Understanding Semantic Association Between Images and Text

Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Engineering

by

Yashaswi Verma
201199649
yashaswi.verma@research.iiit.ac.in

International Institute of Information Technology
(Deemed to be University)
Hyderabad - 500 032, INDIA
July 2017

International Institute of Information Technology

Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Understanding Semantic Association Between Images and Text" by Yashaswi Verma, has been carried out under my supervision and is not submitted elsewhere for a degree.

31 July 2017

Date

Adviser: C. V. Jawahar

To

The Divine Light

Acknowledgments

I would like to sincerely thank everyone who guided, helped and supported me in this part of my life's journey.

Abstract

Since the last two decades, vast amounts of digital data have been created, a large portion of which is the visual data comprising images and videos. To deal with this large amount of data, it has become necessary to build systems that can help humans to efficiently organize, index and retrieve from such data. While modern search engines are quite efficient in text-based indexing and retrieval, their "visual cortex" is still evolving. One way to address this is to enable similar technologies as those used for textual data for archiving and retrieving the visual data. Also, in practice people find it more convenient to interact with visual data using text as an interface rather than a visual interface. However, this in turn would require describing images and videos using natural text. Since it is practically infeasible to annotate these large visual collections manually, we need to develop automatic techniques for the same.

In this thesis, we present our attempts towards modelling and learning semantic associations between images and different forms of text such as labels, phrases and captions. Our first problem is that of tagging images with discrete semantic labels, also called image annotation. To address this, we describe two approaches. In the first approach, we propose a novel extension of the conventional weighted k-nearest neighbour algorithm that tries to address the issues of class-imbalance and incomplete-labelling that are quite common in the image annotation task. In the second approach, we first analyze why the conventional SVM algorithm, despite its strong theoretical properties, does not achieve as good results as nearest neighbour based methods on the image annotation task. Based on this analysis, we propose an extension of SVM by introducing a tolerance parameter into the hinge-loss. This additional parameter helps in making binary models tolerant to practical challenges such as incomplete-labelling, label ambiguity and structural overlap. Next we target the problem of image captioning and caption-based image retrieval. Rather than using either individual words or entire captions, we propose to make use of textual phrases for both these tasks; e.g., "aeroplane at airport", "person riding", etc. These phrases are automatically extracted from available data based on linguistic constraints. To generate a caption for a new image, first the phrases present in the neighbouring (annotated) images are ranked based on

visual similarity. These are then integrated into a pre-defined template for caption generation. During caption based image retrieval, a given query is first decomposed into such phrases, and images are then ranked based on their joint relevance with these phrases. Lastly, we address the problem of cross-modal image-text retrieval. For this, we first present a novel Structural SVM based formulation for this task. We show that our formulation is generic, and can be used with a variety of loss functions as well as feature vector based representations. Next, we try to model higher-level semantics in multi/cross-modal data based on shared category information. For this, we first propose the notion of cross-specificity, and then present a generic framework based on cross-specificity that can be used as a wrapper function over several cross-modal matching approaches, and helps in boosting their performance on the cross-modal retrieval task.

We evaluate the proposed methods on a number of popular and relevant datasets. On the image annotation task, we achieve near state-of-the-art results under multiple evaluation metrics. On the image captioning task, we achieve superior results compared to conventional methods that are mostly based on visual cues and corpus statistics. On the cross-modal retrieval task, both our approaches provide compelling improvements over baseline cross-modal retrieval techniques.

Contents

List of Figures

List of Tables

Publications

Part of the work described in this thesis has previously been presented in the following publications.

Journal:

1. **Yashaswi Verma** and C. V. Jawahar. Image Annotation by Propagating Labels from Semantic Neighbourhoods. In International Journal of Computer Vision (IJCV), 2017.

2. **Yashaswi Verma** and C. V. Jawahar. A Support Vector Approach for Cross-Modal Search of Images and Texts. In Computer Vision and Image Understanding (CVIU), 2017.

Conference:

1. **Yashaswi Verma** and C. V. Jawahar. A Probabilistic Approach for Image Retrieval Using Descriptive Textual Queries. In ACM International Conference on Multimedia (ACM MM), 2015.

2. **Yashaswi Verma** and C. V. Jawahar. Im2Text and Text2Im: Associating Images and Texts for Cross-Modal Retrieval. In British Machine Vision Conference (BMVC), 2014.

3. **Yashaswi Verma** and C. V. Jawahar. Exploring SVM for Image Annotation in Presence of Confusing Labels. In British Machine Vision Conference (BMVC), 2013.

4. **Yashaswi Verma**, Ankush Gupta, Prashanth Mannem and C. V. Jawahar. Generating Image Descriptions Using Semantic Similarities in the Output Space. In IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW), 2013.

5. **Yashaswi Verma** and C. V. Jawahar. Image Annotation Using Metric Learning in Semantic Neighbourhoods. In European Conference on Computer Vision (ECCV), 2012.

6. Ankush Gupta, **Yashaswi Verma** and C. V. Jawahar. Choosing Linguistics over Vision to Describe Images. In Association for the Advancement of Artificial Intelligence (AAAI), 2012.

Other publications during PhD which are not part of this thesis are as follows:

1. **Yashaswi Verma** and C. V. Jawahar. A Robust Distance with Correlated Metric Learning for Multi-Instance Multi-Label Data. In ACM International Conference on Multimedia (ACM MM), 2016.

2. **Yashaswi Verma** and C. V. Jawahar. Exploring Locally Rigid Discriminative Patches for Learning Relative Attributes. In British Machine Vision Conference (BMVC), 2015.

3. Ramachandruni N. Sandeep, **Yashaswi Verma** and C. V. Jawahar. Relative Parts: Distinctive Parts for Learning Relative Attributes. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

4. **Yashaswi Verma** and C. V. Jawahar. Neti Neti: In Search of Deity. In Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP), 2012.

Chapter 1

Introduction

As humans we love photographs. These play an integral role in our lives and help us in remembering and sharing moments. The last decade has witnessed an explosion of photographs (and multimedia content in general) on the Internet. There are two primary reasons for this. First is the availability of online photo-sharing websites such as Flickr, Picasa, Instagram and Facebook that allow people to upload and share practically any number of photographs almost free-of-cost. And second is the rapid technological advancements in portable photo-capturing devices such as mobile phones and digital cameras. As a result, it has become necessary to develop new technologies that can help in efficiently archiving and accessing such large collections of photographs.

Along with visual signals, natural language is another popular mode of communication among humans. Modern search engines are quite efficient in text-based indexing and retrieval, hence it is a natural choice to enable similar technologies for archiving and retrieving images. However, this in turn requires describing images using natural text. Since it is practically infeasible to describe the ever-growing image collections manually, we need to develop automatic techniques for the same.

To automatically describe images using natural text requires an automatic visual system. The difficulty in building such a system comes from several aspects. One of them is the well-known "semantic-gap". There is a fundamental difference between how a machine/computer perceives an image versus how we do. For a computer, an image is simply a chunk of numbers, whereas for us it carries semantic meaning. Moreover, we can also relate several aspects that are not even visible in a photograph. Another is the extent of description required for an image while describing it automatically. The content of an image can be a full description written in prose (i.e., the adage "a picture is worth a thousand words"), or might simply have a few keywords describing spatial, temporal, or emotional aspects. While people are generally quite efficient in describing things at multiple levels of granularity, computers work based on

very precise instructions. Such challenges make it obvious that to automatically describe what is (and what is not) visible in an image using a sensible description is a non-trivial task, and is thus considered as the holy grail of the broad area of computer vision.

1.1 The Context

In order to build an autonomous visual system, it is necessary to develop new methods that would allow a machine to automatically learn new concepts over time based on empirical evidence, rather than a human hard-coding everything. Traditional research in this direction started by targeting fundamental tasks such as classification of single objects. Though it is one of the most basic and crucial steps towards automatically understanding images, it ignores the fact that a real-world image usually contains much more than a single object; it contains multiple objects interacting with each other in a specific context. Thus evolved the problem of image annotation, where an image needs to be tagged with a set of multiple labels describing its semantics in terms of context (or scene), the objects present, their attributes, and also their actions. This was initially approached by adopting ideas from the linguistics domain, by either mapping it as a problem of *translating* an image into a set of labels [24, 92] or learning relevance between image features and textual labels [28, 53, 70]. Motivated by the initial success of such methods, people started investigating more complex problems that further deepened the bonding between images and text, such as learning context in the form of prepositions [38], and training classification models for higher-level concepts such as visual phrases [117] that jointly learn interaction between two objects (e.g., "person on bicycle") rather than individual objects. These methods demonstrated that even for complex images, such information can be extracted under certain practical constraints, and thus can be useful in leveraging the gap between visual perception and semantic grounding. We believe this was the period when the interplay between vision and text witnessed a dramatic increase in attention, with more and more papers targeting this multi-disciplinary area of research.

After achieving certain milestones in the form of reasonably predicting a single label to multiple labels to prepositions and visual phrases, naturally the next extension then was to automatically describe images using complete captions by adapting well-known practices from Computer Vision and Natural Language Processing (NLP). This problem soon gained popularity and was targeted in various forms that included generating novel captions using fixed templates [68], and retrieving existing captions/descriptions in either multi-modal [27, 69, 101] or cross-modal [113] set-ups. The year 2012 ended

with the pioneering paper [67] that demonstrated the success of a deeply trained neural network model on a large scale image classification task. With quick advancements in deep learning based techniques, all the above mentioned tasks attained significant boosts in quantitative performances, be it image annotation [45, 57, 133], image captioning [20, 62, 146] or cross-modal retrieval [131, 156]. As a result, a few newer problems and applications have surfaced in this domain during the last couple of years, such as referring expression generation [65] and automatically answering a question specific to an image [3]. Many of these make use of ideas/approaches from the problems spanning across image classification to image captioning, and assume them to be nearly solved. However, this assumption is not really true, and there are still several challenges that remain to be addressed even in these core problems of associating images and text. Four such important challenges that we have targeted in this thesis are:
(1) To achieve good performance on rare concepts that are usually more informative and distinctive than the frequent ones (without compromising on the frequent concepts), and to learn good models from data when the ground-truth is missing/confusing (Chapter 3).
(2) To seamlessly use data from different sources/views and effectively combine them to both generate semantic captions for images and retrieve images given caption-based queries (Chapter 4).
(3) To extend classification based techniques that provide max-margin guarantees and good generalization for the cross-modal retrieval task (Chapter 5).
(4) To design generic models that can act as wrappers over existing techniques and help in boosting their performance (Chapter 6).

Specifically, our interest is to develop new approaches to automatically describe the semantics of an image using natural text, and to perform retrieval on a collection of images given a textual query. We will consider textual data to be in different forms; e.g., labels, phrases, or captions/descriptions. For simplicity, below we define the problems that we are interested in.

1.2 Problems of Interest

We will be focusing on various sub-problems in the broad context of understanding image semantics (Figure 1.1), as described below:

1. **Image annotation:** This aims at associating a set of semantic labels with an image, and is usually addressed as a multi-label ranking/classification task.

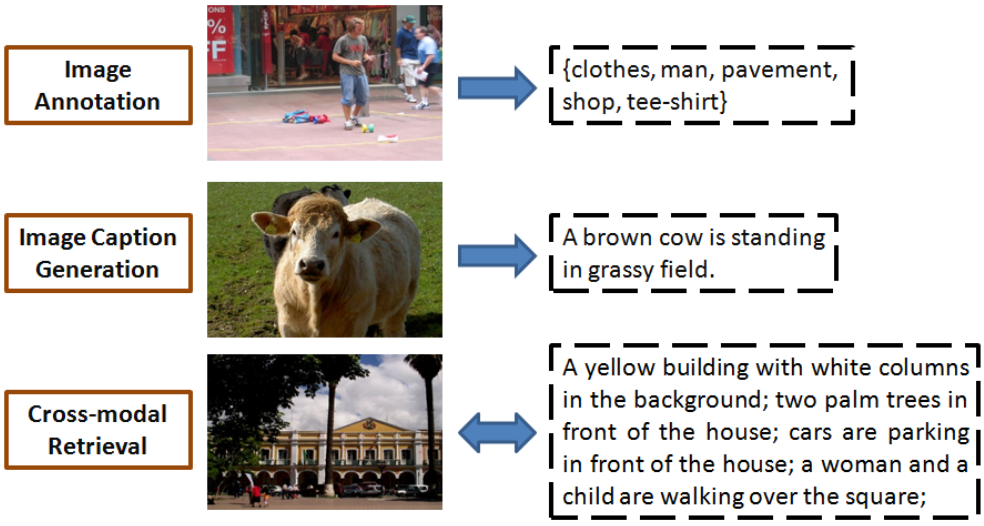


Figure 1.1 An overview of some of the sub-problems related to understanding the semantics of an image that we are interested in.

2. **Image caption generation:** Here, an image is described using an automatically generated short caption/sentence that is usually a few (∼10–15) words long. These captions usually contain some (2–3) objects (nouns), their attributes (properties/adjectives), actions (verbs), and relationships among them (prepositions).

3. **Cross-modal Image-Text retrieval:** Given a query in one modality, here the goal is to retrieve semantically relevant sample(s) from another modality. In other words, this assumes that the query and retrieval set are from different modalities. E.g., given a query caption, retrieve a semantically relevant image from a database of only images with no associated captions, and vice-versa.

1.3 Organization

The organization of the thesis is as follows:

- In Chapter 2, we discuss some of the tools and techniques that motivated and helped us to develop and evaluate new solutions for the problems of our interest.

- In Chapter 3, we describe our approaches for addressing the problem of image annotation. As our first approach, we propose a new nearest neighbour based model that tries to address the issues of class-imbalance and incomplete-labeling in the annotation task [140, 144]. As our second

approach, we propose an SVM based model that tries to make the conventional SVM tolerant against the issues such as incomplete-labeling, label ambiguity and structural overlap [141].

- In Chapter 4, we describe our approaches for addressing the problems of image caption generation [39, 139] and image retrieval using caption based queries [143]. For both these problems, our approaches rely on textual phrases extracted from the available data, and make use of a phrase relevance prediction model that computes the joint relevance of an image with a textual phrase.

- In Chapter 5, we describe our approach for cross-modal image-caption retrieval [142, 145] that is based on a novel Structural SVM based formulation. We show that our formulation is generic, and can be used with a variety of loss functions as well as feature vector based representations.

- In Chapter 6, we first propose the notion of cross-specificity, that models higher-level semantics in multi/cross-modal data based on shared category information. Then we present a generic framework based on cross-specificity that can be used as a plug-in/wrapper over several cross-modal matching approaches, and helps in boosting their performance on the cross-modal retrieval task.

- In Chapter 7, we discuss the summary, some directions for future research and conclusions based on this thesis.

In Chapters 3, 4, 5 and 6, we analyze different aspects of our approaches, and also compare them with other competing approaches. It is important to note that in practice, each of these problems is addressed individually. Hence, the datasets and experimental set-ups will vary across different chapters.

Chapter 2

Background

In this chapter, we present a brief background for some of the techniques that we adopted/adapted during the course of designing and/or evaluating our approaches for different tasks.

2.1 Some Machine Learning Techniques

Here we will discuss two broad categories of machine learning techniques. Under the first category, we present distance metric learning for k-nearest neighbours (kNN) based methods. In this, we discuss a commonly followed approach in metric learning, along with a popular metric learning algorithm [151]. Under the second category, we present an overview of three popular support vector based methods: Support Vector Machine (or SVM) [16], Ranking SVM [56] and Structural SVM [132], and also discuss connections among them.

2.1.1 Distance Metric Learning

In metric learning, the goal is to adapt some pairwise metric function that suits the problem of interest [8]. This is usually done using pair or triplet-based constraints brought by training samples. These constraints are of the following form:

- Must-link / cannot-link constraints (also called positive/negative pairs):

$$\mathcal{S} = \{(x_i, x_j) : x_i \text{ and } x_j \text{ should be similar}\},$$
$$\mathcal{D} = \{(x_i, x_j) : x_i \text{ and } x_j \text{ should be dissimilar}\}.$$

- Relative constraints (also called triplets):

$$\mathcal{R} = \{(x_i, x_j, x_k) : x_i \text{ should be more similar to } x_j \text{ than to } x_k\}.$$

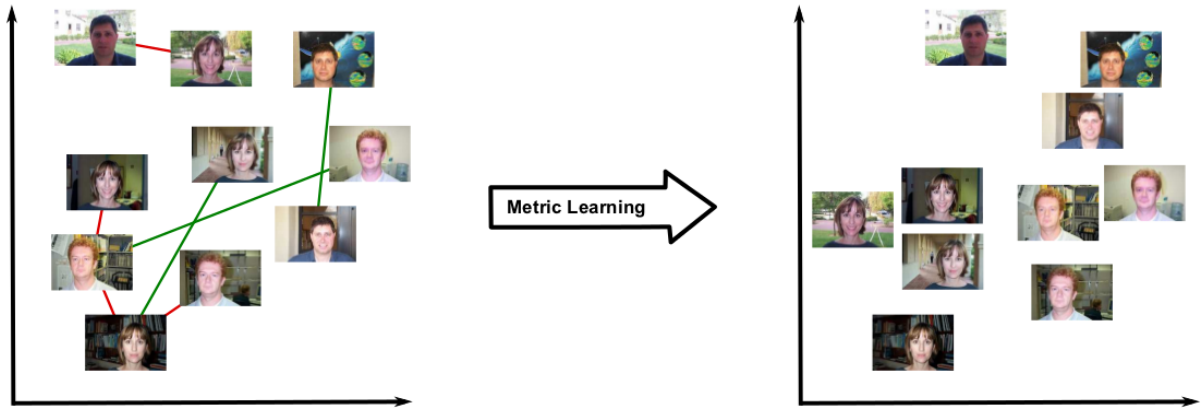


Figure 2.1 An illustration of metric learning applied to a face recognition task [8]. For simplicity, images are represented as points in 2 dimensions. Pairwise constraints, shown in the left pane, are composed of images representing the same person (must-link, shown in green) or different persons (cannot-link, shown in red). We wish to adapt the metric so that there are fewer constraint violations (right pane). Images are taken from the Caltech Faces dataset.

In order to approximate the underlying semantics, a metric learning algorithm learns the parameters of the metric such that the above constraints are satisfied as closely as possible (as illustrated in Figure 2.1). Typically, a metric learning approach is formulated as an optimization problem of the following form:

$$\min_{M} \lambda \, \mathcal{R}(M) + (1 - \lambda) \, l(M, \mathcal{S}, \mathcal{D}, \mathcal{R}) \tag{2.1}$$

where M denotes the parameters of the learned metric, $\mathcal{R}(M)$ denotes some regularization function on M, $l(M, \mathcal{S}, \mathcal{D}, \mathcal{R})$ denotes a loss function on the constraints of the training pairs and $\lambda \in [0, 1]$ takes care of the trade-off between regularization term and loss term. A large category of metric learning formulations essentially differ by their choice of metric, regularizer, constraints and loss functions. Once the metric is learned, it is expected to improve the performance of a metric-based algorithm. In most of the cases, this is some kNN based method, but may also be a clustering algorithm, a ranking algorithm, etc. Figure 2.2 summarizes the common process in metric learning.

Most existing metric learning algorithms aim at achieving either improved accuracy or performance on some specific task. However, each of these algorithms can be characterized based on some intrinsic properties, such as type of distance metric, applicability to unlabelled data, generalization guarantees, etc. Figure 2.3 summarizes five key properties of metric learning algorithms, that one can look for while applying some particular method to a given problem.

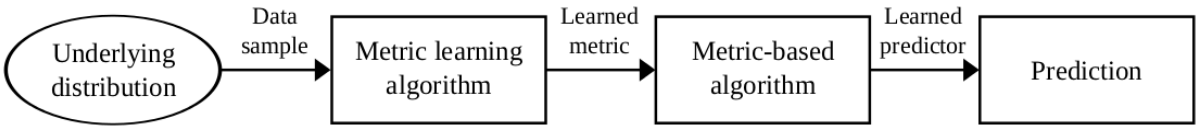


Figure 2.2 The common process in metric learning [8]. A metric is learned from training data and plugged into an algorithm that outputs a predictor (e.g., a classifier, a regressor, a recommender system, etc.) which hopefully performs better than a predictor induced by a standard (non-learned) metric.

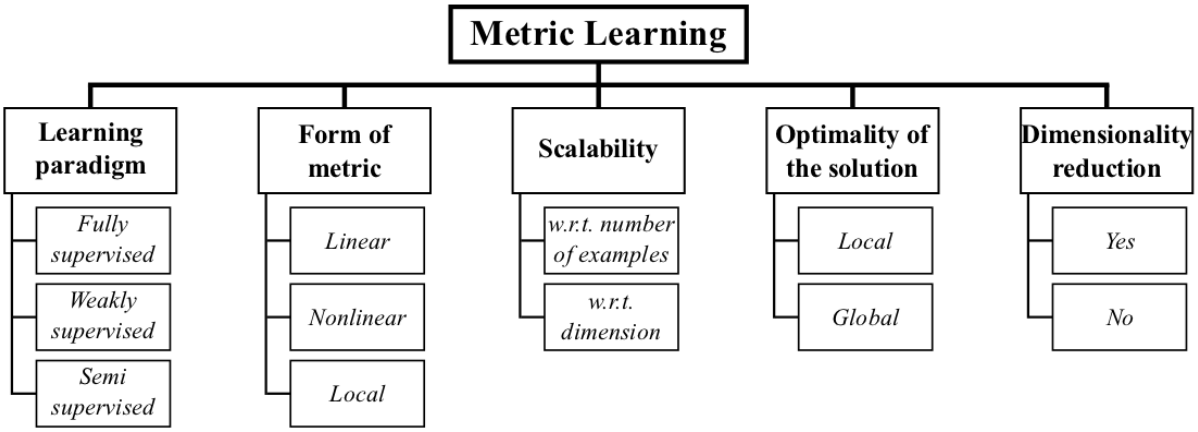


Figure 2.3 Five key properties of metric learning algorithms [8].

2.1.1.1 LMNN

To get an idea about metric learning, now we will discuss a popular kNN-based metric learning algorithm, called Large Margin Nearest Neighbour (or LMNN) algorithm as introduced in [151, 152]. Here, the goal is to learn a Mahalanobis metric M such that the performance of kNN classification is improved. Let us assume a training set $\{(\mathbf{x}_i, y_i)\}$ of n examples, where $\mathbf{x}_i \in \mathbb{R}^d$ denotes a sample and $y_i \in \{1, 2, \ldots, C\}$ denotes its category/class from C classes. Given two samples $\mathbf{x}_i$ and $\mathbf{x}_j$, the Mahalanobis distance between them is given by:

$$d_M(\mathbf{x}_i, \mathbf{x}_j) = \big((\mathbf{x}_i - \mathbf{x}_j)^T M (\mathbf{x}_i - \mathbf{x}_j)\big)^{\frac{1}{2}} \tag{2.2}$$

where M is a symmetric positive semidefinite matrix ($M \succeq 0$). In the above equation, if M is set to the identity matrix, then it reduces to the Euclidean distance.

LMNN enforces the local neighbourhood of a sample to belong to the same category. To do this, for a given sample, its neighbours with a similar label are pulled closer and those with different labels are

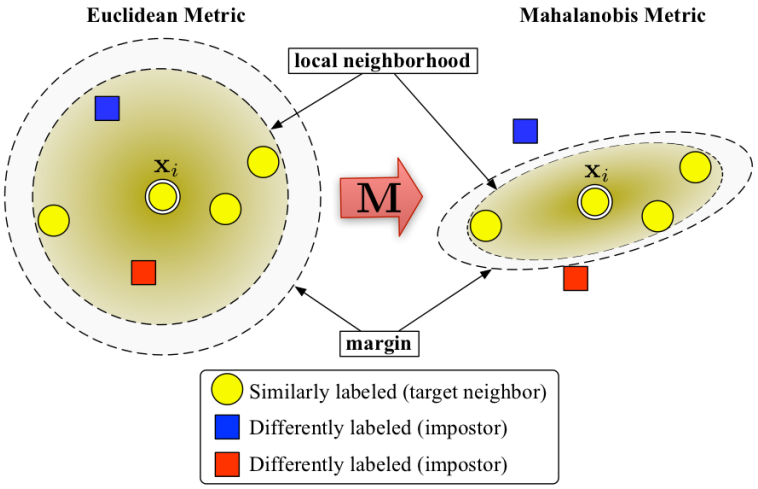


Figure 2.4 An illustration of a data set before and after applying LMNN [103]. The circles represent points of equal distance to the sample $\mathbf{x}_i$. The Mahalanobis metric rescales directions to push impostors further away than target neighbours by a large margin.

pushed away. One of the advantages of LMNN is that it makes use of local neighbourhood information to learn a global metric, which makes it more generalizable than other competing methods.

In LMNN, the classification error of kNN is approximated using a convex loss function. For a given sample $\mathbf{x}_i$, its *target* neighbours are defined as its k nearest samples from the same category, and *impostors* as its neighbours from other categories that are closer than target neighbours. Using this information, a Mahalanobis metric is learned such that each sample is closer to its target neighbours than impostors by a margin. Let $\mathbf{x}_j$ and $\mathbf{x}_k$ be a target neighbour and an impostor for $\mathbf{x}_i$ respectively, then this constraint can be expressed as:

$$d_M^2(\mathbf{x}_i, \mathbf{x}_k) - d_M^2(\mathbf{x}_i, \mathbf{x}_j) \geq 1 \tag{2.3}$$

As illustrated in Figure 2.4, the above constraints are enforced only on local neighbours. Here, all samples that lie on the circle have equal distance from $\mathbf{x}_i$. We can observe that under the learned Mahalanobis metric, this circle gets transformed into an ellipsoid, which in turn causes imposters to be farther from the target neighbours. Let $\mathcal{R} = \{(i, j, k) : j \rightsquigarrow i, y_i = y_j \neq y_k\}$ be a set of triples that denote relative constraints defined on the basis of similar and dissimilar class labels, then the optimization

problem of LMNN is given by:

$$\begin{aligned} \min_{M} \sum_{j \rightsquigarrow i} d^2_M(\mathbf{x}_i, \mathbf{x}_j) + \mu \sum_{(i,j,k) \in \mathcal{R}} \xi_{i,j,k} \\ s.t. : d^2_M(\mathbf{x}_i, \mathbf{x}_k) - d^2_M(\mathbf{x}_i, \mathbf{x}_j) \geq 1 - \xi_{i,j,k} \\ \xi_{i,j,k} \geq 0 \\ M \succeq 0 \end{aligned} \tag{2.4}$$

Here, the first term tries to pull target neighbours closer, the second term penalizes violations of the constraint in Eq. 2.3, $\mu > 0$ manages trade-off between the two terms, $\xi_{i,j,k}$ denotes slack variable, and the last constraint implies that M is constrained to be a positive semi-definite matrix. The above optimization problem has $O(kn^2)$ constraints, along with the positive semidefinite constraint of a $d \times d$ matrix M. To handle large datasets on the order of tens of thousands of samples, the authors of [151] developed a special purpose sub-gradient descent solver [1]. In [151], it was also stated that the optimization problem is not sensitive to the exact choice of the trade-off parameter μ, thus it is usually set to 1.

In practice, LMNN generally performs quite well and has been adapted/extended for several problems other than (single-label) classification, such as multi-task learning [103], multi-label learning [140], etc. However, it is sometimes sensitive to the choice of target neighbours, and also sometimes prone to overfitting due to absence of explicit regularization on M.

2.1.2 Support Vector Based Methods

2.1.2.1 Support Vector Machine

A large number of machine learning problems can be posed as a classification task. Given a sample, here the goal is to assign/predict a category from a finite set of discrete categories. A popular approach to do this is by learning a category specific classifier in a one-versus-rest set-up using SVM [16].

Let us assume a training set $\{\mathbf{x}_i, y_i\}$ of n examples, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$ denotes whether or not it belongs to a particular category $c \in \{1, \ldots, C\}$. In SVM, the goal is to learn a linear function

$$r_{svm}(\mathbf{x}_i) = \mathbf{w} \cdot \mathbf{x}_i \tag{2.5}$$

[1] Available at `http://www.cs.cornell.edu/~kilian/code/code.html`

for each category such that the maximum number of the following constraints are satisfied:

$$\mathbf{w} \cdot \mathbf{x}_i > 0, \;\; if \;\; y_i = +1$$
$$\mathbf{w} \cdot \mathbf{x}_i < 0, \;\; if \;\; y_i = -1$$

Here, $\mathbf{w} \in \mathbb{R}^d$ is the category-specific parameter vector. The above set of constraints can be re-written as

$$y_i(\mathbf{w} \cdot \mathbf{x}_i) > 0 \tag{2.6}$$

Since it is a hard problem to enforce this constraint on every sample, its solution is approximated by introducing non-negative slack variables $\xi_i \; \forall i$.

$$y_i(\mathbf{w} \cdot \mathbf{x}_i) \geq 1 - \xi_i \tag{2.7}$$

This leads to the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{w}, \xi \geq 0} \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i \\ &s.t. \;\; y_i(\mathbf{w} \cdot \mathbf{x}_i) \geq 1 - \xi_i \;\; \forall i \end{aligned} \tag{2.8}$$

Here, $||\cdot||^2$ denotes squared L_2 norm and $\lambda > 0$ is a constant that takes care of the trade-off between the two terms. The first term acts as a regularizer on $\mathbf{w}$, and the second term is the loss function (also called hinge-loss) that penalizes violation of the constraint in Eq. 2.7. Solving this leads to finding a hyper-plane $\mathbf{w}$ that best separates the positive and negative samples with maximum margin. Later on, given a new sample $\mathbf{x}$, we predict whether it belongs to a particular category or not based on $y = sign(r_{svm}(\mathbf{x}))$.

SVM provides several practical advantages such as a convex optimization problem, large margin guarantees, good generalization and scalability, and fast testing time. As a result, it has been found to achieve good quantitative results in a variety of categorization problems, and is considered as a de facto baseline in almost all such tasks.

2.1.2.2 Beyond SVM: Ranking SVM

One of the constraints in SVM is that it considers just one sample at a time and performs classification. However, in some situations, it is more important (or less ambiguous) to rank a set of objects rather than to classify them individually. A popular example is the ranking of web pages based on their

relevance to a given query. A simplified version of this problem can be to rank a pair of objects for a particular class; e.g., as done in the case of relative attributes [105]. This is popularly addressed under the Ranking SVM [56] framework.

Let us assume a training set consisting of n ordered sample pairs $\{(\mathbf{x}_i, \mathbf{x}_j)\}$ such that the sample $\mathbf{x}_i$ should be ranked before $\mathbf{x}_j$. Using these, the goal is to learn a ranking function that can predict the relative ranking between an unseen pair of samples. Under the assumption that this is a linear function, it is defined as:

$$r_{rsvm}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_j \tag{2.9}$$

such that the maximum number of the following constraints are satisfied:

$$\mathbf{w} \cdot \mathbf{x}_i > \mathbf{w} \cdot \mathbf{x}_j \;\; \forall(\mathbf{x}_i, \mathbf{x}_j) \tag{2.10}$$

Since this is an NP-hard problem, its relaxed version is solved by introducing non-negative slack variables ξ_{ij} $\forall(\mathbf{x}_i, \mathbf{x}_j)$.

$$\mathbf{w} \cdot \mathbf{x}_i - \mathbf{w} \cdot \mathbf{x}_j \geq 1 - \xi_{ij} \tag{2.11}$$

$$or, \;\; \mathbf{w} \cdot (\mathbf{x}_i - \mathbf{x}_j) \geq 1 - \xi_{ij} \tag{2.12}$$

This gives us the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{w}, \xi \geq 0} \frac{\lambda}{2} ||\mathbf{w}||^2 + \frac{1}{n} \sum \xi_{ij} \\ &s.t. \;\; \mathbf{w} \cdot (\mathbf{x}_i - \mathbf{x}_j) \geq 1 - \xi_{ij} \;\; \forall(\mathbf{x}_i, \mathbf{x}_j) \end{aligned} \tag{2.13}$$

From this, we can observe that the optimization problem is equivalent to that of classification SVM (as discussed above) on pairwise difference of samples. However, in this case, the margin denotes the distance between the closest two projections within all target rankings. This geometric difference between SVM and Ranking SVM is illustrated in Figure 2.5.

2.1.2.3 Beyond SVM and Ranking SVM: Structural SVM

As discussed above, both SVM and Ranking SVM can be thought of as performing a binary classification on a single sample and a pair of samples respectively. However, this becomes prohibitive when (1) the number of categories is exponentially large, and (2) the categories encode higher-level structure rather than being just simple labels. To overcome these, Structural SVM was introduced in [132] that

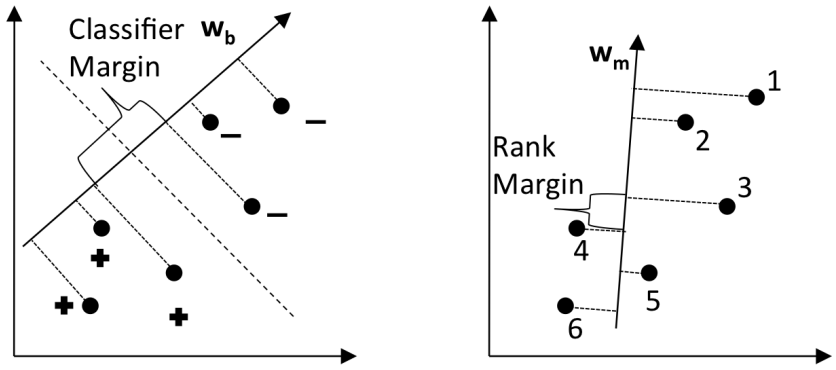


Figure 2.5 Distinction between learning a wide-margin ranking function using Ranking SVM (right) that enforces the desired ordering on training points (1-6) , and a wide-margin binary classifier (left) using classification SVM that only separates the two classes (+ and –), and does not necessarily preserve a desired ordering on the points [105].

simultaneously addresses both these issues. Structural SVM is an oracle framework that can be adopted for a variety of tasks like object detection, classification with taxonomies, label sequence learning, etc. by appropriately defining its components that suit the problem at hand. It is particularly useful in the scenarios where both input as well as output can be quite complex in general and may carry inherent structure in them.

Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be a collection of n samples represented in $\mathbb{R}^d$ and corresponding outputs. Here, each sample $\mathbf{x}$ and its output y belong to a (complex) space $\mathcal{X}$ and $\mathcal{Y}$ respectively. In Structural SVM, the objective is to learn a discriminant function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ that can be used to predict the optimal output y^* given an input $\mathbf{x}$ by maximizing F over the space $\mathcal{Y}$. That is,

$$y^* = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} \;\; F(\mathbf{x}, y; \mathbf{w}) \tag{2.14}$$

where $\mathbf{w}$ is the parameter vector that needs to be learned. A commonly used assumption is $F = \mathbf{w} \cdot \Psi(\mathbf{x}, y)$; i.e., F is a linear function of the joint feature representation $\Psi(\cdot)$ of input-output pair. In the above setting, our goal is to learn $\mathbf{w}$ such that the maximum number of the following constraints are satisfied:

$$\forall i : \;\; \{\mathbf{w} \cdot \Psi(\mathbf{x}_i, y_i) > \mathbf{w} \cdot \Psi(\mathbf{x}_i, y)\} \;\; \forall y \in \mathcal{Y} \setminus y_i \tag{2.15}$$

The above set of constraints signifies that for every sample $\mathbf{x}_i$, the parameter vector $\mathbf{w}$ should be learned such that the prediction score for the true output (i.e., $F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w})$) remains higher than the prediction

score for any other output. Since this is a hard problem, its solution is approximated by introducing non-negative slack variables. The task of learning $\mathbf{w}$ is then formulated as the following optimization problem:

$$\min_{\mathbf{w},\xi\geq 0} \quad \frac{\lambda}{2}||\mathbf{w}||_2^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i \tag{2.16}$$
$$s.t. \;\; \mathbf{w}\cdot\Psi(\mathbf{x}_i, y_i) \geq \mathbf{w}\cdot\Psi(\mathbf{x}_i, y) + \Delta(y_i, y) - \xi_i \quad \forall i, y \in \mathcal{Y}\setminus\{y_i\}$$

Here, ξ_i denotes slack variable, and $\Delta(\mathbf{y}_i, \mathbf{y})$ denotes the loss function that acts as a margin for penalizing any prediction other than the true output. In the above optimization problem, the joint representation $\Psi(\mathbf{x}, \mathbf{y})$ and the loss function $\Delta(\mathbf{y}_i, \mathbf{y})$ are problem specific functions that need to be defined based on the given task.

Generally speaking, Ranking SVM and Structural SVM allow us to include a larger class of problems under the framework of support vector based methods, other than simple classification. As we can observe from Eq. 2.8, 2.13 and 2.16, the objective functions of SVM, Ranking SVM and Structural SVM look almost the same, and they also share some properties such as good scalability, large margin guarantees and good generalization. In each of these, the support vectors are the samples that violate the margin constraints. The difference comes from the definition of these constraints, which impose correct sample classification in SVM (Eq. 2.7), pairwise sample ranking in Ranking SVM (Eq. 2.11) and joint input-output ranking in Structural SVM (Eq. 2.15).

2.2 Modern Representations

Here, we will discuss two of the recently introduced, quite popular and widely successful representations in the image and text domains: using deep CNN for images [67], and word2vec for (textual) words [88]. These both use artificial neural network as the tool to learn the desired properties from training data. It is worth mentioning that along with novel ideas, both these papers have several technical contributions that were crucial in making these methods/representations perform well on challenging real-world large-scale datasets, and those have also been adopted/adapted by several (thousands of) subsequent papers.[2] However, in this section, rather than presenting their network architectures and implementation/technical details, we will try to present a high-level explanation of these. To motivate

[2]In fact, these have become so standard that now people who work with these can easily be noticed to have almost memorized their network architectures and technical details.

the reader, the three key aspects that capture the gist of both these representations are: representative, discriminative and compositional. These will become clear as we describe and discuss these below.

2.2.1 Image Representation Using Deep CNN

In [67], the authors proposed a neural network architecture that demonstrated state-of-the-art results on the ImageNet large scale visual classification task [9]. In this, given an input image (input layer), convolution operations followed by non-linear activations and pooling are performed in the first five layers, and the last three layers are fully connected layers, with the last (output) layer being a soft-max layer denoting the probability of each of the thousand ImageNet classes. To train the model, an error is computed at the output layer and is back-propagated through the hidden layers, thus updating all the convolution filters.

While neural networks have been used in a variety of tasks for several decades, they used to have very few hidden layers. This paper ([67]) showed for the first time the advantage of incorporating a large number of hidden layers in a network. In addition, this paper had several new contributions: a highly optimized parallel implementation using two GPUs, usage of rectified linear units (ReLU) as activation functions rather than sigmoid or tanh to maintain gradients, pooling of convolution responses after certain layers to incorporate invariance, augmenting the dataset with samples automatically generated by applying simple transformations on original images (data augmentation), and randomly dropping-out some nodes during training with 0.5 probability (drop-out).[3] Since then, many variants of this network have been explored; e.g., a similar network but with even more layers [121], and an even deeper network of over a hundred layers with connections across non-consecutive layers [42].

The motivation behind using multiple layers, each with several convolution filters is to capture salient aspects of various categories in a hierarchical/compositional manner. Intuitively, in the beginning, the filters are small and thus capture some basic colour patterns and edge orientations (analogous to Gabor filters). On moving further, the responses from the filters in the previous layer are pooled and given as input to new filters. Each of the convolution filters in every layer can be thus thought of as a part-specific model. In [21], it was empirically shown that since the model is trained using a large (ImageNet) dataset, it has high representational power and generalization ability. As a result, the outputs of the fully-connected layers towards the end can also be used as features for tasks other than ImageNet classification, such as scene classification, fine-grained recognition and domain adaptation.

[3]The last two choices were found to be particularly useful in controlling overfitting over training data.

It is worth mentioning that there were a few attempts around the same time when [67] was published, with their motivation also being to learn local part-models in either an unsupervised [122] or weakly supervised [59] set-up. While there is a long history of part-based models in vision, these papers demonstrated that informative parts learned in a discriminative manner can provide useful mid-level representations for a variety of tasks. However, there are a fundamental distinctions between the discriminative part-learning techniques and deep CNN, that resulted in a wider success of [67] than the other two. Some of these are:

1. In both [59, 122], the step of learning filters is separated from learning the classification models. Due to this, the filters learned may not be the most appropriate ones for the classification task. Whereas in [67], both the steps are performed using a single network in an end-to-end manner. As a result, the filters are learned iteratively guided by the classification errors back-propagated in the network, and thus are in sync with the end goal (i.e., classification).

2. In both [59, 122], in order to learn filters for parts of different sizes, an image is re-scaled at multiple resolutions and then fixed sized parts are picked. Whereas in [67], filters are learned in a selectively non-linear compositional manner by pooling responses from filters in previous layers. Due to this, filters for larger parts capture only the important aspects from smaller parts, and can be thought of as non-linear functions of those.

3. In both [59, 122], HOG [17] is used for part representation. Whereas in [67], the model is trained using raw images as input. Due to this, the filters are not biased towards any hand-crafted feature representation.

4. In both [59,122], explicit steps are taken to ensure that the learned part filters are distinct. However in [67], this is implicitly taken care of due to random initialization of filters.

2.2.2 Word Representation Using word2vec

The primary motivation behind word2vec [88] is to learn vector representations for words that can also be semantically meaningful. Since the meaning of a word depends on the context in which it is used, the hypothesis of word2vec states that words appearing in similar contexts should have similar representations and vice-versa. In other words, the objective function is a linear function which is designed such that it tries to increase the similarity between a positive word-context pair and decrease it

for negative ones, with similarity being measured using a dot product based (sigmoid) function between the learned vectors for each word-context pair. The positive context for a word is the set of words that appear within a fixed-size window around it, with size of the window deciding the extent of context. The negative context for a word is defined as a set of randomly sampled words that do not appear around that word according to the given dataset. This means that each word can act as positive context for some words and negative for some. As the window-size for (positive) context grows, the model starts considering a wider span of word-context similarities and thus becomes more topical. Since the context of word in a given word-context pair is also a word, it turns out that the contexts that share many words will have similar representations, and the words that share many contexts will also have similar representations, and vice-versa for dissimilar ones. In [32], the authors show how the word2vec model described in [88] can be derived in a step-by-step manner.

As the objective function of word2vec models both positive as well as negative analogies between word-context pairs in a linear manner, an interesting property of the word vectors thus learned is that these vectors also capture such analogies, and these can be derived using simple add/subtract operations over them. E.g., if we want to figure out "A *king* is to a *man* as a *queen* is to what?", then we can simply add the vectors for *king* and *man* and subtract the *queen* vector. It turns out that the resultant vector is very close to the vector for the word *woman*. This is because the words *man* and *woman* are very likely to appear near the words *king* and *queen* respectively, and thus act as context words for them.

Chapter 3

Annotating Images Using Labels

3.1 Introduction

Automatic image annotation is a labelling problem that has potential applications in image classification [148], image retrieval [28, 35, 79, 80], image caption generation [69], etc. Given an (unseen) image, the goal of image annotation is to predict a set of textual labels describing the semantics of that image. Over the last decade, the outburst of multimedia content on the Internet as well as in personal collections has raised the demands for auto-annotation methods, thus making it an active area of research [6, 14, 15, 28, 29, 35, 60, 79, 80, 91, 94, 141, 154, 160].

In the past, several methods have been proposed for image auto-annotation. Our attempts falls under the category of supervised annotation models such as those listed above that work with large annotation vocabularies consisting of few hundreds of labels.

The problem of image annotation seems to have close parallels with multi-label classification and multi-label ranking tasks that are popular in the machine learning community. However, there are certain practical aspects that give rise to a different set of challenges. Some of these include class-imbalance, incomplete labelling, label-ambiguity and structural overlap. While some of these are relevant from the machine learning point of view, others are primarily because of the semantic aspects of the visual and textual modalities (i.e., images and labels). E.g., the two pairs of samples shown Figure 3.1 demonstrate the problem of semantic labelling, which is primarily attributed to the lack of good feature representations (expected to improve with the recent advancements in deep feature learning techniques).

circle, orange, round	circle, music, orange, red, round, white	animal, black, brown, ear, eye, horse, nose	animal, brown, ear, eye, nose

Figure 3.1 Sample images from the ESP Game dataset. Though images in the two pairs are both visually as well as conceptually very different, they share several labels.

