ABSTRACT

Dissertation title : Discriminative Interlingual Representations

Jagadeesh Jagarlamudi, Doctor of Philosophy, 2013

Dissertation advised by: Hal Daumé III
Department of Computer Science

The language barrier in many multilingual natural language processing (NLP) tasks can be overcome by mapping objects from different languages ("views") into a common low-dimensional subspace. For example, the name transliteration task involves mapping bilingual names and word translation mining involves mapping bilingual words into a common low-dimensional subspace. Multi-view models learn such a low-dimensional subspace using a training corpus of paired objects, *e.g.,* names written in different languages, represented as feature vectors.

The central idea of my dissertation is to learn low-dimensional subspaces (or interlingual representations) that are effective for various multilingual and monolingual NLP tasks. First, I demonstrate the effectiveness of interlingual representations in mining bilingual word translations, and then proceed to developing models for diverse situations that often arise in NLP tasks. In particular, I design models for the following problem settings: 1) when there are more than two views but we only have training data from a single pivot view into each of the remaining views 2) when an object from one view is associated with a ranked list of objects from another view, and finally 3) when the underlying objects have rich structure, such as a tree.

These problem settings arise often in real world applications. I choose a canonical task for each of the settings and compare my model with existing state-of-the-art baseline systems. I provide empirical evidence for the first two models on multilingual name transliteration and reranking for the part-of-speech tagging tasks, respectively. For the third problem setting, I experiment with the task of re-scoring target language word translations based on the source word's context. The model proposed for this problem builds on the ideas proposed in the previous models and, hence, leads to a natural conclusion.

Discriminative Interlingual Representations

by

Jagadeesh Jagarlamudi

A Dissertation presented to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2013

Advisory Committee:

Professor Hal Daumé III, Chair/Adviser
Professor Jordan Boyd-Graber
Professor Alexander M. Fraser
Professor Lise Getoor
Professor William J. Idsardi
Professor Philip Resnik

To my parents and family members ...

Acknowledgments

I owe my gratitude to all the people who made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost, I'd like to thank my advisor Hal Daumé III who has given me the freedom to work on problems that excited me the most. He ensured that we get to spend most of our time on our research, which I think helped me in a great deal in finishing my dissertation soon. His insight in various NLP/ML problems is invaluable and has immensely helped in shaping up my dissertation. There are numerous occassions where I asked for his help in the last minute, be it for his advice, comments on a paper, support for conference travel, or in an administrative hickup. He never said no and was always available. It has been a pleasure to work with Hal and I couldn't have asked for a better adviser.

I would also like to thank my committee members, Philip Resnik, Jordan Boyd-Graber, Lise Getoor, William J. Idsardi and Alex Fraser. I have heard stories about how hard it was to schedule the thesis defense. Fortunately, it has been a pleasant experience for me and I thank you all for your helpful comments.

I would like to thank my former colleagues and friends, Raghvendra Udupa and A. Kumaran. My prior working experience with Kumaran got me excited towards multilingual NLP problems. I would especially thank Raghavendra for his friendly support through out my graduate studies. It was his converstations that introduced me to interlingual representations for NLP.

My colleagues in the CLIP lab at UMD and the NLP lab at Utah helped me in making my graduate life memorable. I thank you all. I would like to thank Jennifer Story and Fatima Bangura for their administrative assistance and prompt responses. I would like to thank the funding agencies National Sciences Foundation and the BOLT program of the Defense Advanced Research Projects Agency, who supported my research and conference travel through out my graduate studies.

I owe my deepest thanks to my family members, especially my father, mother, and sister, who have always encouraged me to pursue my dreams. Thank you very much for your patience. I sincerely thank my other family members, especially my cousins, who supported me and helped me through rough times. Words cannot express how fortunate I am to have you all around me.

Lastly, thank you all and thank God!

Table of Contents

List of Figures

List of Tables

Notation / List of Abbreviations

Common Notation

n	An italicized lower case letter denotes a scalar.
$\mathbf{x}$	A bold letter represents a column vector.
$\mathbf{x}(i)$	Represents the i^{th} component of vector $\mathbf{x}$.
$\mathbf{x}_i$	Represents i^{th} vector.
$\mathbf{x}_i^j$	Represents i^{th} vector in the j^{th} view. For vectors, subscript denotes the index and superscript to denote the view or language.
X	Capital italicized letter represents a matrix. If the matrix is a data matrix, then columns are observations and rows are dimensions.
$X(i,j)$	Element at i^{th} row and j^{th} column of the matrix X.
X_j	Represents a matrix in the j^{th} view. For matrices and scalars, I use subscript to indicate the view or language.
$\mathbf{x}^T$, X^T	Denotes the transpose of the vector $\mathbf{x}$ and matrix X respectively.
I	Denotes identity matrix and its size will be clear from the context.
$\mathbf{0}$	Represents a zero matrix or vector. Usage will be clear from the context.
$\vec{g}(.)$	A function whose range is a vector.
$\langle \mathbf{u}, \mathbf{v} \rangle$	Represents the dot product of two vectors $\mathbf{u}$ and $\mathbf{v}$.
$\mathbb{R}$	Denotes the set of real numbers.

Evaluation Measures

Accuracy	Accuracy of the top ranked object.
BLEU	Bilingual Evaluation Understudy.
Meteor	Metric for Evaluation of Translation with Explicit ORdering.
MRR	Mean Reciprocal Rank.

Abbreviations

CCA	Canonical Correlation Analysis
CLIR	Cross-lingual Information Retrieval
CL-LSI	Cross-lingual Latent Semantic Indexing
IR	Information Retrieval
LPP	Locality Preserving Projections
MI	Mutual Information
MMR	Multi-view Multivariate Regression
NLP	Natural Language Processing
NPP	Neighbourhood Preserving Projections
OPCA	Orthogonal Principal Component Analysis
OOV	Out-of-Vocabulary
PLS	Partial Least Squares
POS	Part-of-Speech
SMT	Statistical Machine Translation
SVD	Singular Value Decomposition
VSM	Vector Space Model
WSM	Word Space Model

Chapter 1

Introduction

The rapid growth of technologies to publish text in different languages dramatically increased the multilingual content on the web. This situation demands tools for processing text in different languages as well as the tools that bridge the language barrier. The language barrier in many of the multilingual applications can be broken by mapping text from different languages into a common language independent representation (or interlingual representation). Previous attempts for interlingua construction are task independent and hence are not optimal. The development of statistical approaches and the availability of training corpora for different tasks enables us to learn task specific interlingual representations. Inspired by the idea of dimensionality reduction, in this dissertation, I propose models to learn interlingual representations that are effective for diverse multilingual and monolingual natural language processing (NLP) tasks.

1.1 Learning

Machine learning is the problem of learning behaviors based on empirical data. There are two major types of learning problems, supervised and unsupervised learning. In the supervised setting, a system is provided with a set of samples (inputs) and responses (labels or outputs) and it is expected to learn the relation between them so that, in future, it can predict the response of an unseen sample. In the unsupervised learning, we simply have large amounts of samples and the system tries to learn frequent patterns, such as clusters or association pairs, from the input data. Throughout this dissertation I only address problem of supervised learning category.

There are two main steps in supervised machine learning: training and prediction. During the training phase, the system is provided with sufficient amount of input (x) and output (y) pairs which is referred to as training data. A machine learning algorithm takes these input, output pairs and aims to learn the underlying relationship between them. During prediction, given a new unseen input, the algorithm uses the relationship learned from the training data to predict the desired output variable. For example, in document classification task the system is provided with a training data text documents and their class labels (*e.g.,* sports, health, politics *etc.*). During prediction, given a new unseen text document the system assigns an appropriate class label to this document. In the web search task, the training data consists of example query, document pairs along with their relevance judgements – usually a number between 0 and 5, with 0 indicating that the document is not relevant to the query and 5 indicating that the document is perfectly relevant to the query. During prediction, given a new query the system has to rank all the documents based on their relevance towards the query.

In many tasks, the input and output variables are usually represented as vectors, *i.e.,* $\mathbf{x} \in \mathbb{R}^{d_1}$

and $\mathbf{y} \in \mathbb{R}^{d_2}$. For example, in bilingual document alignment task the system is provided with a set of paired (or aligned) documents $(\mathbf{x}_i, \mathbf{y}_i)$ in two different languages as the training data. During prediction, we are given a bilingual comparable corpora – documents in different languages that are talking about same topics but are not translations of each other – and are told that some documents in one language have aligned document in the other language. And the task involves, given a document in one language ($\mathbf{x}$) find its aligned document ($\mathbf{y}$) in the other language. In this task, a document is usually represented as a vector with each dimension corresponding to a word in the vocabulary and the value being some variant of the word's frequency in that document. Since documents are usually much shorter than the vocabulary size, total number of unique words in a language, only few components of a document vector will have a non-zero value. Thus, representing documents as high dimensional vectors leads to sparseness issues. Though this problem is not specific to text documents, it is crucial in NLP applications due to the huge vocabulary sizes.

Under these situations, it is often rewarding to map high dimensional vectors into a lower k-dimensional subspace, where $k \ll d_1$ and $k \ll d_2$, which preserves the necessary information required for downstream applications. Multi-view models find such a lower dimensional subspace such that a pair of aligned objects $(\mathbf{x}_i, \mathbf{y}_i)$, when mapped into this subspace, lie closer to each other. Inspired by this idea of dimensionality reduction, in this dissertation, I propose several models to find lower dimensional subspaces that are suitable for the multilingual NLP applications. Since this low-dimensional subspace is language independent it is referred as interlingual representation. In what follows, I will first describe the relevance of multilinguality using multilingual web search as an example task. I use web search as motivating example because it is widely used, easier to understand and also demonstrates the recurring concept of interlingual representation in a natural way. But in this dissertation, I mainly consider NLP applications.

1.2 Need for Multilinguality

Because of the prevalence of English, it has considerably more resources compared to other languages for many NLP tasks (such as part-of-speech (POS) tagging, dependency parsing, *etc.*) and information retrieval tasks (such as web search, document classification, *etc.*). For example, in the web search task, training data – query-document pairs along with their relevance judgements – is abundantly available for English compared to other languages. In addition, since English web search engines have been deployed for a longer time, rich user behavioral data – user clicks recorded per query basis which is shown to improve accuracies significantly – is abundantly available for English compared to other languages [87]. In dependency parsing, the treebank corpus, a set of parsed sentences used for training a parser model, has been available for long time and contains more number of sentences in English than many of the resource poor languages [23, 133]. Sentiment classification usually relies on word polarity information – whether a word is likely to be positively associated or negatively associated. Since such information is not available in language such as Chinese, it is derived from English words [11].

This situation led the researchers to devise appropriate ways to leverage training data available in resource rich languages to assist applications in the resource poor languages [133, 38, 58, 32]. Comparable corpora usually aids the process of transferring useful information across languages. In the following section, I use multilingual web search to demonstrate the importance of leveraging multilingual resources. At the same time, I introduce the idea of an interlingual representation which recurs often in this dissertation. In the reminder of this dissertation, we will see this technique being applied to diverse NLP tasks.

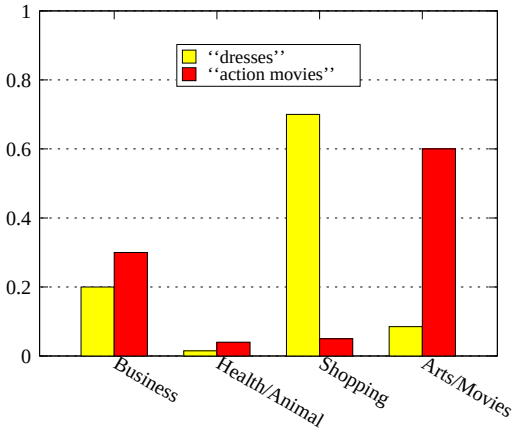


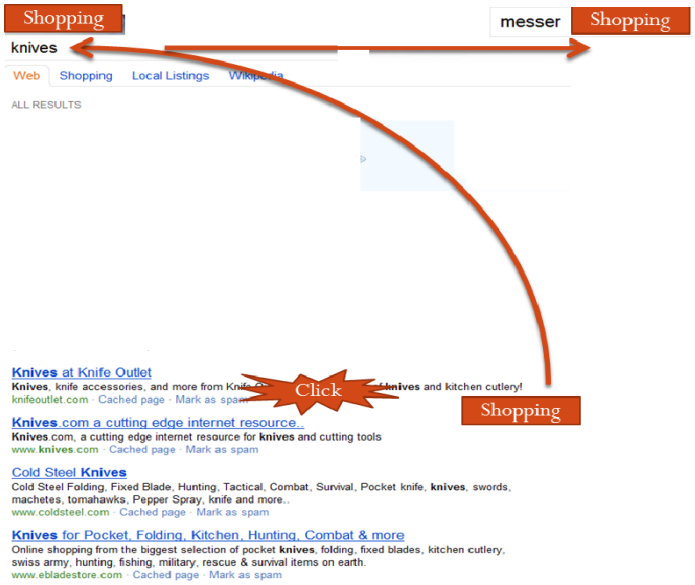
Figure 1.1: Shows the query intent (represented as a distribution over four classes) of two queries. The user query "dresses" may have a hidden intent of 'Shopping' while the query "action movies" is more likely to have an intent of 'Arts/Movies'. Notice that the classes may not be mutually exclusive and so the distribution doesn't need to sum to one.

1.3 Motivating Example – Multilingual Web Search

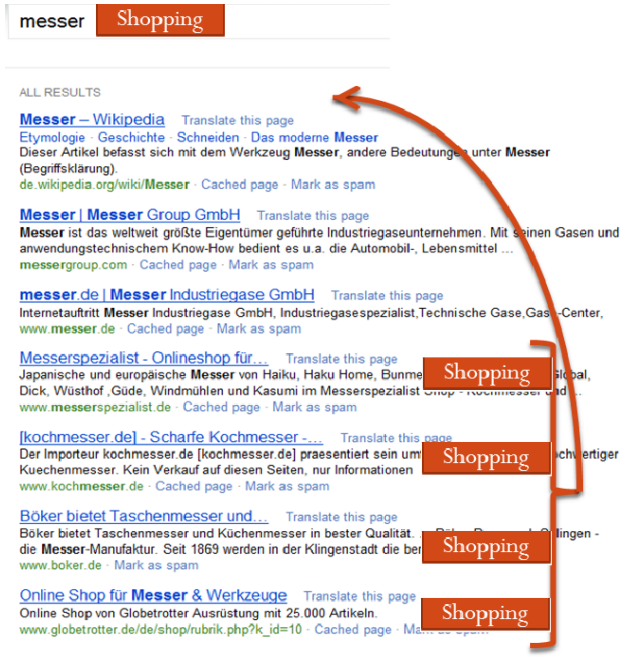
Multilingual web search [58, 87] is the problem of improving web search ranking in a relatively data-scarce *search language* (*e.g.,* German or French) using data from an *assisting language* (*e.g.,* English). For clarity, we use German as the search language and English as the assisting language. Notice that both the user query and the result documents are in the same language (*i.e.,* German). Different techniques exist for addressing this task [58, 33, 32, 57, 7, 87, 60, 59]. But, here, we consider an interlingual classification based approach [88] as it uses a language independent representation (an interlingua) to transfer useful information from English queries to the German queries and fits into the framework of this dissertation.

In the interlingual classification based approach [88], the intent of a user query is represented as a distribution over some fixed set of classes, referred to as query class distribution. For example, Fig. 1.1 shows the query class distributions for two queries over four different classes. It shows that, a user issuing the query "dresses" is likely to click shopping documents, where as a user issuing the query "action movies" is more likely to click documents of the category 'Arts/Movies'. Based on this assumption, ranking of the web pages whose intent matches with that of the query is boosted. Bennett *et al.* [16] show that a query class distribution learnt from the web pages that are clicked by previous users, for the same query, is very effective. But this method requires sufficient amount of click information which is usually not available in the resource poor search language (such as German). So, the interlingual classification based approach uses resource rich assisting language (such as English) to get reliable estimates of the German query's class distribution. At a higher level, it translates the German query into English; uses the English click information to learn the translated English query's intent; and then transfers it onto the original German query. This process is explained briefly with the help of an example (Fig. 1.2) in the next paragraph.

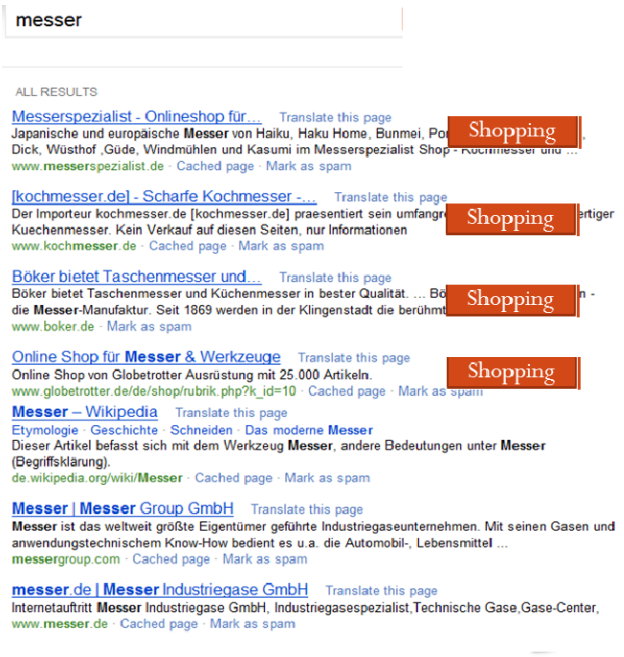
First, documents in different languages are classified into the same set of classes; *e.g.,* the ODP-aligned class labels (Table 1.1). Given a German query (*e.g.,* 'Messer'), it is translated into English ('Knives') using machine translation. After this, the query class distributions for the German query and its English translation are obtained based on their respective language's click data. The query



(a) The English query intent is transferred to German query.



(b) German Results matching with query intent



(c) Re-ranked German Results.

Figure 1.2: a) The English search engine results are shown for the English translation of the German query 'Messer'. The class information of the English query (*i.e.,* , 'Shopping' – shown in orange) is derived using the result document's class along with the click data and is transferred onto the original German query. b) German results belonging to the query's class 'Shopping' are highlighted; notice that these documents are ranked lower than other documents. c) German ranking model trained using the query class information improves the ranking of the result documents that match the query's class.

English	German
Business	Wirtschaft
Arts/Movies	Kultur/Film
Computers/Internet	Computer/Internet
Health/Animal	Gesundheit/Tiere
Recreation/Boating	Freizeit/Boots-_und_Schiffsfahrt
Shopping/Home_and_Garden	Online-Shops/Haus_und_Garten

Table 1.1: Example ODP classes from English and their aligned classes in German. The German aligned class can be obtained by first visiting the English class webpage and then following the "German" link.

class distribution derived from English translation is simply transferred to the German query by invoking interlingual classification (Fig. 1.2(a)). Last, the English and German query-class distributions are used to derive several ranking features that are fed into the ranking model to improve the relevance ranking of the documents (Fig. 1.2(b) and Fig. 1.2(c)). This approach dramatically improves the performance for infrequent German queries, a class of queries that are notoriously difficult to rank accurately due to the lack of click information, since their English translation may have enough click information in English [88].

1.4 Interlingual Representation

In the approach described in the previous section, the key step that allowed us to transfer information from English query onto German query is the interlingual classification. The documents in different languages are classified into a fixed set of k interlingual classes, which enabled us to transfer the query class distribution from English to German. As shown in Fig. 1.3, the class distribution of each document can be represented as a point in the k-dimensional space where each dimension is spanned by a class. Then, the document classifiers in each language can be seen as functions which map documents from original textual representation, which is high dimensional, into points in the low k-dimensional subspace. In this process, we expect documents that describes the same content to get mapped to the same point, in the k-dimensional space, irrespective its language. This is the crucial property that I repeatedly use in formulating the objective functions through out this dissertation. Before proceeding further, I will fix some terminology and use it in the rest of the dissertation.

- We refer to the low k-dimensional subspace as interlingual representation (in the context of multiple languages), simply subspace (in general context) or common space.
- We refer to the mapping functions that transform an object from its original space into the subspace as projection directions, projection matrices or mapping functions.
- The operation of transforming objects from original high dimensional space into points in the low-dimensional subspace is referred as projection or mapping.

Unlike the example in Fig. 1.3 where each dimension of the subspace corresponds to a class, the dimensions of the interlingual representations that we see in the rest of this dissertation doesn't have clear meaning.

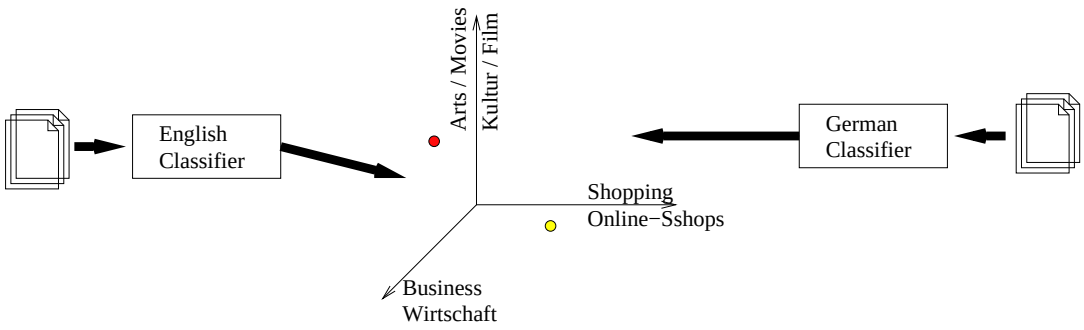


Figure 1.3: Classifiers mapping documents from different languages into an interlingua. The class distribution of each document can be seen as a point in this k dimensional space. To emphasize that this is interlingual representation, I also denoted the class labels in both English and German.

As mentioned in [88], the effectiveness of the interlingual classification approach (described in the previous section) depends on the consistency of the classifiers employed in different languages. That is, the classifiers need not be highly accurate but they should be consistent, in the sense that they should make same kind of errors. For example, even though the English classifier misclassifies a soccer document as a hockey document, it is acceptable as long the German classifier does the same mistake. Otherwise it breaks the central assumption that documents that describe the same content get mapped to the same point. Unfortunately, such consistency is hard to ensure when the classifiers are trained independently as is done in [88]. Moreover, the appropriateness of classification for this task is also arguable.

Thus the main problem that I address in this dissertation is that, given a set of aligned objects, I aim to find a subspace that is appropriate for downstream NLP applications. I observe that this is central problem in many of the multilingual NLP applications such as document alignment, bilingual dictionary mining, multilingual name transliteration as well as the general structure prediction problems such as reranking and other related tasks. Inspired by dimensionality reduction, I propose several models and show their effectiveness in the above mentioned tasks. Finally, I use the low-dimensional word embeddings to enrich the existing bag-of-word models by incorporating the syntactic structure and word meaning into the representation.

1.4.1 Interlingua and Interlingual Representation

The interlingual representations that I discuss in this dissertation are different from the interlingua used for machine translation [45, 84, 83, 47]. The interlingua is motivated from the long-lived belief that, while languages differ in their "surface structures" they share some common "deep structures". These deep structures are assumed to be independent of any language and capture essential information that is required to unambiguously express the information conveyed in a text. KANT [134] and UNITRAN [45] are two earlier machine translations systems that are based on this assumption. These approaches primarily involve two components: an analysis and a generation component. Analysis component includes analysis of the source language text to represent it in the interlingua and the generation component which generates the target language sentences from the interlingua. In order to make the architecture more modular, these steps are done separately and are independent of each other. Moreover, the intermediate representation is predefined and does not depend on the task. Though both these approaches differ in the actual intermediate

representation, it comprises information from all necessary levels of linguistic analysis; lexical, syntactic, semantic and pragmatic information. While such representations are very rich, they are designed for MT and much of the information they capture may not be relevant for any other task.

On the other hand, the interlingual representations that we see in this dissertation are inspired by dimensionality reduction and are task specific. The training data provided for a given task determines the representation and its size. As mentioned earlier, unfortunately, these interlingual representations do not have a clear interpretation of the information being captured by them. But, since the representations are task specific they tend to be optimal and give better results. Moreover, one can modify the models such that the resulting interlingual representations vary in terms of the granularity of the information they capture. For *e.g.,* in Chp. 6, we will see a model that will learn an interlingual representation that can discriminate between different syntactic analyses of a given sentence.

1.5 An Overview of This Dissertation

The rest of this dissertation is organized into three parts. In the first part, I describe the relevant background work that is necessary for understanding the later Chapters. In the second part, I describe my modeling contributions and show their effectiveness compared to state-of-the-art baseline systems on different tasks. In the third part, I conclude the dissertation and also propose some future research directions. Since I address different tasks in this dissertation, I discuss the related work for each of the tasks in the relevant Chapters.

Part I: Background

Chapter 2 : Canonical correlation analysis (CCA) is a dimensionality reduction technique to find a common subspace for multi-view data. In this Chapter, I describe how CCA models sets of related variables. I describe the objective function and give a small worked out example which shows few aligned points in two views, the lower dimensional subspace identified by CCA and the mapping functions from the original spaces into the subspace. I briefly describe the optimization process as I will frequently use a similar process in the remaining Chapters.

Chapter 3 : There have been several attempts to model the ubiquitous multi-view data. In this Chapter, I briefly review the relevant multi-view models. Though these models are not directly related to this dissertation, for completeness, I present them and try to draw relationship to CCA whenever such a relation exist. The reader can skip this Chapter completely and still follow the rest of the dissertation.

Part II: Modeling Contributions

Chapter 4 : I first demonstrate the effectiveness of interlingual representation in mining translations for out-of-vocabulary words for statistical machine translation. Thus, I establish the importance of interlingual representations. Next, I propose a feature selection technique for multi-view models. Multi-view models capture essential feature co-occurrences in terms of covariance matrices. In theory, these covariance matrices should be very sparse but, in practice, they tend to be dense due to the noise in the data. To overcome this, I

introduce the idea of sparsifying covariance matrices and show that this is same as feature selection in the joint feature space. Experiments on the document alignment task show the effectiveness of sparsifying covariance matrices. Sparsification is orthogonal to the models that I discuss in the rest of the dissertation and hence it can be combined with any of the models.

Chapter 5 : As discussed earlier, the prevalence of English has caused many of the bilingual resources to be developed into English than to any other language. This naturally led to the development of bridge language approaches that pivot through English. In this Chapter, I address the problem of finding a low-dimensional subspace when we have training data from multiple languages into a common bridge language. I propose regularized projections which preserve the language specific phenomenon, unlike other approaches, at the same time generalizing the phenomenon that is shared across many languages. I evaluate this model on multilingual name transliteration task, where I use international phonetic alphabet (IPA) in a manner akin to a bridge language. In this task, my model captures language specific phonemic variation and achieves dramatic improvements over other bridge language approaches.

Chapter 6 : Observing that the problem of mapping input vectors to output vectors is a general problem that arises in monolingual structured prediction problems as well, I address reranking for part-of-speech (POS) tagging. Unlike the alignment problem, usually addressed by multi-view models, reranking requires the ability to characterize and discriminate the fine grained differences between candidate outputs of a single input. In this Chapter, using insights from structure prediction literature, I propose a family of models for low-dimensional discriminative reranking problem. My models use features defined *within* the input and output space and automatically learn the interactions between them. Thus they avoid the necessity of a careful feature designing step and also achieves significantly better results.

Chapter 7 : In this Chapter, I propose to enrich the existing bag-of-word representations by incorporating word embeddings into the representation. I introduce a model for computing vector representation of objects, based on the word embeddings. The objective function is formulated such that the vector representations of two sentences that convey same meaning (*e.g.,* "I ate an apple" and "I had a fruit") are close to each other. Learning parameters of this model is cast as multi-view multivariate regression problem, a generalization of multivariate regression. Due to the lack of resource for such a task, I propose a novel use of bilingual parallel data, sentence pairs, as training data to learn the parameters.

Part III: Conclusion and Future Work

Chapter 8 : In the final Chapter, I conclude the dissertation based on the insights that I gained by using dimensionality reduction for various NLP problems. I discuss problem settings where one can expect gains by using dimensionality reduction. Towards the end of this Chapter, I propose some future research directions.

Part I

Background

Chapter 2

Canonical Correlation Analysis

The problem of modeling sets of related variables from different views by means of latent variables arises in many areas such as computer vision, computational musicology and NLP. The language barrier in many of the multilingual NLP applications, such as bilingual document alignment or name transliteration, can be overcome by mapping objects (documents) from different languages into a common interlingual representation. This requires learning two mapping functions one from each language into the common representation. As outlined in the previous Chapter, choosing a common subspace first and then learning the mapping functions independently does not enforce consistency. So in this Chapter, I describe multi-view models that learn the subspace as well as the mapping functions simultaneously. Multi-view models such as canonical correlation analysis (CCA) and partial least squares (PLS) learn a low-dimensional subspace such that the correlation among the sets of variables, when projected into the low-dimensional subspace, is maximized. In this Chapter, I will review these multi-view models and also discuss how they are related among themselves.

2.1 Notation

We assume that we have training data of n aligned pairs in different views (*e.g.,* paired documents in different languages). Let $\mathbf{x}_i \in \mathbb{R}^{d_1}$ and $\mathbf{y}_i \in \mathbb{R}^{d_2}$ $(i = 1 \cdots n)$ represent the vector representation of i^{th} aligned pair in both the views, respectively. Moreover, let X $(d_1 \times n)$ and Y $(d_2 \times n)$ be the representation of data in both the views with i^{th} column given by $\mathbf{x}_i$ and $\mathbf{y}_i$ respectively. Further more assume that the data is centered (*i.e.,* mean is subtracted from each vector; $\mathbf{x}_i \leftarrow \mathbf{x}_i - \mu_x$ and $\mathbf{y}_i \leftarrow \mathbf{y}_i - \mu_y$). Finally, let $C_{xx} = XX^T$, $C_{yy} = YY^T$ denote the autocovariance matrices, and $C_{xy} = XY^T$ be the cross-covariance matrix.

2.2 Canonical Correlation Analysis (CCA)

In this section, I briefly describe how canonical correlation analysis (CCA) models relationship between sets of related variables. Given the data of X $(d_1 \times n)$ and Y $(d_2 \times n)$, CCA finds projection directions $\mathbf{a}^* \in \mathbb{R}^{d_1}$ and $\mathbf{b}^* \in \mathbb{R}^{d_2}$ as follows:

$$(\mathbf{a}^*, \mathbf{b}^*) = \arg\max_{\mathbf{a},\mathbf{b}} \frac{\mathbf{a}^T XY^T \mathbf{b}}{\sqrt{\mathbf{a}^T XX^T \mathbf{a}}\sqrt{\mathbf{b}^T YY^T \mathbf{b}}} = \arg\max_{\mathbf{a},\mathbf{b}} \cos(X^T\mathbf{a}, Y^T\mathbf{b}) \tag{2.1}$$

I slightly abuse the notation and use the same variables $\mathbf{a}$ and $\mathbf{b}$ for the result of the optimization problem as well, so I simply drop the * in the rest of this dissertation.

In other words, CCA maximizes the correlation between vectors $X^T\mathbf{a}$ and $Y^T\mathbf{b}$, where correlation between two vectors $\mathbf{t}$ and $\mathbf{u}$ is defined as

$$corr(\mathbf{t},\mathbf{u}) = \cos(\mathbf{t},\mathbf{u}) = \frac{\mathbf{t}^T\mathbf{u}}{\sqrt{\mathbf{t}^T\mathbf{t}}\sqrt{\mathbf{u}^T\mathbf{u}}} = \frac{cov(\mathbf{t},\mathbf{u})}{\sqrt{var(\mathbf{t})}\sqrt{var(\mathbf{u})}} \tag{2.2}$$

where $\cos(\mathbf{t},\mathbf{u})$ is the cosine angle between the vectors, $cov(\mathbf{t},\mathbf{u}) = \frac{\mathbf{t}^T\mathbf{u}}{n}$ is the covariance between the two vectors and $var(\mathbf{t})$ is the variance of the vector $\mathbf{t}$. This reformulation in terms of *corr*, *cov* and *var* functions becomes relevant when we discuss its relationship with PLS. Geometrically, CCA first projects all the input points along both the directions ($\mathbf{a}$ and $\mathbf{b}$) as:

$$\mathbf{t} = \begin{pmatrix} \mathbf{x}_1^T\mathbf{a} \\ \mathbf{x}_2^T\mathbf{a} \\ \vdots \\ \mathbf{x}_n^T\mathbf{a} \end{pmatrix} \qquad \mathbf{u} = \begin{pmatrix} \mathbf{y}_1^T\mathbf{b} \\ \mathbf{y}_2^T\mathbf{b} \\ \vdots \\ \mathbf{y}_n^T\mathbf{b} \end{pmatrix} \tag{2.3}$$

and then maximizes the cosine angle between the two vectors $\mathbf{t}$ and $\mathbf{u}$. The cosine angle achieves a maximum value of 1 when both the vectors are same up to a constant factor, *i.e.,* $\mathbf{t}_i = c \cdot \mathbf{u}_i \Rightarrow \mathbf{x}_i^T\mathbf{a} = c \cdot \mathbf{y}_i^T\mathbf{b}$ $(\forall i = 1 \cdots n)$.

We will rewrite the objective function shown in Eq. 2.1 in terms of Euclidean distance as it is easier to visualize in some situations [71]. First notice that the objective function shown in Eq. 2.1 is invariant to scaling of the vectors $\mathbf{a}$ and $\mathbf{b}$, so we can rewrite the optimization problem as follows:

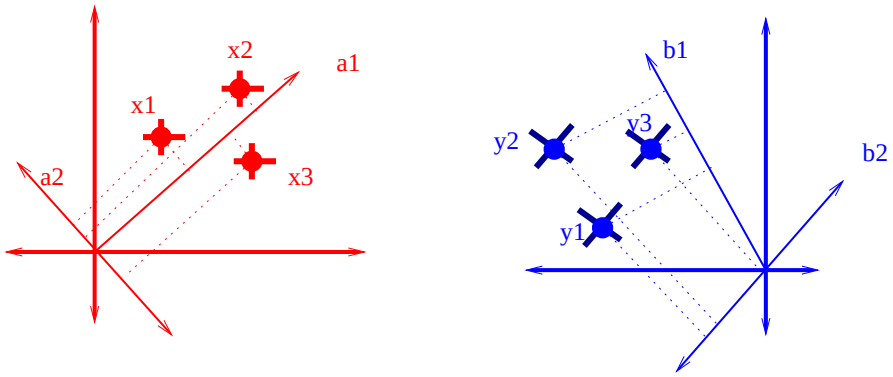
$$\arg\max_{\mathbf{a},\mathbf{b}} \ \mathbf{a}^T XY^T\mathbf{b} \qquad \text{s.t. } \mathbf{a}^T XX^T\mathbf{a} = 1 \quad \text{and} \quad \mathbf{b}^T YY^T\mathbf{b} = 1 \tag{2.4}$$

$$= \arg\min_{\mathbf{a},\mathbf{b}} ||X^T\mathbf{a} - Y^T\mathbf{b}||^2 \qquad \text{s.t. } \mathbf{a}^T XX^T\mathbf{a} = 1 \quad \text{and} \quad \mathbf{b}^T YY^T\mathbf{b} = 1 \tag{2.5}$$

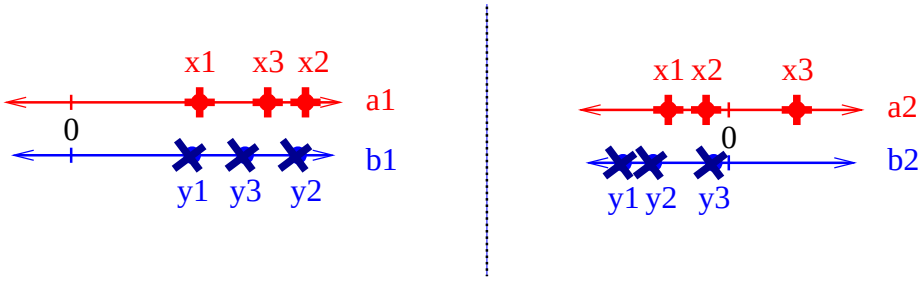
$$= \arg\min_{\mathbf{a},\mathbf{b}} \sum_i \left(\mathbf{x}_i^T\mathbf{a} - \mathbf{y}_i^T\mathbf{b}\right)^2 \qquad \text{s.t. } \sum_i \left(\mathbf{x}_i^T\mathbf{a}\right)^2 = 1 \quad \text{and} \quad \sum_i \left(\mathbf{y}_i^T\mathbf{b}\right)^2 = 1 \tag{2.6}$$

In the rest of this dissertation, we refer to the constraints of the form $\mathbf{a}^T XX^T\mathbf{a} = 1$ and $\mathbf{b}^T YY^T\mathbf{b} = 1$ as length or norm constraints. From the last two equations, it is clear that, in the absence of length constraints, a trivial solution is to set $\mathbf{a}$ and $\mathbf{b}$ to zero vector. The length constraints avoid this trivial solution. In the following sections, I discuss different ways to solve this optimization problem and then give a numerical worked out example. But before that, I present a geometric visualization of CCA for better understanding.

To get an intuition of the directions learned by CCA, consider the example shown in Fig. 2.1. The top row shows three pairs of points in two different (color coded) two dimensional spaces. The points $(\mathbf{x}_i, \mathbf{y}_i)$ are aligned. It also shows two potential pairs of directions $(\mathbf{a}_1, \mathbf{b}_1)$ and $(\mathbf{a}_2, \mathbf{b}_2)$ and the vertical projections of the input points along these directions. The bottom row, Fig. 2.1(b), shows the comparison between these two pairs of directions. From the above discussion, we know that the quality of a pair of directions is indicated by the closeness of the projections ($\mathbf{x}_i^T\mathbf{a}$ and $\mathbf{y}_i^T\mathbf{b}$). Notice that the projections of the points are relatively closer to each other for the first pair of directions, *i.e.,* $\mathbf{x}_i$ is more closer to $\mathbf{y}_i$ when projected onto $\mathbf{a}_1$ and $\mathbf{b}_1$, respectively, compared to $\mathbf{a}_2$ and $\mathbf{b}_2$. This way, CCA searches for all possible pairs of directions and chooses the best pair.



(a) Shows two pairs of possible projection directions $(\mathbf{a}_1, \mathbf{b}_1)$ and $(\mathbf{a}_2, \mathbf{b}_2)$ and the vertical projections of the input points along these directions.



(b) Comparison of the quality of the two pairs of directions. Clearly, the projections of $\mathbf{x}_i$ is closer to $\mathbf{y}_i$ in the first pair of directions ($\mathbf{a}_1$ and $\mathbf{b}_1$). '0' indicates the origin.

Figure 2.1: Shows three pairs of points $(\mathbf{x}_i, \mathbf{y}_i)$ in two different (colored) two dimensional spaces. It also shows two possible pairs of directions and their compares their quality.

2.2.1 Optimization

Though optimizing the objective function shown in Eq. 2.4 is relatively straightforward [72], I will outline the process since it will be referred frequently in the rest of this dissertation. Let α and β be the scalar Lagrangian multipliers corresponding to the length constraints shown in Eq. 2.4. Then, the Lagrangian can be written as:

$$\mathcal{L} = \mathbf{a}^T C_{xy} \mathbf{b} - \alpha(\mathbf{a}^T C_{xx} \mathbf{a} - 1) - \beta(\mathbf{b}^T C_{yy} \mathbf{b} - 1) \tag{2.7}$$

I remind the notation that $C_{xx} = XX^T$, $C_{yy} = YY^T$ are the autocovariance matrices and $C_{xy} = XY^T$ is the cross-covariance matrix. Taking partial derivatives of $\mathcal{L}$ with respect to $\mathbf{a}$ and $\mathbf{b}$ setting them to zero gives:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{a}} = 0 \quad \Rightarrow \quad C_{xy}\, \mathbf{b} = \alpha\, C_{xx}\, \mathbf{a} \tag{2.8}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}} = 0 \quad \Rightarrow \quad C_{xy}^T\, \mathbf{a} = \beta\, C_{yy}\, \mathbf{b} \tag{2.9}$$

Now multiplying Eq. 2.8 with $\mathbf{a}^T$ and Eq. 2.9 with $\mathbf{b}^T$, subtracting them from each other and using the fact that $\mathbf{a}^T C_{xx} \mathbf{a} = 1$ and $\mathbf{b}^T C_{yy} \mathbf{b} = 1$ results in:

$$0 = \alpha\, \mathbf{a}^T C_{xx} \mathbf{a} - \beta\, \mathbf{b}^T C_{yy} \mathbf{b} \;\Rightarrow\; \alpha - \beta \;\Rightarrow\; \alpha = \beta \tag{2.10}$$

Now substituting β with α in Eq. 2.9, the equations can be rewritten as the following generalized eigenvalue problem:

$$\begin{pmatrix} \mathbf{0} & C_{xy} \\ C_{xy}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} = \alpha \begin{pmatrix} C_{xx} & \mathbf{0} \\ \mathbf{0} & C_{yy} \end{pmatrix} \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} \tag{2.11}$$

where $\mathbf{a}$ and $\mathbf{b}$ are the eigenvectors and α is the corresponding eigenvalue. In the regularized version, the monolingual covariance matrices are modified as follows $C_{xx} \leftarrow (1-\lambda)C_{xx} + \lambda I$ and $C_{yy} \leftarrow (1-\lambda)C_{yy} + \lambda I$ where I is an identity matrix of appropriate size.

2.2.2 Practical Considerations

The generalized eigenvector problem shown in Eq. 2.11 is of size $d_1 + d_2$. Moreover, solving the above generalized eigenvalue problem results in $\mathbf{a}$ and $\mathbf{b}$ that satisfy $\mathbf{a}^T C_{xx} \mathbf{a} + \mathbf{b}^T C_{yy} \mathbf{b} = 1$ which is different from the original length constraints shown in Eq. 2.4. There are two alternate approaches to solve for the eigensystem.

2.2.2.1 Implementation Order

By simple algebra, the same equations in Eq. 2.11 can be transformed into a smaller eigenvalue problem as follows:

$$\text{If } d_1 \leq d_2 \qquad C_{xx}^{-1} C_{xy} C_{yy}^{-1} C_{xy}^T \, \mathrm{a} = \alpha^2 \, \mathrm{a} \qquad \mathrm{b} = \frac{1}{\alpha} C_{yy}^{-1} C_{xy}^T \, \mathrm{a} \tag{2.12}$$

$$\text{else} \qquad C_{yy}^{-1} C_{xy}^{-T} C_{xx}^{-1} C_{xy} \, \mathrm{b} = \alpha^2 \, \mathrm{b} \qquad \mathrm{a} = \frac{1}{\alpha} C_{xx}^{-1} C_{xy} \, \mathrm{a} \tag{2.13}$$

The above formulation involves solving an eigenvalue problem of size d_1 or d_2 depending on whichever is smaller. As will be seen in Chapter 6, this simple check saves a lot of computation. But, it still requires computing the inverse of the covariance matrices. Moreover, the projection directions obtained by solving either of the above formulations, doesn't strictly follow the length constraints. For example, $\mathbf{a}$ obtained by solving Eq. 2.12 are normalized such that $\mathbf{a}^T \mathbf{a} = 1$ but $\mathbf{b}^T \mathbf{b} \neq 1$. So it is very important to handle normalization during the prediction time (as shown in Eq. 2.14).

2.2.2.2 CCA using Singular Value Decomposition (SVD)

Though the previous method solves a smaller eigenvalue problem it involves computing inverse matrices and solving an eigenvalue system. These methods doesn't scale well for large matrices. In this section, we will see how to compute the projection directions using singular value decomposition (SVD). Efficient and scalable techniques for solving SVD do exist. In particular, I will describe the method proposed in [78]. Let

$$C_{xx} = R_x^T R_x \qquad \text{and} \qquad C_{yy} = R_y^T R_y$$

where R_x and R_y are upper triangular matrices obtained by the Cholesky decomposition of the respective covariance matrices. Now, we set $\mathbf{l} = R_x \mathbf{a}$ and $\mathbf{m} = R_y \mathbf{b}$ and transform the optimization problem from finding $\mathbf{a}$, $\mathbf{b}$ into the problem of solving for $\mathbf{l}$ and $\mathbf{m}$. The transformed version of

the problem shown in Eq. 2.4 can be rewritten as:

$$\arg\max_{\mathbf{l},\mathbf{m}} \ \mathbf{l}^T R_x^{-T} C_{xy} R_y^{-1} \mathbf{m} \qquad \text{s.t. } \mathbf{l}^T\mathbf{l} = 1 \quad \text{and} \quad \mathbf{m}^T\mathbf{m} = 1$$

Now let $R_x^{-T} C_{xy} R_y^{-1} = USV^T$ where S is a diagonal matrix and U and V are unitary matrices (obtained by singular value decomposition). Using this decomposition, the objective can be rewritten as follows:

$$\arg\max_{\mathbf{l},\mathbf{m}} \ \sum_i S(i,i)\, \mathbf{l}^T \mathrm{u}_i \mathrm{v}_i^T \mathbf{m} \qquad \text{s.t. } \mathbf{l}^T\mathbf{l} = 1 \quad \text{and} \quad \mathbf{m}^T\mathbf{m} = 1$$

This maximum is achieved by setting $\mathbf{l} = \mathrm{u}_1 \Rightarrow \mathbf{a} = R_x^{-1}\mathrm{u}_1$ and $\mathbf{m} = \mathrm{v}_1 \Rightarrow \mathbf{b} = R_y^{-1}\mathrm{v}_1$. Since R_x and R_y are triangular matrices, **a** and **b** can be solved efficiently.

Without regularization : The problem can be further simplified when there is no regularization term at all. In that case, R_x and R_y can be obtained by using the QR decomposition of X^T and Y^T respectively, as follows: $Q_x R_x = X^T$ and $Q_y R_y^T = Y^T$ where Q_x, Q_y are unitary matrices and R_x, R_y are upper triangular matrices.

