parameters of the ConvNet.

5.3.1 Feature Extraction Network

The feature extraction network is dedicated to computing a fixed-size representation for any region of the input image. When using ConvNets as feature extractors, the most naive option is to process input regions independently, i.e. to resize each region to match the size of a full image for ConvNet architectures trained on large scale databases such as ImageNet (e.g. 224 × 224). This is the approach followed in R-CNN (R. Girshick et al. 2014), or in MANTRA (Chapter 4). This is, however, highly inefficient since feature computation in (close) neighbor regions is not shared. Recent improvements in SPP-net (He et al. 2014) or fast R-CNN (Ross Girshick 2015) process images of any size by using only convolutional/pooling layers of ConvNets trained on ImageNet, subsequently applying max pooling to map each region into a fixed-size vector. (Long et al. 2015) introduces the FCN.

The last fully connected layers of standard networks (AlexNet (Krizhevsky et al. 2012), VGG16 (Simonyan et al. 2015)) do not allow to have feature map with spatial resolution. The fully connected layers of these nets have fixed dimensions and throw away spatial coordinates. However, fully connected layers can also be viewed as convolutions with kernels that cover their entire input regions. Doing so casts these nets into fully convolutional networks that take input of any size and make spatial output maps. Convolutionalize fully connected layer allows to transfer `fc6` representations, which are more discriminative features than `conv5` representations. It also enables a faster computation, because the intermediate features over overlapping image regions are shared.

Our feature extraction network is based on VGG16 (Simonyan et al. 2015). The standard input image spatial size is 224 × 224, and with this input, the output spatial size of the feature map after `conv5` is 7 × 7. We transform the `fc6` as a convolution layer of 4096

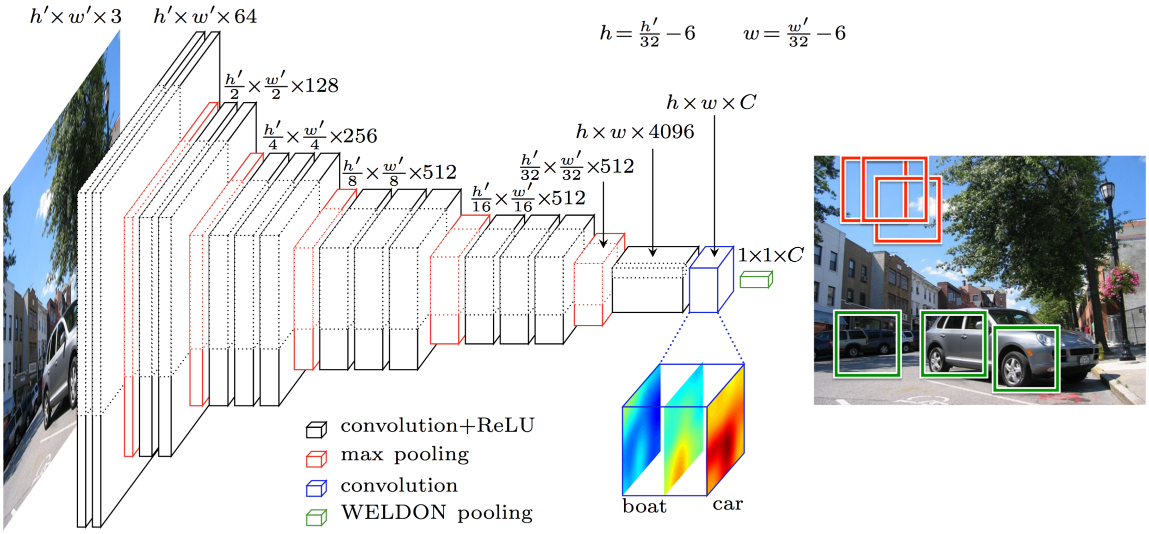

$h' \times w' \times 3$
$h' \times w' \times 64$
$h = \frac{h'}{32} - 6$
$w = \frac{w'}{32} - 6$
$\frac{h'}{2} \times \frac{w'}{2} \times 128$
$h \times w \times C$
$h \times w \times 4096$
$\frac{h'}{4} \times \frac{w'}{4} \times 256$
$\frac{h'}{32} \times \frac{w'}{32} \times 512$
$\frac{h'}{8} \times \frac{w'}{8} \times 512$
$\frac{h'}{16} \times \frac{w'}{16} \times 512$
$1 \times 1 \times C$
convolution+ReLU
max pooling
convolution
WELDON pooling
boat
car

Figure 5.1.: WELDON architecture is decomposed into two sub-networks: a feature extraction network (left) and a prediction network (right). The feature extraction network is based on VGG16 to extract local features from whole images with good spatial resolution. Then a transfer layer is used to learn class-specific heatmaps (*car*, *boat*), and finally a prediction layer aggregates the heatmaps to produce a single score for each class. The dot line shows the feature map size for the standard image size 224 × 224.

filters $512 \times 7 \times 7$ (called conv6). For an input image size 224×224, the feature map size after conv6 is $4096 \times 1 \times 1$. For a larger input image $h' \times w'$, the the feature map size after conv6 is $4096 \times h \times w$, where $h = \frac{h'}{32} - 6$ and $w = \frac{w'}{32} - 6$. We show in Figure 5.1, the feature maps for an arbitrary input image size, and in dot line, we show the feature maps for the standard image size 224×224. In the proposed architecture, input images at a given scale are rescaled to a constant size $I \times I$, with $I \geq 224$. For all I, we consider regions of size 224×224 pixels, so that the region scale is $\alpha = 224/I$ (see details in Table 5.1). Input images are processed with the fully convolutional/pooling layers of ConvNets trained on ImageNet, leading to conv6 outputs of different sizes.

5.3.2 Prediction Network Design

We now present how to select the relevant regions to predict the global image label.

5.3.2.1 Transfer layer

The first layer of the prediction network is a transfer layer. Its goal is to transfer weights of the feature extraction network from large scale datasets to new target datasets. It transforms the output of the feature extraction network F^{fe} into a feature map F^t of size $h \times w \times C$, where C is the number of categories (see Figure 5.1). This layer is convolutional layer, composed of C filters, each of size $1 \times 1 \times 4096$. Due to the kernel size of the convolution, this layer preserves the spatial resolution of the feature maps.

Image size I	Region scale α (%)	conv5 output size	conv6 output size
224×224	100	7×7	1×1
249×249	90	8×8	2×2
280×280	80	9×9	3×3
320×320	70	10×10	4×4
374×374	60	12×12	6×6
448×448	50	14×14	8×8
560×560	40	18×18	12×12
747×747	30	24×24	18×18

Table 5.1.: Proposed multi-scale ConvNet feature extraction networks. Input images are rescale to $I \times I$ images, with I in the range $[224, 747]$. At each scale, regions span 224×224 areas, so that the region scale is $\alpha = 224/I$.

The output of this layer can be seen as localization heatmaps. In Figure 5.1, we show the predicted regions for category *car*.

5.3.2.2 Weakly Supervised Prediction layer

The second layer is a spatial pooling layer s, which aggregates the score maps into classification scores: for each output $\boldsymbol{y} \in \{1, \dots, C\}$, the score over the $h \times w$ regions are aggregated into a single scalar value. We note $F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, \boldsymbol{h})$ is the score of region $\boldsymbol{h}$ from image $\boldsymbol{x}_i$ for category $\boldsymbol{y}$, and $\mathcal{H} = \{1, \dots, r_i\}$ the region index set, and r_i is the number of regions for image $\boldsymbol{x}_i$ after the transfer layer. The output s of the prediction layer is a vector $1 \times 1 \times C$. As mentioned earlier, the standard approach for WSL inherited from MIL is to select the max scoring region. The output score is the score of the region with the maximum score. In Subsection 4.2.1, we improve max pooling by incorporating negative evidence, and we show that negative evidence is important to have accurate predictions. We propose here to improve this strategy by using multiple regions.

WELDON Pooling Based on recent MIL insights on learning with top instances (Subsection 2.2.1.2), we propose to extend MANTRA pooling by using multiple regions instead of single region. Using multiple regions allows to detect several objects or to identify the extent of an object. At the same time, (Bolei Zhou, Khosla, et al. 2016) proposes the Global Average Pooling (GAP) to identify all discriminative regions of an object. But, GAP pools over all regions, even non-relevant regions, while WELDON pools over a subset of selected regions. Formally, let $h_z \in \{0, 1\}$ be the binary variable denoting the selection of the z^{th} region from layer F^t. We propose the scoring function s^{top}, which selects the k^+ highest scoring regions as follows:

$$s^{top}_{\boldsymbol{w},k^+}(F^t(\boldsymbol{x}_i, \boldsymbol{y})) = \frac{1}{k^+} \sum_{z=1}^{r_i} h^+_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, z) \tag{5.4}$$

$$\text{where } \boldsymbol{h}^+ = \underset{\boldsymbol{h} \in \{0,1\}^{r_i}}{\arg\max} \sum_{z=1}^{r_i} h_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, z) \quad \text{s.t.} \quad \sum_{z=1}^{r_i} h_z = k^+$$

where $F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$ is the feature map for class $\boldsymbol{y}$, and $F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, z)$ is the value of the z^{th} region score of feature map $F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y})$. Beyond the relaxation of the Negative Instances in Negative Bags (NINB) assumption, which is sometimes inappropriate (see Subsection 2.2.1.2), the intuition behind s^{top} is to provide a more robust region selection strategy. Indeed, using a single area for training the model necessarily increases the risk of selecting outliers.

To incorporate negative evidence in our prediction function, we propose the scoring function s^{low}, which selects the k^- lowest scoring regions as follows:

$$s^{low}_{\boldsymbol{w},k^-}(F^t(\boldsymbol{x}_i, \boldsymbol{y})) = \frac{1}{k^-} \sum_{z=1}^{r_i} h^-_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, z) \tag{5.5}$$
$$\text{where } \boldsymbol{h}^- = \operatorname*{arg\,min}_{\boldsymbol{h} \in \{0,1\}^{r_i}} \sum_{z=1}^{r_i} h_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}, z) \;\; \text{s.t.} \;\; \sum_{z=1}^{r_i} h_z = k^-$$

The final prediction simply consists in summing s^{top} and s^{low}:

$$s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}) = s^{top}_{\boldsymbol{w},k^+}(F^t(\boldsymbol{x}_i, \boldsymbol{y})) + s^{low}_{\boldsymbol{w},k^-}(F^t(\boldsymbol{x}_i, \boldsymbol{y})) \tag{5.6}$$

This prediction function is equivalent to MANTRA prediction function (Equation 4.3) whenever $k^+ = k^- = 1$.

5.4 Learning & Instantiations

As shown in Figure 5.1, the WELDON model outputs $s \in \mathbb{R}^C$. This vector represents a structured output, which can be used in a multi-class or multi-label classification framework, but also in a ranking problem formulation.

5.4.1 Training Formulation

In this section, we consider three different structured prediction tasks, and their associated loss functions during training.

5.4.1.1 Classification

Multi-class classification In this simple case, C is the number of classes and the output space is $\mathcal{Y} = \{1, \ldots, C\}$. We use the usual softmax activation function on top of the spatial aggregation s. The probability of class $\boldsymbol{y}$ for image $\boldsymbol{x}$ is: $P(\boldsymbol{y}|\boldsymbol{x}) = \exp(s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})) / \sum_{\boldsymbol{y}' \in \mathcal{Y}} \exp(s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}'))$ with its corresponding log loss during training.

Multi-label classification In the case of multiple labels, we use a one-against-all strategy, as (Oquab et al. 2015). For C different classes, we train the C binary classifiers jointly, using logistic regression for prediction $P(\boldsymbol{y}|\boldsymbol{x}) = (1 + \exp(-s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})))^{-1}$, with its associated log loss.

5.4.1.2 Ranking: Average Precision

We also tackle the problem of optimizing ranking metrics, and especially Average Precision (AP). We use similar notations that in Subsection 4.3.2. The goal is to predict a ranking matrix $\boldsymbol{y}$. For the ranking model, we assume that $k^+ = k^- = k$. We define the scoring ranking function for category c as follow

$$s^c_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y}) = s^{top}_{\boldsymbol{w},k}(\boldsymbol{x},\boldsymbol{y},c) + s^{low}_{\boldsymbol{w},k}(\boldsymbol{x},\boldsymbol{y},c) \tag{5.7}$$

$$s^{top}_{\boldsymbol{w},k}(\boldsymbol{x},\boldsymbol{y},c) = \max_{\boldsymbol{h}} \sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}} \boldsymbol{y}_{pn}\left[\sum_{z=1}^{r_p} h^{pn}_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_p,c,z) - \sum_{z'=1}^{r_n} h^{np}_{z'} F^t_{\boldsymbol{w}}(\boldsymbol{x}_n,c,z')\right] \tag{5.8}$$

$$s^{low}_{\boldsymbol{w},k}(\boldsymbol{x},\boldsymbol{y},c) = \min_{\boldsymbol{h}} \sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}} \boldsymbol{y}_{pn}\left[\sum_{z=1}^{r_p} h^{pn}_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_p,c,z) - \sum_{z'=1}^{r_n} h^{np}_{z'} F^t_{\boldsymbol{w}}(\boldsymbol{x}_n,c,z')\right] \tag{5.9}$$

where $\mathcal{P}$ (resp. $\mathcal{N}$) is the set of positive (resp. negative) examples, r_p is the number of regions for image p, $\boldsymbol{h}^{pn}$ (resp. $\boldsymbol{h}^{np}$) is a vector, which represents the selected region for image p (resp. n) when we consider the couple of images (p,n), and

$$\boldsymbol{h} = \{(\boldsymbol{h}^{pn},\boldsymbol{h}^{np}) \in \{0,1\}^{r_p} \times \{0,1\}^{r_n}, \sum_{z=1}^{r_p} h^{pn}_z = k,\ \sum_{z=1}^{r_n} h^{np}_z = k,\ (p,n) \in \mathcal{P}\times\mathcal{N}\} \tag{5.10}$$

During training, we aim at minimizing the following loss: $\Delta_{AP}(\boldsymbol{y}^\star,\boldsymbol{y}) = 1 - AP(\boldsymbol{y}^\star,\boldsymbol{y})$, where $\boldsymbol{y}^\star$ is the ground-truth ranking. Since AP is non-smooth, we use the following surrogate (upper-bound) loss:

$$\mathcal{L}_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y}^\star) = \max_{\boldsymbol{y}\in\mathcal{Y}} [\Delta_{AP}(\boldsymbol{y}^\star,\boldsymbol{y}) + s^c_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y})] - s^c_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y}^\star) \tag{5.11}$$

The maximization is generally called Loss-Augmented Inference (LAI). Exhaustive maximization is intractable due to the huge size of the structured output space. The problem is even exacerbated in the WSL setting, see (Behl et al. 2015). We exhibit here the following result for WELDON:

Proposition 5.1. *For each training example, let us denote $s(i) = s^{top}_{\boldsymbol{w},k}(F^t_{\boldsymbol{w}}(\boldsymbol{x}_i,c)) + s^{low}_{\boldsymbol{w},k}(F^t_{\boldsymbol{w}}(\boldsymbol{x}_i,c))$ in Equation 5.6. Inference and LAI for the WELDON ranking model can be exactly solved by sorting examples in descending order of score $s(i)$.*

Proof. We will show that the inference is equivalent to a supervised inference, where each image x_i is represented by $s^{top}_{\boldsymbol{w},k}(F^t_{\boldsymbol{w}}(\boldsymbol{x}_i,c)) + s^{low}_{\boldsymbol{w},k}(F^t_{\boldsymbol{w}}(\boldsymbol{x}_i,c))$. We first show that it is possible to decouple the optimization over the ranking matrix $\boldsymbol{y}$ and the predicted regions $\boldsymbol{h}$:

$$s_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y}) = s^{top}_{\boldsymbol{w},k}(\boldsymbol{x},\boldsymbol{y},c) + s^{low}_{\boldsymbol{w},k}(\boldsymbol{x},\boldsymbol{y},c) \tag{5.12}$$

$$\begin{aligned} &= \max_{\boldsymbol{h}} \sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}} \boldsymbol{y}_{pn}\left[\sum_{z=1}^{r_p} h^{pn}_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_p,c,z) - \sum_{z'=1}^{r_n} h^{np}_{z'} F^t_{\boldsymbol{w}}(\boldsymbol{x}_n,c,z')\right] \\ &\quad + \min_{\bar{\boldsymbol{h}}} \sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}} \boldsymbol{y}_{pn}\left[\sum_{z=1}^{r_p} \bar{h}^{pn}_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_p,c,z) - \sum_{z'=1}^{r_n} \bar{h}^{np}_{z'} F^t_{\boldsymbol{w}}(\boldsymbol{x}_n,c,z')\right] \end{aligned} \tag{5.13}$$

$$\begin{aligned} &= \sum_{p\in\mathcal{P}}\sum_{n\in\mathcal{N}} \left(\max_{(\boldsymbol{h}^{pn},\boldsymbol{h}^{np})} \boldsymbol{y}_{pn}\left[\sum_{z=1}^{r_p} h^{pn}_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_p,c,z) - \sum_{z'=1}^{r_n} h^{np}_{z'} F^t_{\boldsymbol{w}}(\boldsymbol{x}_n,c,z')\right]\right. \\ &\quad \left. + \min_{(\bar{\boldsymbol{h}}^{pn},\bar{\boldsymbol{h}}^{np})} \boldsymbol{y}_{pn}\left[\sum_{z=1}^{r_p} \bar{h}^{pn}_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_p,c,z) - \sum_{z'=1}^{r_n} \bar{h}^{np}_{z'} F^t_{\boldsymbol{w}}(\boldsymbol{x}_n,c,z')\right]\right) \end{aligned} \tag{5.14}$$

The maximization (resp. minimization) can be decomposed for each term of the sum, so maximizing (resp. minimizing) the sum is equivalent to maximize (resp. minimize) each term of the sum. Now, we analyze the predicted regions with respect to $\boldsymbol{y}_{pn}$ value.
If $\boldsymbol{y}_{pn} = 1$

$$\begin{align}
&\max_{(\boldsymbol{h}^{pn},\boldsymbol{h}^{np})} \left[\sum_{z=1}^{r_p} h_z^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z) - \sum_{z'=1}^{r_n} h_z^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z')\right] \tag{5.15}\\
&+ \min_{(\bar{\boldsymbol{h}}^{pn},\bar{\boldsymbol{h}}^{np})} \left[\sum_{z=1}^{r_p} \bar{h}_z^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z) - \sum_{z'=1}^{r_n} \bar{h}_{z'}^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z')\right] \notag\\
=&\left(\max_{\boldsymbol{h}^{pn}} \sum_{z=1}^{r_p} h_z^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z) + \min_{\bar{\boldsymbol{h}}^{pn}} \sum_{z'=1}^{r_p} \bar{h}_{z'}^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z')\right) \tag{5.16}\\
&- \left(\max_{\bar{\boldsymbol{h}}^{np}} \sum_{z=1}^{n_n'} \bar{h}_z^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z) + \min_{\bar{\boldsymbol{h}}^{np}} \sum_{z'=1}^{n_n'} h_{z'}^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z')\right) \notag\\
=&s_{\boldsymbol{w},k}^{top}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c)) + s_{\boldsymbol{w},k}^{low}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c)) - (s_{\boldsymbol{w},k}^{top}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c)) + s_{\boldsymbol{w},k}^{low}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c)) \tag{5.17}
\end{align}$$

If $\boldsymbol{y}_{pn} = -1$

$$\begin{align}
&\max_{(\boldsymbol{h}^{pn},\boldsymbol{h}^{np})} - \left[\sum_{z=1}^{r_p} h_z^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z) - \sum_{z'=1}^{r_n} h_z^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z')\right] \tag{5.18}\\
&+ \min_{(\bar{\boldsymbol{h}}^{pn},\bar{\boldsymbol{h}}^{np})} - \left[\sum_{z=1}^{r_p} \bar{h}_z^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z) - \sum_{z'=1}^{r_n} \bar{h}_{z'}^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z')\right] \notag\\
=&\left(\max_{\boldsymbol{h}^{np}} \sum_{z=1}^{r_p} h_z^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z) + \min_{\bar{\boldsymbol{h}}^{np}} \sum_{z'=1}^{r_p} \bar{h}_{z'}^{np} F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c, z')\right) \tag{5.19}\\
&- \left(\max_{\bar{\boldsymbol{h}}^{pn}} \sum_{z=1}^{n_n'} \bar{h}_z^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z) + \min_{\bar{\boldsymbol{h}}^{pn}} \sum_{z'=1}^{n_n'} h_{z'}^{pn} F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c, z')\right) \notag\\
=&-\left(s_{\boldsymbol{w},k}^{top}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c)) + s_{\boldsymbol{w},k}^{low}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_p, c)) - (s_{\boldsymbol{w},k}^{top}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c)) + s_{\boldsymbol{w},k}^{low}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_n, c))\right) \notag
\end{align}$$

We notice that the predicted regions are the same in the two cases: the predicted regions can be fixed independently to the value of $\boldsymbol{y}_{pn}$. The inference can be written as a supervised inference, where the region are fixed independently to the ranking matrix $\boldsymbol{y}$, and each image x_i is represented by $s_{\boldsymbol{w},k}^{top}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_i, c)) + s_{\boldsymbol{w},k}^{low}(F_{\boldsymbol{w}}^t(\boldsymbol{x}_i, c))$. □

Proposition 5.1 shows that the optimization over regions, i.e. score $s(i)$, decouples from the maximization over output variables $\boldsymbol{y}$. This reduces inference and LAI optimization to fully supervised problems. Inference solution directly corresponds to $s(i)$ sorting. It also allows to use our model with different loss functions, as soon as there is an algorithm to solve the loss-augmented inference in the fully supervised setting. To solve it with Δ_{AP}, we use the greedy algorithm proposed by (Yue et al. 2007), which finds a globally optimal solution.