3.1.1 Related Work

The goal of an image annotation model is to formulate a mapping between the images and annotation labels. This was initially addressed using translation models such as [24, 92]. These treat it as a problem of machine translation, where an image region needs to be translated into a semantic label. The authors of [24] released pre-computed region-based features for the Corel-5K dataset which were widely used by the subsequent papers. The image annotation domain has primarily been dominated by nearest neighbour based approaches, that predict labels of a test image by computing its similarity with a (sub)set of training images. In the relevance models such as CMRM [53], CRM [70] and MBRM [28], image annotation was modelled as a problem of computing the joint probability of image regions and labels. In MBRM [28], it was shown that using regular blocks rather than arbitrary shaped regions as in CRM/CMRM, and modelling the absolute presence/absence of labels using a Bernoulli distribution rather than modelling their frequency using a multinomial distribution could provide better performance. In [154], a Markov Random Field based approach was proposed that could flexibly accommodate some of the previous generative models. The Joint Equal Contribution (JEC) approach proposed in [79,80] treats the problem of image annotation as that of image retrieval. It demonstrated that a simple nearest-neighbour based greedy algorithm could outperform earlier, relatively complex models, though by using multiple high-dimensional global features rather than simple region-based features. Although JEC is conceptually simple, it achieved the best results on benchmark annotation datasets when it was proposed. Inspired from the success of [28, 53, 70, 79], a weighted kNN based method called TagProp was proposed in [35]. This transfers labels to a test image by taking a weighted average of keywords'

presence among the neighbouring (training) images. To address the class-imbalance problem, logistic discriminant (sigmoid) models are wrapped over the weighted kNN method. This boosts the importance given to infrequent/rare labels, and suppresses the importance given to frequent labels. They also proposed a metric learning approach for learning weights that combines multiple distances computed using different features. As part of this work, the authors released a set of pre-computed global features for the standard annotation datasets. Since then, these features have been used by almost all the annotation approaches. [73] proposed a measure to compute the relevance of a tag to an image by taking into account its frequency in the neighbouring samples of that image, and the entire (training) collection. Another nearest-neighbour based method [160] tries to benefit from feature sparsity and clustering properties using a regularization based algorithm for feature selection. In addition to these, a few discriminative models [14, 29] treat each label as a class of a multi-class multi-labelling problem, and learn separate class-specific models.

Among the recent methods, in [91], a greedy approach was proposed to identify the best kernel for each feature while computing image similarity using a set of features. This in turn results in a sparse subset of features that maximize annotation performance. In [60], a formulation based on weighted multi-view non-negative matrix factorization was proposed, with the goal of learning a generative model specific to each query/test image using its neighbouring samples. In [94], a discrete variant of the MBRM model [28] was combined with binary one-vs-rest SVM models to formulate a hybrid approach for annotating images. This combination particularly helped in increasing the number of labels that were correctly recalled. In [6], the authors showed that learning (kernelized) cross-modal feature embedding can significantly improve the performance of nearest neighbour based methods such as those of [35, 73, 79, 80, 140].

In parallel to the above advances, there have been papers such as [55, 149] that target the problem of multi-label classification. However, usually such methods are shown to work on small vocabularies containing a few tens of labels. Our work falls under the category of supervised image annotation methods [14, 15, 28, 29, 35, 79, 80, 91, 94, 98, 154, 160] that address a more realistic and challenging scenario where the vocabulary contains a few hundred labels and the datasets seriously suffer from class-imbalance and incomplete-labelling.

3.1.2 Our Motivation

As discussed above, while several complex methods have been proposed for image annotation, the relevance models [28, 53, 70] showed that nearest neighbour based approaches could perform surprisingly well on this task. These used simple segment/region-level features for image representation. Later, JEC [79] showed that using multiple global features for image representation along with a greedy nearest neighbour based approach could provide further boost in performance. However, it could not come-up with an effective approach for combining distances computed using multiple features (distance metric learning) especially for multi-label annotation data. Soon after JEC, TagProp [35] again validated the capability of a simple weighted nearest neighbour based criteria on the image annotation task. More importantly, the authors also proposed a systematic metric learning framework that could help in combining distances computed using multiple features.

While a number of papers demonstrated the effectiveness of the nearest neighbour rule in the annotation task, they did not completely address a few aspects particular to this task. Specifically, their neighbour selection criteria was confined to visual features and did not consider label information. This motivated us to formulate a new nearest neighbour based approach that could explicitly make use of label information in the annotation task [140, 144].

Next, during our study of existing methods, we also observed that SVM, which is quite popularly used in several machine learning problems such as classification, was almost unexplored in the annotation task till then. This was probably because of the fundamental distinctions between the conventional single-label classification task and multi-label annotation problem. To examine its performance, we evaluated the performance of the standard one-versus-rest SVM on standard annotation datasets. Surprisingly, it performed quite well (comparable to or better than JEC) without even considering the "multi-label" aspects of the data. To make it compatible to the annotation task, and make it more responsive towards image annotation related aspects such as incomplete-labelling, label ambiguity and structural overlap, we also introduced a new parameter in the conventional SVM hinge-loss that could make the learned model tolerant against them [141].

3.1.3 Organization

This chapter is organized as follows. In the next two sections (Section 3.2 and Section 3.3), we present our two attempts [140, 141, 144] towards addressing the image annotation problem. In Sec-

tion 3.4, we describe and analyze popular image annotation datasets that we have used in our experiments. In Section 3.5, we discuss various features used for image representation. Section 3.6 gives details of our experimental set-up, and Section 3.7 presents empirical analysis and discussion. Finally, we provide a summary in Section 3.9.

3.2 2PKNN: A Nearest Neighbour Based Approach

As discussed in Section 3.1.1, the nearest neighbour based methods such as [28, 35, 79, 80] have been found to give some of the best results on the image annotation task despite their simplicity. The intuition is that "similar images share similar labels" [79, 80]. In most of the existing approaches, this similarity is determined using only visual features. In the nearest neighbour based scenario, since the labels co-occurring in an image are considered together, visual similarity can also handle correlations among labels to some extent. However, it fails to address the two important issues of "class-imbalance" (large variations in the frequency of different labels) and "incomplete-labelling" (many images are not annotated with all the relevant labels from the vocabulary) that are prevalent in the popular annotation datasets as well as real-world databases. To address these issues in the nearest neighbour based set-up, one needs to ensure that (a) for a given image, the (subset of) training images that are considered for label prediction/propagation should not have large variations in the frequency of different labels, and (b) the comparison criteria between two images should make use of both image-to-label and image-to-image similarities (as discussed above, image-to-image similarities can partially capture label-to-label similarities in the nearest neighbour based scenario). With this motivation, we present a two-step variant of the classical k-nearest neighbour (kNN) algorithm that fulfills both these requirements. We call this *2-Pass k-Nearest Neighbour* (2PKNN) algorithm. As part of the 2PKNN algorithm, for an image, we say that its few nearest neighbours from a given class constitute its *semantic neighbourhood* with respect to that class, and these neighbours are its *semantic neighbours*. Based on the above discussed intuition of [79, 80] that similar images share similar labels, we hypothesize that the semantic neighbours of an image from a particular class are the samples that are visually and hence semantically most related with that image with respect to that class. Now, given a new image, in the first step of 2PKNN we identify its semantic neighbours corresponding to all the labels [1]. Then in the second step, only these samples are used for label prediction. In comparison to the conventional kNN algorithm, note that we additionally

[1] We shall use the terms class/label interchangeably.

introduce an initial pruning step where we pick visually similar neighbours that cover all the labels. This also relates with the idea of "bottom-up pruning" common in day-to-day scenarios such as buying a car, or selecting clothes to wear, where first the potential candidates are short-listed based on a preliminary analysis, and then another set of criteria are used for final selection.

It is well-known that the performance of kNN based methods largely depends on how two images are compared [35, 79, 80]. Usually, this comparison is done using a set of features extracted from images and a specific distance metric for each feature (such as L_1 distance for colour histograms, or L_2 for GIST descriptor). As the 2PKNN algorithm works in the nearest neighbour setting, we would like to learn a distance metric that maximizes the annotation performance. With this goal, we perform metric learning over 2PKNN by extending the popular Large Margin Nearest Neighbour (LMNN) metric learning algorithm proposed in [151,152] for multi-label prediction. Since it requires us to perform pair-wise comparisons iteratively, scalability becomes one of the important concerns while working with thousands of images. To address this, we implement metric learning by alternating between stochastic sub-gradient descent and projection steps on subsets of training pairs, that has a motivation similar to the Pegasos algorithm [118]. This allows to optimize the weights iteratively using a small number of comparisons at each iteration, thus making our metric learning formulation scalable.

3.2.1 Label Prediction Model: 2PKNN

Now we present our 2PKNN method for image annotation. Let $\{I_1, \ldots, I_t\}$ be a collection of images and $\mathcal{Y} = \{y_1, \ldots, y_l\}$ be a vocabulary of l labels (or semantic concepts). The training set $\mathcal{T} = \{(I_1, Y_1), \ldots, (I_t, Y_t)\}$ consists of pairs of images and their corresponding label sets, with each $Y_i \subseteq \mathcal{Y}$. Analogous to the relevance models [28, 53, 70] and the Supervised Multiclass Labeling (or SML) method [14], we assume the conditional probabilities $P(J|y_i)$ that models the feature distribution of an (unseen) image J given a semantic concept $y_i \in \mathcal{Y}$. Using this, we model image annotation as a problem of finding the posterior probability for each label:

$$P(y_i|J) = \frac{P(J|y_i)P(y_i)}{P(J)} \tag{3.1}$$

where $P(y_i)$ is the prior probability of the label y_i. Then, given an unannotated image J, the best label for it will be given by

$$y^* = \arg\max_i P(y_i|J) \tag{3.2}$$

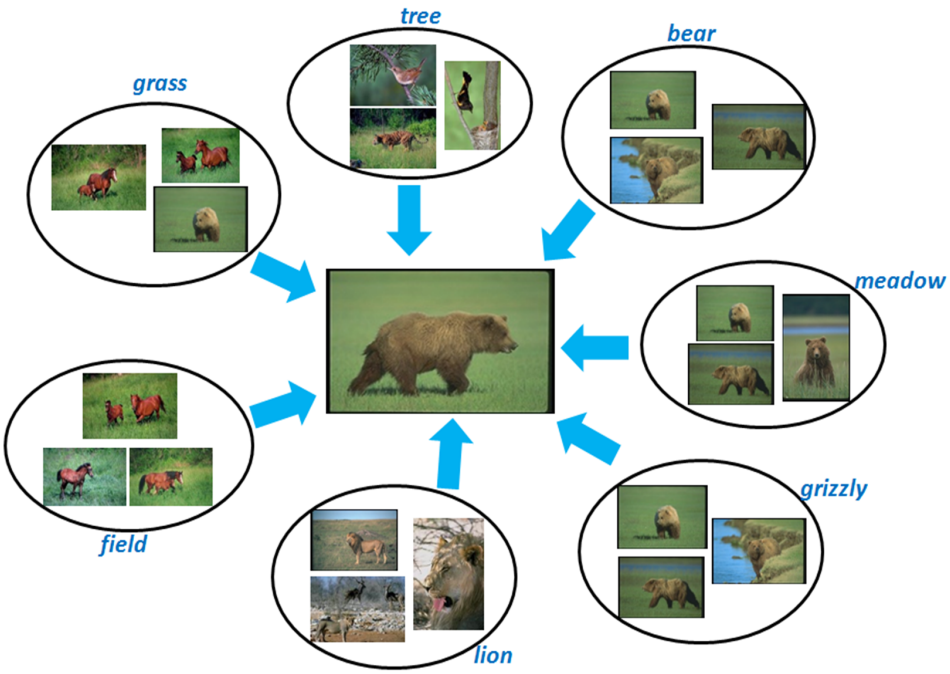


Figure 3.2 In the first pass of 2PKNN, for a given image to be annotated (center), we identify its semantic neighbours corresponding to each semantic group, and only these samples are considered during label prediction. Since each image can have multiple labels, an image can come from more than one semantic group. E.g., the semantic groups corresponding to the labels "bear", "meadow" and "grizzly" have two images in common. It is worth noticing that unlike the usual kNN based image annotation methods that make use of only feature-based similarity while determining the neighbours of a given image and ignore the label information, here we explicitly make use of both.

Let $\mathcal{T}_i \subseteq \mathcal{T}$, $\forall i \in \{1, \ldots, l\}$ be the subset of training data that contains *all* the images annotated with the label y_i. Since each set $\mathcal{T}_i$ contains images with one semantic concept common among them, we call it a *semantic group*. It should be noted that the sets $\mathcal{T}_i$ are not disjoint, as an image usually has multiple labels and hence belongs to multiple semantic groups. Given an unannotated image J, from each semantic group we pick K_1 images that are most similar to J and form corresponding sets $\mathcal{T}_{J,i} \subseteq \mathcal{T}_i$. Thus, each $\mathcal{T}_{J,i}$ contains those images that are *most informative* in predicting the probability of the label y_i for J (as discussed in Section 3.1, our approach is motivated by the observation that "similar images share similar labels" [79, 80]). The samples in each set $\mathcal{T}_{J,i}$ are the semantic neighbours of J corresponding to y_i, and help in incorporating image-to-label similarity. Once $\mathcal{T}_{J,i}$s are determined, we

merge them all to form a set $\mathcal{T}_J = \{\mathcal{T}_{J,1} \bigcup \ldots \bigcup \mathcal{T}_{J,l}\}$. This way, we obtain a subset of the training data $\mathcal{T}_J \subseteq \mathcal{T}$ specific to J that contains its semantic neighbours corresponding to all the labels in the vocabulary $\mathcal{Y}$. This is the *first pass* of 2PKNN, as illustrated in Figure 3.2.

In $\mathcal{T}_J$, each label would appear at least K_1 times, which in turn tries to address the class-imbalance issue. To understand how this step also tries to handle incomplete-labelling, we analyze the cause of this. Incomplete-labelling occurs because some (either too obvious, or highly context-specific) labels are often missed by human annotators while manually annotating a dataset, and hence many images depicting such concepts are actually not annotated with them. Under this situation, given an unseen image, if we use only its *few* nearest neighbours from the *entire* training data (as in [35, 79, 80]), then such labels may not appear among these neighbours and hence would not get appropriate scores. In contrast, the first pass of 2PKNN builds a neighbourhood where all the labels are present explicitly. Therefore, now even those labels that did not appear among the neighbours determined using the usual kNN have better prediction chances.

The *second pass* of 2PKNN is a weighted sum over the samples in $\mathcal{T}_J$ to assign importance to labels based on image similarity. This gives the posterior probability for J given a label $y_k \in \mathcal{Y}$ as

$$P(J|y_k) \quad \propto \sum_{(I_i,Y_i)\in\mathcal{T}_J} \theta_{J,I_i}.P(y_k|I_i) \tag{3.3}$$

where, $\theta_{J,I_i} = exp(-\pi D(J, I_i))$ denotes the contribution of image I_i in predicting the label y_k for J depending on their visual similarity, with π being a scalar that controls the decay of θ_{J,I_i}; $D(J, I_i)$ denotes the distance between J and I_i in feature space (see Eq. 3.7 for the definition of $D(J, I_i)$); and $P(y_k|I_i) = \delta(y_k \in Y_i)$ denotes the presence/absence of label y_k in the label set Y_i of I_i, with $\delta(\cdot)$ being 1 when the argument holds true and 0 otherwise. Assuming that the first pass of 2PKNN gives a subset of the training data where each label has comparable frequency, we set the prior probability in Eq. 3.1 as a uniform distribution; i.e., $P(y_i) \propto \frac{1}{|\mathcal{T}|}, \forall i \in \{1, \ldots, l\}$. Putting Eq. 3.3 in Eq. 3.1 provides a ranking of all the labels based on their probability of getting assigned to the unseen image J. Finally, the probability score $P(y_i|J)$ is regularized using the following normalization (similar to [91]):

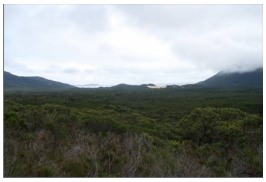
$$P(y_i|J) = \frac{P(y_i|J)}{\max_{J'} P(y_i|J')} \tag{3.4}$$

Note that along with image-to-image similarities, the second pass of 2PKNN implicitly takes care of label-to-label dependencies since the labels appearing together in the same neighbouring image will get equal importance (analogous to [28, 35, 53, 70, 79]). To our knowledge, ours is the first published work

bay, gravel, road, house, landscape, meadow, shrub, sky

bush, coast, grey, sea, **sky**

house, mountain, hill, **sky**, village, tree



bush, **landscape**, cloud, sea, tree

meadow, **house**, sea, bush, dirt, **sky**, **bay**, **road**, coast

bush, coast, grey, sea, **sky**

hill, **sky**, **house**, mountain, village, tree

bush, coast, grey, sea, **sky**

hill, tree, **house**, village, mountain, **sky**

Figure 3.3 For an example test image from the IAPR TC-12 dataset (top row), the middle row shows its 4 nearest images (and their ground-truth labels) from the training data determined after the first pass of 2PKNN, and the bottom row shows its 4 nearest images determined using JEC [79, 80]. The labels in bold are the ones that match with the ground-truth labels of the test image. Note the frequency (9 versus 6) and diversity ({sky, house, landscape, bay, road, meadow} versus {sky, house}) of matching labels for 2PKNN versus JEC respectively.

that proposed to explicitly integrate label information while determining the neighbours of an image in the image annotation task. Here we extend this work in the following ways:

Figure 3.3 shows an example from the IAPR TC-12 dataset illustrating how the first pass of 2PKNN tries to address both class-imbalance and incomplete-labelling. For a given test image (top row) along with its ground-truth labels, we can notice the presence of rare labels {"landscape", "bay", "road","meadow"} among its four nearest images found after the first pass of 2PKNN (middle row), without compromising frequent labels {"sky", "house"}. In contrast, the neighbours obtained using JEC [79, 80] (bottom row) contain only frequent labels. We can also observe that though the labels {"landscape", "meadow"} look obvious for the neighbours found using JEC, these are actually absent in their ground-truth annotations (incomplete-labelling), whereas after the first pass of 2PKNN we can ensure their presence among the neighbours selected for label prediction. To discuss these aspects more formally, below we provide an analysis on the diversity and completeness of labels included in the neighbours identified using 2PKNN and classical kNN. We will provide an empirical analysis on this in Section 3.7.4.3.

3.2.1.1 Analysing Diversity and Completeness

Here we try to analyse two aspects related to the presence of labels in the neighbours selected for label prediction: *diversity* and *completeness*. In the present context, we define "diversity" as the number of distinct labels that are present in the selected neighbours, and "completeness" as the state when all the labels are present in the selected neighbours. In order to achieve completeness, we will require perfect diversity, i.e., presence of all the labels in the neighbours. Along with 2PKNN, we will analyse and compare these aspects with respect to the conventional kNN algorithm.

Recall that in the first pass of 2PKNN, we identify the K_1 nearest neighbours of a given (test) sample J from each semantic group and take their union, thus obtaining a subset of training samples $\mathcal{T}_J \subseteq \mathcal{T}$ specific to J. Let K_2 denote the number of nearest neighbours of J from $\mathcal{T}_J$ that are considered for label prediction. Also, with respect to the kNN algorithm, we pick the K nearest neighbours of J from the complete training set for label prediction. We also assume we have a sufficiently large number of samples for each label.

For simplicity, let us initially consider a vocabulary $\mathcal{Y} = \{y_1, y_2, y_3\}$ of three labels (i.e., a vocabulary of size $a = 3$), and assume that each sample in the (training) data is associated with exactly $b = 2$ distinct labels. For 2PKNN, if we have $K_1 = 1$, then we will get two samples in the set $\mathcal{T}_J$ (since one

sample will be selected twice for two labels). This means that in $\mathcal{T}_J$, the frequency of two labels will be one, and that of the remaining one label will be two. In this case, if we consider $K_2 = 1$, then the diversity of labels in the selected neighbours will be 2, and if we consider $K_2 = 2$, then we will achieve perfect diversity and thus completeness of labels in the selected neighbours. Now, let us consider the case of kNN. If we consider $K = 1$, then the diversity of labels will be 2 similar to 2PKNN, and in the best scenario we will require $K = 2$ samples to achieve perfect diversity. However, since we do not take into account the label information of samples in kNN while selecting the neighbours and make use of only sample features, we may end-up getting neighbouring samples labelled with the same two labels even for very large values of K, and thus we can not guarantee the minimum value of K to achieve perfect diversity.

In general, for any (positive integral) values of a and K_1, if we assume each sample x_u to be labelled with $b_u \leq a$ distinct labels (b_u may be different for different samples in multi-label scenario), then in order to achieve perfect diversity using 2PKNN, we will require K_2 to be $\lfloor \frac{a-1}{\max(b_u)} \rfloor + 1$ in the best scenario (lower bound), and $(a - \min(b_u)) \times K_1 + 1$ in the worst scenario (upper bound). These will depend on the overlap of labels in the selected neighbours. In the case of kNN algorithm, we will require K to be $\lceil \frac{a-1}{\max(b_u)} \rceil + 1$ to achieve perfect diversity in the best scenario, which is the same as that for 2PKNN. However, here we cannot bound the value of K to achieve perfect diversity in the worst scenario, following the same reasoning as above.

Note that in practice, since we consider all the samples in $\mathcal{T}_J$ during label prediction (i.e., $K_2 = |\mathcal{T}_J|$ in Eq. 3.3), we can assure perfect diversity and completeness of labels using 2PKNN.

3.2.2 Metric Learning

Most of the existing classification based metric learning algorithms try to increase inter-class and reduce intra-class distances, thus treating each pair of samples in a binary manner. Since image annotation is a multi-label prediction task, here the similarity between two samples need not be binary, and hence classification based metric learning cannot be applied directly. As part of metric learning, our aim is to learn a distance metric that maximizes the annotation performance for 2PKNN. For this purpose, we extend the LMNN algorithm [152] for multi-label prediction. Below, first we provide an overview of the LMNN algorithm (*c.f.* Section 2.1.1.1 for a detailed discussion on LMNN) , and then present our formulation.

3.2.2.1 Revisiting the LMNN Algorithm

The goal of LMNN is to learn a Mahalanobis metric such that the performance of kNN classification is improved. Let us assume a training set $\{\mathbf{x}_i, y_i\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ denotes a sample and $y_i \in \{1, \ldots, C\}$ denotes its category/class from total C classes. Given two samples $\mathbf{x}_i$ and $\mathbf{x}_j$, the Mahalanobis distance between them is given by:

$$d_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = ((\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j))^{\frac{1}{2}}$$

where $\mathbf{M}$ is a symmetric positive semidefinite matrix ($\mathbf{M} \succeq 0$). In LMNN, $\mathbf{M}$ is learned such that the local neighbourhood of a sample belongs to the same category. To do this, for a given sample, its neighbours from the same class are pulled closer and those from different classes are pushed farther.

For a given sample $\mathbf{x}_i$, its *target* neighbours are defined as its k nearest samples from the same class, and *impostors* as its neighbours from other classes that are closer than the target neighbours. Using this information, a Mahalanobis metric is learned such that each sample is closer to its target neighbours than impostors by a margin. Let $\mathbf{x}_j$ and $\mathbf{x}_k$ be a target neighbour and an impostor respectively for $\mathbf{x}_i$, then this constraint can be expressed as:

$$d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k) - d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) \geq 1 \tag{3.5}$$

Note that the above constraint is enforced only on local neighbours. Based on the above constraints, the objective function of LMNN is given by:

$$\begin{aligned} \min_{\mathbf{M}} \sum_{ij} \eta_{ij} d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) + \mu \sum_{ijk} \eta_{ij}(1 - \lambda_{ik}) \xi_{ijk} \\ s.t. : d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_k) - d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) \geq 1 - \xi_{ijk} \\ \xi_{ijk} \geq 0 \\ \mathbf{M} \succeq 0 \end{aligned} \tag{3.6}$$

Here, the first expression tries to pull target neighbours closer, and the second expression penalizes violations of the constraints in Eq. 3.5. The variable η_{ij} is 1 if $\mathbf{x}_j$ is a target neighbour of $\mathbf{x}_i$ and 0 otherwise; λ_{ik} is 1 if $\mathbf{x}_i$ and $\mathbf{x}_k$ belong to the same class and 0 otherwise; $\mu > 0$ manages tradeoff between the two terms; and ξ_{ijk} denotes slack variable.

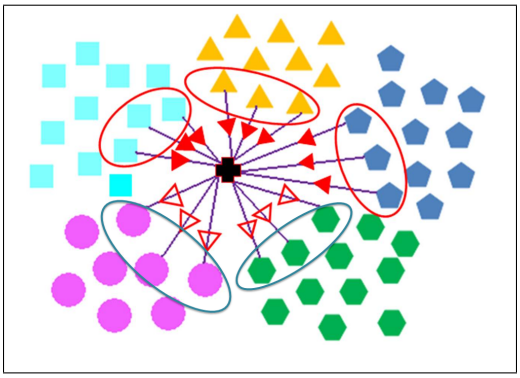


Figure 3.4 Illustration of distance metric learning for 2PKNN. Let there be 5 labels $\mathcal{Y} = \{A, B, C, D, E\}$ denoted as A : squares, B : triangles, C : pentagons, D : hexagons, and E : circles. Each set $\mathcal{T}_\alpha$, $\alpha \in \mathcal{Y}$ consists of samples that have one label as α. For a given sample (denoted by cross in the center), let its actual labels be $\{A, B, C\}$. During distance metric learning with $K = 3$, its three nearest neighbours from $\mathcal{T}_A$, $\mathcal{T}_B$ and $\mathcal{T}_C$ act as target neighbours that need to be pulled closer to it, while those from the remaining ones act as impostors that need to be pushed far from it.

3.2.2.2 Metric Learning for 2PKNN

Let there be two images A and B, each represented by n features $\{\mathbf{f_A^1}, \ldots, \mathbf{f_A^n}\}$ and $\{\mathbf{f_B^1}, \ldots, \mathbf{f_B^n}\}$ respectively. The distance between two images is computed by finding the distance between their corresponding features using some specialized distance measure for each feature (such as L_1 for colour histograms, χ^2 for bag-of-words histograms, etc.), and then combining them all. Let d^i_{AB} denote the distance between A and B computed using the ith feature. In order to optimally combine multiple feature distances, we use a linear distance metric $\mathbf{w} \in \mathcal{R}^n_+$ in the distance space. Based on this, we write the distance between A and B as:

$$D(A, B) = \sum_{i=1}^{n} \mathbf{w}(i)\, d^i_{AB} \tag{3.7}$$

Now we describe how to learn the metric $\mathbf{w}$ for the multi-label image annotation task. For a given labelled sample $(I_p, Y_p) \in \mathcal{T}$, we define its (i) *target neighbours* as its K_1 nearest images from the semantic group $\mathcal{T}_q$, $\forall q$ such that $y_q \in Y_p$, and (ii) *impostors* as its K_1 nearest images from $\mathcal{T}_r$, $\forall r$ such that $y_r \in \mathcal{Y} \setminus Y_p$. Our objective is to learn the metric such that the distance of a sample from its target neighbours is minimized, and is also less than its distance from any of the impostors (i.e., *pull* the target neighbours and *push* the impostors). In other words, given an image I_p along with its

labels Y_p, we want to learn the weights such that its nearest (K_1) semantic neighbours from the semantic groups $\mathcal{T}_q$'s (i.e., the groups corresponding to its ground-truth labels) are pulled closer, and those from the remaining semantic groups are pushed farther (Figure 3.4). With this goal, for a sample image I_p, its target neighbour I_q and its impostor I_r, the loss function will be given by

$$E_1 = \sum_{pq} \eta_{pq} D(I_p, I_q) + \ \mu \sum_{pqr} \eta_{pq}(1 - \lambda_{pr})[1 + D(I_p, I_q) - D(I_p, I_r)]_+ \tag{3.8}$$

where $\mu > 0$ handles the trade-off between the two error terms. The variable η_{pq} is 1 if I_q is a target neighbour of I_p and 0 otherwise. $\lambda_{pr} = \frac{|Y_p \cap Y_r|}{|Y_r|} \in [0, 1]$, with Y_r being the label set of an impostor I_r of I_p. And $[z]_+ = max(0, z)$ is the hinge loss which will be positive only when $D(I_p, I_r) < D(I_p, I_q) + 1$ (i.e., when for a sample I_p, its impostor I_r is nearer than its target neighbour I_q). To make sure that a target neighbour I_q is much closer than an impostor I_r, a margin (of size 1) is used in the error function.

The above loss function is minimized by the following constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \textstyle\sum_{pq} \eta_{pq} D(I_p, I_q) + \mu \sum_{pqr} \eta_{pq}(1 - \lambda_{pr}) \xi_{pqr} \\ s.t. \quad & D(I_p, I_r) - D(I_p, I_q) \geq 1 - \xi_{pqr} \ \ \forall p, q, r \\ & \xi_{pqr} \ \geq 0 \ \ \forall p, q, r \\ & \mathbf{w}(i) \ \geq 0 \ \ \forall i; \ \textstyle\sum_{i=1}^{n} \mathbf{w}(i) = n \end{aligned} \tag{3.9}$$

Here, the slack variables ξ_{pqr} represent the hinge loss in Eq. 3.8. By applying L_1 regularization on $\mathbf{w}$, we try to impose sparsity on the learned weights.

For large datasets on the order of tens of thousands of samples, the above optimization problem can have several millions of constraints. This makes the scalability difficult using conventional gradient descent. To overcome this, we implement it by alternatively using stochastic sub-gradient descent and projection steps (similar to Pegasos [118]) on subsets of training data. This gives an approximate solution using a small number of comparisons, thus making our approach scalable to large datasets containing thousands of samples[2].

3.2.2.3 Comparison with LMNN

Similar to LMNN, the proposed metric learning formulation works on pairs and triplets of samples that are defined in terms of target neighbours and impostors. Recall that while LMNN is meant for

[2] An implementation of 2PKNN and metric learning is available at `http://researchweb.iiit.ac.in/~yashaswi.verma/eccv12/2pknn.zip`

single-label data, ours is of multi-label data. Below we discuss the differences between the two, that are primarily because of the differences in the tasks that each addresses:

- The primary difference between the two formulations lies in terms of the definition of target neighbours and impostors. In case of multi-label data, samples from multiple classes behave as target neighbours and impostors. Moreover, our definitions of such samples especially suit the 2PKNN algorithm, since these are defined based on the semantic neighbours of a sample.

- In LMNN, the variable λ is binary, whereas we define it to be in a continuous range $[0, 1]$, thus scaling the hinge loss depending on the overlap between the label sets of a given image I_p and its impostor I_r. This means that for a given sample, the amount of push applied on its impostor varies depending on conceptual similarity with that sample. An impostor with large similarity will be pushed less, whereas one with small similarity will be pushed more. This makes our formulation suitable for multi-label tasks such as image annotation.

- While working with high-dimensional features, it becomes practically infeasible to learn a square Mahalanobis distance metric ($\mathbf{M}$) as done in LMNN. To overcome this, we learn a linear metric $\mathbf{w}$ in distance space rather than feature space. Since a sample is usually represented using a few tens of features, this makes the dimensionality of $\mathbf{w}$ practically feasible to deal with.

3.3 An SVM Based Approach to Image Annotation

As discussed above, among the image annotation models being proposed in the past, generative or nearest-neighbour (NN)-based models [28, 35, 79, 80, 140, 160] have particularly been shown to be successful for large vocabulary datasets. The reason behind this is that in NN-based models, given a sample, the labels that are not present in the ground-truth of its neighbouring samples are simply ignored, rather than being considered as negative. In contrary, simple one-vs-rest Support Vector Machine (or SVM) [16, 136] has remained almost unexplored in this domain. This might be due to its strict discriminative nature: it considers every sample other than positive as equally negative. E.g., while learning a model for "flowers", samples labeled with "blooms" are considered as negative examples. Due to this, for a given label, the samples that are either incompletely labeled, or tagged with another label that is semantically/structurally similar to the given label get confused as negative examples. This

cars, tracks, prototype

grass, ***flowers***, petals

sky, grass, plane, ***lion***

tree, grass, ***tiger***, park

Figure 3.5 Example images from the Corel-5K dataset [24] and corresponding ground-truth labels. First image is an example of **incomplete-labelling** (tagged with *"car"* but not with *"vehicle"*); second image is an example of **label-ambiguity** (tagged with *"flowers"*, though *"blooms"* would also have been equally correct); & third and fourth images are examples of **structural-overlap** (*"lion"* and *"tiger"* are two different but structurally related labels). These three issues make learning good one-vs-rest discriminative classifiers non-trivial.

in turn inhibits learning good decision boundaries, and hence affects the performance of the learned SVM model.

Specifically, in annotation datasets with large vocabularies of a few hundred or more labels, there exist three practical issues:

1. **Incomplete-labelling:** The training samples are not exhaustively tagged with all relevant labels from the vocabulary. This is because while building a dataset, human annotators find some labels as "obvious" and miss them while preparing the ground-truth. E.g., an image tagged with "car" might not be tagged with "vehicle", or one annotator might label a small object in an image while the other may not.

2. **Label-ambiguity:** There are some labels that convey the same semantic meaning and thus can be used interchangeably, due to which usually only one of them is assigned by an annotator. E.g., an image tagged with "flowers" may not be tagged with "blooms" as both convey the same meaning.

3. **Structural-overlap:** There are some labels that, in spite of being different, share structural properties. This results in feature representations that are quite similar. E.g., though "tiger" and "lion" are two different labels, structurally they are very similar.

All these issues combined give rise to the existence of *confusing* labels within a vocabulary. Figure 3.5 shows such examples from the Corel-5K dataset [24]. It is important to note that some con-

fusing labels might actually be one of the positive labels for a given image, but remain missing in the ground-truth due to these issues. In other words, for a given label l_a, a confusing label l_b is a label that is/could-be used in-place-of/together-with l_a, due to: incompleteness, ambiguity or overlap problems. Below we describe our attempt towards learning from such data [141] where for a given label, there could be several other labels in the vocabulary that act as its confusing labels.

3.3.1 The SVM-VT Model

Let $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ be a collection of m samples and $V = \{l_1, \ldots, l_n\}$ be a vocabulary of n labels. The dataset $T = \{(\mathbf{x}_1, L_1), \ldots, (\mathbf{x}_m, L_m)\}$ is a set of tuples of the form $(\mathbf{x}_i, L_i)$ where $\mathbf{x}_i$ is a sample and $L_i \subseteq V$ is the set of its labels. Let S_i^+ be the set of samples that are annotated with the label l_i . We consider these samples as positive examples of l_i, and denote the remaining samples as $\bar{S_i^+} = S \setminus S_i^+, \forall i \in \{1, \ldots, n\}$. From now onwards, we will discuss considering a single label and omit the subscript index for brevity.

For a given label l, the conventional SVM considers the samples in S^+ as its positive examples and those in $\bar{S^+}$ as its negative examples. Using these two sets, a linear classifier $\mathbf{w}$ is learned (separately for each label) by solving the following optimization problem:

$$\min_{\mathbf{w}} \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{m}\sum_{j=1}^{m}[1 - y_j(\mathbf{w} \cdot \mathbf{x}_j)]_+, \tag{3.10}$$

where $[z]_+ = max(0, z)$ denotes the hinge-loss, $\lambda > 0$ is used to control the trade-off between regularization and loss, and $y_j = 1$ if $\mathbf{x}_j \in S^+$ and -1 otherwise. Solving this leads to finding a hyper-plane $\mathbf{w}$ that best separates the samples in S^+ and $\bar{S^+}$ with maximum margin.

In order to make SVM tolerant against the confusing samples, we define a new loss function based on the hinge-loss. It introduces a tolerance-parameter "t" that adjusts both the margin as well the gradient update-rule for each sample separately. Specifically, we formalize the SVM-VT model as that of solving the following optimization problem:

$$\min_{\mathbf{w}} \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{m}\sum_{j=1}^{m}[1 - y_j t_j(\mathbf{w} \cdot \mathbf{x}_j)]_+, \tag{3.11}$$

where the additional parameter $t_j \in [0, 1]$ controls the tolerance against the errors made in the classification of sample $\mathbf{x}_j$. The hyperplane $\mathbf{w}$ is now learned such that it is more strict towards correctly classifying samples with high values of t_j, and any such error leads to a larger shift in the separating

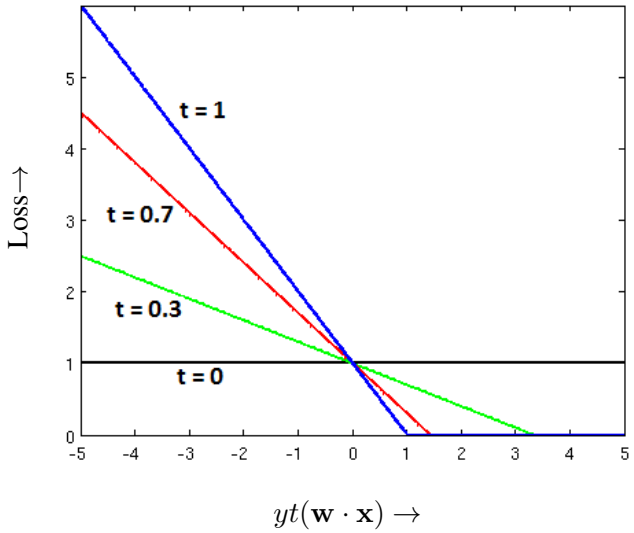


Figure 3.6 Loss function with variation in the tolerance-parameter. The horizontal axis represents the value of $y(\mathbf{w} \cdot \mathbf{x})$ and the vertical axis represents the loss. The conventional hinge-loss corresponds to $t = 1$, and the proposed t_hinge varies with different values of $0 \leq t \leq 1$.

hyperplane. In other words, the hyperplane is more tolerant against errors made in the classification of samples with low values of t_j and such errors lead to a smaller shift in the hyperplane. If $t_j = 1 \forall j$, it becomes exactly the same as that of the standard SVM as shown in Eq. 3.10. In this way, SVM-VT can be viewed as a (strict) generalization of SVM. In Figure 3.6, we show how the hinge-loss function varies with different values of t for some sample $\mathbf{x}$. On the horizontal and vertical axis, we represent the value of $y(\mathbf{w} \cdot \mathbf{x})$ and that of the hinge-loss respectively. It can be seen that for small value of t, the hinge-loss remains small even for large misclassification errors. As we increase t, the hinge-loss becomes more and more sensitive towards misclassification errors and hence these errors get more penalized. Another interesting thing to notice is that as we reduce t, the hinge-loss fires even for the samples which are correctly classified with high confidence, though the loss value remains very small. This is desirable when we are confused about the exact label of a sample and want to penalize its highly confident correct classification. Also, if we set $t = 0$ for some particular sample, then the classifier becomes infinitely tolerant against the error made in its classification.

3.3.2 Determining the Tolerance Parameter

As discussed before, for a given label l, there could exist several other labels in the vocabulary that act as its confusing labels. Due to this, there could be possibly many samples in the set $\bar{S^+}$ that are tagged with some confusing label of l (which we call confusing samples). Though in practice it is possible to automatically learn the tolerance parameter in Eq. 3.11 using either non-convex optimization or convex-relaxation, it would not solve our purpose of identifying such samples. This is because doing so will look only at the features of the samples without considering their semantic properties. Here we propose a heuristic approach for determining the t-value for each sample given a label, that tries to address the three issues (incomplete-labelling, label-ambiguity and structural-overlap) discussed above.

For a given label l, we consider three factors to determine the semantic relatedness of each sample $\mathbf{x}_j \in \bar{S^+}$ with that label:

1. **Reverse nearest-neighbours based score:** For a fixed value of K $(= 5)$, let p_k be the number of samples in S^+ that have $\mathbf{x}_j$ as their kth nearest neighbour. Then we define

$$score_1(\mathbf{x}_j|l) = \frac{\sum_{k=1}^{K}(\frac{p_k}{k})}{\sum_{k=1}^{K} p_k + \epsilon} \tag{3.12}$$

   where $\epsilon > 0$ is a small number to avoid division by zero.

2. **Visual similarity based score:** We compute the visual similarity score $sim(\cdot)$ (scaled to the range $[0, 1]$) of $\mathbf{x}_j$ with its nearest neighbour $\mathbf{x}^* \in S^+$ and define

$$score_2(\mathbf{x}_j|l) = sim(\mathbf{x}_j, x^*) \tag{3.13}$$

3. **Label co-occurrence based score:** Given a label l, let $\mathbf{y} \in \{0, 1\}^m$ be such that its ith entry is 1 if the ith training image is tagged with l, and 0 otherwise. We compute co-occurrence score $co\_occur(l_i, l_j)$ between two labels l_i and l_j by computing cosine similarity between their corresponding vectors $\mathbf{y}_i$ and $\mathbf{y}_j$. Now, let $\mathbf{x}_j$ be tagged with labels L_j . We define

$$score_3(\mathbf{x}_j|l) = \max_{l_j \in L_J} co\_occur(l, l_j) \tag{3.14}$$

Intuitively, while $score_3$ tries to address incomplete-labelling and label-ambiguity, $score_1$ and $score_2$ try to address the issues of label-ambiguity and structural-overlap. Based on these three scores, we define the tolerance parameter for sample $\mathbf{x}_j$ given label l as

$$t_j = 1 - \frac{1}{3}\left(score_1(\mathbf{x}_j|l) + score_2(\mathbf{x}_j|l) + score_3(\mathbf{x}_j|l)\right) \tag{3.15}$$

From Eq. 3.15, we can see that for a given sample in S^+ , smaller tolerance value corresponds to a higher chance of it being related to a given label and vice-versa. However, since it is very difficult to claim if a negative sample is actually positive, we still consider $y_j = -1 \forall \mathbf{x}_j \in \bar{S^+}$. This is because our aim is to learn a classifier that is tolerant against confusing labels, rather than getting learned on them. Also, we take $t_j = 1 \ \ \forall \mathbf{x}_j \in S^+$ assuming that all the positive samples are correctly annotated.

3.3.3 Dual form

By rewriting equation 3.11, the dual form of SVM-VT can be easily derived as below:

$$\min_{\mathbf{w},\xi \geq 0} \ \ \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{m}\sum_{j=1}^{m}\xi_j \quad s.t. \ \ \xi_j \geq 1 - y_j t_j(\mathbf{w}\cdot\mathbf{x}_j) \ \ \forall j \in \{1,\ldots,m\} \tag{3.16}$$

$$= \max_{\alpha} \ \ \sum_{i=1}^{m}\alpha_i - \frac{1}{2\lambda}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j t_i t_j(\mathbf{x}_i\cdot\mathbf{x}_j) \ \ s.t. \ \ 0 \leq \alpha_i \leq \frac{1}{m} \ \ \forall i \in \{1,\ldots,m\} \tag{3.17}$$

The dual of SVM-VT is very much similar to that of the conventional SVM, thus making it feasible to implement an efficient optimizer.

3.3.4 Error Bound

For a sample $\mathbf{x}_j$, let us denote the conventional hinge-loss and misclassification error by

$$hinge(\mathbf{w},\mathbf{x}_j,y_j) = [1 - (y_j(\mathbf{w}\cdot\mathbf{x}_j)]_+$$
$$err(\mathbf{w},\mathbf{x}_j,y_j) = \delta(y_j(\mathbf{w}\cdot\mathbf{x}_j) < 0)$$

where $\delta(\cdot)$ is 1 if the argument holds true and 0 otherwise. Then it can be easily shown that the hinge-loss provides an upper-bound on the misclassification error. Now, let us denote the modified hinge-loss of SVM-VT as:

$$t\_hinge(\mathbf{w},\mathbf{x}_j,y_j,t_j) = [1 - y_j t_j(\mathbf{w}\cdot\mathbf{x}_j)]_+$$

Then we get the following proposition, which can be easily verified from Figure 3.6:

Proposition 1 *For $t_j \in [0,1]$, $t\_hinge(\mathbf{w},\mathbf{x}_j,y_j,t_j) \geq err(\mathbf{w},\mathbf{x}_j,y_j)$, i.e. t_hinge provides an upper-bound on the misclassification error.*

Dataset	Images	Training	Testing	Labels	Labels/Image	Images/Label	Labels#
Corel-5K	4999	4500	499	260	3.4, 4, 5	58.6, 22, 1004	195 (75.0%)
ESP-Game	20770	18689	2081	268	4.7, 5, 15	326.7, 172, 4553	201 (75.0%)
IAPR-TC12	19627	17665	1962	291	5.7, 5, 23	347.7, 153, 4999	217 (74.6%)
MIRFlickr-25K	25000	12500	12500	38	4.7, 5, 17	1560.7, 995.5, 5216	22 (57.9%)

Table 3.1 General (columns 2-5) and some insightful (columns 6-8) statistics of the four image annotation datasets considered in our experiments. In columns 6 and 7, the entries are in the format "mean, median, maximum". Column 8 ("Labels#") shows the number of labels whose frequency is less than the mean label frequency.