Furthermore, in this case, $R_x^{-T} C_{xy} R_y^{-1} = R_x^{-T} X Y^T R_y^{-1} = R_x^{-T} R_x^T Q_x^T Q_y R_y R_y^{-1} = Q_x^T Q_y$. So, we only need to compute the SVD of $Q_x^T Q_y$

2.2.3 Worked-Out Example

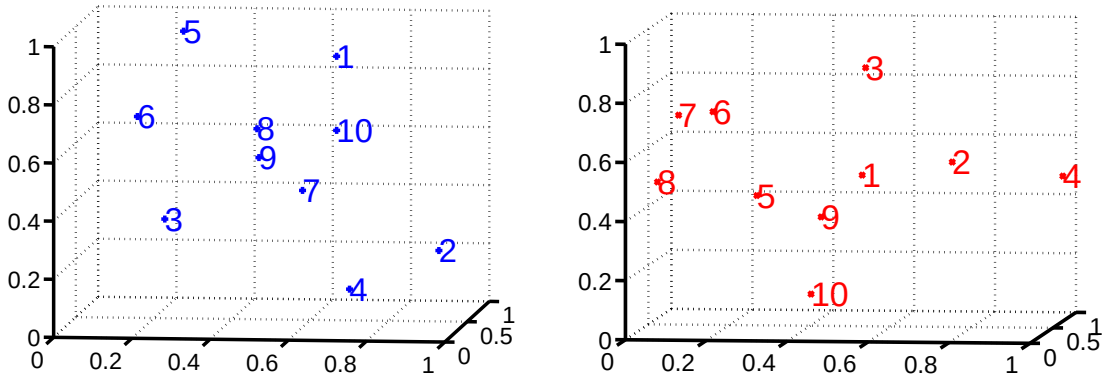
In this section, I will illustrate CCA with a numerical example. Let us assume that we have 10 pairs of points in two 3-dimensional spaces. Here I show 10 pairs of randomly generated points denoted by X (3×10) and Y (3×10). Though both the spaces are 3-dimensional, there is no relation between the dimensions of X and Y, one can as well assume that **x**'s are text documents and **y**'s are images. I choose three dimensional spaces for convenience.

$$X = \begin{pmatrix} 0.6358 & 0.7703 & 0.8699 \\ 0.9452 & 0.3502 & 0.2648 \\ 0.2089 & 0.6620 & 0.3181 \\ 0.7093 & 0.4162 & 0.1192 \\ 0.2362 & 0.8419 & 0.9398 \\ 0.1194 & 0.8329 & 0.6456 \\ 0.6073 & 0.2564 & 0.4795 \\ 0.4501 & 0.6135 & 0.6393 \\ 0.4587 & 0.5822 & 0.5447 \\ 0.6619 & 0.5407 & 0.6473 \end{pmatrix}^{\mathrm{T}} \quad \text{and} \quad Y = \begin{pmatrix} 0.5439 & 0.3658 & 0.5254 \\ 0.7210 & 0.7635 & 0.5303 \\ 0.5225 & 0.6279 & 0.8611 \\ 0.9937 & 0.7720 & 0.4849 \\ 0.2187 & 0.9329 & 0.3935 \\ 0.1058 & 0.9727 & 0.6714 \\ 0.1097 & 0.1920 & 0.7413 \\ 0.0636 & 0.1389 & 0.5201 \\ 0.4046 & 0.6963 & 0.3477 \\ 0.4484 & 0.0938 & 0.1500 \end{pmatrix}^{\mathrm{T}}$$

Fig. 2.2 shows these points represented in their respective 3-dimensional spaces. For distinction, we use blue and red colors for **x** and **y** points respectively. Notice that the relation between pairs of points $(\mathbf{x}_i, \mathbf{y}_i)$ is not clear in these original spaces.

Now, we will use CCA to find a lower dimensional space in which the correlations become clear. Using CCA as described in the previous section we find the following projection directions (I used Matlab's internal implementation *canoncorr* function for this example):

$$\mathbf{a}_1 = [\ -3.9219 \quad -6.6306 \quad 4.2631\ ]^T \qquad \text{and} \qquad \mathbf{b}_1 = [\ -3.4025 \quad 0.1268 \quad -0.5352\ ]^T$$



(a) Shows 10 points (represented as X) in the first 3-dimensional space.

(b) Shows 10 points (represented as Y) in a different first 3-dimensional space.

Figure 2.2: Shows three pairs of points $(\mathbf{x}_i, \mathbf{y}_i)$ in two different (colored) two dimensional spaces. It also shows two possible pairs of directions and their compares their quality.

$$\mathbf{a}_2 = [\ 3.0674 \quad -2.0333 \quad -3.4745\ ]^T \qquad \text{and} \qquad \mathbf{b}_2 = [\ 0.4055 \quad -2.0973 \quad -3.0312\ ]^T$$

Now, we form the projection matrices A and B (or mapping functions) with the above vectors as columns, respectively, as follows:

$$A = \begin{pmatrix} -3.9219 & 3.0674 \\ -5.6306 & -2.0333 \\ 4.2631 & 3.4745 \end{pmatrix} \qquad \text{and} \qquad B = \begin{pmatrix} -3.4025 & 0.4055 \\ 0.1268 & -2.0973 \\ -0.5352 & -3.0312 \end{pmatrix}$$

Subsequently, these projection directions (or mapping functions) map the input points into the lower dimensional subspace. Given a new observation $\mathbf{x}$, assuming that it is centered, it gets mapped to $A^T\mathbf{x}$ and, similarly, $\mathbf{y}$ gets mapped to $B^T\mathbf{y}$. Fig. 2.3 shows the training points after projecting into the common subspace. There are two crucial observations:

1. From this figure it is clear that the paired points $(\mathbf{x}_i, \mathbf{y}_i)$ lie closer to each other in this subspace. Because of this property, finding the aligned point $\mathbf{y}$ of a given input point $\mathbf{x}$ becomes nearest neighbour search in the lower dimensional subspace. We use this property, to find aligned pairs of the input points which I will briefly describe in the next section.

2. Notice that the mapping is far from being perfect. For example, consider the point $\mathbf{x}_8$ (the blue point numbered 8). Its closest red point is $\mathbf{y}_7$ and not $\mathbf{y}_8$. Similarly, $\mathbf{x}_7$ is closest to $\mathbf{y}_8$, in fact they almost coincide. In order to handle this, one can put additional constraints that, given $\mathbf{x}_i$, $\mathbf{y}_i$ should be closer to it than to any other point, which I will describe in Chapter 6.

2.2.4 Prediction

As described in the above example, we use the eigenvectors identified by CCA as columns to form the projection matrices A and B. In general, using all the eigenvectors is sub optimal and thus retaining top eigenvectors leads to better generalizability. The number of top eigenvectors is tuned based on the validation data set. Subsequently, these projection matrices are used to map unseen points in both the views into the lower dimensional subspace. Given a new pair of points

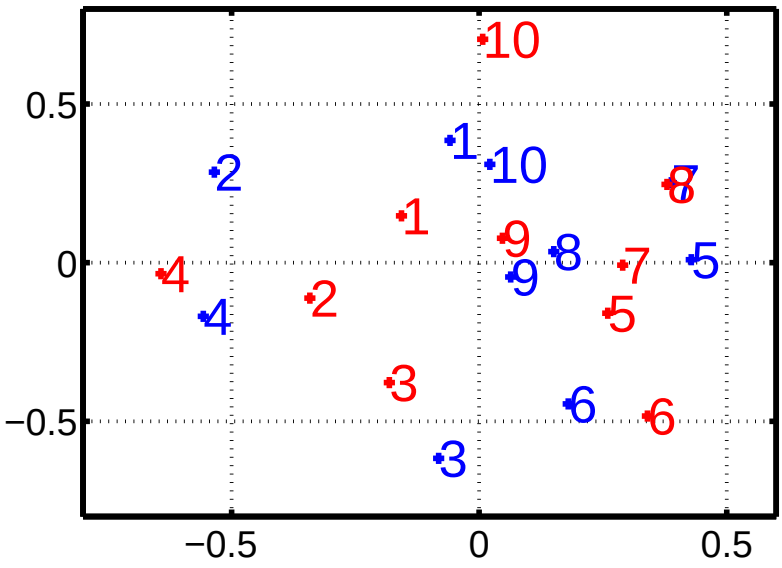


Figure 2.3: Shows the input points (blue points are **x**'s and red points are **y**'s) after projecting into the common 2-dimensional subspace. Notice that the correlation between the paired points is clear in the subspace compared to the original spaces.

x and **y**, their similarity is computed by first mapping them into the subspace and then computing the cosine similarity (or correlation) between their projections as follows:

$$\cos(A^T\mathbf{x}, B^T\mathbf{y}) = \frac{\mathbf{x}^T AB^T\mathbf{b}}{\sqrt{\mathbf{a}^T AA^T\mathbf{a}}\sqrt{\mathbf{b}^T BB^T\mathbf{b}}} \tag{2.14}$$

2.2.5 Generalization to Multiple Views

There are several generalizations of CCA to more than two views. These models assume that we have n observations in M views captured in terms of the data matrices $X^{(i)}$ $(d_i \times n)$ $\forall i = 1 \cdots n$. Notice that we have same number of observations in all languages, in other words we assume that each observation is available in all the views.

Under this assumption, [101] propose several modifications of CCA (SUMCOR, SSQCOR, MAX-VAR, GENVAR) to the multi-view data sets. And [174, 170] propose regularized versions of these models. To convey the main idea, I only present the objective functions of SUMCOR and SSQ-COR variants, but I refer the reader to the original paper for more details and other variants. Let $\mathbf{x}_i^{(j)} \in \mathbb{R}^{d_j}$ represent the $i^{th} = 1 \cdots n$ example in $j^{th} = 1 \cdots m$ view and let $X_j(d_j \times n)$ be the data matrix in the j^{th} view with $\mathbf{x}_i^{(j)}$ as columns. Moreover, let $\mathbf{a}^{(j)} \in \mathbb{R}^{d_j}$ be the projection directions in j^{th} view, then:

$$\text{SUMCOR} \qquad \underset{\mathbf{a}^{(j)}, j=1\cdots m}{\arg\max} \sum_{j,k,j\neq k} corr\Big({X_j}^T\mathbf{a}^{(j)}, {X_k}^T\mathbf{a}^{(k)}\Big) \tag{2.15}$$

$$\text{SSQCOR} \qquad \underset{\mathbf{a}^{(j)}, j=1\cdots m}{\arg\max} \sum_{j,k,j\neq k} \Big[corr\Big({X_j}^T\mathbf{a}^{(j)}, {X_k}^T\mathbf{a}^{(k)}\Big)\Big]^2 \tag{2.16}$$

The above mentioned approaches, consider the correlation between all the pair-wise views and maximize different combination of those correlation functions. On the other hand, Caroll [24] and Steenkamp *et al.* [166] propose a different type of generalization to more than two views. Let $Z\ (n \times k)$ be the representation of the observations in the k-dimensional subspace, then their objective functions is:

$$\arg\min_{Z,A_j} \text{ trace} \sum_{j=1}^{m} \Big(Z - {X_j}^T A_j\Big)^T \Big(Z - {X_j}^T A_j\Big) \qquad \text{s.t.} \qquad Z^T Z = I \tag{2.17}$$

However, all the above approaches assume that an observation exists in all the views. But in practice, some observations may be missing from one or more views. Modifications of the Caroll [24] have been proposed to handle the missing observations. These approaches include an additional indicator diagonal matrix K_j, for each view, which denotes if an observation is present in a view or not. Using this indicator matrix, the modified objective function can be written as follows [184]:

$$\arg\min_{Z,A_j} \text{ trace} \sum_{j=1}^{m} \Big(Z - {X_j}^T A_j\Big)^T K_j \Big(Z - {X_j}^T A_j\Big) \qquad \text{s.t.} \qquad Z^T \Big(\sum_j K_j\Big) Z = I$$

In this section, we saw how CCA models sets of multivariate variables and its generalizations for more than two views. In the reminder of this Chapter, I will briefly discuss a very closely related multivariate model called PLS that is developed around the same time as CCA. Though this material helps in gaining more insights into my contributions, this is not required and the reader can safely jump to the discussion Section (2.4) at the end of this Chapter.

2.3 Partial Least Squares (PLS)

Partial Least Squares (PLS) [191, 152, 79] is another technique to model sets of multivariate data and is originally developed in the field of chemometrics. Given n observations in two views (as described in Sec. 2.1), *i.e.*, $X\ (d_1 \times n)$ and $Y\ (d_2 \times n)$, PLS models the relation between these two data sets in terms of score vectors and loading matrices as follows:

$$X = PT + E \tag{2.18}$$

$$Y = QU + F \tag{2.19}$$

where $P\ (d_1 \times k)$ and $Q\ (d_2 \times k)$ are loading matrices, $T\ (k \times n)$ and $U\ (k \times n)$ are score matrices and E and F are error terms. I follow the description of PLS given in Rosipal and Krämer [152] very closely, but my description differs slightly because I use column vector notation. The PLS method is based on the nonlinear iterative partial least squares (NIPALS) algorithm [191] and solves the following problem [152]:

$$\big[cov(\mathbf{t}, \mathbf{u})\big]^2 = \max_{\|\mathbf{a}\|=\|\mathbf{b}\|=1} \big[cov(X^T\mathbf{a}, Y^T\mathbf{b})\big]^2 \tag{2.20}$$

where, as described previously, $cov(\mathbf{t}, \mathbf{u}) = \frac{\mathbf{t}^T\mathbf{u}}{n}$. NIPALS uses an iterative algorithm to solve for the $\mathbf{a}, \mathbf{b}, \mathbf{t}$ and $\mathbf{u}$ vectors as follows:

$$\begin{aligned} &1) \quad \mathbf{a} = \frac{X^T\mathbf{u}}{\mathbf{u}^T\mathbf{u}} && 4) \quad \mathbf{b} = \frac{Y\mathbf{t}}{\mathbf{t}^T\mathbf{t}} \\ &2) \quad \|\mathbf{a}\| \rightarrow 1 && 5) \quad \|\mathbf{b}\| \rightarrow 1 \\ &3) \quad \mathbf{t} = X^T\mathbf{a} && 6) \quad \mathbf{u} = Y^T\mathbf{b} \end{aligned}$$

By following one cycle of iterations (*i.e.,* steps 2 through 6 and 1) it can be shown that $\mathbf{a}$ is the eigenvector of XY^TYX^T [79]. To see this, let's say we have a current estimate of $\mathbf{a}$, then

$$\begin{aligned} \mathbf{a} &\leftarrow \frac{\mathbf{a}}{\mathbf{a}^T\mathbf{a}} && \text{step 2} \\ \mathbf{t} &= \frac{X^T\mathbf{a}}{\mathbf{a}^T\mathbf{a}} && \text{step 3} \\ \mathbf{b} &= \frac{YX^T\mathbf{a}}{\mathbf{a}^T\mathbf{a}\ \ \mathbf{t}^T\mathbf{t}} && \text{step 4} \\ \mathbf{b} &\leftarrow \frac{YX^T\mathbf{a}}{\mathbf{b}^T\mathbf{b}\ \ \mathbf{t}^T\mathbf{t}\ \ \mathbf{a}^T\mathbf{a}} && \text{step 5} \\ \mathbf{u} &= \frac{Y^TYX^T\mathbf{a}}{\mathbf{b}^T\mathbf{b}\ \ \mathbf{t}^T\mathbf{t}\ \ \mathbf{a}^T\mathbf{a}} && \text{step 6} \\ \mathbf{a} &= \frac{XY^TYX^T\mathbf{a}}{\mathbf{u}^T\mathbf{u}\ \ \mathbf{b}^T\mathbf{b}\ \ \mathbf{t}^T\mathbf{t}\ \ \mathbf{a}^T\mathbf{a}} && \text{step 1} \end{aligned}$$

The final iteration shows that $\mathbf{a}$ is the eigenvector of XY^TYX^T, where the constant in the denominator is the eigenvalue. After solving for $\mathbf{a}$, we can get $\mathbf{t} = X^T\mathbf{a}$, $\mathbf{b} = \frac{Y\mathbf{t}}{\mathbf{t}^T\mathbf{t}}$ and $\mathbf{u} = Y^T\mathbf{b}$.

Connection between CCA and PLS : Now, we will see the connection between the objective functions being optimized by CCA and PLS. As shown in Eq. 2.1, CCA maximizes:

$$\frac{\mathbf{a}^TXY^T\mathbf{b}}{\sqrt{\mathbf{a}^TXX^T\mathbf{a}}\sqrt{\mathbf{b}^TYY^T\mathbf{b}}} = corr(X^T\mathbf{a}, Y^T\mathbf{b}) = \frac{cov(X^T\mathbf{a}, Y^T\mathbf{b})}{\sqrt{var(X^T\mathbf{a})}\sqrt{var(Y^T\mathbf{b})}} \tag{2.21}$$

From the above Equation and Eq. 2.20, it is evident that CCA also considers the variance of the projection vectors ($X^T\mathbf{a}$ and $Y^T\mathbf{b}$) while PLS ignores the variance.

2.3.1 Orthonormalized Partial Least Squares (OPLS)

Orthonormalized Partial Least Squares (OPLS) [192] is a variant of PLS which considers variance in one of the two views making it invariant to the linear transformations in that view. OPLS computes the orthogonal score vectors for X by solving the following optimization problem [167]:

$$\arg\max_{\mathbf{a}} \ \ \mathbf{a}^TXY^TYX^T\mathbf{a} \qquad \text{s.t.} \qquad \mathbf{a}^TXX^T\mathbf{a} = 1 \tag{2.22}$$

Equivalence between CCA and OPLS : Sun *et al.* [167] investigate the connection between CCA and OPLS and show that the projection directions learnt by both these models differ only by a rotation, *i.e.,* , if A_{cca} $(d_1 \times k)$ and A_{opls} $(d_1 \times k)$ denote the projection directions identified by CCA

and OPLS, respectively, then $A_{cca} = RA_{opls}$ where R is a $d_1 \times d_1$ rotation matrix.

2.4 Discussion

In this Chapter, I reviewed the background material that is required for the rest of this dissertation. I also described PLS model and outlined its connection to CCA. Since both these models are very closely related to each other, I will use only CCA based baseline systems for comparison and hope that the results/contributions will hold for the other model as well. I end this Chapter with two important observations.

- In this Chapter, we saw multiple ways to solve the objective function of CCA. Though these approaches differ in the size of the eigenvalue problem that they solve, one might expect that they should give same eigenvalue solution. But they tend to give different solutions depending on the multiplicity of the eigenvalues. When multiple eigenvalues take the same value, then the corresponding eigenvectors are not unique and lie in a subspace. So any basis of this subspace is a valid set of eigenvectors. The different solutions obtained by these different methods can lead to differences in the accuracies of the downstream applications. So, for consistency, I use the method described in Sec. 2.2.2.1 whenever I need to solve the generalized eigenvalue problem.

- Second, the objective function of CCA can be decomposed as follows:

$$\begin{aligned}
\mathbf{a}^T XY^T \mathbf{b} &= \sum_{i=1}^{n} \langle \mathbf{x}_i, \mathbf{a} \rangle \langle \mathbf{y}_i, \mathbf{b} \rangle \\
&= \sum_{i=1}^{n} \Big(\sum_{l=1}^{d_1} \mathbf{x}_i(l)\mathbf{a}(l) \cdot \sum_{m=1}^{d_2} \mathbf{y}_i(m)\mathbf{b}(m) \Big) \\
&= \sum_{i=1}^{n} \Big(\sum_{l=1}^{d_1} \sum_{m=1}^{d_2} \mathbf{x}_i(l)\mathbf{a}(l)\ \mathbf{y}_i(m)\mathbf{b}(m) \Big) \\
&= \sum_{i=1}^{n} \Big(\sum_{l,m=1}^{d_1,d_2} w_{lm}\phi_i^{lm} \Big) \qquad (2.23)
\end{aligned}$$

  where $w_{lm} = \mathbf{a}(l)\mathbf{b}(m)$ and $\phi_i^{lm} = \mathbf{x}_i(l)\mathbf{y}_i(m)$. So, the objective can be rewritten as $\mathbf{a}^T XY^T \mathbf{b} = \sum_i \langle \mathbf{w}, \phi(\mathbf{x}_i, \mathbf{y}_i) \rangle$ where $\mathbf{w}$ is the weight vector and $\phi(\mathbf{x}_i, \mathbf{y}_i)$ is a vector of size $(d_1 \times d_2)$ and is given by the Kronecker product (or outer product) of the two feature vectors $\mathbf{x}_i$ and $\mathbf{y}_i$. In this form, the objective function bears resemblance to the classic linear boundary surface that is widely used in machine learning, except that the boundary is learnt in the joint feature space and the weight vector is constrained. This is the crucial observation for the approaches that I describe in the following Chapters. In specific, Chapter 4 introduces the idea of sparsifying the covariance matrices to learn better projection directions, which reduces to feature selection in the joint feature space. And in Chapter 6, I use this observation along with ideas from structured prediction literature to propose lower-dimensional discriminative reranking models.

Chapter 3

Multi-view Models

Multi-view data is ubiquitous. We have text being published in multiple languages; news being covered in multiple media such as audio, video and text; images of a same object take under different light conditions, and so on. This situation needs techniques that can exploit information available across the views. There has been plenty of work in modeling multi-view data. In this Chapter, I briefly review some of the relevant models. Since most of this dissertation deals with textual data sets, here, I only review the models that were applied to textual data sets. Moreover, I only describe the modeling part of these models, unless the solution makes it easier to understand its connection to the CCA. The models described in this Chapter are not directly relevant to this dissertation, but are included for completeness. The reader can safely skip this Chapter and still follow the rest of this dissertation.

Most of the models reviewed in this Chapter, assume a training data of n aligned pairs of objects in two views – often paired documents in different languages. Let $\mathbf{x}_i \in \mathbb{R}^{d_1}$ and $\mathbf{y}_i \in \mathbb{R}^{d_2}$ $(i = 1 \cdots n)$ be the vector representations of i^{th} aligned pair in both the views, respectively. Moreover, let X $(d_1 \times n)$ and Y $(d_2 \times n)$ be the data matrices with i^{th} column given by $\mathbf{x}_i$ and $\mathbf{y}_i$ respectively. Further more, assume that the data is centered (*i.e.*, mean is subtracted from each vector; $\mathbf{x}_i \leftarrow \mathbf{x}_i - \mu_x$ and $\mathbf{y}_i \leftarrow \mathbf{y}_i - \mu_y$). Finally, let $C_{xx} = XX^T$, $C_{yy} = YY^T$ denote the autocovariance matrices, and $C_{xy} = XY^T$ be the cross-covariance matrix. I try to be consistent with this notation across the models, but they often require variables that are specific to each model. In such cases, I explicitly define the notation within each section.

3.1 Extensions of Principal Component Analysis (PCA)

Principal component analysis (PCA) is one of the earliest known techniques for dimensionality reduction. Given a data set of n points, it tries to find a subspace which explains most of the variance within the data. There has been extensions of PCA to multi-view data. In this section, I briefly review the extensions of PCA to multi-view data.

3.1.1 Cross-lingual Latent Semantic Indexing (CL-LSI)

Latent semantic indexing (LSI) [53, 41] is a technique developed for information retrieval. It uses a document collection to find a latent representation which uncovers the semantic meaning of words. Inherently, it uses co-occurrence patterns of words within a document to find the latent representation. Queries and documents are mapped into this latent semantic representation, and all the documents that are closer to a given query, in the semantic space, are deemed as relevant

to the query and returned as its results. Later, LSI has been used in other NLP problems as well [140, 158, 159, 176].

Cross-lingual latent semantic indexing (CL-LSI) [48] is an extension of LSI to the bilingual setting. Given n-aligned document pairs in two languages, CL-LSI involves constructing new set of n artificial documents by appending the aligned documents from both the languages. As before, let $X(d_1 \times n)$ and $Y(d_2 \times n)$ be the bag-of-word representation of n-pairs of documents in both the languages. Then, CL-LSI involves constructing a new matrix Z of size $(d_1 + d_2) \times n$ by appending X and Y as follows:

$$Z = \begin{bmatrix} X_{(d_1 \times n)} \\ \\ Y_{(d_2 \times n)} \end{bmatrix} \tag{3.1}$$

Subsequently, it uses SVD to find a k-dimensional latent space as $Z = VSU^T$.

$$\underset{(d_1+d_2)\times n}{\begin{bmatrix} \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & & & & \cdot \\ \vdots & & Z & & \vdots \\ \cdot & & & & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \end{bmatrix}} = \underset{(d_1+d_2)\times k}{\begin{bmatrix} \cdot & \cdots & \cdot \\ \cdot & & \cdot \\ \vdots & V & \vdots \\ \cdot & & \cdot \\ \cdot & \cdots & \cdot \end{bmatrix}} \underset{(k\times k)}{\begin{bmatrix} \cdot & \cdots & \cdot \\ \vdots & S & \vdots \\ \cdot & \cdots & \cdot \end{bmatrix}} \underset{(k\times n)}{\begin{bmatrix} \cdot & \cdot & \cdots & \cdot & \cdot \\ \vdots & & U^T & & \vdots \\ \cdot & \cdot & \cdots & \cdot & \cdot \end{bmatrix}}$$

The matrix $V_{(d_1+d_2)\times k}$ represents the projection directions that map documents of both the languages in the low-dimensional space. Thus CL-LSI maps documents of both the languages into a common low-dimensional subspace.

Relation between CL-LSI and CCA : The matrices U and V are orthogonal matrices, *i.e.*, $U^TU = I$ and $V^TV = I$. Using this property, it can be shown that V is captures the eigenvectors of the matrix Z^TZ. That is,

$$ZZ^T\,V = V\,S^2 \tag{3.2}$$

$$\begin{pmatrix} C_{xx} & C_{xy} \\ C_{xy}^T & C_{yy} \end{pmatrix} V = V\,S^2 \tag{3.3}$$

Compare this form with the solution of CCA shown in Eq. 2.11. In mathematical terms, the difference is that, for CCA we use a generalized eigenvalue system and for CL-LSI we use a normal eigenvalue system. For easier understanding, lets consider the top eigenvector $\mathbf{v}_1$ and also decompose it as $\mathbf{v} = [\mathbf{a}^T \quad \mathbf{b}^T]^T$, where $\mathbf{a}$ and $\mathbf{b}$ are vectors of length d_1 and d_2 respectively. Now, the objective function being maximized by CL-LSI can be rewritten as:

$$\underset{\mathbf{v}_1}{\arg\max} \quad \mathbf{v}_1^T \begin{pmatrix} C_{xx} & C_{xy} \\ C_{xy}^T & C_{yy} \end{pmatrix} \mathbf{v}_1 \tag{3.4}$$

$$\underset{\mathbf{a},\mathbf{b}}{\arg\max} \quad \mathbf{a}^T C_{xx}\mathbf{a} + 2\,\mathbf{a}^T C_{xy}\mathbf{b} + \mathbf{b}^T C_{yy}\mathbf{b} \tag{3.5}$$

$$\text{s.t.} \quad \mathbf{a}^T\mathbf{a} + \mathbf{b}^T\mathbf{b} = 1 \tag{3.6}$$

Compare this objective with that of CCA (Eq. 2.1). Intuitively, CCA puts higher weight on a word pair $\mathbf{x}(i)$ and $\mathbf{y}(j)$, if the feature pair co-occurs frequently in the aligned document pairs *but* each of them are infrequent within the monolingual documents. Where as CL-LSI, puts a higher weight even if the features are frequent within the monolingual documents.

3.1.2 Oriented Principal Component Analysis (OPCA)

Oriented principal component analysis (OPCA) is a generalization of PCA in which a user can explicitly provide information about the importance of a feature pair [142]. Given a covariance matrix $C_{d\times d}$, PCA involves finding a vector $\mathbf{v}$ of size d which maximizes the Raleigh quotient given by:

$$\frac{\mathbf{v}^T C \mathbf{v}}{\mathbf{v}^T \mathbf{v}} \tag{3.7}$$

It can be solved using the eigenvalue problem $C\mathbf{v} = \lambda\mathbf{v}$. OPCA generalizes PCA, in the sense that it takes a signal matrix S and a noise matrix N and maximizes the generalized Rayleigh quotient given by:

$$\frac{\mathbf{v}^T S \mathbf{v}}{\mathbf{v}^T N \mathbf{v}} \tag{3.8}$$

Its solution is achieved by solving the generalized eigenvalue problem $S\mathbf{v} = \lambda N\mathbf{v}$.

In [142], the authors use OPCA to find an interlingual representation. The authors first build a composite feature space by appending the feature spaces of all the views. Let there be M views and d_m be the size of the feature space in m^{th} view, then the composite feature space is of $\sum_m d_m$. Notice that, in some situations the features may be shared across multiple views. For *e.g.*, if we consider a bilingual corpus of two languages that share the script, such as English and Spanish, then some of the words have same surface form in both these languages and hence these features will be shared across multiple views. Under such scenarios, the size of the composite feature space will be less than the sum of the individual feature space sizes.

Let D_m represent the data from m^{th} view in this composite feature space. If we assume that there are no shared features and that there are n documents, then this matrix will be of size $\sum_m d_m \times n$. Let $\boldsymbol{\mu}$ be the mean of the data from all the views. Then, the authors suggest using the following signal and noise matrices:

$$S = \sum_m \frac{D_m D_m^T}{n} - \boldsymbol{\mu}^T\boldsymbol{\mu} \tag{3.9}$$

$$N = \sum_m \frac{(D_m - D)(D_m - D)^T}{n} \tag{3.10}$$

where D is the mean data matrix across all the views, *i.e.*, $D = \frac{1}{M}\sum_m D_m$.

Equivalence between OPCA and CCA : For simplicity, assume that there are only two views and also assume that the data in both the views is centered, as we often center the data for CCA, *i.e.*, $\boldsymbol{\mu} = \mathbf{0}$. Maximizing generalized Raleigh quotient becomes:

$$\arg\max_{\mathbf{v}} \quad \frac{\mathbf{v}^T\Big(D_1D_1^T + D_2D_2^T\Big)\mathbf{v}}{\mathbf{v}^T\Big(D_1D_1^T + D_2D_2^T - D_1D_2^T - D_2D_1^T\Big)\mathbf{v}} \tag{3.11}$$

$$\arg\max_{\mathbf{v}} \quad \frac{1}{1 - \frac{\mathbf{v}^T(D_1D_2^T + D_2D_1^T)\mathbf{v}}{\mathbf{v}^T(D_1D_1^T + D_2D_2^T)\mathbf{v}}} \tag{3.12}$$

$$\arg\max_{\mathbf{v}} \quad \frac{\mathbf{v}^T(D_1D_2^T + D_2D_1^T)\mathbf{v}}{\mathbf{v}^T(D_1D_1^T + D_2D_2^T)\mathbf{v}} \tag{3.13}$$

Under the assumption that there are no shared features:

$$D_1 = \begin{bmatrix} X_{(d_1 \times n)} \\ \mathbf{0}_{(d_2 \times n)} \end{bmatrix} \qquad \text{and} \qquad D_2 = \begin{bmatrix} \mathbf{0}_{(d_1 \times n)} \\ Y_{(d_2 \times n)} \end{bmatrix} \tag{3.14}$$

Moreover, let $\mathbf{v} = [\mathbf{a}^T \quad \mathbf{b}^T]^T$. Then the objective function further reduces to:

$$\underset{\mathbf{a},\mathbf{b}}{\arg\max} \quad \frac{\mathbf{a}^T X Y^T \mathbf{b}}{\mathbf{a}^T X X^T \mathbf{a} + \mathbf{b}^T Y Y^T \mathbf{b}} \tag{3.15}$$

Comparing this objective function with that of CCA (Eq. 2.1), it is clear that these objective functions differ slightly in the denominator. I want to emphasize that this result holds under the assumption that there are no shared features, which is often the setting that I address in this dissertation. But, when the features are shared across multiple views, OPCA behaves differently in the sense that it forces the feature weight of a shared feature to be same across the views.

3.2 Spectral Embedding

All the above formulations, including CCA, lead to a solution of the form of an eigenvalue problem. Another line of research that also leads to solutions of this form is the spectral learning literature. In this section, I describe couple of spectral embedding techniques to find lower dimensional embeddings that are very similar to that of CCA.

3.2.1 Laplacian Eigenmaps

Laplacian eigenmaps [15] is a technique to reduce the dimensionality of n points in a single view. Though it operates on single view data, I discuss it here because it can be extended to multi-view data easily. Given n points $\mathbf{x}_i \in \mathbb{R}^{d_1}$ $i = 1 \cdots n$, it finds a low-dimensional subspace that preserves the local structure among the points. This technique first constructs a similarity matrix W of size $n \times n$ using an appropriate kernel function. Similar to CCA, it is easier to discuss this technique in the special case of finding an uni-dimensional embedding. It is trivial to extend it to find a k-dimensional embedding of the points. Let u_i represent the one-dimensional representation of the point $\mathbf{x}_i$, then Laplacian eigenmaps minimizes the following objective function:

$$\min \frac{1}{2} \sum_{i,j=1}^{n} W(i,j) \Big(u_i - u_j \Big)^2 = \mathbf{u}^T L \mathbf{u} \tag{3.16}$$

where $\mathbf{u} = [u_1 \cdots u_n]^T$ and $L = D - W$ is the Laplacian corresponding to the weight matrix W where D is a diagonal matrix with $D(i,i) = \sum_j W(i,j)$. In order to ensure that central points,*i.e.*, points with a high $D(i,i)$ value, are accurate embedded, they impose the constraint $\mathbf{u}^T D \mathbf{u} = 1$. The final optimization problem is given by:

$$\underset{\mathbf{u}}{\arg\min} \ \mathbf{u}^T L \mathbf{u} \qquad \text{s.t. } \mathbf{u}^T D \mathbf{u} = 1 \tag{3.17}$$

It can be solved using the generalized eigenvalue problem $L\mathbf{u} = \lambda D\mathbf{u}$. This problem admits multiple solutions in terms of different eigenvectors. The bottom k eigenvectors (with minimum

eigenvalue) are used to get the low k-dimensional embedding of the data points.

This technique, finds a low dimensional embedding of the given n points. But it doesn't directly tell us how to find a low dimensional embedding of an unseen point. One way is to find the neighbouring training examples of the unseen point, using the same kernel function used to compute W. Then, use the low dimensional embedding of the neighbouring points to find the embedding of the unseen point. This is very similar to the reconstruction step of locality linear embedding (LLE) [153, 30]. Another way is to make certain assumptions about the underlying probability distribution of the points and generalize using laplace Beltrami operator [194].

3.2.2 Locality Preserving Projections (LPP)

Locality preserving projections (LPP) [75] is another technique to find a low-dimensional representation of observations using spectral embedding. This technique is very similar to the Laplacian eigenmaps that we discussed in the previous section, but it generalizes to unseen examples as well. Originally, the technique was proposed to data from a single view, but it can be easily extended to data from multiple views. I first describe this technique in the single view setting and then show its connection to CCA.

Given the data $X_{(d_1 \times n)} = [\mathbf{x}_1 \cdots \mathbf{x}_n]$, similar to Laplacian eigenmaps, LPP constructs a weight matrix W of size $n \times n$. But it differs from Laplacian eigenmaps in the way the low-dimensional embedding is found. It assumes that the low dimensional representation is obtained by a linear transformation of the original point, *i.e.*, $u_i \leftarrow \mathbf{a}^T\mathbf{x}_i$ where $\mathbf{a}$ is a transformation that maps data point to a line. Let $\mathbf{u}$ be $[u_1 \cdots u_n]^T = X^T\mathbf{a}$. Then the optimization problem shown in Eq. 3.17 reduces to:

$$\arg\min_{\mathbf{a}} \ \mathbf{a}^T X L X^T \mathbf{a} \qquad \text{s.t.} \quad \mathbf{a}^T X D X^T \mathbf{a} = 1 \tag{3.18}$$

It can be solved using the generalized eigenvalue problem $XLX^T\mathbf{a} = \lambda XDX^T\mathbf{a}$. Again, this eigenvalue problem admits multiple solutions. The bottom k eigenvectors (with minimum eigenvalue) are used as columns to form the transformation matrix A of size $d_1 \times k$ which is used to transform an unseen point $\mathbf{x}_j$ into the low k-dimensional representation using $A^T\mathbf{x}_j$.

Generalization of CCA using spectral embedding : LPP, as presented here and also as discussed in the original paper, is applied to data from a single view. But it can be extended to multi-view data. In fact, by appropriately defining the feature space and the weight matrix of the Laplacian L, CCA can be seen as a specific instance of LPP. So, one can use LPP to generalize CCA. Here I describe the treatment of LPP to data from two views, but it can be analogously extended to data from more than two views.

First we form a composite feature space by appending the feature spaces of the two views. In the composite feature space, the data becomes:

$$Z = \begin{bmatrix} X & \mathbf{0} \\ \mathbf{0} & Y \end{bmatrix}_{(d_1+d_2)\times 2n} ; \ \mathbf{v} = \begin{bmatrix} \mathbf{a}_{(d_1\times 1)} \\ \mathbf{b}_{(d_2\times 1)} \end{bmatrix}$$
$$W = \begin{bmatrix} \mathbf{0} & I \\ I & \mathbf{0} \end{bmatrix}_{2n\times 2n} \Rightarrow D = \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix}_{2n\times 2n}$$

and let $\mathbf{u} = \mathbf{v}^T Z$ *i.e.*, $\mathbf{u}(i) = \mathbf{v}^T\mathbf{z}_i \ i = 1 \cdots 2n$ where $\mathbf{z}_i$ is the i^{th} column of the Z matrix. Using

these definitions the objective function of LPP becomes:

$$\begin{aligned}
\mathbf{u}^T L\mathbf{u} &= \frac{1}{2}\sum_{i,j=1}^{2n} W(i,j)(\mathbf{u}(i)-\mathbf{u}(j))^2 = \frac{1}{2}\sum_{i,j=1}^{2n} W(i,j)(\mathbf{v}^T\mathbf{z}_i - \mathbf{v}^T\mathbf{z}_j)^2 \\
&= \frac{1}{2}\Big(\sum_{i=1}^{n}(\mathbf{v}^T\mathbf{z}_i - \mathbf{v}^T\mathbf{z}_{i+n})^2 + \sum_{i=1+1}^{2n}(\mathbf{v}^T\mathbf{z}_i - \mathbf{v}^T\mathbf{z}_{i-n})^2\Big) \\
&= \frac{1}{2}\Big(\sum_{i=1}^{n}(\mathrm{a}^T\mathbf{x}_i - \mathrm{b}^T\mathbf{y}_i)^2 + \sum_{i=1}^{n}(\mathrm{b}^T\mathbf{y}_i - \mathrm{a}^T\mathbf{x}_i)^2\Big) \\
&= \sum_{i=1}^{n}(\mathrm{a}^T\mathbf{x}_i - \mathrm{b}^T\mathbf{y}_i)^2 = \|X^T\mathrm{a} - Y^T\mathrm{b}\|^2 \qquad (3.19)
\end{aligned}$$

Similarly, $\mathbf{u}^T D\mathbf{u} = 1 \Rightarrow \mathbf{a}^T XX^T\mathbf{a} + \mathbf{b}^T YY^T\mathbf{b} = 1$. This is very similar to the optimization function of CCA shown in Eq. 2.5, except the minor difference in the norm constraints. An interesting outcome of this reduction is that, by using the same composite feature space but using a different weight matrix we can generalize CCA. As we saw in Sec. 2.2, CCA utilizes only pairs of examples. But in many scenarios, there exists a many-to-many alignment between objects from different views, but the strength of each association may be different. By using this setup, we can generalize CCA to these situations and the only modification needed is to define the similarity matrix W appropriately.

3.2.3 Multi-view Spectral Embedding

In this section, I briefly mention the multi-view extensions of spectral embedding. Here, I would like to point out that the motivation for finding low-dimensional embedding in these models is completely different from that of CCA or the other techniques that I describe in this Chapter. CCA aims to find a subspace where aligned points lie close to each other. Where as spectral embedding, finds an embedding that preserves the structure of the data points within a view[1]. Given n points in M views, multi-view extensions of spectral embedding, first, find the structure of n points with in each view *separately* and then tries to find a pattern that is consistent among all the M structures. This is a widely used technique for clustering n observations in multiple views. The first multi-view extension of spectral embedding called co-clustering is proposed in [117] and several modifications of it are proposed later [193, 107].

3.2.4 Multi-view Hashing

Recently, hashing has gained lot of popularity due to the ever growing data. The idea is to map high dimensional data points into a smaller hash code, such that the points that are close in their original space get mapped to the same hash code. Since the hash code is very small in length, compared to the size of the original space, it is easier and efficient to do a nearest neighbour search in terms of the hash codes. In [194], the authors propose a spectral hashing technique to learn the

[1]Although by appropriately defining the weight matrix and the feature space we can show that CCA becomes a specific instance of spectral embedding, as we saw in the previous Section, that is not the primary motivation for the spectral embedding literature.

hash codes based on the similarity graph constructed between the data points. This was later extended to multiple views [108]. In this section, I quickly review the multi-view extension.

Let $\mathbf{x}_i^{(m)} \in \mathbb{R}^{d_m}\ i = 1 \cdots n$ and $m = 1 \cdots M$ be n objects in M views and X_m be a m^{th} view data matrix of size $d_m \times n$ obtained by using $\mathbf{x}_i^{(m)}$ as the i^{th} column. Let $\vec{g}^{(m)} : \mathbf{x}_i^{(m)} \rightarrow \{-1, 1\}^k$ be a hash function for m^{th} view that maps data point to its hash code. Let $\mathbf{h}_i^{(m)}$ denote the hash code of $\mathbf{x}_i^{(m)}$, *i.e.,* $\mathbf{h}_i^{(m)} \leftarrow \vec{g}^{(m)}\big(\mathbf{x}_i^{(m)}\big)$. Moreover, let $d\big(\mathbf{h}_i, \mathbf{h}_j\big)$ denote the hamming distance between the two hash codes. Then the hamming distance between the hash codes of two points $\mathbf{x}_i$ and $\mathbf{x}_j$, which is aggregated over all the views, is defined as:

$$d_{ij} = \sum_{m=1}^{M} d\Big(\mathbf{h}_i^{(m)}, \mathbf{h}_j^{(m)}\Big) + \sum_{m=1}^{M} \sum_{m'>m}^{M} d\Big(\mathbf{h}_i^{(m)}, \mathbf{h}_j^{(m')}\Big) \tag{3.20}$$

Given a $n \times n$ similarity matrix W between pairs of data points, learning the hash functions is formulated as the following optimization problem:

$$\begin{aligned} \min \quad & \bar{d} = \sum_{i,j=1}^{n} W(i,j)\, d_{ij} \\ \text{s.t.} \quad & H_m e = 0, \;\; H_m(i,j) = \{-1, 1\} \;\; m = 1 \cdots M \\ & \frac{1}{Mn} \sum_{m=1}^{M} H_m H_m^T = I \end{aligned} \tag{3.21}$$

where e is a unity vector of length n, I is an identity matrix of appropriate size and H_m is a matrix of size $k \times n$ formed by using $\mathbf{h}_i^{(m)}$ as the i^{th} column. This is an integer programming problem and is NP hard for $M > 1$ [194]. So it is relaxed such that $\mathbf{h}_i^{(m)} \in \mathbb{R}^k$. Further more, they assume that $\mathbf{h}_i^{(m)} \leftarrow A_m^T \mathbf{x}_i^{(m)}$ where A_m is a $d_m \times k$ matrix. The relaxed optimization problem becomes:

$$\begin{aligned} \min \quad & \bar{d} = \sum_{i,j=1}^{n} W(i,j)\, d_{ij} \\ \text{s.t.} \quad & A_m^T X_m e = 0, \;\; m = 1 \cdots M \\ & \frac{1}{Mn} \sum_{m=1}^{M} A_m^T X_m X_m^T A_m = I \end{aligned} \tag{3.22}$$

The solution of this optimization function is given by:

$$X_m L' X_m^T A_m - \sum_{m' \neq m}^{M} X_m W X_{m'}^T A_{m'} = X_m X_m^T A_m \Lambda \tag{3.23}$$

where $L' = 2L + (M-1)D$, D is a diagonal matrix with $D(i,i) = \sum_j W(i,j)$, $L = D - W$ is the Laplacian and Λ is a $k \times k$ diagonal matrix of eigenvalues.

Equivalence to CCA : Consider the setting where $M = 2$ and $W = I$. In this case, $L = 0$ and $L' = I$. In order to be consistent with the earlier notation, I replace $X \leftarrow X_1$, $Y \leftarrow X_2$, $A \leftarrow A_1$

and $B \leftarrow A_2$. The solution reduces to:

$$\begin{aligned} XY^TB &= XX^TA\Lambda' \\ YX^TA &= YY^TB\Lambda' \end{aligned} \tag{3.24}$$

where $\Lambda' = I - \Lambda$. Now rewriting these equations into a matrix notation:

$$\begin{pmatrix} \mathbf{0} & C_{xy} \\ C_{xy}^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} C_{xx} & \mathbf{0} \\ \mathbf{0} & C_{yy} \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} \Lambda' \tag{3.25}$$

This is same as the solution of CCA as shown in Eq. 2.11.

3.3 Iterative Approaches

In all the approaches that we discussed till now, the objective function is convex and the final solution turned out to be some variant of eigenvalue problem. In the approaches that we see in this section and in the rest of this Chapter, the objective function is non-convex and hence it is not possible to solve them optimally. So, we usually resort to an iterative approach to solve the optimization problem. In this section, I describe two such approaches.

3.3.1 Supervised Semantic Indexing

Supervised semantic indexing [5, 6] is a technique to rank text documents in response to a given query. Though, queries and documents are usually represented as vectors over the same vocabulary, it is observed that the query and document languages are very different. So, it is advantageous to give different treatment for the same word depending on if it is in the query or the document [56]. Supervised semantic indexing treats the queries and documents as if they are from different views. Given a query $\mathbf{x} \in \mathbb{R}^{d_1}$ and a document $\mathbf{y} \in R^{d_2}$, their model scores the query-document pair using:

$$f(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T A\, \mathbf{y} \tag{3.26}$$

where A is a $d_1 \times d_2$ matrix of parameters. Notice that, the model assumes query and document are in a different space, so it directly extends to scenarios where a query can be text and a document can be an image (represented as a feature vector). Usually, a low-rank assumption is enforced on the parameter matrix, *i.e.,*

$$A = U^TV + I \tag{3.27}$$

where U and V are $d_1 \times k$ and $d_2 \times k$ matrices respectively. These parameters are learnt using a sub gradient ascent algorithm [6].

3.3.2 Multi-view Neighbourhood Preserving Projections (Multi-view NPP)

Most of the techniques described till now, focus mainly on minimizing the distance between the aligned pairs of points $(\mathbf{x}_i, \mathbf{y}_i)$ in the low-dimensional subspace. But these techniques do not explicitly require that the projections of $\mathbf{x}_i$ and $\mathbf{y}_j$ $(j \neq i)$ be as far as possible. Techniques like LPP and Multi-view hashing allow us to explicitly model these pairs by arbitrarily changing the weight matrix W. But still, the functional form is same for aligned and unaligned pairs of points, in the

sense we are using euclidean distance between them. In this subsection, we will see a technique called multi-view neighbourhood preserving projections (Multi-view NPP) [144] which treats the similar (aligned) and dis-similar (unaligned) points differently.

