Mach Learn
DOI 10.1007/s10994-013-5336-9

Learning semantic representations of objects and their parts

Grégoire Mesnil · Antoine Bordes · Jason Weston · Gal Chechik · Yoshua Bengio

Received: 24 May 2012 / Accepted: 26 February 2013

Abstract Recently, large scale image annotation datasets have been collected with millions of images and thousands of possible annotations. Latent variable models, or embedding methods, that simultaneously learn semantic representations of object labels and image representations can provide tractable solutions on such tasks. In this work, we are interested in jointly learning representations both for the objects in an image, and the parts of those objects, because such deeper semantic representations could bring a leap forward in image retrieval or browsing. Despite the size of these datasets, the amount of annotated data for objects and parts can be costly and may not be available. In this paper, we propose to bypass this cost with a method able to learn to jointly label objects and parts without requir-

Editors: Ronan Collobert and Luke Zettlemoyer.

G. Mesnil (✉) · Y. Bengio
LISA, Université de Montréal, Montreal, QC, Canada
e-mail: mesnilgr@iro.umontreal.ca

Y. Bengio
e-mail: bengioy@iro.umontreal.ca

G. Mesnil
LITIS, Université de Rouen, Rouen, France

A. Bordes
CNRS—Heudiasyc UMR 7253, Université de Technologie de Compiègne, Compiégne, France
e-mail: antoine.bordes@utc.fr

J. Weston
Google, New York, NY, USA
e-mail: jweston@google.com

G. Chechik
Google, Mountain View, CA, USA
e-mail: gal@google.com

G. Chechik
Gonda Brain research center, Bar-Ilan University, Ramat Gan, Israel
e-mail: gal.chechik@biu.ac.il

Published online: 24 April 2013

ing exhaustively labeled data. We design a model architecture that can be trained under a proxy supervision obtained by combining standard image annotation (from ImageNet) with semantic part-based within-label relations (from WordNet). The model itself is designed to model both object image to object label similarities, and object label to object part label similarities in a single joint system. Experiments conducted on our combined data and a precisely annotated evaluation set demonstrate the usefulness of our approach.

Keywords Ranking · Image retrieval · Embeddings · Object and parts

1 Introduction

Images and language are two complementary representations of information, and learning to translate between the two is of great interest. In one direction, the task of image annotation maps images to words, and in the other direction, the task of image retrieval maps words to images. Both the semantics of words and the semantics of images play a key role in these two tasks, since an accurate mapping is required to retrieve images and text that are semantically similar. At the same time, there also exist complex semantic relations within each modality that are important to model. For example, between-objects relations include the relation *X is a part of Y* and *X is an instance of Y*, and similar relations exist in the semantics of text terms. We wish to develop models that learn these types of semantic relations across and within modalities simultaneously.

Automatic tools provided by machine learning can be designed to capture the semantics described above. However, in real applications both the dictionary of possible words and the set of possible objects are large and learning their semantics requires a large amount of training data. Indeed, the performance of machine learning models highly depends on the quality and size of their training data sets, so there is a clear incentive to design methods able to handle the huge resources now available.

In this work, we develop a machine learning method to learn the semantics of words, objects and parts of objects that is efficient enough to be trained on large scale datasets. The method works by learning a latent semantic representation for each possible word or phrase, and each object and part. Each semantic concept has a vectorial representation in a low dimensional embedding space, the dimensions of which are learnt from data. In the low dimensional semantic embedding space we want to capture similarities between words, objects and part of objects, e.g. so that objects are related to their particular labels and parts in the space. To do this we employ a loss function that optimizes the ranking of words given an object, the loss function tries to get the correct assignment near the top of the ranked list.

The task we consider is of automatically labeling images with the objects in the scene as well as the parts of those objects. Importantly, we are interested in object parts that can be viewed as objects by themselves, like the flash bulb of a camera, or a wheel of a car. This is different from the numerous unnamed parts of which every object consists of. Identifying such "parts that are objects", can be particularly useful in image retrieval and browsing, where a large amount of such precisely labeled data could potentially lead to a compelling advance. For example, this could allow to automatically extend image queries via semantic part-based bridges: when looking for images of a particular object (say a "car"), one could also be presented images of related object parts (such as "wheel" or "windshield"). This would seamlessly enhance the browsing experience. Unfortunately, although some limited datasets with this type of labeling are available (Yao et al. 2007; Russell et al. 2008), collecting such deeply annotated data is costly and time consuming, Moreover, although some image annotation tasks can benefit from collecting indirect data, like the case of users clicking on images in search engines or accompanying text surrounding images, there are not any large scale applications that provide such evidence for object parts in images that we are aware of.

In this work we hence also propose a training method to tackle this problem of labeling objects and their parts in images without requiring any precisely labeled data. Our approach relies on a proxy supervision which bypasses the problem of precise annotation by using part-based semantic information among labels. Our model is composed of two ranking components: one for ranking labels according to an image and one for ranking labels according to another label. The first component can be seen as a standard image annotation model whereas the second one learns to give high ranks to pairs of labels for which one is the part of the other. The two components are trained jointly using combined data built from two sources (ImageNet and WordNet).

This paper builds upon previous work published by Weston et al. (2011). However, the previous work has only focused on the standard image annotation setting, whereas the present version proposes jointly learning object and part representations. Hence, many new elements are provided including the word, object and part joint model, its training scheme, the proxy supervision setting, the ImageNet+WordNet dataset and all experimental results on it and on the LHI data set (Yao et al. 2007).

The paper is organized as follows. Section 2 describes learning joint latent semantic models for words and objects. Section 3 presents our full image annotation model for learning latent semantic models over words, objects and their parts. Section 4 then describes how to train both types of model. In particular, the form of the loss for supervised learning is first described, and then we introduce our proxy supervision setup for training objects and their parts when supervision is limited, and describe our corresponding custom data set. Section 5 describes related work on image annotation and part-based approaches. An empirical evaluation on our custom test set and on precisely labeled images from the LHI data set is given in Sect. 6. Finally, Sect. 7 concludes.

2 Latent semantic word and object models

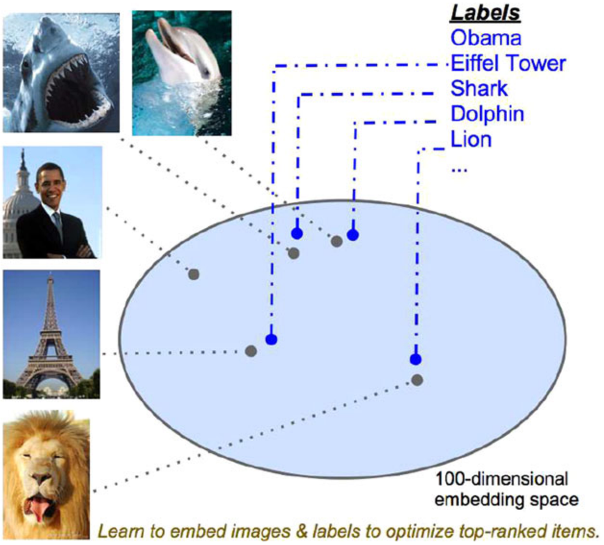
Before we describe the model that learns the semantics of words, objects and their parts we start with a simpler model of representing only words and objects, previously described in Weston et al. (2011). In this case, a large amount of supervised data can be obtained, and we hence detail a supervised training criterion for learning the appropriate ranking function. We will explain below how this setup is extended to learning about objects and their parts.