3.4 Datasets and Their Characteristics

To empirically examine our two proposals (Section 3.2 and Section 3.3), we consider four popular image annotation datasets in our experiments:

- **Corel-5K:** This was introduced by [24], and since then it has become a de facto evaluation benchmark for comparing the annotation performance.
- **ESP-Game:** This was published by [147]. It contains images annotated using an on-line game, where two (mutually unknown) players are randomly given an image, and need to predict the same keyword(s) in order to score points. This way, several people participate in the manual annotation task, thus making this dataset very challenging and diverse.
- **IAPR-TC12:** This was introduced by [34] for cross-lingual information retrieval. In this, each image is associated with a detailed description. Makadia *et al.* [79, 80] extracted nouns from these descriptions and treated them as annotations. Since then, it has been widely used for evaluating image annotation methods.
- **MIRFlickr-25K:** This dataset contains images downloaded from Flickr, and was introduced for evaluating keyword-based image retrieval [46]. In [138], the images of this dataset were manually annotated with 24 concepts for evaluating automatic annotation performance. In the first round of annotation, for each image, the annotators were asked whether it was at least partially relevant

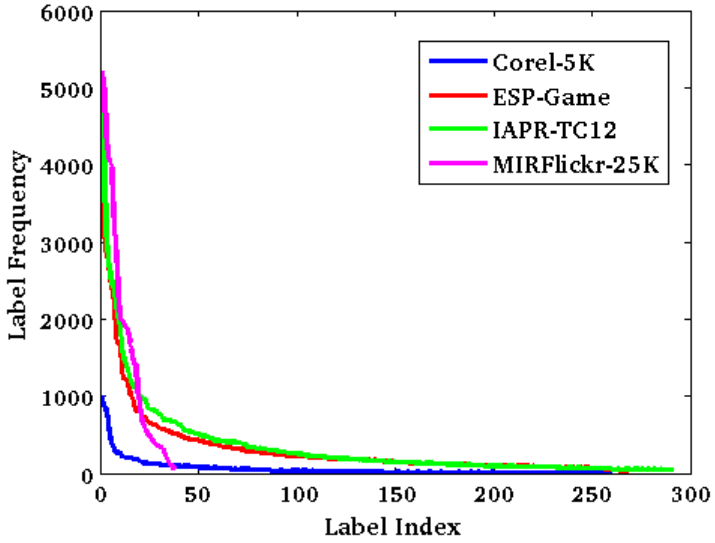


Figure 3.7 Frequency of labels (or, number of images per label) in the training set of each of the four datasets sorted in decreasing order.

for each concept. In the second round, a stricter notion of relevance was used for 14 concepts. For each concept, the images that were annotated as relevant in the first round were considered, and marked as relevant only if that concept was depicted in a significant portion of the image. In this way, each image in this dataset is annotated by its relevance for total 38 labels. Compared to the other three datasets, this dataset has a larger test set, though the number of distinct labels is relatively quite small.

In Table 3.1, columns $2-5$ show some general statistics of the four datasets; and in columns $6-8$, we highlight some other statistics that provide better insights about the properties of these datasets. It can be noticed that for the first three datasets, around 75% of the labels have frequency less than the mean label frequency (column 8), and also the median label frequency is far less than the corresponding mean frequency (column 7). Figure 3.7 shows the frequencies of the labels in these datasets sorted in descending order. The graphs for the Corel-5K, ESP-Game and IAPR TC-12 datasets drop rapidly, and almost level out near the tail (the *long tail* phenomenon [1]). We can also observe that the tail of the MIRFlickr-25K dataset is much higher than that of the other three datasets. Each of those three datasets contains only a small number of high frequency labels, and a huge number of labels with very low frequencies. This suggests that compared to the MIRFlickr-25K dataset, the other three datasets better

model the aspect of class-imbalance. E.g., the most frequent label in the ESP-Game dataset is "man" that has $4,553$ occurrences. Whereas, the second most frequent label "white" is significantly less popular, with $3,175$ occurrences (a drop of around 30%). Moreover, for the Corel-5K, ESP-Game and IAPR-TC12 datasets, the cumulative frequency of the 25 most frequent labels is 52.3%, 45.2% and 45.0% respectively, whereas that of the 25 least frequent labels is just 0.6%, 1.5%, and 1.2% respectively. This indicates that the labels appearing towards the tail (such as "bikini", "icon" and "mail" in the ESP-Game dataset) might be *individually unimportant* in the sense that there are very few images depicting each of these concepts. However, since there are lots of such labels, they become *collectively significant*, and prediction accuracies on these labels have a critical impact on the overall performance.

Though it is not straightforward to quantify incomplete-labelling, we try to analyse it from the number of labels per image (column 6). We posit that a large gap between mean (or median) and maximum number of labels per image indicates that many images are not labelled with all the relevant labels. Based on this, we can infer that both ESP-Game and IAPR-TC12 datasets suffer from incomplete-labelling. For the Corel-5K dataset, we examined the images and their corresponding annotations to realize incomplete-labelling. Unlike these three datasets, since the vocabulary size in the MIRFlickr-25K dataset is very small, and it was annotated under a strict set-up, the chances of incomplete-labelling are rare.

3.5 Features

For image representation, we examine a variety of features. Our first set of features is the standard TagProp-features released by Guillaumin *et al.* [35][3]. These are a combination of local and global features. The local features include the SIFT [78] and robust Hue [135] descriptors obtained densely from multi-scale grid, and from Harris-Laplacian interest points. Each of these descriptors is used to form a histogram of bag-of-words representation. The global features comprise the GIST descriptor [100], and 3-D histograms (with 16-bins per channel) in each of the RGB, HSV and LAB colour spaces. To encode some information about the spatial-layout of an image, all but the GIST descriptor are also computed over three equal horizontal partitions for an image (denoted using "(V3H1)" as a suffix). In this case, the bin-size for colour histograms is reduced to 12-bins per channel to limit histogram sizes.

[3]These features are available at `http://lear.inrialpes.fr/people/guillaumin/data.php`

Feature Name	Dim.	Dist.
(1) Dense SIFT, **(2)** Harris SIFT	1000	χ^2
(3) Dense SIFT (V3H1), **(4)** Harris SIFT (V3H1)	3000	χ^2
(5) GIST	512	L_2
(6) Dense Hue, **(7)** Harris Hue	100	χ^2
(8) Dense Hue (V3H1), **(9)** Harris Hue (V3H1)	300	χ^2
(10) RGB, **(11)** HSV, **(12)** LAB	4096	L_1
(13) RGB (V3H1), **(14)** HSV (V3H1), **(15)** LAB (V3H1)	5184	L_1
(16) Layer-5	9216	L_1
(17) Layer-6, **(18)** Layer-7	4096	L_2
(19) Fisher	65536	L_2
(20) VLAD	32768	L_2

Table 3.2 Different features, their dimensionalities, and distance metrics used for computing pair-wise distances. Top: The fifteen publicly available features provided by [35] ("TagProp-features"). Middle: Features computed using the pre-trained CNN model of [21] ("CNN-features"). Bottom: Features computed using advanced encoding techniques [52, 106] ("Encoding-features").

In addition to the above features, we extract deep learning based features using the pre-trained Convolutional Neural Network (CNN) model of [21]. It is a generic CNN model, and has been found to work well in a variety of visual recognition tasks. In practice, we consider the output of the last three layers of the network as the features.[4] We also compute representations based on two modern encoding techniques: Fisher [106] and VLAD [52]. Following standard practices, for both we consider the 128-dimensional SIFT features and learn a vocabulary of 256 clusters.

Table 3.2 shows the dimensionality of all the features, and the corresponding distance metrics used for computing pair-wise distances. For TagProp-features, we use the same distance metrics as in [35]. For CNN-features, our choice of distance metrics was based on the annotation performance using JEC [79, 80] (since JEC does not involve any learning as such). For Encoding-features, we used L_2 distance following [106]. Also, in our preliminary evaluations using JEC, we empirically observed that

[4] For a more detailed discussion on these, please refer Section 2.2.1.

before computing the pair-wise distances, taking a square-root of each element of some of the feature vectors [5] provided additional boost in the annotation performance. This has a motivation analogous to "power normalization" as described in [106].

3.5.1 Feature Embedding

While there have been a attempts to use features learned using cross-modal embedding directly for the image annotation task [33, 95], in a recent work [6] it was shown that such features can also provide significant improvements in the performance of nearest neighbour based annotation methods such as ours. To examine this, we embed (different combinations of) the above features into a common subspace learned using canonical correlation analysis (CCA) [44], and kernelized canonical correlation analysis (KCCA). This is motivated by the well-known phenomenon of *semantic gap*, because of which it is difficult to build meaningful associations between low-level visual features and high-level semantic concepts/labels. Using cross-modal embeddings learned through (K)CCA, we try to address this partially by learning representations that maximize the correlation between visual and textual content by projecting them into a common subspace.

Let I_i and I_j be two images, both of which are represented using a set of feature vectors h_i^f and h_j^f for all features $\mathbf{f} \in \mathcal{F}$. In the case of CCA, we L_2-normalize each feature and use a linear kernel to compute similarity between two images:

$$K_v^{cca}(I_i, I_j) = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} < h_i^f, h_j^f > \tag{3.18}$$

And for KCCA, we use an exponential kernel:

$$K_v^{kcca}(I_i, I_j) = exp\left(\frac{-D(I_i, I_j)}{Z}\right) \tag{3.19}$$

where $D(I_i, I_j)$ is the distance between the two images computed using Eq. 3.7, and Z is the mean distance among all the training pairs.

For textual features, we represent the labels $Y \subseteq \mathcal{Y}$ that are associated with an image I using a binary vector $g \in \mathbb{R}^l$ (l is the vocabulary size). We keep $g(k) = 1$ if $y_k \in Y$, and 0 otherwise. With this, we use a linear kernel to compute similarity between two textual features, which is the same as counting the number of common labels between two images:

$$K_t(g_i, g_j) =< g_i, g_j >= \sum_{k=1}^{l} g_i(k) g_j(k) \tag{3.20}$$

[5] Features indexed by (3), (4), and (9) to (15) in Table 3.2.

We use the implementation of [40] for learning the common subspace (the common embedding space which is learned using (K)CCA, where one can directly compute similarity between samples from heterogeneous data). It penalizes the norm of the projection vectors, and regularizes the learning to avoid trivial solutions. Similar to [6], we set the precision parameter for Gram-Schmidt decomposition to be $\zeta = 30$, and the regularization parameter to be $\kappa = 0.1$. Also, the visual and textual spaces are swapped before computing the projection vectors. In practice, we consider the full-length feature vectors in the common subspace. After embedding the samples, we compute the distance between two feature vectors using the L_2 distance metric, that was empirically chosen based on the annotation performance of the JEC method [79, 80].

Intuitively, the image representation in the learned subspace better captures the semantics of the data, and the neighbouring samples are semantically more meaningful than those obtained using raw features. As validated in our experiments, this in turn significantly improves the annotation performance. Another practically useful advantage of feature embedding is that it provides a very compact yet effective representation for images. E.g., if we concatenate all the features as mentioned in Table 3.2, an image would be represented by a 152864-dimensional feature vector. Whereas, after feature embedding, this reduces to just a few hundreds of dimensions (or even less).

3.6 Experimental Set-up

3.6.1 Implementation Details

- For the original features (without cross-modal embedding), we learn the distance metric in 2PKNN using stochastic gradient descent on random batches of 1000 samples in leave-one-out manner. For 2PKNN without metric learning, the average distance using all the features is considered while determining the neighbours (similar to JEC [79, 80]). Also, analogous to [35], this is scaled by a linear factor π that controls the decay of θ_{J,I_i}. (Eq. 3.3). For each dataset, the K_1 parameter is set by doing cross-validation on training data in the range $\{1, 2, 3, 4, 5\}$.

- To evaluate JEC [79, 80], we implement this method following the steps outlined in the paper. For a given test image, in order to predict a ranking of labels rather than a fixed set of five labels, the number of nearest neighbours used is ensured to be sufficient to see enough unique labels (as followed in [79, 80]).

- To evaluate the variants of TagProp [35], we use the publicly available code[6], and cross-validate number of neighbours in the range $K = \{10, 20, \ldots, 200\}$.
- For SVM and SVM-VT, we use use the VLFeat library [137], and validate the "C" parameter in the range $\{10^{-5}, 10^{-4}, \ldots, 10^{5}\}$. For both, we calibrate the prediction scores using [108].
- In the case of a common space learned using KCCA (Eq. 3.19), we use distance computed using both without and with the learned distance metric for TagProp and 2PKNN.

3.6.2 Evaluation Measures

To analyse the annotation performance, we compute precision and recall of each label in a dataset following previous methods [28, 35, 79, 80, 160]. Suppose a label y_i is present in the ground-truth of m_1 images, and it is predicted for m_2 images during testing, out of which m_3 predictions are correct ($m_3 \leq m_2$ and $m_3 \leq m_1$). Then its precision will be $= m_3/m_2$, and recall will be $= m_3/m_1$. We average these values over all the labels in a dataset and get (percentage) mean precision P and mean recall R. Using these two scores, we compute F1 score, which is the harmonic mean of P and R; i.e., F1 $= 2{\cdot}$P$\cdot$R/(P+R). This takes care of the tradeoff between precision and recall. We also consider N+, the number of labels that are correctly assigned to at least one test image (in other words, the number of labels with recall greater than zero), as an evaluation metric. This measure is particularly useful in case of class-imbalance, where frequent labels can suppress the recall of rare labels. Thus, different image annotation methods are compared using F1 and N+ scores.

Since image annotation has close parallels with the label ranking task, we additionally report mean average precision (mAP) scores for various methods. However, this is probably not a suitable measure for evaluating image annotation methods as we will describe in Section 3.7.3.

3.6.3 Feature Combinations

For evaluation/comparisons, we consider seven sets of features: (a) TagProp-features (denoted by T) that include fifteen features (features indexed from (1) to (15) in Table 3.2), (b) CNN-features (denoted by C) that include three features (features indexed from (16) to (18) in Table 3.2), (c) Encoding-features (denoted by E) that include two features (features indexed from (19) to (20) in Table 3.2), (d) Combined TagProp and CNN features (denoted by T+C) that include eighteen features, (e) Combined TagProp and

[6]The code is available at `http://lear.inrialpes.fr/people/guillaumin/code.php`

Encoding features (denoted by T+E) that include seventeen features, (f) Combined CNN and Encoding features (denoted by by C+E) that include five features, and (g) Combined TagProp, CNN and Encoding features (denoted by T+C+E) that include twenty features.

3.7 Results and Discussion

Here, first we perform quantitative comparisons with two state-of-the-art nearest neighbour based image annotation methods: JEC [79, 80], and the variants of TagProp [35] under different settings. We consider four variants of TagProp: (a) TagProp-SD that simply uses scaled average distance computed using different features, (b) TagProp-σSD that uses scaled average distance and label-specific sigmoid functions to boost the recall of rare labels, (c) TagProp-ML that learns a distance metric, and (d) TagProp-σML that learns both a distance metric as well as label-specific sigmoid functions. Using all the methods, we annotate each test image with five labels.

Then we compare our results with the reported results of the recent as well as some benchmark methods. Finally we discuss/analyze different aspects of 2PKNN and SVM-VT.

3.7.1 Features and Feature Embeddings

First we evaluate the performance of different methods using different features and feature embeddings: JEC (Table 3.3), TagProp-SD (Table 3.4), TagProp-σSD (Table 3.5), TagProp-ML (Table 3.6), TagProp-σML (Table 3.7), 2PKNN (Table 3.8), 2PKNN+ML (Table 3.9), and SVM and SVM-VT (Table 3.10[7]). Finally, we compare the best performance of each method (giving preference to F1 score over N+) in Table 3.11.

From the results, we can observe that in case of TagProp, learning sigmoid functions usually improves the performance. Also, the performance improves with metric learning for both TagProp and 2PKNN. In general, we can notice that the best performing features usually vary for different methods, which indicates the importance of using different features for different methods. In most of the cases, we can observe that T+C+E features are more useful on the Corel-5K dataset, C+E on the ESP-Game dataset, T+C/T+C+E on the IAPR-TC12 dataset, and C+E on the MIRFlickr-25K dataset. From these results, we can conclude that combinations of both learned as well hand-crafted features can be useful in achieving good results.

[7]Since features learned using KCCA embedding were found to provide the best results for JEC, TagProp and 2PKNN, we evaluated SVM and SVM-VT using only these features.

Table 3.3 Annotation performance of JEC [79, 80] using different features, feature embeddings learned via CCA, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

Dataset →		**Corel-5K**				**ESP-Game**				**IAPR-TC12**				**MIRFlickr-25K**			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	31	36	33.3	148	24	19	21.2	220	31	20	24.3	219	27	24	25.4	36
	C	27	35	30.5	140	27	22	24.2	222	33	21	25.7	222	41	38	**39.4**	**37**
	E	22	26	23.8	126	23	18	20.2	215	24	17	19.9	203	25	25	25.0	36
	T+C	32	39	35.2	153	26	21	23.2	**224**	33	21	25.7	221	31	28	29.4	**37**
	T+E	32	36	33.9	150	25	20	22.2	223	31	21	25.0	220	28	25	26.4	**37**
	C+E	30	38	33.5	147	28	23	**25.3**	223	35	23	**27.8**	**228**	41	38	**39.4**	**37**
	T+C+E	34	41	**37.2**	**159**	26	21	23.2	222	33	22	26.4	222	32	29	30.4	**37**
Feature embedding via CCA	T	29	31	30.0	131	26	18	21.3	216	29	12	17.0	188	34	28	30.7	36
	C	26	21	23.2	97	38	18	24.4	214	20	09	12.4	161	49	37	42.2	37
	E	22	24	23.0	114	34	18	23.5	212	39	16	22.7	188	37	29	32.5	37
	T+C	33	33	33.0	133	38	20	26.2	**223**	40	14	20.7	180	46	37	41.0	36
	T+E	31	32	31.5	133	34	19	24.4	219	38	19	25.3	200	38	31	34.1	**38**
	C+E	30	25	27.3	107	49	18	26.3	209	45	16	23.6	185	50	38	**43.2**	37
	T+C+E	35	34	**34.5**	**136**	43	20	**27.3**	**223**	44	19	**26.5**	**202**	46	38	41.6	36
Feature embedding via KCCA	T	39	39	39.0	148	46	20	27.9	217	44	22	29.3	206	40	32	35.6	**38**
	C	34	34	34.0	132	48	23	31.1	219	46	22	29.8	211	53	41	**46.2**	**38**
	E	23	29	25.7	123	30	22	25.4	216	36	21	26.5	202	40	31	34.9	36
	T+C	41	43	**42.0**	**156**	47	23	30.9	219	47	24	**31.8**	**218**	47	40	43.2	**38**
	T+E	39	39	39.0	148	47	21	29.0	219	45	22	29.6	209	42	34	37.6	**38**
	C+E	32	35	33.4	131	50	23	31.5	218	46	23	30.7	210	51	42	46.1	37
	T+C+E	41	43	**42.0**	155	48	24	**32.0**	**224**	47	24	**31.8**	214	48	40	43.6	**38**

Table 3.4 Annotation performance of TagProp-SD [35] using different features, feature embeddings learned via CCA, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

Dataset →		Corel-5K				ESP-Game				IAPR-TC12				MIRFlickr-25K			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	31	31	31.0	122	38	21	27.1	218	49	20	28.4	201	47	29	35.9	34
	C	25	28	26.4	107	45	21	**28.6**	195	44	23	30.2	**206**	56	46	50.5	35
	E	20	24	21.8	115	39	17	23.7	202	43	18	25.4	196	41	29	34.0	33
	T+C	35	37	**36.0**	133	42	19	26.2	180	47	23	30.9	197	47	39	42.6	**36**
	T+E	32	34	33.0	133	40	20	26.7	**214**	50	20	28.6	202	46	30	36.3	33
	C+E	25	32	28.1	116	43	21	28.2	192	46	23	30.7	200	56	47	**51.1**	35
	T+C+E	34	38	35.9	**136**	42	18	25.2	178	48	23	**31.1**	202	51	41	45.5	35
Feature embedding via CCA	T	28	25	26.4	110	37	22	27.6	230	40	25	30.8	244	34	32	33.0	**38**
	C	21	14	16.8	73	52	22	30.9	227	48	21	29.2	223	58	45	50.7	**38**
	E	21	20	20.5	93	34	22	26.7	231	46	23	30.7	227	40	36	37.9	37
	T+C	33	33	33.0	132	43	25	31.6	233	48	26	33.7	**245**	50	46	47.9	**38**
	T+E	30	32	31.0	130	38	25	30.2	232	48	27	34.6	239	38	37	37.5	**38**
	C+E	28	24	25.8	103	54	22	31.3	225	56	22	31.6	220	55	48	**51.3**	**38**
	T+C+E	33	35	**34.0**	**135**	46	26	**33.2**	**235**	52	27	**35.5**	241	49	48	48.5	**38**
Feature embedding via KCCA	T	35	37	36.0	137	47	26	33.5	237	51	33	40.1	252	42	42	42.0	**38**
	C	32	34	33.0	130	50	30	37.5	**242**	53	32	39.9	250	51	53	**52.0**	**38**
	E	21	29	24.4	123	34	28	30.7	240	42	30	35.0	250	39	40	39.5	**38**
	T+C	27	22	24.2	96	47	29	35.9	239	53	36	42.9	**257**	48	51	49.5	**38**
	T+E	35	38	**36.4**	**140**	48	27	34.6	237	52	34	41.1	249	43	44	43.5	**38**
	C+E	31	32	31.5	116	52	30	**38.0**	238	54	34	41.7	252	50	53	51.5	**38**
	T+C+E	35	33	34.0	124	49	30	37.2	239	54	36	**43.2**	256	49	52	50.5	**38**

Table 3.5 Annotation performance of TagProp-σSD [35] using different features, feature embeddings learned via CCA, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

Dataset →		**Corel-5K**				**ESP-Game**				**IAPR-TC12**				**MIRFlickr-25K**			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	32	32	32.0	125	37	21	26.8	**220**	50	20	28.6	207	48	32	38.4	35
	C	25	28	26.4	109	45	21	28.6	195	44	24	31.1	**209**	57	46	50.9	36
	E	23	25	24.0	118	40	17	23.9	206	43	19	26.4	198	42	29	34.3	33
	T+C	35	37	36.0	138	46	21	**28.8**	203	49	23	**31.3**	207	54	48	50.8	**37**
	T+E	33	35	34.0	135	43	21	28.2	219	49	21	29.4	203	50	33	39.8	34
	C+E	26	32	28.7	119	44	21	28.4	199	47	23	30.9	203	56	47	51.1	35
	T+C+E	36	39	**37.4**	**142**	46	20	27.9	199	48	23	31.1	207	58	47	**51.9**	36
Feature embedding via CCA	T	31	31	31.0	133	33	25	28.4	237	37	28	31.9	**258**	35	34	34.5	**38**
	C	27	22	24.2	102	47	25	32.6	237	46	25	32.4	245	57	48	52.1	**38**
	E	23	25	24.0	109	32	24	27.4	239	45	25	32.1	239	42	38	39.9	**38**
	T+C	33	34	33.5	133	41	27	32.6	**241**	47	30	36.6	256	51	48	49.5	**38**
	T+E	32	33	32.5	136	37	26	30.5	240	47	31	37.4	256	43	38	40.3	**38**
	C+E	29	25	26.9	108	53	23	32.1	231	58	24	34.0	235	58	49	**53.1**	**38**
	T+C+E	34	35	**34.5**	**139**	45	28	**34.5**	**241**	52	31	**38.8**	255	51	49	50.0	**38**
Feature embedding via KCCA	T	37	40	38.4	145	46	28	34.8	241	50	37	42.5	261	41	42	41.5	**38**
	C	33	35	34.0	134	50	31	38.3	243	52	35	41.8	262	52	53	**52.5**	**38**
	E	22	30	25.4	125	30	31	30.5	242	39	34	36.3	261	38	40	39.0	**38**
	T+C	40	42	41.0	150	47	31	37.4	242	52	39	44.6	263	48	51	49.5	**38**
	T+E	35	36	35.5	136	48	28	35.4	243	52	37	43.2	260	41	43	42.0	**38**
	C+E	32	36	33.9	133	51	31	**38.6**	243	53	37	43.6	258	50	53	51.5	**38**
	T+C+E	41	43	**42.0**	**155**	48	31	37.7	**245**	53	40	**45.6**	**265**	49	51	50.0	**38**

Table 3.6 Annotation performance of TagProp-ML [35] using different features, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

	Dataset →	**Corel-5K**				**ESP-Game**				**IAPR-TC12**				**MIRFlickr-25K**			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	32	42	36.3	155	39	24	29.7	232	46	34	39.1	263	44	34	38.4	37
	C	30	42	35.0	156	37	32	34.3	**244**	42	34	37.6	262	59	50	**54.1**	**38**
	E	21	27	23.6	121	35	21	26.2	226	40	27	32.2	246	41	31	35.3	37
	T+C	36	48	**41.1**	**167**	38	30	33.5	230	49	37	42.2	268	51	41	45.5	**38**
	T+E	32	43	36.7	159	40	25	30.8	235	47	35	40.1	263	45	36	40.0	37
	C+E	31	43	36.0	154	38	32	**34.7**	243	45	37	40.6	263	59	50	**54.1**	**38**
	T+C+E	35	47	40.1	166	39	30	33.9	234	49	38	**42.8**	**270**	53	44	48.1	**38**
Feature embedding via KCCA	T	34	35	34.5	**132**	50	25	33.3	235	50	35	41.2	256	43	45	44.0	**38**
	C	33	33	33.0	130	48	32	38.4	245	50	34	40.5	**262**	52	53	52.5	**38**
	E	21	28	24.0	121	31	28	29.4	237	45	29	35.3	241	40	40	40.0	37
	T+C	37	35	36.0	127	47	32	38.1	245	55	36	43.5	259	55	54	**54.5**	**38**
	T+E	33	34	33.5	130	53	24	33.0	233	51	34	40.8	250	44	45	44.5	**38**
	C+E	30	31	30.5	113	50	31	38.3	245	54	33	41.0	247	53	53	53.0	**38**
	T+C+E	37	36	**36.5**	131	49	32	**38.7**	**248**	56	36	**43.8**	254	55	54	**54.5**	**38**

Now we examine the results with features learned using feature embeddings. Note that in case of CCA, we do not include metric learning since it involves a linear kernel. From these results, we can make the following observations: (1) After CCA embedding, the performance generally reduces by a small amount compared to that using the original features. (2) Using KCCA embedding, the performance of all the methods generally improves in terms of F1 score (sometimes by a large margin) compared to both CCA embedding as well as original features, thus demonstrating the advantage of learning kernelized cross-modal embedding. (3) In some cases, the performance drops in terms of N+. This could be because the learned embedding does not efficiently capture the semantics of the labels with relatively low frequency. (4) Using KCCA, the performance after metric learning usually improves for both TagProp and 2PKNN. This indicates that learning a distance metric in the embedded space

Table 3.7 Annotation performance of TagProp-σML [35] using different features, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

Dataset →		Corel-5K				ESP-Game				IAPR-TC12				MIRFlickr-25K			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	33	43	37.3	160	41	24	30.3	233	48	34	39.8	266	47	38	42.0	37
	C	31	43	36.0	157	38	33	35.3	245	43	35	38.6	266	59	50	54.1	**38**
	E	22	29	25.0	124	35	22	27.0	230	39	28	32.6	253	41	32	35.9	37
	T+C	36	50	41.9	**175**	39	31	34.5	244	48	39	**43.0**	**276**	59	50	54.1	**38**
	T+E	34	44	38.4	164	41	26	31.8	239	47	36	40.8	270	47	39	42.6	37
	C+E	32	44	37.1	160	39	33	**35.7**	**246**	45	38	41.2	267	59	51	**54.7**	**38**
	T+C+E	38	48	**42.4**	170	39	32	35.2	244	48	39	**43.0**	**276**	58	51	54.3	**38**
Feature embedding via KCCA	T	37	39	38.0	144	50	26	34.2	240	48	39	43.0	263	40	44	41.9	37
	C	35	35	35.0	135	46	33	38.4	247	47	39	42.6	**274**	55	51	52.9	**38**
	E	21	29	24.4	123	28	31	29.4	242	44	33	37.7	256	40	40	40.0	**38**
	T+C	43	44	**43.5**	158	47	35	**40.1**	**249**	54	40	46.0	269	55	55	**55.0**	**38**
	T+E	34	37	35.4	140	53	25	34.0	237	50	38	43.2	262	42	43	42.5	**38**
	C+E	32	35	33.4	134	50	33	39.8	245	55	36	43.5	256	53	54	53.5	**38**
	T+C+E	42	45	43.4	**160**	48	34	39.8	**249**	56	40	**46.7**	268	55	55	**55.0**	**38**

can benefit from a learned distance metric. (5) SVM-VT consistently improves performance over the conventional one-vs-rest SVM. Also, it performs either comparable to or better than JEC and TagProp-SD. In terms of N+, SVM-VT beats all other methods including 2PKNN, except on the MIRFlickr-25K dataset where all the methods achieve perfect recall because of small vocabulary. (6) In terms of F1 score, the performance of 2PKNN+ML is consistently better than the other methods on the Corel-5K and ESP-Game datasets, comparable to TagProp-σML on the MIRFlickr-25K dataset, and inferior to TagProp-σSD and TagProp-σML on the IAPR-TC12 dataset by around 0.6% and 1.6% respectively.

Based on the above observations, we will use the features that give the best performance using the KCCA embedding, giving preference to F1 score over N+.

Table 3.8 Annotation performance of 2PKNN using different features, feature embeddings learned via CCA, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

	Dataset →	**Corel-5K**				**ESP-Game**				**IAPR-TC12**				**MIRFlickr-25K**			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	40	41	40.5	180	40	25	30.8	248	49	30	37.2	275	39	29	33.3	**38**
	C	39	45	41.8	189	50	29	**36.7**	**251**	54	31	39.4	271	57	45	50.3	**38**
	E	26	27	26.5	142	41	24	30.3	248	45	26	33.0	269	37	33	34.9	**38**
	T+C	42	42	42.0	187	56	19	28.4	243	56	29	38.2	275	46	33	38.4	**38**
	T+E	37	43	39.8	178	44	25	31.9	249	47	32	38.1	275	40	30	34.3	**38**
	C+E	40	43	41.4	**192**	53	27	35.8	246	53	33	**40.7**	**276**	56	46	**50.5**	**38**
	T+C+E	43	42	**42.5**	188	57	19	28.5	240	58	28	37.8	275	46	34	39.1	**38**
Feature embedding via CCA	T	36	35	35.5	155	41	23	29.5	249	50	19	27.5	256	38	35	36.4	**38**
	C	37	32	34.3	154	50	25	33.3	243	50	14	21.9	244	57	47	51.5	**38**
	E	29	33	30.9	143	39	24	29.7	248	48	27	34.6	255	38	39	38.5	**38**
	T+C	41	41	41.0	170	44	27	33.5	**251**	57	22	31.7	250	48	47	47.5	**38**
	T+E	39	37	38.0	161	41	25	31.1	247	50	29	36.7	**266**	40	40	40.0	**38**
	C+E	40	37	38.4	159	58	25	34.9	247	60	26	36.3	256	53	51	**52.0**	**38**
	T+C+E	42	42	**42.0**	**176**	50	27	**35.1**	250	54	31	**39.4**	263	48	49	48.5	**38**
Feature embedding via KCCA	T	45	44	44.5	176	45	29	35.3	251	50	37	42.5	273	43	44	43.5	**38**
	C	41	43	42.0	172	51	32	39.3	253	51	35	41.5	269	51	56	53.4	**38**
	E	32	34	33.0	155	39	30	33.9	251	45	32	37.4	272	41	43	42.0	**38**
	T+C	47	50	48.5	187	45	32	37.4	251	53	39	**44.9**	**279**	48	54	50.8	**38**
	T+E	45	46	45.5	179	45	30	36.0	252	51	38	43.6	272	43	46	44.4	**38**
	C+E	42	45	43.4	180	49	35	**40.8**	**255**	52	37	43.2	273	51	57	**53.8**	**38**
	T+C+E	48	50	**49.0**	**190**	43	33	37.3	250	53	38	44.3	275	48	55	51.3	**38**

Table 3.9 Annotation performance of 2PKNN+ML using different features, and feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

Dataset →		Corel-5K				ESP-Game				IAPR-TC12				MIRFlickr-25K			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
No feature embedding	T	41	46	43.4	186	43	26	32.4	246	53	32	39.9	**277**	41	34	37.2	**38**
	C	38	47	42.0	189	47	31	37.4	253	51	33	40.1	273	58	46	51.3	**38**
	E	26	27	26.5	144	41	24	30.3	248	45	26	33.0	270	36	33	34.4	**38**
	T+C	40	51	**44.8**	**196**	49	30	37.2	254	49	36	**41.5**	**277**	58	46	51.3	**38**
	T+E	40	47	43.2	185	44	26	32.7	250	49	34	40.1	**277**	41	34	37.2	**38**
	C+E	39	47	42.6	192	49	31	**38.0**	**255**	48	36	41.1	276	57	47	**51.5**	**38**
	T+C+E	40	50	44.4	194	48	30	36.9	252	49	35	40.8	276	57	47	**51.5**	**38**
Feature embedding via KCCA	T	45	47	46.0	186	45	29	35.3	249	52	37	43.2	272	39	42	40.4	**38**
	C	43	44	43.5	177	50	33	39.8	253	52	36	42.5	270	53	56	54.5	**38**
	E	32	34	33.0	155	39	30	33.9	251	45	32	37.4	272	41	44	42.4	**38**
	T+C	48	52	**49.9**	191	44	34	38.4	**255**	50	41	**45.1**	275	52	56	53.9	**38**
	T+E	45	48	46.5	185	46	31	37.0	251	53	38	44.3	272	41	43	42.0	**38**
	C+E	44	47	45.5	182	48	36	**41.1**	**255**	53	38	44.3	274	53	57	**54.9**	**38**
	T+C+E	48	52	**49.9**	**196**	46	33	38.4	254	50	40	44.4	**276**	52	56	53.9	**38**

3.7.2 Comparison with Previous Results

Table 3.12 summarizes our best results as well as those reported by the previous methods. Similar to all other methods, we assign the top five labels to each test image, and the average precision and recall values are computed by averaging over all the labels in a dataset. Also, here we do not consider the MIRFlickr-25K dataset since the other three datasets are more popular and challenging, and none of the listed methods has evaluated on it. From these results, we can observe that on all the three datasets, the 2PKNN method outperforms the previous methods in terms of F1 score. Precisely, for the Corel-5K, ESP-Game and IAPR- TC12 datasets, we achieve 3.4%, 0.7% and 3.9% of absolute improvements respectively over the previous best results. In terms of N+, the performance of SVM-VT is inferior only to the recent methods SVM-DMBRM [94] and CCA-KNN method [96] on the Corel-5K dataset.

Table 3.10 Annotation performance of SVM [16,136] and SVM-VT using feature embeddings learned via KCCA. The best F1 and N+ scores in each block are highlighted in bold.

	Dataset →	Corel-5K				ESP-Game				IAPR-TC12				MIRFlickr-25K			
	Ftrs.	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
SVM	T	29	41	34.0	167	29	31	30.0	260	36	39	37.4	281	39	49	43.4	**38**
	C	28	44	34.2	180	28	41	33.3	259	31	43	36.0	282	43	57	49.0	**38**
	E	20	31	24.3	135	29	31	30.0	255	33	37	34.9	282	37	48	41.8	**38**
	T+C	30	50	37.5	**187**	31	35	32.9	260	38	42	39.9	**285**	41	57	47.7	**38**
	T+E	29	47	35.9	180	27	34	30.1	259	37	41	38.9	281	39	49	43.4	37
	C+E	28	46	34.8	174	32	39	**35.2**	**262**	35	43	38.6	283	43	58	**49.4**	**38**
	T+C+E	32	46	**37.7**	179	32	36	33.9	260	37	44	**40.2**	**285**	41	57	47.7	**38**
SVM-VT	T	29	46	35.6	180	34	30	31.9	264	38	39	38.5	284	39	50	43.8	**38**
	C	29	46	35.6	185	34	37	35.4	266	35	40	37.3	285	43	58	49.4	**38**
	E	21	35	26.2	144	30	31	30.5	261	36	36	36.0	283	39	47	42.6	**38**
	T+C	31	51	38.6	190	33	35	34.0	264	38	43	40.3	**286**	43	56	48.6	**38**
	T+E	30	48	36.9	187	32	32	32.0	266	39	40	39.5	283	40	51	44.8	**38**
	C+E	28	49	35.6	183	34	40	**36.8**	**267**	38	42	39.9	284	44	58	**50.0**	**38**
	T+C+E	31	53	**39.1**	**195**	33	37	34.9	266	40	41	**40.5**	285	43	57	49.0	**38**

Also, in terms of F1 score, SVM-VT is inferior only to CCA-KNN [96] on all the three datasets, and SKL-CRM [91] and SVM-DMBRM [94] on the Corel-5K dataset. However, since the F1 scores of 2PKNN are much better than SVM-VT, SVM-DMBRM and CCA-KNN, this indicates that a higher N+ by these methods is probably because of correctly recalling a small number of instances for several of the recalled labels. Whereas, though 2PKNN is able to recall a slightly less number of labels, the number of correctly recalled instances is probably higher than the other three, thus resulting in better F1 scores.

In Table 3.13, we compare our results with the recent work of [60]. We compare this work separately because their evaluation criteria is different from other methods compared in Table 3.12. Precisely, here also we annotate each test image with the top five labels. However, here we compute the average

Table 3.11 Comparison of annotation performance of different methods. Note that for each method, we consider its best performance across different features via KCCA embedding. Since each method may use different features, the results need not always be directly comparable. (Refer to the previous tables for comprehensive comparisons across various features.)

Dataset →	**Corel-5K**				**ESP-Game**				**IAPR-TC12**				**MIRFlickr-25K**			
Method ↓	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
JEC [79, 80]	41	43	42.0	156	48	24	32.0	224	47	24	31.8	218	53	41	46.2	**38**
SVM [16, 136]	32	46	37.7	**179**	32	39	35.2	**262**	37	44	40.2	**285**	43	58	49.4	**38**
TagProp-SD [35]	35	38	36.4	140	52	30	38.0	238	54	36	43.2	256	51	53	52.0	**38**
TagProp-σSD [35]	41	43	42.0	155	51	31	38.6	243	53	40	45.6	265	52	53	52.5	**38**
TagProp-ML [35]	37	36	36.5	131	49	32	38.7	248	56	36	43.8	254	55	54	54.5	**38**
TagProp-σML [35]	43	44	**43.5**	158	47	35	**40.1**	249	56	40	**46.7**	268	55	55	**55.0**	**38**
SVM-VT (Ours)	31	53	39.1	195	34	40	36.8	**267**	40	41	40.5	**285**	44	58	50.0	**38**
2PKNN (Ours)	48	50	49.0	190	49	35	40.8	255	53	39	44.9	279	51	57	53.8	**38**
2PKNN+ML (Ours)	48	52	**49.9**	**196**	48	36	**41.1**	255	50	41	**45.1**	275	53	57	**54.9**	**38**

precision and recall values using only the labels with positive recall following [60]. In terms of both F1 and N+ scores, our method significantly outperforms the NMF-KNN approach. Note that since the average precision and recall values are computed only for the correctly recalled labels, the N+ score acts as a divisive factor while computing the F1 score (from that computed for all the labels). Even with a big margin of $40/46$ on the Corel-5K dataset and 17 on the ESP-Game dataset in terms of N+, our F1 scores are better than NMF-KNN by a factor of around 1.5.

From these comparisons, we can conclude that the 2PKNN and 2PKNN+ML methods along with the new image features achieve state-of-the-art results on all the three prevailing image annotation datasets in terms of F1 score, and competitive performance in terms of N+ score. Also, while SVM-VT does not achieve as high F1 scores as 2PKNN, it is still competitive to other methods.

Dataset →	Corel-5K				ESP-Game				IAPR-TC12			
Method↓	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
CRM [70]	16	19	17.4	107	–	–	–	–	–	–	–	–
MBRM [28]	24	25	24.5	122	18	19	18.5	209	24	23	23.5	223
InfNet [87]	17	24	19.9	112	–	–	–	–	–	–	–	–
NPDE [158]	18	21	19.4	114	–	–	–	–	–	–	–	–
SML [14]	23	29	25.7	137	–	–	–	–	–	–	–	–
TGLM [77]	25	29	26.9	131	–	–	–	–	–	–	–	–
JEC [79, 80]	27	32	29.3	139	23	19	20.8	227	25	16	19.5	196
MRFA [154]	31	36	33.3	172	–	–	–	–	–	–	–	–
CCD (SVRMKL+KPCA) [98]	36	41	38.3	159	36	24	28.8	232	44	29	35.0	251
GroupSparsity [160]	30	33	31.4	146	–	–	–	–	32	29	30.4	252
BS-CRM [90]	22	27	24.2	130	–	–	–	–	24	22	23.0	250
RandomForest [29]	29	40	33.6	157	41	26	31.8	235	44	31	36.4	253
TagProp-SD [35]	30	33	31.4	136	48	19	27.2	212	50	20	28.6	215
TagProp-σSD [35]	28	35	31.1	145	39	24	29.7	232	41	30	34.6	259
TagProp-ML [35]	31	37	33.7	146	49	20	28.4	213	48	25	32.9	227
TagProp-σML [35]	33	42	37.0	160	39	27	31.9	239	46	35	39.8	266
FastTag [15]	32	43	36.7	166	46	22	29.8	247	47	26	33.9	280
SKL-CRM [91]	39	46	42.2	184	41	26	31.8	248	47	32	38.1	274
SVM-DMBRM [94]	36	48	41.1	197	55	25	34.4	259	56	29	38.2	**283**
CCA-KNN [96]	42	52	**46.5**	**201**	46	36	**40.4**	**260**	45	38	**41.2**	278
HHD [97]	31	49	38.0	194	35	36	35.5	257	32	44	37.1	280
SVM-VT	31	53	39.1	195	34	40	36.8	**267**	40	41	40.5	**285**
2PKNN	48	50	49.0	190	49	35	40.8	255	53	39	44.9	279
2PKNN+ML	48	52	**49.9**	**196**	48	36	**41.1**	255	50	41	**45.1**	275

Table 3.12 Comparison of our best results with the reported results of some of the benchmark and recent image annotation methods. Note that the results for SVM-VT are different from those reported in [141] since we use different features here. Also, the results for 2PKNN and 2PKNN+ML are different from [140] and are the same as in [144]. The best F1 and N+ scores are highlighted in bold.

Dataset →	Corel-5K				ESP-Game			
Method↓	P	R	F1	N+	P	R	F1	N+
NMF-KNN [60]	38	56	45.3	150	33	26	29.1	238
SVM-VT	41	70	51.7	195	34	40	36.8	267
2PKNN	65	68	66.5	190	51	37	42.9	255
2PKNN+ML	63	69	65.9	196	50	38	43.2	255

Table 3.13 Comparison of our best performance with the reported results of NMF-KNN [60]. Similar to [60], here the average precision and recall values are computed using only the labels with positive recall, and not all the labels as followed by the methods in Table 3.12.

3.7.3 Label Ranking Performance

Finally, in Table 3.14, we compare the label ranking performance of different methods using mAP as the evaluation metric. Here, we observe that the performance of 2PKNN(+ML) is far inferior than the TagProp variants. Also, SVM-VT performs better than SVM on the Corel-5K dataset and worse on the remaining three datasets, thus not showing a consistent behaviour. However, it is worth noticing that there are fundamental distinctions between the label ranking and image annotation tasks. First, in image annotation, there is no discrimination among the predicted labels based on their order/rank of prediction, whereas label ranking does make such a discrimination. Second, in image annotation, the performance (P, R, F1 and N+) is evaluated on a per-label basis, thus giving more importance to rare labels than the frequent ones. While in label ranking, the performance (mAP) is evaluated on a per-image basis, thus treating all the labels of an image equally. Since it is easier to predict frequent labels than the rare ones, this in turn contributes in increasing the overall mAP. This is also supported from the consistent drop in mAP on using the sigmoid variants of TagProp compared to non-sigmoid variants, that in fact gave better annotation performance (Table 3.11). Hence, mAP may not be as appropriate as F1 and N+ measures for comparing image annotation methods.