5.4.2 Optimization

Our model is based on a deep architecture. We initialize it from a model pre-trained on ImageNet (Russakovsky, Deng, et al. 2015) and train it with Stochastic Gradient Descent (SGD) with momentum (Subsection 2.1.2.2) with image-level labels only. All the layers of the network are fine tuned. For multi-class and multi-label predictions, error gradients are well-known. For the ranking instantiation, given a class c, we detail the gradient:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \frac{\partial s^c_{\boldsymbol{w}}(\boldsymbol{x}, \tilde{\boldsymbol{y}})}{\partial \boldsymbol{w}} - \frac{\partial s^c_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}^\star)}{\partial \boldsymbol{w}} \tag{5.20}$$

where $\tilde{\boldsymbol{y}}$ is the LAI solution. The gradient of $s^c_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$ with respect to $\boldsymbol{w}$ is:

$$\begin{aligned}\frac{\partial s^c_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{w}} = \sum_{p \in \mathcal{P}} \sum_{n \in \mathcal{N}} \boldsymbol{y}_{pn} \Bigg[& \left(\sum_{z=1}^{r_p} (h^{p+}_z + h^{p-}_z) \frac{\partial F^t_{\boldsymbol{w}}(\boldsymbol{x}_p, c, z)}{\partial \boldsymbol{w}} \right) \\ & + \left(\sum_{z'=1}^{r_n} (h^{n+}_{z'} + h^{n-}_{z'}) \frac{\partial F^t_{\boldsymbol{w}}(\boldsymbol{x}_n, c, z')}{\partial \boldsymbol{w}} \right) \Bigg]\end{aligned} \tag{5.21}$$

where the latent variables for example i are:

$$\boldsymbol{h}^{i+} = \max_{\boldsymbol{h}} \sum_{z=1}^{r_i} h_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, c, z), \quad \text{s.t.} \sum_{z=1}^{r_i} h_z = k \qquad \forall z \in \{1, \dots, r_i\} \;\; h_z \in \{0, 1\} \tag{5.22}$$

$$\boldsymbol{h}^{i-} = \min_{\boldsymbol{h}} \sum_{z=1}^{r_i} h_z F^t_{\boldsymbol{w}}(\boldsymbol{x}_i, c, z), \quad \text{s.t.} \sum_{z=1}^{r_i} h_z = k \qquad \forall z \in \{1, \dots, r_i\} \;\; h_z \in \{0, 1\} \tag{5.23}$$

When learning WELDON, the gradients are back-propagated through the spatial pooling layer only within the selected regions, all other gradients being discarded. The selection of relevant regions for back-propagation is a key to learn precisely localized features without any spatial supervision (C. Sun et al. 2016).

5.5 Experiments

In this section, we first analyze our model (Subsection 5.5.1), and then compare it to state-of-the-art methods (Subsection 5.5.2). We now briefly present some experimental details. Our deep ConvNet architecture is based on VGG16 (Simonyan et al. 2015). We implement our model using Torch7[1]. We evaluate our WELDON model on several computer vision benchmarks corresponding to various visual recognition tasks. Contrary to MANTRA experiments (Section 4.4) where we choose pre-trained deep features according to the target task, we use only deep features pre-trained on ImageNet whatever the visual recognition task to highlight the generality of our model. We evaluate our model in different recognition contexts: object recognition (PASCAL VOC 2007, PASCAL VOC 2012), scene categorization (MIT67 and 15 Scene), and visual recognition, where context plays an important role (MS COCO, PASCAL VOC 2012 Action). These datasets are presented in Section 2.4. We first provide an analysis of our method on four datasets, and then, we compare it with state-of-the-art methods.

[1] http://torch.ch/

5.5.1 WELDON Analysis

In this section, we analyze the impact on prediction performances of the different contributions of WELDON given in Section 5.3 and Section 5.4. Our baseline model a) is the WSL ConvNet model using an aggregation function s=`max` at the pooling stage, evaluated at scale $\alpha = 30\%$. It gives a network similar to (Oquab et al. 2015), trained at a single scale. To measure the importance of the difference between WELDON and a), we perform a systematic evaluation on the performance when the following variations are incorporated:

b) Use of k^+ top instances instead of the `max`. We use $k^+ = 3$.

c) Incorporation of negative evidence through `max+min` aggregation function. When b)+c) are combined, we use k^- lowest-instances instead of the `min`, with $k^- = 3$.

d) Learning the deep WSL with ranking loss, e.g. AP, in the concerned datasets (PASCAL VOC).

e) Fine-tuning (FT) of the network on the target dataset.

The results are reported in Table 5.2 for object and context datasets with AP evaluation (VOC 2007 and VOC 2012 action), and in Table 5.3 for scene datasets. From this systematic evaluation, we can draw the following conclusions:

- Both b) and c) improvements result in a very large performance gain on all datasets, with a comparable impact on performances: about +30 pt on MIT67, +15 pt on 15-Scene, +15 pt on VOC 2012 Action and +4 pt on VOC 2007. When looking more accurately, we can notice that `max+min` leads always to a larger improvement, e.g. is 4 pt above on 15-Scene or VOC 2012 Action and 3 pt on MIT67. Note that this model is equivalent to MANTRA with VGG16 deep features.

- Combining b) and c) improvements further boost performances: +3 pt on MIT67 and VOC 2012 Action, +2 pt on 15-Scene, +1pt on VOC 2007. This shows the complementarity of negative evidence and top instances. We perform an additional experiment for comparing b)+c) and c), by setting the same number of regions (e.g. 6-`max` and 3-`max`+3-`min`). It turns out that k^+-`max`+k^--`min` is the best method for various k^+/k^- values, showing that negative evidence contains significant information for visual prediction.

- Similarly to MANTRA experiments (Subsection 4.4.2), we note that minimizing an AP loss enables to further improve performances. Interestingly, the same level of improvement is observed when AP optimizing is added to the c) configuration than to the more powerful b)+c) configuration: +3pt on VOC 2012 Action, +1 pt on VOC 2007. This shows that b) and c) are conditionally independent from the AP optimization.

- Fine-tuning favorably impacts performances, with +0.6 pt gain on MIT67 and 15-Scene. Note that the performance level is already high at the b)+c) configuration, making further improvements challenging.

a) max	b) +top	c) +min	d) +AP	VOC 2007	VOC 2012 ACTION
✓				83.6	53.5
✓	✓			86.3	62.6
✓		✓		87.5	68.4
✓		✓	✓	88.4	71.7
✓	✓	✓		87.8	69.8
✓	✓	✓	✓	88.9	72.6

Table 5.2.: Systematic evaluation of our WSL deep ConvNet contributions on object and context datasets (MAP evaluation).

a) max	b) +top	c) +min	e) +FT	MIT67	15 SCENE
✓				42.3	72.0
✓	✓			69.5	85.9
✓		✓		72.1	89.7
✓	✓	✓		74.5	90.9
✓	✓	✓	✓	75.1	91.5

Table 5.3.: Systematic evaluation of our WSL deep ConvNet contributions on scene datasets (multi-class accuracy).

Finally, we show in Figure 5.2 the performance in different configurations, corresponding to sequentially adding the previous improvements in the following order: a), a)+b), a)+b)+c), and a)+b)+c)+d) for VOC 2007 / VOC 2012 / VOC 2012 Action and a)+b)+c)+e) for MIT67 and 15 Scene. On all dataset, we can see the very large improvement from configuration a) to configuration a)+b)+c)+d)/e). The behavior can, however, be different among datasets: for example, the performance boost is sharp from a) to a)+b) on MIT67 (the following improvements being less pronounced), whereas there is a linear increase from a)+b)+c)+d) on VOC 2007 and VOC 2012. Note that the analysis of the number of selected regions is done in the next chapter (Subsection 6.5.2).

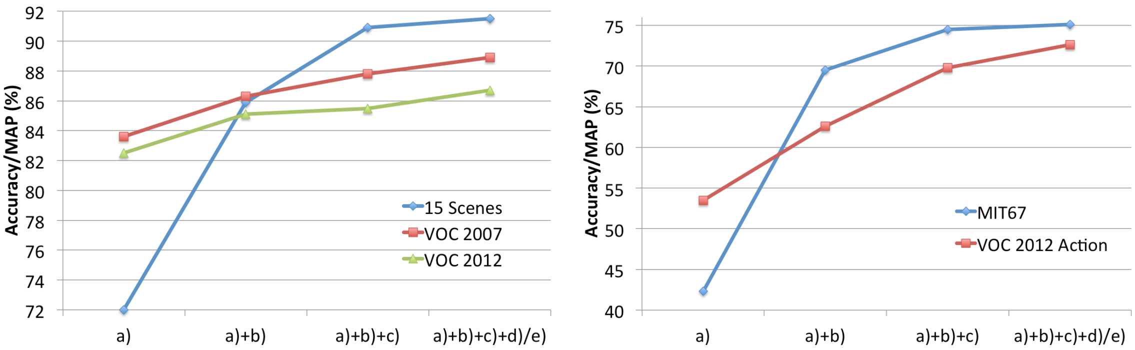


Figure 5.2.: Performance variations when the different improvements are incorporated: from the baseline model a) to, a)+b), a)+b)+c), and a)+b)+c)+d)/e).

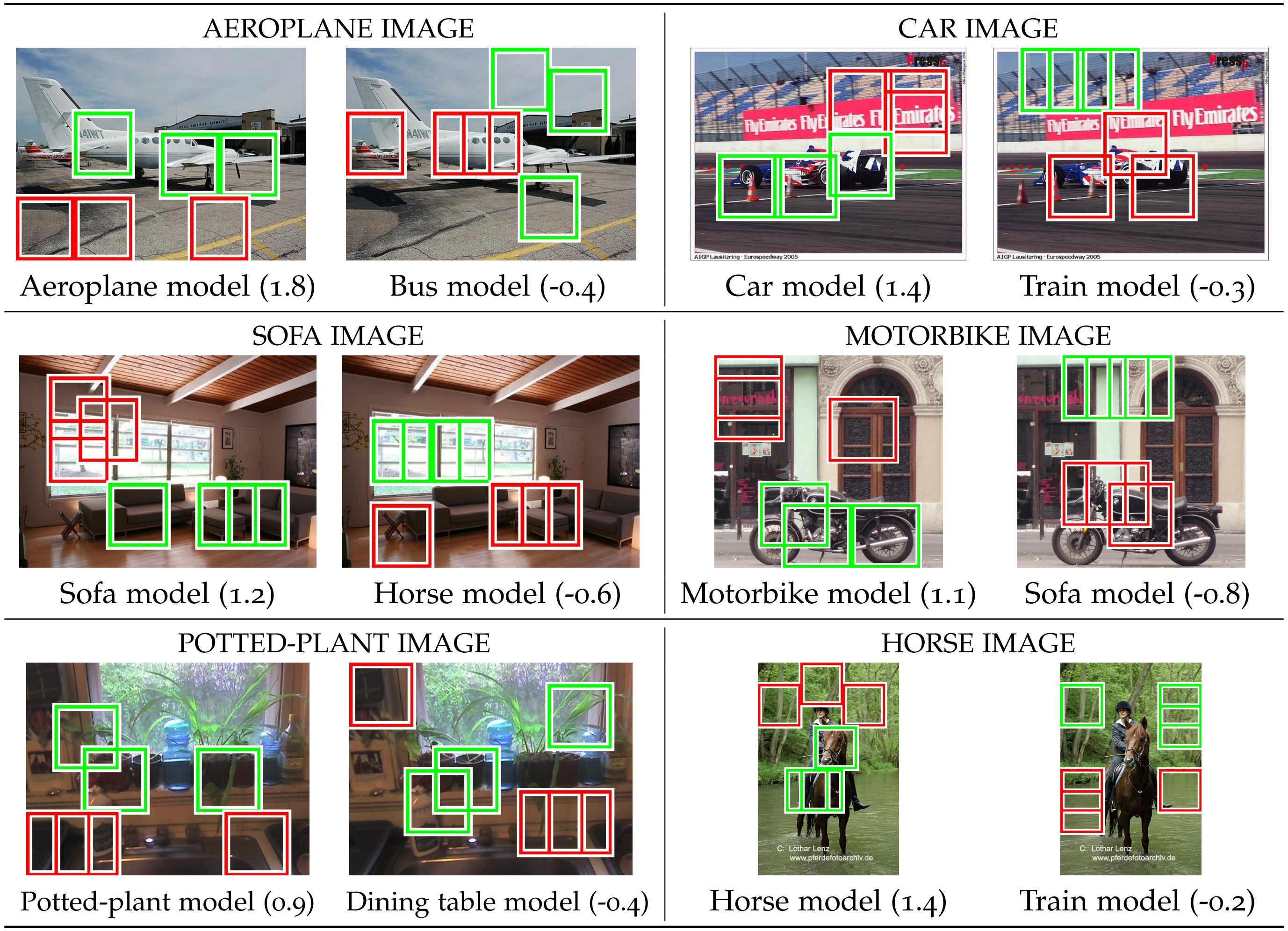


Figure 5.3.: Visual results of WELDON on VOC 2007 with $k^+ = k^- = 3$ instances. The green (resp. red) boxes are the 3 top (resp. 3 low) instances. For each image, the first column represents WELDON prediction for the ground truth classifier (with its corresponding score), and the second column shows prediction and score for an incorrect classifier.

Qualitative analysis of region selection To illustrate the region selection policy performed by WELDON, we show in Figure 5.3 the top 3 positive (resp. top 3 negative) regions selected by the model in green (resp. red), on the VOC 2007 dataset. We show the results for the ground truth classification model in the first column, with its associated prediction score. We can notice that top positive green regions detect several discriminant parts related to the object class, potentially capturing several instances or modalities (e.g. wheels or airfoil for the car model), whereas negative evidence on red regions, which should remain small, encode contextual information (e.g. road or sky for airplane, or trees for horse). The region selection results are shown for incorrect classification models in the second column, again with the prediction score. We can notice that red regions correspond to multiple negative evidences for the class, e.g. parts of coach strongly penalize the prediction of the class horse, or seat or handlebar negatively supports the prediction of the sofa category.

5.5.2 Comparison with State-of-the-Art Methods

We now compare the WELDON model to state-of-the-art methods. We use the multi-scale WSL model described in Subsection 5.3.1, and we use the same scale combination that in Section 4.4. For the selection of top/low instances, we use here the default setting of $k^+ = k^- = 3$, for scale $\alpha \leq 70\%$ (Table 5.1). Results for object (resp. scene and context) datasets are gathered in Table 5.4 (resp. Table 5.5 and Table 5.6).

METHOD	VOC 2007	VOC 2012 (VAL)
Return Devil (Chatfield et al. 2014)	82.4	-
VGG16 (Simonyan et al. 2015)	84.5	82.8
SPP-net (He et al. 2014)	82.4	-
Deep MIL (Oquab et al. 2015)	-	81.8
MANTRA (Chapter 4)	85.8	-
WELDON	**90.2**	**88.5**

Table 5.4.: MAP results on object recognition datasets. WELDON and state-of-the-art methods results are reported.

For object datasets, we can show in Table 5.4 that WELDON outperforms all recent methods based on deep features by a large margin. More specifically, the improvement compared to deep features computed on the whole image (Chatfield et al. 2014; Simonyan et al. 2015) is significant. Note that since we use deep features VGG16 from (Simonyan et al. 2015), the performance gain directly measures the relevance of using a WSL method, which selects localized evidence for performing prediction, rather than relying on the whole image information. Compared to SPP-net (He et al. 2014), the improvement of about 8 pt on VOC 2007 highlights the superiority of region selection based on supervised information, rather than using handcrafted aggregation with SPM strategy. Then, we compare WELDON to recent WSL methods based on deep features: Deep MIL(Oquab et al. 2015) and MANTRA. As observed in Subsection 5.5.1, we note that our multi-regions pooling strategy significantly outperforms max (Deep MIL) and max+min (MANTRA) pooling. This big improvement illustrates the positive impact of incorporating MIL relaxations for WSL training of deep ConvNets, i.e. negative evidence scoring and top-instance selection. Finally, we can point out the outstanding score reached by WELDON on VOC 2007, exceeding the nominal score of 90%.

