TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

THESIS NO: 078MSDSA005

Interpreting Pre-trained Word Embeddings using Parts of Speech Metarepresentations

by

Biraj Silwal

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER ENGINEERING SPECIALIZATION IN DATA SCIENCE AND ANALYTICS

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

LALITPUR, NEPAL

June, 2024

Interpreting Pre-trained Word Embeddings using Parts of Speech Metarepresentations

by

Biraj Silwal

078MSDSA005

Thesis Supervisor

Prof. Dr. Shashidhar Ram Joshi

Sanjivan Satyal

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Engineering Specialization in Data Science and Analytics

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk Campus

Tribhuvan University

Lalitpur, Nepal

June, 2024

COPYRIGHT©

DECLARATION

I declare that the thesis hereby submitted for **"Master of Science in Computer Engineering Specialization in Data Science and Analytics"** at IOE, Pulchowk Campus entitled **"Interpreting Pre-trained Word Embeddings using Parts of Speech Metarepresentations"** is my own work and has not been previously submitted by me at any university for any academic award.

I authorize IOE, Pulchowk Campus to lend this thesis to other institution or individuals for the purpose of scholarly research.

Biraj Silwal

078MSDSA005

June, 2024

RECOMMENDATION

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled **"Interpreting Pre-trained Word Embeddings using Parts of Speech Metarepresentations",** submitted by **Biraj Silwal** in partial fulfillment of the requirement for the award of the degree of **"Master of Science in Computer Engineering Specialization in Data Science and Analytics"**.

.....

Supervisor: Professor Dr. Shashidhar Ram Joshi

Dean of Engineering

Institute of Engineering, Tribhuvan University.

.....

Supervisor: Sanjivan Satyal

Assistant Professor

Institute of Engineering, Tribhuvan University.

.....

External Examiner: Dr. Bal Krishna Bal

Associate Professor

Kathmandu University

.....

Committee Chairperson: Asst. Prof. Dr. Aman Shakya

Program Coordinator, MSDSA

Department of Electronics and Computer Engineering

Institute of Engineering, Tribhuvan University.

Date: June, 2024

DEPARTMENTAL ACCEPTANCE

The thesis entitled "**Interpreting Pre-trained Word Embeddings using Parts of Speech Metarepresentations**", submitted by **Biraj Silwal** in partial fulfillment of the requirement for the award of the degree of **"Master of Science in Computer Engineering Specialization in Data Science and Analytics"** has been accepted as a bonafide record of work independently carried out by him in the department.

.....

Assoc. Prof. Jyoti Tandukar
Head of the Department
Department of Electronics and Computer Engineering,
Pulchowk Campus,
Institute of Engineering,
Tribhuvan University,
Nepal.

ACKNOWLEDGEMENT

I would like to thank my parents Balram Silwal and Bijaya Devi Bhatta and my sister Pragya Silwal for their continuous support and motivation during the progress of this thesis.

I would like to express my sincere gratitude to my supervisors Prof. Dr. Shashidhar Ram Joshi and Mr. Sanjivan Satyal for their valuable guidance in this thesis. I would also like to thank the entire department of Electronics and Computer Engineering, Pulchowk Campus for their unwavering support and inspiration. Finally, I would like to thank my classmates for their views and ideas on this thesis.

Sincerely,
Biraj Silwal
078MSDSA005

ABSTRACT

The word embeddings currently used are dense and uninterpretable, leading to interpretations that themselves are relative, overcomplete, and hard to interpret. This thesis has proposed a method that transforms these word embeddings into reduced syntactic representations. The resulting representations were compact and interpretable allowing better visualization and comparison of the word vectors and it was successively demonstrated that the drawn interpretations were in line with human judgment. The syntactic representations were then used to create hierarchical word vectors using an incremental learning approach similar to the hierarchical aspect of human learning. As these representations were drawn from pre-trained vectors, the generation process and learning approach are computationally efficient. Most importantly, it was find out that syntactic representations provide a plausible interpretation of the vectors and subsequent hierarchical vectors outperform the original vectors in benchmark tests.

Keywords: Word Embeddings, Interpretability, Syntactic Representations, Hierarchical Vectors.

Contents

List of Figures

List of Tables

Chapter 1: Overview

1.1 Introduction

The recent advancement in computation technologies have yielded state-of-the-art performances in multiple Natural Language Processing(NLP) tasks such as Text Classification, Named-Entity Recognition, Question Answering and Sentiment Analysis. The various techniques used to resolve these problems revolve around word representation and the concept of representing human understanding of the language in a machine recognizable form. The increasing use of this technique in the past decade has been accompanied by increasing concerns about lack of interpretability.[1]

Word Embedding is one of the most widely used technique for word representation. In natural language processing (NLP), word embedding is a term used for the representation of words for text analysis, typically in the form of a real-valued vector that encodes the meaning of the word such that the words that are closer in the vector space are expected to be similar in meaning.[2] Word Embeddings can generally be divided into two categories: Context-Independent embeddings such as GloVe[3], Word2Vec[4], and fastText, and Context-Dependent embeddings such as BERT (Bidirectional Encoder Representations from Transformers) and ELMo (Embeddings from Language Models)[5]. Context-Independent word embeddings are generated for words as a function whereas Context-Dependent word embeddings are generated for words as a function of the sentence it occurs in.

Context-Independent word embeddings have the major drawback of conflating words with various meanings into a single representation.[6] Context-Dependent word embeddings, on the other hand, better capture the multi-sense nature of words as they are at the token level and each occurrence of a word has its own embedding. To capture these contextualized representations, BERT, a popular Transform architecture model, uses a transformer that has been trained on tasks like Next Sentence Prediction and Masked Language Modeling.

Even though the introduction of Context-Dependent word embeddings has brought a drastic change to the NLP core and NLP downstream tasks, the issue of interpretability still persists. Additionally, it can be said that the use of 768-dimensional context-dependent embeddings, as in the BERT-Base model, instead of the 300-dimensional context-independent embeddings, as in the GLoVe model, has added a degree of complexity to the issue of interpretability. With the interpretability of NLP approaches having been deemed integral for error detection, research validation and legal reasons such as General Data Protection Regualtion (GDPR [7]) which has established a "right to explanation", it is crucial that the next generation of NLP techniques

adapt to adding interpretability in addition to yeilding state-of-the-art results.

Adding interpretability to word embeddings is the process of making the underlying meaning of words in word embeddings more understandable and meaningful. The process of adding interpretability encompasses various techniques, which are used to uncover the connections between words and offer insights into the underlying patterns in the embeddings. Not only do these techniques help provide insights into the embeddings but they are often integral to developing a deeper understanding of the performance of these embeddings in various downstream NLP tasks.

In order to add interpretability, tools like visualizations, dimensionality reduction strategies, or the extraction of significant characteristics from the embeddings have been used previously. Many of these techniques have developed independent ideas, which in some manner are able to explain the relevant pattern in the word embeddings. Additionally, these embeddings have also been able to provide metrics for analyzing the words visually making it possible to distinguish the differences between the words. However, most of these techniques are unable to provide an absolute metric to distinguish between words.

These word embeddings comprise of coordinates which by themselves have no meaningful interpretation to humans, for instance, it is impossible to distinguish what a "high" value along a certain coordinate signifies compared to a "low" value along the same direction. Addition-ally, this phenomenon also restricts the ability to compare multiple words against each other with respect to individual coordinates. It would appear that the numerical values of a word's representation are meaningful only with respect to representation of other words, under very specific conditions. So, would it be possible to create a representation that is not only comprised of coordinates that are meaningful to humans, but is also made of individual coordinates that enable humans to distinguish between words?

Ideally such a representation would be of reduced size and capable of capturing the meaning of the word along absolute coordinates i.e coordinates whose values are meaningful to humans and can be compared against other coordinates. A common absolute that encompasses all words in the vocabulary are the eight parts of speech, namely noun, verb, adjective, adverb, pronoun, preposition, conjunction and interjection. The parts of speech explain the role of a particular word within the given context and with a coarse grained approach, all words can be classified as at least one part of speech. This work, precisely describes the representation of words in a vector space where each coordinate corresponds to one of the eight parts of speech, derived from the pre-trained word embeddings via post processing. Such representation would be able to capture the absolute syntactic meaning of a word. For instance, the syntactic representation for the words "right" and "I" can be visualized in figures 1.1 and 1.2.

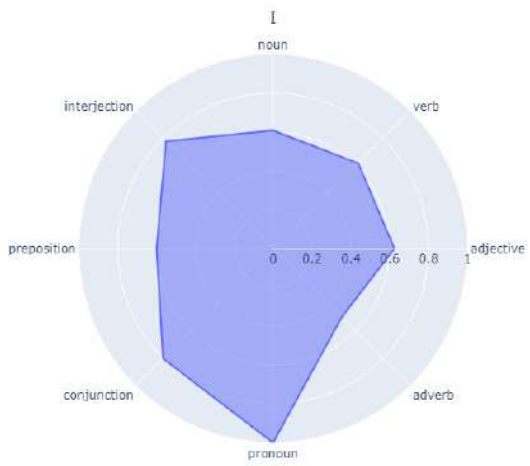


Figure 1.1: Syntactic Representation for *I*

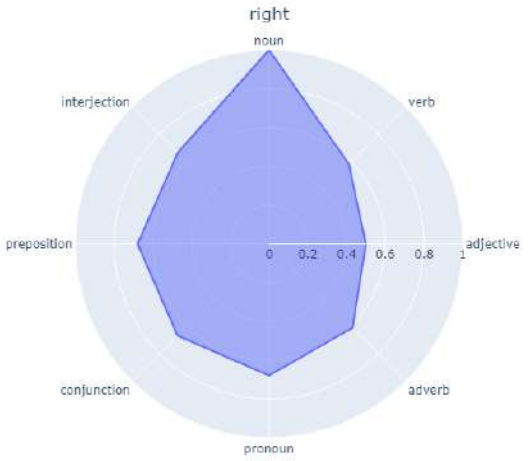


Figure 1.2: Syntactic Representation for *right*

1.2 Problem Statement

With the advancement in Natural Language Processing and introduction of context-dependent word embeddings, multiple studies have been conducted on tasks such as Text Classification, Named-Entity Recognition, Question Answering and Sentiment Analysis . These studies have yeilded state-of-the -art results and have established NLP as one of the foundations of Artificial Intelligence. Even though, the studies have provided exceptional results, the usage of word embeddings as input, instead of more classical and understandable features have led to a lack of interpretability and consequently NLP models being regarded as a black-box.