3.7.4 Discussion on 2PKNN

In this subsection, we analyse/discuss different aspects of the proposed 2PKNN method, such as computational cost, diversity of labels among the neighbours obtained after the first pass of 2PKNN, and effect of the parameter K_1 on performance. We also compare these aspects with competing nearest neighbour based methods JEC [79, 80] and TagProp [35] wherever applicable.

Dataset →	Corel-5K	ESP-Game	IAPR-TC12	MIRFlickr-25K
JEC	52.2	33.6	38.8	62.2
SVM	28.6	31.3	32.2	69.1
TagProp-SD	60.3	46.1	54.9	74.3
TagProp-σSD	59.1	45.5	53.7	68.1
TagProp-ML	65.1	49.2	55.4	73.6
TagProp-σML	61.1	47.2	54.7	71.6
SVM-VT	35.4	28.1	28.8	59.1
2PKNN	53.4	46.0	54.0	69.4
2PKNN+ML	55.7	47.9	54.7	70.4

Table 3.14 Performance comparison in terms of %mAP for the label ranking task.

3.7.4.1 Tradeoff Between Precision and Recall

It is a popular and well-accepted practice among the image annotation methods (such as those listed in Table 3.12) to annotate each image with five labels for comparisons. However, it may not always be justifiable since the actual number of labels can vary significantly for different images. E.g., some (training) images in the ESP-Game and IAPR-TC12 datasets have up to 15 and 23 labels respectively. Assigning only 5 labels artificially restricts the number of labels that can be correctly recalled (N+), as well as the F1 score. Though this inspires us to increase the number of labels assigned per image, it remains ambiguous till what extent. Annotating each image with all the labels would ultimately result in the perfect recall and N+, however it would reduce the precision drastically and thus would not be practically useful. Hence it becomes important to analyse the tradeoff between precision and recall, as well as variation in N+ on increasing the number of labels assigned per image. We study these in of Figure 3.8 by increasing the number of labels assigned per image from one till the vocabulary size of a dataset. In Figure 3.8(a) we compare methods that do not involve metric learning (JEC, TagProp-SD, TagProp-σSD and 2PKNN), and in Figure 3.8(b) we compare methods that involve metric learning (TagProp-ML, TagProp-σML and 2PKNN+ML) along with 2PKNN. From the figure, we can make the following observations: (1) Generally both the precision and recall values increase up to a certain extent, and then precision starts to drop while recall continues to increase. This is expected since initially increasing the number of assigned labels also increases the number of correctly predicted labels. How-

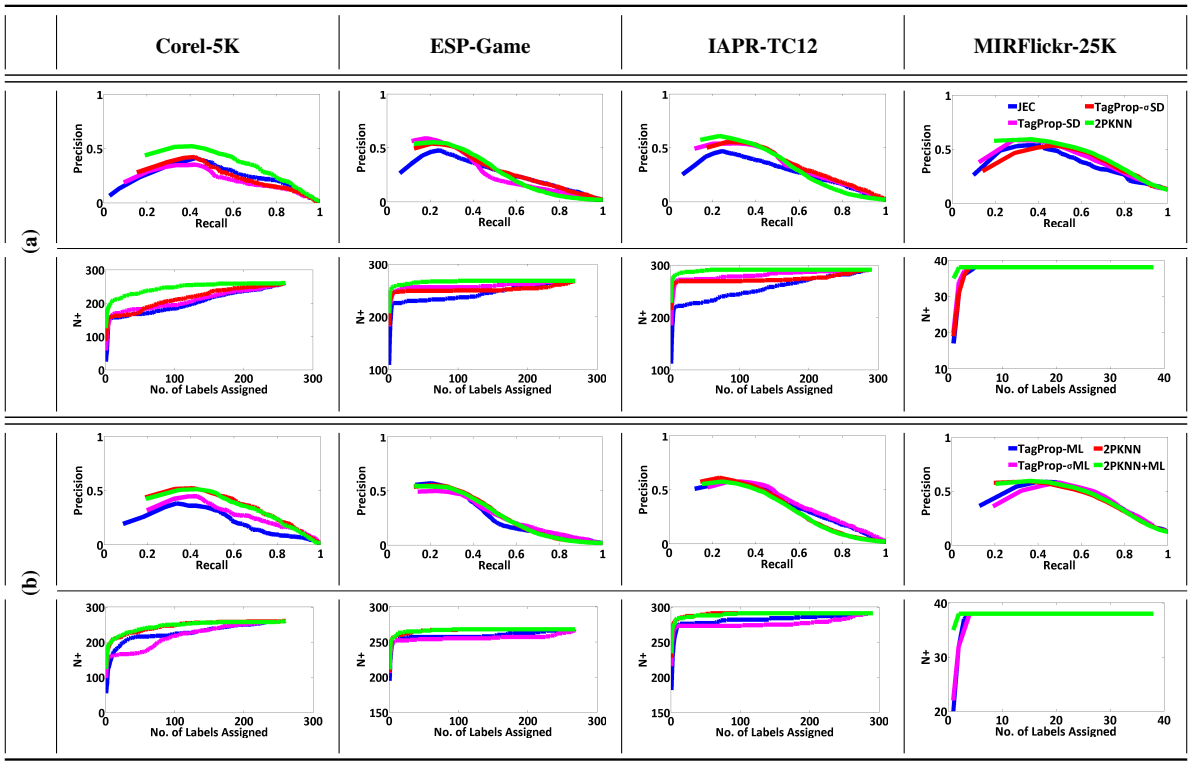


Figure 3.8 Precision-versus-recall plots by varying the number of labels assigned to an image (top row in each block), and variation in N+ on varying the number of labels assigned to an image (bottom row in each block). (Best viewed in colour.)

ever, further increasing this results into increasing the number of incorrect predictions, and thus reduces the precision score. (2) In the beginning, 2PKNN remains above the three methods (JEC, TagProp-SD and TagProp-σSD) on the Corel-5K, IAPR-TC12 and MIRFlickr-25K datasets, and comparable on the ESP-Game dataset. On the Corel-5K dataset 2PKNN remains above the three methods for a very long range of recall. On the other three datasets, as precision starts to drop, the curve of 2PKNN gradually becomes comparable to that of TagProp-σSD and TagProp-SD, and remains so for some time. Then, as recall increases further, TagProp-σSD comes above both 2PKNN and TagProp-SD on the ESP-Game and IAPR-TC12 datasets, but remains comparable on the MIRFlickr-25K dataset. Also, by then JEC catches up with the other two methods 2PKNN and TagProp-SD. We try to explain these from the frequency of labels in a dataset (column 7 of Table 3.1). For the Corel-5K dataset, the average number of images per label is far less than the other three datasets. Thus, building small semantic neighbourhoods results into better performance of 2PKNN compared to the other methods. On the other hand,

the average number of images per label for the MIRFlickr-25K dataset is quite large but the vocabulary size is too small. Due to this, the curves of all the methods become nearly comparable quite early. For the other two datasets, the average number of images per label is in between the Corel-5K and MIRFlickr-25K datasets. Thus, when the number of labels assigned per image is small, the semantic neighbourhoods of 2PKNN result into prediction of more diverse labels than the other three methods. On further increasing the number of labels assigned, the other three methods start predicting the less frequent labels, that were earlier not assigned due to low prediction weight. This trend continues on increasing the number of labels assigned even further for all the methods except TagProp-σSD. This is because it uses label-specific sigmoid functions that boost the recall of rare labels and reduces that for frequent labels. This increase in recall also increases the precision of rare labels, and thus the average precision remains higher than that of the remaining methods. This implies that at higher recall values, TagProp-σSD can provide better performance than the other methods. However, assigning too many labels would reduce the practical utility of automatic prediction. (3) In Figure 3.8(b), we observe that 2PKNN+ML performs either comparable to better than the metric learning based variants of TagProp (TagProp-ML and TagProp-σML) and 2PKNN, as also observed in Table 3.11.

The bottom rows in Figure 3.8(a) and (b) show the variation in N+ on increasing the number of labels assigned per image. Compared to all the other methods, 2PKNN and 2PKNN+ML achieve the perfect N+ much earlier on all the four datasets. This is because they use semantic neighbourhoods for label propagation, that explicit the presence of all the labels among the selected neighbours. It can also be observed that though the sigmoid variants of TagProp (TagProp-σSD and TagProp-σML) achieve better N+ than the corresponding non-sigmoid variants towards the beginning, the latter outperform the former very soon. Moreover, it takes very long for the sigmoid variants to achieve the full N+. While the performance of TagProp-SD/ML is as expected, that of TagProp-σSD/σML is possibly because of the use of per-label sigmoid functions that boost the recall of rare labels, but at the cost of reducing that for the frequent ones.

3.7.4.2 Tradeoff Between Rare and Frequent Labels

In Figure 3.9, we compare the annotation performance in terms of R and N+ on rare and frequent labels of each dataset, by assigning five labels to each test image. In Figure 3.9(a) we compare methods that do not involve metric learning (JEC, TagProp-SD, TagProp-σSD and 2PKNN), and in Figure 3.9(b) we compare methods that involve metric learning (TagProp-ML, TagProp-σML and 2PKNN+ML) along

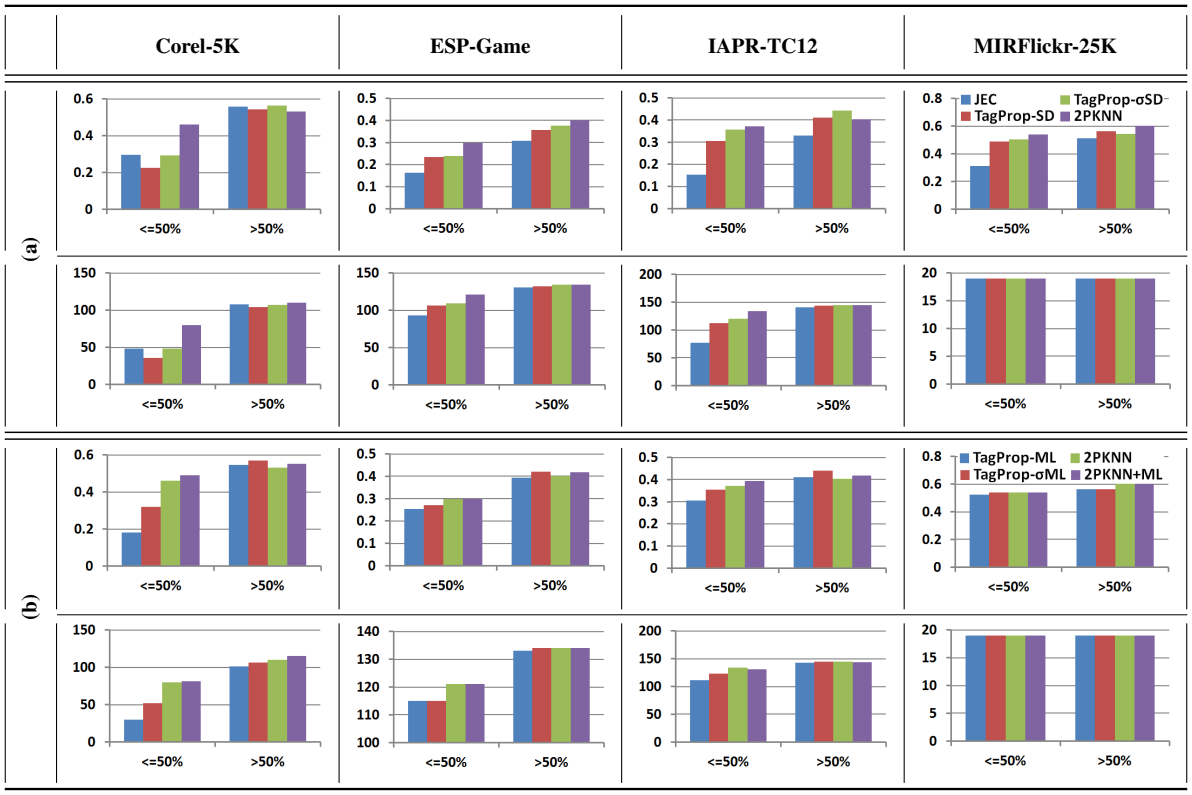


Figure 3.9 Annotation performance in terms of R (top row in each block) and N+ (bottom row in each block) for different label partitions. The labels are grouped based on their frequency in a dataset (horizontal axis). This first bin corresponds to the subset of 50% least frequent labels, and the second bin corresponds to the subset of 50% most frequent labels. (Best viewed in colour.)

with 2PKNN. Recall that TagProp-σSD and TagProp-σML variants of TagProp [35] learn a sigmoid function per label to boost the likelihood of rare labels. For comparison, the labels are partitioned into two groups based on their frequency. The first partition consists of the 50% least frequent (or rare) labels, and the second partition consists of the 50% most frequent labels.

From the figure, we can make the following observations: (1) The JEC method (which is the simplest among all) provides very competitive performance on frequent labels. However, on rare labels it usually does not perform well. Compared to JEC, TagProp-SD usually performs better on all the datasets, except the Corel-5K where its performance is slightly inferior. This is possibly because TagProp-SD considers a larger neighbourhood for label prediction than JEC. However, in the Corel-5K dataset, the frequency of rare labels is too low compared to that of frequent labels (it has the lowest and the largest

tail among all the four datasets, cf. Figure 3.7). Due to this, using larger neighbourhoods results into giving more weight to frequent labels than rare labels, and thus reduces their prediction chances. (2) TagProp-σSD and TagProp-σML mostly provide better recall than TagProp-SD and TagProp-ML, thus demonstrating the advantage of learning label-specific sigmoid functions. (3) 2PKNN performs consistently better than TagProp-σSD on rare labels. This implies that building semantic neighbourhoods before label propagation can provide better boost for rare labels than learning label-specific sigmoid functions. (4) In general, the performance of each method on frequent labels is more than that on rare labels. However, the relative difference in the performance of 2PKNN and 2PKNN+ML on the two label-partitions is mostly less than the other three methods. This indicates that 2PKNN(+ML) addresses the class-imbalance problem better than the compared methods. (5) The performance of 2PKNN and 2PKNN+ML on rare labels is mostly better than the other methods, and that on frequent labels is either better than or comparable to the other methods. This shows that 2PKNN and 2PKNN+ML do not compromise much with the performance on frequent labels while gaining that on rare labels. (6) The performance of 2PKNN+ML is always either better than or comparable to 2PKNN on both the label-partitions for all the datasets. This confirms that our metric learning approach benefits both rare as well as frequent labels. From all the above observations, we can infer that 2PKNN along with metric learning can be a better option than either JEC or TagProp for the image annotation task.

3.7.4.3 Diversity of Labels in Neighbours

Now we try to empirically analyse the diversity of labels in the neighbours identified after the first pass of 2PKNN, and compare this with the conventional kNN algorithm. For 2PKNN, we vary the number of nearest neighbours from 1 to 100 on the subset of samples obtained after the first pass. For kNN, we vary the number of nearest neighbours in the same range but on the complete training set. In Figure 3.10 (top), we calculate the number of distinct labels in the neighbours averaged over all test images. Here we can observe that using 2PKNN, we consistently get more diverse labels than kNN. In Figure 3.10 (bottom), we calculate the frequency (in the full training set) of the distinct labels in the neighbours averaged over all test images. From this, we we can see that the average frequency using 2PKNN is consistently below kNN, thus indicating that using 2PKNN we can improve the occurrence of rare labels in the neighbours, whereas the influence of frequent labels is more than rare ones in kNN. From these, we can conclude that 2PKNN can help in retrieving neighbours that contain more diverse and relatively less frequent labels than kNN.

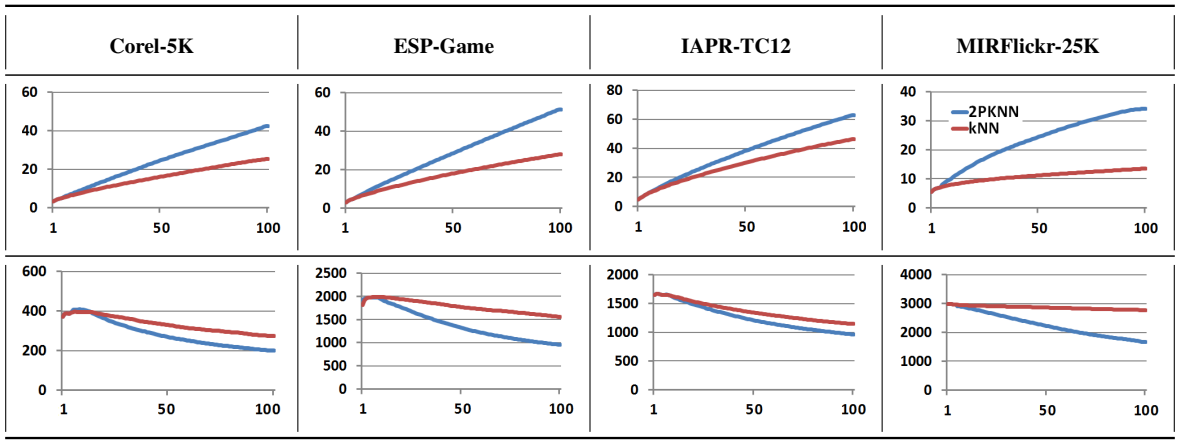


Figure 3.10 Average number of distinct labels included in the neighbours (top), and average frequencies of these labels in the training set (bottom) on increasing the number of nearest neighbours (horizontal axis) after the first pass of 2PKNN and the conventional kNN algorithm. (Best viewed in colour.)

3.7.4.4 Computational Cost

In Figure 3.11, we analyse the training and testing time of 2PKNN on varying the value of K_1 in $\{1, \ldots, 5\}$ using the same combination of features. Here, we can observe that while the increase in training time is almost linear for the Corel-5K, ESP-Game and IAPR-TC12 datasets (the datasets with large vocabularies), it is sub-linear for the MIRFlickr-25K which has a small vocabulary. Also, the increase in testing time as K_1 increases is very small, and is primarily affected by the number of test samples rather than the vocabulary size.

In Table 3.15, we compare the training and testing time of JEC, TagProp and 2PKNN. Our hardware configuration comprises Intel i7-4790K processor with 32 GB RAM. For the comparison, we consider $K = 200$ neighbours in TagProp, and $K_1 = 5$ in the first pass of 2PKNN, and use the combined TagProp, CNN and Encoding-features (T+C+E). For TagProp and 2PKNN, the training time denotes the time required for metric learning. Note that since there is no learning involved in JEC, the training time is not applicable. In all the methods, we do not include the time required for computing pairwise distances and performance evaluation, since these are the same for all. Here, we can observe that the computational cost of 2PKNN is usually higher than JEC and TagProp. This is as expected since 2PKNN is a two-step process, and additionally requires semantic groups (sample-subsets based on label

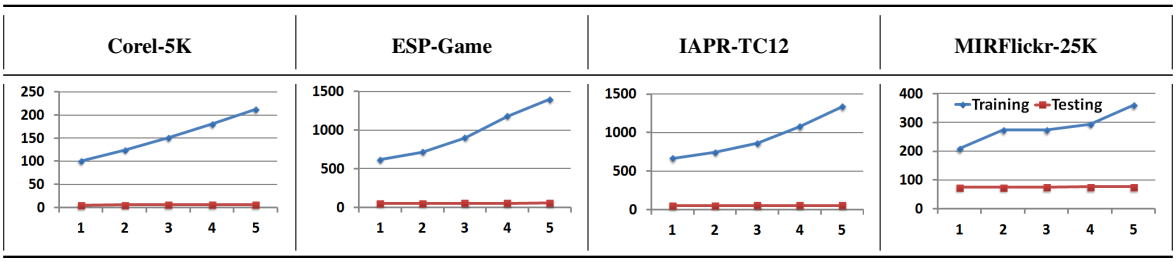


Figure 3.11 Training and testing time (in seconds) for 2PKNN+ML (vertical axis) on varying the value of K_1 (horizontal axis), using the combined TagProp, CNN and Encoding-features.

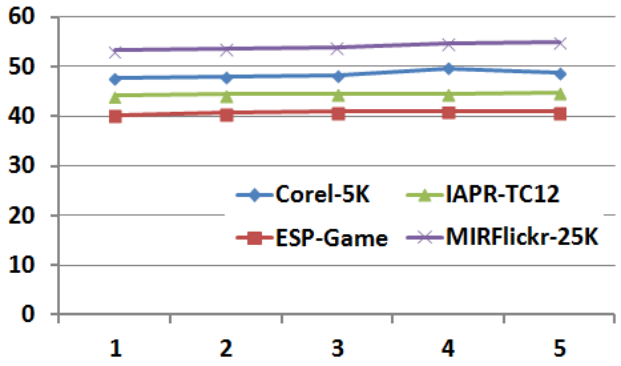


Figure 3.12 Annotation performance of 2PKNN+ML in terms of F1 (vertical axis) on varying the value of the K_1 parameter (horizontal axis).

information) while identifying neighbours. Still, for the larger (ESP-Game and IAPR-TC12) datasets, its training cost is comparable to that of TagProp.

From these, we can conclude that the training (metric learning) of 2PKNN is quite scalable, and can be done in a reasonable time (in under 30 minutes) even for tens of thousands samples and hundreds of labels. The testing time, though substantially higher than the competing nearest neighbour based methods, is also fairly feasible for all practical applications.

3.7.4.5 Performance on Varying "K_1"

Analogous to kNN algorithm, K_1 is a hyper-parameter in 2PKNN that needs to be tuned by doing cross-validation. Figure 3.12 shows the influence of K_1 on the annotation performance on all the four datasets (in terms of F1 score). We can observe that though the performance varies on changing the value of K_1, it is not very sensitive to its choice and remains fairly stable.

	Training			Testing		
	JEC	TagProp	2PKNN	JEC	TagProp	2PKNN
Corel-5K	NA	51.9	212.4	0.2	0.3	6.0
ESP-Game	NA	1308.5	1397.2	0.8	2.5	53.8
IAPR-TC12	NA	1367.8	1336.6	0.7	2.3	54.4
MIRFlickr-25K	NA	609.3	360.4	2.9	15.1	76.7

Table 3.15 Computational time of different methods using the combined TagProp, CNN and Encoding-features (in seconds). For TagProp and 2PKNN, the training time denotes the time required for metric learning. For all the methods, the training and testing times do not take into consideration the time required for computing pair-wise distances. Since there is no learning involved in JEC, the training time is not applicable.

3.7.4.6 Qualitative Analysis

Though it is difficult to measure incomplete-labelling, we try to analyse it from qualitative results. Figure 3.13 shows a few examples of incompletely-labelled images from the Corel-5K, ESP-Game and IAPR-TC12 datasets, along with the top five labels predicted using 2PKNN+ML. It can be observed that for all these images, our method predicts all the ground-truth labels. Moreover, the additional labels predicted are actually depicted in the corresponding images, but missing in their ground-truth annotations. These results qualitatively demonstrate that our method is capable of addressing the incomplete-labelling issue prevalent in the challenging real-world datasets[8]

In Figure 3.14, we present some failure analysis of the annotation results from the Corel-5K dataset. These are some of the examples where most of the predicted labels do not match with the ground-truth ones. For the first two examples (from the left), the method seems to fail completely (except for the label "sky" in the first image). For the last four examples, some of the incorrectly predicted labels would seem justifiable if we carefully examine the corresponding images (i.e., their local structures and overall context). As discussed towards the beginning of this chapter, such failures can be attributed to the lack of good feature representations (even with the recent advancements in deep learning based techniques) that makes the problem of semantic labelling quite challenging.

[8]We also examined a few labels manually and observed consistent improvements in their F1 scores.

Corel-5K		ESP-Game		IAPR-TC12	
					
GT: water, people, pool, swimmers	**GT:** water, beach, boats, sand	**GT:** light, tower	**GT:** red, woman	**GT:** lake	**GT:** monument, stone
Pred: **people**, **pool**, **swimmers**, **water**, *athlete*	**Pred:** **beach**, **sand**, **water**, *sky*, **boats**	**Pred:** *night*, **tower**, **light**, *sky*, *city*	**Pred:** *hair*, *blonde*, *girl*, **woman**, **red**	**Pred:** *middle*, **lake**, *landscape*, *mountain*, *desert*	**Pred:** *bird*, **monument**, **stone**, *tree*, *bush*

Figure 3.13 Example images from the Corel-5K, ESP-Game and IAPR-TC12 datasets, along with the ground-truth set of labels (GT) and the top five labels predicted using 2PKNN+ML (Pred). The labels in **blue** (bold) are those that match with the ground-truth, while those in *red* (italics) are missing in the ground-truth though depicted in the corresponding images.

3.7.5 Discussion on SVM-VT

In this subsection, we analyse/discuss some aspects of the proposed SVM-VT method, such as qualitatively analyzing the tolerance parameter, conceptural comparison with other similar methods, and some practical advantages.

3.7.5.1 Qualitative Analysis of SVM-VT

In Figure 3.15, we show some negative samples (i.e., samples that are not annotated with a particular label) along with the computed t-scores for two labels each from Corel-5K, ESP-Game and IAPR TC-12 datasets. We can make several interesting observations from these examples. All these negative samples actually look semantically related (near positive) with the corresponding labels. In first row, the negative samples for the label "clouds" demonstrate examples of incomplete-labelling, as "clouds" are clearly visible in these images but missing in their ground-truth. Similar is the case with the negative samples for the labels "teeth", "toy", "horse", and "bedcover". The first negative sample for the label "man" is an example of structural-overlap, because though "man" is not there in this image, it has "woman" that is structurally related with "man". The second negative sample for the same label is an example of






GT: sky, jet, plane	**GT:** flowers, petals	**GT:** people, town, street, horses	**GT:** sky	**GT:** mountain, sky, clouds, palace	**GT:** ruins, stone, temple, sculpture
Pred: coast, truck, hills, slope, sky	**Pred:** athlete, close-up, tree, water, trunk	**Pred:** cow, cave, people, vehicle, sky	**Pred:** decoration, vehicle, palace, basket, buddha	**Pred:** sky, locomotive, railroad, train, tree	**Pred:** plaza, architecture, trunk, vendor, porcupine

Figure 3.14 Example images from the Corel-5K dataset, along with the ground-truth set of labels (GT) and the top five labels predicted using 2PKNN+ML (Pred). These are some of the failure cases where most of the predicted labels do not match with the ground-truth ones.

label-ambiguity since in this image "people" is used to refer the "man" climbing the tree. Thus, these examples verify the ability of our method in identifying the so-called confusing samples.

3.7.5.2 Conceptual Comparison with Other Methods

Several extensions of SVM have been proposed in the past that try to modify the loss-function. Here we give a brief overview of some of these methods whose formulation looks similar to that of SVM-VT, and discuss how SVM-VT differs from them. In [93], a separate scaling parameter is used for hinge-loss of positive and negative classes. This is generalized in [155] where hinge-loss corresponding to each sample is scaled individually by a parameter in the range $[0, 1]$. In [126], the loss is made sensitive to the distance of a sample from class-centroid. One similarity among all these methods is that they try to learn a classifier that is robust against outliers, by looking only at the features of samples. SVM-VT differs from these models in at least two ways. First, while these methods modify either the margin constraint [126] or the (classifier) update-rule [93, 155] of the conventional SVM, the proposed hinge-loss of SVM-VT modifies both of these simultaneously. Second, all these methods consider only the distribution of samples in feature space, whereas the hinge-loss of SVM-VT has an associated semantic meaning that relates samples and labels based on semantic properties in addition to visual features.

For the image annotation task, Structured SVM (or SSVM) [132] seems to be an attractive option. Intuitively, the idea behind SSVM is to benefit from the structure in the output space. Through SVM-VT

Figure 3.15 For example labels (in blue) from the Corel-5K, ESP-Game and IAPR TC-12 datasets, examples of "negative" samples along with their t-scores and corresponding ground-truth labels (for a given label, smaller t-score of a negative sample implies higher semantic relevance with that label and vice-versa).

we have tried to infuse this idea in the SVM model, though indirectly. This is because while learning the classifier for a given label, the amount of penalty for each non-positive sample differs depending on how much confusion it introduces while training a classifier. A sample that is more confusing for a given label adds a smaller penalty as compared to others. This way, each negative becomes a negative in its own way as in SSVM. However, the time-complexity and/or memory requirements during training of SSVM-based models increase significantly as we move to large datasets with large vocabularies. This usually makes it difficult to scale such models for the practical scenarios of large-scale learning. SVM-VT provides the flexibility of both introducing semantics in classifier-training as well as efficient optimization comparable to binary SVM.

3.7.5.3 Some Practical Advantages

The SVM-VT model, despite its simplicity, offers some advantages over the existing discriminative and NN-based methods for image annotation which we discuss below:

1. Scalability: Most of the discriminative methods for multi-label problems such as [12, 41] learn model(s) for all the labels in a vocabulary jointly in a single optimization problem. Though this provides the advantage of incorporating inter-label relationships, it is sometimes difficult to scale such methods to very large vocabularies. In contrast, SVM-VT provides a framework for learning a model for each label in an independent manner, and at the same time takes care of inter-label relationships as well. Given an efficient way of computing the tolerance parameter, this can be scaled to very large vocabularies similar to SVM.

2. Time-complexity: Once we have computed the tolerance parameter, time-complexity of SVM-VT is almost comparable to that of SVM. Since each classifier can be learned independent of others, practically it is possible to learn all them simultaneously. Once we have learned all the classifiers, predicting labels for a new image becomes several times faster than the NN-based models.

3. Performance: As discussed before, the performance of SVM has remained (almost) unexplored in the task of image annotation on standard datasets. We demonstrated that simple SVM itself achieves superior performance than several existing methods. Moreover, SVM-VT demonstrates that it is possible to achieve further improvements in performance by relaxing the strict discriminative behaviour of the SVM classifier. To the best of our knowledge, this is the first study where

Table 3.16 Annotation performance using individual annotation models, and that after fusing them.

Dataset →	Corel-5K				ESP-Game				IAPR-TC12				MIRFlickr-25K			
Method ↓	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**	**P**	**R**	**F1**	**N+**
SVM-VT	31	53	39.1	195	34	40	36.8	**267**	40	41	40.5	**285**	44	58	50.0	**38**
2PKNN	48	50	49.0	190	49	35	40.8	255	53	39	44.9	279	51	57	53.8	**38**
2PKNN+ML	48	52	**49.9**	**196**	48	36	**41.1**	255	50	41	**45.1**	275	53	57	**54.9**	**38**
2PKNN ⊕ SVM-VT	46	50	47.9	191	42	37	**39.3**	**256**	48	41	**44.2**	**280**	45	59	**51.1**	**38**
2PKNN+ML ⊕ SVM-VT	45	54	**49.1**	**199**	41	37	38.9	255	45	43	44.0	276	45	59	**51.1**	**38**

a discriminative one-vs-rest type of model has been shown to give promising results on image annotation task with large vocabularies.

Along with the SVM-VT model, we have also proposed a method for determining semantic relationships of negative samples with a given label based on visual similarity and dataset statistics. As vocabulary size grows, such relationships start getting prominent. However, due to limitations of human annotations, these give rise to the three issues discussed before. We show that using our method, we are able to find such relationships efficiently.

3.8 Comparing 2PKNN and SVM-VT

From Table 3.11, we can observe that 2PKNN consistently performs better than SVM-VT in terms of F1 score on all the four datasets. This difference is largest on the Corel-5K dataset with around 28% of improvement. However, in terms of N+, SVM-VT is much better than 2PKNN specially on the ESP Game and IAPR TC-12 datasets. Recall that the vocabulary size of these datasets is 268 and 291 (Table 3.1), out of which SVM-VT correctly recalls at least one instance of 267 and 285 labels respectively. This suggests that the tolerance parameter in SVM-VT does help in learning a model that is tolerant against "confusing" samples. However, there is still a lot of score for improvement on the Corel-5K dataset, for which the best performing method (2PKNN+ML) achieves an N+ score of 196 whereas the vocabulary size is 260.

	Training				Testing			
	SVM	SVM-VT	2PKNN	2PKNN+ML	SVM	SVM-VT	2PKNN	2PKNN+ML
Corel-5K	51.9	96.5	NA	212.4	0.02	0.02	5.7	6.0
ESP-Game	220.8	554.6	NA	1397.2	0.07	0.07	53.3	53.8
IAPR-TC12	240.9	555.4	NA	1336.6	0.07	0.07	53.8	54.4
MIRFlickr-25K	10.2	97.0	NA	360.4	0.06	0.06	76.2	76.7

Table 3.17 Computational time of SVM-VT, 2PKNN and 2PKNN+ML with the features learned using KCCA embedding. For all the methods, the training and testing times do not take into consideration the time required for computing pair-wise distances.

Now, we try to merge these two different class of models. For this, we take average of the prediction scores using both 2PKNN(+ML) and SVM-VT. Table 3.16 shows the annotation performance using individual models as well as their fusion. We observe that the fused models achieve better F1 scores than SVM-VT, however these are worse than 2PKNN and 2PKNN+ML. On all the three datasets with large vocabularies, the fused models are able to achieve better N+ scores than 2PKNN/2PKNN+ML. Interestingly, on the Corel-5K dataset, we get an N+ score of 199 which is better than individual models, suggesting a mild complementary behaviour. In general, we can say that the fused models can be used to manage the trade-off between F1 and N+ scores.

Finally, we compare the training and testing time in Table 3.17 with KCCA embedding learning using the combined TagProp, CNN and Encoding-features. Here, we do not consider the time required for computing pair-wise distances for the training pairs and that for learning the KCCA embedding, since this are done off-line and are common for all except the former in case of SVM. We also do not consider the time required for performance evaluation during the testing phase, which is again the same for all. The training time for SVM-VT is more than SVM since it additionally involves computation of the t-scores for each sample per label. The testing times for both 2PKNN and 2PKNN+ML are almost equivalent since 2PKNN+ML just requires and additional step of multiplying distances with the learned metric. Also, the testing time of both SVM and SVM-VT is equivalent since the testing procedure is the same for both. As discussed in Section 3.7.5.3, the testing time of SVM-VT is far less than 2PKNN(+ML). Also, in practice nearest neighbour based methods require to keep the full training data

into memory during the testing phase. Thus, SVM/SVM-VT can be preferred when there are constraints on run-time and memory, and 2PKNN(+ML) can be preferred when performance is critical.

3.9 Summary

We have introduced two new methods focusing on the image annotation task: 2PKNN and SVM-VT. We have shown that the 2PKNN method along with metric learning achieves either comparable or state-of-the-art results on the challenging image annotation datasets, where class-imbalance is a serious concern and frequencies of labels follow the long-tail phenomenon. We also showed that the SVM-VT model can be useful in handling incomplete-labelling, label-ambiguity and structural-overlap that are frequently encountered in large vocabulary image annotation datasets. Despite their simplicity, they perform either comparable to or better than existing methods.

Our methods are generic and can find applications in a wide variety of classification as well as multi-label tasks. We believe our work will provide a new platform for evaluating and comparing the future techniques in this domain.

Chapter 4

Image Captioning and Caption-based Image Retrieval Using Textual Phrases

4.1 Introduction

In the previous chapter, we focused on describing an image using a set of keywords. However, one limitation of keyword-based annotation is that it does not convey complete meaning. E.g., an image tagged with $\{black, car, dog\}$ does not convey whether it has a "black car" and a "dog", or a "car" and a "black dog", and what are their state(s) and relative positions. Whereas, the sentence "a dog is sitting on a black car" implicitly encodes such relationships.

Image captions/descriptions[1] not only contain information about the different objects present in an image, but also tell us about their states and spatial relationships. In many cases, this information can be easily obtained from available data, hence leveraging the gap between visual perception and semantic grounding. This in turn can be useful in both producing captions for images without any textual meta-data, as well as retrieving semantically relevant images using textual queries.

4.1.1 Related Work

There have been several attempts in the last few years that try to describe images using short captions, such as [26,27,62,63,66,68,69,82,101,109,125,134,146,157]. Most of the initial approaches first try to predict the visual content of an image using some off-the-shelf computer vision techniques, such as pre-trained object detectors and/or scene classifiers [68, 157], or their integration with feature-based image similarity [69, 101]). This information is then fused using some Natural Language Generation (NLG)

[1] We will use the terms "caption" and "description" interchangeably.

technique to construct image descriptions. All these papers show that though the idea of automatically *generating* captions can provide a large set of possible descriptions, most of them usually fail to match descriptions generated or provided by humans. A few other papers [27, 101] try to partly address this by directly transferring existing (human-written) captions to new images, by matching a query image with annotated (training) images.

A complementary task to describing an image is that of image retrieval given a query. Unlike image captioning, the problem of image retrieval is a well-studied topic in computer vision and multimedia [18]. Two popular streams in this field are: (a) content-based retrieval of images (e.g., retrieve images similar to a given query image [4, 107]), and (b) directly retrieving images based on textual category [50, 70]. In the first setting, the role of semantics of the query image is minimal, and retrieval is usually performed using a standard set of visual features (e.g., GIST [100], bag-of-words histogram using SIFT [78], CNN [5], etc.). In the second setting, semantics get introduced in the form of textual labels. This complicates the task since now it is possible to have two visually dissimilar images depicting the same concept. It gets further complicated as more linguistic aspects get introduced into the query. E.g., to perform image retrieval for queries that are bigger than a single category label (say a pair of labels) is more complicated than a single-label query. This now requires us to look for images containing all the concepts rather than just one or a few of them. One possibility is to retrieve images that are tagged with all the query concepts. However, analogous to the image captioning problem, such a setting fails to capture the relationships among the query labels. E.g., the query "person, car" gives no clue about the relative position of "person" and "car". This gives rise to several possibilities, such as a "person" can be sitting inside a "car", standing near a "car", getting hit by a "car", etc. This limitation was partly addressed in [117] which proposed the notion of "visual phrases" and demonstrated that it is semantically more meaningful to learn complete phrases in the visual domain (e.g. "person sitting in car") rather than individual labels ("person" and "car").

Lately, with most of the new approaches focusing on deep neural network based models, there have been significant improvements in both image captioning as well as caption-based image retrieval tasks [26, 62, 63, 66, 82, 109, 125, 146]. Quantitatively, such methods have been shown to achieve much better performance compared to older techniques, thanks to the high capacity of such models and emergence of large datasets. These can be broadly categorized into two categories. The first approach takes the activations from the last hidden layer of an object detection convolutional neural network (CNN) model and feeds them into a recurrent neural network (RNN) language model, also referred to as a multi-modal

RNN (or MRNN) [62, 66, 82]. The second approach is based on first predicting a set of keywords using a CNN model that are likely to depict the visual content, and then using a maximum entropy language model over the predicted words for caption generation [26]. However, such methods have mostly been shown to work with small textual queries/captions, and dealing with detailed captions is still an emerging area with a very few attempts like [76]. Also, as discussed in [20], another limitation of state-of-the-art caption generation methods like above is that they reproduce generic captions from training data quite often, and do not perform well on images that are compositionally very different from previously seen images. To work towards this, a large-scale dataset with region-to-phrase correspondence for image descriptions was introduced in [109]. Such an explicit correspondence is expected to provide better supervision that would help in developing richer models for a variety of image-text compositions.

4.1.2 Our Motivation

The problem of dealing with images and captions was a relatively new area at the time we initially looked into it [39]. At that time, most of the earlier attempts for generating descriptions for (unseen) images such as [27,68,72,101,157] relied mostly on a few object detectors, classifiers and data statistics, and did not utilize the semantic information encoded in available descriptions of annotated (training) images. These descriptions were used either to restrict the set of objects/prepositions/verbs [68,72,157], or pick one or more complete sentences and transfer them to a test image unaltered [27, 101]. Motivated by the concept of visual phrases [117], we proposed a novel approach that works on syntactically and linguistically motivated phrases (e.g., "person on road", "yellow car", etc.) automatically extracted from the available descriptions. These carry a bigger and more meaningful chunk of information, compared to predicting individual bits (such as objects, attributes, verb, preposition, etc.) in a piece-wise manner and then combining them at a later stage as done by the earlier methods. This facilitates the use of such phrases for both describing new images using captions and performing caption-based image retrieval on unannotated images.

4.1.3 Organization

This chapter is organized as follows. In the next section, we describe how we define and extract phrases from a given description/sentence. Then we present our approach for predicting relevance between a phrase and an image in Section 4.3. In Section 4.4 and Section 4.5, we describe how we use the phrase-image relevance prediction model for image captioning and caption-based image retrieval

tasks respectively. In Section 4.6, we conduct experimental analysis. Finally we provide a summary and some directions for future work in Section 4.7.

4.2 Phrase Extraction

As discussed above, the key component of our approach is to effectively use the information in the ground-truth descriptions, and for that we need to extract relation tuples from text automatically. For this purpose, we process the descriptions using the Stanford CoreNLP toolkit [81][2]. We use "collapsed-ccprocessed-dependencies" which is intended to be more useful for relation extraction task [19], as dependencies involving prepositions and conjuncts are collapsed to reflect direct relation between content words.

We look at an image as a collection of phrases in the visual domain, and hypothesize that similar appearing images share identical phrases. We extract syntactic phrases from human-generated image descriptions, and map each sentence to a list of phrases like $(subject, verb)$, $(object, verb)$, $(verb, prep, object)$, etc. The approaches for image captioning before ours obtain relations of the form $(object, action, scene)$ [27], $(object1, object2, verb, scene, prep)$ [157], or $((attribute1, object1), prep, (attribute2, object2))$ [68] by combining the outputs of individual detectors with some heuristic and/or corpus statistics to predict the involved action/preposition(s). However, such predictions can be quite noisy (e.g., $(person, under, road)$ [68]), resulting in absurd sentences. In contrast, phrases implicitly encode ordering preference information, and hence allow semantically meaningful caption generation as well as retrieval.

In practice, we extract 9 distinct types of phrases from human-generated descriptions: $(subject)$, $(object)$, $(subject, verb)$, $(object, verb)$, $(subject, prep, object)$, $(object, prep, object)$, $(attribute, subject)$, $(attribute, object)$, and $(verb, prep, object)$. Each noun (subject/object) is expanded up to at most 3 hyponym levels using its corresponding WordNet synsets. To explore the possibilities of generating varied and interesting descriptions, we also consider scenarios without considering the synonyms. Table 4.1 shows a sample sentence and its phrases extracted using our approach (considering both with and without synonyms). Further details with worked-out examples on phrase extraction can be found in [37].

[2] `http://nlp.stanford.edu/software/corenlp.shtml`

Sentence	Phrases without synonym	Phrases with synonym
Blue and green hummingbird sitting on a branch.	$\left(\textbf{hummingbird}_{(s)}\right)$, $\left(\textbf{green}_{(a)}, \textbf{hummingbird}_{(s)}\right)$, $\left(\textbf{blue}_{(a)}, \textbf{hummingbird}_{(s)}\right)$, $\left(\textbf{hummingbird}_{(s)}, \textbf{sit}_{(v)}\right)$, $\left(\textbf{sit}_{(v)}, \textbf{on}_{(p)}, \textbf{branch}_{(o)}\right)$, $\left(\textbf{branch}_{(o)}\right)$	$\left(\textbf{bird}_{(s)}\right)$, $\left(\textbf{green}_{(a)}, \textbf{bird}_{(s)}\right)$, $\left(\textbf{blue}_{(a)}, \textbf{bird}_{(s)}\right)$, $\left(\textbf{bird}_{(s)}, \textbf{sit}_{(v)}\right)$, $\left(\textbf{sit}_{(v)}, \textbf{on}_{(p)}, \textbf{branch}_{(o)}\right)$, $\left(\textbf{branch}_{(o)}\right)$

Table 4.1 Example sentence with automatically extracted phrases. In "Phrases with synonym", "hummingbird" is replaced by its synonym "bird" (determined using WordNet). Here, s→subject, a→attribute, o→object, v→verb and p→preposition.

4.3 Phrase Relevance Prediction Model (PRPM)

Given a dataset of images with their corresponding descriptions, a set of phrases $\mathcal{Y}$ is extracted from these descriptions. These phrases are restricted to five different types (considering "subject" and "object" as equivalent for practical purposes): $(object)$, $(attribute, object)$, $(object, verb)$, $(verb, prep, object)$, and $(object, prep, object)$.