METHOD	15 SCENE	MIT67
CaffeNet ImageNet (Jia et al. 2014)	84.2	56.8
CaffeNet Places (B. Zhou et al. 2014)	90.2	68.2
VGG16 (Simonyan et al. 2015)	91.2	69.9
MOP CNN (Gong et al. 2014)	-	68.9
MANTRA (Chapter 4)	93.3	76.6
Negative parts (Parizi, Andrea Vedaldi, et al. 2015)	-	77.1
WELDON	**94.3**	**78.0**

Table 5.5.: Multiclass accuracy results on scene categorization datasets. WELDON and state-of-the-art methods results are reported.

The results shown in Table 5.5 for scene recognition also illustrate the significant improvement of WELDON compared to deep features computed on the whole image (Jia et al. 2014; B. Zhou et al. 2014; Simonyan et al. 2015) and MOP CNN (Gong et al. 2014), a BoW method pooling deep features with VLAD. It is worth noticing that WELDON also outperforms recent part-based methods including negative evidence during training (Parizi, Andrea Vedaldi, et al. 2015). This shows the improvement brought out by the end-to-end deep WSL ConvNet training with WELDON. Note that WELDON outperforms MANTRA, which used deep features trained on Places.

In Table 5.6, we show the results in datasets where contextual information is important for performing prediction. On VOC 2012 action and MS COCO, selecting the regions corresponding to objects or parts directly related to the class is important, but contextual features are also strongly related to the decision. WELDON outperforms VGG16 (Simonyan et al. 2015) by ~ 8 pt on both datasets, again validating our WSL deep method in this context. On MS COCO, the improvement is from 62.8% (Oquab et al. 2015) to 68.8% for WELDON. This shows the importance of the negative evidence and top-instance scoring in our prediction module, which better help to capture contextual information than the standard MIL max function used in (Oquab et al. 2015). Finally, note that the very good results in MS COCO also illustrate the efficiency of the proposed WSL training of deep ConvNet with WELDON, which is able to deal with this large dataset (80 classes and $\sim 80,000$ training examples).

5.6 Conclusion

In this chapter, we introduced a novel framework for WEakly supervised Learning of Deep cOnvolutional neural Network (WELDON). WELDON is trained to automatically

METHOD	VOC 2012 ACTION	MS COCO
VGG16 (Simonyan et al. 2015)	67.1	59.7
Deep MIL (Oquab et al. 2015)	-	62.8
WELDON	**75.0**	**68.8**

Table 5.6.: WELDON results and comparison to state-of- the-art methods on context datasets.

select relevant regions from images annotated with global labels, and to perform end-to-end learning of a deep ConvNet from the selected regions. The whole architecture is carefully designed for fast processing by sharing region feature computations, and for robust training. We also proposed a new pooling function, which generalizes the MANTRA pooling function by incorporating top instance strategy.

We showed the excellent performances of our model on very different visual recognition tasks: object class recognition, scene classification, including images with a cluttered context, outperforming state-of-the-art results on six challenging datasets. The analysis of our model on four datasets showed that both negative evidence and top instance are relevant and complementary.

We showed that using several regions is important to have robust predictions. In our experiments, we gave the same importance of the positive and negative evidence terms, i.e. $k^+ = k^-$. Maximum and minimum scoring regions are complementary, but there is no obvious reason that they should have the same importance. In the next chapter, we introduce a more general aggregation strategy, that controls the relative importance of the positive and negative evidence terms.

Chapter 6

WILDCAT: SPATIAL AND CLASS-WISE POOLING FOR IMAGE CLASSIFICATION, LOCALIZATION AND SEGMENTATION

Contents

Chapter abstract

This chapter introduces WILDCAT, a method to learn localized visual features related to class modalities. WILDCAT extends WELDON (Chapter 5) at three major levels: the use of Fully Convolutional Network (FCN) for maintaining spatial resolution, the explicit design in the network of local features related to different class modalities, and a new way to pool these features to provide a global image prediction required for Weakly Supervised Learning (WSL). The proposed model is used to perform image classification as well as weakly supervised pointwise object localization and segmentation.

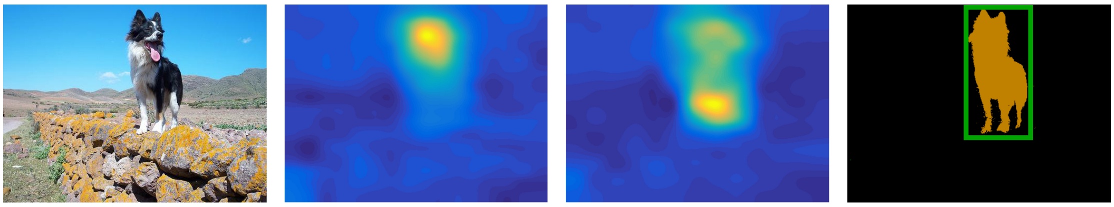
The work in this chapter has led to the publication of a conference paper:

- Thibaut Durand⋆, Taylor Mordan⋆, Nicolas Thome, and Matthieu Cord (2017). "WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

6.1 Introduction

Global spatial pooling is a crucial step for WSL of deep ConvNets with image-level labels. In the Chapter 5, we introduced WELDON pooling, which explicitly encodes negative evidence and top instance strategy. In this chapter, we propose a new pooling function, which generalizes WELDON pooling by introducing a relative weight between `max` and `min` regions. In Section 6.5, we also propose a unified framework for pooling, which generalizes standard WSL pooling functions as special cases, including our model. This analysis of pooling functions is supplemented by a detailed experimental comparison on six datasets.

In addition to the new spatial pooling, our model Weakly supervIsed Learning of Deep Convolutional neurAl neTwork (WILDCAT) extends WELDON model at two other levels. As spatial resolution is crucial for localization, we propose a new Fully Convolutional Network (FCN) architecture based on ResNet-101 (He et al. 2016) to learn feature maps with more resolution than WELDON model. The second novelty, which is the main one, is the introduction of a multi-map transfer stage to enrich the class model. While the WELDON model learns one modality per class, the WILDCAT model explicitly learns multiple localized features related to complementary class modalities in a WSL fashion. In Figure 6.1, we show the heatmaps of two modalities of the *dog* class: the first modality (b) focuses on the head, while the second modality (c) focuses on the legs of the dog. These modalities allow to focus on specific parts of objects to have a more discriminative model. The WILDCAT model can be used to perform image classification as well as weakly supervised pointwise object localization and segmentation (Figure 6.1 (d)).



(a) original image (b) *dog* heatmap 1 (c) *dog* heatmap 2 (d) final predictions

Figure 6.1.: WILDCAT example performing localization and segmentation (d), based on different class-specific modalities, here head (b) and legs (c) for the *dog* class.

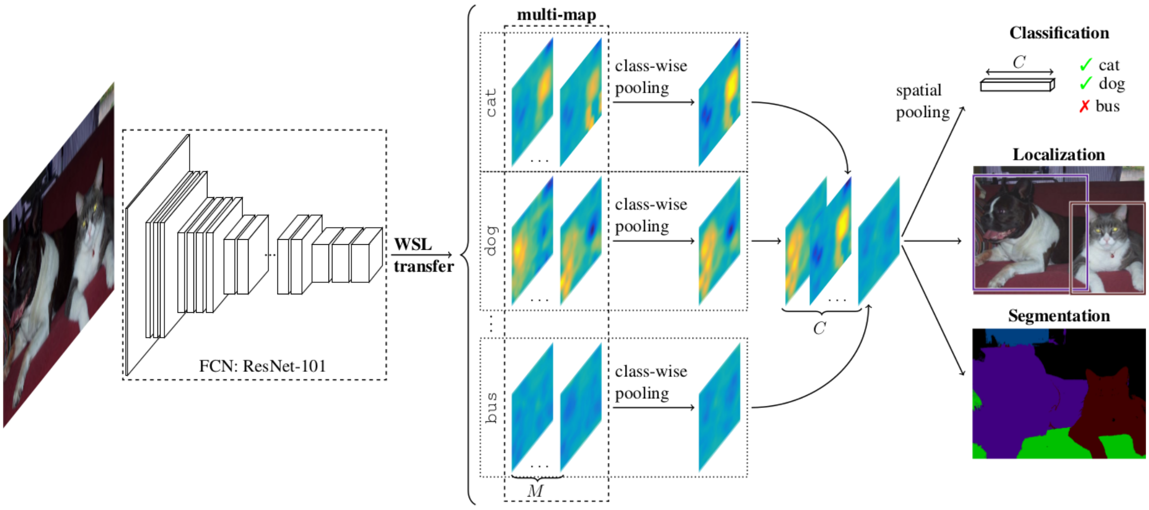


Figure 6.2.: WILDCAT architecture. The first map of dog (resp. cat) category mostly focuses on the head of the dog (resp. cat) whereas the second map focuses on the legs of the dog (resp. cat).

6.2 WILDCAT Model

The overall WILDCAT architecture (Figure 6.2) is based on a FCN which is suitable for spatial predictions (Long et al. 2015), a multi-map WSL transfer layer encoding modalities associated with classes, and a global pooling for WSL that learns accurate localization. We now delve into each of the three parts of the model.

6.2.1 Fully Convolutional Architecture

The selection of relevant information within feature maps is a major issue in WSL. It impacts the localization of the learned representation and the precision of the results (e.g. semantic segmentation or object detection). We thus expect the resolution of the feature maps to be a key component for WILDCAT: finer maps keep more spatial resolution and lead to more specific regions (e.g. objects, parts).

To this end we exploit the recently introduced FCN ResNet-101 (He et al. 2016) (left of Figure 6.2) that naturally preserves spatial information throughout the network. It also computes local features from all the regions in a single forward pass, without resizing them. Besides, ResNet architectures are effective at image classification while being parameter- and time-efficient (He et al. 2016). This kind of architecture has been exploited to speed up computation and to produce accurate spatial predictions in fully supervised setups, e.g. in object detection (Dai, Y. Li, et al. 2016) and semantic segmentation (Dai, He, et al. 2016).

We use the publicly released model pre-trained on ImageNet dataset (Russakovsky, Deng, et al. 2015) and remove the last layers (global average pooling and fully connected) to replace them with WSL transfer and wildcat pooling layers (Figure 6.3) described in the following.

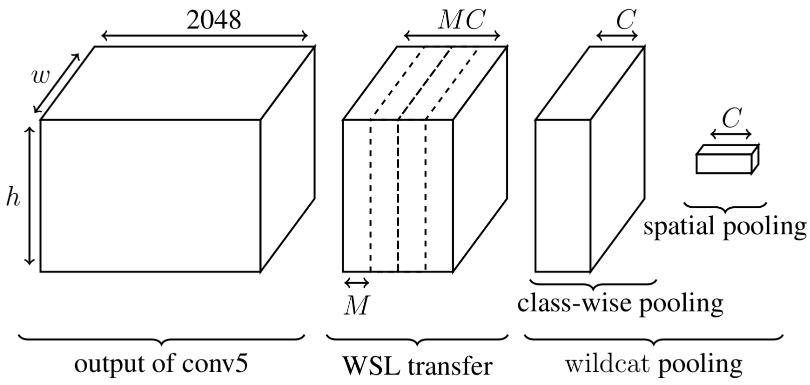


Figure 6.3.: WILDCAT local feature encoding and pooling. Class modalities are encoded with a multi-map WSL transfer layer and pooled separately for all classes. Local features are then aggregated with a global spatial pooling to yield a single score per class.

6.2.2 Multi-map Transfer Layer

We introduce a multi-map WSL transfer layer that learns multiple class-related modalities, encoded into M feature maps per class through 1×1 convolutions (middle of Figure 6.2). The modalities are learned in a WSL fashion with only the image-level labels and the transfer layer keeps spatial resolution, key in WSL. We note $w \times h \times d$ the size of conv5 maps of ResNet-101, which is $\frac{W}{32} \times \frac{H}{32} \times 2048$ for an original image of size $W \times H \times 3$. The transfer output is then of size $w \times h \times MC$ (Figure 6.3). Given an example $\boldsymbol{x}$, we note $F_{\boldsymbol{w}}^{multi}(\boldsymbol{x}, (\boldsymbol{y}, m), \boldsymbol{h})$ the output of the m-th map of class $\boldsymbol{y}$ at location $\boldsymbol{h}$.

The M modalities aim at specializing to different class-specific features, e.g. parts (Dai, He, et al. 2016; Dai, Y. Li, et al. 2016) (head and legs of dog in Figure 6.1) or views (Felzenszwalb et al. 2010; Ross Girshick et al. 2015). We highlight differences with some specific encoding approaches: position-sensitive Region-of-Interest (RoI) pooling in Region-based Fully Convolutional Network (R-FCN) (Dai, Y. Li, et al. 2016) forces position-based specialization (relative to the object) while our method can also learn other kind of features, e.g. semantic parts (Figure 6.1). In the same way DPM (Felzenszwalb et al. 2010) learns only discriminating parts where our multi-map transfer model can find more general features, e.g. context. Furthermore, contrarily to the DPM where a different model is learned for each view, we share most of the computation within the FCN, which is more efficient.

Learning multiple modalities allows to fully exploit available spatial resolution by making a better use of all the information contained at each location. We note that when $M = 1$ this reduces to a standard classification layer, i.e. into C classes. We empirically show that $M > 1$ yields better results than regular $M = 1$.

6.2.3 Wildcat Pooling

WILDCAT learns from image-level labels so we need a way to summarize all information contained in the feature maps for each class (right of Figure 6.2). We note that there are no more learned parameters in this pooling layers, which means we can directly interpret and visualize feature maps at this level (Bolei Zhou, Khosla, et al. 2016; Dai, Y. Li, et al. 2016).

We perform this in two steps (Figure 6.3): a class-wise pooling (Equation 6.2) that combines the M maps from the multi-map transfer layer, then a spatial pooling module (Equation 6.1) that selects relevant regions within the maps to support predictions. This leads to wildcat pooling, a two-stage pooling operation to compute the score $s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$ of class $\boldsymbol{y}$ given an input $\boldsymbol{x}$:

$$s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = \underset{\boldsymbol{h} \in \mathcal{H}}{\mathrm{Sp.\,Pool}} \; F^t_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) \tag{6.1}$$

$$\text{where} \quad F^t_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h}) = \underset{m \in \{1, \ldots, M\}}{\mathrm{Cl.\,Pool}} \; F^{multi}_{\boldsymbol{w}}(\boldsymbol{x}, (\boldsymbol{y}, m), \boldsymbol{h}) \tag{6.2}$$

where $F^t_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$ is the score for category $\boldsymbol{y}$ at location $\boldsymbol{h}$ (after class-wise pooling), Cl. Pool is the chosen class-wise pooling function and Sp. Pool is the spatial aggregation process.

Class-wise pooling The first step consists in combining the M maps for all classes independently, and is described in Equation 6.2 with a generic pooling function Cl. Pool. We use average pooling in the following. The maps are transformed from $w \times h \times MC$ to $w \times h \times C$ (Figure 6.3). When $M = 1$ this operation is not needed as each class is already represented by a single map.

We note that even if a multi-map followed by an average pooling is functionally equivalent to a single convolution (i.e. $M = 1$), the explicit structure it brings with M modalities has important practical advantages making training easier. We empirically show that $M > 1$ yields better results than regular $M = 1$.

Spatial pooling We now introduce our new spatial aggregation method implementing the second, spatial pooling step (Equation 6.1) for each map $\boldsymbol{y}$:

$$s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y}) = s^{top}_{\boldsymbol{w},k^+}(F^t_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})) + \alpha \cdot s^{low}_{\boldsymbol{w},k^-}(F^t_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})) \tag{6.3}$$

where $F^t_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$ is the score map for category $\boldsymbol{y}$ and $s^{top}_{\boldsymbol{w},k^+}$ (resp. $s^{low}_{\boldsymbol{w},k^-}$) are defined in Equation 5.4 (resp. Equation 5.5), and $\alpha \in [0, 1]$ is a trade off parameter. It consists in selecting for each class $\boldsymbol{y}$ the k^+ (resp. k^-) regions with the highest (resp. lowest) activations from input $\bar{z}^c$. The output $s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$ for class $\boldsymbol{y}$ of this layer is the weighted average of scores of all the selected regions. We only consider regions defined by single neurons in the convolutional feature maps.

Maximum and minimum scoring regions both are important for good results, but do not bring the same kind of information. We explore relative weighting of both types of regions by introducing a factor α which trades off relative importance between both terms. We hypothesize that maximum scoring regions are more useful for classification as they directly support the decision, while minimum scoring regions essentially act as regularization. With $\alpha < 1$ WILDCAT should focus more on discriminating regions and then better localize features than with $\alpha = 1$.

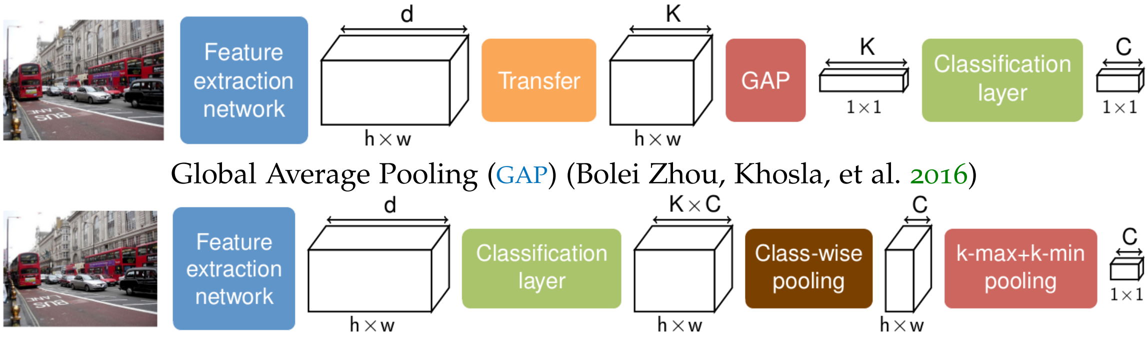
Pooling	k^+	k^-	α
Maximum / MIL (Oquab et al. 2015)	1	0	0
Top instances (W. Li et al. 2015)	k	0	0
LLP (F. X. Yu et al. 2013)	ρn	0	0
GAP (Bolei Zhou, Khosla, et al. 2016)	n	0	0
MANTRA (Chapter 4)	1	1	1
WELDON (Chapter 5)	k	k	1

Table 6.1.: Generalization of wildcat spatial pooling to other existing MIL approaches with corresponding parameters. n is the total number of regions, ρ is the proportion of positive labels in LLP, k is an arbitrary number of regions to choose.

Several similar MIL approaches have been used but our proposed model generalizes them in numerous of ways. The corresponding parameters are described in Table 6.1. The standard max-pooling MIL approach (Oquab et al. 2015) is obtained with only one element, and both top instance model (W. Li et al. 2015), Learning with Label Proportion (LLP) (F. X. Yu et al. 2013) and global average pooling (Bolei Zhou, Khosla, et al. 2016) can be obtained with more. Drawing from negative evidence (Parizi, Andrea Vedaldi, et al. 2015), we can incorporate minimum scoring regions to support classification and our spatial pooling function can reduce to the WELDON pooling (Chapter 5).