To address this issue, the word embeddings will have to be transformed and represented in

an interpretable space as opposed to their natural unintelligible high dimensional space. To perform this, the concept of transforming the embeddings into a different subspace in order to interpret them has to be used. This process utilizes the understanding of embeddings in some known dimensions and the affiliation of the word with respect to given dimensions would give the human understandable interpretation of the embedding. The obtained result could then easily be visualized in various charts making it easier to understand and compare the embeddings generated by various models. Overall, the goal of this thesis was to interpret word embeddings generated by pre-trained models and then visualize the embeddings to observe the models findings using absolute metrics.

1.3 Objectives

The primary objectives of this thesis have been mentioned below:

- To interpret the dense word embeddings in terms of reduced Syntactic Representations and further generate Hierarchical Vectors.
- To evaluate the performance of Hierarchical Vectors in benchmark tests and Syntactic Representations in interpretability experiments.

Chapter 2: Literature Review

Multiple researches have been conducted in the field of word embedding in multiple languages. The research on word embeddings encompasses various topics such as integration to downstream tasks, explainability, interpretation and much more.

2.1 Word Embeddings

Word embeddings are continuous representations of words which serve as the foundation for numerous NLP applications. Word embeddings can be generally defined as dense, distributed, fixed-length word vectors obtained from the rows of the word-context co-occurrence matrix and are based on the distributional hypothesis.[8] Typically, the representation is a real-valued vector that encodes the meaning of the word in such a way that words that are closer in the vector space are expected to be similar in meaning.[9] Various methods can be used to generate word embeddings from a corpus of words by the process of mapping each word to its respective vector.

2.1.1 Count Based Word Embeddings

Count Based Word Embeddings are centered around the frequency which encompasses and discovers the count occurrence of each word, co-occurrence of the words and significance of rare words, in the form of word-context matrices, globally in a corpus. This method examines the target word as a representation of the nature of words co-occurring in multiple contexts throughout the document. These methods are generally fast in training and are expected to use statistics efficiently to better capture word similarity.

The earliest work done to produce count based word embeddings by leveraging word-context matrix is Latent Semantic Analysis (LSA)[10] where Singular Value Decomposition (SVD) is applied on a term-document matrix to take advantage of implicit higher-order structure in the association of terms with the semantic structure i.e document. This method improved the detection of relevant documents on the basis of terms found in queries. A little later, the Hyperspace Analogue to Language (HAL)[11] was introduced which initiated the use of lexical co-occurrence to produce high dimensional semantic spaces. The HAL model was used to scan the whole corpus using a sliding window representing a fixed-sized span of words to calculate the word-word co-occurrence count and subsequently build a word-word co-occurrence matrix over the global corpus.

The newer count based word embeddings models include the well-known GloVe[3]. GloVe was

introduced as a global log bilinear regression model that combines the advantages of the two major model families in the literature: global matrix factorization and local context window methods. This model utilizes the knowledge that the ratios of co-occurrences , rather than the raw co-occurrence counts themselves encode the actual semantic information about the pair of words. This relationship is used to describe a loss function, which is then trained only on the non zero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or on individual context windows in a large corpus. This loss function subsequently maximized the similarity of every word pair as measured by the ratio of co-occurrences. This model reported a better result over all previous count-based models and also the Skip-Gram with Negative Sampling model[4], in multiple NLP tasks. Furthermore, this model produced a vector space with a meaningful substructure as demonstrated by the 75% score on word analogy task.

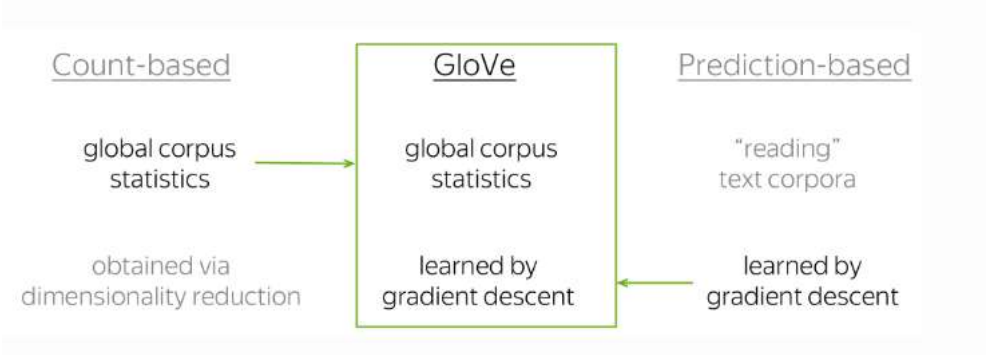


Figure 2.1: Working of `Glove`

2.1.2 Prediction Based Word Embeddings

Prediction Based Word Embeddings are the methods based on generating and assigning probabilities to the words and mapping them to a vector. The generation of embeddings for these methods involve the process of training the lookup tables which convert the word to dense vectors by learning the context. These embeddings are also referred to as static word embeddings as once the context has been learned, it is not altered.

The development of prediction based word embedding models is highly correlated with the development of Neural Language Models (NNLMs) as these embeddings were derived as a by-product of training neural language models. The history of NNLMs started with the neural probabilistic language model which proposed approaches to simultaneously learn distributed representations for each word i.e similarity between words along with learning the probability function for word sequences, represented in terms of the aforementioned distributed representations.[12] Even though the early results, with respect to perplexity, illustrated that neural language models could model language better than their n-gram counterparts, the longer training times for neural language models hindered their development.

After the seminal paper of neural probabilistic language model, various contributions were made towards optimizing neural language models. This led to the improvement on the previous paper, introducing a Monte Carlo method to bypass the partition function or normalization factor required by the softmax layers in the previous model. This introduction subsequently replaced the calculation of costly partition functions subsequently decreasing the training time by a factor of 19.[13] Further works were conducted to speed up the training times by using a binary approach with a Hierarchical Softmax Layer[14] and the subsequent introduction of the Log-bilinear model (LBL)[15].

After subsequent improvement of the log-bilinear model, a unified architecture for natural language processing using deep neural networks was presented which approached the problem with the specific intent of learning embeddings only. This model treated word embeddings as the main product under investigation rather than as a by-product as established by its predecessors. In addition to this, this model introduced the concept of using a words' full context to predict the centre word and the concept of leveraging unlabelled data to produce good embeddings by training a model that could distinguish positives from false examples.[16]

Further down the road, a major breakthrough was seen in the field of word embeddings, with the check for syntactic regularities in word embeddings resulting not only in syntactic regularities but also in semantic regularities.[17] These embeddings had been obtained by training a recurrent neural network and demonstrated that common relationships such as king-queen, man-woman, etc correspond to vector arithmetic.

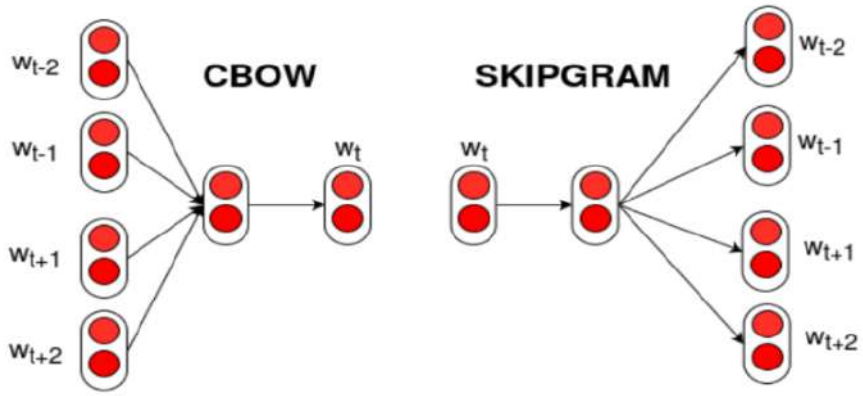


Figure 2.2: Techniques used in `Word2Vec`

2.1.3 Contextual Word Embeddings

Contextual Word Embeddings are numerical representation of words generated with respect to the context in which the word is used. Unlike count based embeddings and prediction based em-

beddings, which work to generate representations of a word type, contextual word embeddings generate a representation of the word token. There was a need for Contextual Word Embeddings as previous word embedding techniques were unable to deal with polysemous words.

Contextual Word Embeddings represent individual word tokens instead of word types, the problem being solved was made harder than it should have been. This was due to the fact that all possible senses of a given word had been conflated to a single representation[6]. For instance, the word *get* has a minimum of thirty possible contextual uses as per WordNet, but representations generated using count based embeddings or prediciton based embeddings would generate a single representation for all use cases. With the switch to word token vectors, it has become possible for the representation to only encapsulate the sense of the word in which it has been used.

One of the first works to present contextual embeddings was TagLM[18], which was a predecessor to most of the modern context-sensitive word embeddings. The work demonstrated a general semi-supervised approach for adding pretrained context embeddings from bidirectional language models to NLP systems and applied it to sequence labeling tasks. The following evaluations resulted in state-of-the-art performances in two tasks namely, named entity recognition and chunking. [18] This work further led to the development of ELMO, which is Embeddings from Language Models which is deep contextualized word representations.