Given a point $\mathbf{x}_i$, let $\mathcal{N}_i$ represent the set of points from the second view that are similar (neighbourhood) to $\mathbf{x}_i$. Let $\vec{g}^{(x)}(\mathbf{x}_i) = A^T\mathbf{x}_i$ and $\vec{g}^{(y)}(\mathbf{y}_j) = B^T\mathbf{y}_j$ be the functions that map points from the original representation to the low k-dimensional subspace, where A and B are matrices of size $d_1 \times k$ and $d_2 \times k$ respectively. Then Multi-view NPP minimizes the following loss function:

$$\mathcal{L}(A, B) = \sum_{i,j=1}^{n} \mathcal{L}_{i,j}\big(\mathbf{x}_i, \mathbf{y}_j\big) + \eta\, \Omega(A) + \gamma\, \Omega(B) \tag{3.28}$$

where $\mathcal{L}_{i,j}$ is the loss function for the pair $\mathbf{x}_i$ and $\mathbf{y}_j$, $\Omega(.)$ is a regularizer and η and γ are the weights for the regularization terms on A and B respectively. The loss function for the pair $\mathcal{L}_{i,j}$ is given as:

$$\begin{aligned}
\mathcal{L}_{i,j}(\mathbf{x}_i, \mathbf{y}_j) &= \frac{1}{2}\mathrm{I}_{[\mathbf{y}_j \in \mathcal{N}_i]} \times \mathcal{L}^1_{i,j} + \frac{1}{2}\Big(1 - \mathrm{I}_{[\mathbf{y}_j \in \mathcal{N}_i]}\Big)\mathcal{L}^2_{i,j} \\
\mathcal{L}^1_{i,j} &= \Big\|\vec{g}^{(x)}(\mathbf{x}_i) - \vec{g}^{(y)}(\mathbf{y}_j)\Big\|^2 \\
\mathcal{L}^2_{i,j}(d) &= \begin{cases} \frac{-d^2 + a\lambda^2}{2} & \text{if } 0 \le |d| < \lambda \\ \frac{\big(|d| - a\lambda\big)^2}{2(a-1)} & \text{if } \lambda \le |d| \le a\lambda \\ 0 & \text{if } |d| \ge a\lambda \end{cases}
\end{aligned} \tag{3.29}$$

where $\mathrm{I}_{[\mathbf{y}_j \in \mathcal{N}_i]}$ is an indicator function which evaluates to 1 if $\mathbf{y}_j$ belongs to the set $\mathcal{N}_i$, $d = \big\|\vec{g}^{(x)}(\mathbf{x}_i) - \vec{g}^{(y)}(\mathbf{y}_j)\big\|$, a and λ are hyper-parameters. Intuitively, if both the points lie in the same neighbourhood then it tries to minimize the distance between their projections. If the points belong to different neighbourhood and if they are not separated by at least $a\lambda$ then it tries to separate their projections. Notice that $\mathcal{L}^2_{i,j}$ is a continuous and differentiable function. The objective function is minimized using concave convex procedure [144].

3.4 Neural Networks

Neural networks is a powerful tool to model the non-linear relationship between input and output variables. Depending on the structure of the network and the error function, it can subsume various models, for *e.g.*, PCA [9]. In this section, I first describe a simple one hidden layer network which, when used to minimize Mahalnobis distance, solves the same problem as that of CCA. Then, I move on to a different model called S2Net [195] which is shown to perform better than CCA.

3.4.1 One Hidden Layer Linear Network and CCA

In [63], Ghahramani shows the connection between CCA and a one hidden layer linear network. Consider a one hidden layer linear bottle-neck network which takes $\mathbf{x}_i \in \mathbb{R}^{d_1}$ as input and tries to predict $\mathbf{y}_i \in \mathbb{R}^{d_2}$ as the output. If the hidden layer contains k nodes, where $k < \min\{d_1, d_2\}$, then the relationship between the input and output vectors can be written as $\widehat{\mathbf{y}}_i = BA^T\mathbf{x}_i$, where

A and B are matrices of sizes $d_1 \times k$ and $d_2 \times k$ respectively. These matrices denote the weights of the edges pointing towards and away from the bottle-neck layer respectively. Assume that the data is centered. Moreover, consider minimizing the Mahalnobis distance between the network prediction ($\widehat{\mathbf{y}}_i$) and the true output $\mathbf{y}_i$, using C_{yy} as the covariance matrix. In this case, the expected squared error is:

$$\begin{aligned}
\mathcal{E} &= \frac{1}{2}\sum_i \big(\mathbf{y}_i - \widehat{\mathbf{y}}_i\big)^T C_{yy}^{-1}\big(\mathbf{y}_i - \widehat{\mathbf{y}}_i\big) \\
&= \frac{1}{2}\sum_i \big(\mathbf{y}_i - BA^T\mathbf{x}_i\big)^T C_{yy}^{-1}\big(\mathbf{y}_i - BA^T\mathbf{x}_i\big) \\
&= \frac{1}{2}\sum_i \Big(\mathbf{y}_i^T C_{yy}^{-1}\mathbf{y}_i - 2\,\mathbf{y}_i^T C_{yy}^{-1}BA^T\mathbf{x}_i + \mathbf{x}_i^T AB^T C_{yy}^{-1}BA^T\mathbf{x}_i\Big) \\
&= \frac{1}{2}\sum_i \Big(\text{trace}\big(\mathbf{y}_i^T C_{yy}^{-1}\mathbf{y}_i\big) - 2\,\text{trace}\big(\mathbf{y}_i^T C_{yy}^{-1}BA^T\mathbf{x}_i\big) + \text{trace}\big(\mathbf{x}_i^T AB^T C_{yy}^{-1}BA^T\mathbf{x}_i\big)\Big) \\
&= \frac{1}{2}\sum_i \Big(\text{trace}\big(C_{yy}^{-1}\mathbf{y}_i\mathbf{y}_i^T\big) - 2\,\text{trace}\big(C_{yy}^{-1}BA^T\mathbf{x}_i\mathbf{y}_i^T\big) + \text{trace}\big(AB^T C_{yy}^{-1}BA^T\mathbf{x}_i\mathbf{x}_i^T\big)\Big) \\
&= \frac{1}{2}\text{trace}\big(C_{yy}^{-1}\sum_i \mathbf{y}_i\mathbf{y}_i^T\big) - 2\,\text{trace}\big(C_{yy}^{-1}BA^T\sum_i \mathbf{x}_i\mathbf{y}_i^T\big) + \text{trace}\big(AB^T C_{yy}^{-1}BA^T\sum_i \mathbf{x}_i\mathbf{x}_i^T\big) \\
&= \frac{1}{2}\,\text{trace}\big(C_{yy}^{-1}C_{yy}\big) - \text{trace}\big(C_{yy}^{-1}BA^T C_{xy}\big) + \frac{1}{2}\,\text{trace}\big(AB^T C_{yy}^{-1}BA^T C_{xx}\big) \\
&= \frac{1}{2}\,\text{trace}\big(I\big) - \text{trace}\big(A^T C_{xy}C_{yy}^{-1}B\big) + \frac{1}{2}\,\text{trace}\big(A^T C_{xx}AB^T C_{yy}^{-1}B\big) \\
&= \frac{d_2}{2} - \text{trace}\big(L^T C_{xx}^{-1/2}C_{xy}C_{yy}^{-1/2}U\big) + \frac{1}{2}\,\text{trace}\big(L^T LU^T U\big) \qquad (3.30)
\end{aligned}$$

where $L = C_{xx}^{1/2}A$ and $U = C_{yy}^{-1/2}B$. Now, consider the case where $k = 1$. In this case, L and U reduce to vectors $\mathbf{l}$ and $\mathbf{u}$. For clarity, consider that these vectors are unit vectors and their magnitudes are given by λ_l and λ_u. Then the error function becomes:

$$\mathcal{E} = \frac{d_2}{2} - \lambda_l\lambda_u\mathbf{l}^T C_{xx}^{-1/2}C_{xy}C_{yy}^{-1/2}\mathbf{u} + \frac{1}{2}\lambda_l^2\lambda_u^2 \qquad (3.31)$$

The minimum of this function is achieved when $\mathbf{l}$ and $\mathbf{u}$ are left and right singular vectors of $C = C_{xx}^{-1/2}C_{xy}C_{yy}^{-1/2}$ with largest singular value. To obtain these directions, we would do a SVD of the C and use the left and right singular vectors. Notice that this is same as the solution of CCA using SVD as discussed in Sec. 2.2.2.2.

3.4.2 Siamese Neural Network (S2Net)

In the previous section, we saw a one hidden layer neural network designed to show the connection between neural networks and CCA. In this section, I describe a practical model called Siamese Neural Network (S2Net) [195]. The proposed model does not use non-linearity, but as the authors pointed out it can be easily modified to learn non-linear mapping between the views. Here, I discuss the model as it was originally proposed [195].

Similar to OPCA in Sec. 3.1.2, S2Net first builds a composite feature space by appending the feature spaces of both the views. When there are two views and if d_1 and d_2 represent the sizes of

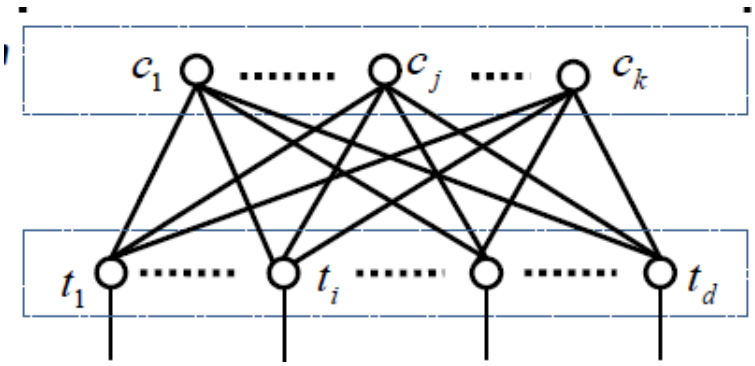


Figure 3.1: Architecture of S2Net model. The bottom layer is the input layer and the top layer is the decoding layer. The outputs are fed into an error function.

the feature space in both the views, then the composite feature space will be of size at most $d_1 + d_2$. It could be less if features are shared across the views. Let $\phi_{(.)}$ denote the representation of a point in the composite feature space, *i.e.*,

$$\phi_{\mathbf{x}_i} = \begin{bmatrix} \mathbf{x}_i^T & \mathbf{0}^T \end{bmatrix}^T \qquad \phi_{\mathbf{y}_j} = \begin{bmatrix} \mathbf{0}^T & \mathbf{y}_j^T \end{bmatrix}^T \tag{3.32}$$

S2Net is a two layered network as shown in the Fig. 3.1. The input layer corresponds to the composite feature vector of a point and the output layer is its low dimensional embedding. Let A represent the parameters of the network and $\vec{g}(\phi_{(.)})$ be the output of the network for the input vector $\phi_{(.)}$. Given an aligned pair $(\mathbf{x}_i, \mathbf{y}_i)$ and another point $\mathbf{y}_j$ such that $j \neq i$, then S2Net minimizes the logistic loss defined as:

$$L(\Delta) = \log(1 + \exp(-\gamma\Delta)) \qquad \text{where} \quad \Delta = \cos(\phi_{\mathbf{x}_i}, \phi_{\mathbf{y}_i}) - \cos(\phi_{\mathbf{x}_i}, \phi_{\mathbf{y}_j}) \tag{3.33}$$

where $\cos(\phi_{\mathbf{x}_i}, \phi_{\mathbf{y}_j})$ is the cosine angle between the vectors $\phi_{\mathbf{x}_i}$ and $\phi_{\mathbf{y}_j}$ and γ is sensitivity parameter which is tuned based on the development data. Notice that, CCA and OPCA only attempt to maximize the similarity between an aligned pair $(\mathbf{x}_i, \mathbf{y}_i)$, where as S2Net not only maximizes the similarity between them but it also attempts to make it higher than that of an another pair $(\mathbf{x}_i, \mathbf{y}_j)$ where $j \neq i$.

The parameters A of the network are learnt using gradient descent methods. While neural network is a powerful tool to learn linear and non-linear mappings, learning the parameters of the network is tricky. Depending on the initialization of the parameters, the network can reach different local minimum. In [195], the authors first use OPCA to find the mapping and then use the solution found by OPCA to initialize the parameters of the network.

3.5 Probabilistic Models

Probabilistic generative models gained lot of popularity recently due to their ability to handle uncertainty. Moreover it is easier to extend these models by including additional assumptions/dependencies into the model. There have been several attempts to model multi-view data using probabilistic techniques [17, 132, 125, 20, 90, 198]. In this section, I first describe the probabilistic interpretation of CCA as introduced in [4] and then briefly present a few probabilistic

multi-view models for modeling multi-view data (mainly textual data sets).

3.5.1 Probabilistic Interpretation of CCA

As discussed in [4], there are at least two plausible generative models for CCA. In this section, I will briefly present the popular model and refer the reader to the original paper for further details. The generative story of CCA is given by:

$$\begin{aligned}
\mathbf{z} &\sim \mathcal{N}\big(\mathbf{0}, I_k\big) && 1 \leq k \leq \min\big\{d_1, d_2\big\} && (3.34)\\
\mathbf{x}|\mathbf{z} &\sim \mathcal{N}\big(W_x\mathbf{z} + \boldsymbol{\mu}_x, \Psi_1\big) && W_x \in \mathbb{R}^{d_1 \times k},\ \boldsymbol{\mu}_x \in \mathbb{R}^{d_1},\ \text{and } \Psi_x \in \mathbb{R}^{d_1 \times d_1} && (3.35)\\
\mathbf{y}|\mathbf{z} &\sim \mathcal{N}\big(W_y\mathbf{z} + \boldsymbol{\mu}_y, \Psi_2\big) && W_y \in \mathbb{R}^{d_2 \times k},\ \boldsymbol{\mu}_y \in \mathbb{R}^{d_2},\ \text{and } \Psi_y \in \mathbb{R}^{d_2 \times d_2} && (3.36)
\end{aligned}$$

where $\mathbf{0}$ is a zero vector of length k, $\mathcal{N}(,)$ is a multivariate normal distribution[2], and I_k is an identity matrix of size $k \times k$. The generative story involves, first generating a k dimensional embedding $\mathbf{z}$ and then generating the aligned pair $\mathbf{x}$ and $\mathbf{y}$ using $\mathbf{z}$. The generative story is convenient because it is easier to modify, for *e.g.*, as is done in non-parametric treatment of CCA [145], learning sparse projections [3], and so on.

The parameters of this model can be learnt using expectation maximization or in a closed form by maximizing the likelihood of the observed data. For completeness, here, I will present the maximum likelihood solution and encourage the reader to refer to [4] for its derivation. Let $C_{xx}^{-1/2} C_{xy} C_{yy}^{-1/2} \approx APB^T$ obtained using singular value decomposition, where A and B are matrices of sizes $d_1 \times k$ and $d_2 \times k$ respectively and P is a diagonal matrix of size $k \times k$. Notice that this step is same as learning projections in CCA. Let $U_x = C_{xx}^{-1/2} A$ and $U_y = C_{yy}^{-1/2} B$, then

$$\begin{aligned}
\widehat{W}_x &= C_{xx} U_x Q_x && (3.37)\\
\widehat{W}_y &= C_{yy} U_y Q_y && (3.38)\\
\widehat{\Psi}_x &= C_{xx} - \widehat{W}_x \widehat{W}_x^T && (3.39)\\
\widehat{\Psi}_y &= C_{yy} - \widehat{W}_y \widehat{W}_y^T && (3.40)\\
\hat{\boldsymbol{\mu}}_x &= \tilde{\boldsymbol{\mu}}_x && (3.41)\\
\hat{\boldsymbol{\mu}}_y &= \tilde{\boldsymbol{\mu}}_y && (3.42)
\end{aligned}$$

where $\widehat{(.)}$ represents the maximum likelihood estimate, $\widetilde{(.)}$ represents the empirical estimate from the data and $Q_x, Q_y \in \mathbb{R}^{k \times k}$ are arbitrary matrices such that $Q_x Q_y^T = P$. Given an unseen points $\mathbf{x}$ and $\mathbf{y}$, their posterior expectations (low dimensional projections) are obtained as:

$$\mathbb{E}(\mathbf{u}|\mathbf{x}) = Q_x^T U_x^T (\mathbf{x} - \boldsymbol{\mu}_x) \qquad \text{and} \qquad \mathbb{E}(\mathbf{u}|\mathbf{y}) = Q_y^T U_y^T (\mathbf{y} - \boldsymbol{\mu}_y) \tag{3.43}$$

3.5.2 Topic Models

Topic models provide an unsupervised way of organizing and exploring document collections [19, 18]. Most of the earlier work in topic models focussed on modeling monolingual document collection. But, recently there have been some attempts to model multilingual collections. These approaches can be broadly categorized into two classes depending on their assumptions about

[2] I slightly abuse the notation here. Earlier, I used the $\mathcal{N}(\mathbf{x}_i)$ to denote the neighbourhood of the point $\mathbf{x}_i$ in Multi-view NPP model and here I am using it to denote a multivariate normal distribution.

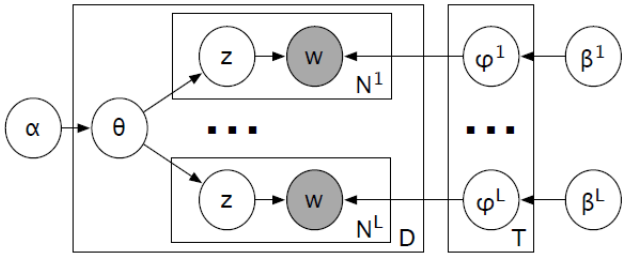


Figure 3.2: Graphical notation of the polylingual topic model (PTM). α and β^l are hyper-parameters.

the data. Models like correspondence LDA [17], multilingual LDA [132], polylingual topic model (PTM) [125], joint probabilistic semantic analysis (JPLSA), coupled probabilistic latent semantic analysis (CPLSA) [142], and bilingual topic model (BLTM) [56] assume that we have aligned pairs of document $(\mathbf{x}_i, \mathbf{y}_i)$ and try to learn topicality structure of the document collection. On the other hand, models like MuTo [20], JointLDA [90], and probabilistic cross-lingual latent semantic analysis [198] assume a bilingual word translation dictionary to model multilingual document collection. Since, most of the techniques that I discussed in this Chapter and in this dissertation use aligned pairs, I only discuss a canonical model (*i.e.,* PTM) from the former set of approaches.

Polylingual Topic Model : Polylingual topic model is proposed to model document collections where each document is represented as a raw term frequency vector. In this section, I will describe the model in the original setting. Given a n documents in L languages denoted as $\left\{\mathbf{w}_i^l, i = 1 \cdots n,\ l = 1 \cdots L\right\}$, PTM assumes the following generative story for the document collection:

1. For each language $l = 1 \ldots L$
   (a) For each topic $z = 1 \ldots T$, choose $\varphi_z^l \sim \text{Dir}(\beta^l)$
2. For document $i = 1 \ldots n$
   (a) Choose $\theta \sim \text{Dir}(\alpha)$
   (b) For each language $l = 1 \ldots L$
   - For each word $j = 1 \ldots N_d$
   - Select a topic $z_{ij}^l \sim \text{Mult}(\theta)$
   - Select a word $w_{ij}^l \sim \text{Mult}(\varphi_{z_{ij}^l})$

Its corresponding graphical notation is shown in Fig. 3.2. In this model, all the aligned documents share the same document-topic distribution θ. This multinomial distribution can be seen as the low-dimensional representation of the documents. Though this model shares the document-topic distribution θ across all the aligned documents, it does not enforce that the actual number of observed tokens of a topic in the aligned document pairs to be same. In practice,they tend to differ. This is enforced in later models (JPLSA [142]) by using posterior regularization. Thus, JPLSA learns better low-dimensional embeddings than PTM. In [142], the authors compare the generative multi-view models with the linear algebra style counter parts, such as CCA and OPCA, and show that the latter approaches outperform the generative models.

3.6 Discussion

In this Chapter, I briefly reviewed relevant multi-view models. I discussed linear projection based approaches, iterative techniques and probabilistic models. All most all of the projection based approaches can be solved for exact solution using some variant of the eigenvalue problem. Iterative and probabilistic approaches allow us to incorporate additional assumptions into the model, but it is always not possible to find global optimum. Moreover, previous research shows that the projection based approaches perform better than the generative counter pars. Further more, the optimization functions of different projection based approaches are strongly connected and vary slightly. So, in this dissertation I use CCA as base model and propose extensions that are appropriate for multilingual NLP problems. But I strongly believe that it is possible to extend other projection based approaches in a similar fashion.

Part II

Modeling Contributions

Chapter 4

Interlingual Representation

In this Chapter, I discuss the application of CCA to document alignment and bilingual dictionary mining tasks. Given a document in one language, document alignment task involves finding an aligned document from a different language. Where as bilingual dictionary mining task involves mining bilingual word translation pairs from a comparable corpus. Here, I first describe a straightforward application of CCA to mine target language translations for out-of-vocabulary (OOV) source language words and show their effectiveness in statistical machine translation. Next, I will proceed to the document alignment task. As discussed in Chapter 2, multi-view models use covariance matrices to find an interlingual representation. Though, in theory, these covariance matrices should be sparse, in practice they tend to be very dense due to the noise in the text documents. In order to tackle this, I propose sparsifying the covariance matrices and motivate it from the perspective of feature selection in the joint feature space (Sec. 2.4). Finally, I demonstrate the effectiveness of sparsification on the document alignment task.

For clarity, we refer to one of the languages as source (say German) and the other as target language (say English), but we do not differentiate between these languages and they are interchangeable. Moreover, we do not include any language specific phenomenon so these techniques should be applicable to other language pairs as well, with the caveat that the improvements may vary. For both these problems, we use a training data of aligned pairs to learn an interlingual representation and the projection directions. During prediction, given a source language object, we use the projection directions to project the source and target language objects into the interlingual representation and return the closest target language object as its aligned pair.

4.1 Bilingual Dictionary Mining

Large amounts of data are currently available to train statistical machine translation (MT) systems. Unfortunately, these training data are often qualitatively different from the domain in which the MT system is deployed. This difference in the domains of the training ("old domain") and test ("new domain") data causes domain divergence [95, 96] and, in this section, we consider one specific aspect of it namely the out-of-vocabulary problem. As we shall see, for most new domains, OOV terms are the source of approximately half of the additional errors made. In this Section, we will see the application of CCA to mine translations for these OOV words [68, 40]. The specific setting we consider is the one in which we have plentiful parallel ("labeled") data in an old domain (*e.g.*, parliament) and plentiful comparable ("unlabeled") data in a new domain (*e.g.*, medical). Finally, we assume access to a very small amount of parallel ("labeled") new domain data, but only enough to evaluate on, or run weight tuning [136]. All knowledge about OOV words must come from the new domain comparable corpus.

	Comparable		Tune	Test
	sents	words	sents	sents
News	35k	753k	1057	2007
Emea	307k	4220k	1388	4145
Subs	30k	237k	1545	2493
PHP	6k	81k	1007	2000

Table 4.1: Statistics of the new domain data in English.

To quickly summarize my approach, the old domain parallel corpus is used to learn a bilingual dictionary and is used to generate training data for CCA. CCA uses the training data to learn an interlingual representation as well as the projection directions. Subsequently, the OOV source words and all the target language words are mapped into the interlingual representation. After the projection, target language words that are closer to a source OOV word are returned as its translations. Finally, I report results on statistical machine translation in two language pairs and four different domains.

4.1.1 Importance of Handling Out-of-Vocabulary (OOV) Words

Before moving on to the details of mining translations, I will first describe the importance of handling OOV words, especially when the domains mismatch. We use European Parliament proceedings as the old domain (`http://www.statmt.org/europarl/`) and use four new domains: the News Commentary corpus (News) used in the MT Shared task at ACL 2007, European Medicines Agency text (Emea), the Open Subtitles data (Subs) and the PHP technical document data, provided as part of the OPUS corpus `http://urd.let.rug.nl/tiedeman/OPUS/`). I extract development and test sets from each of these corpora, except for *news* (and the old domain) where I preserve the published dev. and test data. The data is obtained for two language pairs English-German and English-French. The "old" domain of Europarl has 996k sentences and 2130k words. I count the number of words and sentences in the English side of the parallel data, which is the same for both language pairs (*i.e.,* both French-English and German-English have the same English). The statistics are shown in table 4.1. All of these data sets actually come with *parallel* new domain data. To obtain comparable data, I applied to standard trick of taking the first 50% of the English text as English and the last 50% of the German text as German. While such data is more parallel than, say, Wikipedia, it is far from parallel.

To get a better sense of the differences between these domains, I give some simple statistics about out of vocabulary words and examples in Table 4.2. Here, for each domain, I show the percentage of words (types) in the new domain that are unseen in the Parliament data. As we can see, it is markedly higher in Emea, Subs and PHP than in News.

4.1.2 Mining Translations for OOV Words

Here I describe the use of CCA to find the translations for the source (German) OOV words [68]. From the new domain corpus we extract the most frequent words (approximately 5000) for both the languages. As mentioned in the previous section, many of these frequent words are not observed in the old domain parallel corpus and hence the bilingual dictionary learnt from

Domain	Most frequent OOV Words
News (17%)	behavior, favor, neighbors, fueled, neighboring, abe, wwii, favored, nicolas, favorable, zhao, ahmedinejad, bernanke, favorite, phelps, ccp, skeptical, neighbor, skeptics, skepticism
Emea (49%)	renal, hepatic, subcutaneous, irbesartan, ribavirin, olanzapine, serum, patienten, dl, eine, sie, pharmacokinetics, ritonavir, hydrochlorothiazide, erythropoietin, efavirenz, hypoglycaemia, epoetin, blister, pharmacokinetic
Subs (68%)	gonna, yeah, f...ing, s..., f..., gotta, uh, wanna, mom, lf, ls, em, b....h, daddy, sia, goddamn, sammy, tyler, bye, bigweld
PHP (44%)	php, apache, sql, integer, socket, html, filename, postgresql, unix, mysql, color, constants, syntax, sesam, cookie, cgi, numeric, pdf, ldap, byte

Table 4.2: For each domain, the percentage of target domain word tokens that are unseen in the source domain, together with the most frequent English words in the target domains that do not appear in the source domain. (In the actual data the subtitles words do not appear censored.)

the old domain parallel corpus does not have translations. Of these frequent words, words that have translation in the bilingual dictionary (learnt from the old domain Europarl data) are used as training data. The dictionary mining involves multiple stages.

1. **Feature Extraction :** We extract feature vectors for all the words. We use context and orthographic features.

2. **Training Data Selection / Training :** We use word translation probabilities to identify pairs of words whose feature vectors are fed as training data to CCA. Then, we use CCA to learn an interlingual representation as well as the projection directions.

3. **Mining :** Finally, we project the source OOV and all the target words into the interlingual representation and mine translations for the OOV words.

Feature Extraction : The idea of feature extraction is to find a feature space which can capture the relationships between words. In this regard, we use different kinds of information to construct the feature vectors. Briefly, in each language, we extract context and orthographic features separately, build kernel matrices, and then reduce the size of the kernel matrices.

- For each of the frequent words, we extract the context vectors using a window of length five. To overcome data sparsity issue, we truncate each context word to its first seven characters; discard all the context features which co-occur with less than five words; and consider only the most frequent 2000 features in each language. We convert the frequency vectors into TFIDF vectors, center the data, and then binarize the vectors depending on if the feature value is positive of not. We convert this data into word similarities using linear dot product kernel.

German word	English translation	Score
blutdruckabfall	waste	0.274233
blutdruckabfall	bleeding	0.206440
blutdruckabfall	stroke	0.190345
dysphagie	dysphagia	0.233743
dysphagie	encephalopathy	0.215684
dysphagie	lethargy	0.203176
ribavirin	ribavirin	0.314273
ribavirin	viraferonpeg	0.206194
verfügbarkeit	bioavailability	0.409260
xeristar	xeristar	0.325458
xeristar	cymbalta	0.284616

Table 4.3: Random unseen Emea words in German and their mined translations.

- We also represent each word using the orthographic features, with n-gram character sequences of length 1-3 and convert them into TFIDF weights and subsequently turn them into word similarities (again using the linear kernel). Since we convert the data into word similarities, the orthographic features are relevant even though the script of source and target languages differ. Where as using the features directly is useless for languages whose script is completely different like Arabic and English.

For both the languages, we linearly combine the kernel matrices obtained using the context and the orthographic features. Here we use only these two types of information. But one can use other source of information such as word net or thesaurus to compute another kernel matrix and then include it. For efficiency, we use partial partial Gram Schmidt orthogonalization [72] to reduce the dimensionality of the kernel matrices. We do the same pre-processing for all words, the training and the OOV words. And the resulting feature vectors are used to learn the projection directions.

Training Data Selection / Training : Since a word can have multiple translations, and that CCA uses only pairs as training data, we select the word-pairs based on their translation probability. We form a bipartite graph with the training words in each language as nodes and the edge weight being the translation probability of the word-pair. We then run Hungarian algorithm to extract maximum weighted bipartite matching [98]. Subsequently, we run CCA on the resulting pairs of the bipartite matching to get the projection directions in each language. We retain only the top 35% of the eigenvectors. This setting of CCA was found to perform better in relevant experiments.

Mining : We project all the frequent words, including the training words, in both the languages into the lower dimensional space. And then, for each of the OOV word return the closest five points from the other language as potential new translations. The dictionary mining, viewed subjectively and intrinsically, performs quite well. Table 4.3 shows four randomly selected unseen German words from Emea (that do not occur in the Parliament data), together with the top three translations and associated scores (which are *not* normalized). A cursory evaluation of 5 randomly selected words in French and German by native speakers revealed that 8 out of 10 words have correct mined translations.

4.1.3 Integrating OOV Translations into MT System

The output of the dictionary mining approach is a list of pairs (f, e) of foreign words and predicted English translations. Each of them has an associated score. There are two obvious ways to integrate such a dictionary into a phrase-based translation system: (1) Provide the dictionary entries as (weighted) "sentence" pairs in the parallel corpus. These "sentences" would each contain exactly one word. The weighting can be derived from the translation probability from the dictionary mining. (2) Append the phrase table of a baseline phrase-based translation model trained only on source domain data with the word-pairs. Use the mining probability as the phrase translation probabilities.

In preliminary experiments (on German/Emea), it turned out that neither of these approaches worked particularly well. The first approach did not work at all, even with fairly extensive hand-tuning of the sentence weights. It often hurts translation performance. The second approach did not hurt translation performance, but did not help much either. It led to an average improvement of only about 0.5 Bleu points, on development data. This is likely because weight tuning tuned a single weight to account for the import of the phrase probabilities across both "true" phrases as well as these "mined" phrases.

So, I use a slightly more complex, but still simple, method for adding the dictionary entries to the phrase table. We add the following *four* new features to the model.

1. The dictionary mining translation probability. (Zero for original phrase pairs.)
2. An indicator feature that says whether *all* German words in this phrase pair were seen in the old domain data. (This will always be true for source phrases and always be false for dictionary entries.)
3. An indicator that says whether *all* German words in this phrase pair were seen in the new domain data. (This is *not* the negation of the previous feature, because there are plenty of words in the old domain data that had also been seen. This feature might mean something like "trust this phrase pair a lot.")
4. The conjunction of the previous two features.

Interestingly, only adding the first feature was not helpful (performance remained about 0.5 Bleu points above baseline). Adding only the last three features (the indicator features) alone did not help at all (performance was roughly on par with baseline). Only when all four features were included did performance improve significantly. In the following section, I report results on test data using all the four features.

4.1.4 Experiments

In all the experiments, I use two trigram language models. The first is trained on the Gigaword corpus. The second is trained on the English side of the target domain corpus. The two language models are traded-off against each other during weight tuning. In all cases MERT [136] is used for parameter tuning and results are averaged over three runs with different random initializations.

The first set of experiments is designed to establish baseline performance for the domains. In these experiments, we built a translation model based *only* on the Parliament proceedings. We

		BLEU				Meteor			
		News	**Emea**	**Subs**	**PHP**	**News**	**Emea**	**Subs**	**PHP**
German	BASELINE	23.00	26.62	10.26	38.67	34.58	27.69	15.96	24.66
	ORACLE-OOV	23.77	33.37	11.20	39.77	34.83	30.99	17.03	25.82
	ORACLE	24.62	42.77	11.45	41.01	35.46	36.40	17.80	25.85
French	BASELINE	27.30	40.46	16.91	28.12	37.31	35.62	20.61	20.47
	ORACLE-OOV	27.92	50.03	19.17	29.48	37.57	39.55	21.79	20.91
	ORACLE	28.55	59.49	19.81	30.15	38.12	45.55	23.52	21.77
ORACLE-OOV CHANGE		+2%	+24%	+11%	+5%	+0%	+12%	+6%	+7%
ORACLE CHANGE		+4%	+73%	+15%	+2%	+2%	+29%	+13%	+6%

Table 4.4: Baseline and oracle scores. The last two rows are the change between the baseline and the two types of oracles, averaged over the two languages.

then tune it using the small amount of target-domain tuning data and test on the corresponding test data. This is the row BASELINE in Table 4.4. Next, we build an oracle, based on using the *parallel* new domain data. This system, ORACLE in Table 4.4 is constructed by training a system on a mix of Parliament data and new domain data. The last line in this table (ORACLE CHANGE) shows the percent improvement when moving to this oracle system. As we can see, the gains range from tiny (4% relative Bleu points, or 1.2 absolute Bleu points for news, which may just be because we have more data) to quite significant (73% for medical texts).

Finally, we will see how much of this gain can be achieved using the mined bilingual translations. In order to estimate this, I take the ORACLE system, and remove any phrases that contain source-language words that appear in *neither* the Parliament proceedings *nor* our list of high frequency OOV terms. In other words, if our dictionary mining system found as-good translations for the words in its list as the (cheating) oracle system, this is how well it would do. This is referred to as ORACLE-OOV in Table 4.4. As we can see, the upper bound on performance based only on mining unseen words (ORACLE-OOV CHANGE) is about halfway (absolute) between the baseline and the full Oracle. Except in news, when it is essentially useless (because the vocabulary differences between news and Parliament proceedings are negligible). Results using Meteor are analogous.

The results of the dictionary mining experiment, in terms of its effect on translation performance, are shown in Table 4.5. As we can see, there is a modest improvement in Subtitles and PHP, a markedly large improvement in Emea, and a modest improvement in News. Given how tight the ORACLE results were to the BASELINE results in Subs and PHP, it is quite impressive that dictionary mining is able to improve performance significantly. In general, across all the data sets and both languages, we roughly split the difference (in absolute terms) between the BASELINE and ORACLE-OOV systems.

4.2 Covariance Selection

As discussed in Chapter 2, multi-view models capture essential feature pair co-occurrences in terms of a cross-covariance and two autocovariance matrices. Subsequently, they use these covariance matrices to find projection directions in each language (or view) such that aligned

	German		French	
	BLEU	**Meteor**	**BLEU**	**Meteor**
News	23.80	35.53	27.66	37.41
	+0.80	*+0.95*	*+0.36*	*+0.10*
Emea	28.06	29.18	46.17	37.38
	+1.44	*+1.49*	*+1.51*	*+1.76*
Subs	10.39	16.27	17.52	21.11
	+0.13	*+0.31*	*+0.61*	*+0.50*
PHP	38.95	25.53	28.80	20.82
	+0.28	*+0.88*	*+0.68*	*+0.35*

Table 4.5: Dictionary-mining system results. The italicized number beneath each score is the improvement over the BASELINE approach from Table 4.4.

objects lie close to each other (Fig. 2.3). The strong reliance of these approaches on the covariance matrices leads to problems, especially in the presence of noisy data caused either by the noisy features or the noisy alignments between the objects. Noisy data is not uncommon and is usually the case with data collected from community based resources such as Wikipedia. This degrades performance of a variety of tasks, such as transliteration Mining [103, 76, 147] and multilingual web search [59]. In this section, I describe the idea of *sparsifying covariance matrices* to remove noisy co-occurrences and hence increase the robustness.

For the ease of exposition, I explain the technique in the context of document alignment task between English and German. As discussed in the beginning of this Chapter, in this task, we are given a bilingual comparable corpus and are told that some German documents have an aligned English document. The aim is to recover the hidden alignments. In this particular task, the covariance matrices capture the word co-occurrences. The cross-covariance matrix captures the word co-occurrence strengths of German and English words while the two autocovariance matrices capture the co-occurrence strengths of German words and English words respectively.

To understand the challenge posed by the noisy word-pairs, in the case of document alignment problem, let X ($d_1 \times n$) and Y ($d_2 \times n$) represent n aligned documents in both the languages and further assume that the data is centered. Then CCA finds projection directions $\mathbf{a} \in \mathbb{R}^{d_1}$ and $\mathbf{b} \in \mathbb{R}^{d_2}$ which maximize $\mathbf{a}^T XY^T \mathbf{b}$ (Sec. 2.2). This can be rewritten as :

$$\begin{aligned}
\mathbf{a}^T XY^T \mathbf{b} &= \sum_{k=1}^{n} \langle \mathbf{x}_k, \mathbf{a} \rangle \langle \mathbf{y}_k, \mathbf{b} \rangle \\
&= \sum_{k=1}^{n} \Big(\sum_{i=1}^{d_1} X(i,k) a_i \cdot \sum_{j=1}^{d_2} Y(j,k) b_j \Big) \\
&= \sum_{i=1}^{d_1} \sum_{j=1}^{d_2} a_i b_j \Big(\sum_{k=1}^{n} X(i,k) Y(j,k) \Big) \\
&= \sum_{i,j=1}^{d_1,d_2} a_i b_j C_{xy}(i,j) \qquad (4.1)
\end{aligned}$$

where C_{xy} is the cross covariance matrix. Similarly, the length constraints can also be rewritten

as $\sum_{i,j=1}^{d_1} a_i a_j C_{xx}(i,j) = 1$ and $\sum_{i,j=1}^{d_2} b_i b_j C_{yy}(i,j) = 1$. I remind the reader that C_{xx} and C_{yy} are monolingual autocovariance matrices.

Maximizing this objective function, under the constraints, involves a careful selection of the vectors **a** and **b** such that $a_i b_j$ is high whenever $C_{xy}(i,j)$ is high. So, every non-zero entry of the cross-covariance matrix restricts the choice of the projection directions. While this may not be a severe problem when the training data is clean, but this is very uncommon especially in the case of high dimensional data like text documents. Moreover, the inherent ambiguity of natural language increases the chances of seeing a noisy word in any document. Every occurrence of a noisy word will have a non-zero contribution towards the covariance matrix making it dense, which in turn prevents the selection of appropriate projection directions.

In this section, I describe a technique to recover the sparsity of the covariance matrices by removing the noisy entries from the covariance matrices. As far as I know, this is the first work that attempts to recover the sparseness of the covariance matrices to improve the projection directions. My work is different from the sparse CCA [71, 145] proposed in the machine learning literature. Their objective is to find projection directions such that the original documents are represented as a sparse vectors in the common subspace. Another seemingly relevant but different direction is the sparse covariance matrix selection research [12]. Their objective is to find matrices such that the *inverse* of the covariance matrix is sparse which has applications in Gaussian processes.

I break this task of sparsifying covariance matrices into two sub problems: computing an association score for every word-pair and then using an appropriate strategy to identify the noisy pairs based on their weights. I explore some simple ways to address both the steps in the following two sections. Here I describe only a couple of them to convey the main idea of sparsifying covariance matrices, and encourage the reader to refer to Appendix A for further details. For the sake of convenience and clarity, I describe the technique in the context of cross-covariance matrix between English and German language pair. But these techniques extend directly to monolingual covariance matrices, and to different language pairs as well.

4.2.1 Computing Word Pair Association

The first step in filtering out the noisy word co-occurrences is to use an appropriate measure to compute the strength of a word-pair (English and German words). This is a well studied problem and several association measures have been proposed in the NLP literature [49, 86, 130]. These association measures can be grouped based on the statistics they use [77]. Association measures like covariance and point-wise mutual information, which only use the frequency with which a word pair co-occurs, often overestimate the strength of low frequent words [130]. On the other hand, measures like log-likelihood ratio [49] and mutual information (MI) use other statistics like the marginal probabilities of each of the words. Here I describe three association measures that I use to compute the strength of word-pairs.

1. **Mutual Information :** For any two words, e_i and f_j, let m_{11}, m_{10}, m_{01} and m_{00} denote the number of documents in which both the words co-occur, only English word occurs, only German word occurs and none of the words occur. Then the mutual information (MI) of this word-pair is given by:

$$\text{MI}(e_i, f_j) = \frac{1}{m} \sum_{p,q \in \{0,1\}} m_{pq} \log \frac{m_{pq} \times m}{m_{p(\cdot)} m_{(\cdot)q}} \qquad m_{p(\cdot)} = \sum_{q \in \{0,1\}} m_{pq} \text{ and } m_{(\cdot)q} = \sum_{p \in \{0,1\}} m_{pq}$$

where m is the total number of documents. I treat the occurrence of a word in a document slightly different from others, I treat a word as occurring in a document if it has occurred more than its average frequency in the corpus. Log-likelihood ratio and the MI differ only by a constant, so I use only MI in the experiments.

2. **Yule's ω :** Yule's ω is another popular association measure used in psychology [150]. It uses same statistics used by mutual information but differs in the way they are combined. MI converts the frequencies into probabilities before computing the association measure where as Yule's ω uses the observed frequencies directly, and doesn't make any assumptions about the underlying probability distributions. Given the same interpretation of the variables as introduced above, the Yule's ω is estimated as:

$$\omega = \frac{\sqrt{m_{00}m_{11}} - \sqrt{m_{01}m_{10}}}{\sqrt{m_{00}m_{11}} + \sqrt{m_{01}m_{10}}} \tag{4.2}$$

   This way of combining the frequencies bears similarity with the log-odds ratio.

3. **Bilingual Dictionary :** The above two association measures use the same training data that is available to compute the covariance matrices in CCA. Thus, their utility in bringing additional information, which is not captured by the covariance matrices, is arguable (experiments show that they are indeed helpful). Moreover, they use document level co-occurrence information which is coarse compared to the co-occurrence at sentence level or translational information provided by a bilingual dictionary. So, I use bilingual dictionaries as the final resource to weigh word co-occurrences. I use translation tables learnt using Giza++ [137] on Europarl data set. I use a threshold on the conditional probability to filter out the low probability translation pairs. Since the translation tables are asymmetric, I ensure that the conditional probability in both the directions is above the chosen threshold.

While the first two association measures can also be applied to monolingual data, bilingual dictionary can not be used for weighting monolingual word-pairs. So for sparsifying monolingual covariance matrices, I use either of the first two schemes for weighting monolingual word-pairs.

4.2.2 Selection Strategy – Bipartite Matching

The next step after computing association measure for all word-pairs is to use them in selecting word-pairs that need to be retained. A straightforward way to remove the noisy word co-occurrences is to zero out the entries of the cross-covariance matrix that are lower than a threshold. But this is usually biased to frequent words. Since a frequent word co-occurs with other words often, it naturally tends to have high association with most of the other words. As a result, thresholding tends to remove all the less frequent word-pairs while leaving the co-occurrences of the frequent words untouched. Eventually, this may lead to zeroing out some of the rows or the columns of the cross-covariance matrix [93]. To overcome this, I pose this problem as a bipartite graph matching problem.

Matching : I use matching for sparseness selection. I formulate the selection of the word-pairs as a network flow problem [92]. The objective is to select word-pairs that have high association measure while constraining each word to be associated with only a few words from the other language. Let I denote a binary matrix of same size as that of the cross-covariance matrix with

$I(i,j)$ taking a value of 1 or 0 depending on if the word-pair (e_i, f_j) is selected or not. We want each word to be associated with k words from other language, *i.e.,* $\sum_j I(i,j) = k$ and $\sum_i I(i,j) = k$. Moreover, we want word-pairs with high association score to be selected. We can encode this objective and the constraints as the following optimization problem:

$$\arg\max_{I} \sum_{i,j=1}^{d_1,d_2} C_{xy}(i,j) I(i,j) \tag{4.3}$$
$$\forall i \sum_j I(i,j) = k; \quad \forall j \sum_i I(i,j) = k; \quad \text{and} \quad \forall i,j \; I(i,j) \in \{0,1\}$$

If $k = 1$, then this problem reduces to a linear assignment problem and can be solved optimally using the Hungarian algorithm [98]. For other values of k, this can be solved by relaxing the constraint $I(i,j) \in \{0,1\}$ to $0 \leq I(i,j) \leq 1$. The optimal solution of this relaxed problem can be found efficiently using linear programming [148]. The uni-modular nature of the constraints guarantees an integral solution [156], so relaxing the original integer problem doesn't introduce any error in the optimal solution.

4.2.3 Monolingual Augmentation

The above selection strategy operates on the covariance matrices independently. In this section I update the sparseness of the cross-covariance matrix based on the sparseness structure of the autocovariance matrices. Specifically, I propose to augment the set of selected bilingual word-pairs using the monolingual word-pairs. We first use any of the above mentioned strategies to select bilingual and monolingual word-pairs. Let I_{xy}, I_{xx} and I_{yy} be the binary matrices that indicate the selected word-pairs based on the bilingual and monolingual association scores. Then the monolingual augmentation strategy updates I_{xy} in the following way:

$$I_{xy} \leftarrow \text{Binarize}(I_{xx} I_{xy} I_{yy})$$

i.e., we multiply I_{xy} with the monolingual selection matrices and then binarize the resulting matrix. Monolingual augmentation is motivated by the following probabilistic interpretation:

$$P(x,y) = \sum_{x',y'} P(x|x') P(y|y') P(x',y')$$

which can be rewritten as $P \leftarrow T_x P T_y^T$ where T_x and T_y are monolingual state transition probabilities.

4.2.4 Feature Selection for CCA

In this section I summarize my approach to sparsification for multi-view models. The training phase involves finding projection directions for documents of both the languages. For that, we compute the covariance matrices using the training data. Then use any of the word association measures (Sec. 4.2.1) along with the bipartite matching algorithm (Sec. 4.2.2) to recover the sparseness in either only the cross-covariance or all of the covariance matrices. Let I_{xy}, I_{xx} and I_{yy} be the binary indicator matrices which represent the word-pairs that are selected based on the chosen

sparsification technique. Now, we replace the covariance matrices as follows:

$$C_{xx} \leftarrow C_{xx} \otimes I_{xx} \; ; \qquad C_{yy} \leftarrow C_{yy} \otimes I_{yy} \; ; \qquad C_{xy} \leftarrow C_{xy} \otimes I_{xy} \tag{4.4}$$

where $\otimes$ denotes the element-wise matrix product. Subsequently, we find the projection directions using method described in Sec. 2.2.2.1. Let A and B be the matrices formed with top eigenvectors as the columns. These projection matrices are used to map documents into the interlingual representation. During prediction, we modify the Eq. 2.14 so that it accounts for the sparsity structure. That is, given an English document $\mathbf{x}$, finding an aligned German document $\mathbf{y}$ involves solving:

$$\arg\max_{\mathbf{y}} \frac{\mathbf{x}^T \Big((AB^T) \otimes I_{xy} \Big) \mathbf{y}}{\sqrt{\mathbf{x}^T \Big((AA^T) \otimes I_{xx} \Big) \mathbf{x}} \sqrt{\mathbf{y}^T \Big((BB^T) \otimes I_{yy} \Big) \mathbf{y}}}$$

4.2.5 Experiments on Synthetic Data

I evaluate the effectiveness of the feature selection technique on the synthetic data first and then proceed to real world data set in Sec. 4.3. I follow the generative story introduced in Bach and Jordan [4] to generate synthetic multi-view data (Sec. 3.5.1). Their method does not assume any correspondence between the feature dimensions of both the views. I modify their approach slightly so that we know the actual correspondence between the features and use these true feature correspondences for sparsification of the cross-covariance matrix.

I first generate a d dimensional vector in the common latent space and then map it into the individual feature spaces. This is inverse of the projection step that we usually do. That is, usually we map the objects from the original feature spaces into the low-dimensional subspace but here, for generating synthetic data, we generate the low-dimensional point first and then get its representations in the high dimensional feature spaces. This is done as follows:

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, I_d) \tag{4.5}$$

$$\mathbf{x}|\mathbf{z} \sim (W_1 \mathbf{z} + \mu_1) + \eta \, \mathcal{N}(\mathbf{0}, I_{d_1}) \tag{4.6}$$

$$\mathbf{y}|\mathbf{z} \sim (W_1 \mathbf{z} + \mu_2) + \eta \, \mathcal{N}(\mathbf{0}, I_{d_2}) \tag{4.7}$$

Notice that I use the same projection matrix W_1 for both the views, this ensures a one-to-one correspondence between the features of both the spaces. Moreover, I also introduce a parameter η which controls the amount of noise in the data.

