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
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”).

In 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 \( x_i \) in space \( \mathcal{X} = \mathbb{R}^p \). Similarly, each text \( T_i \) is represented using a \( q \)-dimensional feature vector \( y_i \) in space \( \mathcal{Y} = \mathbb{R}^q \). Similar to the Structural SVM framework [132], our objective is to learn a discriminant function \( F : \mathcal{X} \times \mathcal{Y} \to \mathbb{R} \) that can be
used to predict the optimal output $y^*$ given an input $x$ by maximizing $F$ over the space $\mathcal{Y}$; i.e.,

$$
y^* = f(x; w) = \arg\max_{y \in \mathcal{Y}} F(x, y; w) \tag{5.1}
$$

where $w$ is the parameter vector that needs to be learned. We make the standard assumption of $F = w \cdot \Psi(x, y)$; 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 $w$ such that the maximum number of the following constraints are satisfied:

$$
\forall i : \{ w \cdot \Psi(x_i, y_i) > w \cdot \Psi(x_i, y) \} \forall y \in \mathcal{Y} \setminus y_i \tag{5.2}
$$

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

$$
\min_{w, \xi \geq 0} \frac{1}{2} ||w||^2_2 + \frac{C}{N} \sum_{i=1}^{N} \xi_i \tag{5.3}
$$

$$
s.t. \ w \cdot \Psi(x_i, y_i) \geq w \cdot \Psi(x_i, y) + \Delta(y_i, y) - \xi_i , \forall i, y \in \mathcal{Y} \setminus \{y_i\}
$$

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, $\xi_i$ denotes the slack variable, and $\Delta(y_i, y)$ 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 $\Psi(x, y)$ and the loss function $\Delta(y_i, y)$ 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 $w$.

#### 5.2.2.1 Joint Image-Text Representation

The purpose of $\Psi(x, y)$ 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 $x \in X$ and $T$ using another feature vector $y \in Y$, 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 $x$ and $y$ 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 ($w$) 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 $x$ and $y$ using their tensor product. That is, each dimension of $x$ is multiplicatively combined with every dimension of $y$ to get

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

(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 $\Delta(y_i, y)$ 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 $(x_i, y_i)$ and any other prediction $y \neq y_i$, the function is defined such that its value depends on the degree of dissimilarity between $y_i$ and $y$. That is, if $y_i$ and $y$ are dissimilar, the value of $\Delta(y_i, y)$ 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:

\[
\begin{align*}
\Delta_M(y_i, y) &= ||y_i - y||_1, \quad (5.5) \\
\Delta_E(y_i, y) &= ||y_i - y||_2^2, \quad (5.6) \\
\Delta_C(y_i, y) &= 1 - y_i \cdot y, \quad (5.7)
\end{align*}
\]

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., \(\Delta_Z(y_i, y_i) = 0\), \(\Delta_Z(y_i, y_j) \geq 0\) for \(i \neq j\), and \(\Delta_Z(y_i, y_j) \geq \Delta_Z(y_i, y_i)\) for \(i \neq j\) (where \(Z \in \{M, E\}\)). Under the assumption that both \(y_i\) and \(y\) 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 \(\{x_i, y_i\}\), 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 \((x_i, y_i)\), the most violated constraint is the constraint corresponding to the incorrect output \(\hat{y}\) predicted with the maximum score using the current learned parameter vector \(w\).

---

\(^2\)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.
\[ \hat{y} = \arg\max_{y \in \mathcal{Y} \setminus \{y_i\}} \Delta(y_i, y) + w \cdot \Psi(x_i, y) - w \cdot \Psi(x_i, y_i) \]  
\[ \text{(5.8)} \]

Since the last term is constant with respect to \( y \), this can be re-written as:

\[ \hat{y} = \arg\max_{y \in \mathcal{Y} \setminus \{y_i\}} \Delta(y_i, y) + w \cdot \Psi(x_i, y) \]
\[ \text{(5.9)} \]

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

\[ \hat{y}_M = \arg\max_{y \in \mathcal{Y} \setminus \{y_i\}} ||y_i - y||_1 + w \cdot \Psi(x_i, y) \]
\[ \text{(5.10)} \]
\[ \hat{y}_E = \arg\max_{y \in \mathcal{Y} \setminus \{y_i\}} ||y_i - y||_2^2 + w \cdot \Psi(x_i, y) \]
\[ \text{(5.11)} \]
\[ \hat{y}_C = \arg\max_{y \in \mathcal{Y} \setminus \{y_i\}} 1 - y_i \cdot y + w \cdot \Psi(x_i, y) \]
\[ \text{(5.12)} \]