The following model learns a single latent semantic feature representation where both objects in images and word annotations are represented. The mapping functions for the two views are different, but are learnt jointly to optimize the supervised loss of interest for our final task (here, we concentrate on the task of annotating images). The method is described pictorially in Fig. 1.

Notation summary

- $\mathcal{L}$ is a set depicted by the K words of the dictionary corresponding to the image annotations.
- $\mathcal{I}$ is the raw pixel space of images (no constraints on the size of the images).
- $\Phi_{\mathcal{I}} : \mathcal{I} \to \mathbb{R}^N$ maps an image to its sparse representation.
- $\Phi_{\mathcal{L}} : \mathcal{L} \to \mathbb{N}$ maps an annotation to its index in the dictionary.

Fig. 1 Learning latent semantic word and object models


Labels
Obama
Eiffel Tower
Shark
Dolphin
Lion
...
100-dimensional embedding space
Learn to embed images & labels to optimize top-ranked items.

- $E_{\mathcal{I}} : \mathcal{I} \to \mathbb{R}^D$ *embeds* an image to the semantic space.
- $E_{\mathcal{L}} : \mathcal{L} \to \mathbb{R}^D$ *embeds* an annotation to the same semantic space. In some cases, the semantic space for annotations is different than the images semantic space (see Sect. 6.2 about Unshared models).
- $f_I : \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ returns a score defining the similarity between an annotation and an image.

Given an image I_o containing an object we wish to learn a function $E_{\mathcal{I}} : \mathcal{I} \to \mathbb{R}^D$ that *embeds* the inferred object I_o into a low-dimensional semantic space (where D is typically 50–100 dimensions and $E_{\mathcal{I}}(I_o)$ is the representation of I_o in the semantic space). Simultaneously, given an annotation of an object L_o, we wish to also learn a function $E_{\mathcal{L}} : \mathcal{L} \to \mathbb{R}^D$ that represents the annotation in the same semantic space. Then, our overall model takes the form:

$$f_I(I_o, L_o) = S\big(E_{\mathcal{I}}(I_o), E_{\mathcal{L}}(L_o)\big),$$

where $S : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ is a measure of similarity in the semantic embedding space e.g. a dot product $S(x, y) = x^\top y$.

For the feature representation of images, we first employ a fixed mapping $\Phi_{\mathcal{I}}(\cdot)$ that transforms the pixel representation into an N-dimensional vector which in this paper is a high-dimensional and sparse feature representation, as is also commonly performed in other works. Then, we transform this intermediate representation to the D-dimensional semantic space, via a linear map using a $D \times N$ matrix V of parameters:

$$E_{\mathcal{I}}(I_o) = V\Phi_{\mathcal{I}}(I_o).$$

In our model there is a dictionary with K possible labels that can be embedded using $E_{\mathcal{L}}(\cdot)$. Following other works dealing with embedding representations for text (Bengio 2008; Weston et al. 2010) for each label we learn a D dimensional vector that will represent the label, resulting in a $D \times K$ dimensional matrix W of parameters to learn for the K labels:

$$E_{\mathcal{L}}(L_o) = W_{\Phi_{\mathcal{L}}(L_o)},$$

where $\Phi_{\mathcal{L}}(\cdot)$ maps from the particular label to its index in the dictionary, and thus retrieves the relevant column of a $D \times K$ matrix W. (Note that a standard matrix product with a one-hot vector would perform the same operation.)

Finally, for S we choose the dot product similarity in the semantic space, resulting in the final model:

$$f_I(I_o, L_o) = \left(V \Phi_{\mathcal{I}}(I_o)\right)^\top W_{\Phi_{\mathcal{L}}(L_o)}.$$

Our goal is to rank the candidate annotations of a given image such that the highest ranked annotations describe best the semantic content of the image. We will describe the training procedure in Sect. 4.

3 Latent semantic word, object and part models

We want our model to simultaneously learn about objects being parts of scenes (a mapping from images to object labels) and about parts of objects belonging to the objects (and again, the labels of those parts). We hence now describe the generalization of the word, object model from the previous section to this case.

Notation summary

- $f_L : \mathcal{L} \times \mathcal{L} \rightarrow \mathbb{R}$ returns a score defining the similarity between an object annotation and a part annotation.
- $f_q : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \rightarrow \mathbb{R}$ returns a score given a quadruplet of images and annotations of a part and an object.
- $f_I^U : \mathcal{I} \times \mathcal{L} \rightarrow \mathbb{R}$ differs from f_I since it has its own set of parameters and these parameters correspond to the annotation embeddings that are not shared between the score functions f_I^U and f_L. In this case, the annotations semantic space is different than the images semantic space.

Our full model takes the form:

$$f_q(I_o, L_o, I_p, L_p) = f_I(I_o, L_o) + f_I(I_p, L_p) + f_L(L_o, L_p),$$

where I_o is the image in which an object of interest is located, I_p is the image in which an object part of interest is located, L_o is the label of the object and L_p is the label of the part. The function $f_q(I_o, L_o, I_p, L_p)$ which scores a given quadruplet of two images and two labels is decomposed into three functions. The function $f_I(I_o, L_o)$ scores a given object label with the image of an object, and $f_I(I_p, L_p)$ does the same for a part label with the image of a part. The last function f_L scores the match between an object label and a part label. (Note, we do not have a direct $f_{II}(I_o, I_p)$ term in our model to capture image-image relationships directly although that is also possible.)

This model can be used in several setups. Given the image of an object I_o and a subregion of that image denoted with I_p, with no other prior knowledge, we can label the object in I_o and the part in I_p with:

$$\arg\max_{L'_o, L'_p} f_q\left(I_o, L'_o, I_p, L'_p\right).$$

If the object label is already known and we are just looking for the names of the parts we could fix L_o as well:

$$\arg\max_{L'_p} f_q\left(I_o, L_o, I_p, L'_p\right).$$

Finally, if we are only interested in labeling objects (and not their parts) we can use:

$$\arg\max_{L'_o} f_I\big(I_o, L'_o\big).$$

The experiments below evaluate these different setups. It now remains to define the particular makeup of the functions f_I and f_L.

As before, we assume that we are given a dictionary of K possible labels but now these labels are for both objects and parts, all in the same dictionary. For example, a house has a door, and in turn, the door has a handle, and so on. Again, for each label we will learn a D dimensional vector that will represent it in the semantic space, resulting in a $D \times K$ dimensional matrix W of parameters to learn for the K labels. The similarity between two labels is then defined with:

$$f_L(L_o, L_p) = W^\top_{\Phi_\mathcal{L}(L_o)} W_{\Phi_\mathcal{L}(L_p)},$$

where W_i indexes the ith column (label) of W.

In the function f_I we deal with images of parts and objects. To measure similarity between an image and a label, we then have to transform them into the same space. This is again achieved with a fixed mapping $\Phi_\mathcal{I}(\cdot)$ and another $D \times N$ matrix V of parameters:

$$f_I(I, L) = \big(V\Phi_\mathcal{I}(I)\big)^\top W_{\Phi_\mathcal{L}(L)}.$$

Note also that the label embeddings W are also shared between all functions f_L and f_I. In our experiments we also consider a "non-sharing" setting where we decouple some of these parameters, so we instead consider:

$$f_I^U(I, L) = \big(V\Phi_\mathcal{I}(I)\big)^\top U_{\Phi_\mathcal{L}(L)},$$

where U is now a different set of parameters to W.