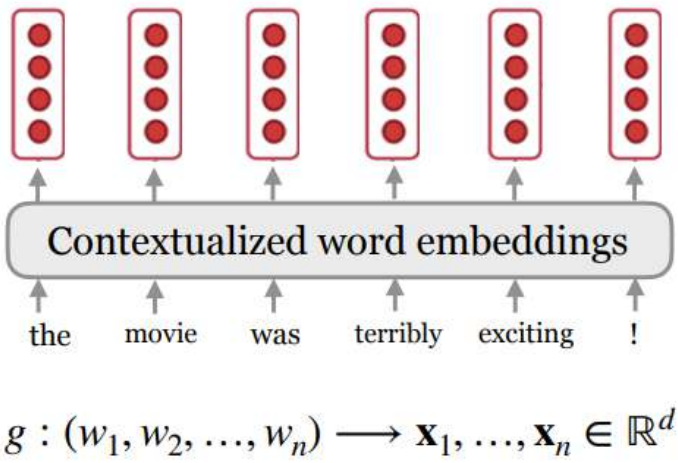


Figure 2.3: Working of Contextual Word Embeddings

2.2 Interpretation of Word Embeddings

Word embedding interpretability is a topic of active research in text representation. Various works have been conducted using multiple techniques to produce all kinds of interpretations.

Some of these works have been detailed below.

2.2.1 Sparse, Binary and Overcomplete Vectors

The word embeddings generated using the aforementioned methods are dense and continuous i.e the coordinates are mostly non-zero. These vectors, although useful, do not follow the general conventions of lexical semantic theories. So, a work was conducted to transform these dense vectors into sparse vectors using a principled sparse coding method. Furthermore, these sparse vectors were also transformed into binary representations and evaluated under various benchmark tests.[19]

This transformation resulted in longer, sparser vectors termed as "overcomplete" vectors. Overcomplete vector is generally defined as the resulting vector whose dimension is greater than that of the original vector.[20] This was the first work done on overcomplete vectors which were sparse and categorical. The interpretability of the vectors was testing using a word intrusion experiment and an unlabeled qualitative analysis experiment, which demonstrated that the transformed sparse overcomplete vectors were more interpretable than the dense original vectors.

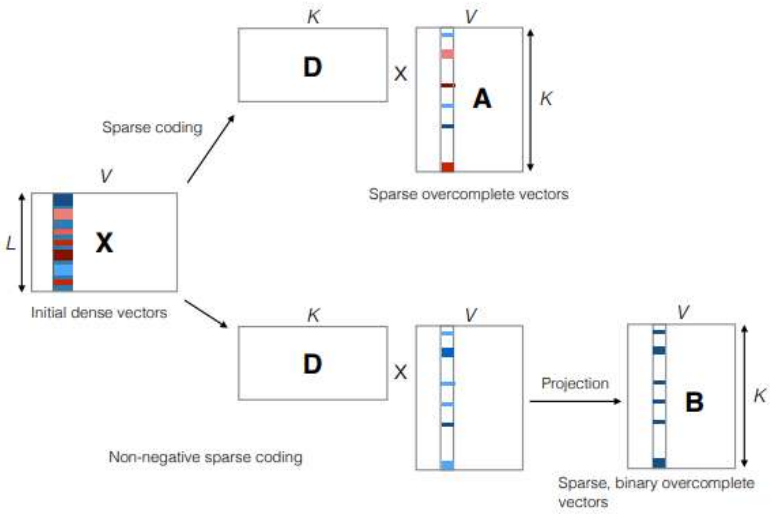


Figure 2.4: Generation of Sparse Overcomplete Vectors

2.2.2 SPINE Embeddings

This work was performed to improve upon the previously mentioned sparse vectors. The principles of sparsity and non-negativity have been used to transform the original dense vectors to interpretable sparse vectors. This have been done by employing a denoising k-sparse autoencoder to obtain SParse Interpretable Neural Embeddings (SPINE): a transformation of dense

input word embeddings. The autoencoder was trained using a novel learning objective and activation function to attain interpretable and efficient representations.[21]

The generated SPINE embeddings were further subjected to an extensive, crowdsourced intrusion detection test, where it substantially outperformed the sparse overcomplete vector. Additionally, a qualitative assessment of the embedding was conducted using a test subject as a label and the words closest to it in the embedding space as the result. The results demonstrated that the top-ranking words from SPINE are coherent and relevent to the word under examination.

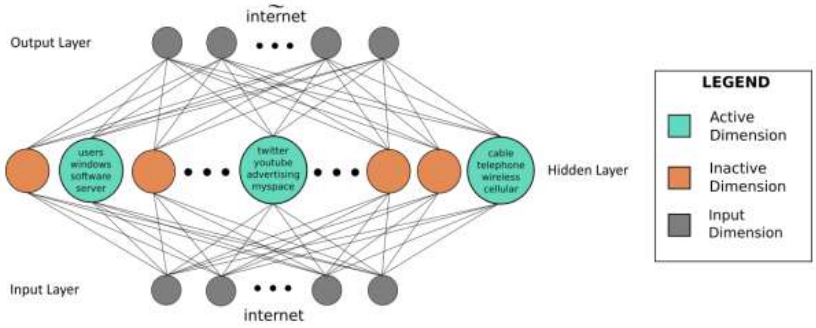


Figure 2.5: Sparse Autoencoders for the word `internet`

2.2.3 Word2Sense

This work presented an unsupervised method to generate Word2Sense word embeddings that were interpretable. This was done by recovering senses from a corpus and representing word embeddings as sparse probability distributions over senses.[22] Each dimension of the embedding space corresponded to a fine-grained sense, and the non-negative value of the embedding along each dimension represented the relevance of that sense to the word. The underlying LDA-based generative model was extended to refine the representation of a polysemous word in a short context, allowing the use of these embeddings in contextual task.

Word2Sense presented and efficient unsupervised method to embed words, in and out of context, in a way that captured their multiple senses in a corpus, in an interpretable manner. It was also demonstrated that the generated interpretable embeddings were competitive with the original dense embeddings on similarity tasks and captured textual entailment effectively. Furthermore, this work demonstrated the qualitative assessment of embeddings with polysemous words as predefined labels which resulted in top-ranking words categorized into various senses.

2.2.4 Visually Analyzing Contextual Embeddings

This paper presented a novel method for visually analyzing contextualized embeddings produced by deep neural network-based language models. This approach was inspired by linguistic probes for natural language processing, where tasks are designed to probe language models for linguistic structure, such as parts-of-speech and named entities. This work deliberately avoids supervised probing tasks, subsequently advocating for unsupervised probes, coupled with visual exploration techniques, to assess what is learned by language models. Specifically, contextualized embeddings produced from a large text corpus, are clustered and a visualization is designed based on this clustering and textual structure – cluster co-occurrences, cluster spans, and cluster-word membership.[23]

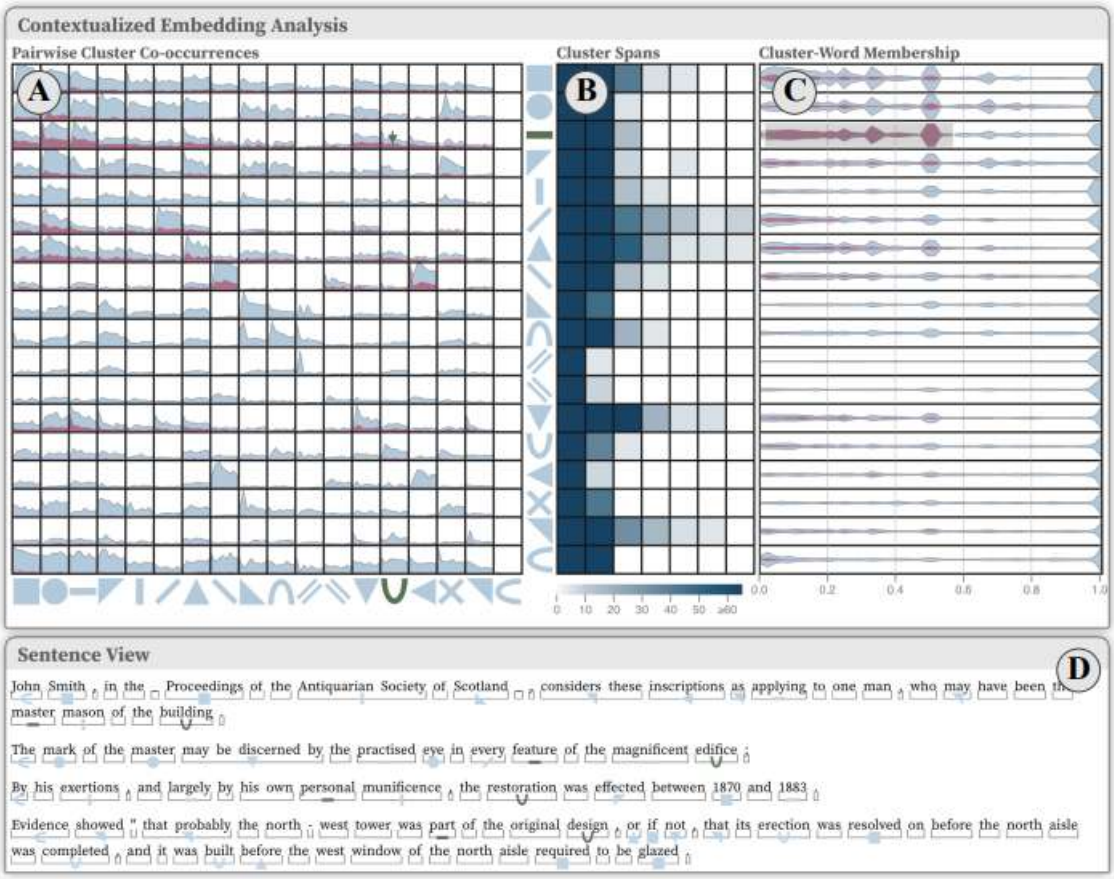


Figure 2.6: Visualization of Contextualized Word Embeddings

2.2.5 WIZMAP

Machine learning models learn representations that capture the domain semantics of their training data. Even though these embeddings are valuable for various downstream tasks and have produced state-of-the-art results, interpreting and using these embeddings has been challenging due to their opaqueness and high dimensionality. With a novel multi-resolution embedding sum-

marization method and a familiar maplike interaction design, this work has presented WIZMAP, an interactive visualization tool that could help researchers and practitioners easily explore large embeddings.[24] WIZMAP enables users to navigate and interpret embedding spaces with ease by leveraging modern web technologies.