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 \( T' = \{T'_1, \ldots, T'_{|T'|}\} \) consisting of only textual samples, where each \( T'_k \) is represented using a feature vector \( y'_k \in \mathcal{Y} \). Given a query image \( J \) represented by \( x \in \mathcal{X} \), Im2Text requires ranking the elements of \( T' \) according to their relevance with \( x \) using the learned parameter vector \( w \). This can be performed by sorting the elements of \( T' \) based on the score \( F(x, y'_k; w) = w \cdot \Psi(x, y'_k), \quad \forall k \in \{1, \ldots, |T'|\} \) (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 \( x_i \) 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 \( y_i \) 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) \( x^* \) given an input (text) \( y \) by maximizing \( F \) over the space \( \mathcal{X} \). That is,

\[
x^* = f(y; w) = \arg \max_{x \in \mathcal{X}} F(y, x; w),
\]

where \( w \) is the parameter vector that needs to be learned, and \( F = w \cdot \Psi(y, x) \). 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 (\( w \)).

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

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

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

$$ W = [w_1w_2 \ldots w_q] \in \mathbb{R}^{p \times q}, $$
where each \( w_k \in \mathbb{R}^p \) denotes the consecutive \( p \) elements of \( w \) in the \( k^{th} \) interval. Using the above, it is easy to verify that the function \( F(x, y; w) \) can be re-written as:

\[
F(x, y; w) = w \cdot \Psi(x, y) = x^t W y
\]  
(5.14)

Rather than evaluating the function \( F(x, y; w) \) individually for each sample in the retrieval set \( T' \), the above transformation allows us to evaluate it for a batch of samples in \( T' \) in a single pass. Here we will illustrate this by computing it for all the samples in \( T' \) in a single pass. Let \( Y = [y_1 y_2 \ldots y_{|T'|}] \in \mathbb{R}^{q \times |T'|} \) denote the matrix formed by concatenating the feature representations of all the samples in \( T' \).

For a given (image) query \( J \) represented by feature vector \( x \), let \( s \in \mathbb{R}^{|T'|} \) be a vector such that its \( k^{th} \) element denotes the relevance score corresponding to the \( k^{th} \) sample in \( T' \). Then it can be computed as:

\[
s = (x^t W Y)^t \quad \text{.} \tag{5.15}
\]

After computing this, the ranking follows by sorting the elements of \( T' \) based on their corresponding scores in \( s \) 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 \( T' \) 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 \( T' \) 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 $T'$ 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 $T'$ 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
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Samples</th>
<th>#Captions/Img.</th>
<th>Words/Caption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pascal</td>
<td>1000</td>
<td>5</td>
<td>9.82 ± 3.51</td>
</tr>
<tr>
<td>IAPRTC-12</td>
<td>19627</td>
<td>1</td>
<td>24.98 ± 10.67</td>
</tr>
<tr>
<td>SBU</td>
<td>1M</td>
<td>1</td>
<td>12.14 ± 6.01</td>
</tr>
</tbody>
</table>

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

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 ($n_{\text{max}}$), then the BLEU score will be ($\frac{n_{\text{max}}}{n_1}$). 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\textsuperscript{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\textsuperscript{[153]} and CCA\textsuperscript{[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

\textsuperscript{6}Obtained from \url{http://www.berouge.com/Pages/default.aspx}. 

114
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 $U$ and $V$ denote the two projection matrices and $a$ and $b$ denote a pair of samples from the two modalities respectively. Thus, CCA can be seen to match the samples using the similarity function $\exp(-||Ua - Vb||_2^2) = \exp(a'(U'V)b)$. 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 $w$ and makes use of $L_2$ regularization on $w$ 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 cannot 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-
The best results using both are highlighted in bold. (↑: higher means better; ↓: lower means better.)