Now, the dataset takes the form $\mathcal{T} = \{(I_i, Y_i)\}$ where I_i is an image and $Y_i \subseteq \mathcal{Y}$ is its corresponding set of phrases. Each image I is represented using a set of n features $\{f_{1,I}, \ldots, f_{n,I}\}$. Given any two images I and J, the distance between them is computed using a weighted linear combination of the base distances corresponding to each feature (e.g., L_1 for colour histograms, L_2 for GIST, etc.):

$$D_{I,J} = w_1 d_{1I,J} + \ldots + w_n d_{nI,J} = \mathbf{w} \cdot \mathbf{d}_{I,J}, \tag{4.1}$$

where $w_i \geq 0$ denotes the weight assigned to the base distance $d_{iI,J}$ corresponding to i^{th} feature. Using this distance function, for a new image I, its K most similar images $\mathcal{T}_I^K \subseteq \mathcal{T}$ are picked fromt the dataset. Then, the joint probability of associating a phrase $y_i \in \mathcal{Y}$ with I is given according to [53]:

$$P(y_i, I) = \sum_{J \in \mathcal{T}_I^K} P_{\mathcal{T}}(J) P_{\mathcal{F}}(I|J) P_{\mathcal{Y}}(y_i|J). \tag{4.2}$$

Here, $P_{\mathcal{T}}(J) = 1/K$ denotes the uniform probability of picking some image J from $\mathcal{T}_I^K$. $P_{\mathcal{F}}(I|J)$ denotes the likelihood of image I given J, defined as:

$$P_{\mathcal{F}}(I|J) = \frac{exp(-D_{I,J})}{\sum_{J' \in \mathcal{T}_I^K} exp(-D_{I,J'})}. \tag{4.3}$$

Finally, $P_{\mathcal{Y}}(y_i|J)$ denotes the probability of seeing the phrase y_i given image J, and is defined according to [28]:

$$P_{\mathcal{Y}}(y_i|J) = \frac{\mu_i \delta_{y_i,J} + N_i}{\mu_i + N} = \frac{\nu_i \delta_{y_i,J} + (N_i/N)}{\nu_i + 1} \tag{4.4}$$

Here, if $y_i \in Y_J$, then $\delta_{y_i,J} = 1$ and ϵ otherwise (ϵ being a small positive real number). N_i is the approximate Google count of the phrase y_i, N denotes the sum of Google counts of all phrases in $\mathcal{Y}$ that are of the same type as that of y_i, and $\nu_i \geq 0$ is the smoothing parameter. In practice we consider ν_i to be the same for all the phrases of the same type.

In data-driven learning techniques, a text corpus is usually employed to estimate the statistical behaviour of different n-grams. In our case, the number and diversity of phrases is quite large, and it is unlikely to predict their general behaviour using only the available descriptions. To address this, we smooth their relative frequencies using the number of approximate search results reported by Google for an exact match query on each phrase similar to [68]. Some of the phrases (along with their types) and their counts obtained using Google are: "aircraft fly" (*subject,verb*): 360,000; "horse with rider" (*subject,prep,object*): 496,000; "golden bridge" (*attribute,object*): 2,570,000; "drive on street" (*verb,prep,object*): 155,000.

In order to learn the two sets of parameters (i.e., the weights w_i's and smoothing parameters μ_i's), a loss function analogous to [152] is used. Given an image J along with its true phrases Y_J, the goal is to learn the parameters such that (i) the probability of predicting the phrases in $\mathcal{Y} \backslash Y_J$ should be minimized, and (ii) the probability of predicting each phrase in Y_J should be more than any other phrase. Precisely, the loss function is given by:

$$e = \sum_{J,y_k} P(y_k, J) + \lambda \sum_{(J,y_k,y_j)\in\mathcal{M}} (P(y_k, J) - P(y_j, J)). \tag{4.5}$$

Here, $y_j \in Y_J$, $y_k \in \mathcal{Y} \setminus Y_J$, $\mathcal{M}$ is the set of triples that violate the second constraint stated above, and $\lambda > 0$ is used to manage the trade-off betweent the two terms. This is optimized using a stochastic gradient descent method, by learning w_is and ν_is in an alternate manner.

4.3.1 Integrating Semantics in Phrase Relevance Prediction Model

One limitation of PRPM is that it treats phrases in a binary manner; i.e., in equation 4.4, $\delta_{y_i,J}$ is either 1 or ϵ depending on the presence or absence of the phrase y_i in Y_J respectively. This results in penalizing semantically similar phrases (e.g., "kid walk" versus "child run"). To address this, now we extend this

model by considering semantic similarities among phrases during phrase relevance prediction. To begin with, first we discuss how we compute semantic similarity between two phrases.

Let a_1 and a_2 be two words (e.g., "boy" and "man"). We use WordNet based JCN similarity measure [54] to compute semantic similarity between the words a_1 and a_2. WordNet is a large lexical database of English where words are inter-linked in a hierarchy based on their semantic and lexical relationships. Given a pair of words (a_1, a_2), the JCN similarity measure returns a score $s_{a_1a_2}$ in the range $[0, \inf)$, with higher score corresponding to more similarity and vice-versa. This similarity score is then mapped into the range $[0, 1]$ using the following non-linear transformation as described in [71] (denoting $s_{a_1a_2}$ by s in short):

$$\gamma(s) = \begin{cases} 1 & s \geq 0.1 \\ 0.6 - 0.4\sin(\frac{25\pi}{2}s + \frac{3}{4}\pi) & s \in [0.06, 0.1] \\ 0.6 - 0.6\sin(\frac{\pi}{2}(1 - \frac{1}{3.471s+0.653})) & s \leq 0.06 \end{cases}$$

Using this, we define a similarity function that takes two words as input and returns the semantic similarity score between them computed using the above equation as

$$W_{sim}(a_1, a_2) = \gamma(s_{a_1a_2}) \tag{4.6}$$

From this, we can also compute semantic dissimilarity score between two words as

$$\overline{W}_{sim}(a_1, a_2) = 1 - W_{sim}(a_1, a_2) \tag{4.7}$$

Based on equation 4.6, we define semantic similarity between two phrases (of the same type) as V_{sim}, which is an average of the similarity between each of their corresponding constituting terms. E.g., if we have two phrases v_1=("person", "walk") and v_2=("boy", "run") of the type $(object, verb)$, then their semantic similarity score will be given by $V_{sim}(v_1, v_2) = 0.5 \times (W_{sim}(\text{"person","boy"}) + W_{sim}(\text{"walk","run"}))$. It should be noted that we cannot compute semantic similarity between two prepositions using WordNet. So, while computing semantic similarity between two phrases that contain prepositions in them (i.e., of type $(verb, prep, object)$ or $(object, prep, object)$), we omit the prepositions. Analogous to equation 4.7, we can compute semantic dissimilarity score between two phrases as $\overline{V}_{sim}(v_1, v_2) = 1 - V_{sim}(v_1, v_2)$.

Finally, given a phrase y_i and a set of phrases Y of the same type as that of y_i, we define the semantic similarity between them as

$$U_{sim}(y_i, Y) = \max_{y_j \in Y} V_{sim}(y_i, y_j). \tag{4.8}$$

In practice, if $|Y| = 0$ then we set $U_{sim}(y_i, Y) = \epsilon$ (a small positive real number).

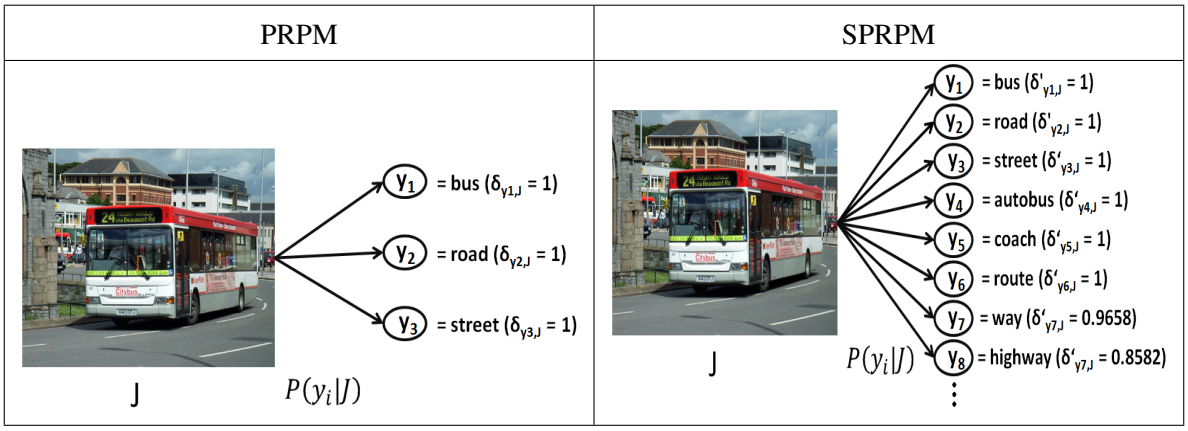


Figure 4.1 An illustration of the difference between PRPM and SPRPM. In PRPM, the conditional probability of a phrase y_i given an image J depends on whether that phrase is present in the ground of the phrases of J (i.e., Y_J) or not. In case the phrase is not present, the corresponding $\delta_{y_i,J}$ (equation 4.4) becomes ϵ without considering the semantic similarity of y_i with other phrases in Y_J. This limitation of PRPM is addressed in SPRPM by finding the phrase in Y_J that is semantically most similar to y_i and using their similarity score instead of ϵ. In the above example, we have Y_J ={"bus", "road", "street"}. Given a phrase y_i = "highway", $\delta_{y_i,J} = \epsilon$ according to PRPM. whereas $\delta'_{y_i,J} = 0.8582$ according to SPRPM (equation 4.9) by considering the similarity of "highway" with "road" (i.e., V_{sim}("highway", "road") $= 0.8582$).

4.3.2 Semantic Phrase Relevance Prediction Model (SPRPM)

In order to benefit from the semantic similarity between two phrases while predicting the presence of some phrase y_i given a set of phrases Y_J of image J, we need to modify equation 4.4 accordingly. Let the phrase y_i be of type t, and the set of phrases of type t in Y_J be $Y_J^t \subseteq Y_J$. Then, we re-define $P_{\mathcal{Y}}(y_i|J)$ as:

$$P_{\mathcal{Y}}(y_i|J) \propto \frac{\nu_i \delta'_{y_i,J} + (N_i/N)}{\nu_i + 1}, \tag{4.9}$$

where $\delta'_{y_i,J} = U_{sim}(y_i, Y_J^t) \in [0,1]$. This means that when the phrase y_i is not present in Y_J^t, we look for that phrase in Y_J^t that is semantically most similar to y_i and use their similarity score. Such a definition allows us to take into account the semantic relationship among phrases while predicting the relevance of a phrase. In other words, it provides a view where the presence of a phrase given an image is a continuous phenomenon rather than a binary one as in equation 4.4.

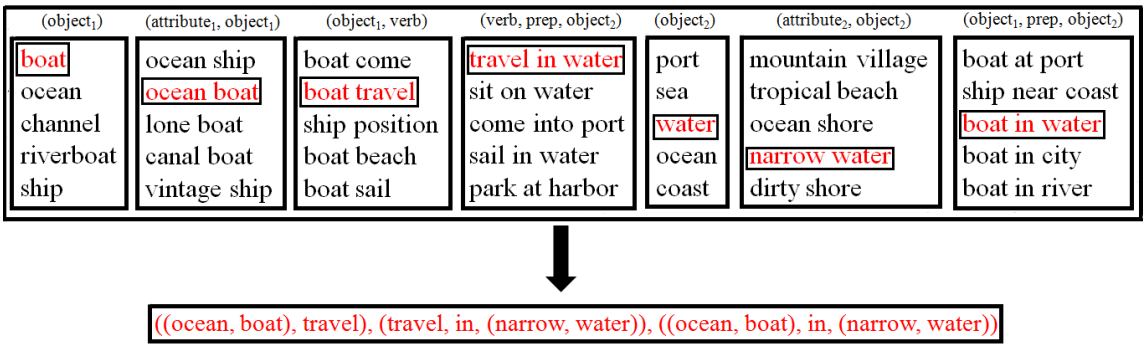


Figure 4.2 An illustration of the phrase integration algorithm for forming a triple, which is an intermediate step of the image captioning pipeline.

Since we have modified the conditional probability model for predicting a phrase given an image, we also need to change the loss function of equation 4.5 accordingly. Given an image J along with its true phrases y_j's in Y_J, now we additionally need to ensure that the penalty imposed for a higher relevance score of some phrase $y_k \in \mathcal{Y} \backslash Y_J$ than any phrase $y_j \in Y_J$ should also depend on the semantic similarity between y_j and y_k. Precisely, we re-define the loss function as:

$$e = \sum_{J, y_k} P(y_k, J) + \lambda \sum_{(J, y_k, y_j) \in \mathcal{M}} \Delta(J, y_k, y_j), \tag{4.10}$$

$$\Delta(J, y_k, y_j) = \overline{V}_{sim}(y_k, y_j)(P(y_k, J) - P(y_j, J)). \tag{4.11}$$

The implication of $\Delta(\cdot)$ is that if two phrases y_j and y_k are semantically similar (e.g., "kid" and "child"), then penalty for predicting y_k instead of y_j should be small and vice-versa. Figure 4.1 illustrates the conceptual difference between PRPM and SPRPM.

4.4 Image Captioning Using Textual Phrases

Using Eq. 4.2, we get relevance scores for all the phrases in $\mathcal{Y}$ for a given test image I. Based on these scores, the phrases of each type are sorted separately. We then integrate these phrases to get a ranked list of *triples* of the form $t = \{\ ((attribute_1, object_1), verb), (verb, prep, (attribute_2, object_2)), (object_1, prep, object_2\ )\ \}$ using an inference method similar to the Viterbi algorithm (with $object_1$ denoting the $subject$). (Figure 4.2). The score of each triple is

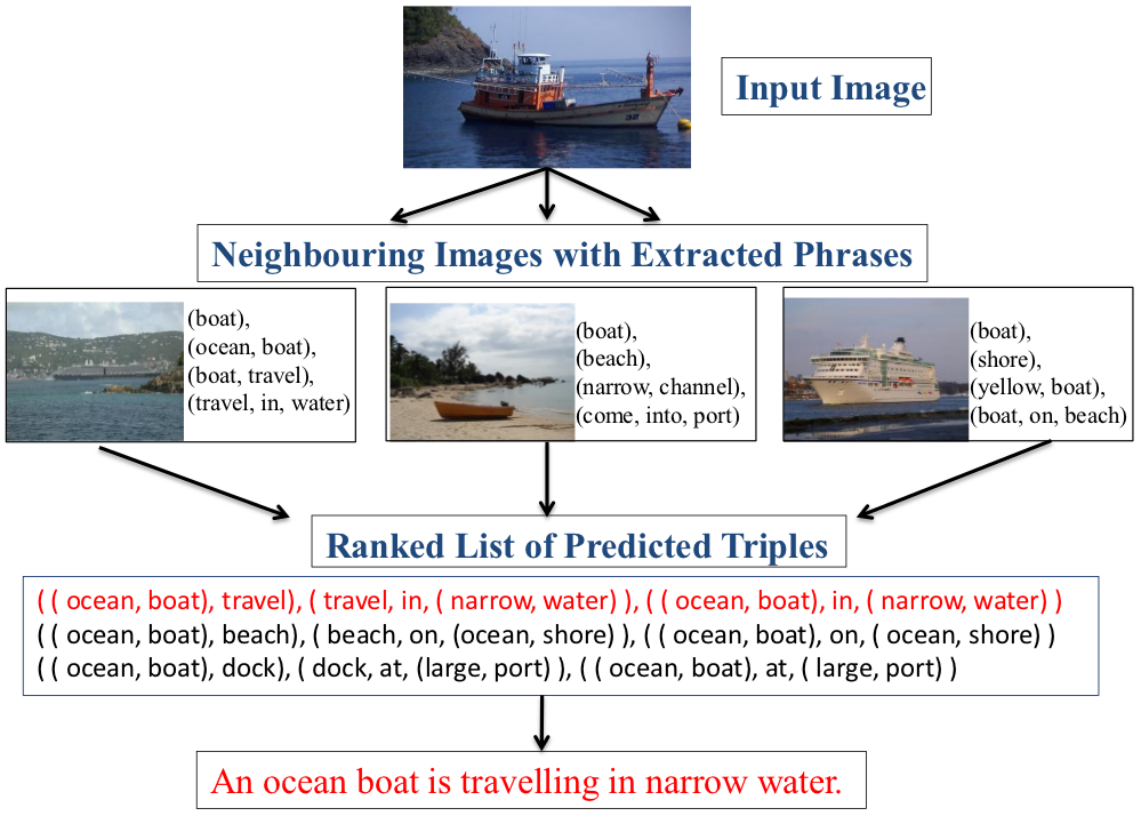


Figure 4.3 Overview of our caption generation approach for an example image. Given an unseen image, first we find K images most similar to it from the training images. Using the phrases extracted from their descriptions, we generate a ranked list of triples. The highest ranked triple is finally used to form the caption.

calculated as

$$t_{score} = \prod_{y_i \in S_t} P(y_i, I). \tag{4.12}$$

where $S_t = \{ (object_1), (attribute_1, object_1), (object_1, verb), (verb, prep, object_2), (object_2), (attribute_2, object_2), (object_1, prep, object_2) \}$. During integration, we look for matching elements in different phrases, as is apparent by the indices; e.g., $(object_1)$, $(attribute_1, object_1)$, $(object_1, verb)$, and $(object_1, prep, object_2)$ have the same object. This helps in restricting the number of feasible triples. These triples are used for caption generation in the next step.

4.4.1 Caption Generation

The output of our phrase integration step is a ranked list of triples. One challenge in automatic generation is to determine the appropriate content. While [157] performs content selection to deal with noisy inputs from visual detectors, [68, 72] use n-gram frequencies for correct ordering preference of words. In our approach, a triple consists of the phrases extracted from the human-generated descriptions. Hence we can expect all such phrases to be clean, and thus we do not require to perform content selection or word reordering explicitly. Once we have determined the content of our sentences (i.e., triple), the task of generation is to frame it into a grammatical form and output the result.

We use the triple with the highest score (Eq. 4.12) for generation. For automatic sentence generation, we use the SimpleNLG [30] library. It is a surface realizer for a simple grammar and has significant coverage of English syntax and morphology. As the extracted triples have a syntactically motivated structure, their mapping into a sentence is straightforward using SimpleNLG. It offers several advantages for our task, which include setting features such as tense (verb), voice (sentence), aspect (verb), etc. Note that though the sentences generated using our approach follow a template-like structure, use of SimpleNLG saves the manual effort of writing individual templates. Figure 4.3 gives an overview of our image captioning pipeline.

4.5 Image Retrieval Using Textual Phrases

In Section 4.3, we described our model for predicting the relevance of a phrase with an image. This is done by combining the similarity of a given image with available images, and similarity of a given phrase with the ground-truth phrases of available images. Then in the previous section (Section 4.4), we described how we use this model for the task of image captioning. Now, we describe how this model can also be used for performing retrieval on an unannotated image collection given a descriptive textual query. In the conventional text-based image retrieval set-up, a query is usually a set of one or more labels. However, these labels carry minimal linguistic information As a result, the retrieved images may not match the desired semantics. Hence it comes as a natural choice to develop approaches that can support image retrieval using descriptive queries. This is also the goal of cross-modal retrieval techniques such as [113]. Such queries inherently carry information about the properties of individual objects as well as relationships among different objects, which can be useful in semantically coherent retrieval.

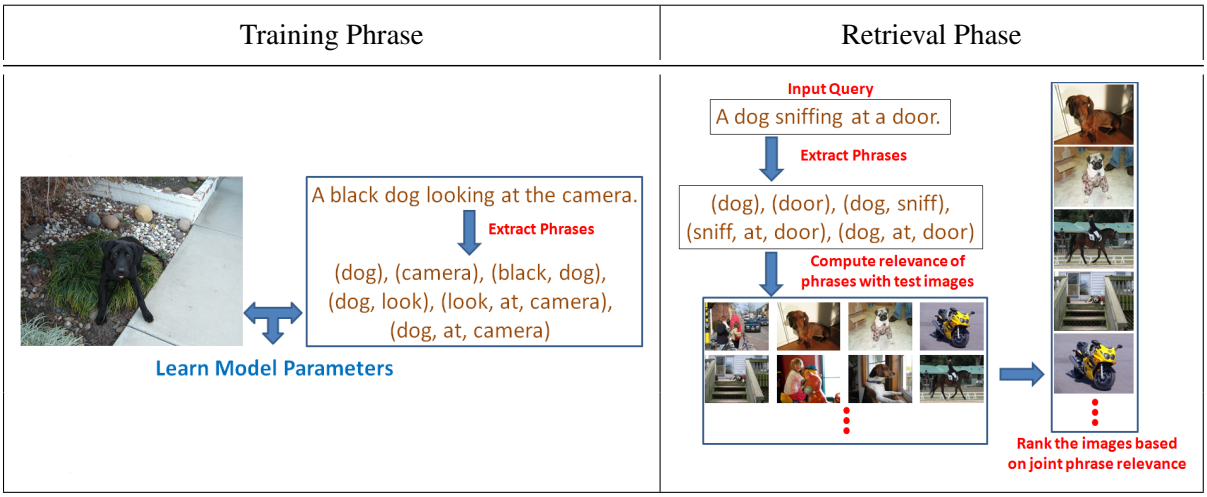


Figure 4.4 Overview of our approach for image retrieval given a caption-based query. During the training phase, we use the phrases extracted from the descriptions of training images to learn the parameters of the phrase relevance prediction model. During the retrieval phrase, first we extract the phrases from the given query. Then we rank the images in the retrieval set based on their joint relevance with these phrases. This is done by matching these images with annotated images, and the query's phrases with those in the available annotations.

Contrary to matching the whole query with images as is done in cross-modal retrieval techniques, here we break it into mid-level phrases, and then perform the matching based on these.

Our approach is motivated by the fact that the available captioned images (i.e., images with corresponding descriptions) can be harvested to perform retrieval on unannotated images. Assuming a collection of annotated images, our goal is to rank unannotated (test/retrieval) images based on their relevance with a given caption-based query. This relevance seamlessly integrates both visual as well as linguistic aspects. Keeping annotated data at the center, visual similarity is computed by matching a test image with the annotated images, and linguistic similarity is computed by matching a query with the descriptions of the annotated images. Rather than using either individual words or the full description as is, images are ranked based on their joint relevance with the phrases extracted from the query.

Let $\mathcal{T}_r$ be the collection of annotated (training) images and $\mathcal{T}_e$ be the collection of unannotated test images (their annotations are used only during evaluation). These images constitute our retrieval set. During the training phase, the samples in $\mathcal{T}_r$ are used to learn the parameters of the phrase relevance prediction model. During the retrieval phase, we are given a sentential/caption-based query Q, and our

goal is to rank the images in $\mathcal{T}_e$ based on their relevance with this query. This is done based on the posterior $P(J|Q)$ of an image $J \in \mathcal{T}_e$ given the query Q:

$$P(J|Q) = \frac{P(Q,J)}{P(Q)} \tag{4.13}$$

Since the denominator is fixed for a given query, we get

$$P(J|Q) \propto P(Q,J) \tag{4.14}$$

To compute $P(Q,J)$, first we parse the query and extract all its phrases Y_Q as described in Section 4.2. Then for each phrase $y \in Y_Q$, we compute its relevance score with the given test image J. This score is the joint probability of associating y with J, and is obtained using Eq. 4.2. Here we consider the annotated images from $\mathcal{T}_r$ to compute the neighbouring images of J, and pick the K most similar images. These images are then used to compute the different components ($P_{\mathcal{F}}(\cdot)$ and $P_{\mathcal{Y}}(\cdot)$) of Eq. 4.2. After computing the relevance of all the phrases in Y_Q with J, we compute the joint relevance score of associating J and query Q as:

$$P(Q,J) = \prod_{y \in Y_Q} P(y,J) \tag{4.15}$$

Similarly, we compute the joint relevance scores for all the images in $\mathcal{T}_e$. Finally, we rank all the test images based on this score in descending order, with higher score implying more relevance and vice-versa.

It is worth noticing that the second term in Eq. 4.2 ($P_{\mathcal{F}}(\cdot)$) considers only visual similarity and is query-independent. Hence, this needs to be computed just once for all the images in the retrieval set, and can be done off-line. That is, we can pre-compute the K nearest neighbours $\mathcal{T}_J^K \subset \mathcal{T}_r$ of each test image J and store their indices and conditional probability scores. Then, during the retrieval phase, we need to compute just the relevance of the phrases in the given query with those of the images in $\mathcal{T}_J^K$ (using Eq. 4.9). Figure 4.4 gives an overview of our image retrieval approach.

4.6 Experiments

4.6.1 Dataset

We use the UIUC Pascal Sentence dataset to evaluate and compare the performance of our approach. This dataset was introduced in [111] and was first used by [27] for the image captioning and caption-based image retrieval tasks. Since then it has been used as a test-bed by most of the methods in this


(1) A girl riding a brown horse.	(1) A bus drives down a busy road.
(2) A girl wearing a red blouse riding a brown horse.	(2) A bus driving in a street.
(3) A woman riding a brown horse.	(3) A passenger bus going down a busy city street.
(4) A young girl riding a brown horse.	(4) A white and red bus traveling on the street.
(5) A young girl wearing a helmet riding a pony.	(5) Red and white bus driving down road.

Figure 4.5 Sample images and their ground-truth captions from the UIUC Pascal Sentence dataset [111].

domain. It comprises 1000 images each described with 5 independent human-generated sentences. Figure 4.5 shows sample images and their ground-truth descriptions from this dataset.

As discussed in Section 4.2, we extract all the phrases from the available descriptions, and consider two set-ups. In the first set-up, all the extracted phrases are used as such. In the second set-up, we replace each subject/object by its synonym determined using WordNet synsets.

4.6.2 Image Features

We represent each image using a set of global (colour, texture and scene) and local (shape) features. The colour features include histograms in the RGB and HSV colour spaces. While RGB is the standard colour space used in the digital displays, HSV encodes some visual properties important for humans such as brightness and colour saturation. To extract texture properties, we use Gabor and Haar descriptors. For scene characteristics, we use the GIST feature [100]. This feature entails a set of perceptual properties such as roughness, ruggedness, naturalness, etc. Finally, for shape we use bag-of-words representation based on SIFT descriptors [78]. These are well-known for extracting local shape properties that are invariant to object size, scale, translation, rotation and illumination. In order to encode some

information about the spatial layout of an image, we also compute all but the GIST features over 3 equal horizontal as well as vertical partitions of an image, which are then concatenated into a single vector. We found that such features were useful in distinguishing between images which differ in their spatial layout. To compute the distance between two feature vectors, we use L_1 distance for colour histograms, L_2 distance for texture and scene features, and χ^2 distance for shape based features.

4.6.3 Image Captioning: Results and Discussion

4.6.3.1 Evaluation Criteria

Similar to [157], we partition the dataset into ten equal parts. Each time, one partition is considered as the test set and all others as the training set. For computing phrase relevance, we consider 50 nearest neighbours of an image. We perform both automatic as well as human-based evaluations for performance analysis.

(I) Automatic Evaluation: BLEU [102] and Rouge [75] are popular metrics in the field of machine translation and text summarization respectively. These compare system generated sentences with respect to human generated sentences. As the image captioning task can be viewed as that of summarizing an image and translating it into text, we use these metrics for evaluation. We consider the BLEU-1 and Rouge-1 scores because the descriptions generated are short. It should be noted that since there is a large scope of variations in descriptions of the same image by different people as compared to the tasks of translating or summarizing text, these two metrics could penalize many correctly generated descriptions. However, we report our results on these metrics as a standard evaluation method. We also report the average BLEU and Rouge scores using the available human generated descriptions for each image in a leave-one-out manner (for each image, considering one out of five captions at a time) as the golden baseline.[3].

(II) Human Evaluation: As discussed above, since an image can be described in several ways, it is not justifiable to evaluate just based on automatic metrics, and hence the need for human evaluation arises. For this, we gathered judgements from two human subjects on 100 randomly picked images randomly picked. The evaluators were asked to verify three aspects on a likert scale of $\{1, 2, 3\}$. We adopt the following definition and guidelines from [72]:

Readability: To measure grammatical correctness of the generated caption by giving the following

[3]To compute BLEU scores, we use version-13a provided by NIST. To compute Rouge scores, we use Release-1.5.5 made available by `http://www.berouge.com/Pages/default.aspx`

	Human (gold)	Kulkarni [68]	PRPM w/ syn.	PRPM w/o syn.	SPRPM w/ syn.	SPRPM w/o syn.
BLEU-1	0.64	0.30	0.41	0.36	0.43	0.36
Rouge-1	0.50	-	0.28	0.21	0.29	0.20

Table 4.2 Automatic evaluation results for sentence generation. (Higher score means better performance.)

Approach	Readability	Relevance
PRPM w/ syn.	1.16	2.51
PRPM w/o syn.	1.25	2.68
SPRPM w/ syn.	1.07	2.39
SPRPM w/o syn.	1.09	2.61

Table 4.3 Human evaluation results corresponding to the measures "Relevance" and "Readability" for the image captioning task. (Lower score means better performance.)

ratings: (1) mostly perfect English sentence, (2) mostly comprehensible with some errors, (3) terrible.
Relevance: To measure the semantic relevance of the generated sentence with the given image by giving the following ratings: (1) very relevant, (2) reasonably relevant, (3) totally off.

Apart from the above two measures, we also try to analyze the *relative relevance* of descriptions generated using PRPM and SPRPM respectively. For this, corresponding to each image, we present the descriptions generated using these two models to the human evaluators (without telling them that they are generated using two different models) and collect judgements based on the following ratings: (1) description generated by PRPM is more relevant, (2) description generated by SPRPM is more relevant, (3) both the descriptions are equally relevant/irrelevant.

4.6.3.2 Quantitative and Qualitative Results

For quantitative comparison, we compare our results with the reported results of [68], which was a state-of-the-art technique for image captioning prior to the recent evolution of deep neural network based methods.[4] One important thing to note here is that it is not fully justifiable to do a direct comparison of our results with those of [68]. This is because in [68], the data used for composing new sentences is quite different from ours. Precisely, in [68], only the top few most frequent objects, prepositions

[4] As discussed earlier, very recently, methods based on deep neural networks have achieved quite promising results for this problem. This is primarily because of the high capacity of the learned models, and the development of efficient techniques for training such models.

With Synonym			Without Synonym		
PRPM count	SPRPM count	Both/None count	PRPM count	SPRPM count	Both/None count
16	28	56	21	25	54

Table 4.4 Human evaluation results corresponding to the "Relative Relevance" measure. The third column in both the blocks denotes the number of times the descriptions generated using the two methods were judged as equally relevant or irrelevant with given image. (Larger count means better performance.)

and verbs are used during caption generation. Whereas, we consider all the phrases extracted from the available descriptions without any omission.

Table 4.2 and Table 4.3 show the results corresponding to automatic and human evaluations respectively. Though not directly comparable, both PRPM and SPRPM provide better quantitative performance than [68]. This suggests usage of textual phrases for caption generation can help in generating semantically and syntactically more relevant captions, as compared to first predicting the various components of an image (objects, attributes, verbs and prepositions) in a piece-wise manner and then combining them using some heuristics as done in [68]. We can also notice that SPRPM mostly performs better than PRPM. This is because SPRPM takes into account semantic similarities among phrases, which in turn results in generating more coherent descriptions than PRPM. This is also highlighted in Figure 4.6 which shows example captions generated using PRPM and SPRPM, along with one of the five ground-truth captions. It can be noticed that the words in the captions generated using SPRPM usually show semantic connectedness, which is not always the case with PRPM. E.g., compare the captions obtained using PRPM (in the second row) with those obtained using SPRPM (in the fourth row) for the last three images. Figure 4.7 shows additional examples of good and bad results as judged by human evaluators. Here we can observe that while the generated captions look syntactically correct, the mistakes are primarily caused due to incorrect prediction of visual clues.

In Table 4.4, the human evaluation results corresponding to "Relative Relevance" of the captions generated using PRPM versus SPRPM are shown. In this case also, SPRPM always performs better than PRPM. This means that the descriptions generated using SPRPM are semantically more relevant than those using PRPM.

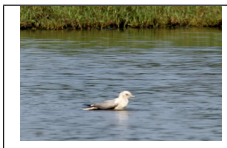
				
A black and white bird on a body of water with grass in the background.	*An old black and white photo of three men on a porch playing music and drinking.*	*A red boat travels down a narrow river surrounded by grassland.*	*People in a yellow school bus sit with their backs to the windows.*	*A woman sitting with a man who is opening a bottle of liquor.*
A mexico aeroplane is taxing along a wet airport.	A groom is posing with a scraggly person.	A sandy field is parked beside a small outpost.	A teal car is sitting atop a white semus.	A decorate room is filling with a snack.
A silver jet is taxing along a wet runway.	A blond woman is posing in a library.	A sandy field is parked beside a small outpost.	A black ferrari is parked in front of a green tree.	A clothed table is filling with a snack.
A black bird is sitting in a blue water.	A young person is posing with a young person.	A small boat is traveling in a blue water.	A yellow bus is parking on a busy road.	A several person is sitting with a several person.
A black bird is sitting in a green water.	A gray man is posing with a gray man.	A small boat is floating on a blue water.	A yellow bus is parking on a city street.	A gray man is sitting at a restaurant table.

Figure 4.6 Example images from the Pascal sentence dataset along with their descriptions. The descriptions in the second row are ground-truth descriptions. The descriptions in the third and the fourth rows are generated using PRPM with and without considering synonyms respectively. The descriptions in the fifth and the sixth rows are generated using our SPRPM with and without considering synonyms respectively.

In Figure 4.8, we try to get some insight about how the internal functioning of SPRPM is different from that of PRPM. For this, we show the top ten phrases of the type *"object"* predicted using the two models for an example image. From these phrases, it can be noticed that the top phrases obtained using SPRPM are semantically very much related with each other (though not all are relevant to the image). Whereas, in case of PRPM, the phrases are quite diversified. This is because in SPRPM, the relevance (or the degree of presence) of a phrase also depends on the presence of other phrases that are semantically similar to it. This results in an indirect propagation of relevance among the phrases, thus collectively pushing semantically related phrases towards the top.

				
A car with a canoe on top is parked on the street near a moped.	*A brown cat sleeping on a sofa.*	*A man and woman are posing for the camera.*	*A man walking a dog on the Beach near large waves.*	*A black and white photo of a glass bottle of Coca Cola.*
A black ferrari is parked in front of a green tree.	An adult hound is laying on an orange couch.	A blond woman is posing with an elvis impersonator.	An osprey is flying over a dirty water.	A motor racer is speeding through a splash mud.
A sporty car is parked on a concrete driveway.	A sweet cat is curling on a pink blanket.	An orange fixture is hanging in a messy kitchen.	A water cow is grazing along a roadside.	A motor person is covering in a splash mud.

Figure 4.7 Additional example images from the Pascal sentence dataset along with their descriptions. The descriptions in the second row are ground-truth descriptions, in the third row are those generated by our method without considering synonyms, and those in the fourth row after considering synonyms. The last two examples are bad results as judged by human evaluators.

4.6.4 Image Retrieval: Results and Discussion

4.6.4.1 Evaluation Criteria

For evaluations, we partition the dataset into 45% training set, 45% retrieval set and 10% query set. The images and their captions in the training set are used to learn the model parameters, the images in the retrieval set constitute the images on which we perform retrieval, and the captions in the query set are used for querying the retrieval set. Analogous to the image captioning task, this is repeated ten times in order to include all the captions into the query set. For computing phrase relevance, we consider 50 nearest neighbours.

We conduct both automatic as well as human evaluations for performance analysis.

(I) Automatic Evaluation: Similar to the image captioning task, we consider BLEU [102] and Rouge [75] metrics for automatic evaluation. For a given query, we average these scores over the top five retrieved images (since top-ranked images are more important in retrieval), by matching the query with their ground-truth captions. For both these measures, we report average unigram scores (i.e., BLEU-1 and Rouge-1).

PRPM:
(1) flap (2) csa (3) symbol (4) aircraft (5) slope (6) crag (7) villa (8) biplane (9) distance (10) sky

SPRPM:
(1) aeroplane (2) airplane (3) plane (4) sky (5) boat (6) water (7) air (8) aircraft (9) jet (10) gear

Figure 4.8 Example image from the Pascal sentence dataset along with the top ten *"objects"* predicted using the two models.

(II) Human Evaluation: We also perform human evaluations to analyze the subjective aspects of image retrieval. For this, we randomly select 100 queries from the Pascal dataset, and collect judgments from two human evaluators on the top five ranked images for each query and take their average. The evaluators are asked to rate the retrieved images on a likert scale of $\{1, 2, 3\}$ based on two aspects:
Relevance of the retrieved images with the given query: (1) none of the retrieved images are relevant to the query, (2) two to three images are relevant, (3) at least four images are relevant.
Diversity among the retrieved images: (1) all the images look quite similar and there is not much diversity, (2) there is moderate level of diversity, (3) there is significant diversity.

It is important to note that diversity need not correspond to relevance, as there may be a high diversity among the retrieved images even with none of them being relevant to a given query. However, diversity among the retrieved images is necessary to satisfy the information need of a user.

4.6.4.2 Quantitative and Qualitative Results

Since our goal is to perform cross-modal image retrieval, we compare our approach with Canonical Correlation Analysis (CCA) [113] which is considered as a popular and de facto baseline in cross-modal retrieval tasks.

In Table 4.5, we report the results for automatic evaluation. Compared to [113], we achieve improvements of up to 7% for both BLEU-1 and Rouge-1 scores. The results also show that analogous to the image captioning task (*c.f.* Table 4.2), we are able to achieve better performance by considering

Method →	CCA [113]	PRPM (w/ syn.)	PRPM (w/o syn.)	SPRPM (w/ syn.)	SPRPM (w/o syn.)
BLEU-1	0.29	0.31	0.35	0.33	0.36
Rouge-1	0.17	0.23	0.22	0.22	0.24

Table 4.5 Automatic evaluation results for the image retrieval task using caption-based queries. (higher score means better performance).

Table 4.6 Human evaluation results corresponding to the measures "Relevance" and "Diversity" for the image retrieval task using caption-based queries. (Higher score means better performance).

Approach	Relevance	Diversity
CCA [113]	1.87	2.13
SPRPM (w/ syn.)	2.16	2.23
SPRPM (w/o syn.)	2.03	2.37

synonyms than without synonyms. This is because nouns play a center role in image retrieval. On replacing each noun with its synonym, the number of distinct phrases reduces. This results in lowering the competition among semantically similar phrases, and thus improves the chances of retrieving images that better match a given query.

Table 4.6 shows results for human evaluation. Here, we consider only SPRPM since its performance was found to be either comparable to or better than PRPM using automatic evaluations. From the results, we can observe that while considering synonyms, the retrieval results are more relevant than without considering synonyms. This indicates that the performance trends using automatic evaluation metrics relate well with human evaluations. However, without synonyms, we achieve more diverse results than those with synonyms. This is because without considering synonyms, there remains more variety among the phrases, even when they have close meanings. This diversity among the phrases in turn gets propagated into the retrieved images. Both automatic as well as human evaluations confirm the superior performance of our approach compared to [113]. This also validates that during retrieval, it is better to consider meaningful chunks of descriptive queries (phrases in our case) rather than whole query as it is.

Figure 4.9 shows the top four images retrieved for sample queries (without considering synonyms). Here it can be observed that for most of the cases, the retrieved images are at least partially relevant to a given query. E.g., in the third column, none of the retrieved images has a "blue and white airplane".

Figure 4.9 Sample queries from the Pascal Sentence dataset along with the top four retrieved images. It can be observed that for most of the cases, the retrieved images are only partially relevant to the given query. E.g., in the first column, none of the images shows a man riding a bike "with one hand". However, each one of these shows a person (a man in the first three, and a woman in the fourth) riding a bike though with both the hands. In the second column, while the people are posing for a picture, only the fourth image completely matches the query, with the first three images not matching the "a man and woman" part of the query. Similar observations can be made for the remaining results, with the degree of relevance between the query and retrieved images degrading as we move from left to right. (See text for further discussion.)

However, there is either a blue or white coloured airplane in each image, with a background depicting the other color (white building or blue sky). Similarly, in the fifth column, while the first images show "boat" and "river", the last two images show a "family". Also, for all the queries, there is significant diversity among the retrieved images, e.g., in terms of views (side-view and front-view) and backgrounds. In the first column, one interesting thing to notice is that while in the first, second and fourth image, a person is riding a bicycle, the third image shows a person riding a motor-bike. All these images closely match the given query "a man riding a bike with one hand". This shows the ability of our approach in retrieving both visually as well as semantically diverse results when a query has some word (particularly noun) with multiple meanings. In the last column, the two images which do not have "cat" have either "dog" and/or "sheep". Thus, rather than retrieving a semantically irrelevant image (such as one showing a "bus"), these images look relatively better since both the nouns "dog" and "sheep" are types of "animal" and have good semantic similarity with "cat". This indicates that even when the image content does not completely match with the query, we are able to retrieve images that either match its components or are semantically close.

4.7 Summary

We have proposed novel approaches for generating fluent and human-like descriptions for images, and performing image retrieval given caption-based queries. Though simple, we achieve promising results by analyzing and extracting the information encoded in the available image descriptions, without relying on any pre-trained object detectors, classifiers, hand-written grammar rules or heuristics.

Both our image captioning as well as image retrieval pipelines rely on a model for predicting the relevance between a textual phrase and an image, which we called as phrase relevance prediction model or PRPM. We further extended this model by incorporating semantic similarities among phrases during phrase prediction and parameter learning steps, which we called as semantic phrase relevance prediction model or SPRPM. In practice, as the diversity in data increases, inter-phrase relationships start becoming prominent. In such situations, analyzing the available data in isolation becomes insufficient. Through SPRPM, we have shown how external sources of knowledge such as WordNet can be useful in expressing such complex relationships in textual data.

Directions for Future Research: As discussed towards the beginning of this chapter, our approach was motivated by classical machine learning techniques based on probabilistic relevance models. However,

most of the recent papers in the area of image captioning as well image retrieval are based on deep neural network based models (Section 4.1.1). An interesting and promising direction for future research would be to investigate the fusion of these two and analyze their utility for different tasks. This would involve additional challenges such as dealing with different forms of text (phrases and captions), generating captions by integrating phrases rather than (or in addition to) using a sequence of discrete words, learning composition of phrases rather than words, etc.

Chapter 5

A Support Vector Approach for Cross-modal Image↔Text Retrieval

5.1 Introduction

During the past decade, there has been an explosion of multimedia content on the Internet. As a result, several interesting as well as challenging research problems have emerged, one of them being automatically describing image content using text. As discussed in the previous chapters, while most of the earlier as well as recent research has focused on automatically annotating images using semantic labels [24, 28, 33, 35, 79, 140, 153], in the past few years, describing images using phrases [39, 69, 117, 139], or one or more simple captions [20, 26, 27, 39, 43, 62, 66, 68, 69, 82, 101, 109, 139, 157] have attained significant attention. A complementary problem to these is to automatically associate one or more semantically relevant images given a piece of text (such as a label, phrase or caption), and is commonly referred to as the image retrieval task [18, 27, 28, 33, 35, 43, 104, 113].

Building bilateral semantic associations between images and texts is among the fundamental problems in computer vision. Although huge amounts of independent visual and textual data are available today, only a small portion of them is semantically connected. Hence, it comes as a natural choice to develop new models that can efficiently learn complex associations between the two modalities using this small portion, and later apply them to automatically build associations between the two in the larger, independent space. In our work [142, 145], we try to address this problem of learning cross-modal associations between visual and textual data. We study two complementary tasks: (i) retrieving semantically relevant text(s) given a query image (*Im2Text*), and (2) retrieving semantically relevant image(s) given a query text (*Text2Im*). We pose both these tasks as retrieval problems, where the output samples are ranked based on their relevance with the query. In contrast to several existing methods such as [27, 28, 35, 39, 69, 79, 101, 140] that make use of data from both the modalities (image and text) during

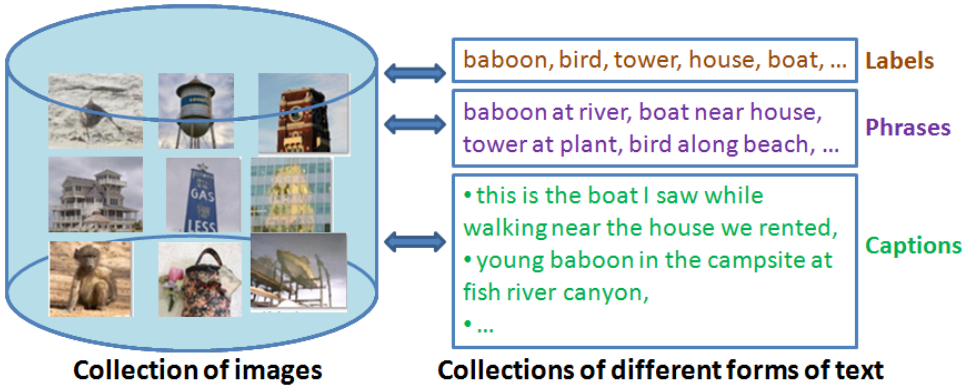


Figure 5.1 We propose a Structural SVM based unified framework that learns bilateral associations between images and different forms of texts (labels/phrases/captions). Our approach can be used to perform cross-modal retrieval on an independent database of textual data given a query image ("Im2Text"), and vice-versa ("Text2Im").

the prediction phase, our approach is similar to the cross-modal retrieval work such as [33,40,43,61,113] that do not make such an assumption. This means that for Im2Text, given a query image, we retrieve a ranked list of semantically relevant texts from a plain text-corpus that has no associated images. Similarly, for Text2Im, given a query text, we retrieve a ranked list of images from an independent collection of images without any associated textual meta-data. Figure 5.1 illustrates the theme of our work.

