Robust Representation Learning for Low Resource Languages

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

MS in Computer Science
by
Research

by

Syed Sarfaraz Akhtar
201202044
syed.akhtar@research.iiit.ac.in

International Institute of Information Technology, Hyderabad
(Deemed to be University)
Hyderabad - 500 032, INDIA
March 2018

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Robust Representation Learning for Low Resource Languages" by Syed Sarfaraz Akhtar, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Manish Shrivastava

This thesis is dedicated to my **Friends** and **Family**.

Acknowledgements

I would like to thank my advisor Dr. Manish Shrivastava for his expertise, guidance and patience. I would also like to thank Dr. Dipti Misra Sharma for helping in building up the essential concepts required for research in NLP. I would like to thank Raveesh Motlani for introducing me to this research area and providing valuable suggestions over time.

I would like to thank Arjit Srivastava for the nights spent in providing finishing touches to the drafts. I would also like to thank Arihant Gupta, Avijit Vajpayee and Madan Jhawar for the many hours spent in research discussions.

I am indebted to my parents for teaching me the value of education and providing me with the numerous opportunities that led to this thesis. I am blessed to have Fahim as my brother and would like to thank him for being a constant source of encouragement.

I would also like to take this opportunity to thank Nazrul, Pandey, Chamadia, Arnav, Krishnam, Umang, Kannan, Tanmay, Anuj and Khawad for being evergreen friends to me. I will always look back at these times as those when I was in good company. I wish you guys all the luck in the world.

Abstract

Understanding the meaning of words is essential for most natural language processing tasks. Word representations are means to mathematically represent the meaning of a word in a way that computers can understand. These representations are often in the form of vectors in which words are represented in a continuous vector-space of fixed dimensionality also referred to as word embeddings.

In this thesis, we focus on generating better and reliable word representations for low resource languages. Many languages, though widely spoken, are largely under-represented in this area of research. One of the main reasons for this is the lack of reliable evaluation metrics to compare between different approaches of building these embeddings.

Word similarity task is a widely used, computationally efficient method to directly evaluate the quality of word vectors. It relies on finding correlation between human assigned similarities between words, and those between corresponding word vectors. We release *word similarity datasets* for six low resource languages – Urdu, Telugu, Marathi, Punjabi, Tamil and Gujarati. For the construction of these datasets, our approach relies on translation and re-annotation of word similarity datasets of English. We also present baseline scores for word representation models using state-of-the-art techniques for Urdu, Telugu and Marathi by evaluating them on newly created word similarity datasets.

For linguistically similar languages, we show that it is possible to use the better trained word representations of the more resourceful language for the other language in the pair, using a projection learning approach which relies on a mapping between words having similar meaning from the two languages. This cross-lingual vector space transformation results in state of the art results on word similarity test sets of French and German - an increase of 13% in case of French and 19% for German, using English as the source language. We also go on to demonstrate that this approach is better suited for linguistically similar language pairs like Hindi-Urdu (where 60% words are simply transliterations of each other) than English-German or English-French. We go on to see how we modelled prefix-suffix based morphoology using a similar technique.

Contents

List of Figures

List of Tables

Chapter *1*

Introduction

Representing the meaning of words in a form that is understandable by computers is essential for many natural processing tasks. Word representations are means to mathematically represent the meaning of a word in a way that computers can understand. These representations are often in the form of vectors represented in a continuous vector-space of fixed dimensionality. When these word representations are in the form of word vectors, they are also called word embeddings.

Word lexicons are another form of word representations where the morpho-syntactic attributes (part-of-speech, tense, etc.) are stored. These contain rich information about the word meaning and also help us to capture the relationship between words. However, these word lexicons often require human annotation. In comparison to word lexicons, techniques used for building word embeddings can captures similar attributes of the words in an unsupervised way from a large monolingual corpus.

A lot of architectures have been proposed in the recent years for building vector space models from large datasets. Latest developments in building high dimensional word vectors rely on the distributional hypothesis: "*You shall know a word by the company it keeps.*" by John Rupert Firth (1957). This distributional information is captured by high dimensional vector representation of words which are learned automatically using co-occurrence statistics. The words that lie close together in this vector-space are supposed to be similar. Word representations are found to manifest semantic and syntactic regularities and have proved to be surprisingly good for a variety of natural language processing tasks. These regularities were observed by Mikolov et. al [38] and later further studied. The vectors of words with similar meanings were found to be close to each other. These regularities have been demonstrated to allow inferences of certain types (eg. king is to man what queen is to woman). These regularities were also found in morphological relations (eg. boys is to boy as women is to woman). In figure 1.1, we see by using vector arithmetic: "King + Man - Woman Queen".

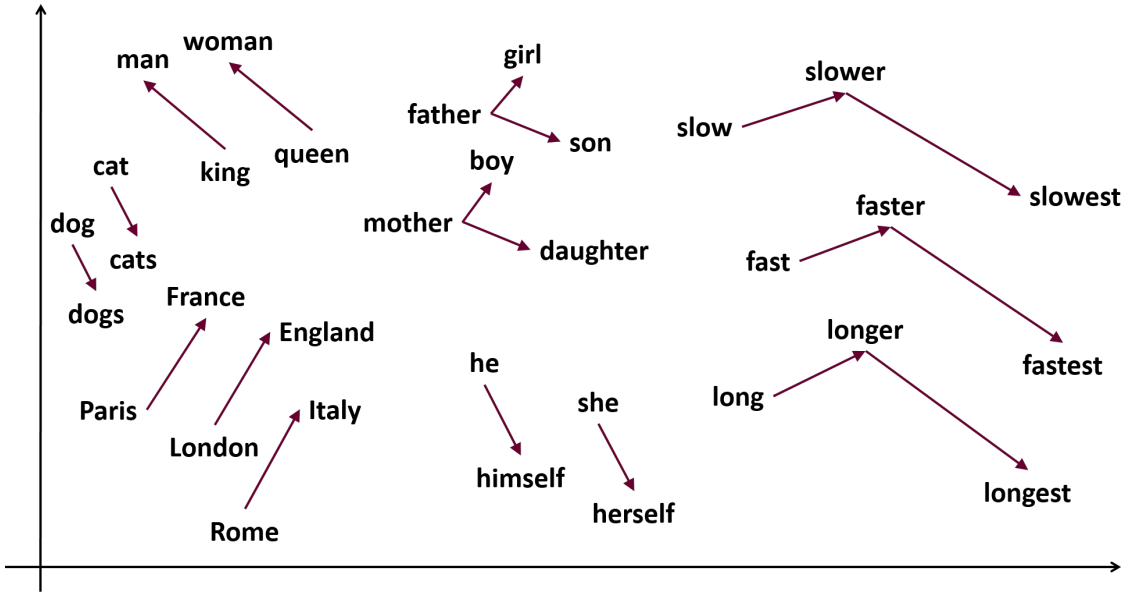


Figure 1.1 Abstract 2D Representation of Word Vectors

1.1 Thesis Statement

Since word vectors rely on multiple occurrences of a word for capture the semantic information of a word, they require a large amount of data for training. Hence, obtaining reliable word representations from comparatively small sized datasets is a difficult task. The main motivation behind this thesis is to allow for training more reliable word representation for low resource and morphologically rich languages. We present techniques for obtaining these representations and also publicly release testing datasets for six low resource languages for facilitating future research.

1.2 Contributions of the Thesis

The main contributions of the thesis are:

- We release Word Similarity datasets for evaluating the quality of word representations for six low resource languages.
- Achieved state of the art results on the word similarity test sets of French and German
- Large improvement in scores using our approach on a pair of similar languages (Hindi-Urdu).
- We show improved results on word similarity and word analogy test sets in English by transforming words between models trained on different corpus by different techniques.

1.3 Thesis Outline

This thesis consists of the following chapters:

- In chapter 2, we discuss the techniques commonly used for training and evaluation of word vector representations. It is to be noted that we would be using these techniques throughout the thesis.

- In chapter 3, we explain how we constructed word similarity for six language. We also present baseline systems using state of the art techniques evaluated using these newly created datasets.

- In chapter 4, we show how we exploited similarities in linguistically similar languages by transforming word embeddings of source language - which is high resource rich and hence well trained, to a corresponding model of target language - which is relatively resource deficient. This technique is similar to that used by Mikolov et al. [39] for machine translation. We further extend our approach by applying it to different models trained for one particular language. This enables us to incorporate the best of various models.

- In this section, we exploit the morphological regularities present in word representations to model prefix and suffix based morphology using a projection learning similar in intuition to that used in the previous section.