I generate a total of 3000 pairs of points and use 2000 of them for training the models and the rest for evaluation. I use the true feature correspondences to form the cross-covariance selection matrix I_{xy} (Sec. 4.2.4). For this experiment, I use the full monolingual covariance matrices. I train both CCA and its sparse version on the training data and evaluate them on the test data. I repeat this multiple times and report the average accuracies. I report the accuracy of the top ranked object and the mean reciprocal rank (MRR) of the correct output. Fig. 4.1 shows the performance of CCA and the sparse CCA, as we vary the noise parameter η. As the noise increases, the performance of CCA drops quickly and the sparse version starts performing significantly better. This oracle experiment demonstrates a significant performance gain when the true correspondences are available. But this information is not available in the case of real world data sets, so we approximate it.

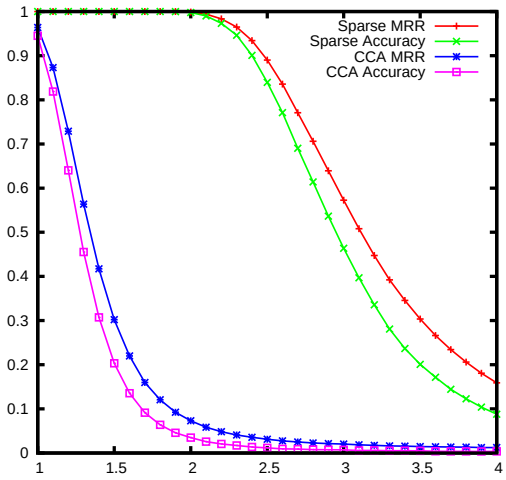


Figure 4.1: Accuracy of CCA and our sparsified version with the noise parameter.

4.3 Document Alignment

In the previous section, I introduced a feature selection technique for CCA and showed its effectiveness on a synthetic data set where the true sparsity is known. But the true sparsity is rarely known in real life applications, hence we approximate it using other sources of information. In this section, we will see the effectiveness of the feature selection for document alignment task [93]. I remind the reader that in this task, we are given a comparable corpora and are told that some source documents have a comparable document in the target language and vice versa. The goal is to recover this hidden alignment and the recovered alignment is compared against the ground truth. This task arises in the following applications:

- **Comparable (parallel) document retrieval :** This involves identifying document pairs from a multilingual corpus that are related (translations) of each other [54, 186]. Cross-lingual information retrieval (CLIR) [135, 10] is a different flavor of this task where a user query is treated as a document in one of the languages.

- **Parallel phrase extraction :** Parallel phrases extracted from a comparable corpora acts as an important source of training data for statistical machine translation (SMT) [131, 146]. Document alignment is used as a preprocessing step in the parallel phrase extraction to limit the number of candidate phrase alignments and is shown to help in both efficiency and accuracy.

- **Transliteration Mining :** Like in SMT, transliteration mining techniques [76, 147] use document alignment as a preprocessing step to avoid the exponential number of possible candidates [183]. Moreover, if we treat each name as a bag-of-ngrams then transliteration mining and document alignment becomes similar problems [182].

- **Cross-language text categorization :** This task involves classifying cross-lingual documents into a set of predefined classes [14]. In this task, usually, training data of documents and their reference class labels is available in only one of the languages and the challenge involves

classifying documents of a different language. Mapping documents of both the languages into an interlingual representation and learning a classifier in this common subspace is a good alternative for this task [142].

- **Multilingual Web Search :** This task [87] uses training data from a resource rich language such as English to improve search accuracy in a relatively resource poor language (both query and results are in the resource poor language). Typical approaches to this problem [58, 59] involve computing cross-lingual document similarity between the candidate result sets from different languages.

A central problem in all of the above tasks is to compute the similarity between a pair of documents from different languages. In the following section, we compare the effectiveness of CCA and the sparse version with other approaches on the document alignment task. Most of the existing approaches to document alignment task use manually aligned document pairs to find a common subspace. The subspace can be found using either generative approaches based on topic modeling [125, 90, 199, 186] or discriminative approaches based on variants of principal component analysis (PCA) and CCA [48, 185, 142, 68]. Both styles rely on document level term co-occurrences to find the latent representation.

4.3.1 Experiments

I evaluate our idea of sparsifying the covariance matrices incrementally. I discussed multiple ways for sparsifying the covariance matrices, with each method having parameters to control the amount of sparseness. I use a small amount of development data for model selection and parameter tuning and choose a few promising models. Finally, I compare these selected models with state-of-the-art baselines on two language pairs and on two different data sets. In each case, I use training data to learn the projection directions. And then, for each of the test documents, I find the aligned document from other language. I report average accuracy of the top ranked document and also the mean reciprocal rank (MRR) of the true aligned document.

4.3.1.1 Model Selection

For model selection, I use approximately 5000 document pairs collected from the Wikipedia between English and Spanish. I use the cross-language links provided by Wikipedia as the ground truth. I tokenize the documents, retain only the most frequent 2000 words in each language and convert the documents into TFIDF vectors. I use 60% of the data for training different models and the rest for model selection. I choose a few promising models based on this development set results and evaluate them on bigger data sets.

Fig. 4.2 shows the performance of sparsifying with different association measures with varying levels of sparsity in the covariance matrices. The horizontal line represents the performance of CCA on this data set. We start with 2000 non-zero entries in the covariance matrices and experiment up to 20,000 non-zero entries. Since our data set has 2000 words in each language, 2000 non-zero entries in a covariance matrix implies that, on an average, every word is associated with only one word. This results in highly sparse covariance matrices. Overall, we can see that reducing the level of sparsity , *i.e.,* selecting more number of elements in the covariance matrices, increases the performance slightly and then decreases again. From the figure, it seems that sparsifying the covariance matrices might help in improving the performance of the task. But it is

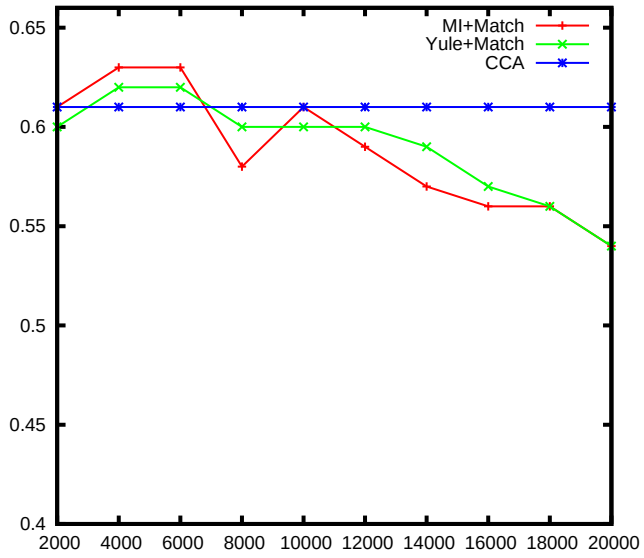
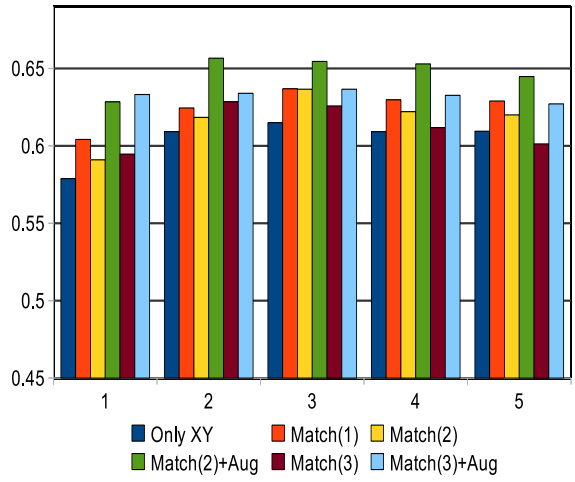


Figure 4.2: Comparison of feature selection using the word association measures with CCA. The x-axis plots the number of non-zero entries in the covariance matrices and the y-axis plots the accuracy of top-ranked document.

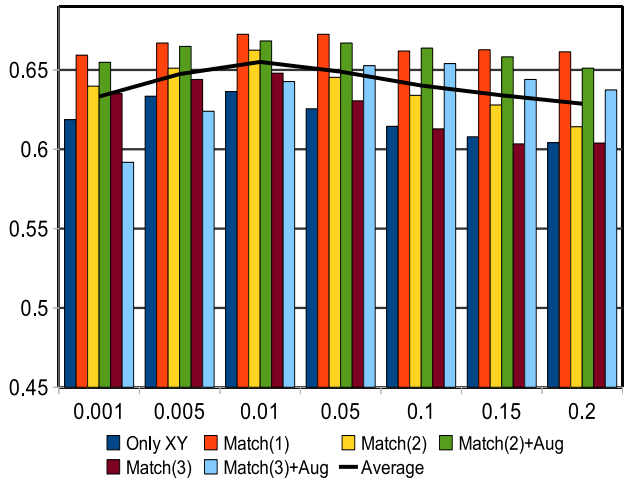
interesting to note that not all the models perform better than CCA. Though I don't report results here, I observe that MI and Yule's ω perform competitively but they consistently outperform models that use other selection strategies [93]. So in the rest of the experiments I report results with MI or Yule's ω.

In the previous experiment, I used same level of sparsity for all the covariance matrices, *i.e.*, same number of associations were selected for each word in all the three covariance matrices. In the following experiment, I use different levels of sparsity for the individual covariance matrices. Fig. 4.3 shows the performance of Yule+Match and Dict+Match combinations with different levels of sparsity. In the Yule+Match combination, I use Yule's ω association measure for weighting the word-pairs and use matching for selection. In the Dict+Match combination, I use bilingual dictionary for sparsifying cross-covariance matrix, *i.e.*, I keep all the word-pairs whose conditional translation probability is above a threshold. And for monolingual word-pairs, I use MI for weighting and matching for word pair selection. For each level of sparsity of the cross-covariance matrix, I experiment with different levels of sparsity on the monolingual covariance matrices. 'Only XY' indicates that I use the full monolingual covariance matrices. In 'Match(k)' runs, I allow each word to be associated with a total of k words (Eq. 4.3). 'Aug' indicates that I use monolingual augmentation to refine the sparsity of the cross-covariance matrix (Sec. 4.2.3).

From both the figures 4.3(a) and 4.3(b), we see that 'Only XY' run (dark blue) performs poorly compared to the other runs, indicating that sparsifying all the covariance matrices is better than sparsifying only the cross-covariance matrix. In the Yule+Match combination, Fig. 4.3(a), all the runs seem to be performing better when each English word is allowed to associate with 2 or 3 Spanish words and vice versa. Among different ways of selecting the monolingual word-pairs, Match(2)+Aug performs better than the remaining runs. So I use Match(2)+Aug combination for the Yule's ω measure.



(a) Performance of Yule+Match combination. The x-axis plots the number of Spanish words selected per each English word and vice versa. This determines the sparsity of C_{xy}. Matching is used as selection criteria for all the covariance matrices.



(b) Performance of Dict+Match combination. The x-axis plots the threshold on bilingual translation probability and it determines the sparsity of C_{xy}. Matching is used to select *only* the monolingual sparsity.

Figure 4.3: Comparison of Yule+Match and Dict+Match combination with different levels of sparsity for the covariance matrices. In both the figures, the x-axis plots the sparsity of the cross-covariance matrix and for each value we try different levels of sparsity on the monolingual covariance matrices (which are grouped together). The description of these individual runs is provided in the relevant parts of the text. The y-axis plots the accuracy of the top-ranked document. CCA achieves 61% accuracy on this data set.

Unlike the Yule+Match combinations, there is no clear winner for Dict+Match combinations. First of all, the performance increase as we increase the translation probability threshold and then decreases again (indicated by the 'Average' performance in Fig. 4.3(b)). On an average, all the systems perform better with a threshold of 0.01, which I use in the final experiments. In this case, both Match(1) and Match(2)+Aug runs (orange and green bars respectively) perform competitively so I use both of these models in our final experiments.

Based on the above experiments, I choose the following combinations for the final experiments. Yule(l)+Match(k), where $l \in \{2, 3\}$ is the number of Spanish words allowed for each English word and vice versa and k=2 is the number of monolingual word associations for each word. I also run both these combinations with monolingual augmentation, indicated by Yule(l)+Match(k)+Aug. For dictionary based weighting, Dict+Match(k), I choose a translation probability threshold of 0.01 and try $k \in \{1, 2\}$. Again, I run these combinations with monolingual augmentation identified by Dict+Match(k)+Aug.

4.3.1.2 Results

For the final experiments, I choose data in two language pairs (English-Spanish and English-German) from two different resources, Europarl [105] and Wikipedia. For Europarl data sets, I artificially make them comparable by considering the first half of English document and the second half of its aligned foreign language document [125]. For Wikipedia data set, I use the

	Wikipedia				Europarl			
	English-Spanish		English-German		English-Spanish		English-German	
	Acc.	**MRR**	**Acc.**	**MRR**	**Acc.**	**MRR**	**Acc.**	**MRR**
CCA	0.776	0.852	0.570	0.699	0.872	0.920	0.748	0.831
OPCA	0.781	0.856	0.570	0.700	0.870	0.920	0.748	0.831
Yule(2)+Match(2)	0.798*	0.866*	0.576	0.703	0.901*	0.939*	0.780*	0.853*
Yule(2)+Match(2)+Aug	0.811*	0.876*	0.602*	0.723*	0.883	0.927	0.771*	0.847*
Yule(3)+Match(2)	0.803*	0.870*	0.572	0.700	0.856	0.907	0.747	0.830
Yule(3)+Match(2)+Aug	0.793*	0.861*	0.610*	0.726*	0.878$^+$	0.925$^+$	0.763$^+$	0.843*
Dict+Match(1)	0.811*	0.875*	**0.656***	**0.762***	**0.928***	**0.957***	**0.874***	**0.922***
Dict+Match(2)	0.811*	0.876*	0.623*	0.736*	0.923*	0.955*	0.853*	0.907*
Dict+Match(2)+Aug	**0.825***	**0.885***	0.630*	0.735*	0.897*	0.935*	0.866*	0.917*

Table 4.6: Performance of our models in comparison with CCA and OPCA on English-Spanish and English-German language pairs. * and $^+$ indicate statistical significance measured by paired t-test at p=0.01 and 0.05 levels respectively. When an improvement is significant at p=0.01 it is automatically significant at p=0.05 and hence is not shown.

cross-language link as the ground truth. For each of these data sets, I choose approximately 5000 aligned document pairs. I remove the stop words and keep all the words that occur in at least five documents. After the preprocessing, on an average, there are 4700 words in each language. Subsequently I convert the documents into their TFIDF representation.

In Platt *et al.* [142], the authors compare different systems on the comparable document retrieval task and show that discriminative approaches work better compared to their generative counter parts. So, here I compare only with the state-of-the-art discriminative systems such as CCA and OPCA [142]. For each of the systems, I report the average results of five-fold cross validation. I divide the data into 3:1:1 ratio for training, validation and test sets. The validation data set is used to select the size of of the interlingual representation. For both CCA and my models, I set the regularization parameter λ to 0.3. For OPCA, I manually tried different regularization parameters ranging from 0.0001 to 1 and found that a value of 0.001 worked best.

The results are shown in Table 4.6. On these data sets, both CCA and OPCA performed competitively. As described in Sec. 3.1.2, OPCA differs from CCA when features are shared across the views. But in the data sets that I used here, vocabulary of both the languages is treated differently, so it is not surprising that they give almost the same results. From the results, it is clear that sparsifying the covariance matrices helps improving the accuracies significantly. In all the four data sets, the best performing method always used dictionary for cross-lingual sparsity selection. This indicates that using fine granular information such as a bilingual dictionary gleaned from an external source is very helpful in improving the accuracies. Among the models that rely solely on the training data, models that use monolingual augmentation performed better on Wikipedia data set, while models that do not use augmentation performed better on Europarl data sets. This suggests that, when the aligned documents are clean (closer to being parallel) the statistics computed from cross-lingual corpora are trustworthy. As the documents become comparable, we need to use monolingual statistics to refine the bilingual statistics. Moreover, these models achieve higher gains in the case of Wikipedia data set compared to the gains in Europarl. This conforms with our

initial hunch that, when the training data is noisy the covariance matrices tend to be dense and hence sparsification is relevant.

4.4 Discussion

In this Chapter, first, I showed that CCA can be applied to mine translations for unseen words for machine translation. We have seen positive, consistent results across two languages and four domains. The proposed approaches for both bilingual dictionary mining and integration into MT are generic enough to be integrated into a wide variety of translation systems other than simple phrase-based translation. Of course, unseen words are not the only cause of translation divergence between two domains. We have not addressed other issues, such as better estimation of translation probabilities or words that change word sense across domains. This problem requires significant additional work, since it is quite a bit more difficult to spot foreign language words that are used in new senses, rather that just never seen before. I will address the problem of mining sense specific translations in Chp. 7.

In this Chapter, I also proposed a novel idea of sparsifyng covariance matrices to learn better interlingual representations. I discussed sparsification in the context of CCA only, but the technique is general and can be applied to other multi-view models, like OPLS and OPCA, that capture feature co-occurrences in terms of covariance matrices. Experimental results show that using external information which is gleaned from cleaner resources brings significant improvements. Moreover, we also observe that computing feature pair associations from the same training data along with an appropriate selection criteria can *also* yield significant improvements. This is certainly encouraging and a promising direction is to explore more sophisticated techniques to recover the sparsity based on the training data itself.

Chapter 5

Regularized Interlingual Representation

Because of the wide usage of English, many NLP tasks have bilingual resources from English into other languages. For *e.g.,* significantly larger parallel texts are available between English and other languages. Similarly, bilingual dictionaries and transliteration data sets are more accessible from a language into English than into a different language. This situation has caused the NLP community to develop approaches which use a resource rich language Q (say English) as pivot (or bridge) to build resources/applications between a new language pair P and R. Previous studies in transliteration and dictionary mining show that these bridge language approaches perform competitively with approaches that use resources between P and R. In this Chapter, I propose a regularization framework for interlingual representations. The key aspect of my approach is that it accounts for language specific variation in the bridge language resources (*i.e.,* variations specific to $P \leftrightarrow Q$ and $Q \leftrightarrow R$) and aims to minimize this variation as much as possible.

5.1 Problem Setting

Recall the notation that I use superscript and subscript to indicate language and index for vector variables where as I simply use subscript to indicate the language for scalars and matrices. That is, $\mathbf{x}_i^{(l)}$ represents i^{th} vector in l^{th} language while n_l and X_l represent a scalar and a matrix of the l^{th} language respectively.

We assume that we have training data from L languages into a bridge language. Let d_l be the size of the feature space in l^{th} language and c be the size of the bridge language's feature space. Let $\{(\mathbf{x}_i^{(l)}, \mathbf{p}_i^{(l)})\}$ denote the $i^{th} = 1 \cdots n_l$ aligned object pair from l^{th} language into the bridge language, where $\mathbf{x}_i^{(l)} \in \mathbb{R}^{d_l}$ and $\mathbf{p}_i^{(l)} \in \mathbb{R}^c$. Though this training data is from l^{th} language into the bridge language, for brevity, I refer to this as the l^{th} *language training data*. Hopefully, this terminology becomes clear when I describe an example task in Section 5.1.1. Finally, let X_l $(d_l \times n_l)$ and P_l $(c \times n_l)$ denote training data matrices of the l^{th} language with $\mathbf{x}_i^{(l)}$ and $\mathbf{p}_i^{(l)}$ $i = 1 \cdots n_l$ as columns respectively.

Notice that the number of training examples (n_l) in each language's training data is different. Moreover, we do not assume any explicit relation between the training examples across languages. These two assumptions differentiate my formulation from other generalizations of multi-view models, Sec. 2.2.5, which assume that a training sample is available in all the languages (views). As will be discussed with an example in Section 5.1.1, it is a restrictive assumption and limits their applicability. In contrast, my formulation relaxes this assumption and hence is more amenable to the bridge language applications.

English		Bulgarian	
Word	**IPA**	**Word**	**IPA**
bashful	/ˈbæʃfəl/	шибам (switch)	/ˈʃibəm/
tuesday	/ˈtuːzdeɪ/	лук (onion)	/luk/
craft	/kɹæft/	как (how)	/kak/
book	/bʊk/	музей (museum)	/mʊˈzej/
head	/hɛd/	спека (spekle)	/spɛˈkɤ/

Table 5.1: Example phoneme dictionaries in English and Bulgarian. The English translation for the Bulgarian words are also provided. Notice that, unlike the usual transliteration approaches, the training data for my approach is words and not names (Sec. 5.4).

5.1.1 Example – Transliteration using Phonemic Alphabet

To understand the problem setting, consider the following name transliteration task. Given L languages, the aim is to build a transliteration system between every pair of languages. This task is relevant in scenarios like Bing translator[1] and Google translate[2], where a user can choose to translate between any language pair. A straightforward supervised learning approach for this task requires training data of name pairs between every pair of languages [104] which is very expensive. A more practical setting is to obtain training data from every language into a common bridge language. In this Chapter, I use international phonetic alphabet (IPA) in a manner akin to a "bridge language". I assume that we have a list of words and their IPA representations in each of the L languages. The words in different languages need not have any relationship to each other. Table 5.1 shows few words and their IPA representations in English and Bulgarian languages. I refer to the set of words and their IPA representations as *phoneme dictionary* in this Chapter. Notice that the common symbols in the IPA sequences indicate vague mapping between the character sequences of English and Bulgarian. For *e.g.,* both the words 'bashful' and 'шибам' have the symbol 'ʃ' in their IPA sequences which indicate the mapping between the character sequences between 'sh' and 'ш'. The use of IPA as a bridge language offers several advantages which I discuss later (Sec. 5.4 or refer to Jagarlamudi and Daumé III [89]). In this particular problem, IPA is not really a language and thus the l^{th} *language training data* is literally the training data in that language.

We convert the phoneme dictionary of each language into feature vectors. That is, we convert each word into a feature vector of n-gram character sequences and similarly we also convert the IPA representations into feature vectors of n-gram IPA symbol sequences. For *e.g.,* if we use unigram and bigram sequences, then the feature vectors of 'head' and its IPA sequence 'hɛd' are given by {h, e, a, d, #h, he, ea, ad, d#} and $\{h, \varepsilon, d, \#h, h\varepsilon, \varepsilon d, d\#\}$ respectively. For brevity, I refer to the spaces of n-gram character and IPA symbol sequences as *character* and *phonemic* spaces respectively. The character space is different for each language while the phonemic space is common to all the languages. Since we use IPA as bridge, even though two languages share a common orthography (*e.g.,* English and French) it is irrelevant in my formulation.

In this Chapter, we will see some techniques to learn an interlingual representation under this problem setting. For clarity, I will describe the techniques in the context of name transliteration

[1] http://www.bing.com/translator/
[2] http://translate.google.com/

Word	IPA sequence by language
China	/ˈʧaɪ.nə/ (En), /ˈʃina/ (Du), /ˈçiːnaː/ (De)
America	/əˈmɛrɪkə/ (En), /aˈme.ri.ka/ (Ro)
France	/ˈfɹɑːns/ (En), /ˈfɹænts/ (En), /fʁɑ̃s/ (Fr)

Table 5.2: IPA sequences of few words in different languages indicated using language codes in the parenthesis ('En' for English, 'Du' for Dutch, 'De' for German, 'Ro' for Romanian, and 'Fr' for French).

task but they are more general and applicable in other bridge language applications as well. First, I will describe Bridge-CCA [102] which is an extension of CCA to handle bridge language applications. Bridge-CCA learns mappings from all the languages into an interlingual representation. It learns a single mapping function, from the phonemic space into the interlingual representation, irrespective of the language and hence it ignores language specific phonemic variation. For *e.g.*, as shown in Table 5.2, the word China has different IPA representations depending on the language. Since Bridge-CCA uses a single mapping function for the phonemic space it fails to map these different representations into a same point in the interlingual representation. Later, I propose a regularization framework which learns multiple mapping functions for the phonemic space depending on the language. Thus my approach can map different representations of China into a same point in the interlingual representation and handles language specific phonemic variation. Subsequently I will report empirical evidence for multilingual transliteration task.

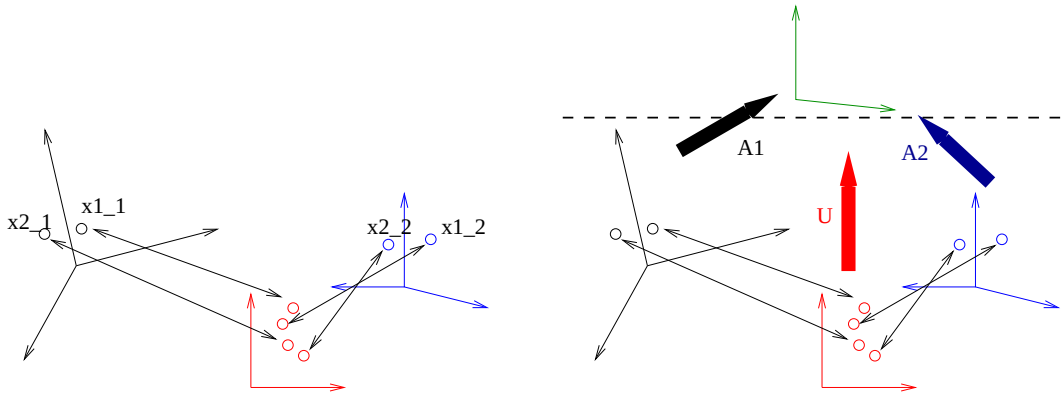
5.2 Bridge-CCA

Bridge-CCA [102] is originally proposed in the context of two language pairs. Though it can be extended for multiple languages, in order to be faithful, I describe it as it is presented by the authors [102]. In this case, the input training data consists of n_1 (word, IPA) pairs in one language represented as $X_1(d_1 \times n_1)$ and $P_1(c \times n_1)$ and n_2 (word, IPA) pairs from another language captured using matrices $X_2(d_2 \times n_2)$ and $P_2(c \times n_2)$. Fig. 5.1(a) shows example points from two character spaces (black and blue spaces) into the common IPA space (red). The directed arrows indicate alignment between examples. Bridge-CCA tries to map points from these individual feature spaces into a common subspace as shown in Fig. 5.1(b). Bridge-CCA reduces this problem into the standard CCA problem by constructing a composite feature space of size $d_1 + d_2$, which is simply a concatenation of the character feature spaces from both the languages. Fig. 5.2 shows example words in English and Bulgarian in the composite feature space. With this composite feature space, the training data can be combined as follows:

$$X_{comp} = \begin{pmatrix} X_1 & \mathbf{0} \\ \mathbf{0} & X_2 \end{pmatrix}_{(d_1+d_2)\times(n_1+n_2)} \quad \text{and} \quad P_{comp} = \begin{pmatrix} P_1 & P_2 \end{pmatrix}_{c\times(n_1+n_2)} \tag{5.1}$$

Now, it finds the projection directions $\mathbf{a} = [\mathbf{a}^{(1)^T} \ \mathbf{a}^{(2)^T}]^T \in \mathbb{R}^{d_1+d_2}$ and $\mathbf{p} \in \mathbb{R}^c$ such that the following objective function is maximized:

$$\frac{\mathbf{a}^T X_{comp} P_{comp}^T \mathbf{p}}{\sqrt{\mathbf{a}^T X_{comp} X_{comp}^T \mathbf{a}} \ \sqrt{\mathbf{p}^T P_{comp} P_{comp}^T \mathbf{p}}} \tag{5.2}$$



(a) Aligned training data between black and blue spaces indicated by arrows.

(b) Bridge-CCA maps all the input points into a common interlingual representation.

Figure 5.1: On the left, we have aligned training from two character spaces (black and blue spaces) into the common phonemic space. Bridge-CCA maps all the points into a common subspace (the 2-dimensional green space at the top).

Composite feature space	**Phonemic space**
{ h, e, a, d, #h, he, ea, ad, d#, 0, 0, 0, 0, 0, 0 ⋯ }	{ h,ɛ, d, $\#h$, hɛ, ɛd, $d\#$ }
{ ⋯ 0, 0, 0, 0, 0, 0, 0, 0, 0, 2 к, а, #к, ка,ак, к# }	{ $2k$, a, $\#k$, ka, ak, $k\#$ }

Figure 5.2: The composite character feature space for the English word 'head' and the Bulgarian word 'как' are shown here. The Bulgarian n-gram character features for the English word head are padded with zero's. Similarly, the English n-gram character features for the Bulgarian word are padded with zero.

$$= \frac{\mathbf{a}^{(1)^T} X_1 P_1^T \mathbf{p} + \mathbf{a}^{(2)^T} X_2 P_2^T \mathbf{p}}{\sqrt{\mathbf{a}^{(1)^T} X_1 X_1^T \mathbf{a}^{(1)} + \mathbf{a}^{(2)^T} X_2 X_2^T \mathbf{a}^{(2)}} \sqrt{\mathbf{p}^T (P_1 P_1^T + P_2 P_2^T) \mathbf{p}}} \tag{5.3}$$

By following the procedure similar to that of CCA (Sec. 2.2.1), the projection directions can be solved using the following generalized eigenvalue system:

$$M = P_1 X_1^T (X_1 X_1^T)^{-1} X_1 P_1^T + P_2 X_2^T (X_2 X_2^T)^{-1} X_2 P_2^T \tag{5.4}$$
$$M\,\mathbf{p} = \lambda^2 (P_1 P_1^T + P_2 P_2^T)\mathbf{p} \tag{5.5}$$
$$X_1 P_1^T\,\mathbf{p} = \lambda\; X_1 X_1^T \mathbf{a}^{(1)} \tag{5.6}$$
$$X_2 P_2^T\,\mathbf{p} = \lambda\; X_2 X_2^T \mathbf{a}^{(2)} \tag{5.7}$$

Similar to the CCA, we form the mapping matrices A_1 and A_2 with the top k eigenvectors $\mathbf{a}^{(1)}$ and $\mathbf{a}^{(2)}$ as columns respectively. During prediction, the similarity between two names in each language $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ is computed using:

$$\text{sim}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \frac{\mathbf{x}^{(1)^T} A_1 A_2^T \mathbf{x}^{(2)}}{\sqrt{\mathbf{x}^{(1)^T} A_1 A_1^T \mathbf{x}^{(1)}} \sqrt{\mathbf{x}^{(2)^T} A_2 A_2^T \mathbf{x}^{(2)}}} \tag{5.8}$$

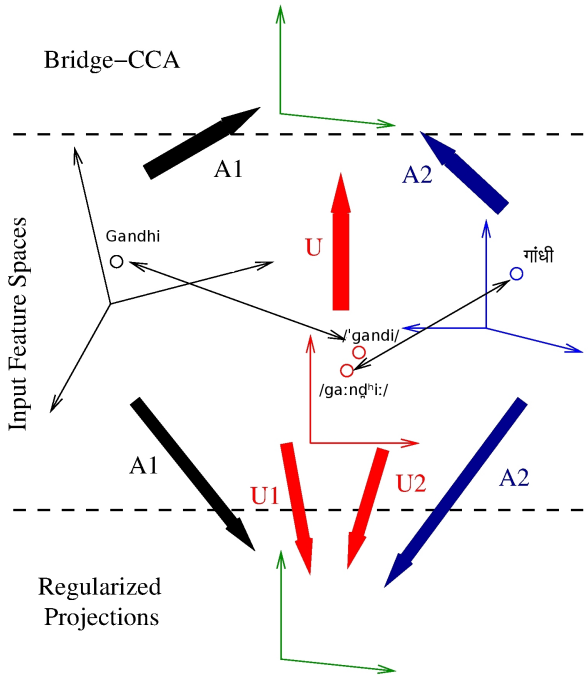


Figure 5.3: A single name (Gandhi) is shown in all the input feature spaces. The alignment between the character and phonemic space is indicated with double dimensional arrows. Bridge-CCA uses a single mapping function U from the phonemic space into the common subspace (the 2-dimensional green space at the top), where as our approach uses two mapping functions U_1 and U_2, one for each language, to map the IPA sequences into the common subspace.

5.3 Regularized Projections

As described earlier, Bridge-CCA uses a single mapping function (U) for the phonemic space irrespective of the language. In contrary, my approach learns different projection directions (U_1 and U_2) for phonemic space depending on the language. In what follows, I will describe the details of my approach.

During the training stage, for each language, we find mappings $A_l \in \mathbb{R}^{(d_l \times k)}$ and $U_l \in \mathbb{R}^{(c \times k)}$ from the character and phonemic spaces into a k dimensional subspace (or an interlingual representation) such that a name gets mapped to the *same* k dimensional vector irrespective of the language. Recall that, given a name $\mathbf{x}^{(l)}$ it gets mapped to the vector $A_l^T \mathbf{x}^{(l)}$ and similarly an IPA sequence $\mathbf{p}^{(l)}$ gets mapped to $U_l^T \mathbf{p}^{(l)}$. During the testing stage, given a name $\mathbf{x}^{(l)}$ in the source (l^{th}) language, we find its transliteration in the target (m^{th}) language $\mathbf{x}^{(m)}$ by solving the following decoding problem:

$$\arg\min_{\mathbf{x}^{(m)}} \; L\big(\mathbf{x}^{(l)}, \mathbf{x}^{(m)}\big) \tag{5.9}$$

$$\text{where} \qquad L\big(\mathbf{x}^{(l)}, \mathbf{x}^{(m)}\big) \;=\; \min_{\mathbf{p} \in \mathbb{R}^c} \|A_l^T \mathbf{x}^{(l)} - U_l^T \mathbf{p}\|^2 + \|A_m^T \mathbf{x}^{(m)} - U_m^T \mathbf{p}\|^2 \tag{5.10}$$

Intuitively, this formulation uses the source language mappings to find the IPA sequence $\mathbf{p}$ that is closest to the source name. Then it uses $\mathbf{p}$, along with the target language mappings, to identify the correct transliteration from a list of candidate transliterations.

At a high level, existing bridge language approaches such as Bridge-CCA [102] assume that $U_l \equiv U_m$ thus ignoring the language specific variation. To understand this, consider the example shown in Fig. 5.3. The middle portion of this Figure shows the name Gandhi (represented as point) in the character spaces of English and Hindi, three-dimensional blue and black spaces, and its IPA sequences in the phonemic space (the two-dimensional red space in the middle). Notice that, because of the phonemic variation, the same name is represented by two distinct points in the common phonemic space.[3] Now, since Bridge-CCA uses a single mapping function for both the IPA sequences, it fails to map these two distinct points into a common point in the interlingual subspace. The new formulation relaxes this hard constraint and learns different mapping directions (U_l and U_m) to map the two sequences and hence my approach can potentially map both the distinct IPA sequences into a single point. In the example shown in Fig. 5.3, my model (called Regularized Projections) finds two different mapping functions U_1 and U_2, one for each language, from the phonemic space into the common two-dimensional space at the bottom. As a result my approach successfully handles the language specific phonemic variation. At the same time we constrain the projection directions such that they behave similarly for the phonemic sounds that are observed in majority of the languages.

5.3.1 Model Formulation

In this section I first formulate the problem of finding the mapping directions (A_l and U_l) of each language as an optimization problem. In subsequent sections, I describe a method for solving the optimization problem (Sec. 5.3.1) and also derive a closed form solution for the prediction problem (Sec. 5.3.3). For simplicity, I describe my approach in terms of a single projection vectors, $\mathbf{a}^{(l)} \in \mathbb{R}^{d_l}$ and $\mathbf{u}^{(l)} \in \mathbb{R}^c$, rather than full matrices, but the generalization is straightforward.

Inspired by CCA [80], we find projection directions in the character and phonemic spaces of each language such that, after projection, a word is closer to its aligned IPA sequence. To understand this, assume that we have a name (say "Barack Obama") in all the languages[4]. Moreover, let its feature vectors be $\mathbf{x}^{(l)}$ and $\mathbf{p}^{(l)}$ $l = 1 \cdots L$ in the character and phonemic spaces respectively. Then, we might try to find projection directions $\mathbf{a}^{(l)}$ in each language and $\mathbf{u}$ in the common phonemic space such that:

$$\underset{\mathbf{a}^{(l)},\mathbf{u}}{\arg\min} \sum_{l=1}^{L} \left(\langle \mathbf{x}^{(l)}, \mathbf{a}^{(l)} \rangle - \langle \mathbf{p}^{(l)}, \mathbf{u} \rangle \right)^2 \tag{5.11}$$

This model assumes that the projection direction $\mathbf{u}$ in the phonemic space is same for all languages which is a hard constraint and does not handle the language specific variability as discussed in the previous section. I model the language specificity by relaxing this hard constraint.

In the new model, intuitively, the parameters corresponding to the phonemic sounds that occur in majority of the languages are shared across the languages while the parameters of the language specific sounds are modeled per each language. This is achieved by modeling the projection directions of the l^{th} language phonemic space $\mathbf{u}^{(l)}$ as $\mathbf{u}^{(l)} \leftarrow \mathbf{u} + \mathbf{r}^{(l)}$. The vector $\mathbf{u} \in \mathbb{R}^c$ is common to the phonemic spaces of all the languages and thus handles sounds that are observed in multiple languages. Where as $\mathbf{r}^{(l)} \in \mathbb{R}^c$, the residual vector, is specific to each language and accounts for

[3] In reality, as explained in the previous section, the common variation that is observed in the IPA sequences is that different features are triggered for different languages. But for visualization purpose, we showed the IPA sequences as if they differ in the feature values.

[4] The model does not require same names in different languages; this is used only for easier understanding.

the language specific phonemic variations. Thus the new formulation is given by:

$$\underset{\mathbf{a}^{(l)},\mathbf{u},\mathbf{r}^{(l)}}{\arg\min} \sum_{l=1}^{L} \Big\| \langle \mathbf{x}^{(l)}, \mathbf{a}^{(l)} \rangle - \langle \mathbf{p}^{(l)}, \mathbf{u} + \mathbf{r}^{(l)} \rangle \Big\|^2 + \tau \|\mathbf{r}^{(l)}\|^2 \tag{5.12}$$

where τ is a residual parameter. The first term of this summation ensures that a word and its IPA sequence get mapped to closer points in the subspace while the second term forces the residual vectors to be as small as possible. By enforcing the residual vectors to be small, this formulation encourages the sounds that occur in majority of the languages to be accounted by $\mathbf{u}$ and the sounds that are specific to the given language by $\mathbf{r}^{(l)}$. The idea of common and residual vectors is akin to a prior distribution in hierarchical Bayesian models [62]. But there is a subtle distinction, in the sense that the prior distribution in Bayesian models usually serves as a smoothing distribution, a distribution to back-off when there is not enough evidence in the data. In contrast, in my model, the common projection vector is responsible for explaining most part of the observed phenomenon and we back-off to the residual vectors only when it can not be explained by the common vector.

The final optimization problem is obtained by summing Eq. 5.12 over all the examples and all the languages and is given by:

$$\begin{aligned} \min_{\mathbf{a}^{(l)},\mathbf{u},\mathbf{r}^{(l)}} & \sum_{l}^{L} \Big(\frac{1}{n_l} \Big\| X_l^T \mathbf{a}^{(l)} - P_l^T (\mathbf{u} + \mathbf{r}^{(l)}) \Big\|^2 + \tau \, \|P_l^T \mathbf{r}^{(l)}\|^2 \Big) \\ & \text{s.t.} \ \sum_{l} \frac{1}{n_l} \|X_l^T \mathbf{a}^{(l)}\|^2 = 1 \ \text{ and } \ \sum_{l} \frac{1}{n_l} \|P_l^T \mathbf{u}\|^2 = 1 \end{aligned} \tag{5.13}$$

The length constraints avoid the trivial solution of setting all the vectors to zero. In the subsequent sections, I derive the solutions for the optimization problems presented here (Eq. 5.13 and Eq. 5.9).

5.3.2 Training the Model

We follow the standard procedure of forming the Lagrangian and setting its derivative to zero. The Lagrangian $\mathcal{L}$ of the optimization problem in Eq. 5.13 is given by:

$$\begin{aligned} \mathcal{L} \ = \ & \sum_{l} \frac{1}{n_l} \Big\| X_l^T \mathbf{a}^{(l)} - P_l^T (\mathbf{u} + \mathbf{r}^{(l)}) \Big\|^2 + \tau \sum_{l} \|P_l^T \mathbf{r}^{(l)}\|^2 \\ & + \alpha \Big(\sum_{l} \frac{1}{n_l} \|X_l^T \mathbf{a}^{(l)}\|^2 - 1 \Big) + \beta \Big(\sum_{l} \frac{1}{n_l} \|P_l^T \mathbf{u}\|^2 - 1 \Big) \end{aligned}$$

where α and β are Lagrangian multipliers corresponding to the length constraints. Differentiating $\mathcal{L}$ with respect to $\mathbf{a}^{(l)}, \mathbf{r}^{(l)}$ and $\mathbf{u}$ and setting the derivatives to zero yields the following equations, respectively:

$$\begin{aligned} (1+\alpha) X_l X_l^T \mathbf{a}^{(l)} - X_l P_l^T \mathbf{r}^{(l)} &= X_l P_l^T \mathbf{u} \\ -P_l X_l^T \mathbf{a}^{(l)} + (1 + \tau n_l) P_l P_l^T \mathbf{r}^{(l)} &= -P_l P_l^T \mathbf{u} \\ \sum_{l}^{L} \frac{1}{n_l} \Big(P_l X_l^T \mathbf{a}^{(l)} - P_l P_l^T \mathbf{r}^{(l)} \Big) &= (1+\beta) \sum_{l} \frac{1}{n_l} P_l P_l^T \mathbf{u} \end{aligned}$$

$$\begin{bmatrix}(1+\alpha)X_lX_l^T & -X_lP_l^T \\ -P_lX_l^T & (1+\tau n_l)P_lP_l^T\end{bmatrix}\begin{bmatrix}\mathbf{a}^{(l)} \\ \mathbf{r}^{(l)}\end{bmatrix} = \begin{bmatrix}X_lP_l^T \\ -P_lP_l^T\end{bmatrix}\mathbf{u} \tag{5.14}$$

$$\sum_l \frac{1}{n_l}\begin{bmatrix}P_lX_l^T & -P_lP_l^T\end{bmatrix}\begin{bmatrix}\mathbf{a}^{(l)} \\ \mathbf{r}^{(l)}\end{bmatrix} = (1+\beta)\sum_l \frac{1}{n_l}P_lP_l^T\mathbf{u} \tag{5.15}$$

$$\sum_l \frac{1}{n_l}\begin{bmatrix}-F_l^T & \frac{-G_l}{1+\tau n_l}\end{bmatrix}\begin{bmatrix}E_l & F_l \\ F_l^T & G_l\end{bmatrix}^{-1}\begin{bmatrix}-F_l \\ \frac{-G_l}{1+\tau n_l}\end{bmatrix}\mathbf{u} = (1+\beta)\sum_l \frac{1}{n_l}P_lP_l^T\,\mathbf{u} \tag{5.16}$$

$$\text{If } M_l = \left(E_l - F_lG_l^{-1}F_l^T\right)^{-1}, \text{ then } \begin{bmatrix}E_l & F_l \\ F_l^T & G_l\end{bmatrix}^{-1} = \begin{bmatrix}M_l & -M_lF_lG_l^{-1} \\ -G_l^{-1}F_l^TM_l & G_l^{-1}+G_l^{-1}F_l^TM_lF_lG_l^{-1}\end{bmatrix} \tag{5.17}$$

We can rewrite these equations in matrix form (as shown in Eqs. 5.14 and 5.15) since the solution becomes clear in this form. For brevity, let $E_l = (1+\alpha)X_lX_l^T$, $F_l = -X_lP_l^T$ and $G_l = (1+\tau n_l)P_lP_l^T$. Then, $\mathbf{u}$ can be solved for using the generalized eigenvalue problem shown in Eq. 5.16. This step involves computing an inverse of a (d_l+c) matrix and an eigenvalue problem of size c. This can be reduced further into a problem of smaller size by using inverse of a partitioned matrix as shown in Eq. 5.17. This identity reduces the matrix inverse computation from a problem of size $d_l + c$ into two smaller problems of size d_l and c. This reduces the time complexity considerably since the inverse computation is cubic in the size of the matrix.

Substituting Eq. 5.17 into Eq. 5.16 and further simplifying results in the following eigenvalue problem for solving $\mathbf{u}$:

$$\sum_l \frac{G_l + (\tau n_l)^2F_l^TM_lF_l}{n_l(1+\tau n_l)^2}\mathbf{u} = (1+\beta)\sum_l \frac{P_lP_l^T}{n_l}\,\mathbf{u}$$

where $M_l = \left(E_l - F_lG_l^{-1}F_l^T\right)^{-1}$. Notice that the term $E_l = (1+\alpha)X_lX_l^T$ depends on the Lagrangian multiplier α. Because of this, we cannot solve for both the parameters and the Lagrangian multipliers at the same time. One possible approach is to do an alternate optimization over the parameters and Lagrangian multipliers, but here I fix $\alpha = 1$ (Sec. 5.4) and solve for $\mathbf{u}$. Subsequently, we use $\mathbf{u}$ to solve for $\mathbf{a}^{(l)}$ and $\mathbf{r}^{(l)}$ as follows:

$$\mathbf{a}^{(l)} = -\frac{\tau n_l M_lF_l}{1+\tau n_l}\,\mathbf{u} \tag{5.18}$$

$$\mathbf{r}^{(l)} = \frac{\tau n_l G_l^{-1}F_l^TM_lF_l - I}{1+\tau n_l}\,\mathbf{u} \tag{5.19}$$

$$= -\frac{\mathbf{u}}{1+\tau n_l} - G_l^{-1}F_l^T\mathbf{a}^{(l)} \tag{5.20}$$

The new equations involve computing an inverse and eigenvalue problem of smaller sizes (*i.e.,* d_l and c). In order to increase the stability of the system we regularize G_l and E_l by adding τI. We retain the top k eigenvectors $\mathbf{u}$ and their corresponding $\mathbf{a}^{(l)}$ and $\mathbf{r}^{(l)}$ vectors and form the mappings U, A_l and R_l respectively. These mappings are used in predicting the transliteration of a word in any language into another language which is described in the following section.

5.3.3 Prediction

During the testing phase, given a source name we find the best possible IPA sequence for this language and then use it to find the appropriate target language transliteration. Formally, given a word $\mathbf{x}^{(l)}$ in l^{th} language, we find its transliteration into m^{th} language $\mathbf{x}^{(m)}$, by solving the optimization problem shown in Eq. 5.9, where $U_l = U + R_l$ and $U_m = U + R_m$. Similar to the previous case, the closed form solution can be found by taking the derivative with respect to the unknown phoneme sequence and the target language transliteration and setting it to zero (Appendix B.3). First, the IPA sequence $\mathbf{p}^*$ that minimizes $L\big(\mathbf{x}^{(l)}, \mathbf{x}^{(m)}\big)$ is given by:

$$\mathbf{p}^* = C_{lm}^{-1}\big(U_l A_l^T \mathbf{x}^{(i)} + U_m A_m^T \mathbf{x}^{(m)}\big) \tag{5.21}$$

where $C_{lm} = U_l U_l^T + U_m U_m^T$. We substitute this back in Eq. 5.10 and then solve for $\mathbf{x}^{(m)}$, the best transliteration in the m^{th} language, as:

$$A_m\big(I - U_m^T C_{lm}^{-1} U_m\big) A_m^T \mathbf{x}^{(m)} = A_m U_m^T C_{lm}^{-1} U_l A_l^T \mathbf{x}^{(l)} \tag{5.22}$$

Since U_l and U_m may not be full rank matrices, to increase the stability of this prediction, we use $C_{lm} = U_l U_l^T + U_m U_m^T + 0.001\ I$ where I is an identity matrix. Notice that this solution doesn't depend on the $\mathbf{p}^*$ and hence we don't need to compute it explicitly.