6.2.4 Architecture Discussion

WILDCAT architecture is composed of a transfer layer followed by pooling. Since there are no parameters to learn in the pooling module, the transfer layer performs classification and it is easy to visualize heatmaps with direct localization of discriminating regions (Figure 6.4 second row). We note that this kind of architecture is reversed in (Bolei Zhou, Khosla, et al. 2016) where pooling is performed before the last fully connected layer (Figure 6.4 first row), as in the original ResNet architecture (He et al. 2016) for example. However this order requires an unnatural way of visualizing class-specific heatmaps.



Global Average Pooling (GAP) (Bolei Zhou, Khosla, et al. 2016)

WILDCAT

Figure 6.4.: Network architecture comparison

It is shown in (Bolei Zhou, Khosla, et al. 2016) that if the spatial aggregation method is linear, e.g. Global Average Pooling (GAP), then the order of both layers is not important, but the two configurations can behave differently with a non linear pooling function such as wildcat spatial pooling. The difference is more significant when $k^+ + k^-$ is low, i.e. when wildcat spatial pooling really differs from global average pooling. We evaluate the impact of this design choice and of the chosen pooling function in the experiments and show that our architecture yields better results.

6.2.5 WILDCAT Applications

Training phase Our WILDCAT model is based on the backbone architecture ResNet-101 (He et al. 2016). We initialize it from a model pre-trained on ImageNet (Russakovsky, Deng, et al. 2015) and train it with Stochastic Gradient Descent (SGD) with momentum with image-level labels only. All the layers of the network are fine-tuned. The input images are warped to a square size at a given scale. We use a multi-scale setup where a different model is learned for each scale and they are combined with Object Bank (OB) (L.-J. Li, Su, et al. 2014) strategy.

WILDCAT is designed to learn from image-level supervision only: the same training procedure is used for image classification, weakly supervised object detection and weakly supervised semantic segmentation. When learning WILDCAT, the gradients are back-propagated through the wildcat layer only within the $k^+ + k^-$ selected regions, all other gradients being discarded. The selection of right regions for back-propagation is key to learn precisely localized features without any spatial supervision (C. Sun et al. 2016).

Inference phase Predictions differ according to the task at hand. For image classification, prediction simply takes the single-value output of the network (like in training). Object detection and semantic segmentation require spatial predictions so we extract the class-specific maps before spatial pooling to keep spatial resolution. They are at resolution $\frac{1}{32}$ with respect to the input image for ResNet-101 architecture (He et al. 2016). For weakly supervised object detection, we extract the region (i.e. neuron in the feature map) with maximum score for each class and use it for point-wise localization, as it is done in (Oquab et al. 2015; Bency et al. 2016). For weakly supervised semantic segmentation we compute the final segmentation mask either by taking the class with maximum score at each spatial position independently or by applying a CRF for spatial prediction as is common practice (L. Chen et al. 2015; Pathak, Krahenbuhl, et al. 2015).

6.3 Classification Experiments

In order to get results in very different recognition contexts, several datasets are used: object recognition (Pascal VOC 2007, Pascal VOC 2012, MS COCO), scene categorization (MIT67), action recognition (Pascal VOC 2012 Action) and fine-grained recognition (CUB-200). These datasets are presented in Section 2.4. We first analyze the hyper-parameters of our model on three datasets (Subsection 6.3.1), and then, we compare it to state-of-the-art methods on six datasets (Subsection 6.3.2).

6.3.1 WILDCAT Analysis

In this section, we analyze the impact of our contributions on three datasets: VOC 2007, VOC 2012 Action and MIT67. We study our network architecture, the impact of the weighting hyper-parameter α and the number of modalities M per class. We present results for an input image of size 448×448 and $k^+ = k^- = 1$, but similar behaviors are observed for other scales and larger k^+ and k^-. By default, our model parameters α and M are fixed to 1.

6.3.1.1 Deep Structure

Firstly, to validate the design choice of the proposed WILDCAT architecture, we evaluate two different configurations presented in Subsection 6.2.5:

(A) conv5 + conv + pooling (our architecture);

(B) conv5 + pooling + conv (architecture proposed in (Bolei Zhou, Khosla, et al. 2016)).

These two configurations are different for the non-linear WILDCAT pooling scheme described in Subsection 6.2.3, and their comparison is reported in Table 6.2. We can see that our architecture (A) leads to a consistent improvement over architecture (B) used in GAP (Bolei Zhou, Khosla, et al. 2016) on all three datasets, e.g. 1.7 pt on VOC 2007. Note that the strategy of architecture (A) has a very different interpretation from (B): (A) classifies each region independently and then pools the region scores, whereas (B) pools the output of the convolution maps and then performs image classification on the pooled space.

Method	VOC 2007	VOC 2012 Action	MIT67
Architecture (A)	**89.0**	**78.9**	**69.6**
Architecture (B)	87.3	77.5	68.1

Table 6.2.: Classification performances for architectures (A) and (B).

6.3.1.2 Impact of Parameter α

We investigate the effect of the weighting hyper-parameter α on classification performances. From the results in Figure 6.5, it is clear that incorporating negative evidence, i.e. $\alpha > 0$, is beneficial for classification, compared to standard max pooling, i.e. $\alpha = 0$. We further note that using different weights for maximum and minimum scores, i.e. $\alpha \neq 1$, yields better results than with $\alpha = 1$ (i.e. WELDON), with best improvement of 1.6 pt (resp. 2 and 1.8) with $\alpha = 0.6$ (resp. 0.7 and 0.8) on VOC 2007 (resp. VOC 2012 Action and MIT67). This confirms the relevance of using a relative weighting for negative evidence. Moreover our model is robust with respect to the value of α.

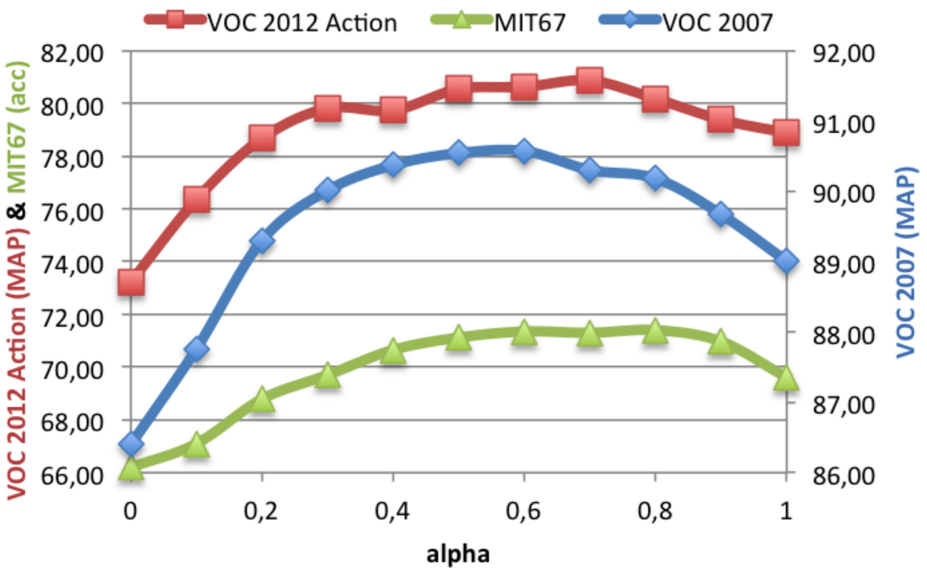


Figure 6.5.: Analysis of parameter α.

6.3.1.3 Number of Modalities

Another important hyper-parameter of our model is the number of modalities (M) used in the multi-map transfer layer. The performances for different values of M are reported in Table 6.3. Explicitly learning multiple modalities, i.e. $M > 1$, yields large gains with respect to a standard classification layer, i.e. $M = 1$ (WELDON). However encoding more modalities than necessary (e.g. $M = 16$) might lead to overfitting since the performances decrease. The best improvement is 3.5 pt (resp. 4.3 and 3.5) with $M = 8$ (resp. 8 and 12) on VOC 2007 (resp. VOC 2012 Action and MIT 67). Examples of heatmaps for the same category are shown in Figure 6.7.

M	1	2	4	8	12	16
VOC 2007	89.0	91.0	91.6	**92.5**	92.3	92.0
VOC 2012 Action	78.9	81.5	82.1	**83.2**	83.0	82.7
MIT67	69.6	71.8	72.0	72.8	**73.1**	72.9

Table 6.3.: Analysis of multi-map transfer layer.

6.3.1.4 Ablation Study

We perform an ablation study to illustrate the effect of each contribution. Our baseline is a WSL transfer with $M = 1$ and the spatial pooling with $\alpha = 1$. The results are reported in Table 6.4. From this ablation study, we can draw the following conclusions:

- Both $\alpha = 0.7$ and $M = 4$ improvements result in large performance gains on all datasets;

- Combining $\alpha = 0.7$ and $M = 4$ improvements further boost performances: 0.4 pt on VOC 2007, 0.8 pt on VOC 2012 Action and 0.8 on MIT67. This shows the complementarity of both these contributions.

max+min	$\alpha = 0.7$	$M = 4$	VOC 2007	VOC 2012 Action	MIT67
✓			89.0	78.9	69.6
✓	✓		90.3	80.9	71.3
✓		✓	91.6	82.1	72.0
✓	✓	✓	**92.0**	**82.9**	**72.8**

Table 6.4.: Ablation study on VOC 2007, VOC 2012 Action and MIT67. The results are different from results of Subsection 6.3.2 because only one scale is used for this analysis.

6.3.2 Comparison with State-of-the-Art Methods

Firstly, we compare the proposed WILDCAT model to state-of-the-art methods. We use different image size as input of our model, and scale combination is performed using an Object-Bank (L.-J. Li, Su, et al. 2014) strategy. The image size and the values of k^+/k^- are given in Table 6.5. The hyper-parameters of our model are fixed at $M = 4$ and $\alpha = 0.7$. An analysis of the number of selected regions is done in Subsection 6.5.2, showing further improvements by careful tuning. Results are gathered in Table 6.6 and Table 6.7.

Image size	Size before pooling	k^+, k^-
224×224	7×7	3
280×280	9×9	5
320×320	10×10	10
374×374	12×12	15
448×448	14×14	20
560×560	18×18	25
747×747	24×24	30

Table 6.5.: Multi-scale setup for WILDCAT model. We detail the input image sizes, along with the sizes of the feature maps before spatial pooling and the parameter values used in the spatial pooling.

Object datasets We report in Table 6.6 the performances for object datasets, and we can see that WILDCAT outperforms all recent methods based on deep features by a large margin. The most important comparison is the improvement over other recent WSL methods on deep features (Oquab et al. 2015; C. Sun et al. 2016; Z. Wei et al. 2016), MANTRA, WELDON. We outperform the deep WSL ConvNet in (Oquab et al. 2015), the approach which is the most closely connected to ours, by 7.1 pt (resp. 17.9) on VOC 2012

Method	VOC 2007	VOC 2012	MS COCO
Return Devil (Chatfield et al. 2014)	82.4	-	-
VGG16 (Simonyan et al. 2015)	89.3	89.0	-
SPP-net (He et al. 2014)	82.4	-	-
NUS-HCP (Y. Wei et al. 2014)	85.2	84.2	-
ResNet-101 (He et al. 2016) †	89.8	89.2	72.5
Deep MIL (Oquab et al. 2015)	-	86.3	62.8
MANTRA (Chapter 4)	85.8	-	-
WELDON (Chapter 5)	90.2	-	68.8
ProNet (C. Sun et al. 2016)	-	89.3	70.9
RRSVM (Z. Wei et al. 2016)	92.9	-	-
SPLeaP (Kulkarni, Jurie, et al. 2016)	88.0	-	-
WILDCAT	**95.0**	**93.4**	**80.7**

Table 6.6.: MAP results on object recognition datasets. WILDCAT and state-of-the-art methods results are reported. Half at the top shows the performances using global image representation, whereas the half at the bottom shows performances for models based on regions selection. † means that the results are obtained by fine-tuning the network on the dataset with the code `https://github.com/facebook/fb.resnet.torch`.

(resp. MS COCO). This big improvement illustrates the positive impact of incorporating MIL relaxations for WSL training of deep ConvNets, i.e. negative evidence scoring and top-instance selection. We also note a significant gain of 4.1 pt (resp 9.8) on VOC 2012 (resp. MS COCO) with ProNet (C. Sun et al. 2016), that relaxes the `max` pooling with a LSE pooling. WILDCAT also outperforms by 2.1 pt the RRSVM (Z. Wei et al. 2016) on VOC 2007, which learn a constrained aggregation operator on all the regions. Compared to WELDON, the improvement of 4.8 pt (resp. 11.9) on VOC 2007 (resp. MS COCO) essentially shows the importance of the ResNet-101 that preserves spatial information throughout the network, and allows finer maps to learn more specific regions.

Scene, action and fine-grained datasets We also validate our model on scene, action and fine-grained classification. The results are reported in Table 6.7 and illustrate the big improvement of WILDCAT compared to deep features computed on the whole image (B. Zhou et al. 2014; Simonyan et al. 2015; He et al. 2016) and global image representation with deep features computed on image regions: MOP CNN (Gong et al. 2014) and Compact Bilinear Pooling (Gao et al. 2016). It is worth noticing that WILDCAT also outperforms recent part-based methods (Wu et al. 2015; Kulkarni, Jurie, et al. 2016) including negative evidence during training (Parizi, Andrea Vedaldi, et al. 2015), MANTRA, WELDON. This shows the improvement brought out by the WILDCAT network architecture. This validates that our region selection approach is better than using all regions. WILDCAT also significantly outperforms, on CUB-200 and MIT67, the recent GoogLeNet-GAP (Bolei Zhou, Khosla, et al. 2016), which used a global average pooling. On CUB-200, we can also

Method	CUB-200	MIT67	VOC Action
CaffeNet Places (B. Zhou et al. 2014)	-	68.2	-
MOP CNN (Gong et al. 2014)	-	68.9	-
Compact Bilinear Pooling (Gao et al. 2016)	84.0	76.2	-
ResNet-101 (He et al. 2016) †	72.5	78.0	77.9
Two-level attention (Xiao et al. 2015)	69.7	-	-
Spatial Transformer (Jaderberg et al. 2015)	84.1	-	-
MANTRA (Chapter 4)	-	76.6	-
Negative parts (Parizi, Andrea Vedaldi, et al. 2015)	-	77.1	-
MetaObject-CNN (Wu et al. 2015)	-	78.9	-
NAC (Simon et al. 2015)	81.0	-	-
GoogLeNet-GAP (Bolei Zhou, Khosla, et al. 2016)	63.0	66.6	-
WELDON (Chapter 5)	-	78.0	75.0
Part-Stacked CNN (Huang et al. 2016) ‡	76.6	-	-
SPLeaP (Kulkarni, Jurie, et al. 2016)	-	73.5	-
WILDCAT	**85.6**	**84.0**	**86.4**

Table 6.7.: Results on scene, action and fine-grained datasets. The performances on MIT67 and CUB-200 (resp. VOC 2012 Action) are evaluated with multi-class accuracy (resp. MAP). WILDCAT and state-of-the-art methods results are reported. Half at the top shows the performances using global image representation, whereas the half at the bottom shows performances for models based on regions selection. ‡ uses part-annotations during training.

note that our model is 9 pt better than Part-Stacked CNN (Huang et al. 2016), which uses bounding boxes and part annotations during training. This validates that our model can automatically find discriminative regions, even in the case of fine-grained classification.

6.3.3 Large-Scale Image Classification

We also evaluate WILDCAT on ILSVRC classification challenge (Russakovsky, Deng, et al. 2015) to show the scalability and the efficiency of our model for large-scale image classification. Table 6.8 summarizes the classification performances of WILDCAT and existing models. To have a fair comparison between models, we only report results for single model. For our model, we use a mono-scale model with an input image size 448×448, and $k^+ = k^- = 50$.

We can see that WILDCAT outperforms most of existing models trained using whole image (VGG16 (Simonyan et al. 2015), GoogleNet (Szegedy, Liu, et al. 2015), ResNet-152 (He et al. 2016)) and regions (RRSVM (Z. Wei et al. 2016), GoogleNet-GAP (Bolei Zhou, Khosla, et al. 2016), VGG16-GAP (Bolei Zhou, Khosla, et al. 2016)). Our model have similar performances that ResNeXt-101 (Xie et al. 2016), which proposes a new residual block with a multi-branch architecture. WILDCAT is slightly worse than Inception-ResNet-v2 (12

Method	Top-1 error	Top-5 error
VGG16 (144 crops) (Simonyan et al. 2015)	24.4	7.2
GoogleNet (144 crops) (Szegedy, Liu, et al. 2015)	-	7.89
ResNet-152 (10 crops) (He et al. 2016)	21.43	5.71
RRSVM (Z. Wei et al. 2016)	22.9	6.7
GoogleNet-GAP (Bolei Zhou, Khosla, et al. 2016)	35.0	13.2
VGG16-GAP (Bolei Zhou, Khosla, et al. 2016)	33.4	12.2
Inception-ResNet-v2 (12 crops) (Szegedy, Ioffe, et al. 2016)	**18.7**	**4.1**
ResNeXt-101 (1 crop) (Xie et al. 2016)	19.1	4.4
ResNet-101 † (1 crop)	22.44	6.21
ResNet-101 † (10 crops)	21.08	5.35
ResNet-152 † (10 crops)	20.69	5.21
ResNet-200 † (10 crops)	20.15	4.93
WILDCAT ($M = 1$)	19.21	4.23

Table 6.8.: Classification error on the ILSVRC validation set with single model. ResNet-101 is our initial model. † is the results of pre-trained model given at https://github.com/facebook/fb.resnet.torch.

crops) (Szegedy, Ioffe, et al. 2016) that combines both ResNet and Inception architectures. Better results can be obtained by learning ensemble of models.

We also reported results for different ResNets. The ResNet-101 is directly comparable to our model, because it corresponds to our initial model. It is important to note that with the same number of parameters and a very similar architecture, our model have a significant performance gain with respect to ResNet-101 (1 crop): +3.2 pt (resp. +2.0) on top-1 (resp. top-5) error. We also report the results with multi-crops post-processing, which a widely used post-processing to boost performances. Compared to our approach, multi-crops strategy extracts regions with a fixed grid, whereas our model automatically selects relevant regions. The important gain validates the relevance of our region selection approach. We also compare our model to deeper ResNet models: WILDCAT is +0.9 pt (resp. +0.7) better than the deeper model ResNet-200, which have about the double of parameters. We can also note that WILDCAT prediction is simple because it needs only 1 forward on the image to predict image label, whereas the multi-crops prediction needs several forwards on different image regions.

6.4 Weakly Supervised Experiments

In this section, we show that our model can be applied to various tasks, while being trained from global image labels only. We evaluate WILDCAT for two challenging weakly supervised applications: localization and segmentation. In this section, we use a mono-scale setup, with an input image size 448×448.