Several existing techniques for cross-modal retrieval such as [33, 43, 112, 113] first learn a joint embedding space using two different modalities, and then perform cross-modal retrieval in this space using some simple similarity/distance function such as Manhattan distance, Euclidean distance, cosine similarity (also called normalized correlation), etc. In order to leverage the potential of these approaches, we proposed a generic cross-modal retrieval technique based on Structural SVM [142] (Figure 5.2). In addition to raw features, this can use the features learned using any of the above mentioned techniques, and can be efficiently trained using a variety of loss functions. Precisely, the major contributions of our work are:

1. We propose a novel Structural SVM [132] based unified framework for both Im2Text and Text2Im, which provides the following three advantages. First, Structural SVM provides a natural framework to work with complex and structured input/output spaces, and a unified framework helps in better understanding and appreciating the complementary nature of the two problems. Second, our general-purpose learning module can be easily applied to different forms of cross-

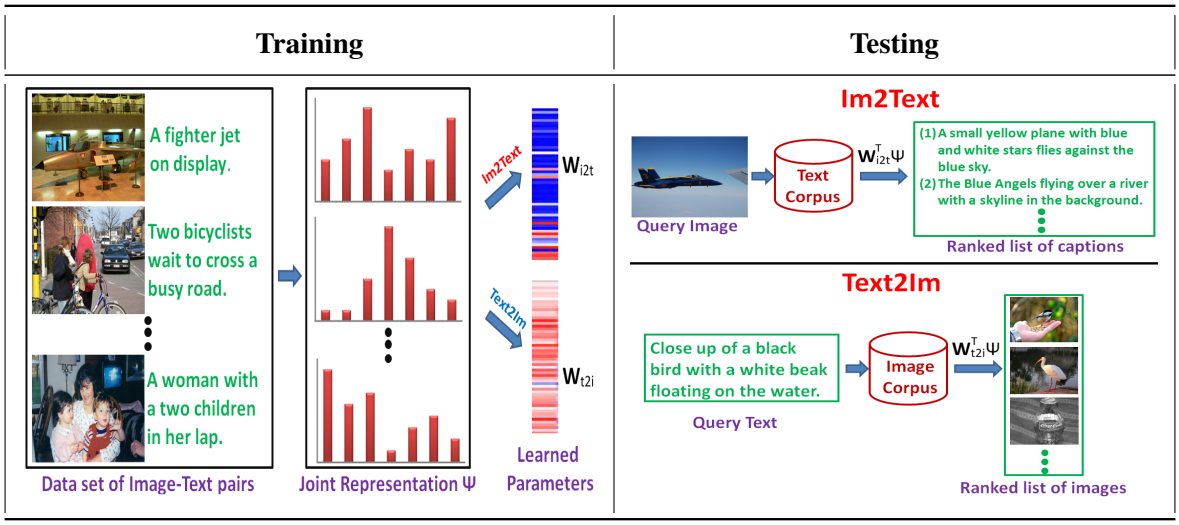


Figure 5.2 While training, given a dataset consisting of pairs of images and corresponding texts (here captions), we learn models for the two tasks (Im2Text and Text2Im) using a joint image-text representation. While testing for Im2Text, given a query image, we perform retrieval on a collection of only textual samples using the learned model. Similarly, for Text2Im, given a query text, retrieval is performed on a database consisting only of images.

modal data (diverse modalities with paired cross-modal samples and feature vector based representations) with little modifications. Third, availability of efficient algorithms for Structural SVM training (such as the cutting-plane algorithm [132]) makes it feasible to efficiently learn max-margin models that scale well with data size. As per our knowledge, this is the first attempt to examine and validate the applicability of Structural SVM for performing cross-modal multimedia retrieval.

2. Since our framework is based on Structural SVM, it allows us to learn model parameters using a variety of loss functions. To demonstrate this adaptability, we examine three loss functions in our work. These loss functions do not make any assumption on the specific form of data, and also connect well with representations popularly used for data from diverse modalities.

3. As a part of our experimental analysis, we examine generalization of ours as well as other competing baseline methods across datasets when textual data is in the form of captions/descriptions. For this, we learn models from one dataset, and perform retrieval on others.

To validate the applicability of our method, we conduct experiments on three diverse and popular datasets, namely, UIUC Pascal Sentence dataset [111], IAPR TC-12 benchmark [34], and SBU-Captioned Photo dataset [101]. Among these, Pascal and IAPR datasets are medium scale datasets containing few thousands of samples, and SBU is a web-scale dataset containing one million samples. Also, while the images in Pascal and SBU datasets are associated with short captions that are a few sentences long, those in the IAPR dataset are coupled with long captions that give a detailed description of an image. Extensive evaluations on these datasets demonstrate the utility of the proposed framework as compared to competing baseline techniques.

5.1.1 Related Work

Here, first we discuss related work on unimodal, multi-modal, and cross-modal retrieval, particularly focusing on images and text as the two modalities. Then we review a few works that perform multi/cross-modal learning in some diverse applications.

Image-Text Retrieval: The problems of image and text retrieval are well-studied research topics [18, 84, 104, 123, 124]. A large number of the existing approaches are based on retrieval of unimodal data; i.e., both query as well as the retrieved samples belong to the same modality (e.g., either image [124] or text [84]). Another approach that is popular among web-based search engines is to use textual meta-data associated with images during retrieval. Given a textual query, it is directly matched with this meta-data instead of looking at the corresponding image. However, such images constitute only a small portion of the enormous amount of images available on the Internet, most of which are without such meta-data. This limitation has led to a growing interest in the problem of automatic image annotation [6, 24, 28, 33, 35, 79, 99, 140, 141, 153] (Chapter- 3). Such models can also support label-based queries during image retrieval without assuming availability of any associated textual meta-data.

In parallel, there have also been several advances in the area of multi-modal retrieval [13, 25, 104, 128], where retrieval is performed based on multiple modalities. These are based on either learning a separate model for each modality and then combining their predictions, or combining features from different modalities and then learning a single model over them. However, these approaches require data from all the modalities during the prediction phase. Moreover, some of them make use of multi-modal queries [128], making these somewhat difficult for large scale retrieval tasks.

In the recent years, cross-modal matching and retrieval have been actively studied [33, 43, 47, 61, 85, 112, 113, 119, 150]. Among these, Canonical Correlation Analysis (CCA) [40] is one of the most

popular methods. It learns a latent projection space where the correlations between paired features from two modalities are maximized. In this space, samples from different modalities are matched using some simple nearest-neighbour based technique. Inspired from its simplicity and efficiency, several approaches have been proposed that perform cross-modal matching based on CCA [40,43,47,112,113]. While in [40,43,112,113], CCA is used to perform cross-modal retrieval of images and their associated descriptions, [47] uses it to learn associations between images and tags. Other than CCA, methods such as Partial Least Squares (PLS) [116] and Bilinear Model (BLM) [130] have been proposed for cross-modal problems. Lately, there have also been some attempts on using deep neural networks for learning cross-modal associations between images and texts [2,63,125,156]. Note that most of the above mentioned approaches make use of two modalities while doing cross-modal matching. However, in some cases, additional information is also available in the form of category labels (third modality/view). To make use of this, there have been some recent attempts in learning the latent embedding space using multi-view data [33,61,110,112,119].

In summary, most of the existing cross-modal matching algorithms try to learn a common space that captures the intrinsic correlations present in the data. This space provides a homogeneous representation for samples from diverse modalities, which in turn allows direct matching between cross-modal samples.

Multimodal Representations: In addition to cross-modal matching of natural scene images and text, there have also been attempts in other domains that focus on dealing with diverse multi-modal data. Some of the examples include scene-text understanding [114], multi-modal clustering [23], modelling pairwise relations [83] and multi-modal image annotation [45,57]. In [114], images of scene-text and text-strings are first embedded into vector spaces, and then a compatibility function is learned that helps in scene-text recognition and retrieval. Since a large portion of images on the web are associated with noisy and/or sparse meta-data (e.g., text, GPS coordinates, camera specifications, etc.), a constrained multi-modal clustering approach was proposed in [23]. In [83], relational meta-data in the form of social connections was harnessed to model pair-wise relations between images. Two recent methods [45,57] demonstrated the utility of additional metadata (such as user-generated tags [57] and label relations based on WordNet taxonomy [45]) in boosting image annotation performance. Similar to these approaches, our interest is in learning higher level semantics using diverse modalities. However, we will concentrate on the task of cross-modal retrieval, and demonstrate the applicability of our approach considering images and text as the two modalities.

5.2 Bilateral Image-Text Retrieval

In the conventional classification task, the goal is to assign a category from a finite set of discrete categories to a given (test) sample. A popular approach to do this is by training a category specific max-margin classifier using one-versus-rest (or multi-class) Support Vector Machine (SVM) [16]. However, this becomes prohibitive when (1) the number of categories is exponentially large, and (2) the categories encode higher-level structure rather than being just simple labels. To overcome these, Structural SVM was introduced in [132]. Structural SVM is an oracle framework that can be adapted for a variety of tasks like object detection, classification with taxonomies, label sequence learning, etc. by appropriately defining its components that suit the problem at hand. In this chapter, we will discuss our approach for addressing the problem of cross-modal multimedia search using Structural SVM. As per our knowledge, a large number of existing methods for cross-modal search are based on nearest-neighbour based similarity matching (in a learned homogeneous latent space). As we will show, Structural SVM naturally suits this task, where both input as well as output modalities can be quite complex in general (image↔text in our case), and may have inherent structure in them. Moreover, availability of efficient algorithms for Structural SVM training (e.g., the cutting-plane algorithm [132]) make it scalable to large scale datasets.

5.2.1 Approach

Here we present our framework for cross-modal search. During the training phase, we learn the associations between images and texts based on a joint representation. During the testing phase, we use the learned model to perform cross-modal search. Figure 5.2 illustrates our framework. As the proposed approach performs two complementary tasks (Im2Text and Text2Im), we will refer to it as *Bilateral Image-Text Retrieval* (*BITR*).

First, we consider the task of retrieving semantically relevant text(s) given a query image (i.e., Im2Text). In Section. 5.2.4, we will discuss how the same framework is applicable for Text2Im as well. Let $\mathcal{D} = \{(I_1, T_1), \ldots, (I_N, T_N)\}$ be a collection of N images and corresponding texts. Each image I_i is represented using a p-dimensional feature vector $\mathbf{x}_i$ in space $\mathcal{X} = \mathbb{R}^p$. Similarly, each text T_i is represented using a q-dimensional feature vector $\mathbf{y}_i$ in space $\mathcal{Y} = \mathbb{R}^q$. Similar to the Structural SVM framework [132], our objective is to learn a discriminant function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ that can be

used to predict the optimal output $\mathbf{y}^*$ given an input $\mathbf{x}$ by maximizing F over the space $\mathcal{Y}$; i.e.,

$$\mathbf{y}^* = f(\mathbf{x}; \mathbf{w}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} \; F(\mathbf{x}, \mathbf{y}; \mathbf{w}) \tag{5.1}$$

where $\mathbf{w}$ is the parameter vector that needs to be learned. We make the standard assumption of $F = \mathbf{w} \cdot \Psi(\mathbf{x}, \mathbf{y})$; i.e., F is a linear function of the joint feature representation $\Psi(\cdot)$ of the input-output pair. In the above setting, our goal is to learn $\mathbf{w}$ such that the maximum number of the following constraints are satisfied:

$$\forall i: \;\; \{\mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}_i) > \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y})\} \;\; \forall \mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i \tag{5.2}$$

The above constraints signify that for every sample $\mathbf{x}_i$, the parameter vector $\mathbf{w}$ should be learned such that the prediction score for the true output (i.e., $F(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w})$) remains higher than that for any other output. In practice, its solution is approximated by introducing non-negative slack variables. The task of learning $\mathbf{w}$ is then formulated as the following optimization problem:

$$\begin{gathered} \min_{\mathbf{w}, \xi \geq 0} \;\; \frac{1}{2}||\mathbf{w}||_2^2 + \frac{C}{N} \sum_{i=1}^{N} \xi_i \\ s.t. \;\; \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}) + \Delta(\mathbf{y}_i, \mathbf{y}) - \xi_i \;, \;\; \forall i, \mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_i\} \end{gathered} \tag{5.3}$$

where $||\cdot||_2^2$ denotes squared L_2-norm, $C > 0$ is a constant that controls the trade-off between the regularization term and the loss term, ξ_i denotes the slack variable, and $\Delta(\mathbf{y}_i, \mathbf{y})$ denotes the loss function that acts as a margin for penalizing any prediction other than the true output.[1] In the above optimization problem, the joint representation $\Psi(\mathbf{x}, \mathbf{y})$ and the loss function $\Delta(\mathbf{y}_i, \mathbf{y})$ are problem specific functions that need to be defined based on the given task.

5.2.2 Details

Now we describe the different components of our approach (i.e., the joint representation and the loss function), and how to efficiently solve the optimization problem in Eq. 5.3 for learning the parameter vector $\mathbf{w}$.

5.2.2.1 Joint Image-Text Representation

The purpose of $\Psi(\mathbf{x}, \mathbf{y})$ is to provide a joint representation for input and output data depending upon their individual forms. In cross-modal search (and in general), one popular way of representing

[1] In [132], two formulations are presented for Structural SVM training. These are based on "margin-rescaling" and "slack-rescaling". We adopt the margin-rescaling one, which uses different margins for different possible outputs based on their degree of similarity with the true output.

a sample is in the form of a feature vector, which is usually computed based on domain knowledge of the modality under consideration. For a given sample, each dimension of its feature vector carries some information that is specific to that sample, and thus helps in distinguishing it from other samples within that modality. Another well-known practice is to normalize a feature vector before using it (e.g., using either L_1 or L_2 normalization), and is commonly adopted by almost all the practical systems including cross-modal matching techniques.

Now let us consider an image-text pair (I, T), where I is represented using a feature vector $\mathbf{x} \in \mathcal{X}$ and T using another feature vector $\mathbf{y} \in \mathcal{Y}$, both of which are appropriately normalized. Since these two feature vectors are computed using different techniques and can have different dimensionality (i.e., p need not be equal to q), direct comparison between the two may be impractical. However, as discussed above, each dimension of a feature vector carries some information that is specific to the sample it represents. Hence, one feasible choice to learn correspondence between $\mathbf{x}$ and $\mathbf{y}$ is by considering all possible pairs of their individual elements. Intuitively, this will capture "cross-interactions" between the elements of the two vectors. When we learn a weight vector ($\mathbf{w}$) over these pairs, each entry in this weight vector would denote the significance of interaction between the corresponding cross-modal feature-element pair.

Thus we propose to use the joint representation constructed from the input-output representations $\mathbf{x}$ and $\mathbf{y}$ using their tensor product. That is, each dimension of $\mathbf{x}$ is multiplicatively combined with every dimension of $\mathbf{y}$ to get

$$\Psi(\mathbf{x}, \mathbf{y}) = \mathbf{x} \otimes \mathbf{y} \in \mathbb{R}^r, \tag{5.4}$$

where $r = p \times q$. This representation has the apparent advantage of not only efficiently capturing linear interactions between the input and output modalities, but also provides computational benefits during inference, as we will discuss in Section 5.3.2.

5.2.2.2 Loss Function

The function $\Delta(\mathbf{y}_i, \mathbf{y})$ in Eq. 5.3 is a problem specific loss function. It acts as a margin in the Structural SVM framework, and is used to penalize incorrect predictions against the true output. Given an input-output pair $(\mathbf{x}_i, \mathbf{y}_i)$ and any other prediction $\mathbf{y} \neq \mathbf{y}_i$, the function is defined such that its value depends on the degree of dissimilarity between $\mathbf{y}_i$ and $\mathbf{y}$. That is, if $\mathbf{y}_i$ and $\mathbf{y}$ are dissimilar, the value of $\Delta(\mathbf{y}_i, \mathbf{y})$ should be high and vice-versa.

Representing the samples in the output data (T_is) in a vector space $\mathcal{Y}$ allows us to define our loss function based on a suitable distance/similarity metric defined in vector space. Though this mapping can be highly non-linear in nature, the assumption here is this space keeps the semantic proximity of the data intact; i.e., data points that are semantically similar are closer to each other in the vector space, than the data points that are semantically dissimilar to each other.[2] Based on this intuition, we define three different loss functions that are based on popular distance/similarity metrics: Manhattan distance $\Delta_{_M}(\cdot)$, squared Euclidean distance $\Delta_{_E}(\cdot)$, and normalized correlation (or cosine similarity) $\Delta_{_C}(\cdot)$. These loss functions are given by:

$$\Delta_{_M}(\mathbf{y}_i, \mathbf{y}) = ||\mathbf{y}_i - \mathbf{y}||_1, \tag{5.5}$$

$$\Delta_{_E}(\mathbf{y}_i, \mathbf{y}) = ||\mathbf{y}_i - \mathbf{y}||_2^2, \tag{5.6}$$

$$\Delta_{_C}(\mathbf{y}_i, \mathbf{y}) = 1 - \mathbf{y}_i \cdot \mathbf{y}, \tag{5.7}$$

where $||\cdot||_1$ denotes L_1-norm. Since both $\Delta_{_M}(\cdot)$ and $\Delta_{_E}(\cdot)$ are distance metrics, they satisfy the properties of a valid loss function [132]; i.e., $\Delta_{_Z}(\mathbf{y}_i, \mathbf{y}_i) = 0$, $\Delta_{_Z}(\mathbf{y}_i, \mathbf{y}_j) \geq 0$ for $i \neq j$, and $\Delta_{_Z}(\mathbf{y}_i, \mathbf{y}_j) \geq \Delta_{_Z}(\mathbf{y}_i, \mathbf{y}_i)$ for $i \neq j$ (where $Z \in \{M, E\}$). Under the assumption that both $\mathbf{y}_i$ and $\mathbf{y}$ are L_2-normalized, $\Delta_{_C}(\cdot)$ also satisfies these properties and thus is a valid loss function. The efficient evaluation of these loss functions helps in a fast computation of the most violated constraint, which is required while solving the optimization problem in Eq. 5.3.

5.2.2.3 Finding the Most Violated Constraint

Since the number of constraints in Eq. 5.2 can be exponentially large, it could be practically infeasible to make even a single pass over all the constraints during optimization.[3] Hence it becomes crucial to efficiently find a small set of active constraints that would ensure a sufficiently accurate solution. This is achieved using the cutting-plane algorithm proposed in [132]. As discussed in [132], rather than considering all the constraints corresponding to a given pair $\{\mathbf{x}_i, \mathbf{y}_i\}$, this algorithm aims at finding the constraint that is violated the most, also called the most violated constraint. This in turn reduces the solution space by creating a nested sequence of tighter relaxations of the original problem.

Given an input-output pair $(\mathbf{x}_i, \mathbf{y}_i)$, the most violated constraint is the constraint corresponding to the incorrect output $\hat{\mathbf{y}}$ predicted with the maximum score using the current learned parameter vector $\mathbf{w}$.

[2] This is a fundamental assumption that is usually at the heart of some machine learning algorithms.

[3] Potentially infinite in our case, since $\mathcal{Y}$ is a continuous real-valued vector space.

It is given by:

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_i\}}{\operatorname{argmax}} \; \Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}) - \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}_i) \tag{5.8}$$

Since the last term is constant with respect to $\mathbf{y}$, this can be re-written as:

$$\hat{\mathbf{y}} = \underset{\mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_i\}}{\operatorname{argmax}} \; \Delta(\mathbf{y}_i, \mathbf{y}) + \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}) \tag{5.9}$$

For the three loss functions in Eq. 5.5, 5.6, and 5.7, this maps to the following problems respectively:

$$\hat{\mathbf{y}}_M = \underset{\mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_i\}}{\operatorname{argmax}} \; ||\mathbf{y}_i - \mathbf{y}||_1 + \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}) \tag{5.10}$$

$$\hat{\mathbf{y}}_E = \underset{\mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_i\}}{\operatorname{argmax}} \; ||\mathbf{y}_i - \mathbf{y}||_2^2 + \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}) \tag{5.11}$$

$$\hat{\mathbf{y}}_C = \underset{\mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_i\}}{\operatorname{argmax}} \; 1 - \mathbf{y}_i \cdot \mathbf{y} + \mathbf{w} \cdot \Psi(\mathbf{x}_i, \mathbf{y}) \tag{5.12}$$

It can be easily verified that each of the above three equations corresponds to maximizing a convex function. In practice, since every feature vector is normalized, each of its elements remains bounded within a range. This allows us to solve the above problems efficiently using an iterative gradient-ascent method. After each iteration of gradient-ascent, the current output is projected depending on the particular type of normalization considered. Further details on this can be found in our publicly available implementation.[4]

5.2.3 Inference: Retrieving a Ranked List of Output

Consider an independent database $\mathcal{T}' = \{T'_1, \ldots, T'_{|\mathcal{T}'|}\}$ consisting of only textual samples, where each T'_k is represented using a feature vector $\mathbf{y}'_k \in \mathcal{Y}$. Given a query image J represented by $\mathbf{x} \in \mathcal{X}$, Im2Text requires ranking the elements of $\mathcal{T}'$ according to their relevance with $\mathbf{x}$ using the learned parameter vector $\mathbf{w}$. This can be performed by sorting the elements of $\mathcal{T}'$ based on the score $F(\mathbf{x}, \mathbf{y}'_k; \mathbf{w}) = \mathbf{w} \cdot \Psi(\mathbf{x}, \mathbf{y}'_k), \;\; \forall k \in \{1, \ldots, |\mathcal{T}'|\}$ (where higher score means more relevance and vice-versa), thus allowing us to retrieve a ranked list of texts.

5.2.4 Performing "Text2Im"

Now we consider the task of retrieving semantically relevant image(s) given a query text (i.e., Text2Im). Similar to Im2Text, we are given a collection $\mathcal{D} = \{(I_1, T_1), \ldots, (I_N, T_N)\}$ of images and corresponding texts. Each image I_i is represented using a p-dimensional feature vector $\mathbf{x}_i$ in space

[4] http://researchweb.iiit.ac.in/~yashaswi.verma/crossmodal/bitr.zip

$\mathcal{X} = \mathbb{R}^p$, and each text T_i is represented using a q-dimensional feature vector $\mathbf{y}_i$ in space $\mathcal{Y} = \mathbb{R}^q$. Our objective now becomes to learn a discriminant function $F : \mathcal{Y} \times \mathcal{X} \rightarrow \mathbb{R}$ that can be used to predict the optimal output (image) $\mathbf{x}^*$ given an input (text) $\mathbf{y}$ by maximizing F over the space $\mathcal{X}$. That is,

$$\mathbf{x}^* = f(\mathbf{y}; \mathbf{w}) = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmax}} \; F(\mathbf{y}, \mathbf{x}; \mathbf{w}), \tag{5.13}$$

where $\mathbf{w}$ is the parameter vector that needs to be learned, and $F = \mathbf{w} \cdot \Psi(\mathbf{y}, \mathbf{x})$. Since we make no specific assumption on the particular representations used for visual and textual data (except that they are represented in the form of feature vectors), the joint representation and loss functions defined above for Im2Text will remain equally applicable for Text2Im as well. Hence, in order to perform Text2Im, we can adopt the same methodology as that for Im2Text. However, note that here since we are dealing with a different (inverse) problem, we learn a separate model ($\mathbf{w}$).

5.3 Training time and Run-time Analysis

Here we will analyze the training and run-time efficiency of the proposed approach, and compare it with two competing baselines: CCA [113] and WSABIE.[5] We will consider the task of Im2Text, with similar reasoning being applicable to Text2Im as well.

5.3.1 Training time analysis

In Figure 5.3, we compare the training time of WSABIE [153] and BITR using synthetic features. Here we do not show the training time of CCA [40, 113] because its standard implementations are quite efficient, and it usually takes less than 1 second to learn the common space in the below mentioned set-up. For WSABIE, we use early stopping after iterating for 20 passes of training samples following [33]. Hence its training time will be the same for different values of C. For comparison, we vary the number of training pairs in $\{5K, 20K, 100K\}$ and the dimensionality of image/text features in $\{50, 100, 150\}$. In the figure, the horizontal axis denotes the value of the C parameter (power of 10), and the vertical axis denotes the training time in seconds. From the figure, we observe that the training time of BITR increases as the feature dimensionality and number of training pairs increases, which is obvious. In addition, it also depends on the value of the C parameter. E.g., the training time of BITR is under 15 minutes even for $100K$ pairs when C is small. However, on increasing C beyond 10, there is a steep

[5] On a twelve-core 2.4 GHz Intel Xeon (E5-2600) processor with 48 GB of RAM.

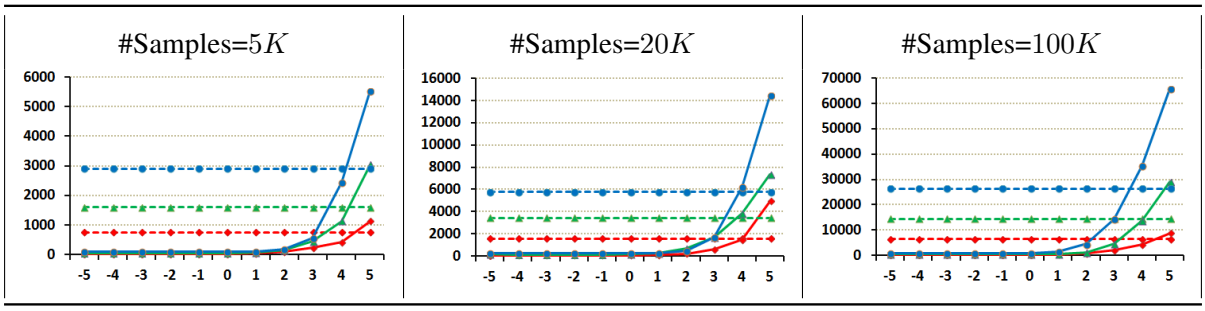


Figure 5.3 Comparison of the training time using WSABIE and BITR. The horizontal axis denotes the value of the C parameter (power of 10), and the vertical axis denotes the training time in seconds. Dashed lines correspond to WSABIE and solid lines correspond to BITR. Each colour denotes the dimensionality of feature vector (same for both the modalities): {red, green, blue} map to {50, 100, 150} in that order.

rise in the training time. This is expected because on increasing C, the algorithm tries to better fit the model to the training data. For example, using $100K$ pairs and 150 dimensional image and text features (joint representation of $150 \times 150 = 22500$ dimensions), with $C = 10^{-5}$ it takes just around 15 minutes to train the model, whereas with $C = 10^{5}$ it takes around 18 hours. This analysis demonstrates even though the training time of BITR can be quite high for large values of C, it is still feasible and thus easily scalable to large datasets.

5.3.2 Run-time Analysis

It is interesting to note that in order to evaluate the function $F(\mathbf{x}, \mathbf{y}; \mathbf{w})$, we do not need to explicitly compute the joint representation $\Psi(\mathbf{x}, \mathbf{y})$. Since $\Psi(\mathbf{x}, \mathbf{y})$ is based on a tensor product of the vectors $\mathbf{x} \in \mathbb{R}^p$ and $\mathbf{y} \in \mathbb{R}^q$, it is a vector of products of pairs of elements from $\mathbf{x}$ and $\mathbf{y}$:

$$\Psi(\mathbf{x}, \mathbf{y}) = [\mathbf{x}(1)\mathbf{y}(1), \ldots, \mathbf{x}(p)\mathbf{y}(1), \ldots, \mathbf{x}(p)\mathbf{y}(q)]^t \in \mathbb{R}^r$$

where the superscript t denotes vector transpose. Since $\mathbf{w}$ is also a vector in $\mathbb{R}^r$, it can be re-written in matrix form:

$$\mathbf{W} = [\mathbf{w}_1 \mathbf{w}_2 \ldots \mathbf{w}_q] \in \mathbb{R}^{p \times q},$$

where each $\mathbf{w}_k \in \mathbb{R}^p$ denotes the consecutive p elements of $\mathbf{w}$ in the k^{th} interval. Using the above, it is easy to verify that the function $F(\mathbf{x}, \mathbf{y}; \mathbf{w})$ can be re-written as:

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \mathbf{w} \cdot \Psi(\mathbf{x}, \mathbf{y}) = \mathbf{x}^t \mathbf{W} \mathbf{y} \tag{5.14}$$

Rather than evaluating the function $F(\mathbf{x}, \mathbf{y}; \mathbf{w})$ individually for each sample in the retrieval set $\mathcal{T}'$, the above transformation allows us to evaluate it for a batch of samples in $\mathcal{T}'$ in a single pass. Here we will illustrate this by computing it for all the samples in $\mathcal{T}'$ in a single pass. Let $\mathbf{Y} = [\mathbf{y}_1 \mathbf{y}_2 \ldots \mathbf{y}_{|\mathcal{T}'|}] \in \mathbb{R}^{q \times |\mathcal{T}'|}$ denote the matrix formed by concatenating the feature representations of all the samples in $\mathcal{T}'$. For a given (image) query J represented by feature vector $\mathbf{x}$, let $\mathbf{s} \in \mathbb{R}^{|\mathcal{T}'|}$ be a vector such that its k^{th} element denotes the relevance score corresponding to the k^{th} sample in $\mathcal{T}'$. Then it can be computed as:

$$\mathbf{s} = \left(\mathbf{x}^t \mathbf{W} \mathbf{Y}\right)^t. \tag{5.15}$$

After computing this, the ranking follows by sorting the elements of $\mathcal{T}'$ based on their corresponding scores in $\mathbf{s}$ in descending order. In popular matrix multiplication software (such as Matlab), the joint computation of similarity scores for a batch of samples can be much faster than computing them individually. This in turn provides significant boost in run-time efficiency.

Assuming the features are already computed, Figure 5.4 (left) compares the relative time required for ranking the samples in a synthetic retrieval set $\mathcal{T}'$ for a single query. In cross-modal search scenarios, the samples from both the modalities are usually represented using feature vectors containing a few tens or hundreds of elements [113]. Keeping this in mind, we keep $p = q = 100$ (recall that the dimensionality of the joint feature representation is $r = p \times q$, which is 10^4 in this case), and vary the number of samples in $\mathcal{T}'$ in $\{10^1, 10^2, \ldots, 10^7\}$. We consider three situations, when the prediction score is computed (a) one sample at a time by first computing the joint representation, (b) one sample at a time without computing the joint representation (Eq. 5.14), and (c) jointly for all the samples without computing the joint representation (Eq. 5.15). From the figure, we observe that for all these three, the total time (including relevance score computation and sorting) increases almost linearly with the number of samples. However, even with a linear increment, the total time required for (c) is significantly lower than that for (a) and (b). For example, when the retrieval set has ten million samples, the time taken when using (a), (b) and (c) are around 345.8, 97.6, and 1.7 seconds respectively. For all three, around 1.2 seconds are taken in sorting the samples based on their scores. If we do not consider this, then (c) takes just around 0.5 seconds in computing the prediction scores for all the samples, which is faster than the time required for the sorting operation.

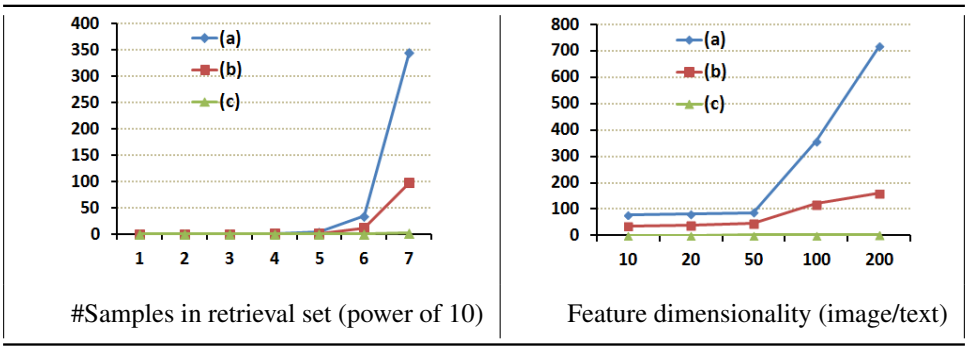


Figure 5.4 Comparison of time required (in seconds on vertical axis) for ranking the samples in a retrieval set $\mathcal{T}'$ for a single query, when the prediction score is computed (a) individually for each sample after computing the joint representation, (b) individually for each sample without computing the joint representation, and (c) jointly for all the samples without computing the joint representation. **Left:** On varying the size of the retrieval set by keeping feature dimensionality of both visual and textual features to be 100 ($p = q = 100$). **Right:** On varying the feature dimensionality (same for image/text samples) for a retrieval set containing 10^7 samples.

In Figure 5.4 (right), we compare the relative time required for ranking the samples in a synthetic retrieval set $\mathcal{T}'$ containing 10^7 samples for a single query. Here we vary the dimensionality of input and output modalities (same for both p and q) as $\{10, 20, 50, 100, 200\}$. These result into joint feature representations of dimensions $\{100, 400, 2.5K, 10K, 40K\}$ respectively. We consider the three situations (a), (b), and (c) as mentioned above. Here we observe that for all these three cases, the total time increases with feature dimensionality. However, in this case, the increments are not simply linear. For lower dimensions, they are sub-linear, while for higher dimensions, they are super-linear. For both (a) and (b), the total time taken is not practically appealing even for lower dimensional features. E.g., these are around 78.2 and 36.4 seconds for (a) and (b) respectively when $p = q = 10$. On the other hand, the total time for (c) using $p = q = 10$ and $p = q = 200$ are just around 1.3 and 2.3 seconds respectively. On discarding the time taken in sorting the elements after score computation (around 1.2 seconds), these become just around 0.1 and 1.1 seconds respectively.

From Figure 5.4, we can conclude that a direct (naïve) implementation could mar the efficiency of our approach during inference. However, using simple transformations that allow batch processing, it is

possible to achieve significant speed-ups, thus making it feasible to perform retrieval on large datasets containing millions of samples.

In our experiments, we compare the BITR approach with two baseline methods: CCA [40, 113] and WSABIE [153]. Comparing the run-time of BITR with CCA and WSABIE, we can easily observe that for all these three methods, in practice we need to project the features in the retrieval set just once and this can be done off-line. Now given a query, we can rank the samples by simply taking their dot product. Hence, the run-time of all the three methods becomes equivalent.

5.4 Image and Text Representations

We consider different types of representations for visual and textual data. These representations are compact, yet known to be effective in capturing data semantics. The first representation captures data characteristics in the form of probability distributions over unimodal topics. We refer to this as topic-based representation (TR). The second representation is based on learning cross-modal correlations between input and output modalities over TR. We refer to this as correlated topic-based representation (CTR). The third representation is based on modern CNN and word2vec features for images and text respectively.

It should be noted that since the complexity of learning a Structural SVM model depends on both the number of training samples (N) as well as the dimensionality of the joint feature representation ($r = p \times q$), in practice it is desirable to work with representations that are compact to maintain computational load. As discussed before, using compact representations for data is also practiced by other cross-modal search techniques such as [33, 112, 113]. Hence we adapt the representations accordingly to satisfy this requirement.

5.4.1 Topic-based Representation

This representation is based on unimodal probability distributions over topics, that are learned using Latent Dirichlet Allocation (LDA) model [11]. LDA is a popular probabilistic generative topic model and can effectively capture complex semantics of data in a compact manner. It considers a given document as a collection of discrete units/words. Based on co-occurrences of these words, it discovers high-level topics, and represents these in the form of multinomial distributions over words. Given a new document, LDA represents it as a probability distribution over the previously learned topics.

5.4.1.1 Representing Images

Since LDA requires each image to be represented as a collection of *words*, first we need to learn the visual words' vocabulary. For this, we randomly sample $0.5M$ SIFT descriptors [78] extracted densely at multiple scales from the training images of the SBU dataset [101], and learn 1000 visual words using the k-means algorithm. Each image is then represented as a bag-of-words histogram of these visual words. From this histogram based representation, the visual topics are learned using LDA by considering 5000 random (training) images from the SBU dataset.

Now, given a new image, first we extract SIFT descriptors densely at multiple scales, and represent it as a bag-of-words histogram of visual words as before. This is then used by LDA to construct a representation in the form of a probability distribution over the topics learned previously.

5.4.1.2 Representing Text

To learn textual topics, we use the captions in the training subset of the SBU dataset [101]. The vocabulary of words obtained from these captions (after simple pre-processing like removing stop-words) is used to represent the captions in the form of bag-of-words histograms, which are then used to learn textual topics using LDA. A new caption is represented as a bag-of-words histogram using the above vocabulary, which is then used to obtain a representation in the form of a probability distribution over the learned topics.

5.4.2 Correlated Topic-based Representation

In this representation, we incorporate cross-modal correlations into the topic-based representations for visual and textual data analogous to [113]. This is done by mapping the data into a maximally correlated vector subspace, that is learned using CCA [40]. This is based on the assumption that the samples coming from two different modalities contain some joint information that can be encoded using the correlations between them [40].

Note that while TR contains only non-negative (latent probability) values, CTR contains both positive as well as negative values. This is because it is obtained by projecting TR using a linear transformation learned through CCA, which projects an input vector into a maximally correlated real-valued vector space.

5.4.3 Modern Representations

Lately features computed using CNN for images [21, 31] and word2vec for text [88] have been popularly used in several tasks that deal with visual and textual data. Hence, we also evaluate using these features on the cross-modal image-caption retrieval task in Section 5.5.8. In practice, we compute features for images using a CNN model pre-trained on the ImageNet dataset [21] for image classification, that was shown to perform well for other visual recognition tasks as well. For captions, we use the pre-trained model of [88] by taking the average of vector representations of all the words in a caption.

5.5 Experiments

We demonstrate the applicability of our approach and extensively compare it with competing baseline methods.

5.5.1 Datasets

We consider three popular datasets in our experiments:

- **UIUC Pascal Sentence Dataset:** This was introduced in [111] and has become a de facto benchmark in the domain of image-caption understanding. It contains 1000 images, each of which is annotated with 5 captions from independent human-annotators.
- **IAPR TC-12 Benchmark:** This was introduced in [34] for the task of cross-language information retrieval. It has 19627 images, each of which is associated with a long description of up to 5 sentences.
- **SBU-Captioned Photo Dataset:** This was published in [101] and contains one million captioned images downloaded from Flickr. To our knowledge, this is the largest publicly available dataset of captioned photographs.

Table 5.1 shows some general statistics of these datasets and Figure 5.5 shows example images along with their ground-truth descriptions. For both Pascal and IAPR datasets, the captions/descriptions were written by guided human annotators. However, for the SBU dataset, the captions were written by the users who had uploaded those photographs on Flickr. Due to this, these captions are quite diverse and noisy. Moreover, they usually contain associated sentiments and abstract semantics that are not actually

Dataset	Samples	#Captions/Img.	Words/Caption
Pascal	1000	5	9.82 ± 3.51
IAPRTC-12	19627	1	24.98 ± 10.67
SBU	$1M$	1	12.14 ± 6.01

Table 5.1 Statistics of the three datasets used in our experiments. The last column shows the average number of words per caption.

(1) A bus drives down a busy road.
(2) A bus driving in a street.
(3) A passenger bus going down a busy city street.
(4) A white and red bus traveling on the street.
(5) Red and white bus driving down road.

A yellow building with white columns in the background. Two palm trees in front of the house. Cars are parking in front of the house. A woman and a child are walking over the square.

Little girl and her dog in northern Thailand. They both seemed interested in what we were doing.

Figure 5.5 Sample images with ground-truth captions from Pascal (left), IAPR (middle) and SBU (right) datasets.

visible in the image (e.g., see the third example in Figure 5.5). This makes the SBU dataset particularly challenging for cross-modal search task.

5.5.2 Evaluation Metrics

For performance evaluation, we consider two types of metrics that have been adopted by (1) image caption generation methods (such as [68,69,157]), and (2) image-caption retrieval methods (such as [43, 125]).

In the first setting, we consider BLEU [102] and Rouge [75] metrics for evaluation. BLEU is a precision based measure that is used to measure the performance of automatic translation of text from one language to another. For a given word (1-gram) with frequency (n_1) in the automatically translated sen-

tence, we find its maximum frequency among the reference sentences (n_{max}), then the BLEU score will be ($\frac{n_{max}}{n_1}$). This can be calculated for n-grams in similar manner. To generate BLEU score for a complete sentence, we take the geometric mean with a penalty such that higher scores for smaller n-grams are penalized. Rouge is a recall-based measure that is used to measure the performance of automatic summarization of text. It determines how well a system-generated summary covers the content present in one or more human-generated model summaries known as references, and encourages systems to include all the important topics in the text. Analogous to BLEU, it can be calculated for n-grams.

To compute BLEU scores, we use the code released by NIST (version-13a). To compute Rouge scores, we use Release-1.5.5[6]. During evaluation, the samples in the test set comprise the query set, and retrieval is performed on the full training set. For both Im2Text and Text2Im, we report mean one-gram BLEU and Rouge scores. For Im2Text, these scores are averaged over the top five retrieved captions, by matching them with the ground-truth caption of query image. For Text2Im, we compute these scores in an inverse manner; i.e., by matching the query caption with the ground-truth captions of the top five retrieved images. For both these metrics, higher score means better performance and vice-versa.

In the second setting, we consider Recall@K (R@K) and MedianRank (MedR) as the metrics for evaluation. For a given query, these are used to evaluate how correctly an approach can retrieve the true output (image/caption), assuming it to be present in the retrieval set. For Im2Text, this is performed by considering the images in the test set as queries, and performing retrieval over the captions in the test set. Similarly, for Text2Im, this is done by querying the captions in the test set and performing retrieval over the images in the test set. Recall@K measures for what percentage of queries, their correct output is present in the top K (K=50 in our case) retrieved samples. MedianRank measures the median of the retrieval ranks of the correct outputs corresponding to all the queries. For Recall@K, higher score means better performance, and for MedianRank, lower score means better performance.

5.5.3 Baselines for Comparisons

We compare the proposed approach with two popular baselines: WSABIE [153] and CCA [40, 113] in all the experiments. Both CCA and WSABIE learn separate projection matrices for input and output data. In practice, they both may converge to a lower dimensional projection space compared to the dimensionality of the input data without really affecting the performance. However in all our experiments, we project data into the same space for both these methods. This not only avoids information

[6]Obtained from `http://www.berouge.com/Pages/default.aspx`.

loss but also allows fair comparisons and avoids the need of tuning the optimal number of projections required by each. For CCA, we use normalized correlation in order to compute nearest-neighbour based similarity between two projected cross-modal features similar to [113].

5.5.4 Conceptual Comparison with CCA and WSABIE

CCA [40, 113] and WSABIE [153] are two well-known methods that can scale to large datasets and have been shown to work well for learning cross-modal associations. Here we present a conceptual comparison of these two with the proposed approach.

5.5.4.1 Comparison with CCA

CCA can be shown to minimize the squared Euclidean distance between pairs of samples from two modalities in the projected space [40, 86]. Let $\mathbf{U}$ and $\mathbf{V}$ denote the two projection matrices and $\mathbf{a}$ and $\mathbf{b}$ denote a pair of samples from the two modalities respectively. Thus, CCA can be seen to match the samples using the similarity function $exp(-||\mathbf{Ua} - \mathbf{Vb}||_2^2) = exp(\mathbf{a}^t(\mathbf{U}^t\mathbf{V})\mathbf{b})$. This maps to minimizing the loss $l(1, z) = -log(z)$ during training. We can observe that both CCA as well as BITR rely on bilateral scoring functions. An important difference is that while CCA makes use of only similar pairs of samples across modalities, BITR explicitly considers the dissimilar pairs during training. However, as discussed in Section 5.3.1, this in turn makes the training of BITR much slower than CCA. Second, while CCA decouples the two projection matrices and constrains each to be low rank, BITR learns a joint full rank parameter vector $\mathbf{w}$ and makes use of L_2 regularization on $\mathbf{w}$ to avoid overfitting. Third, as discussed above, our formulation can work with a variety of loss functions that suit a given cross-modal retrieval task.