5.4 Multilingual Transliteration

In this section, I use regularized projections technique to address the problem of building a transliteration system between every pair of languages. Named entity (NE) transliteration involves transliterating a name in one language into another language and is shown to be crucial for machine translation (MT) [104, 2, 76, 113] and cross-lingual information retrieval (CLIR) [1, 120, 181]. There exists a large body of literature in transliteration, especially in the bilingual setting, well summarized by Ravi and Knight [147]. A straightforward supervised learning approach would require training data of name pairs between every pair of languages [104] or a set of common names transliterated from every language into a pivot language. Though it is relatively easy to obtain names transliterated into a pivot language (such as English), it is unlikely that such data sets contain same names. Bridge language approaches overcome the need for common names [102]. However, such approaches still require training data consisting of bilingual name transliterations (name-to-name mappings). Regularized projection technique enables us to relax the need for name transliterations by using international phonetic alphabet (IPA) in a manner akin to a "bridge language". As described in Sec. 5.1.1, we assume that we have a phonemic dictionary in L languages. An example phonemic dictionary in English and Bulgarian is shown in Table 5.1.

The use of IPA as the bridge language offers several advantages. As shown in Table. 5.1, it allows us to include any (word, IPA) pair in the training data and thus, it relaxes the need for name pairs as the training data. Since we only need a phoneme dictionary in each language, our approach does not require any bilingual resources to build the transliteration system. Moreover, since our training data can contain any word (not only the NEs), it is easier to obtain such a resource, for *e.g.,* the phoneme dictionaries obtained from Wiktionary contain at least 2000 (word, IPA) pairs in 21 languages. Our experiments show that we can build a decent transliteration system with 2000 pairs. Further more, IPA is a better choice for pivot language than say English,

because IPA allows us to represent phonemic sounds that does not exist in English. Finally, by simply adding a phoneme dictionary of $(L+1)^{\text{th}}$ language we can readily get a transliteration system into any of the existing L languages.

Using IPA as the bridge language poses some new challenges such as the language specific phonemic inventory. For *e.g.*, Mandarin doesn't have /v/, so it is frequently substituted with /w/ or /f/. Similarly, !Xóõ (Southern Khoisan, spoken in Botswana) has 122 consonants, mostly consisting of a large inventory of different word-initial click sounds [73], many of which do not exist in any other documented languages. Besides this language specific phonemic inventory, names have different IPA representations in different languages. For *e.g.*, as shown in Table 5.2, the IPA sequences for 'China' in English and Dutch have common IPA symbols but the English IPA sequence has additional symbols. Moreover, a name can have multiple pronunciations with in a language, *e.g.*, 'France' has two different IPA sequences in English (Table 5.2).

To summarize, for the multilingual transliteration problem, my approach uses the phoneme dictionaries of each language to learn mapping functions into an interlingual representation. Subsequently, given a pair of languages, and a query name in one of the languages, we use the mapping functions of those two language to identify possible name transliterations. The mapping functions explicitly model the language specific variability and thus account for fine grained differences.

5.4.1 Related Work

There is a large body of the literature in named entity transliteration, so I will describe only the most relevant ones to my approach. In transliteration, generative approaches aim to generate the target language transliteration of a given source name [104, 99, 69, 2] while discriminative approaches assume a list of target language names, obtained from other sources, and try to identify the correct transliteration [103, 165]. All these approaches require either bilingual name pairs or phoneme sequences to learn to transliterate between two languages. Thus, if we want to build a transliteration system between every pair of languages in a given set of languages then these approaches need resources between every pair of languages which can be prohibitive.

Bridge language approaches propose an alternative and use a resource rich language such as English as common language [102], but they still need bilingual resources. Moreover Bridge-CCA [102] uses a single mapping function for the phonemic space of all the languages and thus it can not handle language specific variability. In the original setting, authors use English as the pivot and since the phonemic inventory of English is fixed, irrespective of the target language, this may not be a serious concern. But it becomes crucial when we use IPA as the bridge language.

Approaches that map words in different languages into the common phonemic space have also been well studied. But most of these approaches use language specific resources such as CMU pronunciation dictionary [61] or a carefully constructed cost matrices for addition, substitution, and deletion of phonemes between a pair of languages [172, 196]. Variants of Soundex algorithm [138] such as Kodex [100] use hand constructed consonant to soundex code tables for name transliteration. Similar to my approach these variants only require soundex mappings of a new language to build transliteration system, but my model does not require mapping between n-gram characters and the IPA symbols. Alternatively unsupervised approaches have also been explored [147], but their accuracies are fairly low compared to the supervised and weekly supervised approaches.

	En.	Bg.	Ru.	Fr.	Ro.
Train	31K	36K	1141	36K	5211
Dev.	–	1264	2000	2717	430
Test	–	1264	2000	2717	431
# Features	5000	3998	2900	5000	3465

Table 5.3: Statistics of different data sets. Training data is monolingual phoneme dictionaries while development/test sets are bilingual name pairs between English and the respective language.

5.4.2 Evaluation Aspects

The experiments are designed to evaluate the following three aspects of my model, and of my approach to transliteration in general:

1. **IPA as bridge :** Unlike other phonemic based approaches (Sec. 5.4.1), I do not *explicitly* model the phoneme modifications between pairs of languages. Moreover, the phonemic dictionary in each language is obtained from Wiktionary (Sec. 5.4.3), which is likely to be noisy. So, the first aspect I want to evaluate is the effectiveness of using IPA as the bridge language. I also compare my approach with other bridge language approaches and establish the importance of modeling language specific variance.

2. **Multilinguality :** In my method, simply adding a phonemic dictionary of a new language allows us to extend our transliteration system into any of the existing languages. I evaluate the effect of data from a related, but different, language on a transliteration system between a given language pair.

3. **Complementarity :** Using IPA as bridge language allows us to build transliteration system into resource poor languages. But I also want to evaluate whether such an approach can help improving a transliteration system trained directly on name-pairs.

5.4.3 Data Sets and Experimental Setup

I use data sets from five languages in order to evaluate the effectiveness of our approach. The phonemic dictionaries (list of words and their IPA representations as shown in Table 5.1) are obtained from Wiktionary. The Wiktionary dump downloaded in October 2011 has at least 2000 (word, IPA) pairs in 21 languages which also includes some resource poor languages (*e.g.,* Armenian, Taiwanese, Turkish, *etc.*). In principle, my method allows us to build transliteration system into any of these language pairs without any additional information, but the final performances may vary depending on the language pair. But, in this paper, I use English (En.), Bulgarian (Bg.), Russian (Ru.), French (Fr.), and Romanian (Ro.) for evaluation purposes, as they suffice to showcase all the three aspects mentioned in the previous section. Table 5.3 shows the sizes of phonemic dictionaries used for training the models. The development and test sets between English and

the remaining language pairs are obtained from geonames data base[5]. These are geographic location names from different countries written in multiple languages. Notice that, the domain of the training and test sets is completely different.

I convert the phoneme dictionaries of each language into feature vectors. I use unigram and bigram features in the phonemic space and unigram, bigram, and trigram features in the character space. An example for feature generation is shown in Sec. 5.3. After converting the data into feature vectors, I retain the most frequent 5000 features. The last row of Table 5.3 shows the number of features in the character space of each of the languages. The phonemic space is common to all the languages and has 3777 features. Though the phonemic features are common to all the languages, as discussed in earlier, only a subset of features will be observed in a given language. For *e.g.,* in the data sets, of the total 3777 common phonetic features only 3312, 882, and 1009 features are observed in English, Bulgarian, and Russian languages respectively. This indicates the diversity in the phonemic inventory of different languages.

I compare my approach against Bridge-CCA, a state-of-the-art bridge language transliteration system which is known to perform competitively with other discriminative approaches [102]. I use the phoneme dictionaries in each language to train my approach, as well as the baseline system. The projection directions learnt during the training are used to find the transliteration for a test word as described in Sec. 5.3.3. I report performance in terms of the accuracy (exact match) of the top ranked transliteration and the mean reciprocal rank (MRR) of the correct transliteration. I find transliterations in both the directions (*i.e.,* target language transliterations given a source name and vice versa) and report average accuracies. The regularization parameter (τ) in both my approach and Bridge-CCA is tuned on the development data set using a line search between $[0, 1]$.

5.4.4 Description of Results

In this section I report experimental results on the three aspects mentioned above. Based on the sizes of the phoneme dictionaries (shown in Table 5.3), we should focus on English-Bulgarian and English-French language pairs to evaluate the 'IPA as bridge' and complementarity aspects. For the multilinguality aspect, we focus on comparing the performance of English-Russian and English-Romanian transliteration systems

5.4.4.1 IPA as Bridge

Fig. 5.4 shows the performance of my approach with the residual parameter τ (in Eq. 5.13) on the development data set. When τ is small, the model does not attempt to constrain the projection directions U_i's and hence they tend to map names to completely unrelated vectors. As we increase the residual parameter, it forces the residual vectors (R_i) to be smaller and thus the subspaces identified for each language are closely tied together. Thus, it models the commonalities across languages and the essential language specific variability. Based on the performance curves on the development data, I fix $\tau = 50$ in the rest of the experiments.

Table 5.4 shows the results of Bridge-CCA and my approach on the four language pairs. I report results of my approach with the decoding proposed in Sec. 5.3.3 and a simple cosine similarity measure in the common-subspace, *i.e.,* $\cos\left(A_l^T \mathbf{x}^{(l)}, A_m^T \mathbf{x}^{(m)}\right)$. Comparison of the accuracies in rows 1, 2 and 3, show that simply using cosine similarity performs almost same as the Bridge-CCA

[5]http://www.geonames.org/

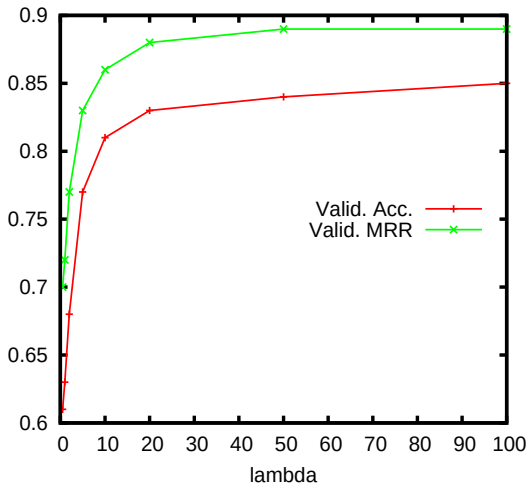


Figure 5.4: Performance of transliteration system with τ on English-Bulgarian language pair.

		En.–Bg.		En.–Ru.		En.–Fr.		En.–Ro.	
		Acc.	**MRR**	**Acc.**	**MRR**	**Acc.**	**MRR**	**Acc.**	**MRR**
1	Bridge-CCA	68.83	77.22	44.50	53.22	41.55	52.89	71.69	79.59
2	Ours (cosine)	67.68	76.52	45.07	53.63	42.45	53.06	74.13	81.28
3	Ours (Eq. 5.22)	83.70	88.32	63.47	73.01	70.68	78.13	77.38	**84.22**
4	Ours (cosine + Multi.)	68.91	77.44	49.15	57.20	42.55	53.02	77.49	84.04
5	Ours (Eq. 5.22 + Multi.)	**84.45**	**88.43**	**66.70**	**75.85**	**71.09**	**78.43**	**77.49**	84.04

Table 5.4: Results of our approach and the baseline system on the test set. The second block shows the results when our approach is trained only on phoneme dictionaries of the language pair, the third block shows results when we include other language data as well.

approach. However, using the decoding suggested in Eq. 5.22 gives significant improvements. To understand why the cosine angle between $A_l^T\mathbf{x}^{(l)}$ and $A_m^T\mathbf{x}^{(m)}$ is not the appropriate measure, assume that the vectors $\mathbf{x}^{(l)}$ and $\mathbf{x}^{(m)}$ are feature vectors of same name in two languages and let $\mathbf{p}$ be its true IPA representation. Then, since my model learns projection directions such that $A_l^T\mathbf{x}_i^{(l)} \approx U_l^T\mathbf{p}$,

$$\cos(A_l^T\mathbf{x}^{(l)}, A_m^T\mathbf{x}^{(m)}) = \cos\left((U+R_l)^T\mathbf{p}, (U+R_m)^T\mathbf{p}\right)$$

The additional residual matrices R_l and R_m make the cosine measure inappropriate. At the same time, my model forces the residual matrices to be minimal and this is probably the reason why it performs competitively with the Bridge-CCA. On the other hand, the decoding method shown in Eq. 5.9 searches over the best possible phoneme sequence and thus yields significant improvements. In the rest of this Chapter, I report results with the decoding in Eq. 5.22 unless specified explicitly. My approach achieves a maximum improvement of 29.13% accuracy over Bridge-CCA in English-French and on an average it achieves 17.17% and 15.19% improvement in accuracy and MRR respectively. Notice that even though our Russian phoneme dictionary has only 1141 (word, IPA) pairs, my approach is able to achieve an accuracy of 63.47% and an MRR of 73% indicating

	En.–Bg.		En.–Fr.	
	Acc.	**MRR**	**Acc.**	**MRR**
CCA	95.57	96.76	95.82	96.67
Ours+CCA (dev. tuned)	95.69	96.90	96.14	96.90
Δ Err	2.7%	4.2%	7.5%	6.8%
Ours+CCA (test tuned)	95.80	96.95	96.34	97.04
Δ Err	5.4%	5.8%	12.3%	11.3%

Table 5.5: Comparison with a system trained on bilingual name pairs. I also show the percentage error reduction achieved by a linear combination of my approach and CCA.

that the correct name transliteration is, on an average, at rank 1 or 2.

5.4.4.2 Multilinguality

The fourth and fifth rows of Table 5.4 also show the multilingual results. In particular, I train our system on data from the three languages En., Bg., and Ru. and test it on En.–Bg. and En.–Ru. test sets. Similarly, I train a different system on data from En., Fr., and Ro. and evaluate it on En.–Fr. and En.–Ro. test sets. I split the languages based on the language family, Russian and Bulgarian are Slavonic languages while French and Romanian are Romance languages, and expect that languages in same family have similar pronunciations. Comparing the performance of my system with and without the multilingual data set, it is clear that having data from other languages helps improve the accuracy.

5.4.4.3 Complementarity

In the final experiment, I want to compare the performance of my approach, which uses only monolingual resources, with a transliteration system trained using bilingual name pairs. I train a CCA based transliteration system [182] on a training data of 3792 and 8151 location name pairs. Notice that, for this approach, the training and test data for this system are from the same domain and thus it has an additional advantage over our approach, which is trained on whatever happens to be on Wiktionary.

The second row of Table 5.5 shows the results of CCA on English-Bulgarian and English-French language pairs. CCA achieves high accuracies even though the training data is relatively small, most likely because of the domain match between training and test data sets. As another baseline, I tried using Google machine translation API to transliterate the English names of the En.–Bg. test set. I hoped that since these are names, the translation engine would simply transliterate them and return the result. Of the output, I observed that about 500 names are passed through the MT system unchanged and so I ignore them. On the remaining names, it achieved an accuracy of 76.15% and the average string edit distance of the returned transliteration to the true transliteration is about 3.74. The relatively low accuracy of the transliterations obtained from Google MT system, compared to the 95% accuracy CCA baseline, is probably due to the fact that, presumably, it is a transliteration generation system unlike CCA which is a transliteration mining approach. For lack of fair comparison, I don't report the accuracies of the Google transliteration output in

Table 5.5.

Table 5.5 also shows the results of my system when combined with the CCA approach. For a given English word, I score the candidate transliterations using my approach and then linearly combine their scores with the scores assigned by CCA. I do a line search between $[0, 1]$ for the appropriate weight combination. The third and fourth rows of Table 5.5 show the results of the linear combination when the weight is tuned based on the development and test sets respectively. The improvements, though not significant, are encouraging and suggest that a sophisticated way of combining these different systems may yield significant improvements. This experiment shows that a transliteration system trained on word and IPA representations can actually augment a system trained on bilingual name pairs leading to an improved performance.

5.5 Discussion

In this Chapter, I proposed a regularization framework for the bridge language approaches and showed its effectiveness on the name transliteration task. My approach learns multiple projection directions for the bridge language based on the language it is paired with. Thus it accounts for language specific phenomenon.

I evaluated my model on multilingual name transliteration task. For this task, I use international phonetic alphabet (IPA) as the bridge language. The use of IPA allowed us to build a transliteration system between L languages using only monolingual resources. Since my model accounts for language specific phenomenon, it achieves significantly better results compared to other bridge language based approaches. Experimental results also show that a transliteration system built using IPA data can also help improve the accuracy of a transliteration system trained on bilingual name pairs.

While the use of phoneme dictionaries (words, IPA pairs) allowed us to build a transliteration system without any bilingual names, a transliteration system built purely on phoneme dictionaries will not be able to handle phonemes that occur only in names. For *e.g.*, Bulgarian has phoneme /x^{j}/ that occur only in the foreign names. In order to handle such phonemes, either we need to include some bilingual name pairs as part of the training data or combine the system with a transliteration system trained on the name pairs (as described in Sec. 5.4.4.3).

Chapter 6

Discriminative Reranking

Mapping inputs to outputs lies at the heart of many NLP applications. For *e.g.,* given a sentence as input: part-of-speech (POS) tagging involves finding the appropriate POS tag sequence [175]; parsing involves finding the appropriate tree structure [106] and statistical machine translation (SMT) involves finding correct target language translation [22]. The accuracy achieved on such tasks can often be improved with the help of a discriminative reranking step [36, 28, 161, 189]. Reranking step allows us to use arbitrary features which are often difficult to include in the baseline models due to the computational tractability issues. But the effectiveness of reranking depends on the joint features defined over both input and output spaces. This has led the community to spend substantial efforts in defining joint features for reranking [52, 31]. Unfortunately, developing joint features over the input and output space can be challenging, especially in problems where the exact mapping between the input and the output is unclear (for instance, in automatic caption generation for images, semantic parsing or non-literal translation).

In this Chapter, inspired by dimensionality reduction, I propose a novel family of reranking algorithms based on learning separate low-dimensional embeddings of the task's input and output spaces [91]. This embedding is learned in such a way that prediction becomes a low-dimensional nearest-neighbor search (Eq. 2.14), which can be done computationally efficiently. A key quality of my approach is that feature engineering can be done *separately* on the input and output spaces; the relationship between inputs and outputs is learned automatically. Since my approach requires within-space features, it makes feature engineering relatively easy.

6.1 Problem Setting

Let $\mathbf{x}_i \in \mathbb{R}^{d_1}$ and $\mathbf{y}_i \in \mathbb{R}^{d_2}$ be the feature vectors representing the $i^{th}(1 \cdots n)$ input and reference output (or true output) from the training data. For *e.g.,* in the POS tagging problem input is a sentence and output is a POS tag sequence while in the parsing task the input is again a sentence and the output is a parse tree. Each input is also associated with m_i number of candidate outputs, usually produced using a baseline system, and are represented as $\widehat{\mathbf{y}}_{ij} \in \mathbb{R}^{d_2}\ j = 1 \cdots m_i$. Each candidate output ($\widehat{\mathbf{y}}_{ij}$) is also associated with a non-negative loss L_{ij}. Notice that there are absolutely no constraints on the loss function. Finally, let $X\ (d_1 \times n)$ and $Y\ (d_2 \times n)$ denote the data matrices with $\mathbf{x}_i$ and $\mathbf{y}_i$ as columns respectively.

For clarity, I will discuss my approach in the context of POS tagging, though of course it generalizes to any reranking problem. As mentioned above, in this particular problem, the input is a sentence and the output is a POS tag sequence. For brevity, I refer to the feature space of sentence and tag sequences as word space and tag space respectively. Table 6.1 shows an example sen-

Input ($\mathbf{x}_i$) **Reference Output** ($\mathbf{y}_i$)	Buyers stepped in to the futures pit . NNS VBD IN TO DT NNS NN .		
Candidate Outputs ($\widehat{\mathbf{y}}_{ij}$)	**Score**	**Tag sequence**	**Loss**
	-0.1947	NNS VBD RP TO DT NNS NN .	0.12
	-6.8068	NNS VBD RB TO DT NNS NN .	0.12
	-7.0514	NNS VBD IN TO DT NNS NN .	0
	-7.1408	NNS VBD RP TO DT NNS VB .	0.25
	-13.7528	NNS VBD RB TO DT NNS VB .	0.25
	-13.9974	NNS VBD IN TO DT NNS VB .	0.12
	-26.1895	NNS VBD FW TO DT NNS NN .	0.12
	-32.9977	NNS VBD RP TO VBP NNS NN .	0.25
	-33.1356	NNS VBD FW TO DT NNS VB .	0.25
	-35.5381	NNS VBD RP TO NNP NNS NN .	0.25

Table 6.1: Example input sentence and output tag sequences for the POS tagging task. The first block shows the input sentence ($\mathbf{x}_i$) and its reference tag sequence ($\mathbf{y}_i$). The next block shows the candidate POS tag sequence output by a baseline trigram HMM tagger. The candidates are ordered based on the baseline decoder score. Notice that the reference tag sequence is ranked third. The incorrect tags are shown in red. The loss of a candidate sequence is calculated as the proportion of mismatched tags.

tence, its reference and candidate POS tag sequences. At test time, given a sentence and a list of candidate POS tag sequences as input, we run a feature extractor on the input sentence to obtain a representation $\mathbf{x} \in \mathbb{R}^{d_1}$; we run an *independent* feature extractor on each of the m-many outputs to obtain representations $\widehat{\mathbf{y}}_1, \ldots, \widehat{\mathbf{y}}_m \in \mathbb{R}^{d_2}$. We will project all of these points down to a low k-dimensional space by means of matrices $A \in \mathbb{R}^{d_1 \times k}$ (for $\mathbf{x}$) and $B \in \mathbb{R}^{d_2 \times k}$ (for $\widehat{\mathbf{y}}$). We then select as the output the $\widehat{\mathbf{y}}_j$ from the tag space that maximizes cosine similar to $\mathbf{x}$ in the lower-dimensional space given as:

$$\cos(A^T\mathbf{x}, B^T\widehat{\mathbf{y}}_j) = \frac{\mathbf{x}^T AB^T\widehat{\mathbf{y}}_j}{\sqrt{\mathbf{x}^T AA^T\mathbf{x}}\sqrt{\widehat{\mathbf{y}}_j^T BB^T\widehat{\mathbf{y}}_j}} \tag{6.1}$$

The goal is to learn the projection matrices A and B so that the result of this operation is a low-loss output. Given training data of sentences and their reference tag sequences, my approach implicitly uses all possible $d_1 \times d_2$ pairwise feature combinations across the views and learns the matrices A and B that can map a given sentence (as its feature vector) to its corresponding tag sequence. Considering all possible pairwise combinations enables our model to automatically handle long range dependencies such as a word at a position effecting the tag choice at any other position.

6.2 Models for Low-Dimensional Reranking

In this section, I describe the approach to learning low-dimensional representations for reranking. For simplicity, I discuss everything in the context of POS tagging but the models are general and apply to other situations as well. I propose both generative-style and discriminative-style approaches to formalizing this intuition, as well as a softened variant of the discriminative model.

In the subsequent section, we discuss computational issues related to these models. For easier exposition, I describe our models in one-dimensional setting (*i.e.,* to find vectors **a** and **b** instead of A and B) as it can be extended trivially.

6.2.1 A Generative-Style Model

The first model I propose is akin to a generative probabilistic model, in the sense that it attempts to model the relationship between an input and its reference output, without taking alternate possible outputs into account. In the context of the intuition sketched in the previous section, the idea is to choose A and B so as to maximize the cosine similarities on the training data between each input and it's correct (or minimal-loss) output. This objective function is same as that of CCA (Sec. 2.2). This model intentionally *ignores* the information present in the alternative, incorrect outputs. The hope is that by making the cosine similarities with the best output as high as possible, all the alternate outputs will look bad in comparison.

Though this model is a direct application of CCA, I briefly describe it here so that the intuition of further models become clear. Given a training data of sentences and their reference tag sequences represented as X and Y (Sec. 6.1), the generative model finds projection directions, in word and tag spaces, along which the aligned sentence and tag sequence pairs have maximum cosine similarity. In the one-dimensional setting, it finds directions $\mathbf{a} \in \mathbb{R}^{d_1}$ and $\mathbf{b} \in \mathbb{R}^{d_2}$ such that the correlation as defined in Eq. 6.2 is maximized.

$$\arg\max_{\mathbf{a},\mathbf{b}} \quad \frac{\mathbf{a}^T X Y^T \mathbf{b}}{\sqrt{\mathbf{a}^T X X^T \mathbf{a}}\sqrt{\mathbf{b}^T Y Y^T \mathbf{b}}} \tag{6.2}$$

As discussed in Sec. 2.4 maximizing this objective function is same as learning a classifier in the joint feature space. To recall the expression Eq. 2.23;

$$\mathbf{a}^T X Y^T \mathbf{b} = \sum_{i=1}^{n} \langle \mathbf{a}, \mathbf{x}_i \rangle \langle \mathbf{b}, \mathbf{y}_j \rangle = \sum_{i=1}^{n} \Big(\sum_{l,m=1}^{d_1,d_2} w_{lm} \phi_i^{lm} \Big) \tag{6.3}$$

where we replaced the scalars $\mathbf{x}_i(l)\mathbf{y}_i(m)$ and $\mathbf{a}(l)\mathbf{b}(m)$ with ϕ_i^{lm} and w_{lm} respectively. From the above expression, it is clear that the generative model considers all possible pairwise combinations of the input and output features ($d_1 \times d_2$) and learns which of them are more important than others. Intuitively, it puts higher weight on a word and tag pair that co-occur frequently in the training data, at the same time each of them is infrequent in its own view.

6.2.2 A Discriminative-Style Model

The primary disadvantage of our generative model is that it only uses input sentences and their reference tag sequences and does *not* use the incorrect candidate tag sequences of a given sentence at all. In what follows, I describe a model that utilize the incorrect candidate tag sequences as negative examples to improve the projection directions (**a** and **b**). I achieve this by imposing constraints that explicitly penalize ranking high-loss outputs higher than low-loss outputs. For *e.g.,* as shown in Table 6.1, the incorrect tag sequences are ranked higher (at positions 1 and 2) compared to the reference tag sequence which has lower loss. So, I will add constraints to penalize such situations. The following two models explicitly attempt to score the reference tag sequence higher

than the incorrect tag sequences, as is often done in the context of maximum-margin structure prediction techniques [173, 178].

In this section, I describe a discriminative model that keeps track of the margin deviations and finds the projection directions iteratively. Intuitively, after projecting into the lower dimensional subspace, the cosine similarity of a sentence to its reference tag sequence must be greater than that of its incorrect candidate tag sequences. Moreover, the margin between these similarities should be proportional to the loss of the candidate translation, *i.e.,* the more dissimilar a candidate tag sequence to its reference is, the farther it should be from the reference in the projected space. From the decomposition shown in Eq. 6.3, for a given pair of source sentence $\mathbf{x}_i$ and a tag sequence $\mathbf{y}_j$, the generative model assigns a score of :

$$\langle \mathbf{a}, \mathbf{x}_i \rangle \langle \mathbf{b}, \mathbf{y}_j \rangle = \mathbf{a}^T \mathbf{x}_i \mathbf{y}_j^T \mathbf{b}$$

Each input sentence is also associated with a list of candidate tag sequences and since each of these candidate sequences are incorrect, they should be assigned a score less than that of the reference tag sequence. Drawing ideas from structure prediction literature [8], I modify the objective function in order to include these terms. This idea can be captured using a loss augmented margin constraint for each sentence, tag sequence pair. Let ξ_i denote a non-negative slack variable, then I define the objective function of the discriminative model as follows:

$$\begin{aligned} &\arg \max_{\mathbf{a},\mathbf{b},\xi \geq 0} \quad \frac{\lambda}{1-\lambda} \mathbf{a}^T X Y^T \mathbf{b} - \sum_i \xi_i \\ &\text{s.t. } \mathbf{a}^T X X^T \mathbf{a} = 1 \quad \text{and} \quad \mathbf{b}^T Y Y^T \mathbf{b} = 1 \\ &\forall i \; \forall j \quad \mathbf{a}^T \mathbf{x}_i \mathbf{y}_i^T \mathbf{b} - \mathbf{a}^T \mathbf{x}_i \widehat{\mathbf{y}}_{ij}^T \mathbf{b} \geq 1 - \frac{\xi_i}{L_{ij}} \end{aligned} \tag{6.4}$$

where $0 \leq \lambda \leq 1$ is a weight parameter. This objective function ensures that the margin between the reference and the candidate tag sequences in the projected space (as given by $\mathbf{a}^T \mathbf{x}_i \mathbf{y}_i^T \mathbf{b} - \mathbf{a}^T \mathbf{x}_i \widehat{\mathbf{y}}_{ij}^T \mathbf{b}$) is proportional to its loss (L_{ij}). Notice that the slack is defined for each sentence and it remains the same for all of its candidate tag sequences.

6.2.3 A Softened Discriminative Model

One disadvantage to the discriminative model described in the previous section is that it cannot be optimized in closed form (as discussed in the next section). In this section, we consider a model that lies between the generative model and the (fully) discriminative model. This softened model has attractive computational properties (it is easy to compute) and will also form a building block for the optimization of the full discriminative model.

For each sentence $\mathbf{x}_i$, its reference tag sequence $\mathbf{y}_i$ should be assigned a higher score than any of its candidate tag sequences $\widehat{\mathbf{y}}_{ij}$ *i.e.,* we want to maximize $\mathbf{a}^T \mathbf{x}_i \mathbf{y}_i^T \mathbf{b} - \mathbf{a}^T \mathbf{x}_i \widehat{\mathbf{y}}_{ij}^T \mathbf{b}$. In the fully discriminative model, we enforce that this is at least one (modulo slack). In the relaxed version, we instead require that this hold *on average*. In order to achieve this we add the following terms to the objective function: $\forall j = 1 \cdots m_i$

$$\mathbf{a}^T \mathbf{x}_i \mathbf{y}_i^T \mathbf{b} - \mathbf{a}^T \mathbf{x}_i \widehat{\mathbf{y}}_{ij}^T \mathbf{b} \quad = \quad \mathbf{a}^T \mathbf{x}_i \mathbf{r}_{ij}^T \mathbf{b} \tag{6.5}$$

where $\mathbf{r}_{ij} = \mathbf{y}_i - \widehat{\mathbf{y}}_{ij}$ is the residual vector between the reference and the candidate sequences.

Now, we simply sum all these terms for a given sentence weighted by their loss and encourage it to be as high as possible, *i.e.*,

$$\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\Big(\mathbf{a}^T\mathbf{x}_i\mathbf{r}_{ij}^T\mathbf{b}\Big) = \mathbf{a}^T\mathbf{x}_i\Big(\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\mathbf{r}_{ij}^T\Big)\mathbf{b} \tag{6.6}$$

The normalization by m_i takes care of unequal numbers of candidate tag sequences. Now let R denote a matrix of the same size as that of Y (*i.e.*, $d_2 \times n$) with its i^{th} column as given by $\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\mathbf{r}_{ij}$, then we add the following term to the generative objective function:

$$\sum_{i=1}^{n}\mathbf{a}^T\mathbf{x}_i\Big(\frac{1}{m_i}\sum_{j=1}^{m_i} L_{ij}\mathbf{r}_{ij}^T\Big)\mathbf{b} = \mathbf{a}^T XR^T\mathbf{b} \tag{6.7}$$

Finally, the projection directions are obtained by solving the following optimization problem :

$$\arg\max_{\mathbf{a},\mathbf{b}}\ (1-\lambda)\mathbf{a}^T XY^T\mathbf{b} + \lambda\,\mathbf{a}^T XR^T\mathbf{b} \qquad \text{s.t. } \mathbf{a}^T XX^T\mathbf{a} = 1 \ \text{ and } \ \mathbf{b}^T YY^T\mathbf{b} = 1 \tag{6.8}$$

where $0 \leq \lambda \leq 1$ is a constant to be tuned on the development set.

6.3 Optimization

In this section, I describe how to solve the optimization problems associated with the models that we discussed in the previous Section. First I discuss the solution of the generative model. Next, I discuss the *softened* discriminative model, since its solution will be used as a subroutine in the optimization of the fully discriminative model.

6.3.1 Optimizing the Generative Model

The optimization problem corresponding to the generative model turns out to be identical to that of CCA [80], which immediately suggests a solution by solving an eigensystem (Sec. 2.2.2.1). In particular, the projection directions are obtained by solving the generalized eigensystem:

$$\begin{pmatrix}\mathbf{0} & C_{xy}\\ C_{yx} & \mathbf{0}\end{pmatrix}\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix} = \begin{pmatrix}C_{xx} & \mathbf{0}\\ \mathbf{0} & C_{yy}\end{pmatrix}\begin{pmatrix}\mathbf{a}\\ \mathbf{b}\end{pmatrix} \tag{6.9}$$

where $C_{xx} = (1-\tau)XX^T + \tau I$, $C_{yy} = (1-\tau)YY^T + \tau I$ are covariance matrices, $C_{xy} = XY^T$ is the cross-covariance matrix, $C_{yx} = C_{xy}^T$, τ is a regularization parameter and I is an identity matrix of appropriate size. Using these eigenvectors as columns, we form projection matrices A and B. These projection matrices are used to map sentences and tag sequences into a common lower dimensional subspace. In general, using all the eigenvectors is sub-optimal from the generalization perspective, so the optimal size of this subspace is tuned based on the development set.

6.3.2 Optimizing the Softened Model

In the softened discriminative version, the summation of all the difference terms over all candidate tag sequences and sentences (Eq. 6.7), enables a simpler objective function whose optimum

can be derived by following a procedure very similar to that of the generative model. In particular, the projection directions are obtained by solving Eq. 6.9 except that C_{xy} is replaced with $X((1-\lambda)Y^T + \lambda R^T)$.

6.3.3 Optimizing the Discriminative Model

To solve the discriminative model, we begin by constructing the Lagrange dual. Let β_1 and β_2 and α_{ij} be the Lagrangian multipliers corresponding to the length and the margin constraints respectively, then the Lagrangian is given by:

$$\begin{aligned}\mathcal{L} &= \frac{1-\lambda}{\lambda}\,\mathbf{a}^T XY^T\mathbf{b} - \sum_{i=1}^{n}\xi_i - \beta_1\Big(\mathbf{a}^T XX^T\mathbf{a} - 1\Big) - \beta_2\Big(\mathbf{b}^T YY^T\mathbf{b} - 1\Big) \\ &\quad + \sum_{i=1,j=1}^{n,m_i}\alpha_{ij}\left(\mathbf{a}^T\mathbf{x}_i\mathbf{r}_{ij}^T\mathbf{b} - 1 + \frac{\xi_i}{L_{ij}}\right)\end{aligned}$$

Differentiating the Lagrangian with respect to the parameters $\mathbf{a}, \mathbf{b}$ and setting them to zero yields the solution for the parameters in terms of the Lagrangian multipliers α_{ij} as follows:

$$\begin{pmatrix}\mathbf{0} & C_{xy}^{\alpha} \\ C_{yx}^{\alpha} & \mathbf{0}\end{pmatrix}\begin{pmatrix}\mathbf{a} \\ \mathbf{b}\end{pmatrix} = \begin{pmatrix}C_{xx} & \mathbf{0} \\ \mathbf{0} & C_{yy}\end{pmatrix}\begin{pmatrix}\mathbf{a} \\ \mathbf{b}\end{pmatrix} \tag{6.10}$$

where $C_{xy}^{\alpha} = X\Big(\frac{1-\lambda}{\lambda}Y^T + R^T\Big)$ and R is a matrix of size $d_2 \times n$ with i^{th} column as given by $\frac{1}{m_i}\sum_{j=1}^{m_i}\alpha_{ij}\mathbf{r}_{ij}$. I use superscript α $\big($in $C_{xy}^{\alpha}\big)$ to indicate that this is dependent on the Lagrangian multipliers α_{ij}. In other words, the solution is similar to that of the previous formulation except that the residual vectors are weighted by the Lagrangian multipliers instead of the loss function. Unlike the max margin formulations of SVM, it is not easy to rewrite the parameters $\mathbf{a}, \mathbf{b}$ in terms of the Lagrangian multipliers α_{ij} as C_{xy}^{α} itself depends on α_{ij}'s. Hence, rewriting the parameters in terms of the Lagrangian multipliers and then solving the dual is not amenable in this case.

In order to solve this optimization problem, we resort to an alternate optimization technique in the primal space. First, we fix the Lagrangian multipliers α_{ij} and then solve for the parameters $\mathbf{a}, \mathbf{b}, \beta_1, \beta_2$ and ξ_i. Projection directions $\mathbf{a}, \mathbf{b}$ and their Lagrangian multipliers β_1, β_2 are obtained by solving the generalized eigenvalue problem given in Eq. 6.10. β_1 and β_2 are the eigenvalues. Using these projection directions, we determine the slack variable ξ_i for each sentence. In the second stage of the alternate optimization, we fix $\mathbf{a}, \mathbf{b}$ and ξ_i and take a gradient descent step along α_{ij} to minimize the function. We repeat this process until the convergence.

The pseudocode is shown in Alg. 1. First we initialize the Lagrangian multipliers proportional to the loss of the candidate tag sequences (step 1). This ensures that the eigenvectors solved in step 6 are same as the output given by the previous formulation (Sec. 6.2.3). In general, in the experiments, I observed that this is a good starting point. After solving the generalized eigenvalue problem in step. 6, we consider the top k eigenvectors, as determined by the error on the development set and normalize them so that they follow the length constraints (steps 7 and 8). In the rest of the algorithm, we use these normalized projection directions to find the slack values which are in turn used to find the update direction for the Lagrangian variables.

In step 10, we compute the potential slack value (ψ_{ij}) for each of the constraints so that the constraint is satisfied and then choose the minimum of the positive ψ_{ij} values as the slack for this

Algorithm 1 Alternate optimization algorithm for solving the parameters of the discriminative model.

Require: $X, Y, \widehat{Y}, L, \lambda, \tau$
Ensure: A, B
1: $\boldsymbol{\alpha} = L$;
2: $\mathbf{r}_{ij} = \mathbf{y}_i - \widehat{\mathbf{y}}_{ij}$; $C_{xx} = (1-\tau)XX^T + \tau I$; $C_{yy} = (1-\tau)YY^T + \tau I$
3: **repeat**
4: Form R with i^{th} column as $\frac{1}{m_i}\sum_{j=1}^{m_i} \alpha_{ij}\mathbf{r}_{ij}$
5: $C_{xy}^{\alpha} = X\left(\frac{1-\lambda}{\lambda}\, Y^T + R^T\right)$
6: Solve for the eigenvectors of Eq. 6.10. .
7: Form matrices A, B with top k eigenvectors as columns; k is determined using dev. set.
8: Let A_n & B_n be normalized versions of A and B s.t. they follow the length constraints.
9: **for** each sentence $i = 1 \cdots n$ **do**
10: $j = 1 \cdots m_i,\ \ \psi_{ij} = \left(1 - \mathbf{x}_i^T A_n B_n^T \mathbf{r}_{ij}\right) L_{ij}$
11: $\xi_i = \min\ \left\{0\ ,\ \psi_{ij} \mid \text{s.t. } \psi_{ij} > 0\right\}$
12: **if** $\xi_i > 0$ **then**
13: $d_{ij} = \mathbf{x}_i^T A_n B_n^T \mathbf{r}_{ij} - \left(1 - \frac{\xi_i}{L_{ij}}\right)$
14: $\alpha_{ij} = \alpha_{ij} - \gamma\, d_{ij}$
15: **end if**
16: **end for**
17: **until** slack values doesn't change
18: **return** A, B

sentence (step 11). If the chosen slack value is equal to zero, it implies that $\psi_{ij} \leq 0\ \forall j = 1 \cdots m_i$ which in turn implies that all the constraints of a given input sentence are satisfied by the current projection directions and hence there is no need to update the Lagrangian multipliers. Otherwise, some of the constraints are still not satisfied and hence we will update their corresponding Lagrangian multipliers in steps 13 and 14. In specific, step 13 computes the deviation of the margin constraints with the new slack value and step 14 updates the Lagrangian multipliers along the gradient direction.

In principle, this approach is similar to the cutting plane algorithm used to optimize slack rescaling version of Structured SVM [178], but it differs in selecting the slack variable (step 11). The cutting plane method chooses ξ_i as the maximum of $\{0, \psi_{ij}\}$ where as we choose the minimum of the positive ψ_{ij} values as the slack. Intuitively, this means that the cutting plane algorithm chooses a constraint that is most violated which results in fewer constraints. This is crucial in structured SVM, because solving the dual problem is cubic in terms of the number of examples and constraints. In contrast, my approach selects the slack such that at least one of the constraints is satisfied and adds all the remaining constraints to the active set. Since step 6 considers a weighted average of all these constraints the complexity depends only on the number of training examples and not the constraints.

6.3.4 Combining with Viterbi Decoding Score

All the three formulations discussed until now do not consider the Viterbi decoding score assigned to each candidate tag sequence. As explained in Collins and Koo [36], the decoding score

plays an important role in reranking the candidate sentences. In this section, I describe a simple linear combination of the Viterbi decoding score and the score obtained by projecting into the lower dimensional subspace, using mappings obtained by any of the three methods we saw earlier.

For a given sentence $\mathbf{x}_i$ and candidate tag sequence pair $\widehat{\mathbf{y}}_{ij}$, let s_{ij} and p_{ij} be the scores assigned by Viterbi decoding and the lower dimensional projections (Eq. 6.1), respectively. Then I define the final score for this pair as a simple linear combination of these two scores as:

$$\text{Score}(\mathbf{x}_i, \widehat{\mathbf{y}}_{ij}) = s_{ij} + w\, p_{ij} \tag{6.11}$$

The weight w is optimized using a grid search on the development data set, I search for w from 0 to 100 with an increments of 1 and choose the value for which the error is minimum on development set.

6.4 Reranking for POS Tagging

Reranking is relative less explored for POS tagging due to the already higher accuracies of baseline taggers in English [35], but it is shown to improve accuracies in other languages such as Chinese [82]. Since my models consider all pair-wise combination features they automatically consider the long range dependencies. In the following sections, I evaluate the effectiveness of our discriminative reranking models on POS tagging.

To summarize the approach, we convert the training data into feature vectors and use any of the three methods discussed above to find the lower dimensional projection directions (**a** and **b**). Each of those approaches involve solving a similar generalized eigenvalue problem (Eq. 6.9) with the cross covariance matrix C_{xy} defined differently in the three approaches. As discussed in Sec. 2.2.2, this problem can be solved in different ways, but I use the following approach since it reduces the size of the eigenvalue problem.

$$C_{yy}^{-1} C_{xy}^{T} C_{xx}^{-1} C_{xy}\, \mathbf{b} = \omega\, \mathbf{b} \tag{6.12}$$

$$\mathbf{a} = \frac{1}{\sqrt{\omega}}\, C_{xx}^{-1} C_{xy}\, \mathbf{b} \tag{6.13}$$

where ω is the eigenvalue. Assuming that $d_2 \ll d_1$, which is usually true in POS tagging because of the smaller tag vocabulary, these equations solve a smaller eivenvalue problem. After solving the eigenvalue problem, we form matrices A and B with columns as the top k eigenvectors **a** and **b** respectively. Given a new sentence and candidate tag sequence pair $(\mathbf{x}_i, \widehat{\mathbf{y}}_{ij})$, their similarity is obtained using Eq. 6.1. Now, based on the development data set we find the weights for the linear combination of the projection and Viterbi decoding scores (Eq. 6.11).

During the reranking stage, we first use Eq. 6.1 to compute the projection score for all the candidate tag sequences and then use Eq. 6.11 to combine this scores with the decoding score. The candidate tag sequences are reranked based on this final score.

6.4.1 Related Work

There has been lot of literature on POS tagging, discriminative reranking and the margin formulations of CCA. In this section, I discuss approaches that are most relevant to the problem and

the approach.

In NLP literature, discriminative reranking has been well explored for parsing [36, 28, 160, 123, 97] and statistical machine translation [161, 189, 115]. Collins [35] proposed two reranking approaches, namely boosting algorithm and a voted perceptron, for the POS tagging task. Later Huang *et al.* [82] propose a regularized version of the objective function used by Collins [35] and show an improved performance for Chinese. Their algorithm finds a weight vector $\mathbf{w}$ that minimizes the following loss function:

$$\begin{aligned} \text{Loss}(\mathbf{w}) &= \sum_{i,j=1}^{n,m_i} S_{ij} e^{-M_{ij}(\mathbf{w})} + R(\mathbf{w}) && (6.14) \\ S_{i,j} &= \text{Score}_d(\mathbf{x}_i, \mathbf{y}_i) - \text{Score}_d(\mathbf{x}_i, \widehat{\mathbf{y}}_{ij}) \\ M_{ij}(\mathbf{w}) &= -w_0 L_{ij} + \sum_k w_k \Big(h_k(\mathbf{x}_i, \mathbf{y}_i) - h_k(\mathbf{x}_i, \widehat{\mathbf{y}}_{ij}) \Big) && (6.15) \end{aligned}$$

where $R(\mathbf{w})$ is the regularization term on the weight vector and h_k are binary feature functions. In all of the above reranking approaches, the feature functions are defined jointly on the input and output, whereas in my approach, the features are defined separately within each view and the algorithm learns the relationship between them automatically. This is the primary difference between my approach and the existing rerankers.

In principle, the margin formulations that I described in this Chapter are similar to the max margin formulations of CCA [169] and maximum margin regression [168, 188]. These approaches solve the following optimization problem:

$$\begin{aligned} &\min \ \|W\|^2 + C\, \mathbf{1}^T \boldsymbol{\xi} && (6.16) \\ &\text{s.t.} \ \langle \mathbf{y}_i, W\phi(\mathbf{x})_i \rangle \geq 1 - \xi_i \quad \forall i = 1 \cdots n \end{aligned}$$

Where C is the cost parameter, W is a transformation between input and output spaces and $\boldsymbol{\xi}$ is the slack vector. My approach differs from these formulations in two main ways: the score assigned by the generative model (equivalent to CCA) for an input-output pair ($\mathbf{x}_i^T \mathbf{a}\mathbf{b}^T \mathbf{y}_i$) can be converted into this format by substituting $W \leftarrow \mathbf{b}\mathbf{a}^T$, but in doing so we are ignoring the rank constraint. It is often observed that, dimensionality reduction leads to an improved performance and thus the rank constraint becomes crucial. Another major difference is that, the constraints in Eq. 6.16 represent that any input and reference output pair should have at least a margin of 1, whereas in my approach, the constraints include incorrect outputs along with their loss value. In other words, my formulation is more suitable for the reranking problem while Eq. 6.16 is more suitable for regression or classification tasks.

6.4.2 Data Sets

In this section, I report POS tagging experiments on four languages: English, Chinese, French and Swedish. The data in all these languages is obtained from the CoNLL 2006 shared task on multilingual dependency parsing[1]. I only consider the word and its fine grained POS tag (columns 2 and 5 in respectively) and ignore the dependency links in the data. Table 6.2 shows the data statistics in each of these languages.