- In chapter 6, we summarize the contributions of the thesis and provide directions for future work.

Chapter *2*

Techniques for Training and Evaluation of Word Representations

2.1 Training

The first time this word 'Word Embeddings' was used in Bengio et al in 2003 [42], where he proposed this new model for distributed representations of words using neural networks. Then in 2008, a paper by Collobert and Weston basically made this approach mainstream by proposing an 'A unified architecture for natural language processing: Deep neural networks with multitask learning.' [34]. We describe in brief the various architectures used in this thesis. In 2013, Mikolov et. al [39] proposed two new log linear models for learning distributed representation of words. These models were optimized for training on very large datasets as compared to other models at that time. These models are credited with popularizing the use of word embedding and especially pre-trained word embeddings in many NLP tasks.

2.1.1 NNLM (Neural Network Language Models)

The probabilistic feedforward neural network language model proposed in [42] consists of four layers: input, projection, hidden and output layers. First, N previous words are encoded in a 1-of-V coding, where V is the size of the vocabulary, and fed to the input layer. This is then projected to the projection layer of size N $\times$ D in the form of a shared projection matrix. As only N inputs are active at any given time, composition of the projection layer is a relatively cheap operation. The NNLM architecture becomes complex for computation between the projection and the hidden layer, as values in the projection layer are dense. For a common choice of N = 10, the size of the projection layer (P) might be 500 to 2000, while the hidden layer size H is typically 500 to 1000 units. Moreover, the hidden layer is used to compute probability distribution over all the words in the vocabulary, resulting in an output layer with dimensionality V . Thus, the computational complexity per each training example is

$$Q = N \times D + N \times D \times H + H \times V$$

where the dominating term is H × V. With binary tree representations of the vocabulary, the number of output units that need to be evaluated can go down to around log2(V). Thus, most of the complexity is caused by the term N × D × H.

2.1.2 New Log-linear Models

Mikolov et. al [39] proposed two new model architectures for learning distributed representations of words that try to minimize computational complexity. The main observation from the previous section was that most of the complexity is caused by the non-linear hidden layer in the model. While this is what makes neural networks so attractive, we decided to explore simpler models that might not be able to represent the data as precisely as neural networks, but can possibly be trained on much more data efficiently.

These new architectures follow that neural network language model can be successfully trained in two steps: first, continuous word vectors are learned using simple model, and then the N-gram NNLM is trained on top of these distributed representations of words.

2.1.2.1 Continuous Bag-of-Words Model

This architecture proposed by Mikolov et. al [39] is similar to the feedforward Neural Network Language Model (NNLM) [42] as the hidden layer is removed and the projection layer is shared for all words (not just the projection matrix). So, all the words are projected into the same position (with their vectors averaged). Since the order of words does not matter, this architecture is called the bag of words model. This model relies on a window of context and tries to classify the middle word. One approach is to treat "He", "over", "and", "fell","down" as a context and from these words, be able to predict or generate the center word "tripped". This type of model we call a Continuous Bag of Words (CBOW) Model. It is to be noted that the weight matrix between the input and the projection layer is shared for all word positions in the same way as in the NNLM. Thus, the computational complexity is

$$Q = N \times D + D \times \log[2](V)$$

2.1.2.2 Continuous Skip-gram Model

The Skip-gram approach [39] is quite similar to CBOW model, the major difference being that instead of predicting the current word based on the context, we use each current word as

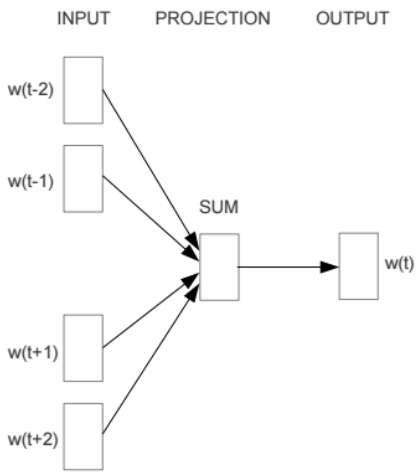


Figure 2.1 CBOW

input to the log-linear classifier with continuous projection layer, and predict words within a certain range before and after the current word. They found that increasing the range improves quality of the resulting word vectors, but it also increases the computational complexity. Any bag-of-words model assumes that we can learn what a word means by looking at the words that tend to appear near it. The CBOW model trains each word against its context. It asks "given this set of context words, what missing word is likely to also appear at the same time?" Skip-gram trains each the context against the word. It asks "given this single word, what are the other words that are likely to appear near it at the same time?" The computational complexity is

$$Q = C \times (D + D \times \log[2](V))$$

where C is the maximum distance of the words. Thus, if we choose C = 5, for each training word we will select randomly a number R in range $< 1; C >$, and then use R words from history and R words from the future of the current word as correct labels. This will require us to do R $\times$ 2 word classifications, with the current word as input, and each of the R + R words as output.

2.1.3 ConceptNet Ensemble Approach

Given a source of structured connections between words, Faruqui et.al [18] proposed a technique to combine embeddings learned from distributional semantics of unstructured text, known

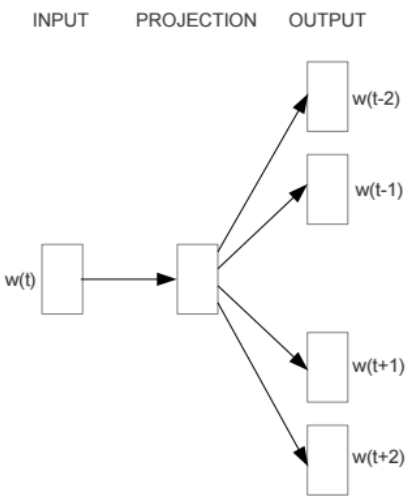


Figure 2.2 Skipgram

as "retrofitting". Speer and Chin [29] extended this technique to produce state of the art word embeddings for English. The method proposed by them resulted in a 16% increase on the Stanford English Rare-Word (RW) dataset [20]. ConceptNet Numberbatch consists of state-of-the-art semantic vectors (also known as word embeddings) that can be used directly as a representation of word meanings or as a starting point for further machine learning. It is built using an ensemble that combines data from ConceptNet, word2vec, GloVe, and (since 17.02) OpenSubtitles 2016, using a variation on retrofitting.

2.1.4 FastText

This approach [27] is based on the Skip-gram model, where each word is represented as a bag of character n-grams. A vector representation is associated with each character n-gram, words being represented as the sum of these representations. The method is fast and hence can be trained on large datasets. This method outperformed baselines which do not take into account subword information, on rare words, morphologically rich languages and small training datasets.

Another distinct feature of this approach is that it outputs not just a word level model but also a character level model alongside it. The length of character n-grams can be predefined while training. Building representations for rare and unknown words has gained a lot of traction in recent years. This model takes a very direct approach. First, it searches for the required word in the word level model. If its not found, then it proceeds to build the embedding of the required word using the character level model.

2.2 Evaluation

Since the concept of word meaning is abstract and not well defined, evaluation of word embeddings is a difficult task. Different forms of word representations capture different aspects of meaning. Similarity correlation measures like the word similarity and the word analogy tasks are often used as means of direct evaluation. The models used to obtain these representations optimize different loss functions, which makes it hard to compare the intrinsic quality of different forms of word representations. Here we discuss two similarity correlation measures used for evaluation of the quality of these word representations. These are the word similarity and the word analogy tasks.

2.2.1 Word Similarity Task

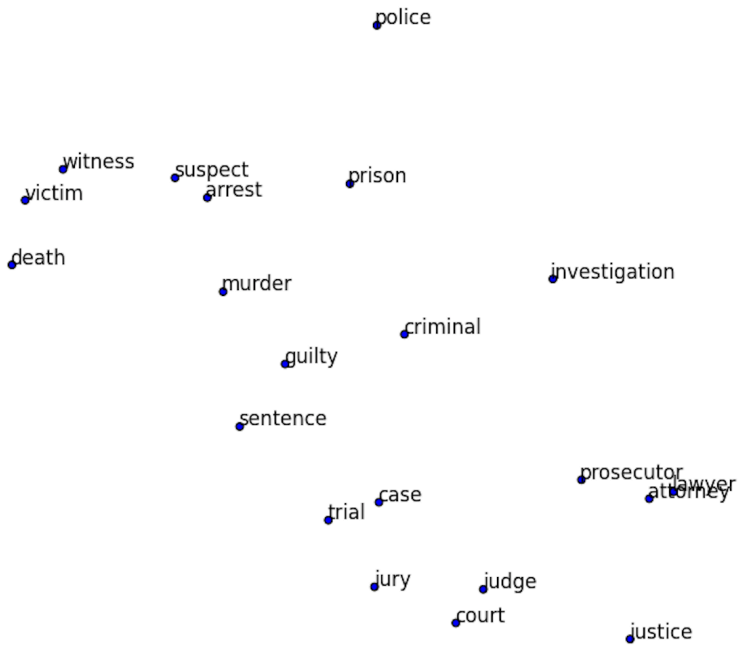


Figure 2.3 Abstract 2D representation of word vectors: Words having similar meaning lie close to each other (Intuition behind the word similarity task)

The word similarity constructed using human annotation measures how well does word similarity in the vector space correspond to the notion of word similarity according to humans. It relies on finding correlation between human assigned semantic similarity (between words) and corresponding word vectors. We have used Spearman's Rho for calculating correlation. A word similarity dataset consists of pairs of words which are presented to humans for annotation, where every annotation measures how similar the two words are as perceived by a human on a given scale. These annotations are then aggregated across all subjects to obtain an average

measure of similarity between the two words. Table 2.1 contains some examples from this dataset.