5.5.4.2 Comparison with WSABIE

WSABIE was originally proposed for the task of label-ranking, and hence can not be directly applied to captions. For our comparisons, we thus modify the WSABIE algorithm, such that instead of learning a separate parameter vector for each label, it learns a single parameter matrix for all the captions. This is analogous to the parameter matrix learned for visual features in the WSABIE algorithm (details are provided in the Appendix at the end of the chapter). Similar to CCA and BITR, WSABIE also relies on a bilateral scoring function. However, unlike BITR and analogous to CCA, WSABIE decouples the

projection matrices for the two modalities, and constraints their individual norms without performing an explicit regularization. Second, during optimization, WSABIE considers any random (negative) sample that violates the margin condition to update the model, whereas BITR picks the sample corresponding to the most violated constraint (Eq. 5.8). This makes the training of WSABIE more scalable than BITR, however the model learned using BITR is more accurate than that using WSABIE (as also validated in the experimental analysis).

5.5.5 Implementation Details

- In all the experiments, each visual and textual sample is represented using a 100 dimensional feature vector for TR and CTR. Note that while the CCA baseline [40,44,113] projects the samples from both modalities into a common space (whose dimensionality is at most the minimum of the dimensionality of the input feature spaces), BITR does not require the features from both modalities to have the same dimensionality for cross-modal matching. However, we keep it the same for fair comparisons. Also, while the training time complexity of CCA is cubic in feature dimensionality, that for BITR is quadratic. Based on this, the chosen dimensionality was found to provide a good trade-off between efficiency and efficacy in the preliminary experiments.

- For BITR, we report results using the three loss functions given in Eq. 5.5, 5.6, and 5.7, and will refer to them as BITR-M, BITR-E, and BITR-C respectively.

- In all the experiments, the particular representation being employed will be denoted using "TR" or "CTR" wherever applicable.

- In all the experiments, the C parameter is tuned using five-fold cross-validation in the range $\{10^{-5}, 10^{-4}, \ldots, 10^{4}, 10^{5}\}$ for BITR and WSABIE.

5.5.6 Retrieval Schemes

We consider the following retrieval schemes in our experiments.

5.5.6.1 Experiment-1: Image-Caption Retrieval

We conduct this experiment on all the three datasets as described in Section 5.5.1.

(1) For the SBU dataset, we follow the train/test splits used in [101], which includes 500 test samples

and $999.5K$ training samples. For all the compared approaches, the model parameters are learned using a subset of 0.1 million samples randomly picked from the training data.
(2) For the other two datasets (IAPR and Pascal), we compute performance over all the samples as in [157]. This is done by creating ten partitions of the datasets. Each time, one partition is used for testing and the others for training. The final performance is computed by averaging the performance over all the splits.

5.5.6.2 Experiment-2: Cross-dataset Image-Caption Retrieval

In this experiment, we analyze the generalization ability of different cross-modal search methods across datasets. For this, instead of learning models for each dataset individually, we use the models learned using the SBU dataset in Experiment-1 and evaluate the performance on the IAPR and the Pascal datasets. For computing BLEU and Rouge scores, we consider as queries all the images from the Pascal or the IAPR dataset, and perform retrieval on all the captions of the SBU dataset for Im2Text. Similarly, for Text2Im, we consider as queries all the captions from the Pascal or the IAPR dataset, and perform retrieval on the full image collection of the SBU dataset. For computing Recall@K and MedianRank, we use the model learned using the SBU dataset, and perform retrieval over the samples in Pascal and IAPR datasets by partitioning them into ten splits as in Experiment-1 (for easy comparison with the results obtained in Experiment-1). The goal of this experiment is to study the effect of dataset specific biases in different methods, and also demonstrates the applicability of different methods on retrieval using large query sets (1000 for Pascal and 19627 for IAPR) and retrieval set (all one million samples of the SBU dataset).

5.5.7 Results and Discussion

5.5.7.1 Experiment-1: Image-Caption Retrieval

Table 5.2 and Table 5.3 compare the performance of different methods on all the datasets for both Im2Text and Text2Im. We can make the following observations from these results: (i) For all the four methods (i.e., WSABIE, BITR-M, BITR-E and BITR-C), the performance usually improves (sometimes by a large margin) by using CTR as compared to TR. This reflects the advantage of explicitly incorporating cross-correlations into data representations. (ii) For the Pascal dataset, relative performances of different methods follow almost similar trends for both Im2Text and Text2Im. However, there is com-

	Method	Im2Text				Text2Im			
		BLEU-1↑	Rouge-1↑	R@50↑	MedR↓	BLEU-1↑	Rouge-1↑	R@50↑	MedR↓
Pascal	CCA	**0.3149**	0.1397	**47.10**	**11.05**	**0.3254**	0.1432	**57.60**	**6.50**
	WSABIE (TR)	0.3077	0.1964	11.40	26.25	0.3135	0.1983	10.10	44.90
	WSABIE (CTR)	0.3119	**0.2172**	13.50	22.15	0.3102	**0.2154**	12.40	41.45
	BITR-M (TR)	0.3151	0.2234	41.20	14.05	0.3240	0.2201	10.90	45.35
	BITR-M (CTR)	0.3204	0.2374	42.10	15.35	0.3301	0.2416	55.00	7.45
	BITR-E (TR)	0.3286	0.2098	39.70	13.45	0.3404	0.2289	11.20	44.15
	BITR-E (CTR)	0.3380	0.2306	40.30	13.85	0.3491	0.2401	**57.20**	6.90
	BITR-C (TR)	0.3267	0.2275	46.50	11.30	0.3373	0.2315	10.60	44.55
	BITR-C (CTR)	**0.3485**	**0.2397**	**51.40**	**9.10**	**0.3489**	**0.2438**	56.80	**6.80**
IAPR	CCA	**0.2946**	**0.3031**	**16.32**	**404.35**	**0.3050**	**0.3016**	**18.53**	**301.45**
	WSABIE (TR)	0.2670	0.2375	2.73	932.10	0.2601	0.2416	2.73	999.10
	WSABIE (CTR)	0.2813	0.2497	4.76	772.30	0.2754	0.2700	3.79	798.95
	BITR-M (TR)	0.3172	0.3043	10.57	519.85	0.2858	0.2650	2.58	987.45
	BITR-M (CTR)	0.3227	0.3130	10.30	480.25	0.3028	0.2820	11.98	409.70
	BITR-E (TR)	0.3167	0.3047	10.68	493.30	0.2874	0.2646	2.63	979.45
	BITR-E (CTR)	0.3391	0.3240	12.41	416.85	0.3090	0.2901	**16.48**	**334.00**
	BITR-C (TR)	0.3219	0.3165	9.37	594.45	0.2963	0.2711	2.66	956.05
	BITR-C (CTR)	**0.3418**	**0.3281**	**13.78**	**335.95**	**0.3149**	**0.2966**	14.36	355.60

Table 5.2 Comparison of the performance using baseline methods (CCA [113] and WSABIE [153]) and variants of our method for image-caption retrieval on Pascal and IAPR datasets (Experiment-1). The best results using both are highlighted in bold. (↑: higher means better; ↓: lower means better.)

	Method	Im2Text				Text2Im			
		BLEU-1↑	Rouge-1↑	R@50↑	MedR↓	BLEU-1↑	Rouge-1↑	R@50↑	MedR↓
SBU	CCA	**0.1391**	**0.1147**	**16.20**	**189.50**	**0.1453**	0.1105	**19.80**	**190.00**
	WSABIE (TR)	0.0774	0.0664	8.40	254.50	0.1394	0.1159	10.60	246.50
	WSABIE (CTR)	0.1250	0.1043	11.60	237.00	0.1415	**0.1172**	11.80	232.00
	BITR-M (TR)	0.0986	0.0836	15.40	213.00	0.1427	0.1170	10.40	248.50
	BITR-M (CTR)	0.1401	0.1121	16.40	212.00	0.1592	0.1320	21.60	159.50
	BITR-E (TR)	0.1008	0.0858	13.20	209.00	0.1662	**0.1332**	10.60	249.50
	BITR-E (CTR)	0.1428	0.1138	19.20	195.00	0.1494	0.1161	21.40	181.50
	BITR-C (TR)	0.1468	**0.1182**	16.00	212.50	**0.1790**	0.1318	11.00	251.00
	BITR-C (CTR)	**0.1519**	0.1139	**24.60**	**144.50**	0.1782	0.1166	**25.20**	**149.00**

Table 5.3 Comparison of the performance using baseline methods (CCA [113] and WSABIE [153]) and variants of our method for image-caption retrieval on SBU dataset (Experiment-1). The best results using both are highlighted in bold. (↑: higher means better; ↓: lower means better.)

paratively more diversity in the other two datasets. This could be because the Pascal dataset is relatively much smaller than the other two datasets, and the diversity of semantic concepts it covers is also less. This may result in dataset specific biases, and thus reflects the necessity of evaluating on big and diverse datasets such as SBU. (iii) For most of the cases, BITR-C (CTR) outperforms the other two variants of BITR. This implies that a normalized correlation based loss function suits the cross-modal retrieval task better than the other two loss functions. (iv) The performance of BITR-C (CTR) is either better than or comparable to the CCA [113] approach throughout, thus indicating the superiority of the proposed Structural SVM based cross-modal search framework over the CCA technique.

5.5.7.2 Experiment-2: Cross-dataset Image-Caption Retrieval

Table 5.4 shows the results for this experiment. Here we can observe that: (i) For all the methods, the performance degrades significantly compared to that in Experiment-1. This reflects the impact of dataset specific biases, and thus emphasizes the necessity of performing cross-dataset evaluations. (ii) As in Experiment-1, BITR-C performs better than other methods in almost all the cases. This suggests

	Method	Im2Text				Text2Im			
		BLEU-1↑	Rouge-1↑	R@50↑	MedR↓	BLEU-1↑	Rouge-1↑	R@50↑	MedR↓
Pascal	CCA	0.2029	**0.1529**	**16.50**	**38.35**	0.2015	**0.1554**	**19.00**	**35.30**
	WSABIE (TR)	**0.2037**	0.1507	10.70	49.40	**0.2101**	0.1529	10.10	49.25
	WSABIE (CTR)	0.2010	0.1528	10.50	48.10	0.2081	0.1507	11.40	47.70
	BITR-M (TR)	0.2078	0.1573	13.80	42.45	0.2155	0.1574	10.30	49.25
	BITR-M (CTR)	0.2175	0.1704	15.50	42.50	0.2253	**0.1745**	22.30	29.85
	BITR-E (TR)	0.1907	0.1449	13.00	43.05	0.1989	0.1460	11.20	49.30
	BITR-E (CTR)	0.2140	0.1601	16.00	40.50	0.2176	0.1651	21.60	32.30
	BITR-C (TR)	0.2127	0.1598	14.80	41.55	0.2282	0.1546	10.10	49.35
	BITR-C (CTR)	**0.2217**	**0.1747**	**22.10**	**30.60**	**0.2376**	0.1741	**24.30**	**28.00**
IAPR	CCA	0.1460	0.1168	**6.30**	**706.20**	0.1468	0.1169	**7.11**	**641.50**
	WSABIE (TR)	**0.1495**	**0.1178**	2.32	988.75	0.1405	0.1098	2.72	1000.40
	WSABIE (CTR)	0.1471	0.1177	3.28	905.75	**0.1484**	**0.1185**	2.56	943.15
	BITR-M (TR)	0.1523	0.1275	4.50	816.90	0.1504	0.1253	2.50	976.55
	BITR-M (CTR)	0.1654	0.1418	4.97	768.85	0.1586	0.1344	7.52	564.85
	BITR-E (TR)	0.1284	0.0986	4.46	774.95	0.1258	0.0941	2.69	978.05
	BITR-E (CTR)	0.1529	0.1222	5.11	764.45	0.1312	0.1068	7.53	568.30
	BITR-C (TR)	0.1584	0.1364	4.72	805.30	0.1449	0.1148	2.70	970.15
	BITR-C (CTR)	**0.1719**	**0.1478**	**8.04**	**550.70**	**0.1676**	**0.1435**	**7.84**	**515.90**

Table 5.4 Comparison of the performance using baseline methods (CCA [113] and WSABIE [153]) and variants of our method for cross-dataset image-caption retrieval (Experiment-2). The best results using both are highlighted in bold. (↑: higher means better; ↓: lower means better.)

<table>
<tr><th colspan="2">Experiment-1</th><th colspan="2">Experiment-2</th></tr>
<tr><td>Im2Text</td><td>Text2Im</td><td>Im2Text</td><td>Text2Im</td></tr>
<tr>
<td>Query Image:
 
Output Text: A long three-storey building with a glass facade; a road with blue boards and a grey fence in the foreground and a blue sky in the background.</td>
<td>Query Text: A room with white walls, black door and window frames, a black carpet and three single beds made of wood with black and white bedcovers.
Output Image:
  </td>
<td>Query Image:
 
Output Text: People are sitting at a laid table made of wood in a little dark restaurant.</td>
<td>Query Text: Two local teachers are standing in a classroom with many children sitting at their wooden desks.
Output Image:
  </td>
</tr>
</table>

Figure 5.6 Qualitative results on IAPR TC-12 dataset for Im2Text and Text2Im. As we can observe, the results for "Im2Text" seem quite relevant to the respective query images. However, those for "Text2Im" are not really good and are only coarsely related to the respective query texts.

that the loss function $\Delta_C(\cdot)$ (Eq. 5.7) could be a better choice in practice than the other two loss functions $\Delta_M(\cdot)$ and $\Delta_E(\cdot)$ for real-world applications. (iii) Unlike Experiment-1, the relative gains using BITR-C compared to CCA [113] are now much more pronounced. This demonstrates the better generalization ability across datasets achieved using BITR than CCA.

5.5.7.3 Qualitative Results

Figure 5.6 shows some qualitative results on the IAPR dataset. We observe that our method is able to correctly identify specific objects such as "building", "bed", "table", etc. Also, it is quite interesting that for Im2Text in Experiment-2, the predicted caption is quite meaningful and representative of the image content even though it is from a different (SBU) dataset. This demonstrates the effectiveness of our approach in learning semantic relationships across the two modalities.

5.5.8 Evaluation Using Contemporary Features

Recently, image features computed using CNN models [21, 31, 159] have become the de facto standards for several visual recognition tasks. Similarly, textual features based on word2vec [88][7] are being popularly used in linguistic applications. While word2vec gives a 300-dimensional vector representation for text, many CNN models give a feature vector for images with a few thousand dimensions. E.g., [21, 31, 159] give a 4096-dimensional image representation. If we directly use these two representations in BITR, the dimensionality of the joint feature vector would become around 1.2 million ($= 4096 \times 300$), and this in turn would be computationally very expensive during training. However, some recent papers such as [5] have shown that it is possible to reduce significantly the size of image representation once they are learned, thus making our method compatible with CNN features. In [5], it was shown that applying dimensionality reduction using Principal Component Analysis (PCA) on the CNN features can provide a very short representation with almost no degradation in performance. Following [5], first we compute a 4096-dimensional CNN representation for images using the pre-trained model from [21], and then compress it to 128-dimensional vector using PCA. This, along with 300-dimensional textual feature vector computed using word2vec, gives a 38400-dimensional joint feature vector, thus making BITR compatible with these features.

We evaluate these features on the image-caption retrieval task as discussed under Experiment-1 (Section 5.5.6.1). Table 5.5 compares the performance of different methods. As compared to using simple bag-of-words based features (*c.f.* Table 5.2), the new features provide better performance for all the methods when we consider generation-based evaluation metrics BLEU and Rouge. Similarly, for retrieval-based evaluation metrics, the performance improves on all the datasets, except on Pascal where it degrades for MedianRank. This indicates that for small-scale datasets, now more number of relevant results come in top-K predictions, however their individual ranks go down. Analogous to the previous results, we can observe that: (i) BITR-C outperforms the other two variants of BITR in most of the cases, thus confirming the practical utility of normalized correlation based loss function. (ii) Also, the performance of BITR-C is either comparable to or better than CCA on all the three datasets. Overall, this experiment demonstrates the applicability of our approach in general, and validates that it can be used with modern CNN and word2vec features as well.

[7] `http://code.google.com/p/word2vec/`

	Method	Im2Text				Text2Im			
		BLEU-1↑	Rouge-1↑	R@50↑	MedR↓	BLEU-1↑	Rouge-1↑	R@50↑	MedR↓
Pascal	CCA	**0.3469**	0.1429	**72.80**	**21.25**	**0.3407**	0.1649	**85.10**	**20.65**
	WSABIE	0.3298	**0.2343**	35.40	28.30	0.3378	**0.2249**	36.90	33.45
	BITR-M	0.3466	0.2473	70.50	24.35	0.3502	0.2597	80.40	19.10
	BITR-E	0.3612	0.2578	74.30	19.60	0.3543	0.2637	84.70	21.15
	BITR-C	**0.3712**	**0.2602**	**75.10**	**18.75**	**0.3741**	**0.2681**	**85.20**	**18.15**
IAPR	CCA	**0.3163**	**0.3278**	**23.18**	**277.05**	**0.3279**	**0.3121**	**22.68**	**259.00**
	WSABIE	0.2964	0.2685	9.42	571.17	0.2840	0.2871	7.24	604.89
	BITR-M	0.3471	0.3398	19.36	261.75	0.3102	0.2943	21.68	304.85
	BITR-E	0.3568	0.3459	21.69	283.25	0.3195	0.3088	22.38	**262.35**
	BITR-C	**0.3672**	**0.3545**	**22.67**	**243.90**	**0.3361**	**0.3176**	**19.82**	423.55
SBU	CCA	**0.1452**	0.1237	**21.40**	**153.50**	**0.1681**	**0.1378**	**25.90**	**158.50**
	WSABIE	0.1395	**0.1280**	16.90	190.00	0.1635	0.1342	15.30	194.50
	BITR-M	0.1514	0.1366	23.70	164.00	0.1825	0.1526	24.30	207.50
	BITR-E	0.1628	**0.1519**	26.10	142.50	0.1903	0.1547	26.80	175.00
	BITR-C	**0.1723**	0.1492	**29.80**	**129.00**	**0.1974**	**0.1586**	**31.40**	**138.00**

Table 5.5 Comparison of the performance using baseline methods (CCA [113] and WSABIE [153]) and variants of our method for image↔caption retrieval (Experiment-1) using CNN based image features [5, 21] and word2vec [88] based textual features. The best results using both are highlighted in bold. (↑: higher means better; ↓: lower means better.)

5.5.9 Discussion

As we observed in the experiments, what features one uses will have a critical impact on the performance of BITR. Moreover, different combinations of features and loss functions may perform better than others for different problems. In the experiments, our primary motivation for performing feature transformation was to maintain computational load. In practice, it is possible to apply our method even if there is no higher level transformation at all. This is because we can form the joint representation Ψ by computing an outer product between any real-valued input and output feature vectors. Also, the three loss functions that we use are based on general distance/similarity metrics (Manhattan distance, Euclidean distance and cosine similarity). Each of these metrics are applicable to real-valued vectors. Only the cosine similarity based loss function (Eq. 5.7) makes an assumption that the feature vectors are L_2-normalized, and this normalization can be easily applied on a real-valued vector.

In a broader sense, our framework can be viewed as a support vector based counterpart of the nearest-neighbour based cross-modal matching techniques such as CCA [113]. The goal of both the techniques is to compute the similarity of a sample in one modality with that in another. Similar to the distance/similarity measures like Manhattan distance, Euclidean distance, and normalized correlation used in cross-modal matching [113], we have shown our framework to work with these measures by mapping them as loss functions of Structural SVM. This analogy is further evident from the fact that while the similarity metric based on normalized correlation was found to achieve the best performance in [113], similar results are observed in our experiments as well, where the BITR-C variant (that uses normalized correlation based loss function) mostly performs better than the other two variants of BITR. However, unlike the nearest-neighbour based method of [113], our approach usually provides better performance in both within-dataset (Experiment-1) as well as cross-dataset (Experiment-2) settings. This is because it is based on Structural SVM that provides good generalization and max-margin guarantees. Particularly in the cross-dataset experiments, our approach consistently outperformed CCA, sometimes significantly.

As discussed in Section 5.1.1, several techniques for cross-modal retrieval such as [33, 61, 112] are based on learning a tranformation of cross-modal input/output features. During inference, they usually adopt some simple similarity criteria such as cosine similarity in the transformed space. Our approach can serve as an improved inference technique for all such methods, where rather than using cosine similarity, one can learn a support vector model $\mathbf{w}$ over the transformed features and use it for inference.

Though this will add another layer of training, it will be a one-time process. Moreover, there will be almost no effect on the testing time as discussed in Section 5.3.2.

5.6 Summary

We have presented a novel Structural SVM based framework for cross-modal multimedia retrieval. We have demonstrated the applicability of our method to cross-modal search on two medium and one web-scale dataset. For both Im2Text and Text2Im, our method achieved promising results and outperformed competing baseline techniques. In our experiments, though we have considered visual (image) and textual data as the two modalities, the fundamental ideas discussed can be applied to cross-modal retrieval tasks in other domains as well.

A promising direction for future research would be to implement an efficient training algorithm for our approach that could scale to millions of samples with high-dimensional joint feature representations.

APPENDIX: Extending WSABIE for Captions

Here, first we briefly discuss the WSABIE algorithm [153], and then present the proposed extension of WSABIE to adapt it for captions.

WSABIE

WSABIE (Web Scale Annotation by Image Embedding) learns a mapping space where both images and annotations (e.g. labels) are represented. The mapping functions for both the modalities are learned jointly by minimizing the WARP (Weighted Approximate-Rank Pairwise) loss, that is based on optimizing precision at k. Each image is represented by $x \in \mathbb{R}^p$, and each annotation $i \in \mathcal{Y} = \{1, \ldots, Y\}$, where Y is the (fixed) vocabulary size. Then, a mapping is learned from image feature space to the joint space $\mathbb{R}^P$:

$$\Phi_I(x) : \mathbb{R}^p \rightarrow \mathbb{R}^P. \tag{5.16}$$

while jointly learning a mapping function for annotations:

$$\Phi_W(i) : \{1, \ldots, Y\} \rightarrow \mathbb{R}^P. \tag{5.17}$$

Both these mappings are chosen to be linear; i.e., $\Phi_I(x) = Vx$, and $\Phi_W(i) = W_i$ where W_i indexes the i^{th} column of a $P \times Y$ matrix. The goal is to learn the possible annotations of a given image such that

Require: labeled data $(x_i, y_i), y_i \in \{1, \ldots, Y\}$

repeat

Pick a random labeled example (x_i, y_i)

Let $f_{y_i}(x_i) = \Phi_W(y_i)^T \Phi_I(x_i)$

Set $N = 0$

repeat

Pick a random annotation $\bar{y} \in \{1, \ldots, Y\} \setminus y_i$.

Let $f_{\bar{y}}(x_i) = \Phi_W(\bar{y})^T \Phi_I(x_i)$

$N = N + 1$

until $f_{\bar{y}}(x_i) > f_{y_i}(x_i) - 1$ or $N \geq Y - 1$

if $f_{\bar{y}} > f_{y_i}(x_i) - 1$ **then**

Make a gradient step to minimize:

$$L(\lfloor \tfrac{Y-1}{N} \rfloor)|1 - f_y(x_i) + f_{\bar{y}}(x_i)|_+$$

Project weights to enforce constraints in Eq. 5.19.

end if

until validation error does not improve.

Algorithm 1: WSABIE Algorithm

the highest ranked ones best describe the semantic content of the image. For this, the following model is considered:

$$f_i(x) = \Phi_W(i)^T \Phi_I(x) = W_i^T V x, \tag{5.18}$$

where the possible annotations i are ranked according to the magnitude of $f_i(x)$ in descending order. This family of models have constrained norm:

$$\begin{aligned} ||V_i||_2 &\leq \lambda, i = 1, \ldots, p, \\ ||W_i||_2 &\leq \lambda, i = 1, \ldots, Y. \end{aligned} \tag{5.19}$$

which acts as a regularizer. Algorithm 1 shows the pseudo-code for learning model variables using a stochastic gradient descent algorithm that minimizes WARP loss (where $L(k) = \sum_{j=1}^{k} \alpha_j$, with $\alpha_j = \frac{1}{j}$).

Require: labeled data (x_i, c_i), y is a feature vector representing caption $c \in \mathcal{C}$

repeat

Pick a random labeled example (x_i, c_i)

Let $g_{y_i}(x_i) = \Phi_Z(y_i)^T \Phi_I(x_i)$

Set $N = 0$

repeat

Pick a random caption $\bar{c} \in \mathcal{C} \setminus c_i$.

Let $g_{\bar{y}}(x_i) = \Phi_Z(\bar{y})^T \Phi_I(x_i)$

$N = N + 1$

until $g_{\bar{y}}(x_i) > g_{y_i}(x_i) - 1$ or $N \geq |\mathcal{C}| - 1$

if $g_{\bar{y}} > g_{y_i}(x_i) - 1$ **then**

Make a gradient step to minimize:

$$L(\lfloor \frac{|\mathcal{C}|-1}{N} \rfloor)|1 - g_y(x_i) + g_{\bar{y}}(x_i)|_+$$

Project weights to enforce constraints in Eq. 5.22.

end if

until validation error does not improve.

Algorithm 2: Adapted WSABIE Algorithm for Captions

Adapting WSABIE for Captions

In case of captions, we have a (training) set of captions $\mathcal{C} = \{c_i\}$ rather than a fixed annotation vocabulary. In order to adapt WSABIE for captions, we modify the feature mapping given in Eq. 5.17 such that instead of learning a separate parameter vector for each annotation, we learn a single parameter matrix for all the captions. Given a caption $c \in \mathcal{C}$ represented by $y \in \mathbb{R}^q$, a mapping is learned from caption feature space to the joint space $\mathbb{R}^P$:

$$\Phi_Z(y) : \mathbb{R}^q \rightarrow \mathbb{R}^P, \tag{5.20}$$

where Z is a $P \times q$ matrix. Now, given a set of captions, the goal is to learn the possible caption(s) of a given image such that the highest ranked one(s) best describe the semantic content of the image. For this, the following model is considered:

$$g_y(x) = \Phi_Z(y)^T \Phi_I(x) = y^T Z^T V x. \tag{5.21}$$

Similar to Eq. 5.19, this family of models have constrained norm:

$$||V_i||_2 \leq \lambda, i = 1, \ldots, p,$$
$$||Z_i||_2 \leq \lambda, i = 1, \ldots, q. \tag{5.22}$$

which acts as a regularizer. Algorithm 2 shows the pseudo-code for learning the model variables using a stochastic gradient descent algorithm. It is similar to Algorithm 1 except that instead of randomly picking an annotation from vocabulary, now we randomly pick a caption from the training set consisting of image-caption pairs.

Chapter 6

Cross-Specificity: Modelling Data Semantics for Cross-Modal Matching and Retrieval

6.1 Introduction

In the previous chapter, we considered the problem of cross-modal retrieval between two diverse modalities. One of the challenges while dealing with multiple modalities is that of the inherent heterogeneity among samples within a modality as well as across different modalities. This can be partly addressed by grouping samples based on some higher-level semantics such as their category [33, 112]. The category of a sample plays a central role in expressing its underlying semantics. Moreover, if two different modalities are known to share a common set of categories, this can be useful in understanding their mutual semantics with respect to each other. E.g., consider the collection of images and captions in Figure 6.1. Just by having a quick look, we can observe that samples in both the sets depict the same semantic category, i.e., "cat".

Motivated by this, we try to leverage this shared information to model cross-modality semantics of a sample. For this, we introduce the notion of cross-specificity. Given collections of samples from two different modalities that share a common set of semantic categories, cross-specificity measures how well a sample in one modality portrays its category relative to samples that belong to the same category in another modality. A sample with high cross-specificity score is considered to be specific with respect to its semantically similar samples in another modality, while that with low score is considered to be comparatively ambiguous. Modelling this association can benefit a variety of applications that involve multiple modalities. One such example is cross-modal retrieval, where given a query in one modality, the goal is to retrieve semantically relevant samples from another modality. In this task, if a sample is known to be ambiguous in depicting its underlying semantic category with respect to samples in another

- A black and brown cat looks up with green eyes.
- A striped cat asleep and curled up.
- Grey and white cat lying on wood.
- A white and black cat laying down outdoors.
- A longed-haired cat with it's eyes closed.
- A cat laying on a surface upside down looks at the camera.

Figure 6.1 Though the samples in these collections belong to two different modalities (image and text), they depict the same semantic category, i.e., "cat". We introduce "cross-specificity" that harnesses this mutual information to model ambiguity in the content of a sample in one modality *relative* to its semantically similar samples in another modality.

modality (i.e., has low cross-specificity), an explicit boosting mechanism can be used to enhance its predictability. However, if a sample is specific, we may not require such boosting.

Given a sample from a particular category/class in one modality, we propose to measure its cross-specificity score using two mechanisms: one based on human judgement of its similarity with samples that belong to the same class in another modality, and the other using an automatic similarity measure. We then demonstrate how cross-specificity can benefit the cross-modal retrieval task. Experiments show that though simple, it helps in achieving consistent improvements over several baseline techniques. We also analyze different aspects of cross-specificity such as correlations between human and automated cross-specificity measurements, its relationship with the underlying semantic category, and the influence of the size of training data.

6.1.1 Related work

After going through the previous chapters, we are now convinced that given the increasing amount of data in the form of multiple modalities, there has been considerable interest in leveraging this data to learn richer models for various applications rather than relying upon individual modalities independently. This has been found to be particularly useful in modelling visual data, where additional cues in the form of textual tags, captions, GPS coordinates, etc. can provide great amount of semantically valuable information. Examples of this are work on multi-modal and cross-modal modelling, image

description and retrieval, and understanding image properties. While we have discussed in detail the related developments in these areas in the previous chapters, here we will again briefly touch upon some of these to provide a coherent picture particularly contrasting with the notion of cross-specificity.

Multi-modal and Cross-modal modelling: Several papers study multiple modalities to address a particular task. E.g., Guillaumin *et al.* [36] and Li *et al.* [74] use tags as additional features for learning image classification models, Rohrbach *et al.* [115] use textual descriptions for human activity recognition in videos, McAuley and Leskovec [83] model pairwise relations between images using relational meta-data in form of social connections, etc. In [22], Duan *et al.* proposed a constrained clustering approach for multi-modal data that allows missing features in any modality channel. In cross-modal analysis, one needs to learn a function that can measure similarity between a pair of samples from diverse modalities [33,40,43,48,112,113,119,142]. In [40,43,113], canonical correlation analysis (CCA) is used for learning a common embedding space using pairs of samples from two modalities. Recent methods such as [33, 112] additionally use semantic information which helps in learning discriminative cross-modal matching functions. In another paper [48], Hwang and Grauman learn a matching function using images and tags ranked by human subjects, such that it respects the relative ordering of tags and the relative importance of objects. Lately, some approaches based on deep neural networks [63, 125] have also been proposed for learning associations between images and text.

While most of these papers focus on *matching* a pair of samples from two distinct modalities based on some cross-modal matching function, our first contribution is to propose a novel *property* of samples in cross-modality data – cross-specificity – that captures the degree of specificity in the content of a sample in one modality relative to its semantically similar samples in another modality. As our second contribution, we also demonstrate how it can benefit several existing cross-modal retrieval approaches.

Image description and retrieval: Both image description as well as image retrieval using text are well-researched topics. While several papers have focused on associating images with discrete labels [7, 28, 35, 80, 120], describing and retrieving images using complete captions has lately gained popularity [27,43,62,64,68,69,72,89,101,113,142,146,156]. Rather than focusing on the task of either describing or retrieving images, through cross-specificity, we try to study a property of how well the content of an image (text) matches with its semantically similar textual (image) samples. This in turn can benefit both image description as well as retrieval approaches. E.g., to describe a query image, we may consider only those captions that are relatively more specific, and during image retrieval, we may devise explicit mechanisms to improve the predictability of images that are identified to be comparatively ambiguous.

Image properties: There have also been several attempts to study image properties such as object saliency [49,58], likelihood of a certain object being mentioned in its description [10,127], and variance in human perception in perceiving and describing an image [51]. Unlike these, our focus is on estimating properties of a sample based on its semantically similar samples from another modality.

The idea of cross-specificity is closely related to the recent paper of Jas and Parikh [51]. This paper introduced the notion of "image specificity", that estimates the degree of specificity in the visual content of an image based on similarity among different captions describing that particular image. We also aim at estimating the specificity in the content of a sample, however we do this in a relative manner based on similarity among samples across different modalities that share common semantics. Moreover, unlike [51], we do not assume any particular modality, and this makes cross-specificity a more general notion. We will provide a detailed conceptual comparison between cross-specificity and image specificity in Section 6.2.3.

6.2 Measuring Cross-Specificity

Let S_X and S_Y be sets of samples from two modalities $\mathcal{M}_X$ and $\mathcal{M}_Y$ respectively (*e.g.*, images and text). We are also given a set of classes $\mathcal{C}$ such that each sample in both the modalities is associated with a class $c \in \mathcal{C}$. We define the cross-specificity of a sample from a particular class as its average pairwise similarity with all the samples from the same class in the other modality. E.g., let $x \in S_X$ be a sample from class c, and $Y_c \subset S_Y$ be the subset of all the samples that belong to the same class in the other modality. To compute cross-specificity of x, we compute its pairwise similarity with all the samples $y \in Y_c$ and average the scores. The similarity between x and y can either be graded by humans or computed automatically.

6.2.1 Human Cross-Specificity Measurement

For this, N different human subjects are asked to rate the similarity between a pair of samples x and $y \in Y_c$, on a likert scale of 1 (very dissimilar) to 10 (very similar). In our study, subjects were not informed that the two samples belonged to the same semantic category, which ensured that they rated the similarity solely based on their perceived content. These scores are then normalized to lie in $[0, 1]$.

Let $sim^n_{\text{hum}}(x, y)$ denote similarity between x and $y \in Y_c$ as perceived by the n-th subject, and $spec_{\text{hum}}(x)$ be the human cross-specificity score for x. Then we get

$$spec_{\text{hum}}(x) = \frac{1}{N|Y_c|} \sum_{y \in Y_c} \sum_{n=1}^{N} sim^n_{hum}(x, y) \tag{6.1}$$

6.2.2 Automated Cross-Specificity Measurement

In order to measure cross-specificity automatically, we would need to compute similarity between a pair of samples from distinct modalities. We pose this as a cross-modal matching task [33,40,112,113]. Given sets of samples from two different modalities, cross-modal matching approaches model and learn a function $\mathcal{F}$ that can measure similarity between a pair of samples from diverse modalities. While different approaches define $\mathcal{F}$ in different ways, the underlying assumption is that it should maintain the semantic proximity of data intact; i.e., samples with similar content should have higher similarity scores and vice-versa.

Let us assume we are given a function $\mathcal{F}$ learned using a cross-modal matching technique (such as [33, 40, 112, 113, 142]). Based on this, let $sim_{\text{auto}}(x, y)$ denote the similarity score between x and $y \in Y_c$ computed using $\mathcal{F}$, normalized to lie between 0 and 1. The automated cross-specificity score $spec_{\text{auto}}(x)$ for x is then obtained by averaging these similarity scores across all (x, y) pairs. That is,

$$spec_{\text{auto}}(x) = \frac{1}{|Y_c|} \sum_{y \in Y_c} sim_{auto}(x, y) \tag{6.2}$$

Analogous to Eq. 6.1 and 6.2, we can compute $spec_{\text{human}}(y)$ and $spec_{\text{auto}}(y)$ for some $y \in S_Y$. Note that both $spec_{\text{hum}}(\cdot)$ as well as $spec_{\text{auto}}(\cdot)$ lie in $[0, 1]$.

Figure 6.2 shows example images from the PASCAL-50S dataset [51] along with their human-annotated and automatically computed cross-specificity scores. We can observe that though each of these images contains the corresponding ground-truth category object, the cross-specificity scores using both the measures reduce as the perceptibility of ground-truth category becomes ambiguous. E.g., in the third and fourth images, their categories "bicycle" (present in between the two persons) and "potted-plant" (placed on the platform in the background) respectively are perceptually small and ambiguous.

6.2.3 Comparison with Image Specificity

As discussed in Section 6.1.1, the concept of image specificity [51] allows us to measure the degree of specificity in the content of an *image*, that relies on the linguistic similarity *among* multiple captions

Figure 6.2 Example images with high to low human-annotated and automatically computed cross-specificity scores, along with their ground-truth category. Note that from left to right, since the category becomes more and more ambiguous, and hence cross-specificity scores (both human and automatic) also reduce

of that image (i.e., text-to-text matching). The aim of cross-specificity is also to estimate the degree of specificity in the content of a sample. However, here this is measured based on the semantic category of that sample, and is defined in terms of *its* similarity with samples from *another* modality (i.e., cross-modal matching) that belong to the same category. More importantly, unlike in image specificity, cross-specificity does not require explicit coupling/mapping among samples; e.g., images and captions shown in Figure 6.1 are not coupled, though they depict the same category. Below we list some of the critical distinctions between image specificity [51] and the proposed cross-specificity[1]:

- To compute image specificity, we require each image to be associated with multiple (at least two) captions. However, this can be too much to expect from the real-world data. In contrary, cross-specificity requires just a single category label associated with a given sample. This requirement is independent of the particular modality under consideration, and is much more relaxed and practically feasible than the former.

- The way image specificity is defined restricts it to a single data point (that consists of an image and all its captions), and thus disconnected from the rest of the data points. On the other hand,

[1]The reader is suggested to refer Section 3.1 of [51] to get further details on the notion of image specificity, and to better appreciate its distinctions with the proposed concept of cross-specificity.

cross-specificity of a sample relies on several other cross-modality samples that belong to the same cateogry. Since a category is not bound to a single sample, cross-specificity is expected to encode data abstraction much better than image specificity.

- As shown in [51], the application of image specificity is limited to the task of textual query based image retrieval. Whereas, the notion of cross-specificity is generic, does not make any assumption on particular modality, and can benefit cross-modal retrieval between any two modalities as discussed next.

6.3 Application of Cross-Specificity to Cross-Modal Retrieval

We now describe how cross-specificity can be used in a cross-modal retrieval task.

6.3.1 Set-up

Let $\mathcal{T}_X^r \subset S_X$ and $\mathcal{T}_Y^r \subset S_Y$ denote the collection of training samples, and $\mathcal{T}_X^e \subset S_X$ and $\mathcal{T}_Y^e \subset S_Y$ be the collection of samples in the retrieval set, such that $\mathcal{T}_X^r \cap \mathcal{T}_X^e = \mathcal{T}_Y^r \cap \mathcal{T}_Y^e = \varnothing$. The training samples are used to learn a cross-modal matching function using some cross-modal learning technique such as [33, 40, 112, 113, 142]. During evaluation, the samples in the two retrieval sets are matched using the learned function, considering samples in one modality as the query set and doing retrieval over samples in the other modality. In cross-modal retrieval, given a query q from the retrieval set of one modality (say $\mathcal{T}_X^e$), the goal is to rank the samples from another modality ($\mathcal{T}_Y^e$) based on their relevance with the query, such that the samples with more relevance should be ranked higher and vice-versa. Without loss of generality, from now onwards we assume the query $q \in \mathcal{T}_X^e$ (the query set), and retrieval is performed over samples in $\mathcal{T}_Y^e$ during the testing phase.

6.3.2 Baseline Approach

Here, we learn a cross-modal matching function using a baseline cross-modal learning approach [33, 40, 112, 113, 142]. Using the learned function, we compute similarity $sim_{\text{auto}}(q, y)$ between q and $y \in \mathcal{T}_Y^e$ that denotes the baseline relevance rel_{baseline}^y between q and y:

$$rel_{\text{baseline}}^y = sim_{\text{auto}}(q, y) \tag{6.3}$$

Finally, all the samples in $\mathcal{T}_Y^e$ are ranked (sorted) in descending order of this relevance score.

6.3.3 Proposed Approach

In the proposed approach, we additionally take into consideration the cross-specificity of a sample rather than ranking based on just the similarity with the query. The rationale is if the underlying (unknown) semantic category of the query is the same as that of the (known) semantic category of a sample in the retrieval set, then that sample is semantically relevant to the query and hence should be ranked higher, and vice-versa.

To do so, rather than sorting just based on $sim_{\text{auto}}(q, y)$ as done in the baseline approach, we model $P(\text{match}|sim_{\text{auto}}(q, y))$ which captures the probability that the query matches a sample. To this end, we model this probability using Logistic Regression (LR) as below:

$$rel^{y}_{\text{cross-spec}} \quad = P(\text{match}|sim_{\text{auto}}(q, y)) = (1 + exp(-\beta^{y}_{0} - \beta^{y}_{1} sim_{\text{auto}}(q, y)))^{-1} \tag{6.4}$$

This model is trained separately for each sample in the retrieval set $\mathcal{T}^{e}_{Y}$. Assume a sample $y \in \mathcal{T}^{e}_{Y}$ that belongs to class c, and let $X^{r}_{c} \subset \mathcal{T}^{r}_{X}$ be the subset of samples in the training set that are from the same class. For y, we consider positive examples as the similarity scores between y and all $x \in X^{r}_{c}$, and negative examples as the similarity scores between y and some random $|X^{r}_{c}|$ number of samples from $\mathcal{T}^{r}_{X} \setminus X^{r}_{c}$ (i.e., the samples that do not belong to class c). Using these positive and negative examples, the LR model for the given y is trained. In real-world applications, since the retrieval is usually performed on a fixed and known set, this makes learning LRs a one-time process and thus can be done off-line.

The parameters of the LR model (β^{y}_{0} and β^{y}_{1}) inherently capture the cross-specificity of each sample y. This is because for a sample, LR tries to boost its relevance with samples from the same class and suppresses it for others. Once we train a separate LR model for each $y \in \mathcal{T}^{e}_{Y}$, it is used to compute $P(\text{match}|sim_{\text{auto}}(q, y))$ for that sample given a query. Finally, all the samples are sorted based on the probability outputs of the LR models.[2]

6.4 Experiments

Now we analyze different aspects of cross-specificity, and evaluate the proposed cross-specificity based cross-modal retrieval approach.

[2]Note that for the query sample, we do not know its category, and the query-independent LR models are trained only for the samples in the retrieval set.

6.4.1 Datasets and Features

We experiment on two publicly available datasets, viz. Wikipedia and PASCAL-50S. The Wikipedia dataset was compiled by Rasiwasia et al. [113] from the Wikipedia articles, and is widely used as a de facto benchmark in cross-modal retrieval. It consists of 2173 train and 693 test pairs of images and text articles from 10 different classes. The PASCAL-50S dataset was introduced by Jas and Parikh [51], and is an extended version of the UIUC PASCAL Sentence dataset [111]. It contains 1000 images from 20 different classes, each of which is captioned with 50 unique captions. In our experiments, we consider only the last caption (the "query" caption from [51]) for each image in order to maintain balance between the two modalities, and a random split of $750/250$ train/test samples.

For both the datasets, we use 4096-dimensional image features computed using the last layer of the VGG-16 model [121] pre-trained on the ImageNet dataset. For text, we use a 10D topic representation learned using Latent Dirichlet allocation model (LDA) [11].[3]

6.4.2 Performing Cross-modal Matching

To compute automatic cross-specificity (Section 6.2.2), we consider five cross-modal matching techniques: Canonical Correlation Analysis (CCA) [40], Semantic Matching (SM) [113], Cluster Canonical Correlation Analysis (C3A) [112], Multiview approach (MVA) [33] and Bilateral Image-Text Retrieval (BITR) [142]. CCA learns a common embedding space between cross-modality samples by maximizing their correlation. Because of its effectiveness, CCA is considered as a de facto benchmark in cross-modal matching tasks. In SM, a sample is represented using a $\mathcal{C}$-dimensional (i.e., number of categories) feature vector. Each dimension of this vector denotes its relevance with a particular category, that is computed using a classifier. This in turn allows direct matching between cross-modality samples. C3A and MVA additionally make use of semantic information during the training phase, which helps in learning discriminative cross-modal matching functions. In BITR (Chapter 5), a Structural SVM based framework is used for performing cross-modal retrieval.

While computing cross-specificity of an image, we need to compute its similarity with text samples from the same class. We will refer to it as "image-to-text" matching or "I2T". Similarly, while computing cross-specificity of a text sample, we need to compute its similarity with images from the same class. We will refer to it as "text-to-image" matching or "T2I".

[3] In our preliminary experiments, we also tried using the word2vec representation [88] for text, however it was found to give slightly inferior performance than the topic-based representation.