6.4.1 Weakly Supervised Localization

We evaluate the localization performances of our model on PASCAL VOC 2012 *validation* set (Everingham, Van Gool, et al. 2012) and MS COCO *validation* set (T.-Y. Lin, Maire, et al. 2014). The performances are evaluated with the point-based object localization metric introduced by (Oquab et al. 2015). This metric measures the quality of the detection, while being less sensitive to misalignments compared to other metrics such as Intersection-over-Union (IoU) (Everingham, Van Gool, et al. 2012), which requires the use of additional steps (e.g. bounding box regression). WILDCAT localization performances are reported in Table 6.9. Our model significantly outperforms existing weakly supervised methods. We can notice an important improvement between WILDCAT and MIL-based architecture Deep MIL (Oquab et al. 2015), which confirms the relevance of our spatial pooling function. In spite of its simple and multipurpose architecture, our model outperforms by a large margin the complex cascaded architecture of ProNet (C. Sun et al. 2016). It also outperforms the recent weakly supervised model (Bency et al. 2016) by 3.2 pt (resp. 4.2 pt) on VOC 2012 (resp. MS COCO), which use a more complex strategy than our model, based on search-trees to predict locations.

Method	VOC 2012	MS COCO
Deep MIL (Oquab et al. 2015)	74.5	41.2
ProNet (C. Sun et al. 2016)	77.7	46.4
WSLocalization (Bency et al. 2016)	79.7	49.2
WILDCAT	**82.9**	**53.4**

Table 6.9.: Object localization performances (MAP) on PASCAL VOC 2012 and MS COCO.

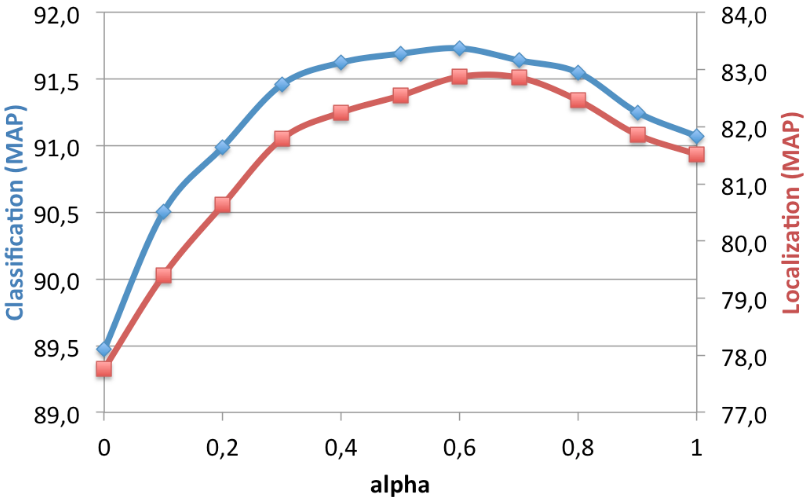


Figure 6.6.: Classification and localization performances with respect to α on VOC 2012.

Note that since the localization prediction is based on classification scores, good classification performance is important for robust object localization. In Figure 6.6, we evaluate the classification and localization performances with respect to α on VOC

2012. Both classification and localization curves are very similar. The best localization performances are obtained for $\alpha \in [0.6, 0.7]$, and the improvement between $\alpha = 1$ and $\alpha = 0.7$ is 1.6 pt. We can note that the worst performance is obtained for $\alpha = 0$, which confirms that the contextual information brought by the minimum is useful for both classification and localization.

6.4.2 Weakly Supervised Segmentation

We evaluate our model on the PASCAL VOC 2012 image segmentation dataset (Everingham, Van Gool, et al. 2012), consisting of 20 foreground object classes and one background class. We train our model with the *train* set (1,464 images) and the extra annotations provided by (Hariharan, Arbelaez, et al. 2011) (resulting in an augmented set of 10,582 images), and test it on the *validation* set (1,449 images). The performance is measured in terms of pixel Intersection-over-Union (IoU) averaged across the 21 categories. As in existing methods, we add a Fully Connected CRF (FC-CRF) (Krähenbühl et al. 2011) to post-process the final output labeling.

Segmentation results The result of our method is presented in Table 6.10. We compare it to weakly supervised methods that use only image labels during training. We can see that WILDCAT without FC-CRF outperforms existing weakly supervised models by a large margin. We note a large gain with respect to MIL models based on (soft-)max pooling (Pathak, Shelhamer, et al. 2015; Pedro O. Pinheiro and Collobert 2015), which validates the relevance of our pooling for segmentation. The improvement between WILDCAT with FC-CRF and the best model is 7.1 pt. This confirms the ability of our model to learn discriminative and accurately localized features. We can note that all the methods evaluated in Table 6.10 have comparable complexity.

Method	Mean IoU
MIL-FCN (Pathak, Shelhamer, et al. 2015)	24.9
MIL-Base+ILP+SP-sppxl (Pedro O. Pinheiro and Collobert 2015)	36.6
EM-Adapt + FC-CRF (Papandreou, L.-C. Chen, et al. 2015)	33.8
CCNN + FC-CRF (Pathak, Krahenbuhl, et al. 2015)	35.3
WILDCAT	39.2
WILDCAT + FC-CRF	**43.7**

Table 6.10.: Comparison of weakly supervised semantic segmentation methods on VOC 2012.

With a quite more complex strategy, the very recent paper (Kolesnikov et al. 2016) presents impressive results (50.7 MIoU). The training scheme in (Kolesnikov et al. 2016) incorporates different terms, which are specifically tailored to segmentation: one enforces the segmentation mask to match low-level image boundaries, another one incorporates prior knowledge to support predicted classes to occupy a certain image proportion.

In contrast, WILDCAT uses a single model which is trained in the same manner for the three tasks, i.e. classification, localization and segmentation. Our model could certainly benefit a lot from the specific terms used in (Kolesnikov et al. 2016) to further improve performances for the segmentation task.

Qualitative Results In Figure 6.7, we show predicted segmentation masks for four images. Compared to ground truth ((b) column), we can see that our predicted segmentation masks ((e) column) are always relevant, except for the last example where the rails and the train are glued together. The heatmaps from the same class (columns (c) and (d)) show different modalities learned by our model. When successful, they focus on different parts of the objects. For example, on the first row, the heatmap (c) focuses on the head of the bird whereas the heatmap (d) focuses on the legs and the tail.

6.5 Pooling Analysis

In this section, we analyze the spatial pooling. We first propose an unified pooling framework to compare standard WSL pooling functions. Then, this analysis is supplemented by a detailed experimental comparison on six datasets.

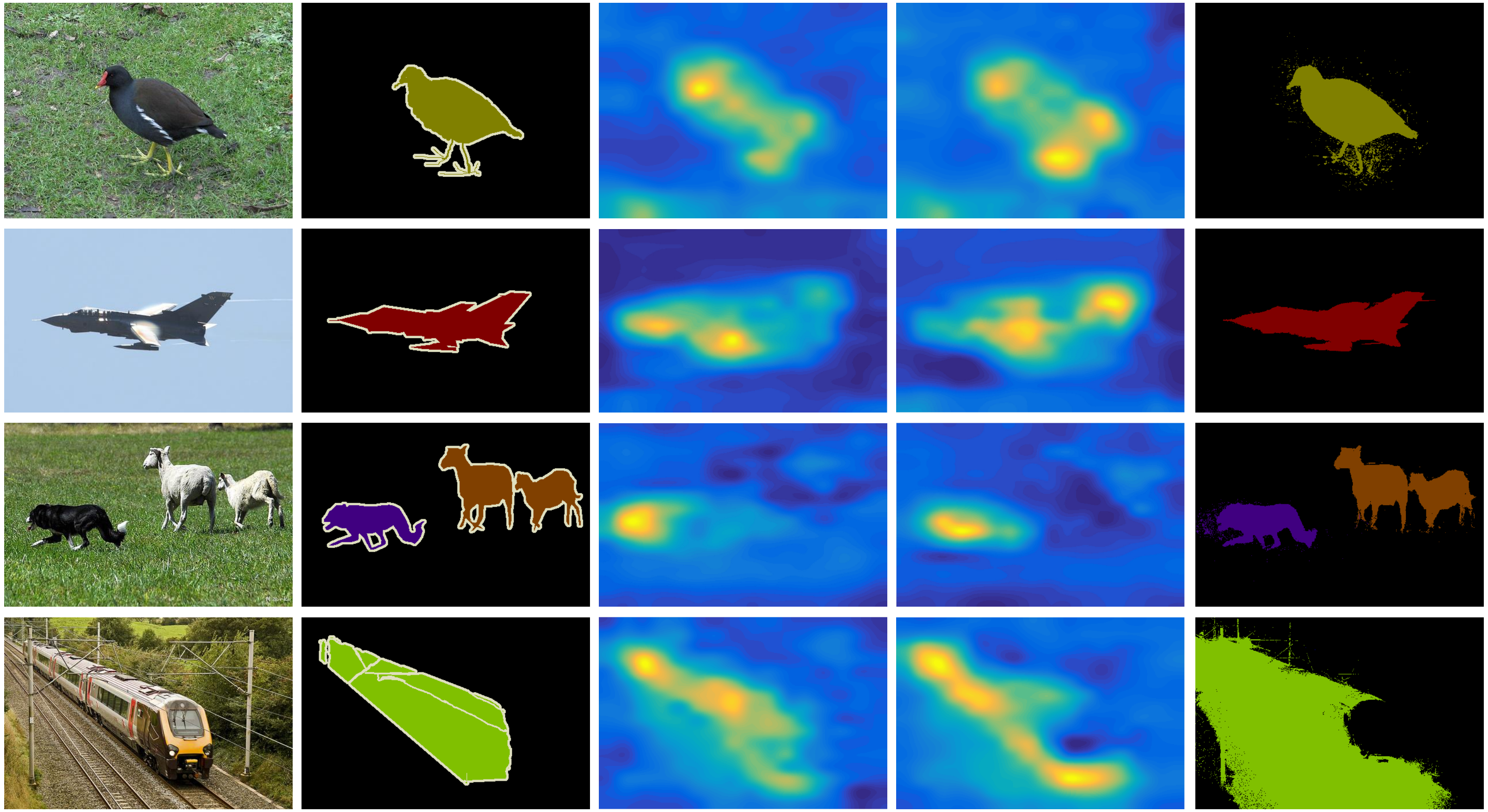
6.5.1 Generalized Pooling Model

To put into perspective connections between negative evidence and existing pooling strategy used in latent structured models, we introduce the following generalized scoring function, with ”inverse temperature” β_h^+ and β_h^- parameters smoothing between max, softmax and average:

$$\begin{aligned} s_{\boldsymbol{w}}^{(\alpha,\beta_h^+,\beta_h^-)}(\boldsymbol{x},\boldsymbol{y}) =& \frac{1}{2\beta_h^+} \log\left(\frac{1}{|\mathcal{H}|}\sum_{\boldsymbol{h}\in\mathcal{H}} \exp[\beta_h^+ F_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})]\right) \\ &+ \alpha \frac{1}{2\beta_h^-} \log\left(\frac{1}{|\mathcal{H}|}\sum_{\boldsymbol{h}\in\mathcal{H}} \exp[\beta_h^- F_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{h})]\right) \end{aligned} \tag{6.4}$$

As shown in Table 6.11, this generalized scoring function (Equation 6.4) includes several existing models as special cases. First, we fix $\alpha = 1$. When $\beta_h^+ = \beta_h^- \to +\infty$, it maximizes over latent variables and is equivalent to LSSVM (C.-N. Yu et al. 2009) or `max` pooling for deep ConvNets (Oquab et al. 2015). When $\beta_h^+ = \beta_h^- = 1$, is is equivalent HCRF (Quattoni and Torralba 2009) or MSSVM (Ping et al. 2014), which marginalize over latent variables. GAP (Bolei Zhou, Khosla, et al. 2016) ($\beta_h^+ = \beta_h^- \to 0$) also sum over latent variables, but unlike HCRF or MSSVM, all the latent variables have the same importance. The ϵ-framework (Schwing et al. 2012) / Log-Sum-Exp (LSE) pooling (Pedro O. Pinheiro and Collobert 2015; C. Sun et al. 2016) propose a trade-off between `max` and average. This pooling function is equivalent to the generalized pooling function when $\beta_h^+ = \beta_h^- \in [1, +\infty)$. We can also see that pooling function as a soft pooling function of top instance model (W. Li et al. 2015) and Learning with Label Proportion (LLP) (F. X. Yu et al. 2013).

To incorporate negative evidence in the generalized pooling function, β_h^- must be negative. When $\beta_h^+ \to +\infty$ and $\beta_h^- \to -\infty$, the generalized pooling function is equivalent



(a) original image (b) ground truth (c) heatmap1 (d) heatmap2 (e) WILDCAT prediction

Figure 6.7.: Segmentation examples on VOC 2012. Our prediction is correct except for the train (last row) where our model aggregated rails and train regions. For objects as *bird* or *plane*, one can see how two heatmaps (heatmap1 (c) and heatmap2 (d) representing the same class: respectively *bird*, *aeroplane*, *dog* and *train*) succeed to focus on different but relevant parts of the objects.

to MANTRA model (Chapter 4) and pools over both maximum and minimum scores. We can derive a soft version of WELDON pooling function when $\beta_h^+ \in [0, +\infty)$ and $\beta_h^- \in (-\infty, 0]$. The generalized pooling function is not exactly equivalent to WELDON (Chapter 5), but it has a similar behavior. Similarly, the generalized pooling is a soft version of WILDCAT when $\alpha \in [0, 1]$.

Model	α	β_h^+	β_h^-
HCRF / MSSVM	1	1	1
GAP	1	→ 0	→ 0
LSSVM / `max`	1	+∞	+∞
ϵ-framework / LSE / LLP⋆ / top instances⋆	1	$\beta_h^+ = \beta_h^- \in [1, +\infty)$	
MANTRA	1	+∞	-∞
WELDON⋆	1	$[0, +\infty)$	$(-\infty, 0]$
WILDCAT⋆	[0, 1]	$[0, +\infty)$	$(-\infty, 0]$

Table 6.11.: Model comparison with corresponding parameters. ⋆ indicates that the model is not exactly equivalent to the generalized pooling function with the given parameters, but has a similar behavior.

In the next section, we provide a systematic comparison of these pooling functions to highlight their strengths and weaknesses in different contexts.

6.5.2 Pooling Analysis Experiments

In this section, we analysis our pooling function, and we compare it with standard pooling functions presented in Section 6.5. We report results for an input image of size 448×448, but similar behaviors are observed for other scales. To have fair comparison, all the experiments uses the same network (ResNet-101). We analyze the impact of the number of selected instances. We show in Figure 6.8 the performance with respect to the proportion of selected regions. We can note that the GAP (resp. MANTRA pooling) is a special case of WELDON pooling, when the proportion of selected regions is 1 (resp. →0).

Firstly, we can see that negative evidence is important, because on all dataset, MANTRA is better than `max` pooling. In particular, we observe a large improvement of +4.5 pt on VOC 2012 Action dataset, where the context plays an important role (Georgia Gkioxari et al. 2015). We can also see that region selection is important: on all dataset except MIT67, the WELDON pooling with a proportion of selected instances in [0.2, 0.8] is equal or better than GAP. The WELDON pooling has similar or better results than GAP by using only 20% of regions. Using more regions (50 %) gives better results than GAP: +0.4 pt on VOC 2007, +0.3 pt on VOC 2012, +0.6 pt on VOC 2012 Action, +1.9 pt and +2.1 pt on CUB-200. On MIT67, we can see that using a large number of regions is better (≥ 80 %). This shows that a large number of regions are discriminants.

We also compare our pooling function to `max` and LSE pooling functions. The LSE pooling is a soft extension of `max` pooling. On all datasets, LSE is better than `max` pooling: +6 pt on CUB-200 and +5.5 on MIT67. This shows that using several regions is more

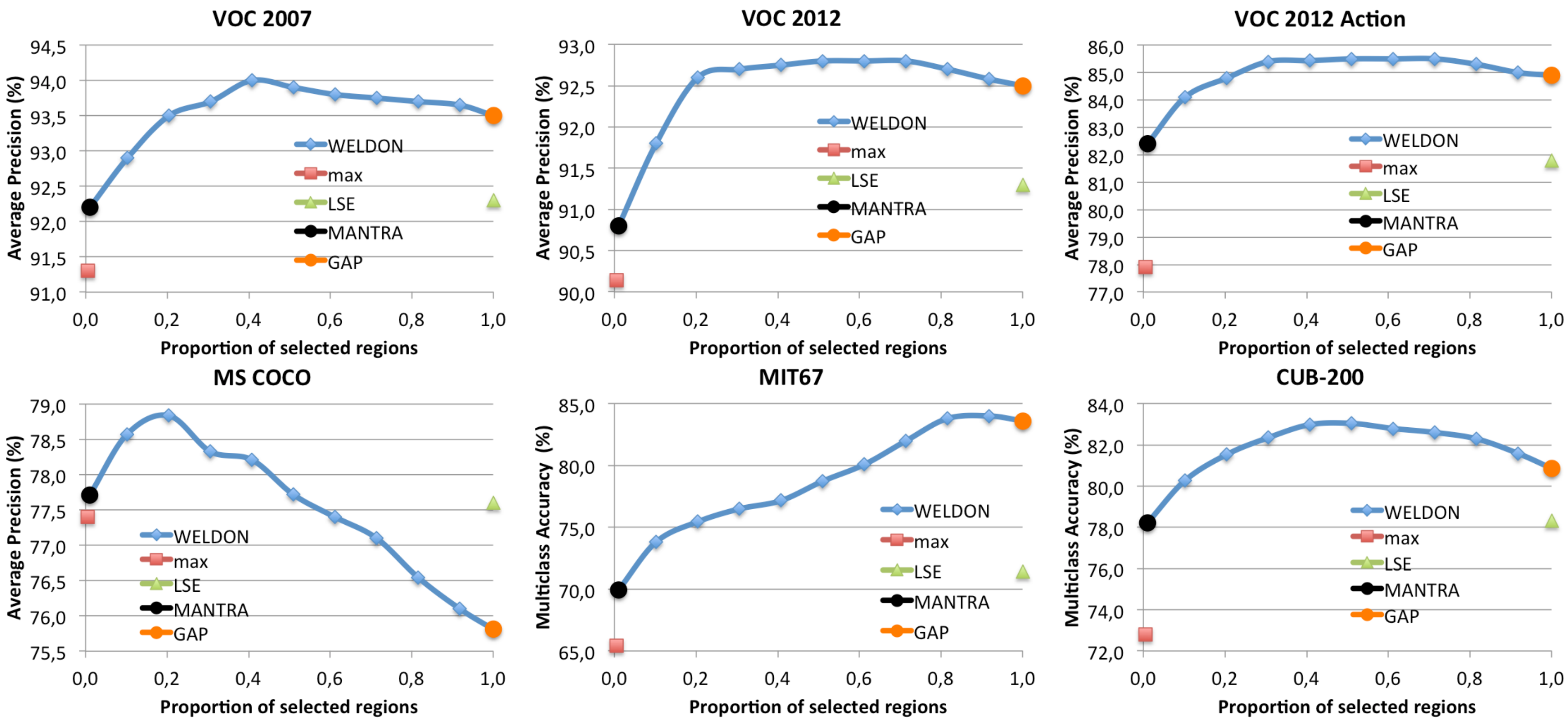


Figure 6.8.: Pooling analysis. We compare different spatial pooling strategies on 6 datasets for an input image of size 448 × 448. The x-axis shows the proportion of selected regions and the y-axis the performance. We can see that MANTRA always outperforms the max pooling, which validates the relevance of negative evidence. We can also see that our spatial outperforms equally or better than GAP with only 20 % of regions.

robust than using only the best region. We also note that LSE pooling performances are closed to MANTRA pooling performances. MANTRA pooling, which used only 2 regions, is as efficient as the LSE pooling which used all regions. Except on MS COCO, we observe that the GAP is better than LSE which is better than `max`: using more regions increase the robustness. On MS COCO, we see that the region selection is important because a lot of objects have small sizes: the gain is +3 pt between WELDON with 20 % of regions and GAP.