In chapter 3, we have described in detail how we created six new word similarity datasets. For English, we use Stanford English Rare-Word (RW) dataset [20], the WS353 [16] and the RG65 dataset [13]. The Stanford Rare-Word dataset contains comparatively more rare words and morphological complexity than other datasets and is central to our experiments. For German, we use the Gur350 and ZG222 datasets [40] and the German RG65 dataset. For French we use the French RG65 [6] dataset. For Spanish and Persian, we use Spanish-RG65 and Persian-RG65 test data-sets [15].

Word1	**Word2**	**Similarity**
love	sex	6.77
tiger	cat	7.35
tiger	tiger	10.00
book	paper	7.46
computer	keyboard	7.62
computer	internet	7.58
plane	car	5.77
train	car	6.31
telephone	communication	7.50
television	radio	6.77
media	radio	7.42

Table 2.1 Examples from WS word similarity dataset

2.2.2 Word Analogy Task

Word representations capture both syntactic and semantic properties [38] of natural language. They have been found to show analogical regularities of the form: "*a is to b as c is to ?*". The regularities are both semantic ("*England : London :: Germany : Berlin*") and syntatic ("*walk : walking :: talk : talking*"). The two most widely used word analogy datasets are: MSR word analogy datsaset and Google word analogy dataset. While MSR dataset contains only syntactic analogy questions, the dataset by Google contains both symantic and syntatic questions. For all the experiments, we have calculated the fraction of answers correctly answered by the system on MSR word analogy dataset.

The MSR dataset [38] contains 8000 analogy questions. This data set has been used by us for testing our model. The relations portrayed by these questions are morpho-syntactic, and can be categorized according to parts of speech - adjectives, nouns and verbs. Adjective relations

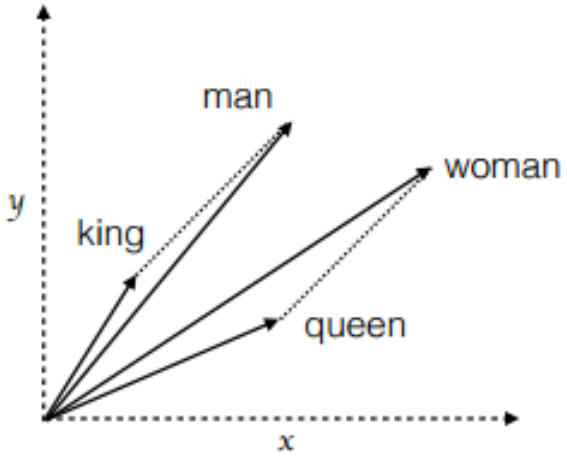


Figure 2.4 Abstract 2D representation of word vectors: Linguistic regularity shown between analogous word pairs. The dotted line connecting the words is the relation vector. So, "king+man-woman=queen" (Intuition behind the word analogy task)

include comparative and superlative (good is to best as smart is to smartest). Noun relations include singular and plural, possessive and non-possessive (dog is to dog's as cat is to cat's). Verb relations are tense modifications (work is to worked as accept is to accepted). Table 3.1 contains some examples from this dataset.

Word1	**Word2**	**Word3**	**Word4**
better	good	mightier	mighty
good	best	mighty	mightiest
best	good	mightiest	mighty
best	better	mightiest	mightier
better	best	mightier	mightiest
good	better	noisy	noisier
better	good	noisier	noisy
good	best	noisy	noisiest
best	good	noisiest	noisy
best	better	noisiest	noisier
better	best	noisier	noisiest
high	higher	bad	worse
higher	high	worse	bad
high	highest	bad	worst

Table 2.2 Examples from MSR word analogy dataset

Chapter *3*

Word Similarity Datasets: Annotation and Baseline Systems

3.1 Introduction

Word Similarity task is a computationally efficient method to evaluate the quality of word vectors. It relies on finding correlation between human assigned semantic similarity (between words) and corresponding word vectors. We have used Spearman's Rho for calculating correlation. Unfortunately, most of the word similarity tasks have been majorly limited to English language because of availability of well annotated different word similarity test datasets and large corpora for learning good word representations, where as for Indian languages like Marathi, Punjabi, Telugu etc - which even though are widely spoken by significant number of people, are still computationally resource poor languages. Even if there are models trained for these languages, word similarity datasets to test reliability of corresponding learned word representations do not exist.

Hence, primary motivation for creation of these six word similarity datasets has been to provide necessary evaluation resources for all the current and future work in field of word representations on these six languages - all ranked in top 25 most spoken languages in the world, since no prior word similarity datasets have been publicly made available.

The main contribution of this section is the set of newly created word similarity datasets which would allow for fast and efficient comparison between. Word similarity is one of the most important evaluation metric for word representations and hence as an evaluation metric, these datasets would promote development of better techniques that employ word representations for these languages. We also present baseline scores using state-of-the-art techniques which were evaluated using these datasets.

Multitude of word similarity datasets have been created for English, like WordSim-353 [16], MC-30 [12], Simlex-999 [11], RG-65 [13] etc. RG-65 is one of the oldest and most popular

datasets, being used as a standard benchmark for measuring reliability of word representations.

RG-65 has also acted as base for various other word similarity datasets created in different languages : French [6], German [40],Portuguese [30], Spanish and Farsi [5]. While German and Portuguese reported IAA (Inter Annotator Agreement) of 0.81 and 0.71 respectively, no IAA was calculated for French. For Spanish and Farsi, inter annotator agreement of 0.83 and 0.88 respectively was reported. Our datasets were created using RG-65 and WordSim-353 as base, and their respective IAA(s) are mentioned later in the paper.

3.2 Datasets

For all the models trained in this section, we have used the Skip-gram, CBOW [37] and FastText [27] algorithms. The dimensionality has been fixed at 300 with a minimum count of 5 along with negative sampling.

As training set of Marathi, we use the monolingual corpus created by IIT-Bombay. This data contains 27 million tokens. For Urdu, we use the untagged corpus released by Jawaid et. al. [4] containing 95 million tokens. For Telugu, we use Telugu wikidump available at "*https://archive.org/details/tewiki-20150305*" having 11 million tokens.

For testing, we use the newly created datasets. The word similarity datatsets for Urdu, Marathi, Telugu, Punjabi, Gujarati and Tamil contain 100, 104, 111, 143, 163 and 97 word pairs respectively.

For rest of the paper, we have calculated the Spearman ρ (multiplied by 100) between human assigned similarity and cosine similarity of our word embeddings for the word-pairs. For any word which was is not found, we assign it a zero vector.

In order to learn initial representations of the words, we train word embeddings (word2vec) using the parameters described above on the training set.

3.3 Methodology

3.3.1 Translation

English RG-65 and WordSim-353 were used as base for creating all of our six different word similarity datasets. Translation of English data set to target language (one of the six languages) was manually done by a set of three annotators who are native speakers of the target language and are fluent in English. Initially, translations are provided by two of them, and in case of disparity, third annotator was used as a tie breaker.

Finally, all three annotators reached a final set of translated word pairs in target language, ensuring that there were no repeated word pairs. This approach was followed by Camacho-Callados et al. [5] where they created word similarity datasets for Spanish and Farsi in a similar manner.

3.3.2 Scoring

For each of the six languages, 8 native speakers were asked to manually evaluate each word similarity data set individually. They were instructed to indicate, for each pair, their opinion of how similar in meaning the two words are on a scale of 0-10, with 10 for words that mean the same thing, and 0 for words that mean completely different things. The guidelines provided to the annotators were based on the SemEval task on Cross-Level Semantic Similarity [9], which provides clear indications in order to distinguish similarity and relatedness and that for the actual scores.