### 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).

<table>
<thead>
<tr>
<th>Method</th>
<th>Pascal</th>
<th></th>
<th></th>
<th>IAPR</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Im2Text</td>
<td>Text2Im</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BLEU-1↑</td>
<td>Rouge-1↑</td>
<td>R@50↑</td>
<td>MedR↓</td>
<td>BLEU-1↑</td>
<td>Rouge-1↑</td>
</tr>
<tr>
<td>CCA</td>
<td>0.3149</td>
<td>0.1397</td>
<td>47.10</td>
<td>11.05</td>
<td>0.3254</td>
<td>0.1432</td>
</tr>
<tr>
<td>WSABIE (TR)</td>
<td>0.3077</td>
<td>0.1964</td>
<td>11.40</td>
<td>26.25</td>
<td>0.3135</td>
<td>0.1983</td>
</tr>
<tr>
<td>WSABIE (CTR)</td>
<td>0.3119</td>
<td><strong>0.2172</strong></td>
<td>13.50</td>
<td>22.15</td>
<td>0.3102</td>
<td><strong>0.2154</strong></td>
</tr>
<tr>
<td>BITR-M (TR)</td>
<td>0.3151</td>
<td>0.2234</td>
<td>41.20</td>
<td>14.05</td>
<td>0.3240</td>
<td>0.2201</td>
</tr>
<tr>
<td>BITR-M (CTR)</td>
<td>0.3204</td>
<td>0.2374</td>
<td>42.10</td>
<td>15.35</td>
<td>0.3301</td>
<td>0.2416</td>
</tr>
<tr>
<td>BITR-E (TR)</td>
<td>0.3286</td>
<td>0.2098</td>
<td>39.70</td>
<td>13.45</td>
<td>0.3404</td>
<td>0.2289</td>
</tr>
<tr>
<td>BITR-E (CTR)</td>
<td>0.3380</td>
<td>0.2306</td>
<td>40.30</td>
<td>13.85</td>
<td>0.3491</td>
<td>0.2401</td>
</tr>
<tr>
<td>BITR-C (TR)</td>
<td>0.3267</td>
<td>0.2275</td>
<td>46.50</td>
<td>11.30</td>
<td>0.3373</td>
<td>0.2315</td>
</tr>
<tr>
<td>BITR-C (CTR)</td>
<td><strong>0.3485</strong></td>
<td><strong>0.2397</strong></td>
<td><strong>51.40</strong></td>
<td>9.10</td>
<td><strong>0.3489</strong></td>
<td><strong>0.2438</strong></td>
</tr>
<tr>
<td>CCA</td>
<td>0.2946</td>
<td>0.3031</td>
<td>16.32</td>
<td>404.35</td>
<td>0.3050</td>
<td>0.3016</td>
</tr>
<tr>
<td>WSABIE (TR)</td>
<td>0.2670</td>
<td>0.2375</td>
<td>2.73</td>
<td>932.10</td>
<td>0.2601</td>
<td>0.2416</td>
</tr>
<tr>
<td>WSABIE (CTR)</td>
<td>0.2813</td>
<td>0.2497</td>
<td>4.76</td>
<td>772.30</td>
<td>0.2754</td>
<td>0.2700</td>
</tr>
<tr>
<td>BITR-M (TR)</td>
<td>0.3172</td>
<td>0.3043</td>
<td>10.57</td>
<td>519.85</td>
<td>0.2858</td>
<td>0.2650</td>
</tr>
<tr>
<td>BITR-M (CTR)</td>
<td>0.3227</td>
<td>0.3130</td>
<td>10.30</td>
<td>480.25</td>
<td>0.3028</td>
<td>0.2820</td>
</tr>
<tr>
<td>BITR-E (TR)</td>
<td>0.3167</td>
<td>0.3047</td>
<td>10.68</td>
<td>493.30</td>
<td>0.2874</td>
<td>0.2646</td>
</tr>
<tr>
<td>BITR-E (CTR)</td>
<td>0.3391</td>
<td>0.3240</td>
<td>12.41</td>
<td>416.85</td>
<td>0.3090</td>
<td>0.2901</td>
</tr>
<tr>
<td>BITR-C (TR)</td>
<td>0.3219</td>
<td>0.3165</td>
<td>9.37</td>
<td>594.45</td>
<td>0.2963</td>
<td>0.2711</td>
</tr>
<tr>
<td>BITR-C (CTR)</td>
<td><strong>0.3418</strong></td>
<td><strong>0.3281</strong></td>
<td><strong>13.78</strong></td>
<td><strong>335.95</strong></td>
<td><strong>0.3149</strong></td>
<td><strong>0.2966</strong></td>
</tr>
</tbody>
</table>
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
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>
<thead>
<tr>
<th>Experiment-1</th>
<th>Experiment-2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Im2Text</strong></td>
<td><strong>Text2Im</strong></td>
</tr>
<tr>
<td><strong>Query Image:</strong></td>
<td><strong>Query Image:</strong></td>
</tr>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td>Two local teachers are standing in a classroom with many children sitting at their wooden desks.</td>
</tr>
<tr>
<td><strong>Query Text:</strong> 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><strong>Query Text:</strong> Two local teachers are standing in a classroom with many children sitting at their wooden desks.</td>
</tr>
<tr>
<td><strong>Output Text:</strong> People are sitting at a laid table made of wood in a little dark restaurant.</td>
<td><strong>Output Image:</strong></td>
</tr>
<tr>
<td><strong>Output Image:</strong></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
</tbody>
</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 × 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.