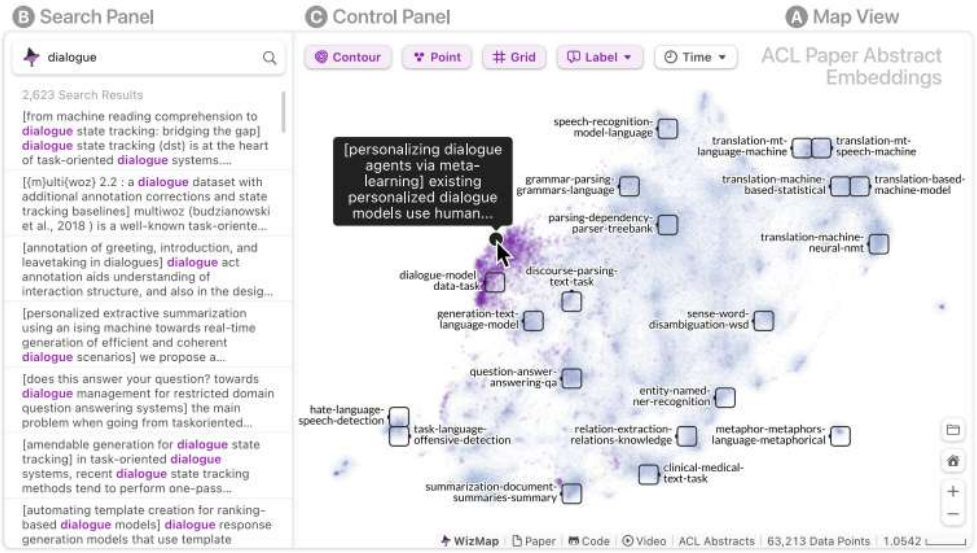


Figure 2.7: Visualization produced by `WIZMAP`

2.2.6 The POLAR Framework

The POLAR Framework: Polar Opposites Enable Interpretability of Pre-Trained Word Embeddings has described a system that uses semantic differentials to give pre-trained word embeddings more interpretability.[25] A psychometric tool known as semantic differentials was used to analyze a words position on a scale between two polar opposites (such as cool and hot, or soft and hard) to determine the new embeddings.

The methods central idea was to use semantic differentials to transform pre-trained word embeddings into a new "polar" space with interpretable dimensions determined by polar opposites. The discriminative dimensions from a set of polar dimensions have been provided by an oracle, which has already been selected, and the dimensions to be used have been selected using various algorithms. The transformed vector was then evaluated on a number of downstream tasks, where the interpretable word embeddings obtained performance that is comparable to the original vectors. On the basis of interpretability, this vector was examined to be more interpretable than the original vector.

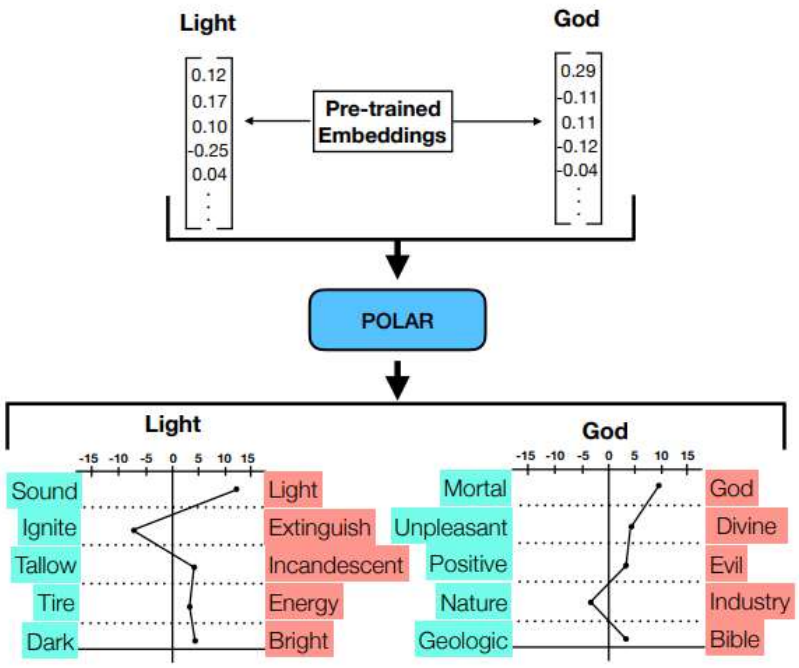


Figure 2.8: The Polar Framework

2.2.7 SensePOLAR

Although the previously mentioned works were able to produce interpretable dimensions, they were not intended to handle polysemy, i.e. it was difficult for the models to differentiate between several word meanings. SensePOLAR was introduced as an addition to the original POLAR framework that allowed word sense aware interpretability for pre-trained contextual word embeddings.[26] This addition ensured that the embeddings could handle multiple senses of the words.

These embeddings were evaluated on an array of benchmark tasks, such as the GLUE and SQuAD benchmarks. The resulting interpretable word embeddings performed on par with the original contextual word embeddings. By providing users with sense-aware interpretations for contextual word embeddings, SensePOLAR had eliminated a fundamental weakness of existing techniques.

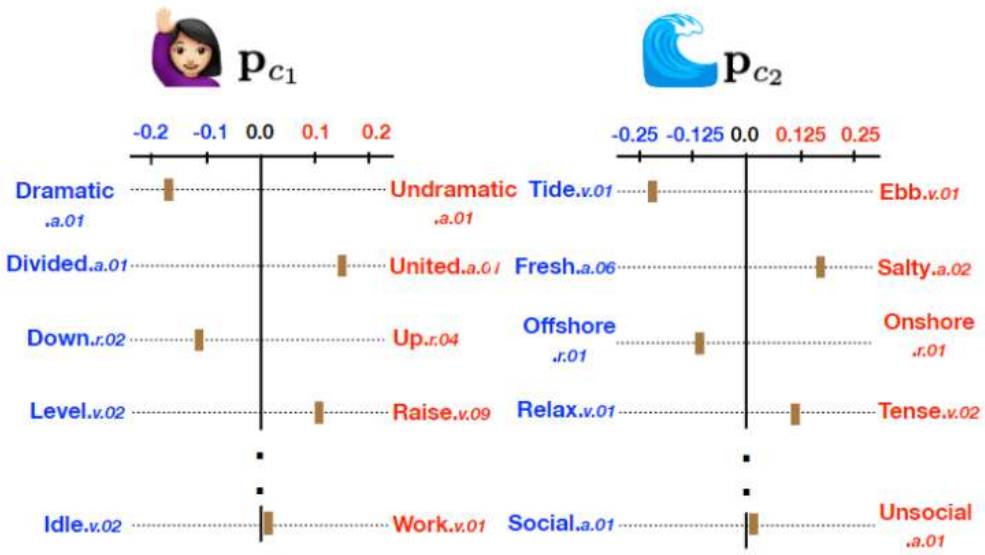


Figure 2.9: SensePOLAR

Chapter 3: Methodology

3.1 Construction of Interpretable Subspace

The fundamental objective of this work was to convert the dense word embeddings to interpretable transformation. For this objective, it was essential that a separate space be designed on which embeddings, when projected, would have some meaning. Therefore, it was decided that the eight parts of speech would the basis for the subspace on which the embeddings would be interpreted. Although the eight parts of speech offered a coarse grained, conflated and limited interpretation to the word under investigation, they are ubiquitous and absolute. This allowed representation for all words in a labeled scale against which all words could be evaluated and compared. The designed subspace is shown in figure 3.1.

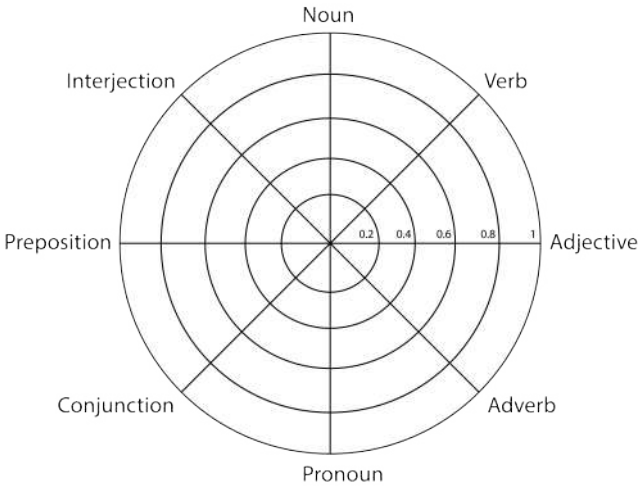


Figure 3.1: The Interpretable Subspace

3.2 Extraction of Words

The first step to generate word embeddings is to select the words for which the embedding has to be generated. This is done via an oracle, WordNet, which is an extensive lexical database of English, categorizing nouns, verbs, adjectives, and adverbs into sets of cognitive synonyms (synsets), each expressing a distinct concept.[27] These words have been categorized into senses which describe the context in which the words have been used. For this work, the evaluation has been conducted on words from WordNet which belong to exactly one part of speech. Furthermore, to complete the entire suite of parts of speech, the words from WordNet were supplemented with frequently used words from remaining parts of speech i.e pronoun, conjunction, preposition and interjection, which were manually collected.