Word1	**Word2**	**Similarity**
ਫਿਲਮ	ਥੀਏਟਰ	8.5
ਪੈਸਾ	ਨਕਦ	9
ਸਜਾਵਟ	ਬਹਾਦਰੀ	2
ਗ੍ਰਹਿ	ਗਲੈਕਸੀ	6.5
ਊਰਜਾ	ਸੰਕਟ	3
ਮੈਰਾਥਨ	ਸਪ੍ਰਿੰਟ	8
ਸੰਭਾਲ	ਸੰਸਾਰ	4
ਫਿਜ਼ਿਕਸ	ਕੈਮਿਸਟਰੀ	5
ਮਨੋਵਿਗਿਆਨ	ਚਿੰਤਾ	7

Table 3.1 Examples from Punjabi word similarity dataset

The results were averaged over the 8 responses for each word similarity data set, and each data set saw good agreement amongst the evaluators, except for Tamil, which saw relatively weaker agreement with respect to other languages (see Table 3.2).

3.4 Evaluation

3.4.1 Inter Annotator Agreement (IAA)

The meaning of a sentence and its words can be interpreted in different ways by different readers. This subjectivity can also reflect in annotation of sentences of a language despite the annotation guidelines being well defined. Therefore, inter-annotator agreement is calculated to give a measure of how well the annotators can make the same annotation decision for a certain category.

Language	Inter Annotator Agreement
Urdu	0.887
Punjabi	0.821
Marathi	0.808
Tamil	0.756
Telugu	0.866
Gujarati	0.867

Table 3.2 Inter Annotator Agreement (Fleiss Kappa) scores for word similarity datasets created for six languages.

3.4.1.1 Fleiss' Kappa

Fleiss' kappa is a statistical measure for assessing the reliability of agreement between a fixed number of raters when assigning categorical ratings to a number of items or classifying items. This contrasts with other kappas such as Cohen's kappa, which only work when assessing the agreement between not more than two raters or the interrater reliability for one appraiser versus himself. The measure calculates the degree of agreement in classification over that which would be expected by chance [41].

We have calculated Fleiss' Kappa for all our word similarity datasets (see table 3.2).

3.5 Results and Analysis

System	Score	OOV	Vocab
CBOW	28.30	19	130K
SG	34.40	19	130K
FastText	34.61	19	130K
FastText w/ OOV	45.47	14	-

Table 3.3 Results for **Urdu**

System	Score	OOV	Vocab
CBOW	36.16	3	194K
SG	41.22	3	194K
FastText	33.68	3	194K
FastText w/ OOV	38.66	0	-

Table 3.4 Results for **Marathi**

System	Score	OOV	Vocab
CBOW	26.01	14	174K
SG	27.04	14	174K
FastText	34.29	14	174K
FastText w/ OOV	46.02	0	-

Table 3.5 Results for **Telugu**

We present baseline scores using state of the art techniques - CBOW and Skipgram [37] and FastText-SG [27], evaluated using our word similarity datasets in tables 3.3, 3.4 and 3.5. As we can see the models trained encountered unseen word pairs when evaluated on their corresponding word similarity datasets. This goes on to show that all word pairs in our word similarity sets are not too common, and contain word pairs with some rarity. The datasets released as a part of this thesis can be found at "*https://github.com/syedsarfarazakhtar/Word-Similarity-Datasets-for-Indian-Languages*"

Chapter 4

Transformation between Vector Space Models

4.1 Introduction

In this section, we exploit similarities in linguistically similar languages by transforming word embeddings of source language - which is highly resource rich and hence well trained, to a corresponding model of target language - which is relatively resource deficient. This technique is similar to that used by Mikolov et. al. [39] for machine translation. We further extend our approach by applying it to different models trained for one particular language. This enables us to incorporate the best of various models.

Given a source of structured connections between words, Faruqui et.al [18] proposed a technique to combine embeddings learned from distributional semantics of unstructured text, known as "retrofitting". Speer and Chin [29] extended this technique to produce state of the art word embeddings for English. The method proposed by them resulted in a 16% increase on the Stanford English Rare-Word (RW) dataset [20].

In this section, we propose a method to transform words from one vector space to another. We use this technique to transform word embeddings between languages and also within the same language. Some languages are richer in resources than others. For example, Hindi is richer in resources than Urdu. We also evaluate our approach on dissimilar language pairs like English - French and English - German. Our aim is to use resource rich techniques like ConceptNet Ensemble [29] for languages poor in resources, for which we need to learn a mapping between their vector spaces.

Using this technique, we are also able to get embeddings of words which are unknown for one embedding space by importing the transformed embedding of the same word from another model of the same language. This method proved to be faster than training a combined em-

bedding space from scratch, while giving high quality word embeddings.

The basis of our approach lies in having a sufficient number of frequent word pairs in both source and target languages to successfully train our transformation matrix. Each word pair is of form <word from source language, translated word of target language>. In this section, we present a method for transforming word representations which, for training, take word representations of these word pairs to create a single transformation matrix, which when applied on any word representation of source word, will give us word representation of corresponding word in target language. We emphasize on highly frequent words, because we believe that highly frequent words have better trained word embeddings and thus result in better results for our approach.

We rely on a bi-lingual dictionary of the language pair for training and evaluating our transformation matrix. For training our matrix, we generate a bi-lingual dictionary from parallel corpus of source and target language in an unsupervised way. But for evaluation, the bi-lingual dictionary was missing most of the word pairs present in our test dataset because the parallel corpus for all the language pairs that we worked on in this section were not large enough. Hence, only for evaluating our transformation matrix, we manually created a bilingual dictionary from the test sets - which was not used for training our transformation matrix because transformation matrix tends to completely remember the word representations it was generated from.

We show that our method performs well on both French and German with state of the art results on both languages. Since there is no prior work done on Urdu, we test our approach against baseline scores of SkipGram embeddings. Our approach showed improvement of 16% over baseline scores. We further test our approach on multiple models of English. Our evaluations show that this is a fast and efficient approach to transform word embeddings from one vector space to another.

4.2 Datasets

For some experiments, we are using word embeddings trained on Google News corpus [39]. For all the models trained in this section, we have used the Skip-gram [37] algorithm. The dimensionality has been fixed at 300 with a minimum count of 5 along with negative sampling.

As training set for English, we use the Wikipedia data [8] (SG/en-SG). Soricut and Och [33] and Luong et. al. [20] had used the same training corpus for their models. The corpus contains about 1 billion tokens. For German and French, we use News Crawl (Articles from 2010)

released as a part of ACL 2014 Ninth Workshop on Statistical Machine Translation (de-SG and fr-SG respectively). For Urdu, we use the untagged corpus released by Jawaid et. al. [4]. For Hindi, we use a monolingual corpus containing 31 million tokens, for training.

As a parallel corpus for Urdu, we use the Hindi-Urdu parallel corpus released by Durrani et. al. [22]. We have used the Europarl parallel corpus version 7 [26] for parallel sentences of English-French and English-German.

We use standard word-similarity datasets for testing. For English, we use Stanford English Rare-Word (RW) dataset [20] and the RG65 dataset [13]. The Stanford Rare-Word dataset contains comparatively more rare words and morphological complexity than other datasets and is central to our experiments. For German, we use the German RG65 [40] dataset. For French we use the French RG65 [6] dataset. In case of Urdu, we are using the word similarity dataset WS-UR-100 [36].

For testing analogical regularities, we have used the MSR word analogy dataset [38]. It contains 8000 analogy question. This dataset has been used by us for testing our model. The relations portrayed by these questions are morpho-syntactic, and can be categorized according to parts of speech - adjectives, nouns and verbs. Adjective relations include comparative and superlative (good is to best as smart is to smartest). Noun relations include singular and plural, possessive and non-possessive (dog is to dog's as cat is to cat's). Verb relations are tense modifications (work is to worked as accept is to accepted).

For rest of the paper, we have calculated the Spearman ρ (multiplied by 100) between human assigned similarity and cosine similarity of our word embeddings for the word-pairs. All the thresholds mentioned have been decided after empirical fine tuning. Even though our experiments were computationally optimized, time and space complexities also played a part in deciding our thresholds. In order to learn initial representations of the words, we train word embeddings (word2vec) using the parameters described above on the training set. This model is referred to as SG (Skip-gram).