4 Training our models

Two following Sects. 4.1 and 4.2 are taken from previous work (Weston et al. 2011) and describe the original object-label setting and the WARP loss method. Afterwards, the object-parts model is built upon that and described in detail in Sect. 4.3.

4.1 Ranking loss function

We first consider the task of ranking labels $i \in \mathcal{Y}$ given an image x, i.e. the image annotation problem. In our setting, labeled pairs (x, y) will be provided for training where only a single annotation $y_i \in \mathcal{Y}$ is labeled correct.[1] Let $f(x) \in \mathbb{R}^K$ be a vector function providing a score for each of the labels, where $f_i(x)$ is the value for label i.

A classical loss function for learning to rank is to maximize AUC by minimizing:

$$err_{AUC}\big(f(x), y\big) = \sum_x \sum_y \sum_{\bar{y} \neq y} \max\big(0, 1 + f_{\bar{y}}(x) - f_y(x)\big),$$

[1] However, the methods described in this paper could be generalized to the multi-label case, naively by averaging the loss over all positive labels.

see, e.g. Herbrich et al. (2000). This tries to make the positive label y ranked above negative labels $\bar{y}$, because it sums over all negative labels it optimizes the mean rank. It also enforces a margin of 1 as in margin-based methods like SVMs (Boser et al. 1992). To make training of such a loss function scalable to large datasets with our model, one can optimize this loss using stochastic gradient descent (SGD): sample triplets $(x, y, \bar{y})$ to make a gradient step on the hinge loss.

However, there is an issue with the loss above that, because all pairwise errors are the same (because it optimizes the mean rank), it may not be the best loss function for getting the correct label at the top of the ranked list (e.g. within the top k). To give a simple example, suppose we are given only two functions to choose from, during our learning step we wish to pick the best one of the two. Given two training images, if function 1 ranks their true labels at position 1 and position 100 respectively, and function 2 ranks both at position 50, then the AUC loss function prefers these functions equally as they both have 100 "constraint violations", assuming the margin is the same. However, function 1 gives a superior precision at 1, because at least it gets one of the two examples correct.

To fix this problem, a class of ranking error functions was recently defined in Usunier et al. (2009a) as:

$$err\big(f(x), y\big) = L\big(rank_y\big(f(x)\big)\big), \tag{1}$$

where $rank_y(f(x))$ is the rank of the true label y given by $f(x)$:

$$rank_y\big(f(x)\big) = \sum_{i \neq y} I\big(f_i(x) \geq f_y(x)\big),$$

where I is the indicator function, and $L(\cdot)$ transforms this rank into a loss:

$$L(k) = \sum_{j=1}^{k} \alpha_j, \quad \text{with } \alpha_1 \geq \alpha_2 \geq \cdots \geq 0. \tag{2}$$

This class of functions allows one to define different choices of $L(\cdot)$ with different minimizers. Minimizing L with $\alpha_j = \frac{1}{K-1}$ would optimize the mean rank, $\alpha_1 = 1$ and $\alpha_{j>1} = 0$ the proportion of top-ranked correct labels, and larger values of α in the first few positions optimize the top k in the ranked list, which is of interest for optimizing precision at k. Now, given our example of two functions from before where function 1 ranks their true labels at position 1 and position 100 respectively, and function 2 ranks both at position 50, then a choice of $\alpha_j = \frac{1}{K-1}$ prefers these functions equally (just like AUC), whereas a choice of $\alpha_j = 1/j$ prefers the first function, which gives superior precision at 1.

To optimize (1) for large scale data one can also use stochastic gradient descent by making updates of the parameters β over randomly sampled examples and negative labels of the form:

$$\beta_{t+1} = \beta_t - \gamma_t \frac{\partial \overline{err}(f(x), y, \bar{y})}{\partial \beta_t}, \tag{3}$$

where γ_t is the learning rate and

$$\overline{err}\big(f(x), y, \bar{y}\big) = L\big(rank_y\big(f(x)\big)\big) \max\big(1 - f_{\bar{y}}(x) + f_y(x), 0\big).$$

Note the difference to the AUC optimization is just that the weighting of the update now depends on the (current) rank of y.

To perform this SGD step we still have the problem that computing $rank^1_y(f(x))$ is quite inefficient: to know the rank of $f_y(x)$ we have to compute the values $f_i(x)$ for $i = 1, \dots, K$. However, there is a sampling trick that can solve this problem making it much more efficient: the idea is that we can sample labels i uniformly with replacement until we find a violating label. If there are $k = rank^1_y(f(x))$ violating labels, the random variable N_k which counts the number of trials in our sampling step follows a geometric distribution of parameter $\frac{k}{K-1}$ (i.e. $\Pr(N_k > q) = (1 - \frac{k}{K-1})^q$). Thus $k = \frac{K-1}{E[N_k]}$. This suggests that the value of $rank^1_y(f(x))$ may be approximated by:

$$rank^1_y\bigl(f(x)\bigr) \approx \left\lfloor \frac{K-1}{N} \right\rfloor,$$

where $\lfloor . \rfloor$ is the floor function and N is the number of trials in the sampling step. This method is called the Weighted Approximate Ranked Pairwise (WARP) loss.

4.2 Training object and label models

Solving the image annotation problem with the semantic embedding model with objects and labels hence consists of the joint word-image embedding model of Sect. 2 trained with the WARP loss of Sect. 4.1. This method is called WSABIE in Weston et al. (2011). The mapping matrices V and W are initialized at random with mean 0, standard deviation $\frac{1}{\sqrt{d}}$, which is a common choice. We regularize the weights of our models by giving them constrained norm:

$$\|V_i\|_2 \le C, \quad i = 1, \dots, d, \tag{4}$$

$$\|W_i\|_2 \le C, \quad i = 1, \dots, K, \tag{5}$$

which acts as a regularizer in the same way as is used in lasso (Tibshirani 1996). During SGD, the initial weights are rescaled if they violate the constraints (4)–(5).

The task described here is fully supervised. The task of learning object and part models is the subject of the next subsection.

4.3 Training word, object and part models

4.3.1 Proxy supervision

Learning to label objects and their parts in images requires a vast amount of precisely labeled data which is costly to collect. This problem is particularly challenging because it involves both a hard data collection problem and a nontrivial model learning problem. We first propose an approach to address the data collection issue.

Our method is based on two observations. First, large datasets exist today with images labeled with their depicted objects (e.g. Deng et al. 2009a). These datasets contain millions of images annotated with thousands of terms. Second, many knowledge bases (including Miller 1995; Bollacker et al. 2008; Suchanek et al. 2008) provide various semantic relation between words (such as *part-of*). We propose to combine these two kinds of data sources to provide training data for our task.

Specifically, we use here ImageNet (Deng et al. 2009a) and WordNet (Miller 1995), two databases based on a common set of semantic categories. WordNet is a large database encompassing a comprehensive lexical knowledge within its graph structure, whose nodes (termed *synsets*) correspond to word senses, and edges define different types of relations including *is-a* and *part-of*. ImageNet is an image database organized according to the WordNet graph, thus providing a visual counterpart to WordNet, whereby a set of images is associated to each synset. As it is hard to obtain many images with labeled objects and parts, we propose to use instead pairs of images whose labels are semantically linked using a *part-of* relation in WordNet. This is the key idea of our proxy supervision. For instance, if we want to learn to label a "car" and its parts, WordNet provides a list of candidates for those such as "wheel", "windshield" or "sunroof". Now, using ImageNet, we can access many images of cars, of wheels, of windshields and of sunroofs. In this way, we can create many training examples by pairing images of cars to images of its parts. The difficulty, and the learning challenge, is that the images of parts do not come from the same images as the object they are supposed to belong to—they can even be very different in scale, texture and lighting. The learning algorithm must be able to somewhat abstractly represent the objects to be able to train under such an indirect supervision.