[1] http://ilk.uvt.nl/conll/

		Train	Dev.	Test
English	# sent. # words	15K 362K	2K 47K	1791 43K
Chinese	# sent. # words	50K 292K	4K 26K	3647 25K
French	# sent. # words	9K 254K	2K 57K	1351 40K
Swedish	# sent. # words	8K 137K	2K 31K	1431 28K

Table 6.2: Training and test data statistics.

My models	Word space: { he, the, ng, ing, ling, re, ure, sure} Tag space: {DT, NN, NN, DT_BOS, NN_DT, NN_NN }
Baseline Rerankers	{ (he,DT), (the,DT), (he,DT_BOS), (the,DT_BOS), (ng,NN), (ng,DT_NN), (ing,NN), (ing,DT_NN), (ling,NN), (ling,DT_NN), (re,NN), (re,NN_NN), (ure,NN), (ure,NN_NN), (sure,NN), (sure,DT_NN) }

Table 6.3: Features generated for the low-dimensional discriminative models and the baseline rerankers on the example sentence "the/DT selling/NN pressure/NN". BOS is the beginning of sentence marker.

I use a second order Hidden Markov Model [175] based tagger as a baseline tagger in our experiments. This model uses trigram transition and emission probabilities and is shown to achieve good accuracies in English and other languages [82]. I refer to this as the baseline tagger in the rest of this Chapter and is used to produce n-best list for each candidate sentence. The n-best list for training data is produced using multi-fold cross-validation similar to Collins and Koo [36] and Charniak and Johnson [28]. The first block of Table 6.4 shows the accuracies of the top-ranked tag sequence (according to the Viterbi decoding score) and the oracle accuracies on the 10-best list. As expected the accuracies on English and French are high and are on par with the state-of-the-art systems. From the oracle scores, it is clear that though there is a chance for improvement using reranking, the scope for improvement in English is less compared to the 5 point improvement reported for dependency parsing [28]. This indicates the difficulty of the reranking problem for POS tagging in well-resourced languages.

6.4.3 Reranking Features and Baselines

In this Chapter, except for Chinese, I use suffixes of length two to four as features in the word space and unigram and bigram tag sequences as features in the tag view. That is, I convert each word of the sentence into suffixes of length two to four and then treat each sentence as a bag of suffixes. Similarly, I treat a candidate POS tag sequence as a bag of unigram and bigram tag features. For Chinese, I use character suffixes of length one and two as features for the sentences and use the same unigram and bigram POS tag sequences on the tag view. We did not include any alignment based features, *i.e.,* features that depend on the position. Although it is possible to

	Developement Set				Test set			
	English	Chinese	French	Swedish	English	Chinese	French	Swedish
Baseline	96.74	92.55	96.94	93.22	96.15	92.31	97.41	93.23
Oracle	98.85	98.41	98.61	96.96	98.39	98.19	99.00	96.48
Collins (Sufx)	96.66	93.00	96.87	**93.50**	96.06	92.81	97.35	**93.44**
Regularized (Sufx)	96.60	93.12	96.90	93.36	96.00	92.88	97.38	93.35
Generative	96.82	93.14	97.18	93.46	96.24	**92.95**	97.43	93.26
Softened-Disc	96.85	93.14	**97.28**	93.49	**96.32**	92.87	**97.53**	93.24
Discriminative	**96.85**	**93.17**	97.27	**93.50**	96.3	92.91	**97.53**	93.36
Collins (n-gm)	96.74	93.14	97.06	93.44	96.13	92.74	97.54	93.45
Regularized (n-gm)	96.78	93.14	97.01	93.45	96.14	92.80	97.52	93.40

Table 6.4: Accuracy of the baseline HMM tagger and different reranking approaches. For comparison purposes, we also showed the results of Collins and Koo [36] its regularized versions with n-gram features. Our models use only suffix features so we bolded the best system among those that use suffix features. The improvements of our discriminative models are statistically significant at $p = 0.01$ and $p = 0.05$ levels on Chinese and English respectively.

incorporate the alignment based features by posing it as a feature selection problem as described in Sec. 4.2. Table 6.3 shows the features that are fired for an example sentence.

I compare my models with a boosting-based discriminative approach [36] and its regularized version [82]. In order to enable a fair comparison, I use suffix and tag pairs as features for both these models. For *e.g.*, we would generate the following features for the word 'selling' in the phrase "the/DT selling/NN pressure/NN": (ng, NN), (ng, DT_NN), (ing,NN), (ing,DT_NN), (ling,NN), (ling,DT_NN). For comparison purposes, I also show results by running these rankers with n-gram features [35].

6.4.4 Results

There are following hyper parameters in each of my models, regularization parameter τ, weight parameter λ in the discriminative and softened discriminative models, the weight for the linear combination of the Viterbi decoding score and the projection score (w) and, finally, the size of the lower dimensional subspace (k). I use grid search to tune these parameters based on the development data set. For discussion on the sensitivity to the hyperparameters and more discussion please refer to Appendix C.

Table 6.4 shows the results of different models on the development and test data sets. On the test data set, the baseline reranking approaches perform better than the HMM decoder in Chinese and Swedish languages, but they underperform in English and French languages. This is justifiable because the individual characters are good indicators of POS tag information for Chinese and this additional information is being exploited by the reranking approaches. Swedish, as a Germanic language has compound word phenomenon which makes the baseline HMM decoder weaker compared to English and French.

The fourth block shows the performance of my models on the same data sets. Except in Swedish, one of my models outperforms the baseline decoder and the other reranking approaches.

	English	Chinese	French	Swedish
Softened-Disc (all sentences)	+0.17	+0.56	+0.12	+0.01
Discriminative (all sentences)	+0.15	+0.6	+0.12	+0.13
Softened-Disc (changed sent. only)	+**0.92**	+4.31	+1.12	+0.08
Discriminative (changed sent. only)	+0.88	+**4.77**	+**0.9**	+**0.73**

Table 6.5: Improvement over the baseline systems on all sentences and only on the sentences for which the reranking models changed the top scoring candidate sequence.

	En.	Zh.	Fr.	Sv.
Generative	94.83	89.89	96.1	91.89
Softened-Disc	95.04	89.61	95.97	91.95
Discriminative	94.95	89.76	95.82	92.11

Table 6.6: Accuracies without combining with Viterbi decoding score.

The fact that my models outperform the baseline system and the other reranking approaches indicate that, by considering all the pairwise combinations of the input features low-dimensional discriminative reranking models capture long range dependencies that are left by other models. Among the different formulations of my approach, maximizing the margin between the correct and incorrect candidates performed better than generative, and ensuring that the margin is proportional to the loss of the candidate sequence (discriminative) led to even more improved results. Except in Chinese, the discriminative version performed at least as well as the other variants. Compared to the baseline decoder, the discriminative version achieves a maximum improvement of 0.6 points in Chinese while achieving 0.13, 0.12 and 0.15 points of improvement in Swedish, French and English languages respectively.

I also reported the results of the baseline rerankers with n-gram features in the fifth block of Table 6.4. I remind the reader that my models use only suffix features, so for a fair comparison the reader should compare my results with the baseline rerankers run with the suffix features. The performance of these baseline rankers improved when I include the n-gram features but it is still less than the discriminative model in most cases.

In the analysis, I found that on some sentences the top scoring POS tag sequence did not change even after reranking. Table 6.5 shows the performance improvement of my models on only the sentences whose top ranked tag sequence changed after reranking. For comparison, it also shows the average improvements on all sentences. From the Table, we can clearly see that the improvements are substantial. Finally, Table 6.6 shows the performance of our models without combining with the Viterbi decoding score. By comparing the accuracies in Tables. 6.4 and 6.6, it is clear that the low-dimensional discriminative rerankers are not as effective by themselves – which is in accordance with the behavior observed elsewhere [36] – but they provide an invaluable signal to the existing models.

6.5 Discussion

In this Chapter, I showed a novel use of dimensionality reduction techniques for discriminative reranking problem and show improvements for the POS tagging task in four different languages. Here, I restricted myself to showing the utility of the techniques and, hence, did not experiment with different features, though it is an important direction. By using only with-in space features, my models are able to beat the reranking approaches that use potentially more informative alignment-based features. As discussed in Chapter 4, it is also possible to include alignment-based features into the reranking models by posing the problem as a feature selection problem on the covariance matrices [93].

As in the previous Chapters, the reranking models involve an inverse computation and an eigenvalue problem. While these operations can be potentially expensive, there are alternative approximation techniques that scale well to large data sets [70]. I leave this for future work.

Chapter 7

Compositional Embeddings

In the previous Chapters, I discussed how to learn a lower dimensional subspace under two different situations: 1) when we have training data from multiple languages into a common bridge language, and 2) when an object is associated with a ranked list of objects from another view. In both the above settings, an object in each of the views is represented as a single vector and we learnt transformation functions to map these objects into a common low-dimensional representation. But in many cases, the objects have richer structure that is usually ignored. For example, in many IR tasks sentences/documents are typically represented as bag-of-words. While this representation is convenient for computation, it throws away lot of useful information such as the syntactic structure and the word meaning. An alternative representation is to use the syntactic tree structure of a sentence with nodes being words and an edge between two words characterizing the syntactic relation between the two words. This representation can be further enriched by replacing the lexical identity of each word by a d-dimensional vector which captures the meaning of the word. Using word vectors allows applications to automatically detect synonyms and handle them appropriately. But as mentioned earlier, representing an object as a vector is desirable due to its computational advantages. So we want to design models to compute a vector representation of an object by taking advantage of its structure. The vector representation output by our model can be substituted for a bag-of-word representation in any of the existing applications.

In this Chapter, I propose models to compute vector representation of an object based on its structure. For conciseness I call such function as *composition function* and discuss a general problem setting called *multi-view multivariate regression* (MMR) to learn the parameters of the composition function. For generality, I discuss MMR in the context of vector based compositionality learning (Sec. 7.1) and then a specific case of MMR to the problem of learning token based embeddings from type based representations (Sec. 7.2).

- **Vector based Compositionality Learning:** In this problem setting, we assume that the meaning of a word is represented as a point in a d-dimensional vector space [51, 159, 66] and the meaning of a phrase or a sentence can be represented by *a collection of points* in the same vector space. Since a phrase or a sentence has much more information than a word, we use a collection of points as opposed to a single point representation [126, 163]. The collection size depends on whether we are representing a phrase or a sentence and also on the task. The optimal value of collection size can be estimated from development data. In the rest of this Chapter, I mainly discuss models to learn point-wise representation of a sentence and it is trivial to obtain the rest of the points in the collection.

  Vector based semantic learning problem is to compute the semantic representation of a sentence based on the individual word representations and its syntactic structure. The aim of this problem is to find vector representations for sentences such that two sentences that

express the same meaning (*e.g.,* "I ate an apple" and "I had a fruit") lie closer in the d-dimensional space. This problem is different from the compositionality learning as setup in [114, 197, 171] where the aim to learn the representation of a sentence in terms of logic statements. While the representation in terms of logic is suitable for tasks such as question answering, it is not well suited for computing similarity between two texts [94]. In the rest of this Chapter, by compositionality learning I refer to the problem of vector based compositionality learning[1].

- **Learning Token Based Embeddings:** The second problem we consider in this Chapter is to learn token based embeddings from type based representations. Techniques like distributional semantics [51, 118, 176, 66] or interlingual representations (Chp. 4) learn vector representations of a word by aggregating information over all of its contexts. The learnt representation is dependent only on the word and is independent of any given context [128, 37]. But words are used in different senses depending on the context. Consider the classic example of the word 'bank'. It means river bank in the sentence "they pulled the canoe up on the bank" where as it means a financial institution in the sentence "he cashed a check at the bank". Representing a word using a single vector can not capture its different senses. In order to differentiate these two meanings of the same word, we may want to learn vector representations of the same word specific to each context.

  For conciseness, I refer to the representation of a word in isolation as *type* based representation/embedding and its representation in a specific context as *token* based representation/embedding[2]. In this problem setting, I discuss how to adapt type based embeddings to their tokens. As we will see later, this can be cast as a specific case of MMR and is very close to the problem of multivariate linear regression.

In my preliminary experiments I found out that compositionality learning depends crucially on the quality of the word embeddings. Simply using type based embeddings is not sufficient to perform better than a bag-of-word representation. So in this Chapter, I only discuss the modeling aspect of the compositionality learning problem. But I will present both modeling aspect and experimental evidence for the second problem, learning token based embeddings.

7.1 Compositionality Learning

Appropriate models for computing the meaning conveyed by a natural language text unit (such as a phrase, a sentence or a document) are essential in many NLP applications such document classification [127], question answering [122], information retrieval [121], and so on. The earliest work on compositionality and composition operators is the logic-based semantic frameworks developed by Montague [129]. The principle of compositionality states that the meaning of a sentence can be expressed in terms of the meaning of its constituent expressions and the rules of composition – which are in turn defined by the syntax of the sentence. As argued in [94], representation in terms of logic formulas is not well suited for modeling similarity between textual units. So, researchers started looking into alternative representations that are amenable to similarity computations.

[1] In general, compositionality learning refers to computing the meaning of textual n-grams such as phrases or sentences. But in this Chapter, I mainly discuss in the context of sentences but, once the composition function is learnt, with no additional effort it can be used to learn the meaning of phrases as well.

[2] I use token and "a word in a context" interchangeably.

7.1.1 Representations for Text

Since their introduction, vector-space models (VSM) [155, 180] have shown great success in NLP applications. Vector space models represent a document as a bag of words. While this representation is simple, it encounters sparseness problems due to the growing vocabulary size. Moreover, the synonymous and polysemous nature of words pose another problem (referred to as lexical gap) in these models. Dimensionality reduction techniques such as latent semantic analysis (LSA) [53] are proposed to reduce the dimensionality and are shown to address the sparseness and the lexical gap problem to some extent. But even these approaches do not fare well for shorter documents or sentences due to the increased sparseness, which is worsened by the fact that most of the content words do not repeat within a single sentence. On the other hand, word space models (WSM) [157] represent a word as a vector of all its context words. These models rely on the distributional hypothesis that the meaning of a word is determined "by the company it keeps" [51]. WSMs have been shown to be very useful in many tasks such word sense discrimination [159], synonymy identification [176], bilingual dictionary mining [40], and so on.

However, both these models are insufficient to represent the meaning of natural language text [94]. VSMs represent text as bag of words and hence completely ignore the context information. Where as the WSMs are limited to representing words and suffer from sparsity problem when we consider longer word n-grams. Computational models of semantics, such as Montague grammar [129], address the problem of computing the meaning of phrases or sentences. The central problem in these approaches is the compositionality problem, which attempts to compose the meaning of longer word n-grams from its constituents (like words or phrases).

7.1.2 Vector based Compositionality Learning

In the context of VSMs, there has been some work in the compositionality, mainly, for short phrases or specific word combinations such as noun-noun or adjective-noun combinations. In this line of work, the most common composition operator is the averaging operator [176, 140, 159]. The main disadvantage of the averaging operator is its insensitivity to the word order and the syntactic role of words [110]. Mitchell and Lapata [126] and Widdows [190] discuss several other composition operators for bigrams and provide some preliminary experimental results. Tensor product of vectors has also been used as a compositionality operator [162, 141]. But most of these operators are illustrated only on hand picked examples and are not thoroughly tested. Studies that address empirical validation are done on smaller data sets and hence the results are inconclusive. Mitchell and Lapata [126] observe that multiplicative models perform better for verb-noun compositions, Geisbrecht [64] finds tensor product to be working better where as Guevara [67] observes that addition operator performs better at adjective-noun compositions. Recently, Reddy *et al.* [149] evaluate different compositionality operators for noun-noun compositions on a relatively bigger data set. They observed that weighted vector addition [126] performs competitively with other operators. For this reason, apart from the computational advantages, I choose weighted vector addition as the basic composition operator for my models.

Matrix operators have also been used for modeling the composition operation. Baroni and Zamparelli [13] model adjectives and nouns as matrices and vectors respectively. Subsequently, they model the adjective-noun composition as a matrix vector multiplication. Further along this direction, Rudolph and Giesbrecht [154] represent all the words as matrices and use matrix multiplication as the composition operator. The authors argue that, because matrix multiplication is not commutative, it is sensitive to the word order and hence it is preferable over vector opera-

tors. They also show that by appropriately defining the matrices the matrix multiplication can subsume most of the vector composition operators. Despite the compelling arguments in favor of the matrix operators, vector operators are simple, amenable to learning, and are easier to extend to longer word n-grams such as sentences.

All the above approaches discussed till now consider composition at small word n-grams (usually bigrams). Though some of them can be extended to longer units such as a sentence, they do not consider the syntactic structure. Alternative formulations using recursive autoencoders, used in parsing sentences [164] and paraphrase detection [163], compute vector representation of a sentence using its syntactic structure. In these approaches, the authors use binary parse tree of a sentence to compute the vector representation of the sentence based on the representations of its words. Each node of the parse tree is associated with a neural network which takes a $2 \times d$ length vector as input (concatenated vector of both the children) and outputs a d-dimensional vector, which is passed along to its parent. Starting from the leaf nodes, which are words, they compute the vector representation of all the internal nodes and then finally of the root node. They use paraphrase corpus [44] to learn parameters of the neural network. In what follows, I will describe a model to compute the vector representation of a sentence based on its dependency tree structure as well as the word representations. In that aspect, my approach is similar to these approaches but the composition function and the way we optimize is complete different from theirs.

7.1.3 Problem Setting

To reiterate, we assume that the meaning of a word or a sentence can be represented as a point in a d-dimensional vector space. Furthermore, we assume that the vector representations of individual words are given to us by other means (such as distributional semantics [51, 118, 176, 66]). The aim is to compose the meaning of a sentence based on its constituent words. In order to achieve that, I use weighted vector addition as the basic composition operator [176, 140, 159]. For example, the vector representation of the phrase "green apple" is given by $\alpha\, \vec{g}(\text{green}) + \beta\, \vec{g}(\text{apple})$ where α and β are the weights and $\vec{g}(w)$ is a function that returns the vector representation of the word w. Both the basic assumptions made in my model – representability of a sentence/word as vector and vector addition as the composition operator – are arguable and, as discussed earlier (Sec. 7.1.2), alternative representations and operators do exist. But I choose them for their computational simplicity. Notice that although this model seems simple, based on the way the weights are defined it can exhibit diverse modeling abilities. For *e.g.,* the weight can be as simple as a scalar that does *not* depend on the syntactic role of the word to as sophisticated as a linear transformation matrix which encodes the word's syntactic role. In the latter case, the resulting composition function is sensitive to the word order as well.

With the above setup, I will address the compositionality learning problem in the first half of this Chapter. During this process, similar to [163], I use the dependency parse of a given sentence to identify the syntactic role of a word. At a higher level, I design the composition function such that the vector representation of two sentences that express the same meaning (*e.g.,* "I ate an apple." and "I had a fruit.") are close to each other. The parameters of such a composition function can be learnt using paraphrased sentence pairs as training data. But unfortunately, such paraphrased sentences are not widely available in many languages. To overcome this, I propose to use bilingual parallel corpus developed for machine translation as the training data.

In the bilingual scenario, we simultaneously model two composition functions one per each language such that the two vector representations output by the composition functions, for a par-

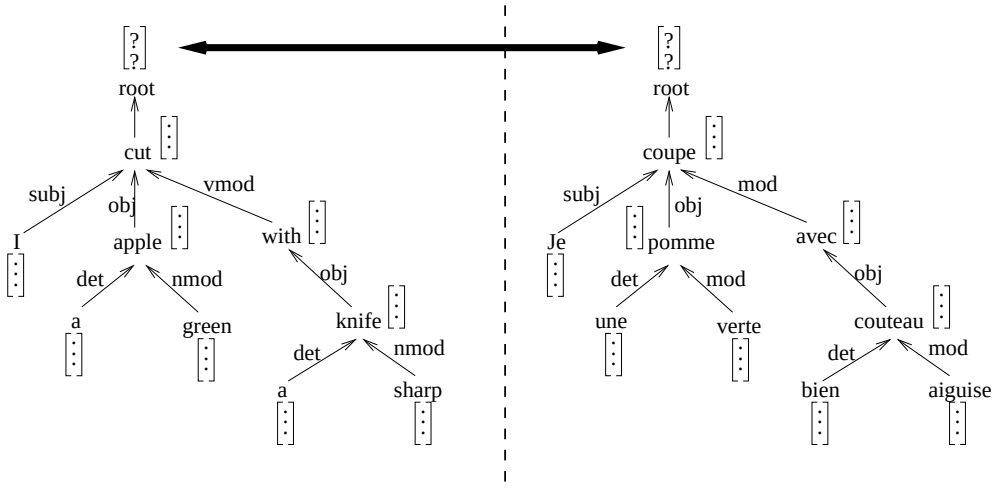


Figure 7.1: Dependency parse trees for English sentence "I cut a green apple with a sharp knife" and its French translation "Je coupé une pomme verte avec un couteau bien aiguisé". Notice that the dependency relations (edge labels) need not be same in different languages.

allel sentence pair, are close to each other. In order to compare the vector representations of sentences in different languages, they should be in the same space. So, we first need to map the vocabulary of both the languages into a common d-dimensional space. In this section, we assume that such representation is available, but it can be obtained from techniques described in Chapter 4 or using multilingual topic models [90, 20]. Next, we independently get the dependency tree representations of the parallel sentences. Fig. 7.1 shows parse trees of an example parallel sentence pair in English and French. It is a coincidence that both the parse trees have the same structure for this example sentence pair but, in general, they can differ in several ways. Please refer to [46] for a detailed description on the translational divergences. Also, notice that the dependency relations in both the languages need not be same. As shown in the Figure, in both the parse trees, all the nodes (words) except the root are associated with a d dimensional vector. Finally, the task is to *compose* the vector representation of the individual nodes (or words) to find the representation for the root node (proxy for the entire sentence). We will simultaneously learn composition functions in both the languages such that the "root vectors" of both the sentences are close to each other.

7.1.4 Multi-view Multivariate Regression (MMR)

Multivariate linear regression [116] generalizes univariate linear regression to multiple variables. In univariate linear regression, input variables ($x_i \; i = 1 \cdots k$) and the response variable (z) are scalars and the relationship between them is modeled in a linear fashion, *i.e.,*

$$z = \sum_{i=1}^{k} \beta_i x_i + \epsilon$$

where β_i is the weight for the i^{th} input variable and ϵ is an error term which is normally distributed. Given n such observations, the weights β_i are usually estimated using linear least squares estimation [74].

In multivariate regression, the input variables ($\mathbf{x}_i \in \mathbb{R}^{d_1} \; i = 1 \cdots k$) and the response variable

$\mathbf{z} \in \mathbb{R}^{d_2}$ are vectors and the relation between them is again modeled using a linear combination, *i.e.*,

$$\mathbf{z} = \sum_{i=1}^{k} W_i \mathbf{x}_i + \boldsymbol{\epsilon} \tag{7.1}$$

where $W_i \in \mathbb{R}^{d_2 \times d_1}$ captures the dependence of the i^{th} input variable on the response variable and $\boldsymbol{\epsilon}$ is a d_2 dimensional error term which is normally distributed. This formulation has $k \times d_2 \times d_1$ number of unknowns. If we assume that the error terms corresponding to each of the d_2 dimensions are uncorrelated, *i.e.*, $\mathbb{E}[\boldsymbol{\epsilon}(i)\boldsymbol{\epsilon}(j)] = 0$ when $i \neq j$, then it can be reduced to a set of d_2 independent univariate linear regression problems. Given n observations, we form a univariate linear regression problem corresponding to each dimension of the response variable. Thus we get d_2 univariate linear regression problems each with $k \times d_1$ unknown variables and n equations.

In this section, I consider a further generalization of multivariate linear regression problem which I refer to as multi-view multivariate regression (MMR). We assume that we have two sets of input variables, $\left(\mathbf{x}_l \in \mathbb{R}^{d_1}\ l = 1 \cdots V^f\right)$ and $\left(\mathbf{y}_m \in \mathbb{R}^{d_2}\ m = 1 \cdots V^e\right)$, and we want to model the correlation between these sets of input variables. To relate to the compositionality learning problem described in Sec. 7.1.3, think of each view as a language, each object as a sentence and each variable as a word in that language. In the MMR setting, we can represent each sentence/document as a collection of words where a word is in turn represented as a multidimensional vector. This representation can be understood as enriching the traditional bag-of-word representation by replacing the lexical identity of a word with richer information about the word itself. A major difference between the MMR setting and that of CCA is that, here, each variable ($\mathbf{x}_l$ or $\mathbf{y}_m$) is multivariate where as variables in CCA are scalars. In fact, MMR problem setting can be converted into a CCA problem, by constructing a single giant multivariate random variable per each view by appending all the individual input variables, *i.e.*, $\mathbf{x} = [\mathbf{x}_1^T \cdots \mathbf{x}_{V^f}^T]^T$. But this poses serious computational challenges as discussed in later sections.

7.1.4.1 Vector based Compositionality Learning as MMR

To remind the notation, I use superscript to indicate language and subscript to indicate the index. Let $\mathbf{x}_l \in \mathbb{R}^d$ and $\mathbf{y}_m \in \mathbb{R}^d$ be the vectors representing l^{th} and m^{th} word in both the languages. Notice that words from both the languages lie in the same d-dimensional space. Such a representation can be obtained using techniques described in Chapter 4 or by multilingual topic modeling techniques [90, 20]. Let w be a word in either of the languages and $\vec{g}(w)$ be a function that returns the vector representation of the word in the interlingual space, *i.e.*, $\vec{g}(w) \in \mathbb{R}^d$. I used CCA as described in Sec. 4.1.2 to map the vocabulary of both the languages into the interlingual representation. Notice that for simplicity I am using the same function $\vec{g}(.)$ to map a word from either of the languages into the interlingual representation. But, based on the word's language it first computes its feature vector and then uses appropriate transformation to map the word into the interlingual representation. Let $\left\{(\mathbf{s}_i^f, \mathbf{s}_i^e) \mid i \in 1 \cdots n\right\}$ represent n parallel sentence pairs in both the languages. Let $\mathcal{T}_i^f$ and $\mathcal{T}_i^e$ represent the dependency tree structure of sentences $\mathbf{s}_i^f$ and $\mathbf{s}_i^e$ respectively.

I model compositionality as a function $\mathcal{G}(\mathcal{T}, \boldsymbol{\Theta}) \in \mathbb{R}^d$, parameterized by $\boldsymbol{\Theta}$, that takes a tree $\mathcal{T}$ as input and outputs a d-dimensional vector. I keep the functional form of the composition function $\left(\mathcal{G}\right)$ same in both the languages, but the parameters $\left(\boldsymbol{\Theta}^f \text{ and } \boldsymbol{\Theta}^e\right)$ vary based on the language. As a result, we will learn different composition functions for both the languages. There

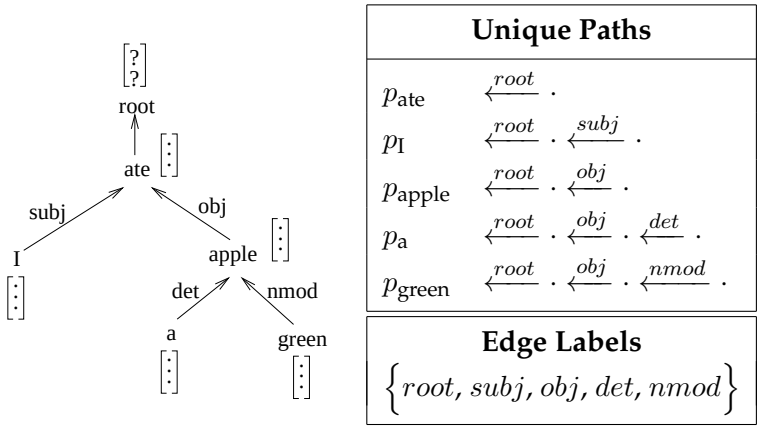


Figure 7.2: Dependency parse trees for English sentence "I ate a green apple". The paths of different words to its root and the set of edge labels are also shown. p_w denotes the path of the word w to the root.

are several possible formulations of the composition function and I will discuss one such function in Sec. 7.1.5. Once the composition function is fixed, the compositionality learning becomes the problem of learning the parameters of the function. As briefed in the previous section, the aim of compositionality learning is to find the parameters $\mathbf{\Theta}^f$ and $\mathbf{\Theta}^e$ such that, for a given parallel sentence pair $(\mathbf{s}_i^f, \mathbf{s}_i^e)$, the squared Euclidean distance between their vector representations in the interlingual space $\|\mathcal{G}(\mathcal{T}_i^f, \mathbf{\Theta}^f) - \mathcal{G}(\mathcal{T}_i^e, \mathbf{\Theta}^e)\|^2$ is as small as possible. This can be expressed as the following optimization problem:

$$\underset{\mathbf{\Theta}^f, \mathbf{\Theta}^e}{\arg\min} \quad \sum_{i=1}^{n} \|\mathcal{G}(\mathcal{T}_i^f; \mathbf{\Theta}^f) - \mathcal{G}(\mathcal{T}_i^e; \mathbf{\Theta}^e)\|^2 \tag{7.2}$$

This is the general form of the objective function and later we show that with a particular choice of the composition function the optimization problem reduces to MMR problem.

7.1.5 Path Parameterized Model

In this Section, for simplicity, I first describe the composition model on a smaller sentence shown in Fig. 7.2. But when I discuss the objective function, in the bilingual context, I use the example sentences shown in Fig. 7.1.

In this model, we consider a labelled path from a node (word) in the tree to the root as a unit. Fig. 7.2 shows an example parse tree and all the paths. I also showed the paths of the example parallel bilingual sentence pair in Table. 7.1. I would like to point out that it is possible for a path to be observed multiple times in a single sentence. For *e.g.,* a ditransitive main verb needs two objects in a sentence so the path from the verb the root, *i.e.,* $\xleftarrow{root} \cdot \xleftarrow{obj} \cdot$ will be observed twice. At first it may appear that the total number of paths can grow exponentially, but I remind the reader that each path is associated with a word token in the sentence. So the total number of paths is bounded by the total number of tokens. But we can expect that the number of unique paths to be much smaller than the total number of tokens because certain paths like $\xleftarrow{root} \cdot \xleftarrow{obj} \cdot$ and $\xleftarrow{root} \cdot \xleftarrow{subj} \cdot$ are very frequent.

English sentence paths	French sentence paths
$\xleftarrow{root}$.	$\xleftarrow{root}$.
$\xleftarrow{root} \cdot \xleftarrow{subj}$.	$\xleftarrow{root} \cdot \xleftarrow{subj}$.
$\xleftarrow{root} \cdot \xleftarrow{obj}$.	$\xleftarrow{root} \cdot \xleftarrow{obj}$.
$\xleftarrow{root} \cdot \xleftarrow{obj} \cdot \xleftarrow{det}$.	$\xleftarrow{root} \cdot \xleftarrow{obj} \cdot \xleftarrow{det}$.
$\xleftarrow{root} \cdot \xleftarrow{obj} \cdot \xleftarrow{nmod}$.	$\xleftarrow{root} \cdot \xleftarrow{obj} \cdot \xleftarrow{mod}$.
$\xleftarrow{root} \cdot \xleftarrow{vmod}$.	$\xleftarrow{root} \cdot \xleftarrow{vmod}$.
$\xleftarrow{root} \cdot \xleftarrow{vmod} \cdot \xleftarrow{obj}$.	$\xleftarrow{root} \cdot \xleftarrow{mod} \cdot \xleftarrow{obj}$.
$\xleftarrow{root} \cdot \xleftarrow{vmod} \cdot \xleftarrow{obj} \cdot \xleftarrow{det\cdot}$	$\xleftarrow{root} \cdot \xleftarrow{mod} \cdot \xleftarrow{obj} \cdot \xleftarrow{det}$.
$\xleftarrow{root} \cdot \xleftarrow{vmod} \cdot \xleftarrow{obj} \cdot \xleftarrow{nmod}$.	$\xleftarrow{root} \cdot \xleftarrow{mod} \cdot \xleftarrow{obj} \cdot \xleftarrow{mod}$.

Table 7.1: All the paths in the dependency parse of the English and French sentences shown in Fig. 7.1.

In this model, each unique path is associated with a weight which specifies the contribution of the vector at that node towards the root. The intuition for considering path as a unit is that, the path captures syntactic role of a word within a sentence and hence we can see it as a proxy for the syntactic role. We want words with different syntactic roles to contribute differently towards the meaning of the sentence. For example, one might expect that the meaning of a sentence to be more closer to the meaning of the main verb, so the path corresponding to the main verb should get relatively higher weight. In what follows, I first describe the compositionality model for the example sentence shown in Fig.7.2 and then formulate the objective function for the bilingual case in the next section.

Let P be the set of unique paths and $|P|$ denote the total number of unique paths. Given a sentence $\mathbf{s}$ and its dependency tree $\mathcal{T}$, let $w \in \mathcal{T}$ be a word in the tree and p_w denote the path of the word w to the root node in the tree, $\big($*e.g.*, p_{green} is $\xleftarrow{root} \cdot \xleftarrow{obj} \cdot \xleftarrow{nmod} \cdot \big)$ then the vector representation of the entire sentence is given by:

$$\mathcal{G}(\mathcal{T}, \boldsymbol{\theta}) = \sum_{w \in \mathcal{T}} \boldsymbol{\theta}(p_w) \vec{g}(w) \tag{7.3}$$

where $\boldsymbol{\theta}$ is a parameter vector of size $|P|$ and $\boldsymbol{\theta}(p_w)$ is the weight associated with the path p_w. To illustrate, under this model, the vector representation of the example sentence shown in Fig. 7.2 is given as follows:

$$\boldsymbol{\theta}(p_{\text{ate}})\vec{g}(\text{ate}) + \boldsymbol{\theta}(p_{\text{I}})\vec{g}(\text{I}) + \boldsymbol{\theta}(p_{\text{apple}})\vec{g}(\text{apple}) + \boldsymbol{\theta}(p_{\text{a}})\vec{g}(\text{a}) + \boldsymbol{\theta}(p_{\text{green}})\vec{g}(\text{green}) \tag{7.4}$$

Since this is linear in terms of $\boldsymbol{\theta}$, the composition function of a sentence can be rewritten as a matrix-vector multiplication as follows:

$$\mathcal{G}(\mathcal{T}, \boldsymbol{\theta}) = X\,\boldsymbol{\theta} \tag{7.5}$$

where the word vectors of the sentence are captured in the matrix X, which is a $d \times |P|$ matrix

with i^{th} column given by the word vector occurring at the path p_i.[3] If a path is not observed in the tree of this sentence then we will set the column to a zero vector. Since only a tiny fraction of the total paths will be observed in a sentence, this matrix tends to be very sparse. Moreover, if multiple words play the same syntactic role with in a sentence then the vectors corresponding to those words are added.

7.1.6 Optimizing Path Parameterized Model

Now, I will use the above composition function and setup the parameter learning as an MMR problem. Let P^f and P^e represent the set of unique paths in both the languages and let $\boldsymbol{\theta}^f$ and $\boldsymbol{\theta}^e$ be the parameter vectors of size $|P^f|$ and $|P^e|$ respectively. Given a parallel sentence pair $(\mathbf{s}_i^f, \mathbf{s}_i^e)$ and their dependency trees $(\mathcal{T}^f, \mathcal{T}^e)$, using the path parameterized model as the composition function, the relation between these two sentences is captured in a linear fashion and the combination weights are estimated using:

$$\underset{\boldsymbol{\theta}^f, \boldsymbol{\theta}^e}{\arg\min} \Big\| \sum_{w_l^f \in \mathcal{T}^f} \boldsymbol{\theta}^f(p_{w_l^f}) \vec{g}(w_l^f) - \sum_{w_m^e \in \mathcal{T}^e} \boldsymbol{\theta}^e(p_{w_m^e}) \vec{g}(w_m^e) \Big\|^2 \tag{7.6}$$

where $p_{w_l^f}$ and $p_{w_l^e}$ are the paths of the words w_l^f and w_m^e in their respective dependency trees. In order to avoid the trivial solution of setting both the parameter vectors to zero vectors, we put l_2-norm constraints on the parameter vectors. Notice that this is an MMR problem, a multi-view generalization of the multivariate linear regression problem.

We will rewrite the objective in terms of matrix-vector products as it immediately admits a solution. As explained in the previous paragraph, for the given parallel sentence pair $(\mathbf{s}_i^f, \mathbf{s}_i^e)$ their vector representations are given by $X_i\, \boldsymbol{\theta}^f$ and $Y_i\, \boldsymbol{\theta}^e$ where X_i and Y_i are matrices capturing the respective sentence word vectors and are of sizes $d \times |P^f|$ and $d \times |P^e|$ respectively. By rewriting the Eq. 7.6 in terms of matrix-vector products and summing over all the n parallel sentence pairs, we arrive at the following optimization problem:

$$\underset{\boldsymbol{\theta}^f, \boldsymbol{\theta}^e}{\arg\min} \sum_{i=1}^{n} \|X_i\boldsymbol{\theta}^f - Y_i\boldsymbol{\theta}^e\|^2 \qquad \text{s.t.} \sum_i \boldsymbol{\theta}^{fT} X_i^T X_i \boldsymbol{\theta}^f = 1 \;\text{ and }\; \sum_i \boldsymbol{\theta}^{eT} Y_i^T Y_i \boldsymbol{\theta}^e = 1 \tag{7.7}$$

With simple algebra this can be rewritten as follows:

$$\underset{\boldsymbol{\theta}^f, \boldsymbol{\theta}^e}{\arg\max}\; \boldsymbol{\theta}^{fT} \Big(\sum_{i=1}^{n} X_i^T Y_i \Big) \boldsymbol{\theta}^e \qquad \text{s.t.}\;\; \boldsymbol{\theta}^{fT} \Big(\sum_i X_i^T X_i \Big) \boldsymbol{\theta}^f = 1 \;\text{ and }\; \boldsymbol{\theta}^{eT} \Big(\sum_i Y_i^T Y_i \Big) \boldsymbol{\theta}^e = 1 \tag{7.8}$$

The objective function in this form is very similar to that of CCA (Sec. 2.2.1), which immediately suggests a solution by solving an eigensystem. By following a procedure very similar to solving CCA, the parameter vectors are obtained by solving the generalized eigensystem:

$$\begin{pmatrix} \mathbf{0} & C_{xy} \\ C_{yx} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}^x \\ \boldsymbol{\theta}^y \end{pmatrix} = \begin{pmatrix} C_{xx} & \mathbf{0} \\ \mathbf{0} & C_{yy} \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}^x \\ \boldsymbol{\theta}^y \end{pmatrix} \tag{7.9}$$

[3] I slightly abuse the notation here. I use $p_{(.)}$ to index a path in the set of all paths P and also as a function that returns the path of a word in a sentence. To make it clear, I use p_i when i is index to represent the i^{th} path in the set P; p_{word} as a function that returns the path of a word in a sentence.

where $C_{xy} = \sum_i X_i^T Y_i$, $C_{xx} = (1-\tau)\sum_i X_i^T X_i + \tau I$ and $C_{yy} = (1-\tau)\sum_i Y_i^T Y_i + \tau I$ are the covariance matrices, τ is a regularization parameter and I is the identity matrix of appropriate size.

7.1.7 Discussion

The generalized eigensystem shown in Eq. 7.9 has multiple solutions, in terms of different eigenvectors, but with different eigenvalues. Each eigenvector is a valid parameter vector and can be used to map an input sentence to a point in the interlingual representation. So, by considering the top k eigenvectors (with high eigenvalue) we can represent the sentence as *a collection* of k points in the interlingual representation. The value of k depends on the task and is determined based on the development data.

Here, I described a compositionality model and one can come up with alternative models which handle the deficiencies of the path parameterized model. For *e.g.,* one disadvantage of this formulation is that the new covariance matrices $\left(\sum_i X_i^T Y_i, \sum_i X_i^T X_i \text{ and } \sum_i Y_i^T Y_i\right)$ are in the order of number of unique paths and can potentially be very big. Another detail is that, it learns weights for only the paths that are seen in the training data. If a test sentence contain paths that are not observed in the training data, then they will get a zero weight and hence doesn't contribute to the sentence meaning at all. This can be addressed in two ways; using back-off paths as is done in language modeling literature or by decomposing the path weight onto its edges.

In my preliminary experiments, I used compositionality learning to address the task of mining a English translation sentence given a French sentence and vice versa. I compared my approach with a CCA based baseline system where each sentence is represented as a bag-of-words. In my experiments, I found out that the compositionality learning approach under-performs the bag-of-word model. In the analysis, I found out that this is due to the poor quality word embeddings. An oracle experiment where I used accurate embeddings yielded significant gains and the resulting model outperformed the bag-of-word model by a huge margin. The major difference between the original word embeddings and the oracle embeddings is that, the latter differentiate different occurrences of a single word. That is, multiple tokens of the same word type are represented differently. So, instead of trying to improve the compositionality model, in the reminder of this Chapter, I discuss how to learn word embeddings at the token level. As we shall see later, this can be cast as a multivariate linear regression problem – a specific case of the MMR problem.

7.2 Learning Token Based Embeddings

Most of the approaches to compute word embeddings compute them for words in isolation [118, 128, 37, 139]. That is, though these approaches consider all the contexts in which a word has appeared to construct its representation, the final representation is a property of the word type and is independent of any given context. In reality, words are polysemous and a word can be used in different senses depending on its context. For example, the word 'shot' means the 'act of shooting' in the sentence "his shot was slow but accurate" but it means 'a small drink' in the sentence "he poured a shot of whiskey". These are only two examples of the many possible senses of the word 'shot'. Computing a single vector representation for 'shot' will not be able to capture its different senses. To better capture the granularity, we may want to compute a unique representation for each of its different senses. Since we do not always know the number of different senses of a word,

we consider a slightly different but relevant problem of learning a context specific representation for words [159, 158, 42]. For clarity, I refer to the context specific word representation as token based embedding/representation and the representation independent of the context as type based embedding/representation.

I address the task of learning token based embeddings in two stages: 1) in the first stage we use an existing distributional semantic method to compute the type based representations and then, 2) adapt the type based representations to each context. In this section, I assume that the type based representations are given to us and primarily focus on the task of learning token based word embeddings from the type based representations. I formulate this task as a multivariate linear regression problem which is a special case of MMR setting that we considered in the previous section.

7.2.1 Potential Applications

But before moving on to the modeling details, I will describe couple of potential applications where token based embeddings may become relevant.

7.2.1.1 Unsupervised POS Tagging

There are two main approaches to unsupervised POS tagging: clustering using latent variable models [55, 65] or clustering based on dimensionality reduction [158, 109]. In [109], authors show that, approaches using dimensionality reduction outperform their generative counter parts. In brief, these approaches use WSMs to represent each word, *i.e.,* each word is represented as a vector of all its context words. Then, they use SVD to reduce the dimensionality followed by k-means algorithm [119] to cluster the words into different groups. Subsequently they align the clusters to POS tags and decide the POS tag of a word based on its cluster label. This line of approach starts with word type vectors and labels the POS tag based on the word type. All the tokens of a single word are assigned the same POS tag. This can be relaxed if we have a context specific representations. We first construct token based embeddings and cluster at token level. Thus we should be able to assign different POS tag to different tokens of same word type.

7.2.1.2 Re-scoring Candidate Translations for MT

SMT systems rely on word/phrase level translation tables to translate sentences from one language into another. A translation table lists possible translations of a word irrespective of the context in which it occurred. For example, Table 7.2 shows a few English translations of the French word 'rapport' along with their translation probabilities. For concreteness, we call them type level translations since they are independent of the word's context. If we have a decent mechanism to learn token based embeddings for a word, we can use it to re-score the type level translations based on the word's context. The re-scored translations are referred to as token level translations. We want to re-score the translations such that, given a French sentence "Il a rédigé un rapport" (whose English translation is "He wrote a report") we score the English translation 'report' higher, where as given the sentence "Quel est le rapport" (translates to "What is the relationship") we assign high score to the translation 'relationship'.

Mining token level translations can interact with SMT in the following ways: 1) to mine translations in the new sense. In [26], it is shown that significant gains can be achieved in domain

French	English	$p(e\|f)$
rapport	report	0.3
rapport	document	0.3
rapport	relationship	0.1
rapport	reporting	0.05

Table 7.2: Type level translations of the French word rapport.

adaptation for MT by first spotting the words that take a new sense in the new domain and then mining translations in the new sense. Once a French word has been spotted as it is used in a new sense [25], we can use our approach to mine the translations in the new sense. 2) In domain adaptation for MT, phrase sense disambiguation (PSD) classifier is used to re-score the translations learnt from an old domain to the new domain [27]. The token-level translation score can be fed as an additional feature to the PSD classifier.

7.2.2 Problem Setting

In the following sections, I describe my approach to adapt type level word embeddings to the token level. Since the aim of this task is to learn a specific representation for each sense of a word we need a corpus labelled with word senses. Unfortunately, such data sets are not readily available in many languages and even in the languages they are available, they are limited in both size and the vocabulary coverage. In order to overcome this, we again fall back to the bilingual parallel sentence pairs. In my approach, I first word-align the parallel sentence pairs and then look at the target language translation of a given source word. The target language translation is treated as a proxy for its sense [21, 151, 43, 50, 85, 124, 112]. At a higher level, all the contexts in which a source word is translated into the same target word are treated to be of a single usage of the source word. For example, Table 7.3 shows two different contexts in which the French word 'rapport' occurred. It also shows the aligned English translation in both the cases. In the first sentence 'rapport' is translated as 'report' and in the second sentence it is translated as 'relationship'. This is an indicator that the usage of the word has changed and that the token embedding of 'rapport' should be different in these two sentences.

I use such word aligned parallel data as training data for the token level adaptation. Notice that the words in the context give an indication of the target translation, *e.g.*, the cue word 'rédigé' in the first sentence is a good indicator that the translation could be 'report'. We refer to the word that we are adapting as *focus* word and the words in its context as *cue* words. We limit the cue words to be a window of words around the focus word. I discuss token level adaptation in the context of French and English, but I do not use any language specific information so the approach should be applicable to other language pairs as well.

Table 7.3: Two different contexts/tokens of the word 'rapport'. Notice that the word translates into different English words depending on the context.

7.2.3 Learning Token Based Embeddings as Multivariate Linear Regression

I remind the notation that, given a word w in either of the languages, $\vec{g}(w) \in \mathbb{R}^d$ is a function that returns the type level interlingual representation of the word. I used CCA as described in Sec. 4.1.2 to map the vocabulary of both the languages into the interlingual representation. Let $\left\{(w_i^f, w_i^e),\ i = 1 \cdots n\right\}$ be n aligned French-English word token pairs derived from the word-aligned parallel data and let $\vec{g}_{tok}(w_i^f)$ be the token adapted embedding of the word w_i^f. I want to highlight that w_i^f and w_i^e are tokens, so we can get context information about them.