4.3 Cross-Lingual Transformation Matrix

Since we are generating our transformation matrix from parallel word pairs of two different languages, we first need a list of highly frequent word pairs. For generating this list, we have parallel corpus of two different languages - in our case, Hindi and Urdu, and English and French/German. This parallel corpus contains aligned sentences, from which we generate a

one-to-one mapping between words.

First we obtain word alignments using fast align [7] which gives us many to many word mapping, which we further use to construct our confidence matrix. This confidence matrix is generated using the frequency count of each individual mapping. For each word of the source language in the confidence matrix, we find out the word in the target language that it has been matched with most frequently and its fraction among all matches. After this, we proceed in decreasing order of count of matches while building a one-to-one matching above a threshold of match count and fraction of matches (25 and 0.5 respectively).

For this word pair list, we have a frequency threshold, which decides the word pairs that will be chosen for generating our transformation matrix (see Table 5.1). For our experiments, this threshold has been set at 500 (to ensure that they are well trained).

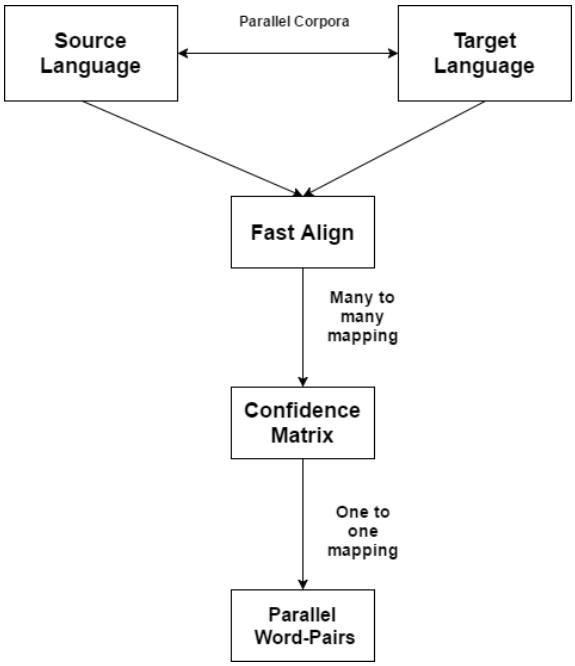


Figure 4.1 Generating Cross-Lingual Word Pairs

Figure 4.1 gives a high level overview of how we generate highly frequent word pairs from parallel corpus of source and target language in an unsupervised way.

Suppose we get "N" highly frequent word pairs. Dimensions of word embedding of a word in our model is "D". Using first word of our "N" chosen word pairs, we create a matrix "A" of dimensions N*D, where each row is vector representation of the first word. Similarly, we

Table 4.1 Count is the number of times English word was aligned with French word and "Fraction" is the fraction of the count over all the times English word was aligned with any word.

English Word	French Word	Count	Fraction
and	et	1.1M	0.87
of	de	1M	0.51
that	que	422K	0.51
we	nous	319K	0.65
not	pas	229K	0.55
We	Nous	108K	0.63
Mr	Monsieur	99K	0.5
-	-	98K	0.79

create another matrix B, of similar dimensions as A, using second word of our chosen word pairs.

We now propose that a matrix "X" (our transformation matrix) will exist such that A*X = B, i.e. X = A^{-1}*B. Our matrix "X" will be of dimensions "D*D" and when applied to a word embedding (matrix of dimensions 1*D, it gives a matrix of dimensions 1*D as output), it results in the word embedding of the transformed form of the word. Due to inverse property of a matrix, it accurately remembers the word pairs used for computing it. The matrix also appears to align itself with the word embedding of other words (not used for its training) to transform them according to the patterns that the matrix follows.

4.3.1 Transformation between Similar Language Pairs

We tried our approach on a very similar language pair - "Hindi-Urdu" with Hindi as the source language and Urdu as the target language. Durrnai et. al. [22] observed that 62% of the Hindi vocabulary are also a part of the Urdu vocabulary after transliteration. The approach smoothly maps embeddings between pairs of linguistically similar or derivative languages, as well as between languages of diverse linguistic properties. Similarity between languages further improves the performance as is evident from results shown in table 4.2. WS-UR-100 is the **Urdu** version of en-RG-65 and en-WS353 test sets.

4.3.2 Transformation between Diverse Language Pairs

Comparison between different vector space models of English is shown in 4.3. SO denotes the scores of Soricut and Och [33]. ConceptNet denotes the results of ensemble approach by

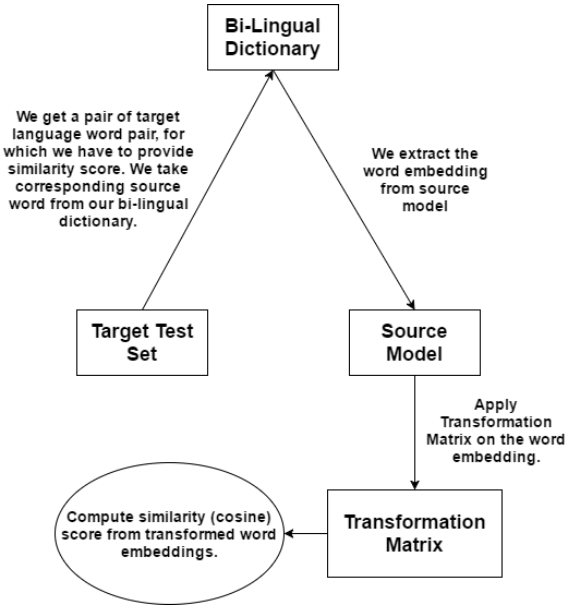


Figure 4.2 Computing similarity scores via transformation matrix.

Table 4.2 Note that hi-SG-T denotes the scores of transformed word embeddings of hi-SG.

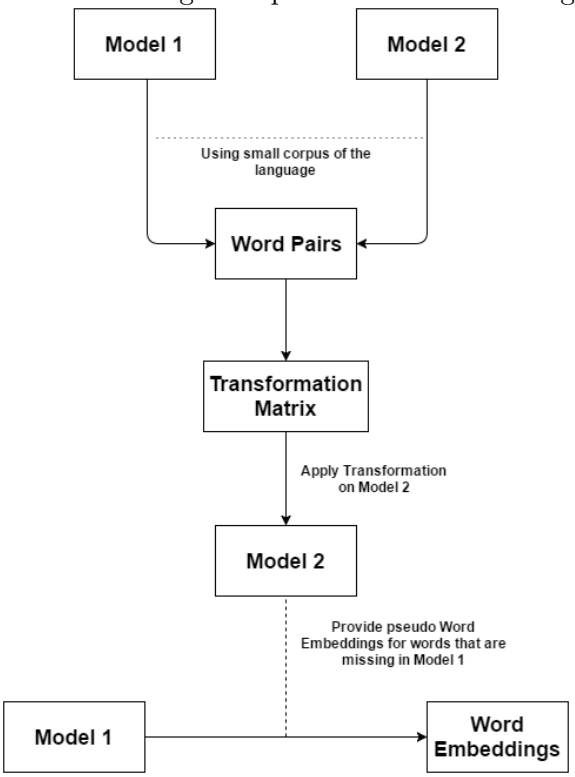
System	WS-UR-100	Vocab
ur-SG	34.50	130K
hi-SG-T	**50.08**	-

Speer and Chin [29] and en-SG denotes the scores of the model trained by us. This table is for comparison between en-SG and ConceptNet and how this difference is reflected when these models are transformed to other languages.

We have used English RG-65 test set as a translation table for French RG-65 and German RG-65 with minor corrections. It is to be noted that the French and German versions of RG-65 were constructed by translating and re-annotating English RG-65.

We use word similarity task as our evaluation criteria. Given a word pair (French or German), we find its translation in English using English RG-65 test as a translation table. We then find the corresponding word embedding of the word (English) in ConceptNet/SG, and apply our transformation matrix on the word embedding, to generate word embedding for the word in the vector space of target language (French or German) which is further used to generate word similarity score. The scores are shown in Tables 4.3, 4.4 and 4.5. de-RG-65 and fr-RG-65 are the German and French versions of en-RG-65 test set. Note that en-SG-T and ConceptNet-T denote the scores of transformed word embeddings of en-SG and ConceptNet.

Figure 4.3 Porting multiple models of same language.



We see that there is a proportional increase in the scores of the transformed embeddings of en-SG-T and ConceptNet-T.

We see that the approach performs better in the case of Hindi-Urdu as compared to English-German and English-French, owing to similarity between the two languages. Please note that the comparison is not made between ConceptNet-T and hi-SG-T but between en-SG-T (of both French and German) and hi-SG-T because anything of similar nature to ConceptNet does not exist for Hindi.