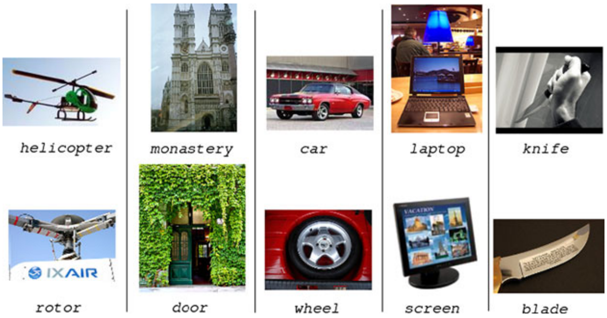
The next section details how we created our dataset. Note that, in this paper, we only consider the *part-of* relation in WordNet but our approach can be extended to other relations such as *type-of* or *instance-of*, and/or using knowledge bases other than WordNet.

4.3.2 The ImageNet+WordNet data set

To build our dataset, we selected 1,000 synsets appearing in both sides of *part-of* relations of WordNet and which were depicted by (at least) 400 images in ImageNet. These 1,000 synsets compose 771 *part-of* relations and are associated with a total of 400,490 pictures. After splitting, we were left with a training set of 324,158 word-image couples, a validation set of 37,920, and a test set of 38,412. To ensure the representativeness of the different sets, we made sure that at least one pair of examples representing all 771 *part-of* relations was present in each split.

The different models presented in the next section are trained using quadruplets (I_o, L_o, I_p, L_p) where I_o is an object image (and L_o its label) and I_p a part image (and L_p its label). We recall that the originality of our approach comes from the fact that I_p is not taken from the image I_o. Hence this proxy training data will have quadruplets that are labeled images of doors and door handles, for example, but the *door handle image is not from that particular door*. Nevertheless, one can hope to generalize from this type of proxy data in the absence of direct supervision. Examples of such quadruplets are given in Fig. 2. An image is potentially labeled with several synsets since different objects can be present in the same image. Using the more than 400,000 labeled images and the 771 *part-of* relations, we could construct more than 100 M of such quadruplets. Such a number of training examples is impossible to reach using exhaustive image labeling. We trained on 10 M quadruplets constructed using the 324,158 training couples. For evaluation, we created 50 k validation quadruplets out of the 37,920 pairs and 100 k test ones out of the 38,412 pairs.

Fig. 2 Quadruplets from ImageNet+WordNet dataset e.g. "rotor" is *part-of* "helicopter". Note that objects and their parts are not coming from the same image



4.3.3 Proxy training method

Proxy ranking task. Given a positive quadruplet $x = (I_o, L_o, I_p, L_p)$, we find the parameters that try to satisfy two kinds of constraint:

$$f_q(I_o, L_o, I_p, L_p) > f_q\big(I_o, L', I_p, L_p\big), \quad \forall L' \neq L_o, \tag{6}$$

$$f_q(I_o, L_o, I_p, L_p) > f_q\big(I_o, L_o, I_p, L'\big), \quad \forall L' \neq L_p. \tag{7}$$

In practice not all training data constraints can be satisfied so we instead optimize a *ranking loss* that tries to optimize the precision at the top of the ranked list of labels. Following the loss defined in Usunier et al. (2009b) and Weston et al. (2010) we would like to minimize (for a single training example):

$$\mathcal{E}\big(rank_o\big(f_q(x)\big)\big) + \mathcal{E}\big(rank_p\big(f_q(x)\big)\big),$$

where:

$$rank_o\big(f_q(x)\big) = \sum_{L' \neq L_o} I\big(f_q\big(I_o, L', I_p, L_p\big) \geq f_q(x)\big),$$

$$rank_p\big(f_q(x)\big) = \sum_{L' \neq L_p} I\big(f_q\big(I_o, L_o, I_p, L'\big) \geq f_q(x)\big),$$

i.e., $rank_o$ counts the number of labels that are ranked above the true object label and $rank_p$ counts the number of labels that are ranked above the true part label, and

$$\mathcal{E}(k) = \sum_{j=1}^{k} \alpha_j, \quad \text{with } \alpha_1 \geq \alpha_2 \geq \cdots \geq 0$$

is a function that transforms the ranks into a loss. If the first few indices of α index large values and smaller values follow this gives more weight to optimizing the top of the ranked list (Usunier et al. 2009b). In our experiments we set $\alpha_1 = 1$ and $\alpha_{j>1} = 0$ to optimize the proportion of top-ranked correct labels.

Object and part training algorithm. Following the authors of Weston et al. (2010) we optimize such a loss function via stochastic gradient descent by iterating over the following scheme:

1. Select a positive training quadruplet x at random.
2. Select at random either constraint (6) or (7).
3. Set $N = 0$.
4. Repeat:
   (a) Create a negative triplet $\tilde{x}$ by sampling a label L' at random and constructing $\tilde{x} = (I_o, L', I_p, L_p)$ for constraint (6) or $\tilde{x} = (I_o, L_o, I_p, L')$ for constraint (7).
   (b) $N = N + 1$
   until $f_q(x) > f_q(\tilde{x}) + 1$ or $N \geq \gamma$.
5. Make a stochastic gradient step to minimize $\mathcal{E}(\lfloor \frac{\gamma}{N} \rfloor)|1 - f_q(x) + f_q(\tilde{x})|_+$.
6. Enforce the constraints that each column $\|W_i\| \leq 1$, $\|V_i\| \leq 1$, $\forall i$ (this only involves updating the parameters that have changed in the gradient step).

Similarly to the method previously described in Sect. 4.1, step 4 approximates the calculation of $rank_o$ or $rank_p$ rather than explicitly calculating them: instead of counting all violating constraints one keeps sampling N times (up to a maximum of γ) until one finds a single violating constraint. The rank is then approximated with $\frac{\gamma}{N}$ instead of using the true rank. Intuitively, if we have to sample for a long time to find a violating constraint, the true rank must be relatively small. This results in practically useful algorithms in terms of training time.

Hyperparameters. We use an embedding dimension of $D = \{50, 100\}$ in all our experiments. For sampling negative samples, we choose $\gamma \in \{10, 100\}$ on various size of mini-batch $\{100, 1000\}$. The learning rates are set to $\{0.01, 0.001\}$ for W, U and $\{0.1, 0.01\}$ for V.

For the image mapping $\Phi(\cdot)$ we use a standard bag-of-visual-terms type representation, which has a sparse vector representation. In particular, we use the bag-of-terms feature setup of Grangier and Bengio (2008), which was shown to perform well on image ranking tasks. The image is first segmented into several overlapping square blocks at various scales. Each block is then represented by the concatenation of color and edge features. These are discretized into a dictionary of $N = 10{,}000$ blocks, by training k-means on a large corpus of images. Each image can then be represented as a bag of visual words: a histogram of the number of times each visual word was present in the image, yielding N dimensional vectors with an average of 245 non-zero values.