Following the arguments mentioned in Sec. 7.1.2, I use weighted linear combination as the composition operator. That is, the token vector of a focus word is assumed to be a weighted linear combination of the cue word type vectors. Formally, given a French word w_i^f its token vector $\vec{g}_{tok}(w_i^f)$ is given as:

$$\vec{g}_{tok}(w_i^f) = \boldsymbol{\beta}(0)\vec{g}(w_i^f) + \sum_{w_j^f \in \mathcal{N}(w_i^f)} \boldsymbol{\beta}(w_j^f)\vec{g}(w_j^f) \tag{7.10}$$

where $\mathcal{N}(w_i^f)$ is the set of cue words that are in the neighbourhood of the focus word w_i^f, $\boldsymbol{\beta}(w_j^f)$ is the contribution of the cue word w_j^f and $\boldsymbol{\beta}(0)$ is a special weight which indicates the contribution of the focus word towards itself. Intuitively, we would expect the type and token vectors to be similar to each other, so we would expect $\boldsymbol{\beta}(0)$ to be higher than the other weights.

We use word aligned parallel data to learn the weight vector $\boldsymbol{\beta}$. For each French token, we know its English translation. So, we want to find the weight vector such that the token vector of the French word is closer to the type vector of the English word[4]. This can be expressed using the following objective function:

$$\arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left|\left|\vec{g}_{tok}(w_i^f) - \vec{g}(w_i^e)\right|\right|^2 \tag{7.11}$$

$$\arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left|\left|\left(\boldsymbol{\beta}(0)\vec{g}(w_i^f) + \sum_{w_j^f \in \mathcal{N}(w_i^f)} \boldsymbol{\beta}(w_j^f)\vec{g}(w_j^f)\right) - \vec{g}(w_i^e)\right|\right|^2 \tag{7.12}$$

Intuitively, we are using the cue words to capture the contextual information. And the target language translation provides additional information about the sense in which the focus word is used in each of the contexts [21, 151, 43, 50, 85, 124, 112]. The idea is that all the contexts in which the focus word translates to the same word should modify the type vector in the same way.

If we consider a window of three words around the focus word, then the token vector of 'rapport' in the above two running examples is given as:

$$\boldsymbol{\beta}(0)\vec{g}(\text{rapport}) + \boldsymbol{\beta}(\text{rédigé})\vec{g}(\text{rédigé}) + \boldsymbol{\beta}(\text{un})\vec{g}(\text{un}) + \boldsymbol{\beta}(\text{a})\vec{g}(\text{a}) \tag{7.13}$$

$$\boldsymbol{\beta}(0)\vec{g}(\text{rapport}) + \boldsymbol{\beta}(\text{est})\vec{g}(\text{est}) + \boldsymbol{\beta}(\text{le})\vec{g}(\text{le}) + \boldsymbol{\beta}(\text{Quel})\vec{g}(\text{Quel}) \tag{7.14}$$

And the weights are learned such that the resulting token vectors are close to the type vectors $\vec{g}(\text{report})$ and $\vec{g}(\text{relationship})$ respectively. Thus the target language aligned words help us differentiate the different senses of the focus word. In the above example, we also considered the stop words but this is an experimental choice and in my experiments I ignore them.

[4]Token and Type rather than Token vectors in both the languages. We use type vectors on the target side because, the context on the target side is usually unknown.

The adaptation model shown in Eq. 7.10 is very similar to the multivariate linear regression (Eq. 7.1) except that dependence between the input variables and the response variable is captured using scalars instead of a transformation matrix. This simplification is done purposefully to make it computationally feasible. If we formulate the objective in terms of the full multivariate regression problem then it involves $V^f \times d \times d$ parameters – V^f being the vocabulary size in French – which is really big and poses computational as well as overfitting problems.

Since the adaptation function (Eq. 7.10) is linear in terms of the weight vector, it can rewritten as a matrix-vector product which will become useful in learning the weight vector. Let Z be a $(d \times V^f)$ matrix which stores the type vectors of all the French words. The l^{th} column of Z stores the type vector of the l^{th} French word w_l^f. Let I_i be a $V^f \times (V^f + 1)$ indicator matrix corresponding to the i^{th} French token. The first column of this matrix indicates the type of focus word and the rest of the columns indicate the cue words around the focus word. In particular,

$$I_i(j,1) = \begin{cases} 1, & \text{if focus word at } i^{th} \text{ token is } w_j^f \\ 0, & \text{otherwise} \end{cases} \tag{7.15}$$

$$I_i(j,j+1) = \text{Frequency of the cue word } w_j^f \text{ in the context of the focus word} \tag{7.16}$$

The rest of the elements of this matrix are set to zero, hence this is highly sparse. This matrix has a special structure as shown in Fig. 7.3 which we will use in improving the efficiency of optimization (Sec. 7.2.4). The first column has only one non-zero value and it is also one, the remaining matrix is a diagonal matrix which stores the frequency of the context words.

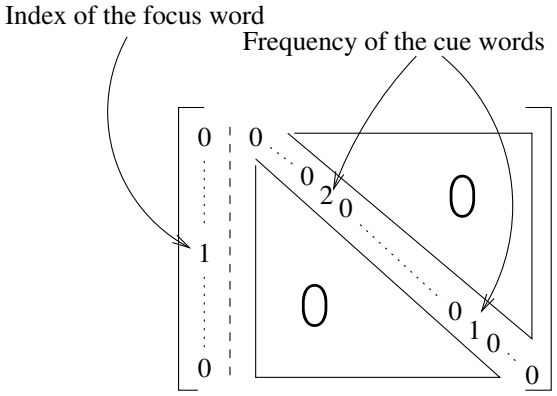


Figure 7.3: Structure of the indicator matrix for the i^{th} token.

With this indicator matrix the adaptation function in Eq. 7.10 can be rewritten as $\vec{g}_{tok}(w_i^f) = Z\, I_i\, \boldsymbol{\beta}$ and the objective function in Eq. 7.12 can be rewritten as follows:

$$\arg\min_{\boldsymbol{\beta}} \sum_i \|Z I_i \boldsymbol{\beta} - \vec{g}(w_i^e)\|^2 \tag{7.17}$$

7.2.3.1 Laplacian Regularization

I add two kinds of regularization on the weight vector to prevent overfitting and to enforce related cue words to get similar weights. Intuitively, given a focus word say 'rapport' we want the cue words 'rédiger', 'rédigé' and 'écrire' (all of them mean 'write') to influence the token vector in

the same way. But there is no term in the objective function that encourages this property. From Eq. 7.10, notice that the contribution of a cue word w_j^f towards the token adapted vector is given by $\boldsymbol{\beta}(w_j^f)\vec{g}(w_j^f)$. To make sure that all the related cue words contribute in a similar fashion, we need to enforce that their type vectors, $\vec{g}(w_j^f)$, are close and their weights are also same. Based on the construction of the mapping function $\vec{g}(.)$, hopefully, it maps all of these points to a closer points in the interlingual representation so their type based embeddings are close to each other. So in order to ensure that they influence the token based embedding in the same way, we need to make sure that the weights for all these cue words are also same. I use Laplacian regularization [29] to achieve this.

$$\tilde{S} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & S & \\ 0 & & & \end{bmatrix} \tag{7.18}$$

Let S be a $V^f \times V^f$ matrix which stores the pairwise similarities of the French word pairs and let $\tilde{S}$ be a $(V^f+1)\times(V^f+1)$ matrix constructed by extending S as shown in Eq. 7.18. This modification is to accommodate for the special weight $\boldsymbol{\beta}(0)$ that we introduced for the focus word itself. Then we add regularization term of the form:

$$\sum_{i,j=1}^{V^f+1} \tilde{S}(i,j)\Big(\boldsymbol{\beta}(i) - \boldsymbol{\beta}(j)\Big)^2 = \boldsymbol{\beta}^T L \boldsymbol{\beta}$$

where L is the Laplacian corresponding to the similarity matrix $\tilde{S}$ and is computed using $L = D - \tilde{S}$ where D is a diagonal matrix with $D(i,i) = \sum_j \tilde{S}(i,j)$. This regularization encourages cue words that are similar to have similar weights. I use $S = ZZ^T$ where Z is a $d \times V^f$ matrix defined in the previous section. Based on our observations from Sec. 4.3, we notice that using a dense similarity matrix S introduces lot of noise and hence we sparsify it by running maximal matching algorithm [98]. I run maximal matching multiple times so that each row is aligned to multiple columns.

After adding the l_2 and Laplacian regularization terms, the final objective function is given by:

$$\arg\min_{\boldsymbol{\beta}} \sum_i \|ZI_i\boldsymbol{\beta} - \vec{g}(w_i^e)\|^2 + \tau \|\boldsymbol{\beta}\|^2 + \lambda\, \boldsymbol{\beta}^T L \boldsymbol{\beta} \tag{7.19}$$

where τ and λ are hyperparameters which decide the weight of the corresponding regularization terms.

7.2.3.2 Discriminative Adaptation

In this section, I use the discriminative ranking idea introduced in Chap. 6 to take advantage of additional information that is available in terms of candidate translations. The above model only uses the target translation of a French word and it ignores the remaining candidate translations. In the running example shown in Table 7.3, for the first sentence the previous model only uses the fact that the word 'rapport' translates to 'report'. But it ignores the fact that there are other candidate translations (such as 'document', 'relationship', etc.) and that 'report' is a better choice

than the remaining candidate translations. In this model, we explicitly use this information. The aim is to learn a weight vector such that the token vector is closer to the correct translation but is farther from the other candidate translations. From the experimental results in Chp. 6, we saw that soft-discriminative model is as good as the full discriminative model and is simpler. So, here we limit to the soft-discriminative treatment and not the fuller version.

Before formulating the new objective function, we slightly rewrite the objective function shown in Eq. 7.19 as follows:

$$\arg\min_{\boldsymbol{\beta}} \sum_i \|ZI_i\boldsymbol{\beta} - \vec{g}(w_i^e)\|^2 + \tau \|\boldsymbol{\beta}\|^2 + \lambda\, \boldsymbol{\beta}^T L\boldsymbol{\beta} \tag{7.20}$$

$$\arg\max_{\boldsymbol{\beta}} \sum_i 2\, \boldsymbol{\beta}^T I_i^T Z^T \vec{g}(w_i^e) - \sum_i \boldsymbol{\beta}^T I_i^T Z^T Z I_i \boldsymbol{\beta} - \tau \|\boldsymbol{\beta}\|^2 - \lambda\, \boldsymbol{\beta}^T L\boldsymbol{\beta} \tag{7.21}$$

We add the discriminative term to the above function such that the token vector moves away from the other candidate translations. The resulting new objective function is given by:

$$\begin{aligned}\arg\max_{\boldsymbol{\beta}} \;\; & (1-\mu)\Big(\sum_i 2\, \boldsymbol{\beta}^T I_i^T Z^T \vec{g}(w_i^e) - \sum_i \boldsymbol{\beta}^T I_i^T Z^T Z I_i \boldsymbol{\beta}\Big) - \tau \|\boldsymbol{\beta}\|^2 - \lambda\, \boldsymbol{\beta}^T L\boldsymbol{\beta} \\ & + \mu \sum_i \sum_{w_j^e \in \text{Trans}(w_i^f)\ \&\ w_i^e \neq w_j^e} \boldsymbol{\beta}^T I_i^T Z^T \Big(\vec{g}(w_i^e) - \vec{g}(w_j^e)\Big)\end{aligned} \tag{7.22}$$

where Trans(w_i^f) returns the English translations of the French word w_i^f and μ is a hyperparameter which indicates the weight of the discriminative constraints with respect to the original objective. In Chp. 6 each of the candidate output sequences has loss value and we used it to weigh the constraint corresponding to that candidate. But unfortunately, the loss of the candidate translations given a French token is unknown so we treat all of them equally likely.

7.2.4 Optimization and Efficient Implementation

Optimizing the objective function in Eq. 7.22 is trivial. Differentiating it with respect to $\boldsymbol{\beta}$ and setting it to zero will result in the following system of linear equations which can be solved very efficiently.

$$\begin{aligned}\Big(\big(1-\mu\big)\sum_i I_i^T Z^T Z I_i + \tau I + \lambda L\Big)\;\; & \boldsymbol{\beta} \\ = \big(1-\mu\big)\sum_i I_i^T Z^T \vec{g}(w_i^e) + \mu & \sum_{i\ ,\ w_j^e \in \text{Trans}(w_i^f)} I_i^T Z^T \Big(\vec{g}(w_i^e) - \vec{g}(w_j^e)\Big)\end{aligned} \tag{7.23}$$

where I is an identity matrix of size $(V^f+1)\times(V^f+1)$. Notice that it is different from the I_i matrix which is an indicator matrix for i^{th} token. This is a system of linear equations of the form $A\boldsymbol{\beta} = \mathbf{b}$ and efficient techniques exist to solve this problem once A and $\mathbf{b}$ are known. Once we solve for the weight vector $\boldsymbol{\beta}$, we use Eq. 7.10 to compute the token embedding of any given token.

The quantities in the above equation can be computed efficiently by exploiting the special structure of the indicator matrix I_i. The reader can skip the rest of this section, if he/she is not interested in the implementation details. Let's first consider the term $\sum_i I_i Z^T Z I_i$. Using the structure of the

I_i matrix as shown in Fig. 7.3,

$$ZI_i = \begin{bmatrix} \vec{g}(w_i^f) & Z \text{ diag}(\mathbf{c}_i^f) \end{bmatrix}$$

where w_i^f is the French word at i^{th} token, $\mathbf{c}_i^f$ is a bag-of-word representation ($V^f \times 1$ vector) of the i^{th} token's context, window of words around it, and diag($\mathbf{c}_i^f$) returns a diagonal matrix of size $V^f \times V^f$ with $\mathbf{c}_i^f$ as its diagonal elements. Using this

$$\sum_i I_i^T Z^T Z I_i = \begin{bmatrix} \sum_i \vec{g}(w_i^f)^T \vec{g}(w_i^f) & \sum_i \vec{g}(w_i^f)^T Z \text{ diag}(\mathbf{c}_i^f) \\ \sum_i \text{diag}(\mathbf{c}_i^f) Z^T \vec{g}(w_i^f) & \sum_i \text{diag}(\mathbf{c}_i^f)\ Z^T Z \text{ diag}(\mathbf{c}_i^f) \end{bmatrix} \tag{7.24}$$

It turns out that all the four quantities in the above matrix can be efficiently computed using the covariance matrix $C_{ff} = ZZ^T$. The top-left corner term is nothing but $\sum_{l=1}^{V^f} \#(w_l^f) C_{ff}(l,l)$ where $\#(w_l^f)$ is the number of tokens of the word type w_l^f observed in the training data. Let F be a $V^f \times V^f$ matrix such that $F(l,m)$ denotes the total number of times the word w_m^f appeared in the context of any of the tokens of the word w_l^f. Notice that this matrix is not symmetric. Then the top-right corner term is given by the column-wise sum of the matrix $F \otimes C_{ff}$ where $\otimes$ is the element-wise matrix multiplication or the Hadamard product. Bottom-left term is the transpose of the top-right hand corner term. Finally, let C be the matrix storing the context of all the tokens, *i.e.*, C is a $V^f \times n$ matrix with i^{th} column denoting the context of the i^{th} token, $\mathbf{c}_i^f$. Then the bottom-right hand corner term is given by $\left(CC^T\right) \otimes C_{ff}$

The computation of the terms in the right hand side of Eq. 7.23 is very similar to the procedure for computing the top-left and bottom-left hand corner terms, except we use cross-covariance matrix instead of the auto-covariance matrix.

On a final note, softwares such as Matlab has very efficient and parallelized implementations of matrix-matrix multiplications. So implementing it as described above saves significant amount of computing time. Moreover, when computing the top-right hand corner term, notice that the matrix F is very sparse, usually the number of non-zero elements are of the order $O(n)$ where n is the total number of tokens, so it is wiser to find the sparse structure of this matrix first and multiply only the non-zero elements instead of naive element-wise matrix multiplication. In doing so, you can usually finish the operation in $O(n)$ time instead of $O({V^f}^2)$ time complexity.

7.2.5 Co-regularization

The model described in the previous section uses only one single parameter vector for all the focus words. It inherently assumes that the contribution of a cue word towards adapting a focus word does not depend on the identity of the focus word. This is too restrictive. The presence of the cue word 'rédigé' may be a good indicator to disambiguate the sense of the focus word 'rapport'. But it may not be an equally good indicator for a different focus word such as 'premier' (whose possible translations are 'prime minister' and 'first day'). One obvious way to relax this assumption is to use one parameter vector for each focus word type. This modification is very easy to incorporate in to the model. Under this relaxation, the objective function defined in Eq. 7.22 decomposes over the individual focus word types. Given a French word type w^f, the $\boldsymbol{\beta}$ in Eq. 7.22 is replaced with $\boldsymbol{\beta}_{w^f}$ and we only sum over the tokens of that word. I refer to this model as 'Local' model in the reminder of this Chapter.

Unfortunately, this decomposition over the word types does not use evidence from other focus words. For example, if a cue word co-occurs with many different focus words then it may not be a good indicator. A more elegant way to incorporate this information is to use the co-regularization idea proposed in Chp. 5. Each focus word (w^f) has its own weight vector $\boldsymbol{\beta}_{w^f}$ but all these weight vectors are co-regularized as follows:

$$\boldsymbol{\beta}_{w^f} = \boldsymbol{\beta} + \mathbf{r}_{w^f} \tag{7.25}$$

That is, there is a common weight vector $\boldsymbol{\beta}$ – akin to the prior distribution in Bayesian models – and each focus word has a residual weight vector $\mathbf{r}_{w^f}$. We explicitly try to minimize the residual vectors as much as possible. That is, a cue word is allowed to take a non-zero weight in the focus word's residual weight vector ($\mathbf{r}_{w^f}$) only if the common weight vector ($\boldsymbol{\beta}$) fails to explain it. Under this model, the new objective function is modified as follows:

$$\begin{aligned}\underset{\boldsymbol{\beta},\mathbf{r}_{w^{(.)}}}{\arg\max}\ \ & (1-\mu)\Big(\sum_i 2\,\boldsymbol{\beta}^T_{w_i^f} I_i^T Z^T \vec{g}(w_i^e) - \sum_i \boldsymbol{\beta}^T_{w_i^f} I_i^T Z^T Z I_i \boldsymbol{\beta}_{w_i^f}\Big) - \tau\,\|\boldsymbol{\beta}_{w_i^f}\|^2 - \lambda\,\boldsymbol{\beta}^T_{w_i^f} L \boldsymbol{\beta}_{w_i^f} \\ & + \mu \sum_i \sum_{w_j^e \in \text{Trans}(w_i^f)\ \&\ w_i^e \neq w_j^e} \boldsymbol{\beta}^T_{w_i^f} I_i^T Z^T \Big(\vec{g}(w_i^e) - \vec{g}(w_j^e)\Big) + \gamma \sum_{w^f} \|\mathbf{r}_{w^f}\|^2 \end{aligned} \tag{7.26}$$

The only modifications are, the replacement of $\boldsymbol{\beta}$ with $\boldsymbol{\beta}_{w_i^f} = \boldsymbol{\beta} + \mathbf{r}_{w^f}$ and addition of regularization term for the residual vectors $\sum_{w^f} \|\mathbf{r}_{w^f}\|^2$. The optimization is over both $\boldsymbol{\beta}$ and $\mathbf{r}_{w^{(.)}}$.

The above objective function can be solved by following a procedure very similar to the one outlined in Chp. 5. As we saw previously, the solution first involves solving for the common weight vector $\boldsymbol{\beta}$ and then the residual vectors $\mathbf{r}_{w^{(.)}}$. Unfortunately, solving for the common vector involves computing an inverse matrix for each word type, which is computationally very intensive given the vocabulary size. So, in the experiments I present experimental evidence only for the local model, but I outlined the co-regularized model for completeness.

7.2.6 Data Sets and Experimental Setup

In this section, I present experimental evidence on the task of re-scoring English translation candidates of a French token. I present experimental results on corpora from three different genre *i.e.,* Scientific (Science), Medical (EMEA), and Movie subtitles (Subs) [177]. In each of the domains, I use parallel data to generate French tokens for training and testing the models. Recall that the models use window of words around the source token to learn the token based embedding. So, in order to filter out tokens that do not have enough contextual evidence, I filter out sentences that are shorter than four and longer than 40. Some of these corpora contain a considerable fraction of repeated sentences. These sentences can skew the context statistics, so I also remove the duplicate sentence pairs. I word align the remaining parallel sentence pairs using the word aligner from the Portage machine translation system [111]. I word-align in both directions and intersect the alignments.

In each of these corpora, I first extract the most frequent 20K words (excluding the stop words) and learn interlingual representations for these words using CCA (Sec. 4.1.2). The bilingual dictionary required to construct the training data for learning interlingual representations is also constructed from the same parallel sentence pairs. Then, I go through the each of the corpora again and extract all the French and English word pairs that are aligned at least 10 times. Hence

Domain	# sent. pairs	# Fr. word types	# (Fr.,En.) word-pairs	# Tokens
Science	107 K	730	1831	127 K
EMEA	82.4 K	553	1551	94 K
Subs	7.55 M	4014	13512	1.4 M

Table 7.4: Data statistics in each of the three different corpora.

the French words that we consider are ambiguous. Of these tokens, I retain only those French tokens whose context (window of three words onto its left and right) contains at least one word from the list of 20K frequent words. Table 7.4 shows the data statistics after the preprocessing.

For each French token, its aligned English word is treated as the reference translation and the rest of its translations are considered as candidate translations. And the task is, given the French token, to re-score all the candidate translation such that the reference translation is at the top. Notice that the reference translations, which are used to compute the accuracies, are also generated automatically and hence likely to contain errors. We measure the effectiveness of reranked translations using accuracy of the top-ranked translation and the mean reciprocal rank (MRR) of the correct translation.

We use the following three step process to re-score the candidate translations: 1) first we learn the type based embeddings for both French and English words, 2) learn the token based embeddings for the French tokens using the method described above, and then 3) compute the distance between the French token and English type embeddings and use it to rerank the English translations. I compare my approach with two other baseline systems. The first approach ('Max. Probable') ranks the candidate translations based on the lexical probability $p(w_m^e|w_l^f)$ learnt on the parallel data from that domain. Second baseline is the phrase-sense disambiguation classifier ('PSD. Classifier') [27]. Given a French token, we train a classifier to predict the most likely translation depending on the context. We use the same set of features as in Carpuat and Wu [27]. For all these approaches, I run 10 fold cross-validation on all the three domain data sets and report average micro and macro evaluation measures[5]. In each split, for each word type we split 80%, 10% and 10% of the tokens into training, validation and test splits respectively. We use the validation data to do a grid search for the optimal hyperparameter setting of τ, μ and λ for our approach and the l_1 and l_2 regularization parameters for the 'PSD. Classifier'. We use VW classifier[6] as the classifier for 'PSD. Classifier'.

7.2.7 Description of Results

In this section, I present experimental results comparing the efficacy of token based embeddings with the above mentioned baseline systems. Table 7.5 shows the accuracy of the top-ranked translation and Table 7.6 shows the MRR values. Overall, the observed trend is same under both the evaluation measures, so I resort my discussion to mainly about the accuracy of the top ranked

[5]Macro evaluation measure is average of averages, we first compute accuracy per each word and then average the word accuracies. Where as for micro evaluation measure is simply the ratio of total number of correct tokens to the total number of tokens. Macro evaluation is dominated by the accuracy on the less frequent words where as micro evaluation gets dominated by the most frequent words.

[6]We use `http://hunch.net/~vw/` version 7.1.2, and run it with the following arguments that affect learning behavior: `--exactadaptivenorm --power` t 0.5

Method	Science		EMEA		Subs	
	Micro	**Macro**	**Micro**	**Macro**	**Micro**	**Macro**
Max. Probable	77.97	69.26	67.07	62.73	61.24	65.75
PSD. Classifier	78.48	71.90	**83.55**	**82.70**	60.21	64.61
Type Embed.	75.98	68.22	67.10	65.18	62.23	63.67
Avg. Embed.	71.76	73.19	62.29	72.8	57.55	66.22
Token Embed.	79.64	74.35	72.23	73.92	61.93	66.78
Token Embed. (Local)	**82.2**	**78.91**	79.7	82.48	**65.17**	**70.72**

Table 7.5: Accuracy of the top-ranked translation.

translation but the observations equally hold to MRR measure as well. Overall, the accuracies are higher in Science domain followed by the EMEA and the Subs domains.

The second block of the Table 7.5 shows the results of the baseline systems. The 'Max. Probable' baseline achieves decent accuracies on all the three data sets. Recall that we used the lexical translation probability table learnt from the same domain parallel data. This baseline achieves better scores in this setting compared to using a translation probability table learnt from a different domain, but, with bigger data (such as parliamentary proceedings). The next baseline system is the 'PSD. Classifier'. It performed better than 'Max. Probable' baseline in Science and EMEA data sets but performs slightly lesser on the Subs data. Moreover, the improvements are prominent on the EMEA data set. This seems in accordance with the number of parallel sentence pairs in each of the data sets. EMEA data set has the least number of parallel sentence pairs and hence the word-alignments learnt from this domain are likely noisy. In such cases, using additional information derived from the French token such as its context and the POS tag seems to be helpful. At this point, I want to highlight that, as mentioned earlier, the gold truth data is also generated automatically and hence likely to contain errors. So it is possible that the accuracy numbers on EMEA data, which has less number of parallel sentence pairs, are erroneous.

The third and fourth blocks of the Table 7.5 show performance of different systems which use interlingual representation. Before discussing the performance on the three data sets, I first present the efficacy of the type embeddings learnt by the CCA on these data sets. As explained earlier, we use the same lexical probability table used by 'Max. Probable' baseline to learn the type embeddings for the French and English words. To estimate the quality of these embeddings, for each French word I compute distance between its type embedding and the embeddings of all the English words[7]. I choose the closest English word and check if it is indeed a valid translation by checking against the list of candidate translations for the French word. If the type embeddings are of high quality, then the closest English word returned must be a valid translation. Under this task, the type embeddings achieve 94.07%, 89.88% and 74% accuracy on Science, EMEA and Subs data sets respectively. This shows that for majority of the words, the closest candidate returned by the type embeddings is indeed a valid translation.

Now, we move on to the discussion about the performance of the adapted token embeddings on the task of re-scoring translations. I present experimental results with multiple variants in the third block of Table 7.5. The first row within this column ('Type Embed.'), presents the results

[7]Notice that the comparison here is between the type embeddings of both language words. There is no adaptation here.

Method	Science		EMEA		Subs	
	Micro	**Macro**	**Micro**	**Macro**	**Micro**	**Macro**
Max. Probable	87.71	83.61	80.61	79.63	74.52	80.32
PSD. Classifier	88.08	85.05	**90.66**	**90.79**	74.26	79.74
Type Embed.	86.67	83.12	80.17	80.82	75.83	79.17
Avg. Embed.	84.22	85.52	77.32	84.81	72.74	80.26
Token Embed.	88.68	86.32	83.08	85.55	75.26	80.59
Token Embed. (Local)	**90.01**	**88.71**	87.82	90.38	**77.69**	**83.00**

Table 7.6: MRR of the reference translation.

without any token level adaptation, *i.e.,* we simply use the word's type embedding as its token embedding. It is encouraging to see that the accuracies of this variant are comparable to the 'Max. Probable' baseline. It is justifiable given that the type embeddings are learnt from the same lexical translation probability table used by the baseline system as well. The next variant ('Avg. Embed.') simply averages the type embeddings of the focus word and the cue words. This is a special case of the token adaptation where we use equal weight for the focus and the cue words. This variant is referred as second order context vectors in the word sense discrimination literature [159, 143]. Surprisingly, averaging seems to perform better than 'Type Embed.' in macro evaluation while it performs poorly under micro evaluation measures. Macro accuracy, being the average of averages, is dominated by the less frequent words. So, this indicates that a simple averaging is better for less frequent words but it hurts on the most frequent words.

The next variant ('Token Embed.') uses the token adaptation as described in Sec. 7.2.4. This model uses a single global parameter vector β for all the focus words. Comparing the performance of this model with the 'Type Embed.' variant, it is clear that adapting the type embeddings to the token level is useful. Except on EMEA data set, this model performs competitively with the 'PSD. Classifier'. In the final variant, 'Token Embed. (Local)', we learn a parameter vector for each focus word as described in the beginning of Sec. 7.2.5. This variant has more degrees of freedom to fit to the data and it outperforms previous variants. Except, on EMEA data set this variant outperforms all the baselines as well as all the previous variants. On EMEA data set, 'PSD. Classifier' still wins. As specified earlier, the trend of different systems is same under the MRR evaluation measure as well. Though the local token adaptation based model wins over the global model, it can not handle word types that are not seen during the training. But the global model can handle the unseen word types as well. The co-regularized model (Sec. 7.2.5), being a hybrid of local and global models can potentially be advantageous over both the models. But it needs to be verified.

7.3 Discussion

In this chapter, I discussed models to compute the vector representation of an object based on either its structure or its context. I used these models to address two problems: vector based compositionality learning and learning token based embeddings from the type based embeddings. For both these tasks, the required training data is not available with in a language. Hence, I discussed a novel use of parallel data developed for MT to learn the parameters of the composition functions. Though these two problems look different, at a superficial level, the underlying prob-

lem being solved in both the tasks is same, *i.e.,* composing a set of individual vectors into a single vector. Moreover, as observed in my preliminary experiments, the former problem depends on the latter as it requires token specific word embeddings. I presented experimental evidence only on the task of learning token based embeddings but, in future, I would like to evaluate if the compositionality learning can be improved by the token based embeddings.

The composition functions that I presented here use weighted linear combination of the vectors. This is a very basic model and extensions of it are definitely possible. For *e.g.,* in the task of learning the token embeddings, weighted linear combination implies that the token representations can be obtained by shifting the type embeddings. While this is a reasonable assumption to handle polysemous words (words that take related senses) but it is inadequate for homonyms – words that have same surface form but refer to complete difference senses *e.g.,* 'bank' has 'financial institution' and 'river bank' sense. A simple linear shifting of the type embedding may not be sufficient to handle homonyms. In future, I would like to extend the composition functions by associating a transformation matrix with each syntactic role/cue word rather than a scalar weight.

Moreover, since the weighted linear combination operator is insensitive to the word order, our adaptation model does not take advantage of the order. But, the word order can be incorporated into our model, by adding the position information into the context representation of the focus word. Even without using the position information, our model is still able to perform better than the 'PSD. Classifier' baseline on two data sets. This shows the effectiveness of using word embeddings. The information learnt by our approach can easily be fed into the PSD classifier as an additional feature.

The problem of learning token based embeddings is especially interesting from a modeling perspective. It provides an opportunity to incorporate the ideas that are independently explored in the previous chapters on a single problem bringing my dissertation to a natural conclusion.

Part III

Conclusion and Future Work

Chapter 8

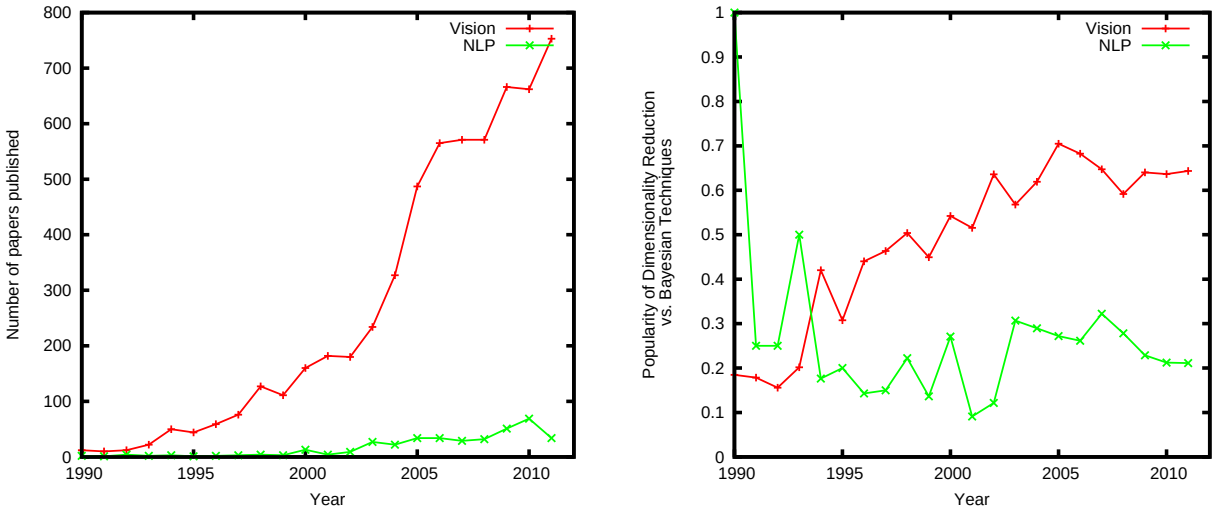
Conclusion and Future Research

To summarize my contributions, in this dissertation, first I demonstrated the effectiveness of interlingual representation by mining word translations for out-of-vocabulary words and incorporating them into an MT system. Subsequently, I identified that interlingual representations are relevant in several other NLP applications. Afterwards, I showed how to incorporate the well known ideas of co-regularization and max-margin learning to learn better interlingual representations. Finally, I discussed how to enrich the existing text representations such as bag-of-word models by incorporating the word meanings into the representation. In this Chapter, I conclude based on the insights I gained in modeling various NLP problems in terms of dimensionality reduction and also discuss some future research directions.

8.1 Conclusion

The idea of dimensionality reduction has been around for a long time. It has been explored in multiple areas of computer science such as computer vision, information retrieval, NLP *etc.*. Fig. 8.1 shows the popularity of dimensionality reduction in computer vision vs. NLP. In both the sub Figures, the x-axis plots the year. Figure 8.1(a) shows the number of papers published in vision and NLP conferences per year. These numbers are obtained using Google scholar by issuing the query 'PCA OR SVD OR "dimensionality reduction" OR CCA'. The number of papers published in vision community is obtained by restricting the search query to conferences that have "vision" in their name, where as the number of NLP papers is obtained by restricting the query to conferences with "computational linguistics" or "natural language processing" in their name. Before going into the discussion of this Figure, I want to emphasize that these numbers are only approximate, but they are sufficient illustrate the trend.

From the Fig. 8.1(a), we can see that dimensionality reduction is definitely more popular in vision than in NLP. This could be because, in computer vision, pixels are represented using real valued numbers and most of the image processing operators are defined as matrix operations. So, it is natural to think in terms of operations from linear algebra. Moreover, the number of vision conferences and the total number of papers published in these conferences is significantly higher that that of NLP. In order to remove this bias, I plot the popularity of dimensionality reduction compared to Bayesian approaches within these areas in the next Figure. I use the query 'Bayesian OR "expectation maximization" OR posterior' to estimate the number of papers that use Bayesian approaches. Here, the x-axis denotes the year and the y-axis denotes the ratio of the number of papers that use dimensionality reduction to the number papers that use Bayesian techniques. A value of 1 on the y-axis indicates that there are equal number of papers using each of the tech-



(a) Number of papers published using dimensionality reduction in computer vision and NLP.

(b) Popularity of dimensionality reduction compared to expectation maximization in computer vision and NLP.

Figure 8.1: Popularity of SVD in Computer Vision and Natural Language Processing. Please refer to the paragraphs for description of how these values are computed.

niques and a higher value indicates that more number of papers use dimensionality reduction. As shown in Fig. 8.1(b), dimensionality reduction is not as popular in NLP as it is in vision.

The high popularity of Bayesian approaches in NLP is probably due to the discrete nature of words. In computational linguistics, we usually represent a word using its lexical identity. This representation makes it a discrete variable. If a word tends to be in multiple states (a state can be a syntactic role such as a POS tag or semantic meaning such as its sense) then we usually model the phenomenon using latent variables. Even in the case where a word is allowed to be in multiple states, the data that we observe contain the word in one of the states and there is no clear interpretation, yet, for a continuous transition between these states. Discrete latent variable models seem a natural fit for such situations. Having said that, the work that I presented in Sec. 7.2 addresses the problem of allowing a word to be in multiple states, by learning token specific representations. Though it is not complete, it appears to be a promising step along the direction of enabling dimensionality reduction based approaches to incorporate this supposition.

Despite the past trend, I have high hopes that dimensionality reduction techniques will gain popularity in NLP. Because, traditionally, it has been used primarily in an unsupervised setting to reduce the size of the feature space and to remove feature correlations. While it served its purpose in reducing the feature space, thus enabling faster computation, the reduction step is not tied to the task at hand. Hence, the learnt low-dimensional subspaces are usually not optimal for the task. The recent advances in multi-view dimensionality reduction enable us to learn low-dimensional representations that are effective for a given task. The work that I presented in this dissertation provides an extensive evidence for this hypothesis. Though, in some cases, the models relying solely on the low-dimensional embeddings are unable to beat the state-of-the-art systems, as observed in the reranking for POS tagging in Chp. 6, they still provide invaluable signal to the state-of-the-art systems. Incorporating this signal into the existing systems yields significant gains. Work by other researchers also confirms this. In [179], the authors show that features derived from low-dimensional word embeddings can be used to improve the accuracy of named entity tagging

and chunking.

Aligning objects from different views is at the core of many NLP problems. If you can formulate your problem as a mapping problem between two objects, then I highly encourage you to try the multi-view dimensionality reduction based techniques. Here I give another example task where I was pleasantly surprised by the value addition brought by the low-dimensional embeddings. Consider sentence paraphrase identification task. Given a pair of sentences within a language, the task is to identify if they are paraphrases of each other or not. As a training data, we are provided with sentence pairs along with their labels (paraphrased or not). This task fits right into the framework of mapping problem, *i.e.,* given a sentence find its paraphrased sentence. I used a simple modification of CCA, to allow the label information into the model, to compute the low-dimensional representation such that paraphrased sentences lie closer in the reduced space. Subsequently, given a test sentence pair, I project both of them into the sub-space and compute the distance between their projections. If the distance is less than a threshold, I hypothesize that they are paraphrases otherwise not. I experimented with the Microsot Research paraphrase corpus [44]. The data is split into 3001, 1075 and 1725 sentence pairs for training, validation and test purposes respectively. By selecting the threshold based on the validation data set, this approach is able to get approximately 72% accuracy. This is less than the state-of-the-art performance. A classifier trained for this task using POS tag, dependency tree based features and various other features achieves 75.42% accuracy [187, 39]. But by adding the distance in the reduced space as an additional feature I was able to improve it 76.36% accuracy. Though this is still less than the state-of-the-art system (76.8% [163]), the value added by this feature over a system that uses carefully hand constructed features is surprising.

Another salient aspect of the dimensionality reduction based approaches is that we can solve for the global optimum. On the other hand, most of the latent variable models that we use in NLP are solved using expectation maximization or Gibbs sampling which are prone to local optima. Recent developments in spectral methods for latent variables [81] show that even in the latent variable models the global optimum can be found using SVD. The core functionality needed by the dimensionality reduction based techniques is matrix-matrix, matrix-vector multiplications, eigen decomposition, inverse computation *etc.*. These problems have been studied for a long time and tools are readily available in most of the programming languages. In most cases, these tools are highly optimized for both efficiency and numerical stability. So, it is very easy to try and implement these techniques. Overall, based on my insights I strongly feel that dimensionality reduction has unexplored merits for NLP tasks.

8.2 Future Research

I conclude this dissertation by pointing some future research directions. I see two major future directions that can benefit from interlingual representations: integrating word embeddings into the generative models and exploring the use of interlingual representations for machine translation.

8.2.1 Integrating Word Embeddings into Generative Models

As observed through out this dissertation, and especially in Chp. 4, the low-dimensional vector representations provide an effective way to compute the similarity between words. More impor-

tantly, once the mapping functions to the interlingual representation are learnt, we only need monolingual corpora to learn the low-dimensional embedding of an unseen word. The abundant availability of the monolingual copora makes it possible to learn word representations for many words. Now we use both the above observations to address one of the crucial and recurring problems in NLP, smoothing to handle unseen words.

The generative models being used in NLP, usually, contain two kinds of parameters: non-lexical and lexical probabilities. The former are of the form $p(z_i|z_j)$, the probability of a latent variable given another latent variable, and the latter are of the form $p(w|z_i)$, the probability of a word (lexical entity) given a latent variable. The lexical probabilities are hard to estimate for the rare and unknown words because there is not enough evidence for them in the training data. So, most of the NLP applications require some smoothing to handle these unknown words.

Word embeddings provide an elegant way of handling the rare words. The idea is, when we encounter a rare word, we compute its word embedding (which we can do because of abundant monolingual data) and use it to find a frequent word that we can back-off to. This can be introduced into the generative models by modeling the lexical probabilities as a function of the word embeddings. For example:

$$p(w|z_i) \propto \exp\left(\lambda(z_i)^T \vec{g}(w)\right)$$

where $\vec{g}(w)$ returns the low-dimensional embedding of the word w and $\lambda(z_i)$ is the weight vector associated with the latent variable z_i. Since we only need monolingual corpora to obtain the low-dimensional embedding it can handle words that are rare and unknown with respect to the training data. Moreover, this function automatically handles smoothing because two words that are related are hopefully closer in the reduced space and hence it assigns roughly same probability to both these words. In this way, low-dimensional word embeddings provide a cleaner way to handle the unknown words.

This direction becomes even more promising for multilingual NLP applications. Because, the past research in multilingual modeling only shares non-lexical probabilities across languages [34]. But if we model lexical probabilities as shown above then they can also be shared across languages. The only missing link is the representation of words from both the languages into a common space which we can do using an interlingual representation.

8.2.2 Interlingual Representations and MT

Machine translation is a natural and a potentially important application of interlingual representation. I want to highlight that the interlingual representation that I introduced in this dissertation are completely different from the traditional interlingua, where we believe that there are "deep structures" that underly the surface realizations in different languages. The connection between such interlingua and MT has already been studied [45, 84, 83, 47]. Instead, here, I discuss about the potential benefits of dimensionality reduction based interlingual representation for MT.

The low-dimensional reranking models that I discussed in Chp. 6 are especially interesting in the context of machine translation. As I discussed in that Chapter, and shown in Fig. 8.2, the reranking models inherently consider a giant feature vector of all possible feature pairs, from the input and output space, and learns a weight vector in this outer product space. One way to use this for MT is to rerank the n-best translations output by the decoder. In this setting, the input will be a source language sentence and the output is the target language translation. If we represent each sentence as a bag-of-words, then the rerankers inherently considers all possible source, target

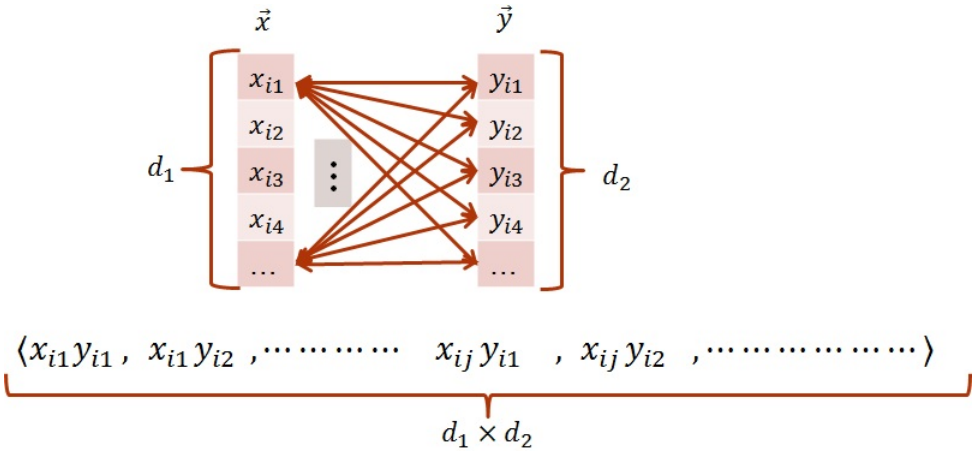


Figure 8.2: Feature space of the Low-dimensional discriminative rerankers.

language word pairs during reranking. In doing so, the approach considers long range word dependencies that are hard to model during decoding. But, the challenge is the scalability of the rerankers. When we represent a sentence as a bag-of-words, the size of the feature space in each language is really high. As we saw in Chp. 2, computing interlingual representation is an $O(n^3)$ operation where n, in this case, is the size of the vocabulary. This is computationally infeasible in the naive implementation. In order to overcome this, we need to explore online and randomized techniques to solving SVD. Fortunately, the research in randomized SVD techniques has matured and some tools are readily available [70]. In future, I would like to explore the scalability of discriminative reranking models using randomized SVD and test its effectiveness on the task of reranking n-best translations for machine translation.

Other components of my research that are related to MT are the vector based compositionality learning and mining token level translations. In Chp. 7, I presented only an intrinsic evaluation of the token based embeddings. In future, I would like to work on how to incorporate the rescored translations into the MT system and test their effectiveness on the sentence level translations. Moreover, I would also like to use token based representations in the vector based compositionality learning. An interesting direction, that has not been explored at all, is to study if the composed sentence level vectors can help MT.

Appendix A

Selection Strategies

After computing association measure for all word pairs, the next step in sparsification is the selection of word pairs to retain. In this section, I describe some approaches such as thresholding and matching for the word pair selection. Again, I describe them in the context of cross-covariance matrix but all of them extend to with in language covariance matrices.

A.1 Thresholding

A straight forward way to remove the noisy word co-occurrences is to zero out the entries of the cross-covariance matrix that are lower than a threshold. To understand the motivation, consider the rewritten objective function of CCA, $\mathbf{a}^T XY^T \mathbf{b} = \sum_{ij} C_{xy}(i,j)\mathbf{a}(i)\mathbf{b}(j)$. This is linear in terms of the individual components of the cross-covariance matrix. So, if we want to remove some of the entries of the covariance matrix with minimal change in the value of the objective function, then the optimal choice is to sort the entries of the covariance matrix and filter out the less confident word pairs.

A.2 Relative Thresholding

While the thresholding strategy described in the above section is very simple, it is often biased by the frequent words. Since a frequent word co-occurs with other words often, it naturally tends to have high association with most of the other words. As a result, absolute thresholding tends to remove all the less frequent word pairs while leaving the co-occurrences of the frequent words untouched. Eventually, this may lead to zeroing out some of the rows or the columns of the cross-covariance matrix. To circumvent this, we try thresholding at word level. For every English word, we choose a few Spanish words that have high association and vice versa. every Spanish word we choose a few Since the nearest neighbour property is asymmetric, we take the union of all the selected word pairs. That is, we retain a word pair, if either the Spanish word is in the top ranked list of the English word or vice versa.