6.6 Conclusion

In this chapter, we proposed WILDCAT, a new WSL model dedicated to learning discriminative localized visual features by using only image-level labels during training. WILDCAT extends WELDON at three major levels: the use of FCN for maintaining spatial resolution, the explicit design in the network of local features related to different class modalities, and a new way to pool these features to provide a global image prediction required for weakly supervised training. We also introduced a unified framework for pooling, which generalizes standard WSL models as special cases, including our model.

Extensive experiments showed the effectiveness of WILDCAT on image classification, for which we report outstanding performances on six challenging datasets. We also showed that our framework can be used to improve the performance of state-of-the-art deep models for large-scale image classification on ImageNet. We showed that our model can be efficiently applied to weakly supervised applications (pointwise localization and segmentation), while being trained from global image labels only. Finally, the spatial pooling analysis showed that the number of regions is crucial for good performances. We observed that for scene datasets we need a large number of regions whereas on datasets with small objects we need a small number of regions. Our experiments on ImageNet also showed that fine-tuning a pre-trained network with our WSL architecture leads to a significant performance gain. It would be instesting to learn our WSL network on large-scale datasets from scratch, e.g. on ImageNet or Places2.

Chapter 7

CONCLUSION

7.1 Summary of Contributions

In this thesis, we proposed several models to learn localized representations with image-level labels only. Our contributions can be summarized in three points.

Pooling The key issue of WSL is to find how to pool the region scores into image scores. To address this problem, we introduced the `max+min` pooling function. We first applied this pooling function on bag classification with binary labels (Chapter 3). We then generalized it for structured output prediction (Chapter 4). We showed that the minimum regions bring complementary information to max regions and explicitly model negative evidence, i.e. look for "counter evidence" for the class. To have a more robust prediction, we extended this pooling function by selecting several regions (Chapter 5). As both maximum and minimum scoring regions are important for good results, but do not bring the same kind of information, we introduced a trade-off parameter to weight the importance between both terms (Chapter 6). Finally, we performed an empirical comparison of different pooling functions used for WSL on several datasets.

Optimization Another challenging problem in machine learning is how to efficiently learn the parameters of the model. This usually reduces to solving an optimization problem. For each of our models presented in this thesis, we proposed an objective function and a solver to optimize it. For SyMIL model (Chapter 3), we provided a primal solver that scales to large datasets and a dual solver to learn non-linear models. In Chapter 4, we proposed an efficient non convex cutting plane algorithm based on the one-slack formulation to train the MANTRA model. To learn deep architectures (Chapters 5 and 6), we used the Stochastic Gradient Descent (SGD). We also need an efficient optimization method to deal with structured output prediction, because we have to solve the (loss-augmented) inference problem. We tackled the Average Precision (AP) ranking problem because it is a standard evaluation measure, and we proposed a new AP ranking instantiation (Chapter 4) and its extension (Chapter 5), for which efficient solutions are introduced to solve the inference problems.

Deep architecture We proposed several deep ConvNet architectures to learn localized representations with image-level labels only. In Chapter 5, we introduced the WELDON architecture, which is composed of two sub-networks: a deep feature extraction network based on VGG16 and a prediction network based on a global pooling function to automatically select relevant regions from images annotated with image labels. The whole architecture is carefully designed for fast processing by sharing region feature computations and allows end-to-end training. In Chapter 6, we extended WELDON by introducing WILDCAT, a new architecture based on a new multi-map transfer layer, which automatically learns several class specific modalities. Unlike WELDON, this architecture is based on the Fully Convolutional Network (FCN) ResNet-101, which allows to learn feature maps with more accurate resolution than WELDON. We also showed that this architecture is multipurpose and can be efficiently used for image classification as well as weakly supervised pointwise object localization and segmentation.

7.2 Future Work

7.2.1 Pooling for WSL

The analysis of pooling functions shows that the number of maps per class and the number of selected regions are important hyper-parameters. We think that it would be interesting to learn these hyper-parameters.

In this manuscript, we showed that using several maps per class to learn multiple class-related modalities is an interesting approach. In our experiments, the number of maps is fixed for all the classes, but different classes have different complexities and therefore may not be well modeled by a fixed number of parts. Indeed, learning several modalities is useful for objects that can be represented as a collection of parts, e.g. a car has wheels, doors, windows, mirrors, license plates. On the contrary, using several modalities on objects without part (e.g. TV monitor) is not efficient, because there is only one modality. We think that learning a specific number of maps for each class is a promising research direction to improve classification performances. A method to achieve this is to use a ℓ_1-normalized weighted average of the maps to keep only the relevant maps for each class (Kulkarni, Zepeda, et al. 2015).

Other important hyper-parameters are the numbers of selected regions k^+ & k^- in the spatial pooling. Our experiments showed that the optimal number of regions depends on the task. For scene datasets we need a lot of regions because the context information is relevant. On the contrary for object datasets with complex scenes and small objects we need few regions to select relevant regions only. Another limitation is that the numbers of selected regions k^+ & k^- are fixed for all classes. It would be more interesting to learn different numbers of regions per class, because some object classes are naturally bigger (e.g. aeroplane $>$ person $>$ bottle), and the context can be very important for some classes.

7.2.2 Deep learning for complex images

Spatial resolution of the detection maps Our models use the outputs of the last convolution layers as feature representations, but the spatial information in these layers is too coarse to allow precise localization. However, having feature maps with high spatial resolution is crucial to deal with complex images, because the objects can be at different locations and can be very small. For example on MS COCO, 41% of the objects are small, i.e. their areas are less than 0.4% of the whole image area. To deal with small objects, we need dense feature maps. A solution to generate both high-resolution and semantic feature maps is to combine predictions from different layers in a ConvNet, e.g. Hypercolumns (Hariharan, Arbeláez, et al. 2015), Inside-Outside Net (Bell et al. 2016), Feature Pyramid Network (FPN) (T.-Y. Lin, Dollár, et al. 2017). Another strategy is to use dilated convolutions to reduce the number of spatial pooling layers in the network (L. Chen et al. 2015). It would also be interesting to use multi-scale feature maps, to detect objects at different scales.

Applications to WSL tasks We showed that our model can be successfully used for weakly supervised pointwise object localization. However, it cannot be directly applied to predict object bounding boxes, because the sizes of the regions are implicitly fixed by the network architecture. This architecture enables fast feature region computation, but does not allow to deal with arbitrary region sizes. To be effective on detection task, we need regions with different sizes and aspect ratios. A potential future direction would be to study how we can adapt popular detection models to our task. We also showed that our generic architecture has competitive results on weakly supervised semantic segmentation task. On the contrary, the state-of-the-art method uses a more complex strategy, specifically tailored to segmentation, e.g. a loss to constrain the segmentations to match with object boundaries. To improve the segmentation performances, our model could certainly benefit a lot from additional specific segmentation terms. It would also be interesting to evaluate our model on other WSL applications, e.g. pose estimation, key point detection, instance segmentation, dense captioning (Johnson et al. 2016) and visual grounding (Fukui et al. 2016).

Deep Structured ConvNets Real world images generally contain multiple labels, which could correspond to different objects, scenes, actions or attributes in an image. To address this problem, our models learn independent classifiers for all categories, but they fail to explicitly exploit the label dependencies in images. Modeling the rich semantic information and their dependencies is essential for complex scene understanding. To exploit the correlations between labels, it would be interesting to combine our approach with structured models, e.g. Structured Prediction Energy Network (SPEN) (Belanger et al. 2016), Architectures Deep In Output Space (ADIOS) (Cisse et al. 2016). Another complementary direction to analyze complex image is to model the interactions and relative spatial relations between the entities (i.e. persons, objects) in a scene with a graphical model. We think that models reasoning over structures can bring benefits, allowing the classification of higher-level concepts built from recognition over lower-level entities.

Appendix A

SYMIL: COMBINATION WITH LABEL PROPORTION

A limitation of MIL methods and SyMIL relies on the prediction function which selects a single instance, making the method sensitive to false alarm outliers. Recently, the Learning with Label Proportion (LLP) framework (Subsection 2.2.1.2) has been proposed to address this problem. Unlike MIL, in LLP the bags are labeled with the proportion of positive instances. LLP methods are appealing for solving MIL problems, because they limit the impact of instance outliers, improving robustness. In this section, we combine the local information in SyMIL and the global information from the LLP framework to learn a prediction function more robust to outliers. For a given bag $\boldsymbol{x}_i$ and weight vector $\boldsymbol{w}$, the true label proportion of positive instances is:

$$p(\boldsymbol{x}_i;\boldsymbol{w}) = \frac{|\{h|h \in \mathcal{H},\ \text{sign}[\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i,h)\rangle] = 1\}|}{|\boldsymbol{x}_i|} = \frac{\sum_{h\in\mathcal{H}} \mathbb{1}[\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i,h)\rangle]}{|\boldsymbol{x}_i|} \tag{A.1}$$

where $\mathbb{1}$ is the Heavyside function. It is not possible to directly optimize over the Heavyside function with gradient-based approaches. Therefore, we approximate it by a sigmoid function $f_\rho(x) = \frac{1}{1+\exp(-\rho x)}$. We define the estimated proportion of $\oplus$ instances in a bag $\boldsymbol{x}_i$ as:

$$\tilde{p}(\boldsymbol{x}_i;\boldsymbol{w}) = \frac{\sum_{h\in\mathcal{H}} f_\rho(\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i,h)\rangle)}{|\boldsymbol{x}_i|} = \frac{1}{|\boldsymbol{x}_i|}\sum_{h\in\mathcal{H}} \frac{1}{1+\exp(-\rho\langle \boldsymbol{w}, \Phi(\boldsymbol{x}_i,h)\rangle)} \tag{A.2}$$

Note that our approximation of $\tilde{p}$ is more accurate than the one used in InvCal (Rueping 2010), where the sigmoid is applied to the mean score of each bag.

In standard LLP contexts, the goal is to match the estimated proportion of $\oplus$ instances $\tilde{p}(\boldsymbol{x}_i)$ to the known proportion coming from ground truth annotations $p_i^\star$. Therefore, a regression loss is generally used during training, e.g. using the ℓ_1 norm $|\tilde{p}(\boldsymbol{x}_i) - p_i^\star|$ (Rueping 2010; F. X. Yu et al. 2013; Lai et al. 2014). When applied to MIL, such a regression loss is however questionable, since there is no clear definition of $p_i^\star$ for each bag. Instead, we propose to enforce the proportion of $\oplus$ instances to be larger (resp. smaller) than a target proportion $p_i^\star$ for a positive (resp. negative) bag. To this end, we introduce the following "hinge-like" loss:

$$\mathcal{L}_p(\tilde{p}(\boldsymbol{x}_i;\boldsymbol{w}), p_i^\star) = \begin{cases} |\delta_p^i| & \text{if } \left[y_i^\star \cdot \text{sign}(\delta_p^i)\right] < 0 \\ 0 & \text{otherwise} \end{cases} \tag{A.3}$$

where $\delta_p^i = \tilde{p}(\boldsymbol{x}_i) - p_i^\star$. The loss $\mathcal{L}_p(\tilde{p}(\boldsymbol{x}_i;\boldsymbol{w}), p_i^*)$ penalizes the positive (resp. negative) bags which have a proportion of $\oplus$ instances inferior (resp. superior) to the target proportion p_i^*. To use label proportion during learning, we add the label proportion term $\mathcal{L}_p$ to the SyMIL objective function (Equation 3.7). The new objective function is:

$$\mathcal{P}(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\mathcal{L}(\boldsymbol{w}, \mathcal{D}) + \frac{C_p}{N}\sum_{i\in\mathcal{D}} \mathcal{L}_p(\tilde{p}(\boldsymbol{x}_i; \boldsymbol{w}), p_i^*) \tag{A.4}$$

The optimization problem (Equation A.4) is solved with a Stochastic Gradient Descent (SGD) strategy (Léon Bottou 2010).

Results We perform experiments on the MIL datasets presented in Subsection 3.4.2. The target proportion $p_i^\star$ is set to 0.8 (resp. 0.2) for positive (resp. negative) bags, and ρ (sigmoid parameter) is cross-validated on the training set. Results are shown in Table A.1 & Table A.2. As we can notice, the proposed label proportion is itself competitive, since it outperforms baselines mi-SVM / MI-SVM / LSVM in all datasets (see Table 3.4), but remains below SyMIL performances. The combination SyMIL+label proportion is able to further improve SyMIL performances, especially on the image and molecule datasets. This experiment validates the complementarity potential between the two approaches.

Dataset	Elephant	Fox	Tiger	Avg.	Musk1	Musk2	Avg.
label proportion	82.4	62.5	82.1	75.7	83.5	85.6	84.6
SyMIL	87.2	64.9	85.3	79.1	88.5	87.8	88.2
SyMIL+label prop.	**87.9**	**65.7**	**86.3**	**80.0**	**89.3**	**89.3**	**89.3**

Table A.1.: Results on image and molecule datasets for SyMIL and SyMIL+label proportion

Dataset	TST1	TST2	TST3	TST4	TST7	TST9	TST10	Avg.
label proportion	93.8	81.1	86.1	86.0	81.7	68.9	83.9	83.0
SyMIL	94.3	84.3	**88.8**	87.1	82.3	71.2	85.8	84.8
SyMIL+label prop.	**94.8**	**84.9**	88.7	**87.5**	**82.6**	**71.5**	**86.0**	**85.1**

Table A.2.: Results on Text datasets for SyMIL and SyMIL+label proportion

Appendix B

MANTRA: 1-SLACK DUAL FORMULATION

First, we write the Lagrangian of primal formulation (Equation 4.10):

$$\begin{aligned} L(\boldsymbol{w},\xi,\alpha) =& \frac{1}{2}\|\boldsymbol{w}\|^2 + C\xi - \alpha'\xi \\ &- \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \left(\xi - \frac{1}{N}\sum_{i=1}^{N} \Delta(\boldsymbol{y}_i,\hat{\boldsymbol{y}}_i) + s_{\boldsymbol{w}}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}_i) - s_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}_i)\right) \end{aligned} \tag{B.1}$$

where $\alpha' \geq 0$ and $\forall \bar{\boldsymbol{y}} = (\hat{\boldsymbol{y}}_1,\ldots,\hat{\boldsymbol{y}}_N) \in \mathcal{Y}^N,\ \alpha_{\bar{\boldsymbol{y}}} \geq 0$. Then, we differentiate the constraints with respect to the primal variables:

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w},\xi,\alpha) = \boldsymbol{w} + \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \left(\nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}_i)\right) = 0 \tag{B.2}$$

The equation of $\boldsymbol{w}$ with dual variables is:

$$\boldsymbol{w} = -\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i,\hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i,\boldsymbol{y}_i) \tag{B.3}$$

$$\frac{\partial L(\boldsymbol{w},\xi,\alpha)}{\partial\xi} = C - \alpha' - \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} = 0 \tag{B.4}$$

This differentiation gives a condition on the sum of dual variables:

$$0 \leq \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \leq C \tag{B.5}$$

Dual formulation Applying the Equation B.3,Equation B.4, in the Lagrangian (Equation B.1), the dual formulation of the optimization problem (Equation 4.10) is

$$
\begin{aligned}
\mathcal{D}(\boldsymbol{\alpha}) =& \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \Delta(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i) + s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i) && \text{(B.6)}\\
=& \frac{1}{2}\left\langle \boldsymbol{w}, -\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle && \text{(B.7)}\\
& + \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \Delta(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i) + s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i) \\
=& -\frac{1}{2}\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \langle \boldsymbol{w}, \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i)\rangle && \text{(B.8)}\\
& + \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} (\Delta(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i) + \langle \boldsymbol{w}, \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i)\rangle) \\
=& \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \Delta(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i) && \text{(B.9)}\\
& + \frac{1}{2}\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \langle \boldsymbol{w}, \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i)\rangle \\
=& \sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \Delta(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i) && \text{(B.10)}\\
& - \frac{1}{2}\left\langle \boldsymbol{w}, -\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^N} \alpha_{\bar{\boldsymbol{y}}} \frac{1}{N}\sum_{i=1}^{N} \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle
\end{aligned}
$$

The Eq. (B.10) can be rewritten in the standard formulation

$$
\mathcal{D}(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \boldsymbol{c} - \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{H} \boldsymbol{\alpha} \tag{B.11}
$$

where $\boldsymbol{H}$ is a square matrix and $\forall \bar{\boldsymbol{y}}, \bar{\boldsymbol{y}}' \in \mathcal{Y}^N \quad H_{\bar{\boldsymbol{y}}\bar{\boldsymbol{y}}'} = \langle \boldsymbol{g}_{\bar{\boldsymbol{y}}}, \boldsymbol{g}_{\bar{\boldsymbol{y}}'}\rangle$ with

$$
\boldsymbol{g}_{\bar{\boldsymbol{y}}} = -\frac{1}{N}\sum_{i=1}^{N} \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i) - \nabla_{\boldsymbol{w}} s_{\boldsymbol{w}}(\boldsymbol{x}_i, \boldsymbol{y}_i) \tag{B.12}
$$

and $c_{\bar{\boldsymbol{y}}} = \frac{1}{N}\sum_{i=1}^{N} \Delta(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)$

We solve the QP problem, with an interior-point optimizer, as in (C.-N. Yu et al. 2009).

BIBLIOGRAPHY

Andrews, Stuart, Ioannis Tsochantaridis, and Thomas Hofmann (2003). "Support Vector Machines for Multiple-Instance Learning". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 19, 20, 33, 45).

Arandjelović, R., P. Gronat, A. Torii, T. Pajdla, and J. Sivic (2016). "NetVLAD: CNN architecture for weakly supervised place recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 16).

Avila, Sandra, Nicolas Thome, Matthieu Cord, Eduardo Valle, and Arnaldo Araujo (2012). "Pooling in Image Representation: the Visual Codeword Point of View". In: *Computer Vision and Image Understanding (CVIU)* (cit. on p. 10).

Azizpour, Hossein, Mostafa Arefiyan, Sobhan Naderi Parizi, and Stefan Carlsson (2015). "Spotlight the Negatives: A Generalized Discriminative Latent Model". In: *British Machine Vision Conference (BMVC)* (cit. on p. 20).

Azizpour, Hossein, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson (2016). "Factors of Transferability for a Generic ConvNet Representation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on pp. 15, 16).

Baeza-Yates, Ricardo A. and Berthier Ribeiro-Neto (1999). *Modern Information Retrieval* (cit. on p. 58).