3.3 Generation of Word Embeddings

Two arbitrary word embedding generating models were used for this task. The word, for which embedding is to be generated, was passed to the embedding model **M**. The embedding of the word can then be retrieved from the output of **M**. The embedding thus generated is the embedding of the respective word type. For the extracted words with greater than one word constituent, separate embeddings were calculated for individual constituents and averaged to obtain the resulting embedding. These embeddings were evaluated in groups of their original parts of speech classification and they were averaged to produce an absolute embedding of the respective part of speech. These absolute embedding were used as the flag bearers for each parts of speech. The process of embedding generation for words with only one word token is shown in figure 3.2 and for words with multiple tokens is shown in figure 3.3.

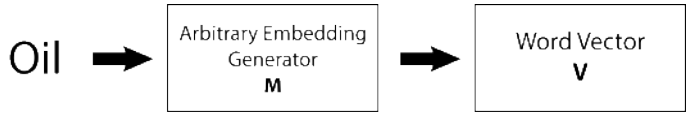


Figure 3.2: Word Embedding generation for words with only one token, eg. `Oil`

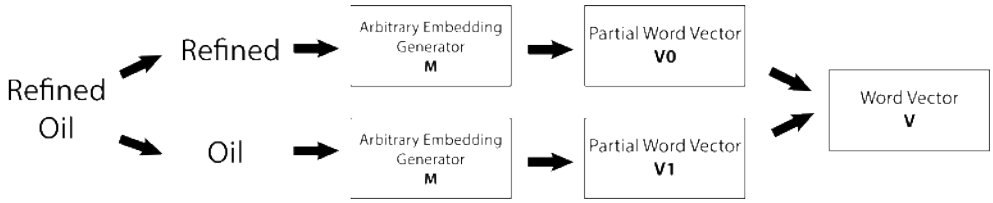


Figure 3.3: Word Embedding generation for words with multiple tokens, eg. `Refined Oil`

3.4 Transformation to Syntactic Representations

For each of the extracted word, the syntactic representation is generated using the Absolute Embeddings. The primary objective of generating Syntactic Representations was to convert the dense, continuous word embeddings into meaningful representations with reduced dimensions. These representations have been derived from the original vectors through post-processing methods.

Let V be the size of the vocabulary. $\boldsymbol{X} \in \mathbb{R}^{V \times L}$ is the matrix created by stacking V word vectors of size L, trained using an arbitrary unsupervised word embedding estimator. This matrix represents the original word vectors that will be transformed into syntactic representations.

Let $\mathbb{N}$ denote the set of word vectors for all nouns in WordNet that are also present in the vocabulary. The Noun coordinate of the change of basis matrix, i.e., the transition matrix $\boldsymbol{C}$ for the syntactic subspace $\boldsymbol{S}$, can then be obtained as:

$$\boldsymbol{C}_{1,:} = \sum_{i=1}^{V} \frac{n_i}{V} \tag{3.1}$$

where $n \in \mathbb{N}$ represents an individual word vector in $\mathbb{N}$.

These coordinates were calculated across all eight parts of speech and stacked to form the transition matrix for the syntactic subspace, $\boldsymbol{C} \in \mathbb{R}^{(V \cap W) \times 8}$, where W is the vocabulary of all words in WordNet . Let $v \in V$ be a word in the vocabulary. For v to be projected into the syntactic subspace, it was necessary that $v \in V \cap W$.

$$\boldsymbol{C}^T \cdot \boldsymbol{S}_{v,:} = \boldsymbol{X}_{v,:} \tag{3.2}$$

$$\boldsymbol{S}_{v,:} = (\boldsymbol{C}^T)^+ \cdot \boldsymbol{X}_{v,:} \tag{3.3}$$

where $(\boldsymbol{C}^T)^+$ denotes the Moore-Penrose generalized inverse of the transpose of the transition matrix [28].

Therefore, the Syntactic Representations for the given words $v \in V \cap W$ were generated as shown in equation 3.3. The equation has been used to generate three models of Syntactic Representations which have been discussed below.

3.4.1 Absolute Syntactic Representations

These Syntactic Representations were the actual outcomes obtained through post-processing of the original word embeddings. Unlike conventional normalization, which scales data to a specific range, these representations were intentionally preserved in their unnormalized form. The absolute representations for all parts of speech were constructed based on the words within each category from WordNet, ensuring that these words were also included in the vocabulary of the word embedding generator. These syntactic representations were subsequently used to derive hierarchical vectors, by employing them in conjunction with the original vectors, as illustrated in equations 3.7 and 3.9. Maintaining the unnormalized form of these representations offered a unique advantage, as it preserved the inherent richness of the original data, and proved invaluable for capturing the intricate details of syntactic structure and relationships, contributing to a more faithful and granular understanding of the underlying linguistic phenomena.

3.4.2 Interpretable Syntactic Representations

These Syntactic Representations were thoughtfully presented in an interpretable format, ensuring their accessibility and utility. They were meticulously derived from the representations

discussed in Section 3.4.1, employing a normalization process that scaled them to a range between 0.5 and 1. This ensured that the most likely coordinate could be easily determined while each coordinate in the representation had a value. In this normalized context, the value 1 serves as a clear indicator that the vector corresponded to a specific class.

If $\boldsymbol{S}$ represents the Syntactic Representations, the Interpretable Syntactic Representation, $\boldsymbol{I}$, for coordinate i can be calculated as:

$$\boldsymbol{I}_i = \frac{\boldsymbol{I}_{intermediate} + 1}{2} \tag{3.4}$$

where,

$$\boldsymbol{I}_{intermediate} = \frac{\boldsymbol{S}_i - min\{\boldsymbol{S}\}}{max\{\boldsymbol{S}_i\} - min\{\boldsymbol{S}\}} \tag{3.5}$$

3.4.3 L2 Normalized Syntactic Representations

These Syntactic Representations have been meticulously normalized such that their all the square sum of the coordinate values equals to one. The process of normalization is crucial as it ensured that the representations were consistent and comparable. By constraining the values to this specific range, the ease of analysis and modeling was enabled. This normalization process facilitated more robust and meaningful computations, contributing to the overall utility of the syntactic representations.

If $\boldsymbol{S}$ represents the Syntactic Representations, the Interpretable Syntactic Representation, $\boldsymbol{L}$, for coordinate i can be calculated as:

$$\boldsymbol{L}_i = \frac{\boldsymbol{S}_i}{\sqrt{\sum_{k=1}^{8} \boldsymbol{S}_k^2}} \tag{3.6}$$

3.5 Visualization of Syntactic Representations

After the syntactic representations had been generated, the interpretable syntactic representations mentioned in 3.4.2 were projected into the syntactic subspace. An example of the generated visual representation of these interpretations can be seen in figure 3.4, providing a visually intuitive means for users to comprehend and leverage the syntactic richness encapsulated within the representations.

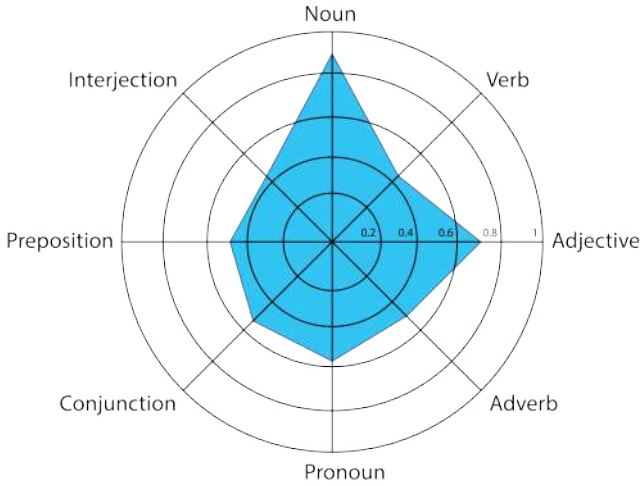


Figure 3.4: Example of Visualization of Syntactic Representations

3.6 Creation of Hierarchical Vectors

This work has presented Hierarchical Vectors, composite vectors formed by combining derived Syntactic Representations with the original word embeddings. The central idea behind Hierarchical Vectors was to emulate the human learning process. As humans, knowledge and wisdom is acquired incrementally, gradually developing understanding, mastering skills, and adapting to changes in our environment. This incremental learning process, while valuable, inadvertently gives rise to bias as a de facto aspect of human behavior.

However, the predominant approach in most NLP models has relied on word embeddings that encapsulate syntactic regularities, semantic nuances, and various linguistic subtleties, all compressed into static representations. While these pre-trained word vectors have provided a strong foundation for language understanding, they have often lacked the adaptability, interpretability, and means for quantifying bias. To address these limitations, Hierarchical Vectors have been introduced, which embrace the concept of incremental learning. This approach entails an initial phase of learning from the reduced Syntactic Representations, followed by the gradual incorporation of the broader original word vectors into the learning process.

By employing the incremental learning concept, the main objective was to isolate syntactic regularities from the original word vectors, leading to a more refined and enriched representation of the linguistic structures present in the data. This strategic use of incremental learning aids in demonstrating learning as a step-wise process involving multiple isolated embeddings, ultimately building robustness and interpretability in the learning process. This work has presented the following two models of Hierarchical Vectors.