5 Related work

Part-based approaches. There has been extensive work on decomposing objects into parts, and using parts and their relations to model objects and their shapes. Usually, the "parts" in these approaches are not viewed as isolated objects by their own merits (like the flashlight of a camera), but rather capture recurring patterns in patches that cover parts of the images (like a bottom left side of a camera). For example, Agarwal et al. (2004) learned models of parts and their spatial relations from sets of patches in images, and then used them to label objects, such as cars in urban environments. The current work has a different aim of explicitly tagging the presence of parts in images. Another related work is by Crandall and Huttenlocher (2006), who used a Markov random field to learn a rich appearance and shape model with training data that only had class labels. Once again the parts learned in this work do not correspond to smaller objects but rather to patches that cover parts of objects.

Object detection using part or attribute-based models. Using parts or attributes has shown promising results for object detection. Farhadi et al. (2009) build an object representation where objects are described as a collection of semantically meaningful attributes e.g. parts, material. Also, Felzenszwalb et al. (2009) shows how to train state of the art object detectors which first localize parts of an object in combination with a whole object detector. These approaches differ from ours since we are focusing on classification and image retrieval and not on detection.

Image labeling with embedding-based models. The idea of using embedding algorithms for image labeling has been tried using several varying methods. PLSA (Monay and Gatica-Perez 2004) was applied but has been shown to perform worse than (non-embedding based) supervised ranking models such as the large margin classifier PAMIR (Grangier and Bengio 2008). Embedding for image retrieval (rather than annotation) using KCCA was also explored in Zhou et al. (1999). The most related work to ours is that of Weston et al. (2010)

which uses a ranking criterion optimizing the top of the ranked list like ours to label images where they obtained very good results compared to non-embedding approaches (such as PAMIR) on large scale tasks. None of these works however explored labeling parts in images. The current work aims to learn richer models which incorporate both image-label similarity functions and object-label to part-of-label similarity functions. We focus on the difficulty of training such models given a lack of large-scale supervision for the part-based tasks we are interested in.

6 Empirical evaluation

We first briefly give experimental results in an image annotation setup which labels *objects only* (Sect. 6.1). These results justify the general embedding approach and choice of loss function that we use in the rest of our experiments on objects and parts, and are a summary of the results from Weston et al. (2011). Then, we present new results for the object and part model we are interested in (Sect. 6.2).

6.1 Objects only

The approach we advocate in this setup is to use the embedding algorithm detailed in Sect. 2 with the WARP loss of Sect. 4.1. This method is called WSABIE (Web Scale Annotation By Image Embedding).

Dataset. ImageNet (Deng et al. 2009b) is a large-scale image dataset organized according to WordNet (Fellbaum 1998). Concepts in WordNet, described by multiple words or word phrases, are hierarchically organized. ImageNet is a growing image dataset that attaches quality-controlled human-verified images to these concepts. The version that was downloaded for these experiments had 4.1 million images and 15,952 classes. The data was split into 2.5 M images for training, 0.8 M for validation and 0.8 M for testing, removing duplicates between train, validation and test by throwing away test examples which had too close a nearest neighbor training or validation example in feature space.

Feature representations. As mentioned before, we will consider a bag-of-visual-terms type representation, but many other types of feature representation appear in the literature. We hence also explored the possibility that an ensemble of feature representations that can improve performance as has been shown before (Makadia et al. 2008). We thus combined multiple feature representations which are the concatenation of various spatial (Grauman and Darrell 2007) and multiscale color and texton histograms (Leung and Malik 1999) for a total of about 5×10^5 dimensions. We then perform Kernel PCA (Schoelkopf et al. 1999) on the combined feature representation using the intersection kernel (Barla et al. 2003) to produce a 1024 dimensional input vector for training WSABIE. We then train WSABIE as before.

Baselines. We compare the embedding approach WSABIE to several baselines: approximate k-nearest neighbors (k-NN), exact k-NN one-versus-rest linear large margin classifiers (One-Vs-Rest) of the form $f(I_o, L_o) = w_i^\top \Phi_{\mathcal{I}}(I_o)$ trained separately for each class L_o, or the same models trained with an AUC type ranking loss instead (sometimes called the multiclass loss). For all methods, hyperparameters are chosen via the validation set.

We tested approximate k-NN because k-NN is often not feasible. There are many flavors of approximation (see, e.g. Torralba et al. 2008). We chose the following: a random projection at each node of the tree is chosen with a threshold to go left or right that is the median of the projected training data to make the tree balanced. After traversing p nodes we arrive at a leaf node containing $t \approx n/2^p$ of the original n training points from which we calculate the nearest neighbors. Choosing p trades off accuracy with speed.

Metrics. We compared our models using the precision at the top k of the list (p@k). This metric gives more importance to the true annotations appearing in the top of the list and is defined as:

$$\text{p@k} = \frac{1}{N} \sum_{i=1}^{N} \frac{I\{\text{PredictedRank}(x_i) < k\}}{k},$$

where N is the size of the set, x_i is an image and its label, PredictedRank a function that returns the rank of x_i according to the considered score function and $I\{X\}$ returns the number of elements of the set X.

Results. The results of comparing all methods on ImageNet are summarized in Table 1. As expected, the choice of feature representation has a large impact on the results. The ensemble+KPCA features outperform Bag of Visterms independent of the learning algorithm. However, for both feature types WSABIE outperforms the competing methods tried. The choice of embedding dimension D for WSABIE is given in the Table. Smaller values give slightly worse results, e.g. $D = 100$ with Bag of Visterms gives 4.03 %.

We also compared different models trained with either WARP or AUC optimization: either the embedding approach of WSABIE ($D = 100$) or a full linear model. The results given in Table 2 show WARP gives superior performance.

Table 1 Summary of image annotation (object labeling only) test set results on ImageNet. Precision at 1 and 10 are given

Algorithm	Features used	p@1	p@10
Approx. k-NN	Bag of Visterms	1.55 %	–
Exact k-NN	Bag of Visterms	4.50 %	–
One-vs-Rest linear machine	Bag of Visterms	2.27 %	1.02 %
AUC ranking-loss linear machine	Bag of Visterms	3.14 %	1.26 %
WSABIE [$d = 500$]	Bag of Visterms	6.14 %	2.09 %
Exact k-NN	Ensemble+KPCA	7.73 %	–
WSABIE [$d = 1000$]	Ensemble+KPCA	10.03 %	3.02 %

Table 2 WARP vs. AUC optimization. We compared the two types of loss function WARP and AUC or two types of model: semantic embedding models and linear models. For both model types, WARP improves over AUC

Model	AUC p@1	WARP p@1
$f(I_o, L_o) = (V\Phi_I(I_o))^\top W_{\Phi_L(L_o)}$ [$D = 100$]	1.65 %	**4.03 %**
$f(I_o, L_o) = w_i^\top \Phi_I(I_o)$	3.14 %	**4.25 %**

6.2 Objects and parts

In this section we present results with our models that learn the semantics of both objects and their parts.

Notation summary

- $f_{IW}^{S} : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ returns a score given a quadruplet of images and annotations of a part and an object. This model uses the same semantic space for encoding the WordNet relationships and image visual similarities.
- $f_{IW}^{U} : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ uses different unshared semantic spaces for WordNet relationships and image visual similarities.
- $f_{I}^{S} : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ shares the same semantic space for part-of relationships and image visual similarities but does not use the Part-Object score f_L for ranking.
- $f_{I}^{U} : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ corresponds to the original WSABIE system.