A.3 Experimental Results

In the experiment, we combine the three association measures, Covariance (Cov), MI and Yule's ω, with the three selection criteria, Threshold, Relative Threshold (RelThreshold) and Matching

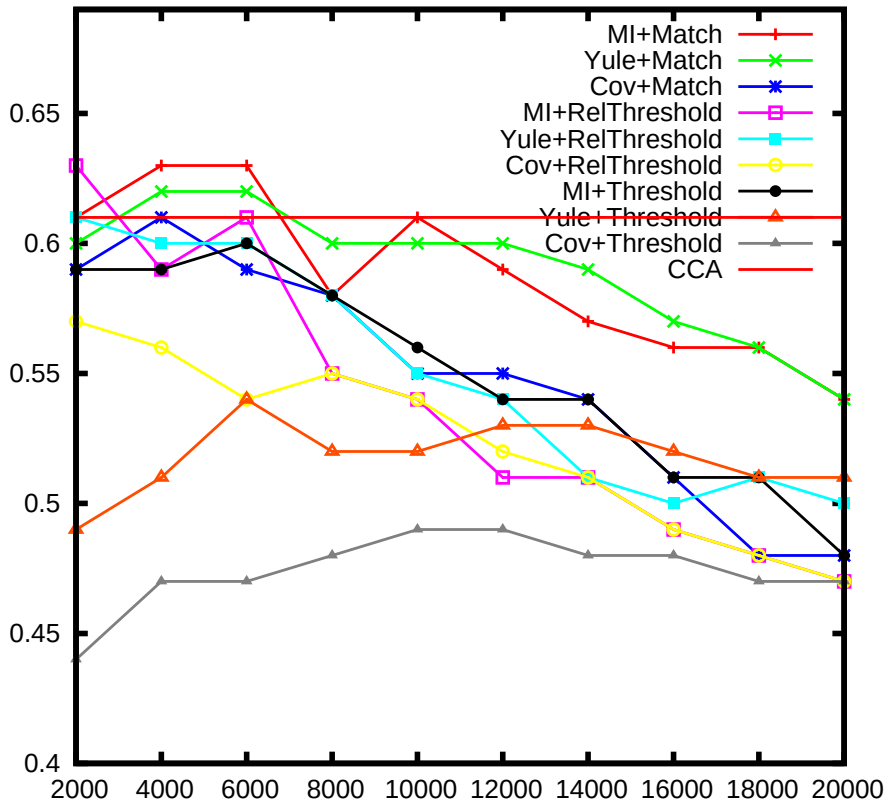


Figure A.1: Comparison of the word association measures along with different selection criteria. The x-axis plots the number of non-zero entries in the covariance matrices and the y-axis plots the accuracy of top-ranked document.

(Match). Fig. A.1 shows the performance of these different combinations with varying levels of sparsity in the covariance matrices. The horizontal line represents the performance of CCA on this data set. We start with 2000 non-zero entries in the covariance matrices and experiment up to 20,000 non-zero entries. Since our data set has 2000 words in each language, 2000 non-zero entries in a covariance matrix implies that, on an average, every word is associated with only one word. This results in highly sparse covariance matrices.

Overall, we observe that reducing the level of sparsity , *i.e.,* selecting more number of elements in the covariance matrices, increases the performance slightly and then decreases again. From the figure, it seems that sparsifying the covariance matrices might help in improving the performance of the task. But it is interesting to note that not all the models perform better than CCA. In fact, both the models that achieve better scores use Matching as the selection criteria. This suggests that, apart from the weighting of the word pairs, appropriate selection of the word pairs is also equally important. This is the reason why I only reported results with Matching as the selection criteria in Sec. 4.3.1. From this figure, we observe that Mutual Information and Yule's ω perform competitively but they consistently outperform models that use covariance as the association measure.

Appendix B

Derivations

B.1 Inverse of a Block Matrix

Let $M = \left(E - FG^{-1}F^T\right)^{-1}$, then the inverse of the block matrix $\begin{bmatrix} E & F \\ F^T & G \end{bmatrix}$ is given by:

$$\begin{bmatrix} E & F \\ F^T & G \end{bmatrix}^{-1} = \begin{bmatrix} M & -MFG^{-1} \\ -G^{-1}F^TM & G^{-1} + G^{-1}F^TMFG^{-1} \end{bmatrix}$$

It is easy to verify this identity by multiplying with the block matrix and then showing that it is identity matrix.

B.2 Optimizing Regularized Interlingual Projections

The Laplacian of the regularized projections is:

$$\mathcal{L} = \sum_l \frac{1}{n_l} ||X_l^T\mathbf{a}^{(l)} - P_l^T(\mathbf{u}+\mathbf{r}^{(l)})||^2 + \tau \sum_l \|P_l^T\mathbf{r}^{(l)}\|^2 + \alpha\Big(\sum_l \frac{1}{n_l}\|X_l^T\mathbf{a}^{(l)}\|^2 - 1\Big) + \beta\Big(\sum_l \frac{1}{n_l}\|P_l^T\mathbf{u}\|^2 - 1\Big)$$

Differentiating the Lagrangian with respect to $\mathbf{a}^{(l)}$, $\mathbf{r}^{(l)}$ and $\mathbf{u}$ and setting them to zero, yields the following constraints:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(l)}} &= \frac{1}{n_l} X_l\Big(X_l^T\mathbf{a}^{(l)} - P_l^T\big(\mathbf{u}+\mathbf{r}^{(l)}\big)\Big) + \alpha\frac{X_lX_l^T\mathbf{a}^{(l)}}{n_l} \\
&= (1+\alpha)\frac{1}{n_l}X_lX_l^T\mathbf{a}^{(l)} - \frac{1}{n_l}X_lP_l^T(\mathbf{u}+\mathbf{r}^{(l)}) \\
\frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(l)}} = 0 &\Rightarrow (1+\alpha)X_lX_l^T\mathbf{a}^{(l)} - X_lP_l^T\mathbf{r}^{(l)} = X_lP_l^T\mathbf{u} \qquad \text{(B.1)} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{r}^{(l)}} &= -\frac{1}{n_l}P_l\Big(X_l^T\mathbf{a}^{(l)} - P_l^T(\mathbf{u}+\mathbf{r}^{(l)}\Big) + \tau P_lP_l^T\mathbf{r}^{(l)} \\
&= -\frac{1}{n_l}P_lX_l^T\mathbf{a}^{(l)} + \Big(\frac{1}{n_l}+\tau\Big)P_lP_l^T\mathbf{r}^{(l)} + \frac{1}{n_l}P_lP_l^T\mathbf{u} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{r}^{(l)}} = 0 &\Rightarrow -P_lX_l^T\mathbf{a}^{(l)} + (1+\tau n_l)P_lP_l^T\mathbf{r}^{(l)} = -P_lP_l^T\mathbf{u} \qquad \text{(B.2)}
\end{aligned}$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{u}} &= -\sum_l \frac{1}{n_l} P_l \Big(X_l^T \mathbf{a}^{(l)} - P_l^T (\mathbf{u} + \mathbf{r}^{(l)}) \Big) + \beta \sum_l \frac{1}{n_l} P_l P_l^T \mathbf{u} \\
&= -\sum_l \frac{1}{n_l} P_l X_l^T \mathbf{a}^{(l)} + \sum_l \frac{1}{n_l} P_l P_l^T \mathbf{r}^{(l)} + (1+\beta) \sum_l \frac{1}{n_l} P_l P_l^T \mathbf{u} \\
\frac{\partial \mathcal{L}}{\partial \mathbf{u}} = 0 \;\; &\Rightarrow \;\; \sum_l \frac{1}{n_l} P_l X_l^T \mathbf{a}^{(l)} - \sum_l \frac{1}{n_l} P_l P_l^T \mathbf{r}^{(l)} = (1+\beta) \sum_l \frac{1}{n_l} P_l P_l^T \mathbf{u} \qquad \text{(B.3)}
\end{aligned}
$$

These constraints can be comprehensively written in matrix format as follows:

$$
\begin{bmatrix} (1+\alpha) X_l X_l^T & -X_l P_l^T \\ -P_l X_l^T & (1+\tau n_l) P_l P_l^T \end{bmatrix} \begin{bmatrix} \mathbf{a}^{(l)} \\ \mathbf{r}^{(l)} \end{bmatrix} = \begin{bmatrix} X_l P_l^T \\ -P_l P_l^T \end{bmatrix} \mathbf{u} \qquad \text{(B.4)}
$$

$$
\sum_l \frac{1}{n_l} \begin{bmatrix} P_l X_l^T & -P_l P_l^T \end{bmatrix} \begin{bmatrix} \mathbf{a}^{(l)} \\ \mathbf{r}^{(l)} \end{bmatrix} = (1+\beta) \sum_l \frac{1}{n_l} P_l P_l^T \mathbf{u} \qquad \text{(B.5)}
$$

Let $E_l = (1+\alpha) X_l X_l^T$, $F_l = -X_l P_l^T$ and $G_l = (1+\tau n_l) P_l P_l^T$. Then the constraints become:

$$
\begin{bmatrix} E_l & F_l \\ F_l^T & G_l \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{a}^{(l)} \\ \mathbf{r}^{(l)} \end{bmatrix} = \begin{bmatrix} -F_l \\ \frac{-G_l}{1+\tau n_l} \end{bmatrix} \mathbf{u} \qquad \text{(B.6)}
$$

$$
\sum_l \frac{1}{n_l} \begin{bmatrix} -F_l^T & \frac{-G_l}{1+\tau n_l} \end{bmatrix} \begin{bmatrix} \mathbf{a}^{(l)} \\ \mathbf{r}^{(l)} \end{bmatrix} = (1+\beta) \sum_l \frac{1}{n_l} P_l P_l^T \, \mathbf{u} \qquad \text{(B.7)}
$$

Then, we write $\mathbf{a}^{(l)}$ and $\mathbf{r}^{(l)}$ interms of $\mathbf{u}$ using Eq. B.6 and substitute in Eq. B.7. Then, $\mathbf{u}$ can be solved for using the following generalized eigenvalue problem:

$$
\sum_l \frac{1}{n_l} \begin{bmatrix} -F_l^T & \frac{-G_l}{1+\tau n_l} \end{bmatrix} \begin{bmatrix} E_l & F_l \\ F_l^T & G_l \end{bmatrix}^{-1} \begin{bmatrix} -F_l \\ \frac{-G_l}{1+\tau n_l} \end{bmatrix} \mathbf{u} = (1+\beta) \sum_l \frac{1}{n_l} P_l P_l^T \, \mathbf{u} \qquad \text{(B.8)}
$$

After solving for $\mathbf{u}$, we can compute $\mathbf{a}^{(l)}$ and $\mathbf{r}^{(l)}$ as follows:

$$
\begin{bmatrix} \mathbf{a}^{(l)} \\ \mathbf{r}^{(l)} \end{bmatrix} = \begin{bmatrix} E_l & F_l \\ F_l^T & G_l \end{bmatrix} \begin{bmatrix} -F_l \\ \frac{-G_l}{1+\tau n_l} \end{bmatrix} \mathbf{u} \qquad \text{(B.9)}
$$

B.3 Prediction Problem in Regularized Projections

Given a name in l^{th} language as $\mathbf{x}^{(l)}$, its transliteration in m^{th} language is obtained by solving:

$$
\begin{aligned}
\mathbf{x}^{(m)*} &= \arg\min_{\mathbf{x}^{(m)}} L\big(\mathbf{x}^{(l)}, \mathbf{x}^{(m)}\big) \qquad \text{where} \qquad \text{(B.10)} \\
L\big(\mathbf{x}^{(l)}, \mathbf{x}^{(m)}\big) &= \min_{\mathbf{p} \in \mathbb{R}^c} \|A_l^T \mathbf{x}^{(l)} - U_l^T \mathbf{p}\|^2 + \|A_m^T \mathbf{x}^{(m)} - U_m^T \mathbf{p}\|^2
\end{aligned}
$$

Differentiating the loss $L\big(\mathbf{x}^{(l)}, \mathbf{x}^{(m)}\big)$ with respect to $\mathbf{p}$ and setting it to zero, results in:

$$
\frac{\partial L}{\partial \mathbf{p}} = U_l \Big(U_l^T \mathbf{p} - A_l^T \mathbf{x}^{(l)} \Big) + U_m \Big(U_m^T \mathbf{p} - A_m^T \mathbf{x}^{(m)} \Big)
$$

$$\frac{\partial L}{\partial \mathbf{p}} = 0 \quad \Rightarrow \quad \Big(U_l U_l^T + U_m U_m^T\Big)\mathbf{p} = \Big(U_l A_l^T \mathbf{x}^{(l)} + U_m A_m^T \mathbf{x}^{(m)}\Big) \tag{B.11}$$

$$\Rightarrow \mathbf{p} \quad = \quad \Big(U_l U_l^T + U_m U_m^T\Big)^{-1}\Big(U_l A_l^T \mathbf{x}^{(l)} + U_m A_m^T \mathbf{x}^{(m)}\Big) \tag{B.12}$$

$$\Rightarrow \mathbf{p} \quad = \quad C_{lm}^{-1}\Big(U_l A_l^T \mathbf{x}^{(l)} + U_m A_m^T \mathbf{x}^{(m)}\Big) \text{ where } C_{lm} = U_l U_l^T + U_m U_m^T \tag{B.13}$$

Now in order to solve for the relation between $\mathbf{x}^{(l)}$ and the optimal $\mathbf{x}^{(m)}$, we need to substitute $\mathbf{p}$ back in the loss function and solve for the $\mathbf{x}^{(m)}$. Solving it analytically leads to a gaudy expression. Instead, to keep it simple, we treat the optimal $\mathbf{p}$ solved by Eq. B.13 *as if it is independent* of $\mathbf{x}^{(m)}$ (which it is clearly not) and proceed for solving $\mathbf{x}^{(m)}$. Under this assumption, the relation between $\mathbf{x}^{(l)}$ and $\mathbf{x}^{(m)}$ is obtained by differentiating the loss function w.r.t to $\mathbf{x}^{(m)}$ and setting it to zero.

$$\begin{aligned} A_m(A_m^T \mathbf{x}^{(m)} - U_m^T \mathbf{p}) &= 0 \\ A_m A_m^T \mathbf{x}^{(m)} &= A_m U_m^T C_{lm}^{-1}\Big(U_l A_l^T \mathbf{x}^{(l)} + U_m A_m^T \mathbf{x}^{(m)}\Big) \\ A_m\Big(I - U_m^T C_{lm}^{-1} U_m\Big) A_m^T \mathbf{x}^{(m)} &= A_m U_m^T C_{lm}^{-1} U_l A_l^T \mathbf{x}^{(l)} \end{aligned} \tag{B.14}$$

Appendix C

Reranking Sensitivity

There are following hyper parameters in the low-dimensional discriminative reranking models: regularization parameter τ, weight parameter λ in the discriminative and softened discriminative models, the linear combination weight w with the Viterbi decoding score, and finally, the size of the lower dimensional subspace (k). I use grid search to tune these parameters based on the development data set. The optimal hyperparameter values differ based on the model and the language, but the tagging accuracy is relatively robust with respect to these parameter values. For English, the best values for the discriminative model are $\tau = 0.95$, $\lambda = 0.3$ and $k = 75$. For the same language, Fig. C.1 shows the performance with respect to τ and λ parameters, respectively, with other parameters fixed to their optimal values. Notice that, although the performance varies it is always more than the accuracy of the baseline tagger (96.74%).

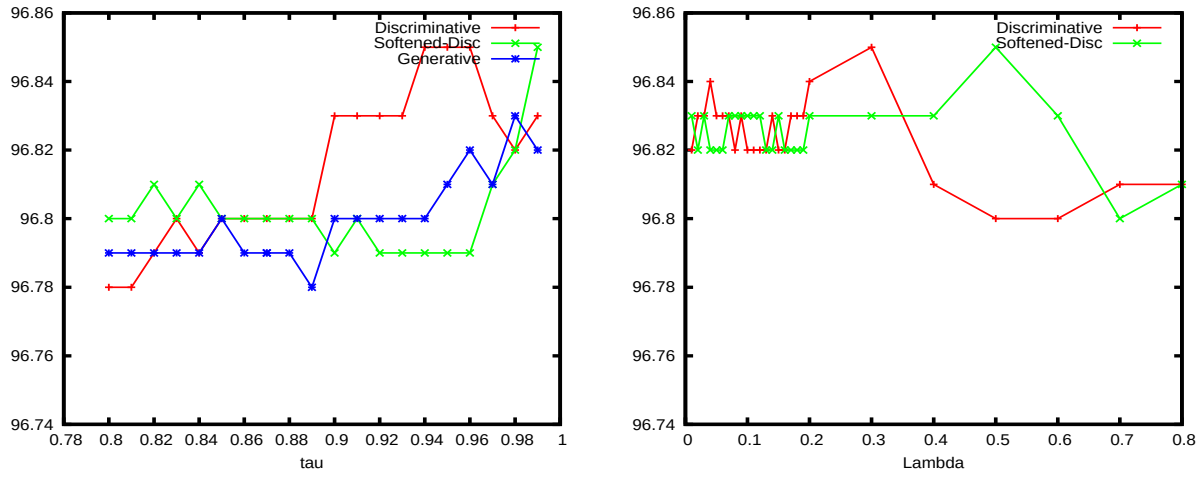


Figure C.1: Tagging accuracy with hyperparameters τ and λ on English development data set.

Appendix D

Relevant Publications

Chapter 4

Hal Daumé III and Jagadeesh Jagarlamudi. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 407–412, Portland, Oregon, USA, June 2011. ACL.

Jagadeesh Jagarlamudi, Raghavendra Udupa, Hal Daumé III, and Abhijit Bhole. Improving bilingual projections via sparse covariance matrices. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 930–940, Edinburgh, Scotland, UK., July 2011. ACL.

Chapter 5

Jagadeesh Jagarlamudi and Hal Daumé, III. Regularized interlingual projections: evaluation on multilingual transliteration. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 12–23, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Chapter 6

Jagadeesh Jagarlamudi and Hal Daumé III. Low-dimensional discriminative reranking. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Montreal, Canada, June 2012. ACL.

Bibliography

[1] Nasreen AbdulJaleel and Leah S. Larkey. Statistical transliteration for english-arabic cross language information retrieval. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM '03, pages 139–146, New York, NY, USA, 2003. ACM.

[2] Yaser Al-Onaizan and Kevin Knight. Machine transliteration of names in arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, SEMITIC '02, pages 1–13, Stroudsburg, PA, USA, 2002. ACL.

[3] Cédric Archambeau and Francis R. Bach. Sparse probabilistic projections. In *Neural Information Processing Systems*, 2008.

[4] Francis R. Bach and Michael I. Jordan. A probabilistic interpretation of canonical correlation analysis. Technical report, Dept Statist Univ California Berkeley CA Tech, 2005.

[5] Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. Supervised semantic indexing. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 187–196, New York, NY, USA, 2009. ACM.

[6] Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. Learning to rank with (a lot of) word features. *Inf. Retr.*, 13(3):291–314, June 2010.

[7] Jing Bai, Ke Zhou, Guirong Xue, Hongyuan Zha, Gordon Sun, Belle Tseng, Zhaohui Zheng, and Yi Chang. Multi-task learning for learning to rank in web search. In *Proceeding of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1549–1552, New York, NY, USA, 2009. ACM.

[8] Gükhan H. Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data (Neural Information Processing)*. The MIT Press, 2007.

[9] P. Baldi and K. Hornik. Neural networks and principal component analysis: learning from examples without local minima. *Neural Netw.*, 2(1):53–58, January 1989.

[10] Lisa Ballesteros and W. Bruce Croft. Dictionary methods for cross-lingual information retrieval. In *Proceedings of the 7th International Conference on Database and Expert Systems Applications*, DEXA '96, pages 791–801, London, UK, 1996. Springer-Verlag.

[11] Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. Multilingual subjectivity analysis using machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 127–135, Stroudsburg, PA, USA, 2008. ACL.

[12] Onureena Banerjee, Alexandre d'Aspremont, and Laurent El Ghaoui. Sparse covariance selection via robust maximum likelihood estimation. *CoRR*, abs/cs/0506023, 2005.

[13] Marco Baroni and Roberto Zamparelli. Nouns are vectors, adjectives are matrices: representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1183–1193, Stroudsburg, PA, USA, 2010. ACL.

[14] Nuria Bel, Cornelis H. Koster, and Marta Villegas. Cross-lingual text categorization. In *European Conference on Digital Libraries*, pages 126–139, 2003.

[15] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, June 2003.

[16] Paul N. Bennett, Krysta Svore, and Susan T. Dumais. Classification-enhanced ranking. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 111–120, New York, NY, USA, 2010. ACM.

[17] David M. Blei and Michael I. Jordan. Modeling annotated data. In *SIGIR '03*, pages 127–134, New York, NY, USA, 2003. ACM.

[18] David M. Blei and John Lafferty. Topic models. In *Text Mining: Theory and Applications*. Taylor and Francis, 2009.

[19] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Maching Learning Research*, 3:993–1022, 2003.

[20] Jordan Boyd-Graber and David M. Blei. Multilingual topic models for unaligned text. In *Uncertainty in Artificial Intelligence*, 2009.

[21] Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. A statistical approach to sense disambiguation in machine translation. In *Speech and Natural Language, Proceedings of a Workshop held at Pacific Grove, HLT*, 1991.

[22] Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19:263–311, June 1993.

[23] Sabine Buchholz and Erwin Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 149–164, Stroudsburg, PA, USA, 2006. ACL.

[24] J. D. Caroll. Generalization of canonical correlation analysis to three or more sets of variables. *Proceedings of the Americal Psychological Association*, pages 227–228, 1968.

[25] Marine Carpuat, Hal Daumé III, Katie Henry, Ann Irvine, Jagadeesh Jagarlamudi, and Rachel Rudinger. Sensespotting: Never let your parallel data tie you to an old domain. In *Association for Computational Linguistics*, Sophia, Bulgaria, July 2013.

[26] Marine Carpuat, Hal Daumé III, Ann Irvine, John Morgan, and Dragos Munteanu. Measuring machine translation errors in new domains. In *manuscript*, 2013.

[27] Marine Carpuat and Dekai Wu. Improving statistical machine translation using word sense disambiguation. In *In The 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 61–72, 2007.

[28] Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Stroudsburg, PA, USA, 2005. ACL.

[29] Jiming Chen, Chengqun Wang, Youxian Sun, and Xuemin (Sherman) Shen. Semi-supervised laplacian regularized least squares algorithm for localization in wireless sensor networks. *Computer Networks*, pages 2481–2491, 2011.

[30] Jing Chen and Yang Liu. Locally linear embedding: a survey. *Artif. Intell. Rev.*, 36(1):29–48, June 2011.

[31] David Chiang, Kevin Knight, and Wei Wang. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226, Stroudsburg, PA, USA, 2009. ACL.

[32] Manoj K. Chinnakotla, Karthik Raman, and Pushpak Bhattacharyya. Multilingual PRF: english lends a helping hand. In *SIGIR '10*, pages 659–666, New York, NY, USA, 2010. ACM.

[33] Manoj K. Chinnakotla, Karthik Raman, and Pushpak Bhattacharyya. Multilingual pseudo-relevance feedback: performance study of assisting languages. In *ACL '10:*, pages 1346–1356, Morristown, NJ, USA, 2010.

[34] Shay B. Cohen, Dipanjan Das, and Noah A. Smith. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 50–61, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[35] Michael Collins. Ranking algorithms for named-entity extraction: boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 489–496, Stroudsburg, PA, USA, 2002. ACL.

[36] Michael Collins and Terry Koo. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31:25–70, March 2005.

[37] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM.

[38] Dipanjan Das and Slav Petrov. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 600–609, Stroudsburg, PA, USA, 2011. ACL.

[39] Dipanjan Das and Noah A. Smith. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proc. of ACL-IJCNLP*, 2009.

[40] Hal Daume III and Jagadeesh Jagarlamudi. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 407–412, Portland, Oregon, USA, June 2011. ACL.

[41] Scott Deerwester. Improving Information Retrieval with Latent Semantic Indexing. In Christine L. Borgman and Edward Y. H. Pai, editors, *Proceedings of the 51st ASIS Annual Meeting (ASIS '88)*, volume 25, Atlanta, Georgia, October 1988. American Society for Information Science.

[42] Paramveer S. Dhillon, Dean Foster, and Lyle Ungar. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, 2011.

[43] Mona Diab and Philip Resnik. An unsupervised method for word sense tagging using parallel corpora. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 255–262, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[44] Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. ACL.

[45] Bonnie Dorr. Parameterization of the interlingua in machine translation. In *Proceedings of the 14th conference on Computational linguistics - Volume 2*, COLING '92, pages 624–630, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics.

[46] Bonnie J. Dorr. Machine translation divergences: a formal description and proposed solution. *Comput. Linguist.*, 20(4):597–633, December 1994.

[47] Bonnie J. Dorr, Eduard H. Hovy, and Lori S. Levin. Machine translation: Interlingual methods, 2004.

[48] Susan T. Dumais, Thomas K. Landauer, and Michael L. Littman. Automatic cross-linguistic information retrieval using latent semantic indexing. In *Working Notes of the Workshop on Cross-Linguistic Information Retrieval, SIGIR*, pages 16–23, Zurich, Switzerland, 1996. ACM.

[49] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74, March 1993.

[50] Helge Dyvik. Translations as semantic mirrors. In *Proceedings of the Workshop W13: Multilinguality in the lexicon II*, pages 24–44. The 13th biennial European Conference on Artificial Intelligence ECAI 98, 1998.

[51] J. R. Firth. A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis (special volume of the Philological Society)*, 1952-59:1–32, 1957.

[52] Alexander Fraser, Renjing Wang, and Hinrich Schütze. Rich bitext projection features for parse reranking. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 282–290, Stroudsburg, PA, USA, 2009. ACL.

[53] G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '88, pages 465–480, New York, NY, USA, 1988. ACM.

[54] William A. Gale and Kenneth W. Church. A program for aligning sentences in bilingual corpora. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 177–184, Morristown, NJ, USA, 1991. ACL.

[55] Jianfeng Gao and Mark Johnson. A comparison of Bayesian estimators for unsupervised Hidden Markov Model POS taggers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 344–352, Honolulu, Hawaii, October 2008. ACL.

[56] Jianfeng Gao, Kristina Toutanova, and Wen-tau Yih. Clickthrough-based latent semantic models for web search. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 675–684, New York, NY, USA, 2011. ACM.

[57] Jianfeng Gao, Qiang Wu, Christopher Burges, Krysta Svore, Yi Su, Nazan Khan, Shalin Shah, and Hongyan Zhou. Model adaptation via model interpolation and boosting for web search ranking. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 505–513, Morristown, NJ, USA, 2009.

[58] Wei Gao, John Blitzer, and Ming Zhou. Using english information in non-english web search. In *iNEWS '08: Proceeding of the 2nd ACM workshop on Improving non english web searching*, pages 17–24, New York, NY, USA, 2008. ACM.

[59] Wei Gao, John Blitzer, Ming Zhou, and Kam-Fai Wong. Exploiting bilingual information to improve web search. In *Proceedings of Human Language Technologies: The 2009 Conference of the Association for Computational Linguistics*, ACL-IJCNLP '09, pages 1075–1083, Morristown, NJ, USA, 2009. ACL.

[60] Wei Gao, Cheng Niu, Ming Zhou, and Kam-Fai Wong. Joint ranking for multilingual web search. In *Advances in Information Retrieval*, volume 5478 of *Lecture Notes in Computer Science*, pages 114–125. Springer Berlin / Heidelberg, 2009.

[61] Wei Gao, Kam-fai Wong, and Wai Lam. Phoneme-based transliteration of foreign names for OOV problem. In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP)*, pages 374–381, 2004.

[62] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis, Second Edition (Chapman & Hall/CRC Texts in Statistical Science)*. Chapman and Hall/CRC, 2 edition, July 2003.

[63] Zoubin Ghahramani. One Hidden Layer Linear Networks and Canonical Correlations. In *manuscript*, 1996.

[64] Eugenie Giesbrecht. In search of semantic compositionality in vector spaces. In *Proceedings of the 17th International Conference on Conceptual Structures: Conceptual Structures: Leveraging Semantic Technologies*, ICCS '09, pages 173–184, Berlin, Heidelberg, 2009. Springer-Verlag.

[65] Joao Gräca, Kuzman Ganchev, Ben Taskar, and Fernando Pereira. Posterior vs Parameter Sparsity in Latent Variable Models. In *Proceedings of NIPS*, 2009.

[66] Tl Griffiths, M. Steyvers, and Jb Tenenbaum. Topics in semantic representation. *Psychological Review*, 114(2):211–244, 2007.

[67] Emiliano Guevara. Computing semantic compositionality in distributional semantics. In *Proceedings of the Ninth International Conference on Computational Semantics*, IWCS '11, pages 135–144, Stroudsburg, PA, USA, 2011. ACL.

[68] Aria Haghighi, Percy Liang, Taylor B. Kirkpatrick, and Dan Klein. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: HLT*, pages 771–779, Columbus, Ohio, June 2008. ACL.

[69] Li Haizhou, Zhang Min, and Su Jian. A joint source-channel model for machine transliteration. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA, 2004. ACL.

[70] Nathan Halko, Per-Gunnar. Martinsson, and A. Joel Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. Technical report, California Institute of Technology, 2009.

[71] David R. Hardoon and John Shawe-Taylor. Sparse canonical correlation analysis. *Journal of Machine Learning*, 83(3):331–353, 2011.

[72] David R. Hardoon, Sandor R. Szedmak, and John R. Shawe-taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Comput.*, 16:2639–2664, December 2004.

[73] Martin Haspelmath, Matthew Dryer, David Gil, and Bernard Comrie, editors. *The World Atlas of Language Structures*. Oxford University Press, 2005.

[74] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[75] Xiaofei He and Partha Niyogi. Locality preserving projections. In *In Advances in Neural Information Processing Systems 16*. MIT Press, 2003.

[76] Ulf Hermjakob, Kevin Knight, and Hal Daumé III. Name translation in statistical machine translation - learning when to transliterate. In *Proceedings of ACL-08: HLT*, pages 389–397, Columbus, Ohio, June 2008. ACL.

[77] Hung Huu Hoang, Su Nam Kim, and Min-Yen Kan. A Re-examination of Lexical Association Measures. In *Proceedings of ACL-IJCNLP 2009 Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*, Singapore, August 2009. ACL.

[78] Paul Horst. Relations among m sets of measures. *Psychometrika*, 26:129–149, 1961. 10.1007/BF02289710.

[79] Agnar Höskuldsson. PLS regression methods. *Journal of Chemometrics*, 2(3):211–228, 1988.

[80] Harold Hotelling. Relation between two sets of variables. *Biometrica*, 28:322–377, 1936.

[81] Daniel Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden markov models. *CoRR*, abs/0811.4413, 2008.

[82] Zhongqiang Huang, Mary Harper, and Wen Wang. Mandarin part-of-speech tagging and discriminative reranking. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1093–1102, Prague, Czech Republic, June 2007. ACL.

[83] W. John Hutchins. The evoluation of machine translation systems. *Lawson*, pages 21–37, 1982.

[84] W. John Hutchins and Harold L. Somers. *An Introduction to Machine Translation*. Academic Press, 1992.

[85] Nancy Ide. Cross-lingual sense determination: Can it work? *Computers and the Humanities*, 34(1-2):223–234, 2000.

[86] Diana Zaiu Inkpen and Graeme Hirst. Acquiring collocations for lexical choice between near-synonyms. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition - Volume 9*, ULA '02, pages 67–76, Stroudsburg, PA, USA, 2002. ACL.

[87] Jagadeesh Jagarlamudi and Paul Bennett. Fractional similarity: Cross-lingual feature selection for search. In *Proceedings of The 33rd European Conference on Information Retrieval*, ECIR 2011, Lecture Notes in Computer Science. Springer, 2011.

[88] Jagadeesh Jagarlamudi, Paul N. Bennett, and Krysta M. Svore. Leveraging interlingual classification for web search. Technical Report MSR-TR-2012-11, Microsoft Research, 2012.

[89] Jagadeesh Jagarlamudi and Hal Daumé, III. Regularized interlingual projections: evaluation on multilingual transliteration. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 12–23, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[90] Jagadeesh Jagarlamudi and Hal Daumé III. Extracting multilingual topics from unaligned comparable corpora. In *Advances in Information Retrieval, 32nd European Conference on IR Research, ECIR*, volume 5993, pages 444–456, Milton Keynes, UK, 2010. Springer.

[91] Jagadeesh Jagarlamudi and Hal Daumé III. Low-dimensional discriminative reranking. In *Proceedings of Human Language Technologies: The 2012 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Montreal, Canada, June 2012. ACL.

[92] Jagadeesh Jagarlamudi, Hal Daume III, and Raghavendra Udupa. From bilingual dictionaries to interlingual document representations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 147–152, Portland, Oregon, USA, June 2011. ACL.

[93] Jagadeesh Jagarlamudi, Raghavendra Udupa, Hal Daumé III, and Abhijit Bhole. Improving bilingual projections via sparse covariance matrices. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 930–940, Edinburgh, Scotland, UK., July 2011. ACL.

[94] Mitchell Jeff and Lapata Mirella. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010.

[95] Jiang Jing. A literature survey on domain adaptation of statistical classifiers, 2008.

[96] Blitzer John and Hal Daumé III. Domain adaptation: Tutorial at the international conference on machine learning, 2010.

[97] Mark Johnson and Ahmet Engin Ural. Reranking the Berkeley and Brown parsers. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 665–668, Stroudsburg, PA, USA, 2010. ACL.

[98] R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987.

[99] Sung Young Jung, SungLim Hong, and Eunok Paek. An english to korean transliteration model of extended markov window. In *Proceedings of the 18th conference on Computational linguistics - Volume 1*, COLING '00, pages 383–389, Stroudsburg, PA, USA, 2000. ACL.

[100] Byung-Ju Kang and Key-Sun Choi. Two approaches for the resolution of word mismatch problem caused by english words and foreign words in korean information retrieval. In *Proceedings of the 5th international workshop on on Information retrieval with Asian languages*, IRAL '00, pages 133–140, New York, NY, USA, 2000. ACM.

[101] J. R. Kettenring. Canonical analysis of several sets of variables. *Biometrika*, 58:433–451, 1971.

[102] Mitesh M. Khapra, Raghavendra Udupa, A. Kumaran, and Pushpak Bhattacharyya. Pr + rq $\approx$ pq: Transliteration mining using bridge language. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010*, Atlanta, Georgia, USA, July 2010. AAAI Press.

[103] Alexandre Klementiev and Dan Roth. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 817–824, Stroudsburg, PA, USA, 2006. ACL.

[104] Kevin Knight and Jonathan Graehl. Machine transliteration. *Computational Linguistics*, 24(4):599–612, 1998.

[105] Philipp Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005. AAMT, AAMT.

[106] Sandra Kubler, Ryan McDonald, Joakim Nivre, and Graeme Hirst. *Dependency Parsing*. Morgan and Claypool Publishers, 2009.

[107] Abhishek Kumar, Piyush Rai, and Hal Daume III. Co-regularized multi-view spectral clustering. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1413–1421, 2011.

[108] Shaishav Kumar and Raghavendra Udupa. Learning hash functions for cross-view similarity search. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1360–1365. AAAI Press, 2011.

[109] Michael Lamar, Yariv Maron, Mark Johnson, and Elie Bienenstock. Svd and clustering for unsupervised pos tagging. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 215–219, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[110] Thomas K. Landauer, Darrell Laham, Bob Rehder, and M. E. Schreiner. How well can passage meaning be derived without using word order? a comparison of latent semantic analysis and humans. In *Proceedings of the 19th Annual Conference of the Cognitive Science Society*, pages 412–417, 1997.

[111] Samuel Larkin, Boxing Chen, George Foster, Ulrich Germann, Eric Joanis, Howard Johnson, and Roland Kuhn. Lessons from nrc's portage system at wmt 2010. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 127–132, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[112] Els Lefever and Veronique Hoste. Semeval-2010 task 3: cross-lingual word sense disambiguation. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, DEW '09, pages 82–87, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[113] Haizhou Li, A. Kumaran, Vladimir Pervouchine, and Min Zhang. Report of news 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, NEWS '09, pages 1–18, Stroudsburg, PA, USA, 2009. ACL.

[114] P. Liang, M. I. Jordan, and D. Klein. Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*, pages 590–599, 2011.

[115] Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 761–768, Stroudsburg, PA, USA, 2006. ACL.

[116] Roderick J A Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

[117] Bo Long, Philip S. Yu, and Zhongfei Zhang. A General Model for Multiple View Unsupervised Learning. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, 2008.

[118] Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods*, 28(2):203–208, June 1996.

[119] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[120] Thomas Mandl and Christa Womser-Hacker. The effect of named entities on effectiveness in cross-language information retrieval evaluation. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 1059–1064, New York, NY, USA, 2005. ACM.

[121] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[122] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.

[123] Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 91–98, Stroudsburg, PA, USA, 2005. ACL.

[124] Rada Mihalcea, Ravi Sinha, and Diana McCarthy. Semeval-2010 task 2: Cross-lingual lexical substitution. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 9–14, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[125] David Mimno, Hanna M. Wallach, Jason Naradowsky, David A. Smith, and Andrew McCallum. Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 880–889, Stroudsburg, PA, USA, 2009. ACL.

[126] Jeff Mitchell and Mirella Lapata. Vector-based Models of Semantic Composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June 2008. ACL.

[127] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[128] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 641–648, New York, NY, USA, 2007. ACM.

[129] Richard Montague. English as a formal language. In Richmond H. Thomason, editor, *Formal Philosophy: Selected Papers of Richard Montague*, pages 188–222. Yale University Press, New Haven, London, 1974.

[130] Robert C. Moore. On Log-Likelihood-Ratios and the Significance of Rare Events. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 333–340, Barcelona, Spain, July 2004. ACL.

[131] Dragos Stefan Munteanu and Daniel Marcu. Improving machine translation performance by exploiting non-parallel corpora. *Comput. Linguist.*, 31:477–504, December 2005.

[132] Xiaochuan Ni, Jian-Tao Sun, Jian Hu, and Zheng Chen. Mining multilingual topics from wikipedia. In *18th International World Wide Web Conference*, pages 1155–1155, April 2009.

[133] Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June 2007. ACL.

[134] Eric Nyberg, Teruko Mitamura, and Jaime G. Carbonell. The kant machine translation system: From r&d to initial deployment. In *Proceedings of LISA (The Library and Information Services in Astronomy) Workshop on Integrating Advanced Translation Technology*, 1997.

[135] Douglas W. Oard and R. Anne Diekema. Cross-language information retrieval. *Annual Review of Information Science (ARIST)*, 33:223–256, 1998.

[136] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA, 2003. ACL.

[137] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[138] M. K. Odel and R. C. Russel. U.s. patent numbers, 1,261,167 (1918) and 1,435,663(1922), 1918.

[139] Dean P. Foster Paramveer S. Dhillon, Jordan Rodu and Lyle H. Ungar. Two step cca: A new spectral method for estimating vector models of words. In *Proceedings of the 29th International Conference on Machine learning*, ICML'12, 2012.

[140] Foltz Peter W., Kintsch Walter, and Landauer Thomas K. The measurement of textual coherence with latent semantic analysis. *Discourse Process*, 15:85–307, 1998.

[141] Tony Plate. Holographic reduced representations: convolution algebra for compositional distributed representations. In *Proceedings of the 12th international joint conference on Artificial intelligence - Volume 1*, IJCAI'91, pages 30–35, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

[142] John C. Platt, Kristina Toutanova, and Wen-tau Yih. Translingual document representations from discriminative projections. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 251–261, Stroudsburg, PA, USA, 2010. ACL.

[143] Amruta Purandare and Ted Pedersen. Word Sense Discrimination by Clustering Contexts in Vector and Similarity Spaces. In Hwee T. Ng and Ellen Riloff, editors, *HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 41–48, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics.

[144] Novi Quadrianto and Christoph H. Lampert. Learning multi-view neighborhood preserving projections. In Lise Getoor and Tobias Scheffer, editors, *International Conference on Machine Learning (ICML)*, pages 425–432. Omnipress, 2011.

[145] Piyush Rai and Hal Daumé III. Multi-label prediction via sparse infinite cca. In *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2009.

[146] Reinhard Rapp. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 519–526, Stroudsburg, PA, USA, 1999. ACL.

[147] Sujith Ravi and Kevin Knight. Learning phoneme mappings for transliteration without parallel data. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 37–45, Boulder, Colorado, June 2009. ACL.

[148] K. Ahuja Ravindra, L. Magnanti Thomas, and B. Orlin James. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., 1993.

[149] Siva Reddy, Diana McCarthy, and Suresh Manandhar. An empirical study on compositionality in compound nouns. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP-11)*, Chiang Mai, Thailand, 2011.

[150] Harry T Reis and Charles M Judd. *Handbook of Research Methods in Social and Personality Psychology*. Cambridge University Press, 2000.

[151] Philip Resnik and David Yarowsky. Distinguishing systems and distinguishing senses: new evaluation methods for word sense disambiguation. *Nat. Lang. Eng.*, 5(2):113–133, June 1999.

[152] Roman Rosipal and Nicole Krämer. Overview and recent advances in partial least squares. In *in Subspace, Latent Structure and Feature Selection Techniques, Lecture Notes in Computer Science*, pages 34–51. Springer, 2006.

[153] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[154] Sebastian Rudolph and Eugenie Giesbrecht. Compositional matrix-space models of language. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 907–916, Stroudsburg, PA, USA, 2010. ACL.

[155] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[156] Alexander Schrijver. *Combinatorial Optimization*. Springer, 2003.

[157] Hinrich Schütze. Word space. In Lee C. Giles, Stephen J. Hanson, and Jack D. Cowan, editors, *Advances in Neural Information Processing Systems 5*, pages 895–902. San Francisco, CA: Morgan Kaufmann, 1993.

[158] Hinrich Schütze. Distributional part-of-speech tagging. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, EACL '95, pages 141–148, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[159] Hinrich Schütze. Automatic word sense discrimination. *Comput. Linguist.*, 24(1):97–123, March 1998.

[160] Libin Shen and Aravind K. Joshi. An SVM based voting algorithm with application to parse reranking. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 9–16, Stroudsburg, PA, USA, 2003. ACL.

[161] Libin Shen, Anoop Sarkar, and Franz Och. Discriminative reranking for machine translation. In *Human Language Technology Conference and the 5th Meeting of the North American Association for Computational Linguistics: HLT-NAACL 2004*, Boston, USA, May 2004.

[162] P. Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artif. Intell.*, 46(1-2):159–216, November 1990.

[163] Richard Socher, Eric H. Huang, Jeffrey Pennin, Andrew Y. Ng, and Christopher D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 801–809, 2011.

[164] Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing natural scenes and natural language with recursive neural networks. In Lise Getoor and Tobias Scheffer, editors, *ICML*, pages 129–136. Omnipress, 2011.

[165] Richard Sproat, Tao Tao, and ChengXiang Zhai. Named entity transliteration with comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 73–80, Stroudsburg, PA, USA, 2006. ACL.

[166] Jan-Benedict E. M. Steenkamp, Hans C. M. Van Trijp, and Jos M. F. Ten Berge. Perceptual mapping based on idiosyncratic sets of attributes. *Journal of Marketing Research*, 31(1):pp. 15–27, 1994.

[167] Liang Sun, Shuiwang Ji, Shipeng Yu, and Jieping Ye. On the equivalence between canonical correlation analysis and orthonormalized partial least squares. In *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI'09, pages 1230–1235, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[168] S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: Maximum margin regression; multiclass and multiview learning at one-class complexity. Technical report, University of Southampton, 2006.

[169] Sandor Szedmak, Tijl De Bie, and David R. Hardoon. A metamorphosis of canonical correlation analysis into multivariate maximum margin learning. In *Proceedings of the fifteenth European Symposium on Artificial Neural Networks*, 2007.

[170] Yoshio Takane, Heungsun Hwang, and Hervé Abdi. Regularized multiple-set canonical correlation analysis. *Psychometrika*, 73(4):753–775, December 2008.

[171] Lappoon R. Tang and Raymond J. Mooney. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 466–477, London, UK, UK, 2001. Springer-Verlag.

[172] Tao Tao, Su-Youn Yoon, Andrew Fister, Richard Sproat, and ChengXiang Zhai. Unsupervised named entity transliteration using temporal and phonetic correlation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 250–257, Stroudsburg, PA, USA, 2006. ACL.

[173] Ben Taskar, Carlos. Guestrin, and Daphne Koller. Max margin markov networks. In *Proceedings of NIPS 16*, 2004.

[174] Arthur Tenenhaus and Michel Tenenhaus. Regularized generalized canonical correlation analysis. *Psychometrika*, 76(2):257–284, 2011.

[175] Scott M. Thede and Mary P. Harper. A second-order Hidden Markov Model for part-of-speech tagging. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, pages 175–182. ACL, 1999.

[176] Landauer Thomas K. and Dumais Susan T. A solution to platos problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240, 1997.

[177] Jörg Tiedemann. News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing (vol V)*, pages 237–248. John Benjamins, Amsterdam/Philadelphia, 2009.

[178] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, pages 104–, New York, NY, USA, 2004. ACM.

[179] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[180] Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res. (JAIR)*, 37:141–188, 2010.

[181] Raghavendra Udupa, Saravanan K, Anton Bakalov, and Abhijit Bhole. "they are out there, if you know where to look": Mining transliterations of oov query terms for cross-language information retrieval. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ECIR '09, pages 437–448, Berlin, Heidelberg, 2009. Springer-Verlag.

[182] Raghavendra Udupa and Mitesh M. Khapra. Transliteration equivalence using canonical correlation analysis. In *ECIR'10*, pages 75–86, 2010.

[183] Raghavendra Udupa, K. Saravanan, A. Kumaran, and Jagadeesh Jagarlamudi. Mint: A method for effective and scalable mining of named entity transliterations from large comparable corpora. In *EACL*, pages 799–807. The Association for Computer Linguistics, 2009.

[184] M. van de Velden and Y. Takane. Generalized canonical correlation analysis with missing values. Econometric Institute Report EI 2009-28, Erasmus University Rotterdam, Econometric Institute, November 2009.

[185] Alexei Vinokourov, John Shawe-taylor, and Nello Cristianini. Inferring a semantic representation of text via cross-language correlation analysis. In *Advances in Neural Information Processing Systems*, pages 1473–1480, Cambridge, MA, 2003. MIT Press.

[186] Thuy Vu, AiTi Aw, and Min Zhang. Feature-based method for document alignment in comparable news corpora. In *EACL*, pages 843–851, 2009.

[187] Stephen Wan, Mark Dras, Robert Dale, and Cecile Paris. Using dependency-based features to take the 'para-farce' out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 131–138, Sydney, Australia, November 2006.

[188] Zhuoran Wang, John Shawe-Taylor, and Sandor Szedmak. Kernel regression based machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, NAACL-Short '07, pages 185–188, Stroudsburg, PA, USA, 2007. ACL.

[189] Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. Online Large-Margin Training for Statistical Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773, Prague, Czech Republic, June 2007. ACL.

[190] Dominic Widdows. Semantic vector products: Some initial investigations. In *Second AAAI Symposium on Quantum Interaction*, 2008.

[191] H. Wold. Path models with latent variables: The niplas approach. *Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building*, pages 307–357, 1975.

[192] K. J. Worsley, J-B. Poline, K. J. Friston, and A.C. Evans. Characterizing the response of pet and fmri data using multivariate linear models. *NeuroImage*, 6(4):305 – 319, 1997.

[193] Tian Xia, Dacheng Tao, Tao Mei, and Yongdong Zhang. Multiview spectral embedding. *Trans. Sys. Man Cyber. Part B*, 40(6):1438–1446, December 2010.

[194] Antonio T. Yair Weiss. Spectral hashing. In *Advances in Neural Information Processing Systems*, 2008.

[195] Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL '11, pages 247–256, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[196] Su-Youn Yoon, Kyoung-Young Kim, and Richard Sproat. Multilingual transliteration using feature based phonetic method. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 112–119, Prague, Czech Republic, June 2007. ACL.

[197] John M. Zelle and Raymond J. Mooney. Learning to parse database queries using inductive logic programming. In *Proceedings of the thirteenth national conference on Artificial intelligence - Volume 2*, AAAI'96, pages 1050–1055. AAAI Press, 1996.

[198] Duo Zhang, Qiaozhu Mei, and ChengXiang Zhai. Cross-lingual latent topic extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1128–1137, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[199] Duo Zhang, Qiaozhu Mei, and ChengXiang Zhai. Cross-lingual latent topic extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1128–1137, Uppsala, Sweden, July 2010. ACL.