Bartlett, Peter L. and Shahar Mendelson (2003). "Rademacher and Gaussian Complexities: Risk Bounds and Structural Results". In: *Journal of Machine Learning Research (JMLR)* (cit. on p. 36).

Bearman, Amy, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei (2016). "What's the Point: Semantic Segmentation with Point Supervision". In: *European Conference on Computer Vision (ECCV)* (cit. on p. 6).

Behl, Aseem, Pritish Mohapatra, C. V. Jawahar, and M. Pawan Kumar (2015). "Optimizing Average Precision Using Weakly Supervised Data". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on pp. 57, 58, 69, 70, 79).

Belanger, David and Andrew McCallum (2016). "Structured Prediction Energy Networks". In: *International Conference on Machine Learning (ICML)* (cit. on p. 111).

Bell, Sean, C. Lawrence Zitnick, Kavita Bala, and Ross Girshick (2016). "Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 111).

Bency, Archith J., Heesung Kwon, Hyungtae Lee, S. Karthikeyan, and B. S. Manjunath (2016). "Weakly Supervised Localization using Deep Feature Maps". In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 95, 102).

Bengio, Y., P. Simard, and P. Frasconi (1994). "Learning Long-term Dependencies with Gradient Descent is Difficult". In: *IEEE Transactions on Neural Networks* (cit. on p. 14).

Bilen, H., V.P. Namboodiri, and L.J. Van Gool (2013). "Object Classification with Latent Window Parameters". In: *International Journal of Computer Vision (IJCV)* (cit. on p. 22).

Boser, Bernhard E., Isabelle M. Guyon, and Vladimir N. Vapnik (1992). "A Training Algorithm for Optimal Margin Classifiers". In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory* (cit. on p. 11).

Bossard, Lukas, Matthieu Guillaumin, and L.J. Van Gool (2014). "Food-101 – Mining Discriminative Components with Random Forests". In: *European Conference on Computer Vision (ECCV)* (cit. on p. 68).

Bottou, Léon (2010). "Large-Scale Machine Learning with Stochastic Gradient Descent". In: *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT)* (cit. on pp. 38, 114).

Boureau, Y-Lan, Jean Ponce, and Yann LeCun (2010). "A theoretical analysis of feature pooling in vision algorithms". In: *International Conference on Machine Learning (ICML)* (cit. on p. 10).

Breiman, Leo, Jerome Friedman, Charles J Stone, and Richard A Olshen (1984). *Classification and regression trees*. CRC press (cit. on p. 11).

Bunescu, Razvan C. and Raymond J. Mooney (2007). "Multiple Instance Learning for Sparse Positive Bags". In: *International Conference on Machine Learning (ICML)* (cit. on p. 19).

Chatfield, Ken., K. Simonyan, Andrea Vedaldi, and Andrew. Zisserman (2014). "Return of the Devil in the Details: Delving Deep into Convolutional Nets". In: *British Machine Vision Conference (BMVC)* (cit. on pp. 13, 15, 16, 70, 71, 85, 99).

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille (2015). "Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs". In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 16, 95, 111).

Chéron, Guilhem, Ivan Laptev, and Cordelia Schmid (2015). "P-CNN: Pose-based CNN Features for Action Recognition". In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 16).

Cinbis, Ramazan Gokberk, Jakob Verbeek, and Cordelia Schmid (2016). "Weakly Supervised Object Localization with Multi-fold Multiple Instance Learning". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on p. 19).

Cisse, Moustapha, Maruan Al-Shedivat, and Samy Bengio (2016). "ADIOS: Architectures Deep In Output Space". In: *International Conference on Machine Learning (ICML)* (cit. on p. 111).

Cover, T. and P. Hart (1967). "Nearest Neighbor Pattern Classification". In: *IEEE Transactions on Information Theory* (cit. on p. 10).

Dai, Jifeng, Kaiming He, Yi Li, Shaoqing Ren, and Jian Sun (2016). "Instance-sensitive fully convolutional networks". In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 16, 91, 92).

Dai, Jifeng, Yi Li, Kaiming He, and Jian Sun (2016). "R-FCN: Object Detection via Region-based Fully Convolutional Networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 16, 91–93).

Dalal, Navneet and Bill Triggs (2005). "Histograms of Oriented Gradients for Human Detection". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 42).

Deselaers, Thomas and Vittorio Ferrari (2010). “A Conditional Random Field for Multiple-Instance Learning”. In: *International Conference on Machine Learning (ICML)* (cit. on pp. 19, 45, 46).

Diba, Ali, Ali Mohammad Pazandeh, Hamed Pirsiavash, and Luc Van Gool (2016). “DeepCAMP: Deep Convolutional Action & Attribute Mid-Level Patterns”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 16).

Diba, Ali, Vivek Sharma, Ali Pazandeh, Hamed Pirsiavash, and Luc Van Gool (2016). “Weakly Supervised Cascaded Convolutional Networks”. In: *arXiv:1611.08258* (cit. on p. 27).

Dietterich, Thomas G., Richard H. Lathrop, and Tomás Lozano-Pérez (1997). “Solving the Multiple Instance Problem with Axis-parallel Rectangles”. In: *Artif. Intell.* (Cit. on p. 18).

Do, Trinh-Minh-Tri and Thierry Artières (2012). “Regularized Bundle Methods for Convex and Non-convex Risks”. In: *Journal of Machine Learning Research (JMLR)* (cit. on p. 55).

Doersch, Carl, Abhinav Gupta, and Alexei A. Efros (2013). “Mid-Level Visual Element Discovery as Discriminative Mode Seeking”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 68).

Dong, Lin (2006). “A comparison of multi-instance learning algorithms”. PhD thesis. The University of Waikato (cit. on p. 18).

Dosovitskiy, Alexey, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox (2015). “Flownet: Learning optical flow with convolutional networks”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 16).

Duchi, John C., Elad Hazan, and Yoram Singer (2011). “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization”. In: *Journal of Machine Learning Research (JMLR)* (cit. on p. 13).

Durand*, Thibaut, Taylor Mordan*, Nicolas Thome, and Matthieu Cord (2017). “WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. vii, 7, 90).

Durand, Thibaut, David Picard, Nicolas Thome, and Matthieu Cord (2014). “Semantic Pooling for Image Categorization using Multiple Kernel Learning”. In: *IEEE International Conference on Image Processing (ICIP)* (cit. on pp. vii, 4, 5).

Durand, Thibaut, Nicolas Thome, and Matthieu Cord (2015). “MANTRA: Minimum Maximum Latent Structural SVM for Image Classification and Ranking”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. vii, 7, 51).

Durand, Thibaut, Nicolas Thome, and Matthieu Cord (2016). “WELDON: Weakly Supervised Learning of Deep Convolutional Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. vii, 7, 73).

Durand, Thibaut, Nicolas Thome, and Matthieu Cord (2017a). “Negative Evidence for Weakly Supervised Learning of Deep Structured Models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence [Submission]* (cit. on p. vii).

Durand, Thibaut, Nicolas Thome, and Matthieu Cord (2017b). “SyMIL: MinMax Latent SVM for Weakly Labeled Data”. In: *IEEE Transactions on Neural Networks and Learning Systems (TNNLS) [Submission]* (cit. on pp. vii, 31).

Durand, Thibaut, Nicolas Thome, Matthieu Cord, and Sandra Avila (2013). "Image Classification using Object Detectors". In: *IEEE International Conference on Image Processing (ICIP)* (cit. on pp. vii, 4).

Durand, Thibaut, Nicolas Thome, Matthieu Cord, and David Picard (2014). "Incremental Learning of Latent Structural SVM for Weakly Supervised Image Classification". In: *IEEE International Conference on Image Processing (ICIP)* (cit. on p. vii).

Everingham, M., S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2015). "The Pascal Visual Object Classes Challenge: A Retrospective". In: *International Journal of Computer Vision (IJCV)* (cit. on p. 57).

Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2007). *The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results*. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html (cit. on pp. 27, 48).

Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (2012). *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html (cit. on pp. 27, 28, 102, 103).

Felzenszwalb, Pedro F, Ross B Girshick, David McAllester, and Deva Ramanan (2010). "Object detection with discriminatively trained part-based models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on pp. 19, 33, 45, 47, 48, 62, 92).

Fournier, J., M. Cord, and S. Philipp-Foliguet (2001). "RETIN: A Content-Based Image Indexing and Retrieval System". In: *Pattern Analysis and Applications Journal, Special issue on image indexation* (cit. on p. 10).

Fukui, Akira, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach (2016). "Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding". In: *CoRR* abs/1606.01847. URL: http://arxiv.org/abs/1606.01847 (cit. on pp. 16, 111).

Fukushima, Kunihiko (1980). "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". In: *Biological Cybernetics* (cit. on p. 2).

Gao, Yang, Oscar Beijbom, Ning Zhang, and Trevor Darrell (2016). "Compact Bilinear Pooling". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 99, 100).

Gärtner, Thomas, Peter A. Flach, Adam Kowalczyk, and Alex J. Smola (2002). "Multi-Instance Kernels". In: *International Conference on Machine Learning (ICML)* (cit. on p. 18).

Gehler, Peter and Olivier Chapelle (2007). "Deterministic Annealing for Multiple-Instance Learning". In: *International Conference on Artificial Intelligence and Statistics (AISTAT)* (cit. on pp. 19, 45).

Gemert, J. C. van, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek (2010). "Visual Word Ambiguity". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on p. 10).

Girshick, R., J. Donahue, T. Darrell, and J. Malik (2014). "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 17, 75).

Girshick, Ross (2015). “Fast R-CNN”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 16, 75).

Girshick, Ross, Forrest Iandola, Trevor Darrell, and Jitendra Malik (2015). “Deformable Part Models are Convolutional Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 92).

Gkioxari, G., A. Toshev, and N. Jaitly (2016). “Chained Predictions Using Convolutional Neural Networks”. In: (cit. on p. 16).

Gkioxari, Georgia, Ross Girshick, and Jitendra Malik (2015). “Contextual action recognition with R* CNN”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 106).

Glorot, Xavier and Yoshua Bengio (2010). “Understanding the difficulty of training deep feedforward neural networks”. In: *International Conference on Artificial Intelligence and Statistics (AISTAT)* (cit. on p. 14).

Gong, Yunchao, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik (2014). “Multi-scale Orderless Pooling of Deep Convolutional Activation Features”. In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 16, 68, 69, 86, 99, 100).

Hajimirsadeghi, Hossein and Greg Mori (2016). “Multi-Instance Classification by Max-Margin Training of Cardinality-Based Markov Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on p. 20).

Hariharan, Bharath, Pablo Arbelaez, Lubomir D. Bourdev, Subhransu Maji, and Jitendra Malik (2011). “Semantic contours from inverse detectors.” In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 103).

Hariharan, Bharath, Pablo Arbeláez, Ross Girshick, and Jitendra Malik (2015). “Hypercolumns for Object Segmentation and Fine-grained Localization”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 111).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2014). “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”. In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 16, 71, 75, 85, 99).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep Residual Learning for Image Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 14, 16, 90, 91, 94, 95, 99–101).

Heitz, Geremy, Gal Elidan, Benjamin Packer, and Daphne Koller (2009). “Shape-Based Object Localization for Descriptive Classification”. In: *International Journal of Computer Vision (IJCV)* (cit. on p. 42).

Huang, Shaoli, Zhe Xu, Dacheng Tao, and Ya Zhang (2016). “Part-Stacked CNN for Fine-Grained Visual Categorization”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 100).

Hubel, David H and Torsten N Wiesel (1962). “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”. In: *The Journal of physiology* (cit. on p. 3).

Ioffe, Sergey and Christian Szegedy (2015). “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *International Conference on Machine Learning (ICML)* (cit. on p. 14).

Jaderberg, Max, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu koray (2015). "Spatial Transformer Networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 26, 100).

Jégou, Hervé, Matthijs Douze, Cordelia Schmid, and Patrick Pérez (2010). "Aggregating local descriptors into a compact image representation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 10).

Jia, Yangqing, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell (2014). "Caffe: Convolutional Architecture for Fast Feature Embedding". In: *ACM International Conference on Multimedia* (cit. on pp. 13, 61, 68, 86).

Joachims, T., T. Finley, and Chun-Nam Yu (2009). "Cutting-Plane Training of Structural SVMs". In: *Machine Learning* (cit. on pp. 24, 40, 55).

Johnson, Justin, Andrej Karpathy, and Li Fei-Fei (2016). "DenseCap: Fully Convolutional Localization Networks for Dense Captioning". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 111).

Joulin, Armand and Francis Bach (2012). "A convex relaxation for weakly supervised classifiers". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 19, 45).

Juneja, M., A. Vedaldi, C. V. Jawahar, and Andrew. Zisserman (2013). "Blocks that Shout: Distinctive Parts for Scene Classification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 68).

Kim, M. and Fernando De la Torre (2013). "Multiple Instance Learning via Gaussian Processes". In: *Data Mining and Knowledge Discovery (DMKD)* (cit. on pp. 45, 46).

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 13).

Kolesnikov, Alexander and Christoph H. Lampert (2016). "Seed, Expand and Constrain: Three Principles for Weakly-Supervised Image Segmentation". In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 27, 103, 104).

Krähenbühl, Philipp and Vladlen Koltun (2011). "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 103).

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 2, 11, 13, 14, 16, 75).

Krummenacher, Gabriel, Cheng S. Ong, and Joachim Buhmann (2013). "Ellipsoidal Multiple Instance Learning". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 45, 46).

Kulkarni, Praveen, Frédéric Jurie, Joaquin Zepeda, Patrick Pérez, and Louis Chevallier (2016). "SPLeaP: Soft Pooling of Learned Parts for Image Classification". In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 99, 100).

Kulkarni, Praveen, Joaquin Zepeda, Frederic Jurie, Patrick Pérez, and Louis Chevallier (2015). "Learning the Structure of Deep Architectures Using L1 Regularization". In: *British Machine Vision Conference (BMVC)* (cit. on pp. 15, 110).

Lacoste-Julien, Simon, Martin Jaggi, Mark Schmidt, and Patrick Pletscher (2013). "Block-Coordinate Frank-Wolfe Optimization for Structural SVMs". In: *International Conference on Machine Learning (ICML)* (cit. on p. 24).

Lafferty, John, Andrew McCallum, and Fernando Pereira (2001). “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data”. In: *International Conference on Machine Learning (ICML)* (cit. on p. 25).

Lai, Kuan-Ting, Felix X. Yu, Ming-Syan Chen, and Shih-Fu Chang (2014). “Video Event Detection by Inferring Temporal Instance Labels”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 19, 20, 113).

Lazebnik, Svetlana, Cordelia Schmid, and Jean Ponce (2006). “Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 10, 28, 68).

Le Roux, Nicolas, Mark Schmidt, and Francis Bach (2012). “A Stochastic Gradient Method with an Exponential Convergence Rate for Strongly-Convex Optimization with Finite Training Sets”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 38).

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel (1989). “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Comput.* (Cit. on p. 3).

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* (cit. on p. 3).

Li, Hanxi, Yi Li, and Fatih Porikli (2016). “Deeptrack: Learning discriminative feature representations online for robust visual tracking”. In: *IEEE Transactions on Image Processing (TIP)* (cit. on p. 16).

Li, Li-Jia and Fei-Fei Li (2007). “What, where and who? Classifying events by scene and object recognition”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 28).

Li, Li-Jia, Hao Su, Yongwhan Lim, and Li Fei-Fei (2014). “Object Bank: An Object-Level Image Representation for High-Level Visual Recognition”. In: *International Journal of Computer Vision (IJCV)* (cit. on pp. 4, 68, 70, 95, 98).

Li, Weixin and Nuno Vasconcelos (2015). “Multiple Instance Learning for Soft Bags via Top Instances”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 19, 20, 94, 104).

Lin, Tsung-Yi, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie (2017). “Feature Pyramid Networks for Object Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 111).

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (2014). “Microsoft COCO: Common Objects in Context”. In: *European Conference on Computer Vision (ECCV)*. URL: http://mscoco.org (cit. on pp. 27, 102).

Lloyd, S. (1982). “Least Squares Quantization in PCM”. In: *IEEE Transactions on Information Theory* (cit. on p. 10).

Lobel, Hans, Rene Vidal, and Alvaro Soto (2013). “Hierarchical Joint Max-Margin Learning of Mid and Top Level Representations for Visual Recognition”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 68).

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). “Fully Convolutional Networks for Semantic Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 17, 75, 91).

Lowe, David G. (2004). “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision (IJCV)* (cit. on p. 10).

Ma, Wei-Ying and B.S. Manjunath (1999). “NeTra: A toolbox for navigating large image databases”. In: *Multimedia Systems* (cit. on pp. 2, 10).

Mangasarian, O.L. and E.W. Wild (2008). “Multiple Instance Classification via Successive Linear Programming”. In: *Journal of Optimization Theory and Applications* (cit. on p. 45).

Miller, Kevin, M. Pawan Kumar, Benjamin Packer, Danny Goodman, and Daphne Koller (2012). “Max-Margin Min-Entropy Models”. In: *International Conference on Artificial Intelligence and Statistics (AISTAT)* (cit. on pp. 42, 47, 48).

Mohapatra, Pritish, C.V. Jawahar, and M. Pawan Kumar (2014). “Efficient Optimization for Average Precision SVM”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 57).

Oquab, Maxime, Léon Bottou, Ivan Laptev, and Josef Sivic (2014). “Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 4, 16, 17, 71).

Oquab, Maxime, Léon Bottou, Ivan Laptev, and Josef Sivic (2015). “Is object localization for free? – Weakly-supervised learning with convolutional neural networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 26, 28, 78, 82, 85–87, 94, 95, 98, 99, 102, 104).

Palmer, Martha Stone and Zhibiao Wu (1995). “Verb semantics for English-Chinese translation”. In: *Machine Translation* (cit. on p. 43).

Papandreou, George, Liang-Chieh Chen, Kevin Murphy, and Alan L. Yuille (2015). “Weakly-and semi-supervised learning of a DCNN for semantic image segmentation”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 103).

Papandreou, George, Iasonas Kokkinos, and Pierre-Andre Savalle (2015). “Modeling Local and Global Deformations in Deep Learning: Epitomic Convolution, Multiple Instance Learning, and Sliding Window Detection”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 26).

Parizi, Sobhan Naderi, John G Oberlin, and Pedro F Felzenszwalb (2012). “Reconfigurable models for scene recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 68).

Parizi, Sobhan Naderi, Andrea Vedaldi, Andrew Zisserman, and Pedro F Felzenszwalb (2015). “Automatic discovery and optimization of parts for image classification”. In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 19, 20, 86, 94, 99, 100).

Pathak, Deepak, Philipp Krahenbuhl, and Trevor Darrell (2015). “Constrained Convolutional Neural Networks for Weakly Supervised Segmentation”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 95, 103).

Pathak, Deepak, Evan Shelhamer, Jonathan Long, and Trevor Darrell (2015). “Fully Convolutional Multi-Class Multiple Instance Learning”. In: *International Conference on Learning Representations Workshop (ICLR-W)* (cit. on pp. 26, 103).

Paulin, Mattis, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid (2015). “Local Convolutional Features with Unsupervised Training for Image Retrieval". In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 16).