Models. We consider two models denoted "Shared" and "Unshared". The "Shared" model shares the label embedding matrix W for both functions f_I and f_L, and the "Unshared" model has an additional label embedding matrix U for f_I^U (see Sect. 3).

The two models are considered in two different setups. In the first case, "Image-Word", the score of a quadruplet (I_o, L_o, I_p, L_p) is computed using images and word label-label interactions. The score function for the "Shared" model is:

$$f_{IW}^{S}(I_o, L_o, I_p, L_p) = f_I(I_o, L_o) + f_I(I_p, L_p) + \alpha f_L(L_o, L_p)$$

and for the "Unshared" model:

$$f_{IW}^{U}(I_o, L_o, I_p, L_p) = f_I^U(I_o, L_o) + f_I^U(I_p, L_p) + \alpha f_L(L_o, L_p),$$

where α is an hyperparameter set to 0.1 in our experiments.

In the second case (denoted "Image"), the score is given by image-label interactions only (it does not use the within-labels score f_L) in the "Shared" model:

$$f_I^S(I_o, L_o, I_p, L_p) = f_I(I_o, L_o) + f_I(I_p, L_p)$$

and in the "Unshared" model:

$$f_I^U(I_o, L_o, I_p, L_p) = f_I^U(I_o, L_o) + f_I^U(I_p, L_p).$$

As a baseline classifier, we also compare our models with a multiclass SVM trained in a one-vs-rest setting with LibLinear (Fan et al. 2008).

Metrics. As before, we compared our models using the precision at the top k of the list (p@k), as well as the mean average precision (MAP), which is defined as:

$$\text{MAP} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{PredictedRank}(x_i)},$$

where N is the size of the set, x_i is a quadruplet, PredictedRank a function that returns the rank of x_i according to the considered score function and $I\{X\}$ returns the number of elements of the set X.

Tasks. We consider three different tasks.

The first task, termed **Image Annotation**, is a standard image labeling task where our part-based architecture is not really useful, but still serves as a kind of sanity check of our system. Given one image I_o of an object or a part I_p, the model has to predict the correct label among $K = 1000$ different synsets. This task is used to assess the basic quality of our image labeling model f_I^U or f_I^S w.r.t. the SVM. Since the Image-Word models use $f_L(.,.)$ to compute the ranking score which has no relation whatsoever with the image property, we will not evaluate them in this task.

Our second task, termed **Part-Object Labeling**, is particularly of interest for our setting to evaluate the gain of using a joint prediction model using image and words as we propose. Given an object I_o and a part I_p, one has to predict the correct pair of labels (L_o, L_p) over $K^2 = 1$ M possible pairs.

Our third task, termed **Triplet**, is derived from the previous, but it assumes that either the object I_o or the object part I_p has already been labeled (or is known using prior knowledge). Hence, given a pair of images I_p and I_o, and the label of either the part L_p or the object L_o, the model has to correctly predict the missing label.

The two last tasks use the ranking functions f as defined above i.e. f_{IW}^U, f_{IW}^S, f_I^U and f_I^S depending on the setup of the models that are being evaluated. The best models are selected w.r.t. their mean rank error on the joint part-object labeling task by using f_{IW}^S, f_{IW}^U, f_I^S or f_I^U computed on 50 k quadruplets of the validation set from ImageNet+WordNet and ranked over 1 M pairs.

Training and testing. The difference between the Shared (Image) and Unshared (Image) appears during training. Considering **Image Annotation**, even if f_L is not used for computing the score at the end, the training process will perform a gradient step on W with respect to (L_p, L_o), (I_o, L_o) and (I_p, L_p) for the shared model while the unshared parameters only receive updates coming from (I_o, L_o) and (I_p, L_p).

Unshared (Image) is similar to the original WSABIE model and is presumably better than Shared (Image) which will push part-object word embeddings to be closer. For example in the Shared (Image) model, *car* and *wheel* will be close in the embedding space but having *wheel* ranked high for *car* is naturally not optimal.

6.3 ImageNet-WordNet data

Results on the test set of our custom data are presented in Table 3. On the standard **Image annotation** task, all considered models reach reasonable performance and compare nicely

Table 3 Summary of test set results on ImageNet-WordNet. Precision at 1 and 10, and Mean Average Precision (MAP) are given. (IW) resp. (I) refers to the (Image, Word) setup resp. (Image)

Models	Image annotation			Part-Object Labeling			Triplet		
	p@1	p@10	MAP	p@1	p@10	MAP	p@1	p@10	MAP
Shared (IW)	–	–	–	**11.48 %**	**3.40 %**	**0.1892**	26.31 %	**9.90 %**	0.5545
UnShared (IW)	–	–	–	10.01 %	3.02 %	0.1669	**33.13 %**	9.62 %	**0.5595**
Shared (I)	11.21 %	3.85 %	0.2021	5.13 %	1.84 %	0.0955	11.21 %	3.85 %	0.2021
UnShared (I)	**12.94 %**	**4.10 %**	**0.2219**	6.08 %	2.11 %	0.1118	12.94 %	4.10 %	0.2219
SVM	10.02 %	3.72 %	0.1864	–	–	–	10.02 %	3.72 %	0.1864

with the SVM. We would expect that the models using (I) would achieve the same value because on this particular task. However our models are selected using a validation score on the **Part-Object Labeling** and thus different configurations are picked, resulting in different performances.

On the **Triplet** task, models (I) have the same performance as for Image annotation because they are unable, by construction, to use the information coming from the already assigned label. Models (IW) using that information experience an increase in accuracy for all criteria: using the within-label score f_L is a major plus that these methods can exploit. They reach values of 33.13 % p@1 which on a task with 1,000 labels is a fairly remarkable achievement.

Results of the **Part-Object Labeling** task confirm the usefulness of using our joint scoring model. Indeed, both models (IW) using all scores clearly outperform the competing methods. On top of that, sharing embeddings during training brings an additional improvement because the labels have acquired richer representations through their multi-task training within f_I and f_L.

6.4 LHI data

We further evaluated our models on a set of objects pairs that were manually annotated as objects and their parts. To create this data, we used a set of images annotated by Labelme (2008) and parsed by the Lotus Hill Research Institute (LHI) (Yao et al. 2009).

Data collection. We downloaded the LHI data from yoshi.cs.ucla.edu/yao/data/. For every image, this dataset contains an annotation of the image and list of bounding boxes surrounding parts of the object together with the annotation of all parts. We manually matched these annotations with the list of WordNet synsets (for example by matching the term ‘camera’ to the relevant synset “n03358726”). We then intersected the list of synset-pairs which exist in ImageNet with the list of annotation pairs in the LHI data, and extracted images of parts from the bounding boxes. This yielded a set of image pairs, each having an object and its part, and each labeled with a synset that can be modeled using ImageNet. An example of such a set is depicted in Fig. 3.

The LHI data contains 489 image pairs (Object, Part) with 19 *part-of* relations. Note that 10 of those relations do not appear in our training set. A perfect example: “Propeller is *part-of* Wing is *part-of* Aircraft”. “Propeller is *part-of* Aircraft” is not in our training set of 771 relations but “Propeller is *part-of* Wing” and “Wing is *part-of* Aircraft” are. Moreover, it is also important to add that the patches representing objects are also usually in a much lower resolution than the images from ImageNet.