4.4 Transformation between Same Language Models

While evaluating and testing our approach, we realized that it might be possible to port and use multiple models as and when needed, for a single language. We often encounter models trained for different types of data, for different purposes, but for the same language. Our approach enables us to generate word embeddings for words that are missing in one model but are present in another model. This enables us to reduce number of unseen words encountered, and after careful evaluation, we found that this approach indeed helps us. We tested this approach

Table 4.3 This table denotes the results of various systems on en-RG-65 test set.

System	en-RG-65	Vocab
SO [33]	75.1	1.2M
SO w/ Morph [33]	75.1	1.2M
en-SG	74.12	2.4M
ConceptNet	**90.16**	0.4M

Table 4.4

System	fr-RG-65	Vocab
SO [33]	63.6	1.2M
SO w/ Morph [33]	67.3	1.2M
fr-SG	62.48	0.5M
en-SG-T	**68.62**	-
ConceptNet-T	**80.13**	-

using word similarity and word analogy tasks and it showed significant improvement in results.

For creating transformation matrix for two different models of same language, we initially require two different models trained on two different datasets - so that there are certain set of words which are not present in both the models. Then we take a small corpus, for determining the frequent words of the language (above a frequency threshold of 500 and also present in both the models). Using these frequent words, we create our transformation matrix, procedure for which is similar to what we did earlier in section 3.

After creating this matrix, now whenever we encounter a word which is not present in our first model, we look for the word in our second model, and if found, we apply our transformation matrix to its embedding in our second model. This results in a representation of the word, which proved to be good enough, when we ran word similarity and word analogy tasks on it. Figure 4.3 gives a high level overview of how we try to incorporate different models of the same language via transformation matrix. The scores are shown in tables 5.2 and 4.7. GN denotes the scores of embeddings trained in Google-News word embeddings. SG denotes the scores of the embeddings trained by us. GN+SG denotes the system in which we import the embeddings of any word missing in GN from SG.

Table 4.5 Results for **German**

System	de-RG-65	Vocab
SO [33]	62.4	2.9M
SO w/ Morph [33]	64.1	2.9M
de-SG	64.96	1.8M
en-SG-T	**67.3**	-
ConceptNet-T	**83.17**	-

Table 4.6 Scores on Stanford Rare word test set.

System	RW	Unseen Words
GN	45.27	173
SG	40.08	88
GN+SG	48.56	58

4.5 Results and Analysis

The method seems to perform well not only for the same language but also for cross lingual vector space transformation. Cross-lingual vector space transformation results in state of the art results on word similarity test sets of French and German - an increase of 13% in case of French and 19% for German (See tables 4.4 and 4.5).

We see in table 4.8 that for the language pair Hindi-Urdu, there was considerably greater increase than en-SG-T for both English-French and English-German. This may be because Hindi-Urdu are very much more similar than either English-French and English-German. "Increase on SG" denotes the difference between the mentioned systems and the scores of SkipGram embeddings trained on the target languages - fr-SG, de-SG and ur-SG respectively. We are not using ConceptNet-T for this comparison because any equivalent embedding is not available for Hindi.

For transformation within the same language we saw a significant improvement on both word similarity and word analogy test sets for English (See tables 5.2 and 4.7).

ConceptNet proved to be state of the art model for English, with significant improvement in quality of word embeddings. However, its not possible to apply techniques like ConceptNet on other languages because of their relative data scarcity. Our approach allows us to overcome this hurdle with encouraging results for languages that are linguistically similar to English.

Table 4.7 Scores on MSR word analogy test set.

System	CosSum	CosMul	Unseen Words
GN	0.646	0.67	2000
SG	0.484	0.533	0
GN+SG	**0.674**	**0.701**	0

Table 4.8 Inter-Language comparison of results.

System	Language Pair	Increase on SG
hi-SG-T	Hindi-Urdu	16.3
en-SG-T	English-French	6.14
en-SG-T	English-German	2.34

Chapter *5*

Projection Learning for Morphology

5.1 Introduction

In this section, we discuss a projection learning approach similar to the projection learning technique discussed in section 3 for building reliable word representations of words using their morphological variants. It is to be noted that though this approach can be directly used to solve the problem of rare and OOV words (and hence tackle data scarcity) in word similarity tasks by combining our approach with heuristic methods, but it would lead to the evaluation of our approach being diluted by it. This approach can instead be used to solve a broader variety of problems. In section 5, we saw that Soricut and Och [33] exploited the morphological regularities present in high dimensional word representations to generate prefix/suffix based morphological transformation rules in an unsupervised manner. These morphological transformations were represented as word pairs in the same embeddings space and they relied on vector arithmetic to calculate vectors for rare and unseen words. For example, for building the embedding for "regionalism", they might have evaluated the expression "provincialism - provincial + regional". We had used this idea for generating embeddings for rare and unseen words in section 5.

Here, we develop on their approach for generating rare and unseen words. In the above example (of "regionalism"), the choice of the word pair (here "provincialism-provincial") that acts as the transformation rule is very important. Suppose we want to generate an embedding for "talking" Experimental results showed that "walk - walks" gives better results than rules like "invent - invents" or "object - objects" in generating word embedding for "runs".

Hence, we present a global morphological transformation operator, which aligns itself with the source word, to give an embedding for target word. We will have a transformation operator for each rule, irrespective of the form of root word (like verb or a noun). Our transformation operator is in the form of a matrix, which when applied on a word embedding (cross product

with the word representation) gives us a word embedding for target word. The main idea is not to solve for “invent is to invents as run is to ?” or ”object is to objects as run is to ?”, but instead solve for “walk is to walks, object is to objects, invent is to invents, ... as run is to ?”.

Note that this research work was a joint effort with Arihant Gupta (LTRC-IIITH).

5.1.1 Datasets

We are using word embeddings trained on Google News corpus [39] for our experiments. For the model trained in this paper, we have used the Skip-gram [37] algorithm. The dimensionality has been fixed at 300 with a minimum count of 5 along with negative sampling.

As training set and for estimating the frequencies of words, we use the Wikipedia data [8]. The corpus contains about 1 billion tokens.

The MSR dataset [38] contains 8000 analogy questions. This data set has been used by us for testing our model. The relations portrayed by these questions are morpho-syntactic, and can be categorized according to parts of speech - adjectives, nouns and verbs. Adjective relations include comparative and superlative (good is to best as smart is to smartest). Noun relations include singular and plural, possessive and non-possessive (dog is to dog’s as cat is to cat’s). Verb relations are tense modifications (work is to worked as accept is to accepted).

For all the experiments, we have calculated the fraction of answers correctly answered by the system on MSR word analogy dataset.

5.1.2 Morphological Transformation Matrix

To compute the transformation matrix of a rule, we first extract in an unsupervised way all the word pairs following that transition rule. For example, in case of the rule <null,s>, we find word pairs such as <boy,boys>, <object,objects> and <invent,invents>. In this paper, we used the data structure TRIE for computational optimization. However, we don’t want cases which do not follow the general regularity of a rule. One such case may be <hat,hated> which does not follow the general trend of the transformation <null,ed>. For eliminating these cases, we set a threshold of cosine similarity of the word vectors of the two words of the pair at 0.2. Also, the frequency of both these words should be greater than 1000 (so that they are well trained). Since our transformation matrix is derived from all the word pairs following a particular transition rule, we carefully use only those word pairs which are of high frequency.

We do so because highly frequent words have better trained word embeddings.

Suppose we get "N" highly frequent word pairs following the same regularity(transition rule). For our experiments, the lower threshold of "N" is set at 50. Dimensions of word embedding of a word in our model is "D". Using first word of our "N" chosen word pairs, we create a matrix "A" of dimensions N*D, where each row is vector representation of a word. Similarly, we create another matrix B, of similar dimensions as A, using second word of our chosen word pairs.

We now propose that a matrix "X" (our transformation matrix) exists such that A*X = B, i.e. $X = A^{-1}*B$ (all instances of A that we encountered were non-singular). Our matrix "X" will be of dimensions "D*D" and when applied to a word embedding (matrix of dimensions 1*D, it gives a matrix of dimensions 1*D as output), it results in the word embedding of the transformed form of the word. Due to inverse property of a matrix, it accurately remembers the word pairs used for computing. The matrix also appears to align itself with the word embedding of other words (not used for its training) to transform them according to the rule that the matrix follows. Some interesting results are shown in table 5.1.