7http://code.google.com/p/word2vec/
<table>
<thead>
<tr>
<th>Method</th>
<th>Im2Text</th>
<th>Text2Im</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU-1↑</td>
<td>Rouge-1↑</td>
</tr>
<tr>
<td>Pascal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCA</td>
<td>0.3469</td>
<td>0.1429</td>
</tr>
<tr>
<td>WSABIE</td>
<td>0.3298</td>
<td><strong>0.2343</strong></td>
</tr>
<tr>
<td>BITR-M</td>
<td>0.3466</td>
<td>0.2473</td>
</tr>
<tr>
<td>BITR-E</td>
<td>0.3612</td>
<td>0.2578</td>
</tr>
<tr>
<td>BITR-C</td>
<td><strong>0.3712</strong></td>
<td>0.2602</td>
</tr>
<tr>
<td>IAIP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCA</td>
<td>0.3163</td>
<td>0.3278</td>
</tr>
<tr>
<td>WSABIE</td>
<td>0.2964</td>
<td>0.2685</td>
</tr>
<tr>
<td>BITR-M</td>
<td>0.3471</td>
<td>0.3398</td>
</tr>
<tr>
<td>BITR-E</td>
<td>0.3568</td>
<td>0.3459</td>
</tr>
<tr>
<td>BITR-C</td>
<td><strong>0.3672</strong></td>
<td><strong>0.3545</strong></td>
</tr>
<tr>
<td>SBU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCA</td>
<td>0.1452</td>
<td>0.1237</td>
</tr>
<tr>
<td>WSABIE</td>
<td>0.1395</td>
<td><strong>0.1280</strong></td>
</tr>
<tr>
<td>BITR-M</td>
<td>0.1514</td>
<td>0.1366</td>
</tr>
<tr>
<td>BITR-E</td>
<td>0.1628</td>
<td><strong>0.1519</strong></td>
</tr>
<tr>
<td>BITR-C</td>
<td><strong>0.1723</strong></td>
<td>0.1492</td>
</tr>
</tbody>
</table>

**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 $\Psi$ 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 transformation 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 $w$ 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 $\mathbb{R}^P$:

$$\Phi_I(x) : \mathbb{R}^p \rightarrow \mathbb{R}^P.$$  \hspace{1cm} (5.16)

while jointly learning a mapping function for annotations:

$$\Phi_W(i) : \{1, \ldots, Y\} \rightarrow \mathbb{R}^P.$$  \hspace{1cm} (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:
   \[ L\left(\left\lfloor \frac{Y-1}{N} \right\rfloor \right)|1 - f_{y_i}(x_i) + f_{\bar{y}}(x_i)|_+ \]
   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:

\[ f_i(x) = \Phi_W(i)^T \Phi_I(x) = W_i^T V x, \quad (5.18) \]

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:

\[
||V_i||_2 \leq \lambda, i = 1, \ldots, p, \\
||W_i||_2 \leq \lambda, i = 1, \ldots, Y. \quad (5.19)
\]

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:

\[ L(\left(\frac{|\mathcal{C}| - 1}{N}\right)|1 - g_y(x_i) + g_{\bar{y}}(x_i)|) \]

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 \(\mathbb{R}^P\):

\[ \Phi_Z(y) : \mathbb{R}^q \rightarrow \mathbb{R}^P, \quad (5.20) \]

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:

\[ g_y(x) = \Phi_Z(y)^T \Phi_I(x) = y^T Z^T V x. \quad (5.21) \]
Similar to Eq. 5.19, this family of models have constrained norm:

\[
\|V_i\|_2 \leq \lambda, \ i = 1, \ldots, p, \\
\|Z_i\|_2 \leq \lambda, \ i = 1, \ldots, q.
\] (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.