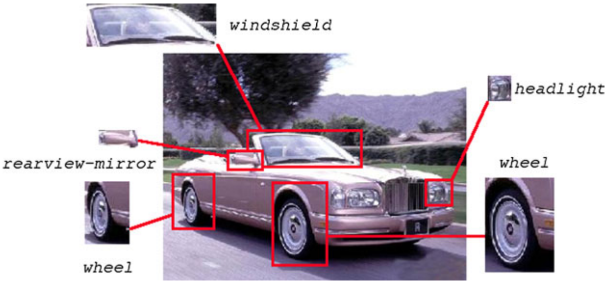


Fig. 3 Example from LHI data depicting an image of an object and its parts

Table 4 Summary of results on LHI data

Models	Image annotation			Part-Object Labeling			Triplet		
	p@1	p@10	MAP	p@1	p@10	MAP	p@1	p@10	MAP
Shared (IW)	–	–	–	**0.25 %**	**0.13 %**	**0.0075**	9.10 %	4.43 %	0.2044
UnShared (IW)	–	–	–	0.0 %	0.10 %	0.0043	**18.20 %**	**4.79 %**	**0.2778**
Shared (I)	**10.85 %**	**2.92 %**	**0.1728**	0.0 %	0.08 %	0.0039	10.85 %	2.92 %	0.1728
UnShared (I)	7.26 %	2.37 %	0.1288	0.0 %	0.03 %	0.0025	7.26 %	2.37 %	0.1288

Results. This evaluation set is harder than the previous one due to its real-world conditions. This is reflected in the results presented in Table 4. We can still draw the same conclusions as for the previous section. That is, using the within-label score improves performances i.e. (IW) is always better than (I). Furthermore, these results show that our proxy supervision is useful and allows the model to learn a prediction function that can nicely transfer to real-world data even though such data *has never been seen in training*.

6.5 WordNet tree and generalization to unseen relationships

The Image-Word models (Shared and Unshared) learn an embedding encoding the part-of relationships but one can think of directly using the WordNet tree.

Notation summary

- $f_{IO}^S : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ shares the same semantic space for part-of relationships and images visual similarities. An offset is added to the ranking score if the part-of relationship was present in the training set.
- $f_{IO}^U : \mathcal{I} \times \mathcal{L} \times \mathcal{I} \times \mathcal{L} \to \mathbb{R}$ corresponds to the original WSABIE system where an offset is added to the score in the same way as before in order to filter out the training relationships.

In order to use this prior in the Image models (Shared and Unshared), we add an offset to the original score if the part-of relationship is in the training set:

$$f_{IO}^S(I_o, L_o, I_p, L_p) = f_I^S(I_o, L_o, I_p, L_p) + O(L_p, L_o),$$
$$f_{IO}^U(I_o, L_o, I_p, L_p) = f_I^U(I_o, L_o, I_p, L_p) + O(L_p, L_o),$$

where

$$O(L_p, L_o) = \begin{cases} \lambda & \text{if } (L_p, L_o) \text{ is in the training relationships,} \\ 0 & \text{otherwise.} \end{cases}$$

In addition, we wanted to stress the ability of Image-Word models for generalizing to unseen part-of relationships. Indeed given ‘wheel is-part-of car’ and that ‘car’ shares visual similarities with ‘truck’, Image-Word models learn to push truck-wheel embeddings together even if ‘wheel is-part-of truck’ is not present in the training data.

The LHI dataset is particularly appropriate for these two experiments. All the models have been trained on 771 part-of relationships but 50 % of the part-of relationships in LHI (10/19) are not in those training relationships. When testing on the LHI dataset, we filtered out the results keeping all the part-of relationships among these 781 couples (771 training relationships and 10 unseen during training coming from LHI). Therefore, in the Image

Table 5 WordNet tree and unseen relationships Image-Word shared models allow to generalize to unseen relationships. Using the WordNet tree also improves the performance. (IO) refers to the original image where an offset was added to filter out relations both in the WordNet tree and the training set. Part-Object relationships are ranked among 781 couples

Models	Part-Object Labeling using WordNet		
	p@1	p@10	MAP
Shared (IW)	**3.06 %**	**2.45 %**	**0.1005**
UnShared (IW)	2.04 %	1.64 %	0.0670
Shared (IO)	1.43 %	1.17 %	0.0483
UnShared (IO)	1.23 %	1.45 %	0.0519

Table 6 Nearest neighbors of word embeddings learned with shared and unshared models. Shared embeddings exhibit visual similarities in addition to semantics e.g. strings of an instrument and laces of a shoe, a radio-phonograph is usually set up on a desk or a dresser

Toe		Radio-phonograph		Loudspeaker	
Shared	Unshared	Shared	Unshared	Shared	Unshared
footwear	shoe	dresser	tuner	lens	public address system
tongue	accelerator pedal	rotor head	amplifier	sustaining pedal	amplifier
stringed instrument	storage	firebox	public address system	public address system	tuner
shoe	boot	rear light	loudspeaker	optical instrument	radio-phonograph

models score (Shared and Unshared), an offset (we used $\lambda = 100$ in our experiments) was added if the part-of relationship was among the 771 training relationships. The Image-Word models (Shared and Unshared) are kept as before.

Results in Table 5 highlight the advantage of using Image-Word models, those are able to generalize to unknown relationships (10 relationships in LHI out of the training set) through word smoothing in the embedding space. Using the WordNet tree to filter out relationships among the 1 M possible couples (by keeping only 781 relationships) is also helpful as the performance increased drastically for all models (in comparison with results w/o using WordNet tree in Table 4).

6.6 Encoded semantics

Example word embeddings learned by our shared and unshared models on the ImageNet-WordNet data are given in Table 6. We observe that the embeddings capture the semantic structure of the labels. Besides, the shared model also encodes some information regarding the visual aspect of the object (or part) in its features. Hence, stringed instruments and shoes have strings and laces as parts, which are visually alike.

Figure 4 illustrates an application for which our model could readily be applied, which we could term *semantically augmented search*. Given a text query, our model can retrieve corresponding images using the f_I function, as standard *object only* image retrieval approaches do. However, for no additional cost, our model can also extend the original query

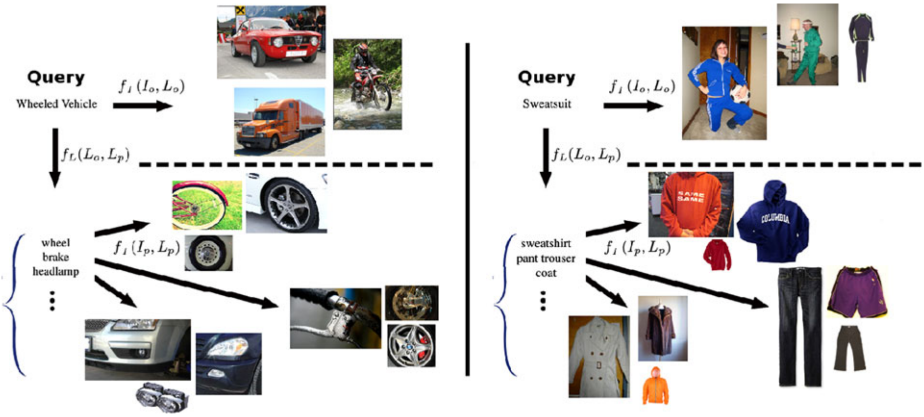


Fig. 4 Semantically augmented image search. For a given query ("wheeled vehicule" on the left and "Sweatsuit" on the right), our model can retrieve corresponding images using f_I (*above the dashed line*) but can also automatically extend the query to associated terms (object parts in this work) using f_L and hence retrieve semantically related images using f_I (*below the dashed line*). These results were obtained with the unshared model

by searching for semantically associated terms (which are only parts in our current work). The model uses first the f_L function to retrieve parts labels and finally return part images related to those by leveraging the f_I function. Therefore, our model can generalize an image query to search for other objects which have little or no visual similarity with the original object of interest but are semantically linked to it, hence improving the original search and browsing experience. This is related to recent efforts on attribute-based search pulled off by search engines.