3.6.1 Overcomplete Hierarchical Vectors

Both the Syntactic Representations and the original vectors of a word have been leveraged to generate Overcomplete Hierarchical Vector for that word. Let $\boldsymbol{V}^s \in \mathbb{R}^{1\times 8}$ and $\boldsymbol{V}^r \in \mathbb{R}^{1\times L}$ represent the Syntactic Representation and the original representation of the word v, respectively. The Overcomplete Hierarchical Vectors for v was then computed as follows:

$$\boldsymbol{V}^o = \boldsymbol{V}^s \otimes \boldsymbol{V}^r \tag{3.7}$$

where $\otimes$ represents the Kronecker product and $\boldsymbol{V}^o \in \mathbb{R}^{1\times(8\times L)}$ represents the Overcomplete Hierarchical Vectors.

for eg: if $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} a' & b' \\ c' & d' \end{bmatrix}$ then,

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} aa' & ab' & ba' & bb' \\ ac' & ad' & bc' & bd' \\ ca' & cb' & da' & db' \\ cc' & cd' & dc' & dd' \end{bmatrix} \tag{3.8}$$

3.6.2 Weighted Hierarchical Vectors

The coordinate values in the Syntactic Representations were used as weights for the original vector of the word to generate the Weighted Hierarchical Vector for the respective word. Let $\boldsymbol{V}^s \in \mathbb{R}^{1\times 8}$ and $\boldsymbol{V}^r \in \mathbb{R}^{1\times L}$ represent the Syntactic Representation and the original representation of the word v, respectively. Each coordinate, *say* $\boldsymbol{j}$, of the Weighted Hierarchical Vector $\boldsymbol{V}^w$ was then computed as follows:

$$\boldsymbol{V}^w_{j:} = \sum_{i=1}^{8} \frac{\boldsymbol{V}^s_{i:} \times \boldsymbol{V}^r_{j:}}{8} \tag{3.9}$$

The coordinates for the Weighted Hierarchical Vectors were calculated as the element-wise weighted average of the original vector by the Syntactic Representations. The resulting Weight Hierarchical Vector was of the same dimension as the original vector, i.e $\boldsymbol{V}^w \in \mathbb{R}^{1\times L}$.

for eg: if $\mathbf{A} = \begin{bmatrix} a & b \end{bmatrix}$ represent the Syntactic Representation, $\mathbf{B} = \begin{bmatrix} a' & b' \\ c' & d' \end{bmatrix}$ represent the pretrained vector and $\mathbf{W}$ represent the Weighted Hierarchical Vectors then,

$$\textbf{Weighted Hierarchical Vector} = \begin{bmatrix} \frac{aa'+ba'}{2} & \frac{ab'+bb'}{2} \\ \frac{ac'+bc'}{2} & \frac{ad'+bd'}{2} \end{bmatrix} \tag{3.10}$$

3.7 Block Diagram

Figure 3.5 represents the thesis in a block diagram.

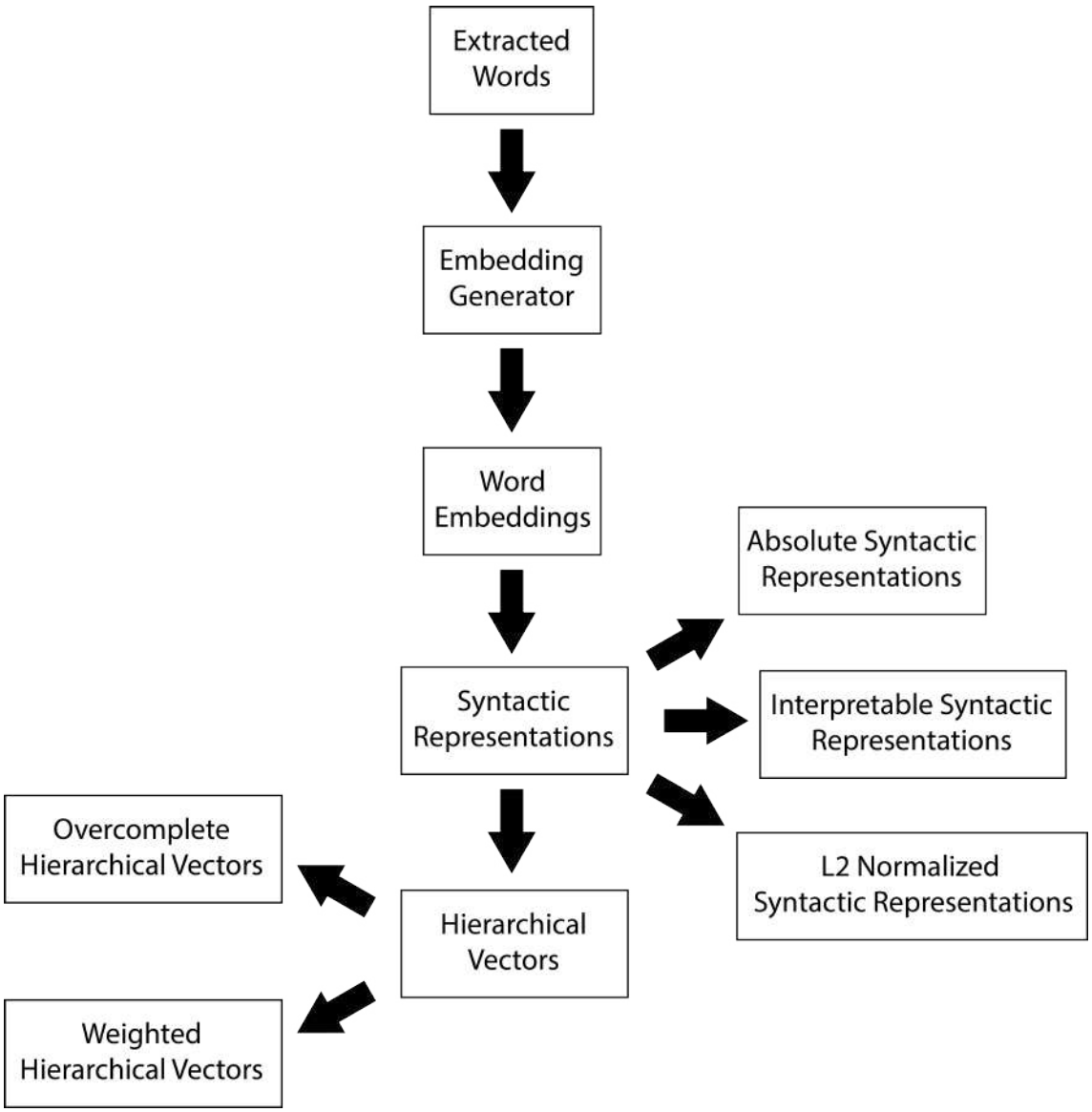


Figure 3.5: Block Diagram of the Thesis work

Chapter 4: Output

4.1 Experimental Setup

As this work does not require any raw text corpus for the generation of representations, two popular pretrained word vector representations have been used, **Word2Vec** [4] with 300 dimensions, trained on 3 million words of the Google News dataset and **GloVe** [3] trained on 2.2 Million words of the Common Crawl dataset.

Three models of Syntactic Representations have been used with the two types of Hierarchical Vectors to produce six subsequent Hierarchical Vectors. These vectors have then been compared against the respective original word embeddings and their vector siblings. It is important to reiterate that the major intention of this work was to isolate the syntactic regularities from the dense, continuous word vectors to enable interpretation, without improving the performance of the embedding. Rather, it was intend that word vectors would be interpreted without any loss in performance.

4.2 Notations

The following notations have been used to represent the respective elements under evaluation:

NOTATION		
$\mathbf{W}$	:	Pretrained Word2Vec Representations
$\mathbf{WO^A}$	:	Hierarchical Overcomplete Absolute Word2Vec
$\mathbf{WO^I}$	:	Hierarchical Overcomplete Interpretable Word2Vec
$\mathbf{WO^L}$	:	Hierarchical Overcomplete L2 Word2Vec
$\mathbf{WW^A}$	:	Hierarchical Weighted Absolute Word2Vec
$\mathbf{WW^I}$	:	Hierarchical Weighted Interpretable Word2Vec
$\mathbf{WW^L}$	:	Hierarchical Weighted L2 Word2Vec
$\mathbf{G}$	:	Pretrained GloVe Representations
$\mathbf{GO^A}$	:	Hierarchical Overcomplete Absolute GloVe
$\mathbf{GO^I}$	:	Hierarchical Overcomplete Interpretable GloVe
$\mathbf{GO^L}$	:	Hierarchical Overcomplete L2 GloVe
$\mathbf{GW^A}$	:	Hierarchical Weighted Absolute GloVe
$\mathbf{GW^I}$	:	Hierarchical Weighted Interpretable GloVe
$\mathbf{GW^L}$	:	Hierarchical Weighted L2 GloVe

Table 4.1: Notations and their respective meanings

4.3 Benchmark Tests

The generated Hierarchical Vectors have been used to perform the following benchmark downstream tasks: news classification, noun phrase bracketing, question classification, capturing discriminative attributes, sentiment classification and word similarity.

4.3.1 News Classification

The 20 news-groups dataset has been used and three binary classification tasks have been considered, namely on the Sports dataset, Religion dataset and the Computer dataset.[19] Each of these tasks consisted of classifying the dataset into two distinct classes: *Hockey* vs *Baseball* for the Sports dataset with a training/validation/test split of 957/240/796, *Atheism* vs *Christian* for the Religion dataset with a training/validation/test split of 863/216/717 and *IBM* vs *Mac* for the Computer dataset with a training/validation/test split of 934/234/777. The average of the Hierarchical Vectors of the words in a given sentence have been used as features for classification.[21]

4.3.2 Noun Phrase Bracketing

The generated Hierarchical Vectors were evaluated on the NP Bracketing task [29], where a noun phrase of three words had to be categorized as either left beacketed or right bracketed. For instance, given a noun phrase *monthly cash flow*, the task was to decide whether the phrase was *{(monthly cash) (flow)}* or *{(monthly) (cash flow)}*. The dataset consisting of 2,227 noun phrases, built using the Penn Treebank [30], in a training/validation/test split of 1602/179/446 data instances was used for evaluation. The task was performed using the feature vector obtained by averaging over the vectors of words in the phrase. SVC (with both Linear and RBF kernel), Random Forest Classifier and Logistic Regression were used for the task, and the model with highest validation accuracy was used for tesing purposes.

4.3.3 Capturing Discriminative Attributes

The Capturing Discriminative Attributes task was proposed as a novel task for semantic difference detection, as Task 10 of the SemEval 2018 workshop.[31] The fundamental understanding behind this task was to check if an attribute could be used to discriminate between two concepts. For instance, the model must be able to determine that *straps* is the discriminating attribute between two concepts, *sandals* and *gloves*. This task is a binary classification task on the given SemEval 2018 Task 10 dataset consisting of triplets (in the form *(concept1, concept2, attribute)*) with a train/test/validation split of 17501/2722/2340. As suggested, an unsupervised distributed vector cosine baseline was used on the entire data regardless of the training/validation/test

split.[31] The main idea was that the cosine similarity between the attribute and each concept must enable in discriminating between the concepts.