Word1	**Word2**	**Word3**	**Operator**	**Word4**	**Cosine**
joined	joins	became	<ed , s>	becomes	0.68
decides	decided	studies	<s , d>	studied	0.89
learn	learned	build	<null , ed>	built	0.80
support	supported	see	<null , ed>	saw	0.72
reach	reached	go	<null , ed>	went	0.80
reach	reaches	go	<null , es>	goes	1.0
member	members	school	<null , s>	schools	0.88
ask	asks	reduce	<null , s>	reduces	0.91
resident	residents	rate	<null , s>	rates	0.86
recognize	recognizes	be	<null , s>	is	0.70
get	gets	show	<null , s>	shows	0.83
turned	turns	said	<ed , s>	says	0.74
higher	highest	stricter	<r , st>	strictest	1.0
wild	wilder	harsh	<null , er>	harsher	0.91

Table 5.1 Some example results of transformation operators.

While testing, we extract the syntactic transition using the first two words of the analogy question. For example, for pairs like <reach, reached>, < walk, walked>, we are able to extract that they follow <null, ed> rule syntactically. But, for <go, went>, we are not able to find any transformation operator after syntactic analysis, and for such cases, we fall back on CosSum/CosMul [23] approaches as our backup. Mikolov et al. showed that relations between words are reflected to a large extent in the offsets between their vector embeddings (queen - king = woman - man), and thus the vector of the hidden word b^* will be similar to the vector $b-a+a^*$.

Levy et. al. [23] proposed the systems CosSum and CosMul in which they showed that tuning the hyperparameters has a significant impact on the performance. Hyperparameters are all the modifications and system design choices which are a part of the final algorithm.

Even though our transformation operator can handle any sort of transformation, but if we are not able to detect the rule syntactically, we are not able to determine which transformation operator to use, and hence, we fall back on CosSum/CosMul. Like for the above mentioned examples, we will use transformation operator (if existing) for transformations like <reach, reached>, since we can find the rule syntactically, but for <go, went>, we can not, since we can not extract the corresponding rule itself - even if the matrix can handle such transitions.

If a transformation matrix exists for a transition rule, we apply the corresponding transformation matrix on the word embedding of the third word and search the whole vocabulary for the word with an embedding most similar to the transformed embedding (ignoring the third word itself). If the similarity of the resultant word's embedding with our transformed embedding is less than 0.68 (determined empirically) or the transformation matrix itself does not exist, we fall back on the CosSum/CosMul techniques.

5.1.3 Result and Analysis

Model	**CosSum**	**CosSum w/ M**	**CosMul**	**CosMul w/ M**
SGNS-L	0.69	-	**0.729**	-
Glove-L	0.628	-	0.685	-
SG	0.269	0.554	0.282	0.566
GN	0.646	0.718	0.67	**0.733**
GN-SG Hybrid	0.674	0.835	0.698	**0.85**

Table 5.2 Scores on MSR word analogy test set.

In table 5.2, GN denotes the scores of Google-News word embeddings on the test set. SGNS-L and Glove-L [23] denote the results of Skip-gram with negative sampling and Glove word embeddings respectively, both trained on large datasets. SG denotes the scores of our word2vec trained model (on 1B tokens). "w/ M" implies that we have used matrix arithmetic (along with CosSum/CosMul as backup) for word analogy answering questions. Our model uses "CosSum" and "CosMul" as backup transformation method in case a transformation operator (matrix) does not exist. We see that the results of GN+Matrix are better than the previously used models.

However, one thing we noticed was that the model trained on Google-News did not contain words with apostrophe sign(s) and 1000 out of 8000 words in MSR word analogy test set contained apostrophe sign(s). Also, we noticed that in SG, the matrix approach was able to answer word analogy queries where words contained apostrophe sign(s), with an accuracy of 93.7% since it is a very common transformation - which resulted in well trained transformation matrix. So, we used SG as a backup for words which were not found in GN. The results of this hybrid model are denoted by GN-SG Hybrid. We see that this model performs considerably better than the existing state of the art system.

In our current approach, for problem "If A is to B, then C is to ?", we do syntactic analysis on "A" and "B" to find out the transformation rule (and hence the transformation matrix) to be applied on "C" to find our "?". Rather than doing syntactic analysis, we will focus on finding out the correct transformation matrix by applying all transformation matrices on "A" and then analyzing output of each matrix. The one which will give closest result to "B" can be safely assumed to be the correct rule (transformation matrix), and hence will be applied on "C" to find "?". We will also analyze the impact of this approach in terms of space and time complexities with respect to our current system. This will also enable us to find transformation matrix for word pairs which do not follow a syntactic transformation.

5.2 Discussion

Carrying the idea forward and with the same intuition as Chapter 3 in mind, we proposed an approach which applied projection learning to these morphological regularities, we get an approach which provided a large improvement in the scores (12% as compared to the previous state of the art system) on MSR word analogy dataset. The idea of projection learning has been applied to a multitude of tasks such as in the learning of cross lingual mappings for translation of English to Mandarin (Mikolov et. al. 2013 [39]). Our approach has its basis on the same lines but with a different formulation and end goal to learn morphological rules rather than semantic

associations and translational constraints (as done by Mikolov). We realize that the robustness will be validated further by combining our approach with heuristic methods to handle tasks such as word similarity as well. We didn't want to dilute our approach by handling multiple tasks and hence chose to limit our approach to the treatment of word analogy task only. Our approach works by statistically creating global transformation operators and is agnostic in applying them (i.e. applied on a verb or a noun). Our transformation rules learn from both noun transitions and verb transitions and hence, even though we agree that linguistically there is a difference between noun and verb transitions, our approach performed better than previously existing systems. Currently the system does not handle symmetry in rules (i.e <ed, s> and <s, ed> are treated as separate rules). A hierarchical approach can be made to merge rules based on symmetry. The approach is notationally the same for both suffix and prefix rules(i.e. the approach treats both in the same way).

Chapter *6*

Conclusions

This chapter summarizes the thesis, discusses its findings, shortcomings and contributions and also outlines directions for future research.

In section 3, we release word similarity datasets for six languages, namely Urdu, Telugu, Punjabi, Marathi, Gujarati and Tamil. We also present baseline scores using state of the art techniques, evaluated using our word similarity datasets for Telugu, Urdu and Marathi. We see that FastText w/ OOV (Out of Vocabulary) performed better than FastText in all the experiments, because character based models perform better than rest of the models since they are able to handle unseen words by generating word embeddings for missing words via character model. There are a lot of languages that are still computationally resource poor even though they are widely spoken by significant number of people. Our work is a small step towards generating resources to further the research involving word representations on these languages.

In section 4, we significantly improve on baseline scores of Urdu and beat previous state of the art systems for German and French, and also improve scores by porting two models to generate word embeddings for missing words. We could further extend this approach by creating transformation matrix for not only two, but for all possible combinations of models available. This way, when ever a particular model is being used for a particular task, we can look for a missing word in other models and use its transformed representation accordingly. Significant improvement in training word embeddings by mapping from a source language to a linguistically similar target language, gives us the hypothesis that similar improvement can be achieved for training between different dialects of the same language and we would like to test the approach further on such pairs. What we could also do is choosing best available word representation of a word in two or more models. For example, even though we have a word present in our model, but its representation is not reliable because of its low frequency in the corpus it was trained on or we are able to detect somehow that it is not well trained. We could then use a more reliable word embedding from other models, transform it for our model, and

then use it. We would still need to run various evaluation tasks on this approach to see its impact and usage in future. We will also try to see if it can be modified to retain its character in one model, and import characteristics from its other word representations in other models, which are relatively more reliable. We could define a heuristic for the same, and evaluate on various different values of parameters in the heuristics.

The main application of the approach discussed in section 5 lies in its ability to generate representations for unseen/unreliable words on the go. If we encounter a word such as "preparedness" for which we do not have a representation or our representation is not reliable, we can identify any reliable form of the word, say "prepared" and apply <null,ness> operator on it, resulting in a representation for "preparedness". In a similar case, we can generate embeddings for words such as "unpreparedness" from "prepared" by sequentially applying <null,ness> and a prefix operator trained in a similar manner - <null,un>. Overall, this results in a much larger vocabulary than of the model initially being used. In our current approach, for problem "If A is to B, then C is to ?", we do syntactic analysis on "A" and "B" to find out the transformation rule (and hence the transformation matrix) to be applied on "C" to find our "?". Rather than doing syntactic analysis, we will focus on finding out the correct transformation matrix by applying all transformation matrices on "A" and then analyzing output of each matrix. The one which will give closest result to "B" can be safely assumed to be the correct rule (transformation matrix), and hence will be applied on "C" to find "?". We will also analyze this approach's impact in terms of space and time complexities with respect to our current system. This will also enable us to find transformation matrix for word pairs which do not follow a syntactic transformation.