Perronnin, Florent and Christopher Dance (2007). "Fisher Kernels on Visual Vocabularies for Image Categorization". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 10).

Picard, David and Philippe-Henri Gosselin (2011). "Improving Image Similarity With Vectors of Locally Aggregated Tensors". In: *IEEE International Conference on Image Processing (ICIP)* (cit. on p. 10).

Ping, Wei, Qiang Liu, and Alex Ihler (2014). "Marginal Structured SVM with Hidden Variables". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 24, 25, 104).

Pinheiro, Pedro O. and Ronan Collobert (2015). "From Image-level to Pixel-level Labeling with Convolutional Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 26, 27, 103, 104).

Pinheiro, Pedro O, Ronan Collobert, and Piotr Dollar (2015). "Learning to Segment Object Candidates". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 16).

Pinheiro, Pedro O., Tsung-Yi Lin, Ronan Collobert, and Piotr Dollàr (2016). "Learning to Refine Object Segments". In: *European Conference on Computer Vision (ECCV)* (cit. on p. 16).

Pletscher, Patrick, Cheng Soon Ong, and Joachim M. Buhmann (2010). "Entropy and Margin Maximization for Structured Output Learning". In: *European Conference on Machine Learning (ECML)* (cit. on pp. 25, 26).

Polyak, Boris Teodorovich (1964). "Some methods of speeding up the convergence of iteration methods". In: *USSR Computational Mathematics and Mathematical Physics* (cit. on p. 13).

Quattoni, Ariadna and Antonio Torralba (2009). "Recognizing indoor scenes". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 28, 104).

Quattoni, Ariadna, Sy Bor Wang, Louis-Philippe Morency, Michael Collins, and Trevor Darrell (2007). "Hidden Conditional Random Fields". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on p. 25).

Rakotomamonjy, A., F. Bach, S. Canu, and Y. Grandvalet (2008). "SimpleMKL". In: *Journal of Machine Learning Research (JMLR)* (cit. on p. 5).

Razavian, Ali, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson (2014). "CNN features off-the-shelf: an astounding baseline for recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPR-W)* (cit. on p. 15).

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi (2016). "You Only Look Once: Unified, Real-Time Object Detection". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 16).

Reed, Scott, Zeynep Akata, Bernt Schiele, and Honglak Lee (2016). "Learning Deep Representations of Fine-grained Visual Descriptions". In: (cit. on p. 16).

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 16).

Rueping, Stefan (2010). "SVM Classifier Estimation from Group Probabilities." In: *International Conference on Machine Learning (ICML)* (cit. on p. 113).

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). “Learning representations by back-propagating errors”. In: *Nature* (cit. on pp. 11, 13).

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei (2015). “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* (cit. on pp. 2, 11, 27, 61, 81, 91, 95, 100).

Russakovsky, Olga, Yuanqing Lin, Kai Yu, and Li Fei-Fei (2012). “Object-centric spatial pooling for image classification”. In: *European Conference on Computer Vision (ECCV)* (cit. on p. 4).

Sabato, S. and N. Tishby (2012). “Multi-Instance Learning with Any Hypothesis Class”. In: *Journal of Machine Learning Research (JMLR)* (cit. on p. 36).

Sadeghi, Fereshteh and Marshall F Tappen (2012). “Latent Pyramidal Regions for Recognizing Scenes”. In: *European Conference on Computer Vision (ECCV)* (cit. on p. 68).

Salton, Gerard and Michael J. McGill (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc. (cit. on pp. 2, 10).

Saxena, Shreyas and Jakob Verbeek (2016). “Convolutional Neural Fabrics”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 15).

Schwing, A. G., T. Hazan, M. Pollefeys, and R. Urtasun (2011). “Distributed Message Passing for Large Scale Graphical Models”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 23).

Schwing, A. G., T. Hazan, M. Pollefeys, and R. Urtasun (2012). “Efficient Structured Prediction with Latent Variables for General Graphical Models”. In: *International Conference on Machine Learning (ICML)* (cit. on pp. 22, 25, 104).

Sermanet, Pierre, David Eigen, Xiang Zhang, Michaël Mathieu, Robert Fergus, and Yann Lecun (2014). “Overfeat: Integrated recognition, localization and detection using convolutional networks”. In: *International Conference on Learning Representations (ICLR)* (cit. on p. 13).

Shah, N., V. Kolmogorov, and C. H. Lampert (2015). “A multi-plane block-coordinate frank-wolfe algorithm for training structural SVMs with a costly max-oracle”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 24).

Shalev-Shwartz, Shai, Yoram Singer, Nathan Srebro, and Andrew Cotter (2011). “Pegasos: Primal estimated sub-gradient solver for svm”. In: *Mathematical programming* (cit. on p. 24).

Sharma, Gaurav, Frederic Jurie, and Cordelia Schmid (2012). “Discriminative spatial saliency for image classification”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 68).

Simon, Marcel and Erik Rodner (2015). “Neural activation constellations: Unsupervised part model discovery with convolutional networks”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 100).

Simonyan, Karen and Andrew Zisserman (2015). “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *International Conference on Learning Representations (ICLR)* (cit. on pp. 13, 14, 75, 81, 85–87, 99–101).

Sivic, Josef and Andrew Zisserman (2003). “Video Google: A Text Retrieval Approach to Object Matching in Videos”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 2, 10).

Smeulders, Arnold W. M., Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh C. Jain (2000). "Content-Based Image Retrieval at the End of the Early Years". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on p. 2).

Song, Yang, Alexander G. Schwing, Richard S. Zemel, and Raquel Urtasun (2016). "Training Deep Neural Networks via Direct Loss Minimization". In: *International Conference on Machine Learning (ICML)* (cit. on p. 57).

Sriperumbudur, Bharath K. and Gert R. G. Lanckriet (2009). "On the Convergence of the Concave-Convex Procedure". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 37).

Srivastava, Rupesh K, Klaus Greff, and Juergen Schmidhuber (2015). "Training Very Deep Networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 15).

Sun, Chen, Manohar Paluri, Ronan Collobert, Ram Nevatia, and Lubomir Bourdev (2016). "ProNet: Learning to Propose Object-Specific Boxes for Cascaded Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 27, 81, 95, 98, 99, 102, 104).

Sun, Jian and Jean Ponce (2013). "Learning discriminative part detectors for image classification and cosegmentation". In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 68).

Sutskever, Ilya, James Martens, George Dahl, and Geoffrey Hinton (2013). "On the importance of initialization and momentum in deep learning". In: *International Conference on Machine Learning (ICML)* (cit. on p. 13).

Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi (2016). "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning". In: arXiv: 1602.07261. URL: http://arxiv.org/pdf/1602.07261v2 (cit. on p. 101).

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich (2015). "Going Deeper with Convolutions". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 13, 14, 16, 100, 101).

Szummer, Martin, Pushmeet Kohli, and Derek Hoiem (2008). "Learning CRFs using graph cuts". In: *European Conference on Computer Vision (ECCV)* (cit. on p. 23).

Taskar, Ben, Carlos Guestrin, and Daphne Koller (2003). "Max-Margin Markov Networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 22).

Teh, Eu Wern, Mrigank Rochan, and Yang Wang (2016). "Attention Networks for Weakly Supervised Object Localization". In: *British Machine Vision Conference (BMVC)* (cit. on p. 27).

Tieleman, T. and G. Hinton (2012). "RMSprop Gradient Optimization". In: (cit. on p. 13).

Toshev, Alexander and Christian Szegedy (2014). "DeepPose: Human Pose Estimation via Deep Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 16).

Tsochantaridis, Ioannis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun (2005). "Large Margin Methods for Structured and Interdependent Output Variables". In: *Journal of Machine Learning Research (JMLR)* (cit. on pp. 22, 24, 55).

Vapnik, Vladimir (1991). "Principles of risk minimization for learning theory". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 12).

Vapnik, Vladimir (1995). *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer New York Inc. (cit. on p. 11).

Vedaldi, A. and K. Lenc (2015). "MatConvNet – Convolutional Neural Networks for MATLAB". In: *ACM Int. Conf. on Multimedia* (cit. on p. 48).

Wah, C., S. Branson, P. Welinder, P. Perona, and S. Belongie (2011). *The Caltech-UCSD Birds-200-2011 Dataset*. Tech. rep. CNS-TR-2011-001. California Institute of Technology (cit. on p. 28).

Wang, Hua, Feiping Nie, and Heng Huang (2012). "Robust and discriminative distance for multi-instance learning". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 18).

Wang, Jun and Jean-Daniel Zucker (2000). "Solving the Multiple-Instance Problem: A Lazy Learning Approach". In: *International Conference on Machine Learning (ICML)* (cit. on p. 18).

Wang, Lijun, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu (2015). "Visual tracking with fully convolutional networks". In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 16).

Wang, Xin, D. Kumar, N. Thome, M. Cord, and F. Precioso (2015). "Recipe recognition with large multimodal food dataset". In: *2015 IEEE International Conference on Multimedia Expo Workshops (ICMEW)* (cit. on p. 16).

Wang, Xinggang, Baoyuan Wang, Xiang Bai, Wenyu Liu, and Zhuowen Tu (2013). "Max-Margin Multiple-Instance Dictionary Learning". In: *International Conference on Machine Learning (ICML)* (cit. on p. 68).

Wang, Yaming, Jonghyun Choi, Vlad Morariu, and Larry S. Davis (2016). "Mining Discriminative Triplets of Patches for Fine-Grained Classification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 16).

Wei, Yunchao, Wei Xia, Junshi Huang, Bingbing Ni, Jian Dong, Yao Zhao, and Shuicheng Yan (2014). "CNN: Single-label to Multi-label". In: *CoRR* abs/1406.5726. URL: http://arxiv.org/abs/1406.5726 (cit. on p. 99).

Wei, Zijun and Minh Hoai (2016). "Region Ranking SVM for Image Classification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 98–101).

Wu, Ruobing, Baoyuan Wang, Wenping Wang, and Yizhou Yu (2015). "Harvesting Discriminative Meta Objects With Deep CNN Features for Scene Classification". In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 99, 100).

Xiao, Tianjun, Yichong Xu, Kuiyuan Yang, Jiaxing Zhang, Yuxin Peng, and Zheng Zhang (2015). "The Application of Two-Level Attention Models in Deep Convolutional Neural Network for Fine-Grained Image Classification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 100).

Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He (2016). "Aggregated Residual Transformations for Deep Neural Networks". In: arXiv: 1611.05431. URL: http://arxiv.org/pdf/1611.05431v1 (cit. on pp. 100, 101).

Xu, Huijuan and Kate Saenko (2016). "Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering". In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 16, 27).

Xu, Jia, Alexander G. Schwing, and Raquel Urtasun (2014). "Tell Me What You See and I will Show You Where It Is". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 22).

Xu, Zhe, Dacheng Tao, Ya Zhang, Junjie Wu, and Ah Chung Tsoi (2014). "Architectural style classification using multinomial latent logistic regression". In: *European Conference on Computer Vision (ECCV)* (cit. on p. 25).

Yang, Jimei, Brian L. Price, Scott Cohen, Honglak Lee, and Ming-Hsuan Yang (2016). "Object Contour Detection with a Fully Convolutional Encoder-Decoder Network". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 16).

Yang, Weilong, Yang Wang, Arash Vahdat, and Greg Mori (2012). "Kernel Latent SVM for Visual Recognition". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 47, 48).

Yao, Bangpeng and Li Fei-Fei (2010). "Grouplet: A Structured Image Representation for Recognizing Human and Object Interactions". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 28).

Yu, Chun-Nam and Thorsten Joachims (2009). "Learning Structural SVMs with Latent Variables". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 22, 23, 58, 61, 62, 68, 104, 116).

Yu, Felix X., Dong Liu, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang (2013). "∝SVM for Learning with Label Proportions." In: *International Conference on Machine Learning (ICML)* (cit. on pp. 19, 20, 94, 104, 113).

Yue, Yisong, T. Finley, F. Radlinski, and T. Joachims (2007). "A Support Vector Method for Optimizing Average Precision". In: *SIGIR* (cit. on pp. 57, 58, 60, 80).

Yuille, Alan L and Anand Rangarajan (2003). "The concave-convex procedure". In: *Neural Computation* (cit. on pp. 23, 24, 37, 55, 62).

Zagoruyko, S., A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár (2016). "A MultiPath Network for Object Detection". In: *British Machine Vision Conference (BMVC)* (cit. on p. 16).

Zeiler, Matthew D. (2012). "ADADELTA: An Adaptive Learning Rate Method". In: *CoRR* abs/1212.5701. URL: http://arxiv.org/abs/1212.5701 (cit. on p. 13).

Zeiler, Matthew D. and Rob Fergus (2014). "Visualizing and Understanding Convolutional Networks". In: *European Conference on Computer Vision (ECCV)* (cit. on p. 13).

Zeiler, Matthew and Robert Fergus (2013). "Stochastic pooling for regularization of deep convolutional neural networks". In: *International Conference on Learning Representations (ICLR)* (cit. on p. 16).

Zhang, Dan, Jingrui He, Luo Si, and Richard D. Lawrence (2013). "MILEAGE: Multiple Instance LEArning with Global Embedding". In: *International Conference on Machine Learning (ICML)* (cit. on pp. 45, 46).

Zhang, Ning, Manohar Paluri, Marc'Aurelio Ranzato, Trevor Darrell, and Lubomir Bourdev (2014). "PANDA: Pose Aligned Networks for Deep Attribute Modeling". In: *European Conference on Computer Vision (ECCV)* (cit. on pp. 4, 17, 69).

Zhang, Ziyu, Sanja Fidler, and Raquel Urtasun (2016). "Instance-Level Segmentation with Deep Densely Connected MRFs". In: (cit. on p. 16).

Zheng, Shuai, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr (2015). "Conditional Random Fields as Recurrent Neural Networks”. In: *IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 16).

Zhou, B., A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva (2014). “Learning Deep Features for Scene Recognition using Places Database.” In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 16, 61, 68, 69, 86, 99, 100).

Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba (2016). “Learning Deep Features for Discriminative Localization”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 26, 77, 93–96, 99–101, 104).

Zhou, Bolei, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba (2016). “Semantic understanding of scenes through the ade20k dataset”. In: *arXiv preprint arXiv:1608.05442* (cit. on p. 21).

Zhou, Feng and Yuanqing Lin (2016). “Fine-grained Image Classification by Exploring Bipartite-Graph Labels”. In: (cit. on p. 16).

Zhou, Xi, Kai Yu, Tong Zhang, and Thomas Huang (2010). “Image Classification Using Super-vector Coding of Local Image Descriptors”. In: *European Conference on Computer Vision (ECCV)* (cit. on p. 10).

Zhou, Zhi-Hua, Yu-Yin Sun, and Yu-Feng Li (2009). “Multi-instance Learning by Treating Instances As non-I.I.D. Samples”. In: *International Conference on Machine Learning (ICML)* (cit. on pp. 18, 45, 46).

Zuo, Zhen, Gang Wang, Bing Shuai, Lifan Zhao, Qingxiong Yang, and Xudong Jiang (2014). “Learning Discriminative and Shareable Features for Scene Classification”. In: *European Conference on Computer Vision (ECCV)* (cit. on p. 68).

NOTATION

SYMBOL	MEANING
$\boldsymbol{w} \in \mathbb{R}^d$	vector of parameters (dimension d)
$f_{\boldsymbol{w}}(\boldsymbol{x})$	prediction function $f_{\boldsymbol{w}} : \mathcal{X} \to \mathcal{Y}$
$s_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{y})$	scoring function $s_{\boldsymbol{w}} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$
$\mathcal{X}$	input space
$\boldsymbol{x} \in \mathcal{X}$	input variable
$\mathcal{Y}$	output space
$\boldsymbol{y} \in \mathcal{Y}$	output variable
$\boldsymbol{y}^\star$	ground truth output variable
$\mathcal{H}$	latent space
$\boldsymbol{h} \in \mathcal{H}$	latent variable
$\Psi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{h})$	joint feature map $\Psi : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}^d$
$\Phi(\boldsymbol{x}, \boldsymbol{h})$	joint feature map $\Phi : \mathcal{X} \times \mathcal{H} \to \mathbb{R}^d$
$\mathcal{P}$	set of positive examples
$\mathcal{N}$	set of negative examples
$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^\star)\}_{i=1}^N$	training set with N examples
$\mathcal{L}(\boldsymbol{y}^\star, \boldsymbol{y})$	loss $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
$\mathcal{R}$	regularizer
$[x]_+ = \max(0, x)$	hinge loss

ACRONYMS

ADIOS	Architectures Deep In Output Space
AP	Average Precision
BN	BossaNova
BoW	Bag of Words
CCCP	Concave-Convex Procedure
ConvNet	Convolutional Network
CPU	Central Processing Unit
CRF	Conditional Random Field
CUB-200	Caltech-UCSD Birds 200
DC	Difference of Convex functions
DPM	Deformable Part Model
FC-CRF	Fully Connected CRF
FCN	Fully Convolutional Network
FPN	Feature Pyramid Network
FT	Fine-tuning
FV	Fisher Vectors
GAP	Global Average Pooling
GPU	Graphics Processing Unit
GT	Ground Truth
GWRP	Global Weighted Rank-Pooling
HCRF	Hidden Conditional Random Field
HOG	Histogram of Oriented Gradients
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoU	Intersection-over-Union
i.i.d.	independent and identically distributed

k-NN	*k*-Nearest Neighbors
KLSVM	Kernel Latent SVM
M3E	Max-Margin Min-Entropy
MANTRA	Minimum mAximum lateNt sTRucturAl SVM
MAP	Mean Average Precision
MIL	Multiple-Instance Learning
MKL	Multiple Kernel Learning
MLLR	Multinomial Latent Logistic Regression
MS COCO	Microsoft Common Objects in Context
MSSVM	Marginal Structured SVM
NCCP	Non-Convex Cutting-Plane
NINB	Negative Instances in Negative Bags
LAI	Loss-Augmented Inference
LAPSVM	Latent AP-SVM
LLP	Learning with Label Proportion
LSE	Log-Sum-Exp
LSVM	Latent SVM
LSSVM	Latent Structured SVM
OB	Object Bank
PPMI	People Playing Musical Instrument
QP	Quadratic Program
R-FCN	Region-based Fully Convolutional Network
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RMIMN	Ratio-constrained Multiple Instance Markov Network
RNN	Recurrent Neural Network
RoI	Region-of-Interest
SAG	Stochastic Average Gradient

SIFT	Scale-Invariant Feature Transform
SGD	Stochastic Gradient Descent
SPEN	Structured Prediction Energy Network
SPM	Spatial Pyramid Matching
SSVM	Structured SVM
STN	Spatial Transformer Network
SVC	Super-Vector Coding
SVM	Support Vector Machine
VLAD	Vector of Locally Aggregated Descriptors
VLAT	Vector of Locally Aggregated Tensors
VQA	Visual Question Answering
WELDON	WEakly supervised Learning of Deep cOnvolutional neural Network
WILDCAT	Weakly supervIsed Learning of Deep Convolutional neurAl neTwork
WSL	Weakly Supervised Learning