4.3.4 Question Classification (TREC)

To assist in Question Answering, a dataset was proposed to categorize a given question into one of six different classes, eg. whether the question was about a person, some numeric information or a certain location.[32] The TREC dataset was made up of 5452 labeled questions as the training dataset and a test dataset of 500 questions. By isolating 10% as the validation dataset, a training/validation/test split of 4906/546/500 questions was prepared. As in previous tasks, the feature vector was constructed by averaging over the vectors of the constituent words in the question. Various classification models were trained and the test was performed using the model with highest reported validation accuracy.

4.3.5 Sentiment Analysis

The Semantic Analysis task involved testing the semantic property of the word vectors by categorizing a given sentence into a positive or negative class. This test was performed using the Stanford Sentiment Treebank dataset [33] which consisted of a training/dev/test split of 6920/872/1821 sentences. The feature vector was generated by averaging over all constituent words of the sentence, only on the data instances with non-neutral labels, and report the test accuracy obtained by the model with the highest validation accuracy.

4.3.6 Word Similarity

The Word Similarity test intends to capture the closeness between a pair of related words. An array of datasets have been used namely, Simlex-999 [34], WS353, WS353-S and WS353-R [35], MEN [36] and MT-771 [37]. Each pair of data in all of these datasets were annotated a human generated similarity score. The cosine similarity of all of the pairs of words in each of the above dataset were calculated and the Spearman's rank correlation coefficient ρ between the model generated similarity and the human annotated similarity has been reported. Only the pairs where both words were present in the vocabulary have been considered for this test.

4.4 Benchmark Test Results

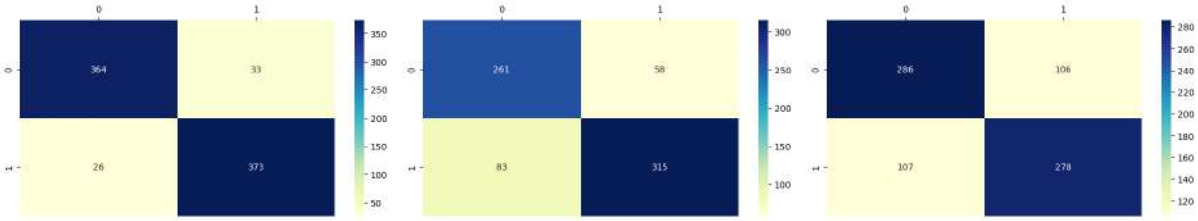
4.4.1 News Classification

Category	Vectors		Precision	Recall	f1-Score	Accuracy
Sports	Word2Vec		93.33	91.69	92.50	92.59
	Hierarchical	$\mathbf{WO^A}$	**96.67**	94.96	**95.81**	**95.85**
	Overcomplete	$\mathbf{WO^I}$	93.85	92.19	93.01	93.09
	Word2Vec	$\mathbf{WO^L}$	95.00	**95.72**	95.36	95.35
	Hierarchical	$\mathbf{WW^A}$	93.18	92.95	93.06	93.09
	Weighted	$\mathbf{WW^I}$	93.64	92.70	93.16	93.22
	Word2Vec	$\mathbf{WW^L}$	95.05	91.94	93.47	93.59
Religion	Word2Vec		75.87	81.82	78.73	80.33
	Hierarchical	$\mathbf{WO^A}$	80.72	84.01	82.33	83.96
	Overcomplete	$\mathbf{WO^I}$	76.85	81.19	78.96	80.75
	Word2Vec	$\mathbf{WO^L}$	83.44	85.27	84.34	85.91
	Hierarchical	$\mathbf{WW^A}$	**93.18**	**92.95**	**93.06**	**93.09**
	Weighted	$\mathbf{WW^I}$	70.16	68.37	69.25	69.37
	Word2Vec	$\mathbf{WW^L}$	81.29	83.07	82.17	83.96
Computer	Word2Vec		72.77	72.96	72.87	72.59
	Hierarchical	$\mathbf{WO^A}$	81.63	**81.63**	81.63	81.47
	Overcomplete	$\mathbf{WO^I}$	75.73	73.21	74.45	74.65
	Word2Vec	$\mathbf{WO^L}$	**82.86**	81.38	**82.11**	**82.11**
	Hierarchical	$\mathbf{WW^A}$	74.23	73.47	73.85	73.75
	Weighted	$\mathbf{WW^I}$	70.16	68.37	69.25	69.37
	Word2Vec	$\mathbf{WW^L}$	76.01	71.94	73.92	74.39

Scores are represented as %

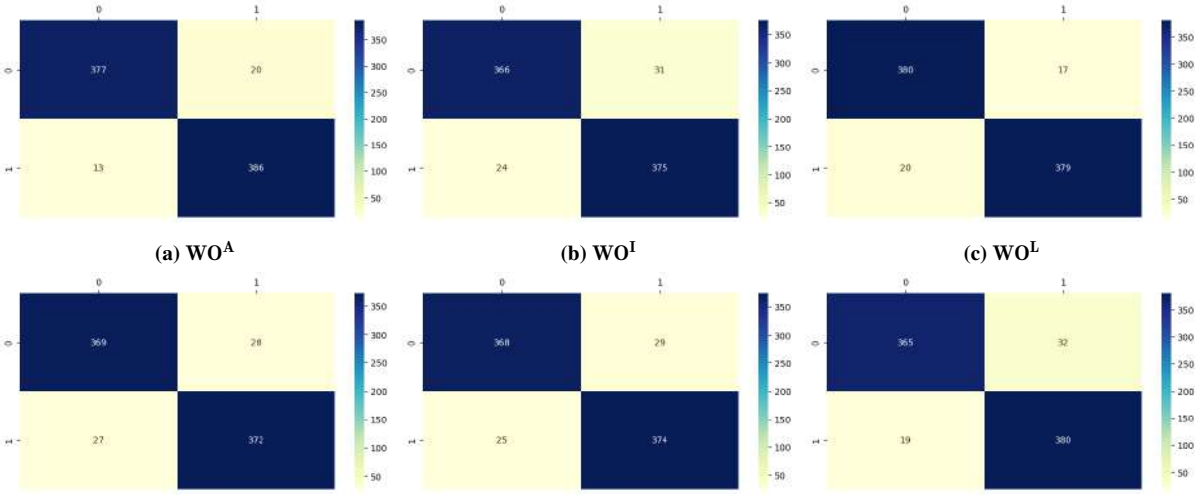
Table 4.2: Performance of word2vec and its descendant vectors in news classification task.

The performance of Word2vec and its descendant vectors have been presented in table 4.2. From the table, we can see the descendants outperform the base vector in terms of precision, recall, f1-score and accuracy, in all three news categories. The confusion matrices for the news classification task have been presented in figure 4.1 for word2vec and in figure 4.2, figure 4.3 and figure 4.4 for its descendants in the respective classification tasks.



(a) Sports News **(b)** Religion news **(c)** Computer news

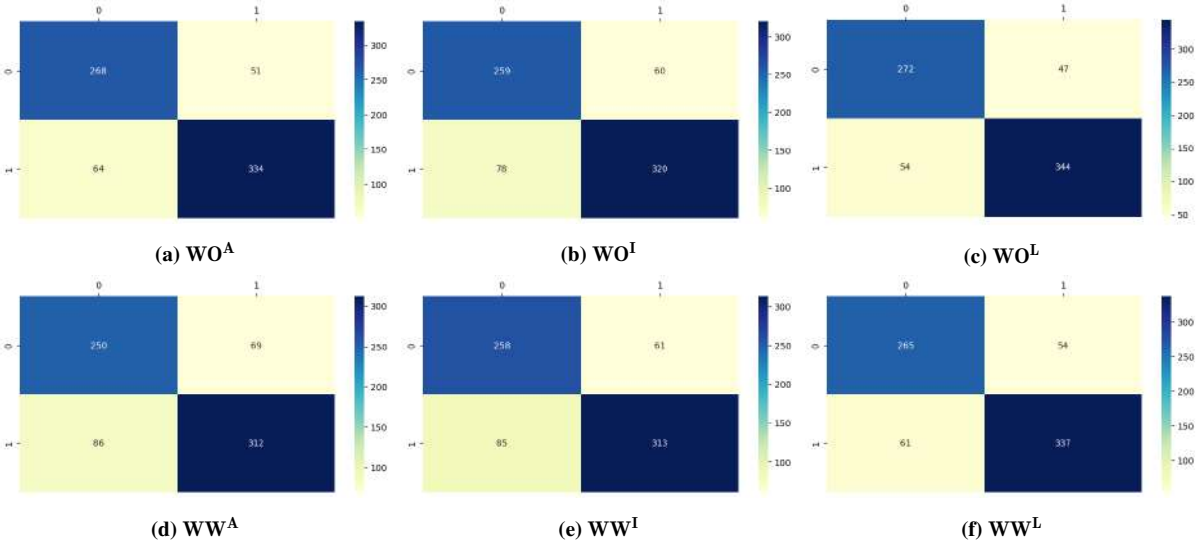
Figure 4.1: Confusion matrix for news classification using word2vec.