7 Conclusion and future directions

This paper presented an original approach to learn to automatically tag images with their objects and their object-parts. In particular, we proposed a proxy supervision that allows to bypass the high cost of annotating such data. We illustrated this setting by introducing a new dataset created by connecting ImageNet and WordNet through part-based within-label relations. We also presented a model that can be trained under such an indirect supervision by jointly encoding both object image to object label, and object label to object part label similarities. Our experimental evaluation performed on our custom test set as well as on exhaustively annotated data showed that this model can be successfully trained under our proxy supervision.

Our idea of merging vision and semantic data is not limited to labeling objects and object parts. We chose this setting because it may be the most obvious and because it has many potential applications. However, our proxy supervision and our associated algorithm could also be used in other contexts and with other databases. For instance, one could think of connecting Labeled Faces in the Wild (LFW) (Huang et al. 2007) as the image data and Freebase (Bollacker et al. 2008) as the semantic graph among labels. Indeed, Freebase proposes a vast graph linking persons with various relations such as *married-to*, *son-of*, etc. that matches well the set of labels of LFW. By connecting these two resources, one could build up a model able to leverage some underlying connection between the members of a group picture to jointly label them. This is just an example of future directions/applications that could be explored with our new proxy supervision for image annotation.

Acknowledgements This work was supported by the DARPA Deep Learning Program, NSERC, CIFAR, the Canada Research Chairs, Compute Canada and by the French ANR Project ASAP ANR-09-EMER-001. Codes for the experiments has been implemented using both Torch (2011) and Theano (2010) Machine Learning libraries. Gal Chechik was supported by the Israeli science foundation Grant 1090/12, and by a Marie Curie reintegration grant PIRG06-GA-2009-256566.

References

Agarwal, S., Awan, A., & Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1475–1490.

Barla, A., Odone, F., & Verri, A. (2003). Histogram intersection kernel for image classification. In *International conference on image processing (ICIP)* (Vol. 2). 3:III–513–16.

Bengio, Y. (2008). Neural net language models. In *Scholarpedia* (Vol. 3, p. 3881).

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., & Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)*, Oral, June 2010.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD'08: proceedings of the 2008 ACM SIGMOD international conference on management of data* (pp. 1247–1250). New York: ACM.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on computational learning theory* (pp. 144–152). New York: ACM.

Collobert, R., Kavukcuoglu, K., & Farabet, C. (2011). Torch7: a matlab-like environment for machine learning. In *BigLearn, Advances in neural information processing systems workshops*.

Crandall, D., & Huttenlocher, D. (2006). Weakly supervised learning of part-based spatial models for visual object recognition. In *European conference on computer vision, 2006 (ECCV)* (pp. 16–29).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009a). ImageNet: a large-scale hierarchical image database. In *Computer vision and pattern recognition, 2009 (CVPR)*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009b). ImageNet: a large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J. (2008). LibLinear: a library for large linear classification. *Journal of Machine Learning Research*, *9*, 1871–1874.

Farhadi, A., Endres, I., Hoiem, D., & Forsyth, D. A. (2009). Describing objects by their attributes. In *Computer vision and pattern recognition, 2009 (CVPR)*.

Fellbaum, C. (Ed.) (1998). *WordNet: an electronic lexical database*. Cambridge: MIT Press.

Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D. (2009). Object detection with discriminatively trained part-based models. In *IEEE pattern analysis and machine intelligence (PAMI)*.

Grangier, D., & Bengio, S. (2008). A discriminative kernel-based model to rank images from text queries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(8), 1371–1384.

Grauman, K., & Darrell, T. (2007). The pyramid match Kernel: efficient learning with sets of features. *Journal of Machine Learning Research*, *8*, 725–760.

Herbrich, R., Graepel, T., & Obermayer, K. (2000). Large margin rank boundaries for ordinal regression. In *Advances in large margin classifiers* (Vol. 88, pp. 115–132).

Huang, G. B., Ramesh, M., Berg, T., & Learned-Miller, E. (2007). *Labeled faces in the wild: a database for studying face recognition in unconstrained environments*. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

Leung, T., & Malik, J. (1999). Recognizing surface using three-dimensional textons. In *Proc. of 7th Int'l Conf. on Computer Vision*, Corfu, Greece.

Makadia, A., Pavlovic, V., & Kumar, S. (2008). A new baseline for image annotation. In *European conference on computer vision (ECCV)*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39–41.

Monay, F., & Gatica-Perez, D. (2004). PLSA-based image auto-annotation: constraining the latent space. In *Proceedings of the 12th annual ACM international conference on multimedia* (pp. 348–351). New York: ACM.

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008). Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision*, *77*, 157–173.

Schoelkopf, B., Smola, A. J., & Müller, K. R. (1999). Kernel principal component analysis. In *Advances in kernel methods: support vector learning* (pp. 327–352).

Suchanek, F., Kasneci, G., & Weikum, G. (2008). Yago: a large ontology from wikipedia and wordnet. *Journal of Web Semantics*, *6*, 203–217.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, *58*(1), 267–288.

Torralba, A. B., Fergus, R., & Weiss, Y. (2008). Small codes and large image databases for recognition. In *Computer vision and pattern recognition, 2008 (CVPR)*. Los Alamitos: IEEE Comput. Soc.

Usunier, N., Buffoni, D., & Gallinari, P. (2009a). Ranking with ordered weighted pairwise classification. In L. Bottou & M. Littman (Eds.), *Proceedings of the 26th international conference on machine learning*, Montreal, June 2009 (pp. 1057–1064). Madison: Omnipress.

Usunier, N., Buffoni, D., & Gallinari, P. (2009b). Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th annual international conference on machine learning* (pp. 1057–1064).

Weston, J., Bengio, S., & Usunier, N. (2011). Large scale image annotation: learning to rank with joint word-image embeddings. In *International joint conference on artificial intelligence (IJCAI)*.

Weston, J., Bengio, S., & Usunier, N. (2010). Large scale image annotation: learning to rank with joint word-image embeddings. *Machine Learning*, *81*, 21–35.

Yao, B., Yang, X., & Wu, T. (2009). Image parsing with stochastic grammar: the lotus hill dataset and inference scheme. In *Computer vision and pattern recognition workshops, 2009. CVPR workshops 2009* (p. 8). New York: IEEE Press.

Yao, B., Yang, X., & Zhu, S.-C. (2007). Introduction to a large-scale general purpose ground truth database: methodology, annotation tool and benchmarks. In *Proceedings of the 6th international conference on energy minimization methods in computer vision and pattern recognition* (pp. 169–183).

Zhou, Z. H., Zhan, D. C., & Yang, Q. (1999). Semi-supervised learning with very few labeled training examples. In *Proceedings of the national conference on artificial intelligence* (Vol. 22, p. 675). Menlo Park: AAAI Press. Cambridge: MIT Press, 2007.