Chapter 7

Related Publications

Syed Sarfaraz Akhtar, Arihant Gupta, Avijit Vajpayee, Arjit Srivastava, Manish Shrivastava 2017. *Word Similarity Datasets for Indian Languages: Annotation and Baseline Systems.* Proceedings of the 11th Linguistic Annotation Workshop, pages 91-94.

Syed Sarfaraz Akhtar, Arihant Gupta, Avijit Vajpayee, Arjit Srivastava, Manish Shrivastava 2017. *An Unsupervised Approach for Mapping between Vector Spaces.* CICLing 2017.

Syed Sarfaraz Akhtar, Arihant Gupta, Avijit Vajpayee, Arjit Srivastava, Madan Gopal Jhanwar, Manish Shrivastava 2017. *Exploiting Morphological Regularities in Distributional Word Representations.* EMNLP 2017.

Bibliography

[1] Alon Lavie, and Michael J. Denkowski.: The METEOR metric for automatic evaluation of machine translation. Machine translation 23, no. 2-3 (2009): 105-115.

[2] Arihant Gupta, Syed Sarfaraz Akhtar, Avijit Vajpayee, Arjit Srivastava, Madan Gopal Jhanwar, Manish Shrivastava 2017. *Exploiting Morphological Regularities in Distributional Word Representations.* Proceedings of EMNLP 2017.

[3] Arihant Gupta, Syed Sarfaraz Akhtar, Avijit Vajpayee, Arjit Srivastava, Manish Shrivastava 2017. *Unsupervised Morphological Expansion of Small Datasets for better Word Representations.* Proceedings of CICLing 2017.

[4] Bushra Jawaid, Amir Kamran, and Ondrej Bojar.: A Tagged Corpus and a Tagger for Urdu. LREC 2014.

[5] Camacho-Collados, José, Mohammad Taher Pilehvar, and Roberto Navigli.: A Framework for the Construction of Monolingual and Cross-lingual Word Similarity Datasets. In ACL (2) (pp. 1-7).

[6] Colette Joubarne and Diana Inkpen.: Comparison of semantic similarity for different languages using the Google n-gram corpus and second-order co-occurrence measure. In Advances in Artificial Intelligence - 24th Canadian Conference on Artificial Intelligence, pages 216-221 (2011).

[7] Chris Dyer, Victor Chahuneau, and Noah A. Smith.: A Simple, Fast, and Effective Reparameterization of IBM Model 2. In proceedings of NAACL.

[8] Cyrus Shaoul and Chris Westbury.: The Westbury lab Wikipedia corpus. In Edmonton, AB: University of Alberta.

[9] David Jurgens, Mohammad Taher Pilehvar, and Roberto Navigli.: Semeval-2014 task 3: Cross-level semantic similarity.: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), in conjunction with COLING. 2014.

[10] Diana McCarthy, and Roberto Navigli.: The English lexical substitution task. Language resources and evaluation 43, no. 2 (2009): 139-159.

[11] Felix Hill, Roi Reichart, and Anna Korhonen.: Simlex-999: Evaluating semantic models with (genuine) similarity estimation. Computational Linguistics (2016).

[12] George A. Miller, and Walter G. Charles.: Contextual correlates of semantic similarity. Language and cognitive processes 6.1 (1991): 1-28.

[13] Herbert Rubenstein and John B. Goodenough.: Contextual correlates of synonym. Communications of the ACM, volume 8, number 10, pages 627-633 (2006).

[14] Jeffrey Pennington, Richard Socher and Christopher D. Manning.: GloVe: Global Vectors for Word Representation. In EMNLP, volume 14, pages 1532-43.

[15] José Camacho-Collados, Mohammad Taher Pilehvar and Roberto Navigli.: A Framework for the Construction of Monolingual and Cross-lingual Word Similarity Datasets. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015), Beijing, China, July 27-29, 2015.

[16] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman and Eytan Ruppin.: Placing search in context: The concept revisited. Proceedings of the 10th international conference on World Wide Web, pages 406-414 (2002).

[17] Luong, Minh-Thang, and Christopher D. Manning.: Achieving open vocabulary neural machine translation with hybrid word-character models. arXiv preprint arXiv:1604.00788.

[18] Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy.: Retrofitting word vectors to semantic lexicons. In proceedings of NAACL.

[19] Michael Mohler, Razvan Bunescu, and Rada Mihalcea.: Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1 (pp. 752-762). Association for Computational Linguistics (2011).

[20] Minh-Thang Luong, Richard Socher and Christopher D. Manning.: Minh-Thang Luong, Richard Socher and Christopher D. Manning. In CoNLL. Pages 104-113.

[21] Mohit Bansal, Kevin Gimpel, and Karen Livescu.: Tailoring continuous word representations for dependency parsing. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, Baltimore, MD, USA, Volume 2: Short Papers, pages 809–815 (2014).

[22] Nadir Durrani, Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2010. *Hindi-to-Urdu machine translation through transliteration.* in Proceedings of the 48th Annual meeting of the Association for Computational Linguistics, pp. 465-474. Association for Computational Linguistics, 2010.

[23] Omer Levy, Yoav Goldberg, and Ido Dagan. 2002. *Improving distributional similarity with lessons learned from word embeddings.* Transactions of the Association for Computational Linguistics 3 (2015): 211-225

[24] Omer Levy, Yoav Goldberg, and Ramat-Gan. 2002. *Linguistic Regularities in Sparse and Explicit Word Representations* In CoNLL, pp. 171-180. 2014.

[25] Ondřej Bojar, Vojtěch Diatka and Rychlý, Pavel and Straňák, Pavel and Suchomel, Vít and Tamchyna, Aleš and Zeman, Daniel 2014. *HindMonoCorp 0.5.* LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University in Prague.

[26] Philipp Koehn. 2002. *Europarl: A multilingual corpus for evaluation of machine translation.* Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2005.

[27] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016).

[28] Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng.: Parsing With Compositional Vector Grammars. In ACL, pages 455-465 (2013).

[29] Robert Speer and Joshua Chin. 2016. *Ensemble Method to Produce High-Quality Word Embeddings.* arXiv preprint arXiv:1604.01692 (2016).

[30] Roger Granada, Cassia Trojahn, and Renata Vieira.: Comparing semantic relatedness between word pairs in Portuguese using Wikipedia. International Conference on Computational Processing of the Portuguese Language. Springer International Publishing, 2014.

[31] Scott Miller, Jethran Guinness, and Alex Zamanian.: Name tagging with word clusters and discriminative training. In Proceedings of HLT-NAACL, volume 4, pages 337-342 (2004).

[32] Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen.: SenseRelate:: TargetWord: a generalized framework for word sense disambiguation. Proceedings of the ACL 2005 on Interactive poster and demonstration sessions (pp. 73-76). Association for Computational Linguistics (2005).

[33] Radu Soricut and Franz Och. 2015. *Unsupervised Morphology Induction using Word Embeddings.* In Proceedings of NAACL.

[34] Ronan Collobert and Jason Weston.: A unified architecture for natural language processing: Deep neural networks with multitask learning. Proceedings of the 25th international conference on Machine learning. ACM, 2008.

[35] Syed Sarfaraz Akhtar, Arihant Gupta, Avijit Vajpayee, Arjit Srivastava, Manish Shrivastava 2017. *Unsupervised Vector Space Transformation.* Proceedings of CICLing 2017.

[36] Syed Sarfaraz Akhtar, Arihant Gupta, Avijit Vajpayee, Arjit Srivastava, Manish Shrivastava 2017. *Word Similarity Datasets for Indian Languages: Annotation and Baseline Systems.* Proceedings of the 11th Linguistic Annotation Workshop, pages 91-94.

[37] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeff Dean.: Efficient estimation of word representations in vector space. In arXiv preprint arXiv:1301.3781 (2013).

[38] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig.: Linguistic regularities in continuous space word representations. In Proceedings of HLT-NAACL, volume 13, pages 746-751 (2013).

[39] Tomas Mikolov, Quoc V. Le, and lya Sutskever.: Exploiting similarities among languages for machine translation. arXiv preprint arXiv:1309.4168 (2013).

[40] Torsten Zesch and Iryna Gurevyc.: Automatically creating datasets for measures of semantic relatedness. In Proceedings of the Workshop on Linguistic Distances, pages 16-24 (2006).

[41] Wikipedia contributors.: "Fleiss' kappa." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 6 Feb. 2017. Web. 16 Feb. 2017.

[42] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin.: A neural probabilistic language model. Journal of machine learning research 3, no. Feb (2003): 1137-1155.