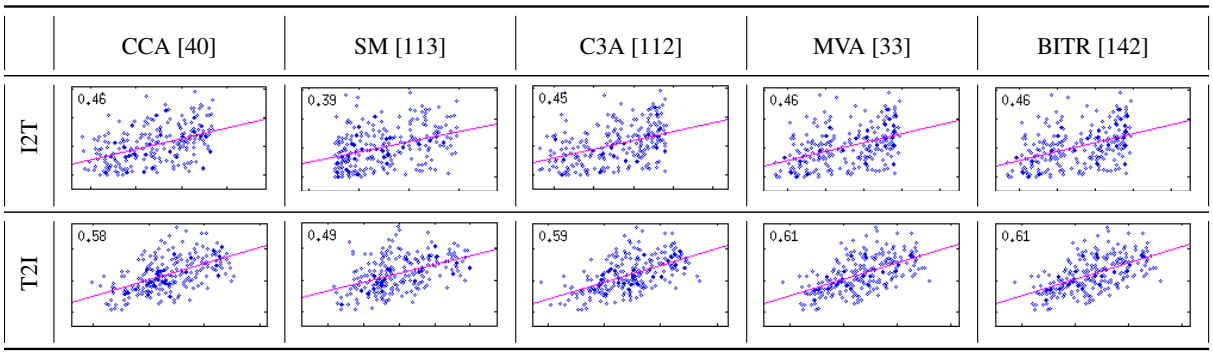


Figure 6.3 Correlation between human cross-specificity and automatic cross-specificity computed using different methods for I2T and T2I. On top of each distribution we show the Spearman's rank correlation score.

6.4.3 Consistency Analysis

As described in Section 6.2, cross-specificity of a sample can be measured using two mechanisms. In the first, humans rate the similarity between pairs of cross-modality samples and in the second, an automatic method is used. We collect human measurements on the PASCAL-50S dataset. For I2T, we collect similarity ratings of each image in the retrieval set with all the captions in the training set that belong to the same class. Similarly, for T2I, we collect similarity ratings between each caption in the retrieval set with all the images in the training set from the same class. This gives 9170 pairs for each I2T and T2I. For each pair, we collected judgments from 2 human subjects.

Figure 6.3 shows the Spearman's rank correlation. between human-annotated and automatically measured cross-specificity using the five cross-modal matching methods (CCA, SM, C3A, MVA and BITR). Overall, the correlation scores lie between 0.39 to 0.47 for I2T and 0.49 to 0.63 for T2I[4]. Note that even though the matching is performed between two diverse sources of information (image and text) and is inherently quite subjective, we achieve statistically significant positive correlations between the two mechanisms. These results confirm that cross-specificity is a well-defined phenomenon.

In Figure 6.4, we study how cross-specificity scores vary across samples. The wide range of cross-specificity values using both human as well as automatic measurements denote that some samples do have more cross-specificity than others. Also, while the range of cross-specificity values using human and automatic measurements are different, they mostly follow a similar pattern. As discussed in Sec-

[4]All the correlations were found to be statistically significant at $p < 0.0001$.

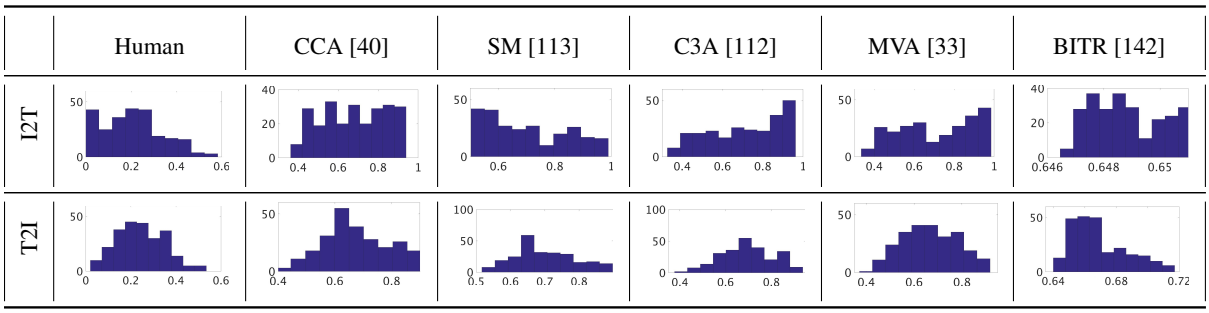


Figure 6.4 Distribution of human and automated cross-specificity scores computed using different methods for the PASCAL-50S dataset. The samples (vertical axis) are grouped with respect to their cross-specificity scores (horizontal axis), with each bin indicating the number of samples whose cross-specificity scores fall in that range.

tion 6.3.3, this variation can be employed to improve cross-modal retrieval performance. Note that automatic computation of cross-specificity suits real-world applications better than human annotation. Hence, automated cross-specificity measurement may be the more relevant criteria.

6.4.4 Cross-modal Retrieval

6.4.4.1 Baselines

We consider the five methods discussed in Section 6.4.2 as our baselines for cross-modal retrieval. Following these approaches, we consider two cross-modal retrieval tasks: retrieving text samples given a query image, and retrieving images given a query text. For convenience, we do a slight abuse of notation, and will refer to these tasks as "I2T" and "T2I" respectively. To denote the proposed approach that integrates cross-specificity with baseline cross-modal techniques (Section 6.3.3), we use "(CS)" as a suffix. To measure cross-modal retrieval performance, we use mean average precision (mAP) as the evaluation metric.

6.4.4.2 Results

Table 6.1 shows the results for cross-modal retrieval obtained using the baseline approach (Section 6.3.2) and those using the proposed approach based on cross-specificity (Section 6.3.3). From these results, we observe that the proposed approach performs significantly better than the baselines in most

		Method →	CCA [40]	SM [113]	C3A [112]	MVA [33]	BITR [142]
Image-to-Text (I2T)	Wikipedia	Baseline	41.72	41.47	41.71	31.49	43.03
		Cross-Spec	46.57	44.81	46.12	35.92	44.55
	Pascal-50S	Baseline	25.40	22.14	26.52	24.02	26.17
		Cross-Spec	35.40	32.36	37.98	34.60	31.60
Text-to-Image (T2I)	Wikipedia	Baseline	40.04	28.34	39.85	28.34	40.53
		Cross-Spec	50.29	41.93	50.67	41.93	45.52
	Pascal-50S	Baseline	29.53	26.12	32.59	29.03	30.49
		Cross-Spec	35.53	34.91	37.41	34.36	32.71

Table 6.1 Cross-modal retrieval results using different methods: CCA [40], SM [113], C3A [112], MVA [33] and BITR [142]. The "Baseline" row denotes the performance (percentage mean Average Precision) obtained using the baseline methods. The "Cross-spec" row denotes the performance obtained using the proposed cross-specificity based approach.

of the cases. In some cases, it achieves up to $10 - 12\%$ of improvements. These results clearly validate the utility of modelling cross-specificity in cross-modal retrieval tasks. In Figure 6.5, we analyze what percentage of queries benefit the most from the proposed approach. The horizontal axis denotes the percentage of queries, and the vertical axis denotes the margin range by which the baseline is beaten: the first bin denotes the percentage of queries where baseline is beaten by $\leq 5\%$ mAP, the second bin denotes the percentage of queries where baseline is beaten by $> 5\%$ and $\leq 10\%$ mAP, and so on till the last bin that denotes the percentage of queries where baseline is beaten by $> 35\%$ mAP. From these results, we observe that using the proposed approach, around $12 - 30\%$ of queries achieve up to 5% of improvement, and around $2 - 10\%$ of queries achieve more than 35% of improvement in mAP. Also, for T2I, the improvements are more spread-out over the different margin ranges compared to I2T for the Wikipedia dataset. Recall that in the Wikipedia dataset, each text sample is a long article and contains a lot of ambiguous information. By modelling this ambiguity using cross-specificity, we are able to

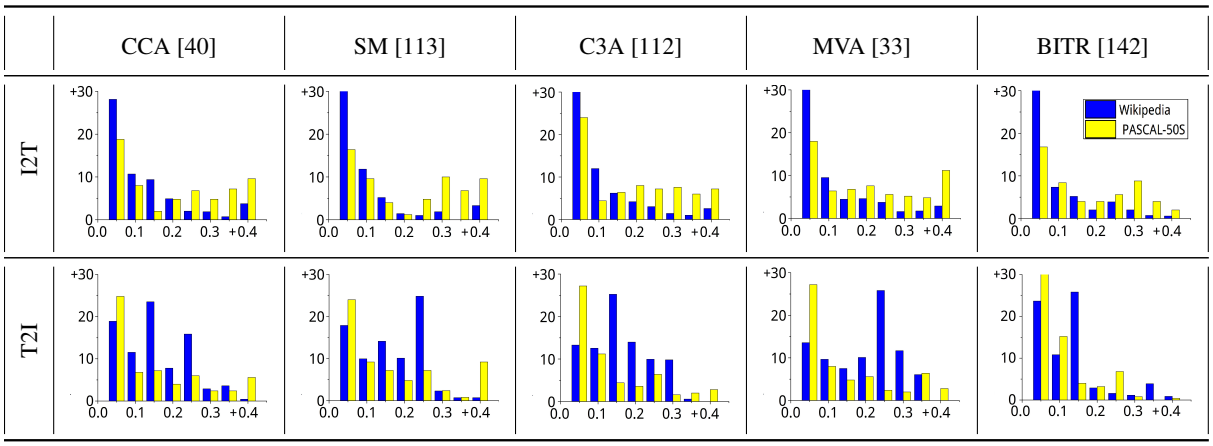


Figure 6.5 Cross-modal retrieval results for I2T and T2I using different methods. The horizontal-axis denotes the margin (in terms of mAP) by which baseline is beaten, and the vertical-axis denotes the percentage of queries where baseline is beaten. See text for details.

retrieve semantically more relevant images compared to the baselines. This is also validated from the results in Table 6.1 where cross-specificity achieves improvements for $70 - 85\%$ of queries.

In Figure 6.6, we investigate the effect of number of training pairs per category used for learning cross-modal matching function. Note that we do not make any change while evaluating the baseline techniques for the ease of comparison. We can observe that on increasing the number of training pairs, the performance generally improves. This is expected since more training pairs help in learning a richer cross-modal matching function. In comparison to the baselines that use the full training data, cross-specificity usually helps to achieve either better or comparable performance by using just $\sim 10\%$ of the training data.

In Figure 6.7, we show the average cross-specificity values for individual categories in the PASCAL-50S dataset using automatic measurements. Here we observe that the scores for categories such as "potted-plant", "chair" and "bottle" are generally low, whereas those for categories such as "cow", "train", "boat", "aeroplane", "bus" and "cat" are generally high. This is justifiable because in this dataset, the former objects are usually not the central component in their images/captions, and occupy only a small portion. This makes them less specific, and hence they give low cross-specificity scores. However, the latter ones are usually quite prominent and unambiguous, thus achieving higher cross-specificity scores.

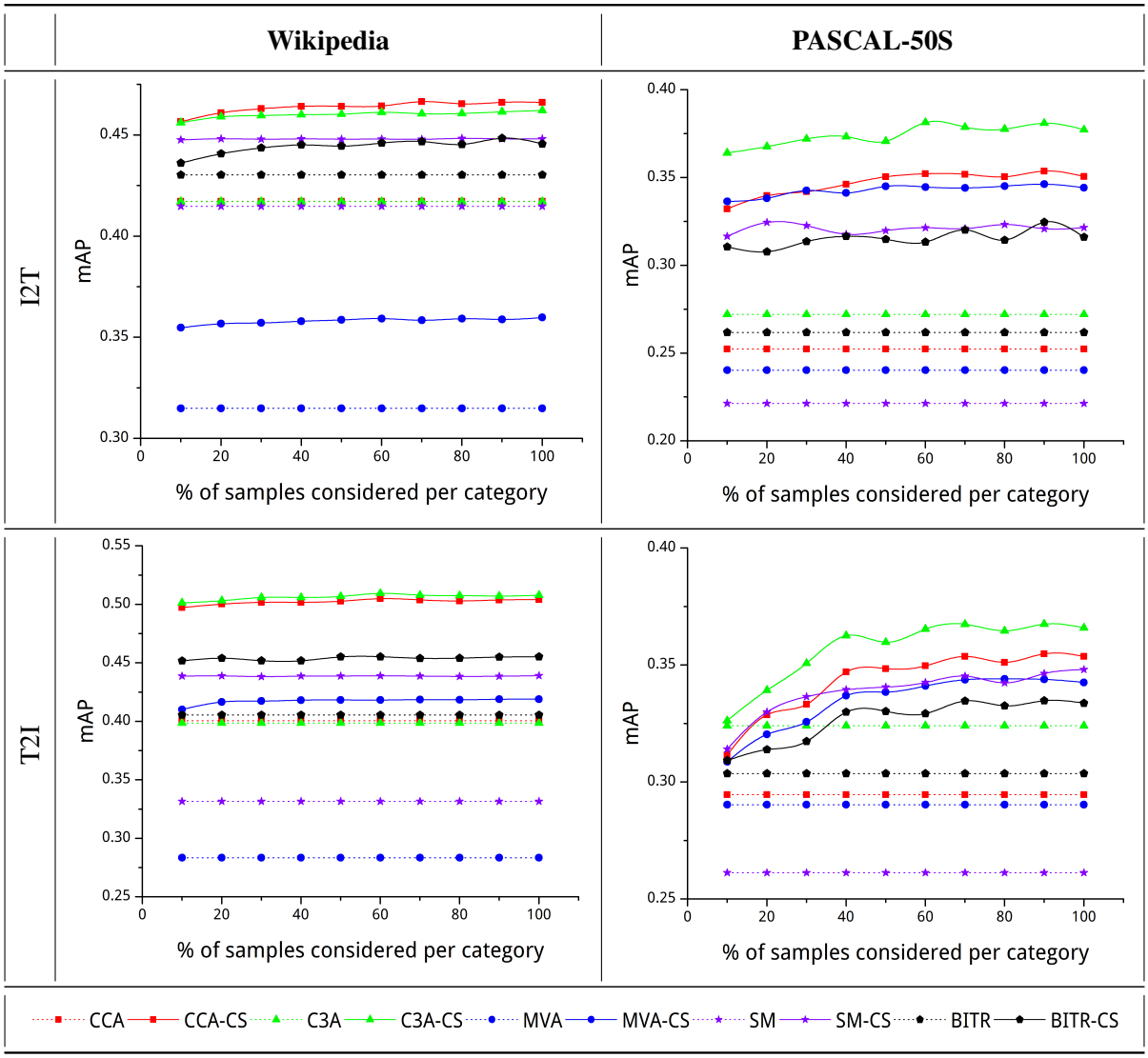


Figure 6.6 Analysis of the proposed cross-specificity based cross-modal retrieval compared to baselines by varying the percentage of training samples per category used for computing cross-specificity of each test sample.

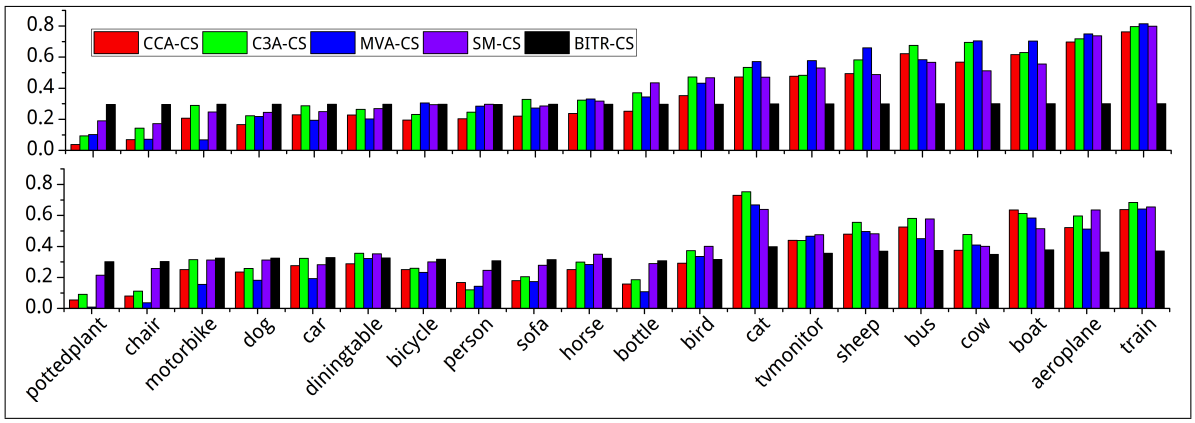


Figure 6.7 Average cross-specificity scores per category for I2T (top) and T2I (bottom) for the PASCAL-50S dataset

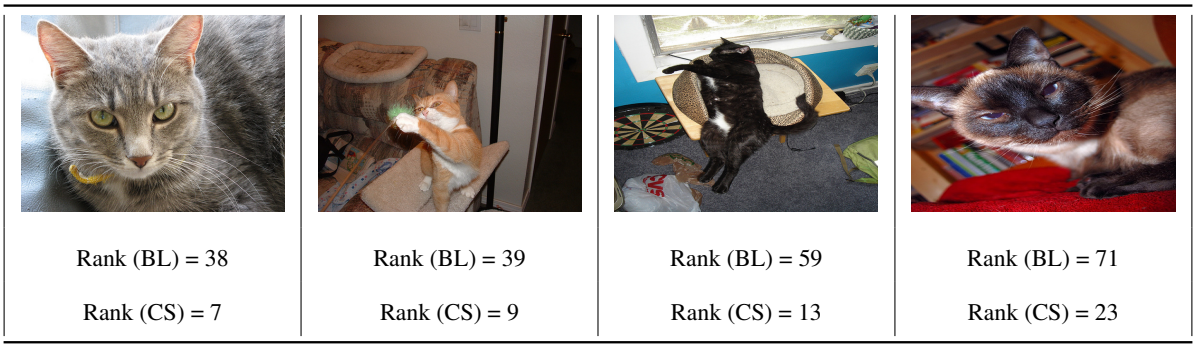


Figure 6.8 Qualitative cross-modal retrieval result for a query text "A cat posing for picture." from the PASCAL-50S dataset. All the four images are semantically relevant to the query (i.e., belong to the same category "bicycle"). With each image, we show its rank using baseline (BL) approach and the proposed cross-specificity (CS) based approach. Note how cross-specificity helps in improving the rank of all the images

Query	Ground-truth caption	Rank (BL)	Rank (CS)
	**The television is on top of the silver stand.**	**181**	**63**
	**A hummingbird with a blue head and long tail perches on a twig.**	**240**	**74**
	**A few pieces of furniture sit in a room.**	**2**	**9**
	**A train to Trenton is stopped at a station.**	**56**	**61**

Figure 6.9 Some examples for I2T where cross-specificity (CS) improves (the first two rows) and does not improve (the last two rows) the retrieval rank of the ground-truth caption in comparison to baseline (BL) approach. Here, CCA is considered as the baseline approach.

Figure 6.8 shows an example from the PASCAL-50S dataset for T2I where cross-specificity helps in improving the rank of semantically relevant images for a given query. Note that the first image is the actual image corresponding to that query, where the rank improves from 38 to 7. Also note that in the third image, the category "cat" is perceptually ambiguous. Here also, cross-specificity helps in improving its retrieval rank from 59 to 13. Figure 6.9 and 6.10 show additional positive and negative examples for I2T and T2I respectively. In practice, we observed that cross-specificity helps in improving the retrieval rank in most of the cases, sometimes by a big margin. In cases where the retrieval rank degrades, it is mostly by a relatively smaller margin. These results qualitatively confirm the practical advantages of cross-specificity in improving cross-modal retrieval performance.

Query	Ground-truth image	Rank (BL)	Rank (CS)
A hooded person repairs their upturned bike.		**146**	**90**
A small statue holds a burning candle.		**90**	**41**
A small statue holds a burning candle.		**38**	**51**
A large cruise ship docked to a loading dock.		**14**	**25**

Figure 6.10 Some examples for T2I where cross-specificity (CS) improves (the first two rows) and does not improve (the last two rows) the retrieval rank of the ground-truth caption in comparison to baseline (BL) approach. Here, CCA is considered as the baseline approach.

6.5 Summary

We have introduced the notion of cross-specificity, and showed that it is a well-defined phenomenon. We studied various aspects of cross-specificity, and demonstrated its applicability on cross-modal retrieval task. Experiments showed that the proposed approach can provide significant improvements in the performance compared to several baseline techniques.

Presently, the notion of cross-specificity is suitable for single-category cross-modal data. In the future, we plan to generalize it for multi-category (or multi-label) data, that would involve additional challenges such as correlation among categories and weak labelling [28, 35, 80].

Chapter 7

Summary, Conclusions and Future Directions

We now summarize this thesis, derive the conclusions, and discuss some possible directions for future research.

7.1 Summary

We started by motivating the reader about the general problem of automatically learning semantic associations between visual and textual data, why it is an important problem to address, and what are the challenges involved. While this is a very wide domain with several connected sub-problems and applications, we limited our focus on the core problems of semantically associating natural scene images with different forms of texts such as labels, phrases, and captions. After introducing and defining the problems of our interest, we briefly discussed some machine learning based techniques in that helped us in coming-up with new solutions for these.

In the subsequent chapters, we presented our approaches for addressing these problems. In Chapter 3, we presented two approaches for the problem of image annotation. In our first approach, we built upon the idea of propagating labels from visually similar (or neighbouring training) images and tried to address the challenges of class-imbalance and incomplete-labelling. For this, we introduced an additional pre-step of forming semantic groups of images based on label information, and then identifying a fixed number of neighbours from each given a test image. This additional step helped in balancing the frequencies of labels in the neighbourhood and ensured the presence of at least a minimum number of samples corresponding to each label. In our second approach, first we studied the utility of binary (one-versus-rest) SVM on multi-label image annotation task, and tried to analyze why it is difficult to learn good label-specific models. We identified that there are at least three reasons responsible for this,

namely incomplete-labelling, label-ambiguity, and structural overlap. To partly address these, we introduced a novel tolerance parameter in the conventional hinge-loss of SVM. Given a label, this parameter is computed for each sample automatically based on dataset statistics and is used in regulating the impact of error on that sample on the learned model. In Chapter 4, we extended the idea of propagating discrete labels from visually similar images to that of propagating textual phrases extracted from available data, and presented a simple approach for generating fluent and human-like captions for images. We also showed how the same idea could be adopted for doing image retrieval using caption-based queries. In Chapter 5, we presented a novel approach based on Structural SVM for cross-modal retrieval. We showed how the popular similarity/distance metrics used in unimodal data (such as Manhattan and Euclidean distance, and cosine similarity), and a simple representation computed using the outer product of feature vectors of a pair of cross-modality samples could be efficiently modelled as the loss functions and joint feature representation in Structural SVM respectively. Finally, in Chapter 6, we introduced a novel term called "cross-specificity" that tries to encode the degree of categorical semantics in a sample in cross-modal data. It measures the average similarity of a sample in one modality with all the samples from the same category in the other modality. Based on good correlation between human-evaluated and automatically computed cross-specificity scores, we showed that it is a well-defined phenomenon. To demonstrate an application of cross-specificity, we also presented a simple logistic regression based technique that can be used as a wrapper over a number of cross-modal matching models and helps in consistently improving their performance on cross-modal retrieval task.

7.2 Conclusions

While the previous decade witnessed the success of the Internet, the current one is about efficiently organizing and extracting semantically meaningful information from the ever-growing raw digital data, with a major portion of this being in the form of images and videos. We hope that this thesis would help in better understanding and appreciating various challenges involved in the general problem of semantic understanding of visual data, and in driving future efforts in this direction.

As a side remark, we would like to draw the attention of the reader to the fact, which probably would have become obvious by now, that all our approaches for different tasks are quite easy to grasp, and mostly based on simple (dis)similarity based measures. Though simple, they were able to achieve considerable improvements over several baseline/competing techniques, with some of them being rather

complex. Through this, we would like to reiterate the well-known belief that even for complex problems, as complex as the ones we have focused on, it is usually helpful to design, analyze and evaluate simple solutions first to start with. These help in not only establishing baselines in many cases and give a basic know-how about the problem, but sometimes can also perform much better than expected.

7.3 Future Directions

Below, we briefly discuss few promising directions for future research based on this thesis:

- To design improved representations for visual data using multimodal data: We saw in Chapter 3 and 5 that representations learned by integrating visual data with additional sources of information are semantically richer than unimodal representations. We examined one such technique, (kernelized) CCA, that tries to address this task by learning aligned subspaces for different modalities, favouring cross-correlations against modality-specific information. However, the subspace alignment obtained using CCA and similar methods facilitates only coarse mappings. This motivates to develop new multi-modal representation learning techniques that address such limitations, and learn more powerful and discriminative representations [131]. Moreover, these can be further enhanced for the cases where data is incomplete/sparse (i.e., entities in some of the modalities are missing).

- Generating richer descriptions for domain-specific visual data: Most of the existing techniques in this domain have focused on generating free-form descriptions for images/videos in the wild. Due to this, the generated descriptions usually read quite generic and coarse. A natural extension to this would be to generate detailed descriptions focusing on fine-grained details [76, 129]. This in turn would demand to work with domain-specific data and models.

- Generating linguistic descriptions for visual data in Indian Languages: Almost all of the existing approaches for generating textual descriptions (be it labels, phrases or captions) for visual data have focused on the English language. In the context of Indian languages, there has not been any attempt in this direction as per our knowledge. It would be interesting to evaluate existing techniques for the same, and develop new ones specifically suiting Indian languages and context. Moreover, since several Indian languages are usually resource-crunch, this will also require to

explore domain adaptation techniques through which it would be relatively easy to adapt existing methods for English for Indian languages with little data.

- Building human-computer interface systems based on fusion of language and vision: Apart from addressing the core problems in visual+textual domain, it would be interesting to extend/adapt existing techniques towards building novel applications dealing with such data. One such example is to build language based interface for educational and instructional videos. This will have utility in indexing and searching such videos. E.g., this would facilitate viewers to directly jump to a specific point in an educational video (such as video lectures) where some specific concept is explained. Similarly, one could build a dynamic interface to explain the procedure for different activities in a step-by-step manner; e.g., how to make a wooden chair or how to fit a ceiling fan.

Bibliography

[1] C. Anderson. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion, 2006.

[2] G. Andrew, R. Arora, J. Bilmes, and K. Livescu. Deep canonical correlation analysis. In *ICML*, 2013.

[3] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual question answering. In *ICCV*, 2015.

[4] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.

[5] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014.

[6] L. Ballan, T. Uricchio, L. Seidenari, and A. D. Bimbo. A cross-media model for automatic image annotation. In *Proc. ICMR*, 2014.

[7] K. Barnard and D. Forsyth. Learning the semantics of words and pictures. In *ICCV*, 2001.

[8] A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709, 2013.

[9] A. Berg, J. Deng, and L. Fei-Fei. ImageNet large scale visual recognition challenge 2012. 2012.

[10] A. C. Berg, T. L. Berg, H. Daumé, J. Dodge, A. Goyal, X. Han, A. Mensch, M. Mitchell, A. Sood, K. Stratos, and K. Yamaguchi. Understanding and predicting importance in images. In *CVPR*, 2012.

[11] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *JMLR*, 2003.

[12] S. S. Bucak, R. Jin, and A. K. Jain. Multi-label learning with incomplete class assignments. In *Proc. CVPR*, pages 2801–2808, 2011.

[13] J. C. Caicedo, J. BenAbdallah, F. A. González, and O. Nasraoui. Multimodal representation, indexing, automated annotation and retrieval of image collections via non-negative matrix factorization. *Neurocomput.*, 76(1):50–60, 2012.

[14] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3):394–410, 2007.

[15] M. Chen, A. Zheng, and K. Q. Weinberger. Fast image tagging. In *Proc. ICML*, 2013.

[16] N. Cristianini and J. Shawe-Taylor. *An introduction to support Vector Machines: and other kernel-based learning methods*. Cambridge University Press, New York, NY, USA, 2000.

[17] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[18] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):5:1–5:60, 2008.

[19] M.-C. de Marneffe and C. D. Manning. The stanford typed dependencies representation. In *COLING Workshop*, 2008.

[20] J. Devlin, H. Cheng, H. Fang, S. Gupta, L. Deng, X. He, G. Zweig, and M. Mitchell. Language models for image captioning: The quirks and what works. In *ACL*, 2015.

[21] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. *Proc. ICML*, 2014.

[22] K. Duan, D. Crandall, and D. Batra. Multimodal learning in loosely-organized web images. In *CVPR*, 2014.

[23] K. Duan, D. J. Crandall, and D. Batra. Multimodal learning in loosely-organized web images. In *CVPR*, 2014.

[24] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, pages 97–112, 2002.

[25] H. J. Escalante, C. A. Hérnadez, L. E. Sucar, and M. Montes. Late fusion of heterogeneous methods for multimedia image retrieval. In *MIR*, 2008.

[26] H. Fang, S. Gupta, F. Iandola, R. Srivastava, L. Deng, P. Dollar, J. Gao, X. He, M. Mitchell, J. Platt, L. Zitnick, and G. Zweig. From captions to visual concepts and back. In *CVPR*, 2015.

[27] A. Farhadi, M. Hejrati, A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences for images. In *ECCV*, 2010.

[28] S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple bernoulli relevance models for image and video annotation. In *Proc. CVPR*, pages 1002–1009, 2004.

[29] H. Fu, Q. Zhang, and G. Qiu. Random forest for image annotation. In *Proc. ECCV*, pages 86–99, 2012.

[30] A. Gatt and E. Reiter. Simplenlg: A realisation engine for practical applications. In *ENLG*, 2009.

[31] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[32] Y. Goldberg and O. Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722, 2014.

[33] Y. Gong, Q. Ke, M. Isard, and S. Lazebnik. A multi-view embedding space for modeling internet images, tags, and their semantics. *Int. J. Comput. Vision*, 106(2):210–233, 2014.

[34] M. Grubinger. *Analysis and Evaluation of Visual Information Systems Performance.* PhD thesis, Victoria University, Melbourne, Australia, 2007.

[35] M. Guillaumin, T. Mensink, J. J. Verbeek, and C. Schmid. TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *Proc. ICCV*, pages 309–316, 2009.

[36] M. Guillaumin, J. Verbeek, and C. Schmid. Multimodal semi-supervised learning for image classification. In *CVPR*, 2010.

[37] A. Gupta. Generating descriptions for images. Master's thesis, IIIT Hyderabad, Nov. 2013.

[38] A. Gupta and L. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *ECCV*, 2008.

[39] A. Gupta, Y. Verma, and C. V. Jawahar. Choosing linguistics over vision to describe images. In *AAAI*, 2012.

[40] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16(12):2639–2664, 2004.

[41] B. Hariharan, L. Zelnik-Manor, S. V. N. Vishwanathan, and M. Varma. Large scale max-margin multi-label classification with priors. In *ICML*, 2010.

[42] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[43] M. Hodosh, P. Young, and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *JAIR*, 2013.

[44] H. Hotelling. Relations between two sets of variates. *Biometrika*, 28:321–377, 1936.

[45] H. Hu, G.-T. Zhou, Z. Deng, and Z. L. andGreg Mori. Learning structured inference neural networks with label relations. In *CVPR*, 2016.

[46] M. J. Huiskes and M. S. Lew. The MIR Flickr retrieval evaluation. In *MIR*, 2008.

[47] S. J. Hwang and K. Grauman. Learning the relative importance of objects from tagged images for retrieval and cross-modal search. *Int. J. Comput. Vision*, 100(2):134–153, 2012.

[48] S. J. Hwang and K. Grauman. Learning the relative importance of objects from tagged images for retrieval and cross-modal search. *Int. J. Comput. Vision*, 100(2):134–153, 2012.

[49] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(11):1254–1259, 1998.

[50] V. Jain and M. Varma. Learning to re-rank: Query-dependent image re-ranking using click data. In *WWW*, 2010.

[51] M. Jas and D. Parikh. Image specificity. *CVPR*, 2015.

[52] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proc. CVPR*, pages 3304–3311, 2010.

[53] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *Proc. ACM SIGIR*, pages 119–126, 2003.

[54] J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *ICRCL*, 1997.

[55] R. Jin, S. Wang, and Z.-H. Zhou. Learning a distance metric from multi-instance multi-label data. In *Proc. CVPR*, pages 896–902, 2009.

[56] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, 2002.

[57] J. Johnson, L. Ballan, and L. Fei-Fei. Love thy neighbors: Image annotation by exploiting image metadata. In *ICCV*, 2015.

[58] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *ICCV*, 2009.

[59] M. Juneja, A. Vedaldi, C. V. Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *CVPR*, 2013.

[60] M. M. Kalayeh, H. Idrees, and M. Shah. NMF-KNN: Image annotation using weighted multi-view non-negative matrix factorization. In *CVPR*, 2014.

[61] C. Kang, S. Xiang, S. Liao, C. Xu, and C. Pan. Learning consistent feature representation for cross-modal multimedia retrieval. *IEEE Transactions on Multimedia*, 17(3):370–381, 2015.

[62] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[63] A. Karpathy, A. Joulin, and L. Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *NIPS*, 2014.

[64] A. Karpathy, A. Joulin, and L. Fei-Fei. Deep fragment embeddings for bidirectional image-sentence mapping. In *NIPS*, 2014.

[65] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014.

[66] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *TACL*, 2015.

[67] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

[68] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating simple image descriptions. In *Proc. CVPR*, pages 1601–1608, 2011.

[69] P. Kuznetsova, V. Ordonez, A. C. Berg, T. L. Berg, and Y. Choi. Collective generation of natural image descriptions. In *ACL*, 2012.

[70] V. Lavrenko, R. Manmatha, and J. Jeon. A model for learning the semantics of pictures. In *NIPS*, 2003.

[71] J. Li. A mutual semantic endorsement approach to image retrieval and context provision. In *MIR*, 2005.

[72] S. Li, G. Kulkarni, T. L. Berg, A. C. Berg, and Y. Choi. Composing simple image descriptions using web-scale n-grams. In *CoNLL*, 2011.

[73] X. Li, C. G. M. Snoek, and M. Worring. Learning social tag relevance by neighbor voting. *IEEE Transactions on Multimedia*, 11(7):1310–1322, 2009.

[74] Y. Li, D. Crandall, and D. Huttenlocher. Landmark classication in large-scale image collections. In *ICCV*, 2009.

[75] C.-Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *NAACLHLT*, 2003.

[76] D. Lin, C. Kong, S. Fidler, and R. Urtasun. Generating multi-sentence lingual descriptions of indoor scenes. In *BMVC*, 2015.

[77] J. Liu, M. Li, Q. Liu, H. Lu, and S. Ma. Image annotation via graph learning. *Pattern Recogn.*, 42(2):218–228, 2009.

[78] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[79] A. Makadia, V. Pavlovic, and S. Kumar. A new baseline for image annotation. In *Proc. ECCV*, pages 316–329, 2008.

[80] A. Makadia, V. Pavlovic, and S. Kumar. Baselines for image annotation. *Int. J. Comput. Vision*, 90(1):88–105, 2010.

[81] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP natural language processing toolkit. In *ACL: System Demonstrations*, 2014.

[82] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille. Explain images with multimodal recurrent neural networks. In *NIPS Deep Learning Workshop*, 2014.

[83] J. J. McAuley and J. Leskovec. Image labeling on a network: Using social-network metadata for image classification. In *ECCV*, 2012.

[84] C. Meadow, B. Boyce, D. Kraft, and C. Barry. Text information retrieval systems. *Emerald Group Pub Ltd*, 2007.

[85] T. Mei, Y. Rui, S. Li, and Q. Tian. Multimedia search reranking: A literature survey. *ACM Comput. Surv.*, 46(3):38:1–38:38, 2014.

[86] A. K. Menon, D. Surian, and S. Chawla. Cross-modal retrieval: A pairwise classification approach. In *SIAM International Conference on Data Mining*, 2015.

[87] D. Metzler and R. Manmatha. An inference network approach to image retrieval. In *Proc. CIVR*, pages 42–50, 2004.

[88] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[89] M. Mitchell, J. Dodge, A. Goyal, K. Yamaguchi, K. Sratos, X. Han, A. Mensch, A. C. Berg, T. L. Berg, and H. D. III. Midge: Generating image descriptions from computer vision detections . In *EACL*, 2012.

[90] S. Moran and V. Lavrenko. Optimal tag sets for automatic image annotation. In *Proc. BMVC*, pages 1.1–1.11, 2011.

[91] S. Moran and V. Lavrenko. A sparse kernel relevance model for automatic image annotation. *International Journal of Multimedia Information Retrieval*, pages 1–21, 2014.

[92] Y. Mori, H. Takahashi, and R. Oka. Image-to-word transformation based on dividing and vector quantizing images with words. In *MISRM'99 First International Workshop on Multimedia Intelligent Storage and Retrieval Management*, 1999.

[93] K. Morik, P. Brockhausen, and T. Joachims. Combining statistical learning with a knowledge-based approach - A case study in intensive care monitoring. In *ICML*, 1999.

[94] V. N. Murthy, E. F. Can, and R. Manmatha. A hybrid model for automatic image annotation. In *Proc. ICMR*, 2014.

[95] V. N. Murthy, S. Maji, and R. Manmatha. Automatic image annotation using deep learning representations. In *Proc. ICMR*, 2015.

[96] V. N. Murthy, S. Maji, and R. Manmatha. Automatic image annotation using deep learning representations. In *ICMR*, 2015.

[97] V. N. Murthy, A. Sharma, V. Chari, and R. Manmatha. Image annotation using multi-scale hypergraph heat diffusion framework. In *ICMR*, 2016.

[98] H. Nakayama. *Linear distance metric Learning for large-scale generic image recognition.* PhD thesis, The University of Tokyo, Japan, 2011.

[99] Z. Niu, G. Hua, X. Gao, and Q. Tian. Semi-supervised relational topic model for weakly annotated image recognition in social media. In *CVPR*, 2014.

[100] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int. J. Comput. Vision*, 42(3):145–175, 2001.

[101] V. Ordonez, G. Kulkarni, and T. L. Berg. Im2Text: Describing images using 1 million captioned photographs. In *NIPS*, 2011.

[102] K. Papineni, S. Roukos, T. Ward, and W. Zhu. BLEU: A method for automatic evaluation of machine translation. In *ACL*, 2002.

[103] S. Parameswaran and K. Q. Weinberger. Large margin multi-task metric learning. In *NIPS*, pages 1867–1875, 2010.

[104] M. Paramita, M. Sanderson, and P. Clough. Diversity in photo retrieval: overview of the ImageCLEFPhoto task 2009. *CLEF working notes*, 2009.

[105] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, 2011.

[106] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *Proc. ECCV*, pages 143–156, 2010.

[107] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. *CVPR*, 2007.

[108] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, 2000.

[109] B. Plummer, L. Wang, C. Cervantes, J. Caicedo, J. Hockenmaier, and S. Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, 2015.

[110] V. Ranjan, N. Rasiwasia, and C. V. Jawahar. Multi-label cross-modal retrieval. In *ICCV*, 2015.

[111] C. Rashtchian, P. Young, M. Hodosh, and J. Hockenmaier. Collective image annotation using amazon's mechanical turk. In *NAACLHLT Workshop*, 2010.

[112] N. Rasiwasia, D. Mahajan, V. Mahadevan, and G. Aggarwal. Cluster canonical correlation analysis. In *AISTATS*, 2014.

[113] N. Rasiwasia, J. C. Pereira, E. Coviello, G. Doyle, G. R. G. Lanckriet, R. Levy, and N. Vasconcelos. A new approach to cross-modal multimedia retrieval. In *ACM MM*, 2010.

[114] J. Rodriguez and F. Perronnin. Label embedding for text recognition. In *BMVC*, 2013.

[115] M. Rohrbach, M. Regneri, M. Andriluka, S. Amin, M. Pinkal, and B. Schiele. Script data for attribute-based recognition of composite activities. In *ECCV*, 2012.

[116] R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. In *SLSFS*, 2006.

[117] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, 2011.

[118] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proc. ICML*, pages 807–814, 2007.

[119] A. Sharma, A. Kumar, H. D. III, and D. W. Jacobs. Generalized multiview analysis: A discriminative latent space. In *CVPR*, 2012.

[120] B. Siddiquie, R. S. Feris, and L. S. Davis. Image ranking and retrieval based on multi-attribute queries. In *CVPR*, 2011.

[121] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[122] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, 2012.

[123] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *MIR: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, 2006.

[124] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *PAMI*, 22(12):1349–1380, 2000.

[125] R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng. Grounded compositional semantics for finding and describing images with sentences. *TACL*, 2013.

[126] Q. Song, W. Hu, and W. Xie. Robust support vector machine with bullet hole image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(4):440–448, 2002.

[127] M. Spain and P. Perona. Measuring and predicting object importance. *Int. J. Comput. Vision*, 91(1):59–76, 2011.

[128] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In *NIPS*, 2012.

[129] M. Sukhwani and C. V. Jawahar. Tennisvid2text: Fine-grained descriptions for domain specific videos. In *BMVC*, 2015.

[130] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Comput.*, 12(6):1247–1283, 2000.

[131] T. Q. N. Tran, H. L. Borgne, and M. Crucianu. Aggregating image and text quantized correlated components. In *CVPR*, 2016.

[132] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004.

[133] T. Uricchio, L. Ballan, L. Seidenari, and A. D. Bimbo. Automatic image annotation via label transfer in the semantic space. *CoRR*, abs/1605.04770, 2016.

[134] Y. Ushiku, T. Harada, and Y. Kuniyoshi. Automatic sentence generation from images. In *ACM MM*, 2011.

[135] J. van de Weijer and C. Schmid. Coloring local feature extraction. In *Proc. ECCV*, pages 334–348, 2006.

[136] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[137] A. Vedaldi and B. Fulkerson. Vlfeat – an open and portable library of computer vision algorithms. In *ACM Multimedia*, 2010. `http://www.vlfeat.org/`.

[138] J. Verbeek, M. Guillaumin, T. Mensink, and C. Schmid. Image Annotation with TagProp on the MIR-FLICKR set. In *MIR*, 2010.

[139] Y. Verma, A. Gupta, P. Mannem, and C. V. Jawahar. Generating image descriptions using semantic similarities in the output space. In *CVPR Workshop*, 2013.

[140] Y. Verma and C. V. Jawahar. Image annotation using metric learning in semantic neighbourhoods. In *Proc. ECCV*, pages 836–849, 2012.

[141] Y. Verma and C. V. Jawahar. Exploring SVM for image annotation in presence of confusing labels. In *Proc. BMVC*, 2013.

[142] Y. Verma and C. V. Jawahar. Im2Text and Text2Im: Associating images and texts for cross-modal retrieval. In *BMVC*, 2014.

[143] Y. Verma and C. V. Jawahar. A probabilistic approach for image retrieval using descriptive textual queries. In *ACM Multimedia*, 2015.

[144] Y. Verma and C. V. Jawahar. Image annotation by propagating labels from semantic neighbourhoods. *Int. J. Comput. Vision*, 2016.

[145] Y. Verma and C. V. Jawahar. A support vector approach for cross-modal search of images and texts. *Computer Vision and Image Understanding*, 2016.

[146] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[147] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 319–326, 2004.

[148] C. Wang, D. Blei, and L. Fei-Fei. Simultaneous image classification and annotation. In *Proc. CVPR*, 2009.

[149] H. Wang, H. Huang, and C. H. Q. Ding. Image annotation using bi-relational graph of images and semantic labels. In *Proc. CVPR*, pages 793–800, 2011.

[150] K. Wang, R. He, W. Wang, L. Wang, and T. Tan. Learning coupled feature spaces for cross-modal matching. In *ICCV*, 2013.

[151] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, 2005.

[152] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[153] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proc. IJCAI*, pages 2764–2770, 2011.

[154] Y. Xiang, X. Zhou, T.-S. Chua, and C.-W. Ngo. A revisit of generative model for automatic image annotation using markov random fields. In *CVPR*, pages 1153–1160, 2009.

[155] L. Xu, K. Crammer, and D. Schuurmans. Robust support vector machine training via convex outlier ablation. In *AAAI*, 2006.

[156] F. Yan and K. Mikolajczyk. Deep correlation for matching images and text. In *CVPR*, 2015.

[157] Y. Yang, C. L. Teo, H. D. III, and Y. Aloimonos. Corpus-guided sentence generation of natural images. In *EMNLP*, 2011.

[158] A. Yavlinsky, E. Schofield, and S. Rüger. Automated image annotation using global features and robust nonparametric density estimation. In *Proc. CIVR*, pages 507–517, 2005.

[159] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

[160] S. Zhang, J. Huang, Y. Huang, Y. Yu, H. Li, and D. N. Metaxas. Automatic image annotation using group sparsity. In *Proc. CVPR*, pages 3312–3319, 2010.