(a) WO^A **(b)** WO^I **(c)** WO^L

(d) WW^A **(e)** WW^I **(f)** WW^L

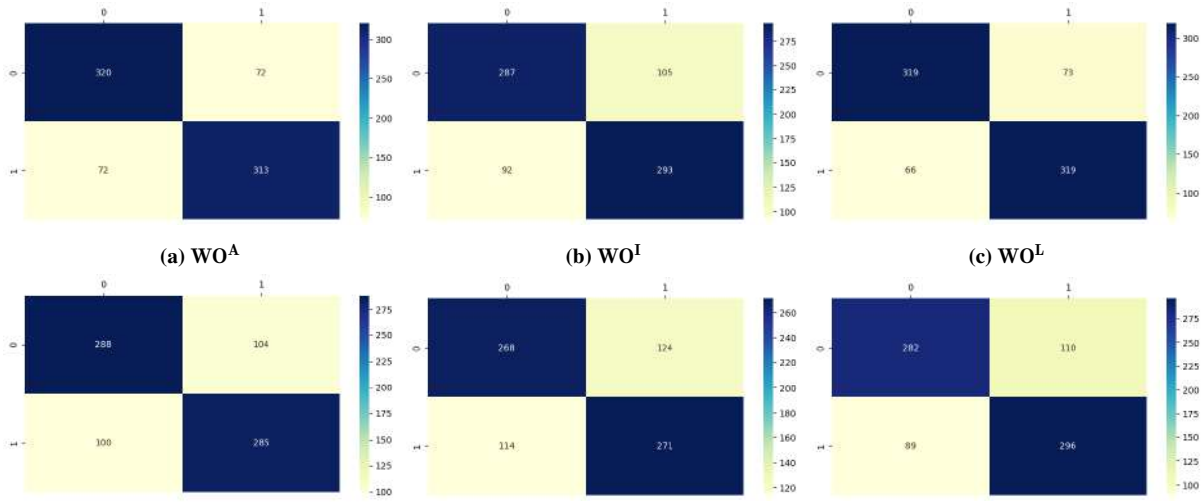
Figure 4.2: Confusion matrix for sport news classification using hierarchical vectors generated from word2vec.



(a) WO^A **(b)** WO^I **(c)** WO^L

(d) WW^A **(e)** WW^I **(f)** WW^L

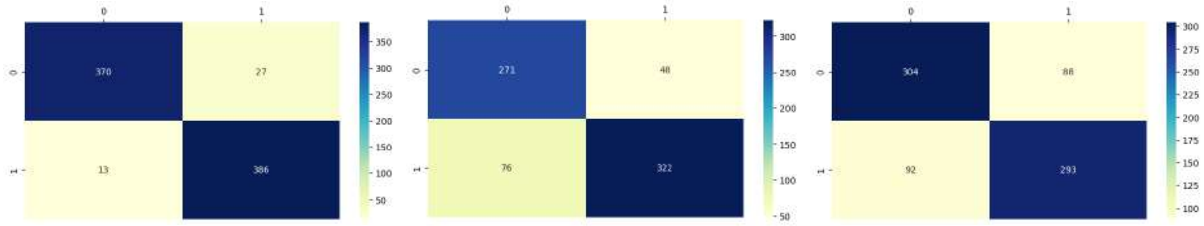
Figure 4.3: Confusion matrix for religion news classification using hierarchical vectors generated from word2vec.



(a) WO^A **(b)** WO^I **(c)** WO^L

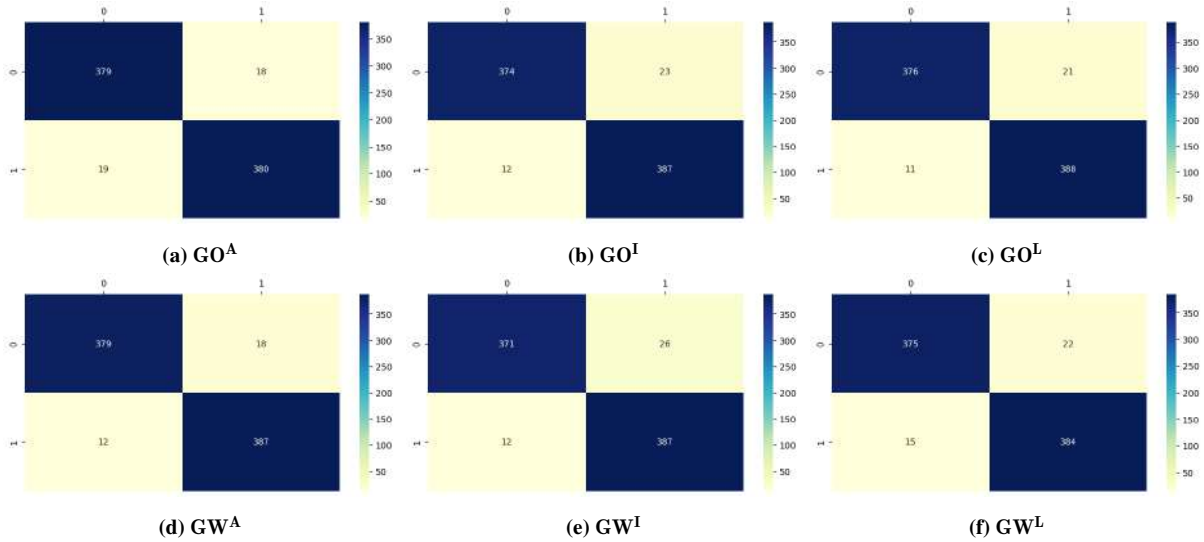
(d) WW^A **(e)** WW^I **(f)** WW^L

Figure 4.4: Confusion matrix for computer news classification using hierarchical vectors generated from word2vec.



(a) Sports News **(b)** Religion news **(c)** Computer news

Figure 4.5: Confusion matrix for news classification using glove.



(a) GO^A **(b)** GO^I **(c)** GO^L

(d) GW^A **(e)** GW^I **(f)** GW^L

Figure 4.6: Confusion matrix for sport news classification using hierarchical vectors generated from glove.

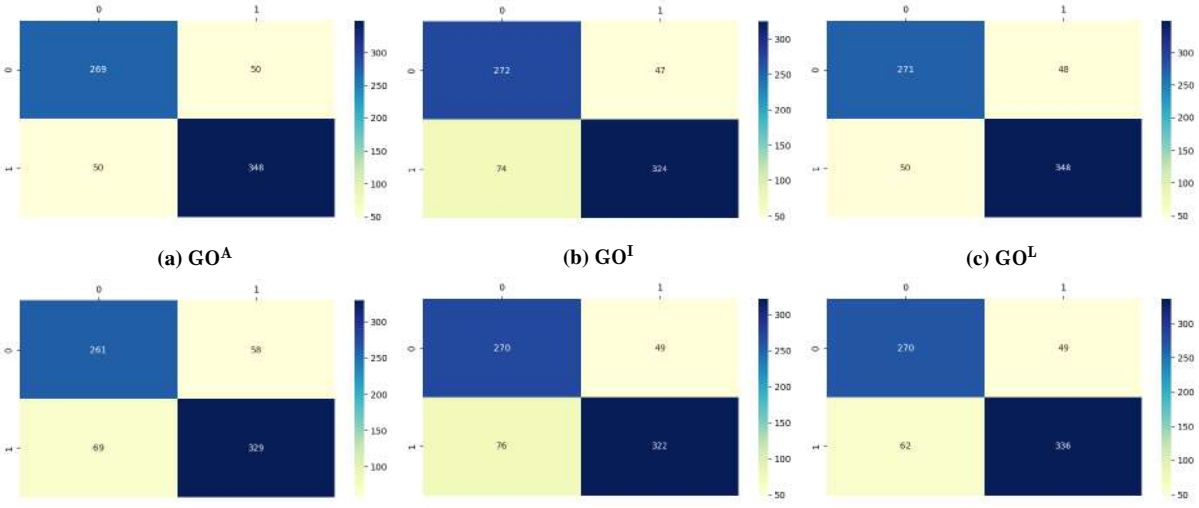
The performance of GloVe and its descendant vectors have been presented in table 4.3. From the table, we can see the descendants outperform the base vector in terms of precision,

recall, f1-score and accuracy, in all three news categories. The confusion matrices for the news classification task have been presented in figure 4.5 for glove and in figure 4.6, figure 4.7 and figure 4.8 for its descendants in the respective classification tasks.

Category	Vectors		Precision	Recall	f1-Score	Accuracy
Sports	Glove		96.61	93.20	94.87	94.97
	Hierarchical	$\mathbf{GO^A}$	95.23	**95.47**	95.35	95.35
	Overcomplete	$\mathbf{GO^I}$	96.89	94.21	95.53	95.60
	Glove	$\mathbf{GO^L}$	**97.16**	94.71	95.92	95.98
	Hierarchical	$\mathbf{GW^A}$	96.93	**95.47**	**96.19**	**96.23**
	Weighted	$\mathbf{GW^I}$	96.87	93.45	95.13	95.23
	Glove	$\mathbf{GW^L}$	96.15	94.46	95.30	95.35
Religion	Glove		78.10	**84.95**	81.38	82.71
	Hierarchical	$\mathbf{GO^A}$	84.33	84.33	84.33	**86.05**
	Overcomplete	$\mathbf{GO^I}$	**85.27**	78.61	81.80	83.12
	Glove	$\mathbf{GO^L}$	84.42	84.95	**84.69**	86.33
	Hierarchical	$\mathbf{GW^A}$	79.09	81.82	80.43	82.29
	Weighted	$\mathbf{GW^I}$	78.03	84.64	81.20	82.57
	Glove	$\mathbf{GW^L}$	81.33	84.64	82.95	84.52
Computer	Glove		76.77	77.55	77.16	76.83
	Hierarchical	$\mathbf{GO^A}$	82.64	86.22	84.39	83.91
	Overcomplete	$\mathbf{GO^I}$	79.28	84.36	81.74	81.03
	Glove	$\mathbf{GO^L}$	**83.54**	**86.73**	**85.11**	**84.68**
	Hierarchical	$\mathbf{GW^A}$	74.94	81.63	78.14	76.96
	Weighted	$\mathbf{GW^I}$	77.69	79.08	78.38	77.99
	Glove	$\mathbf{GW^L}$	76.77	77.55	77.16	76.83

Scores are represented as %

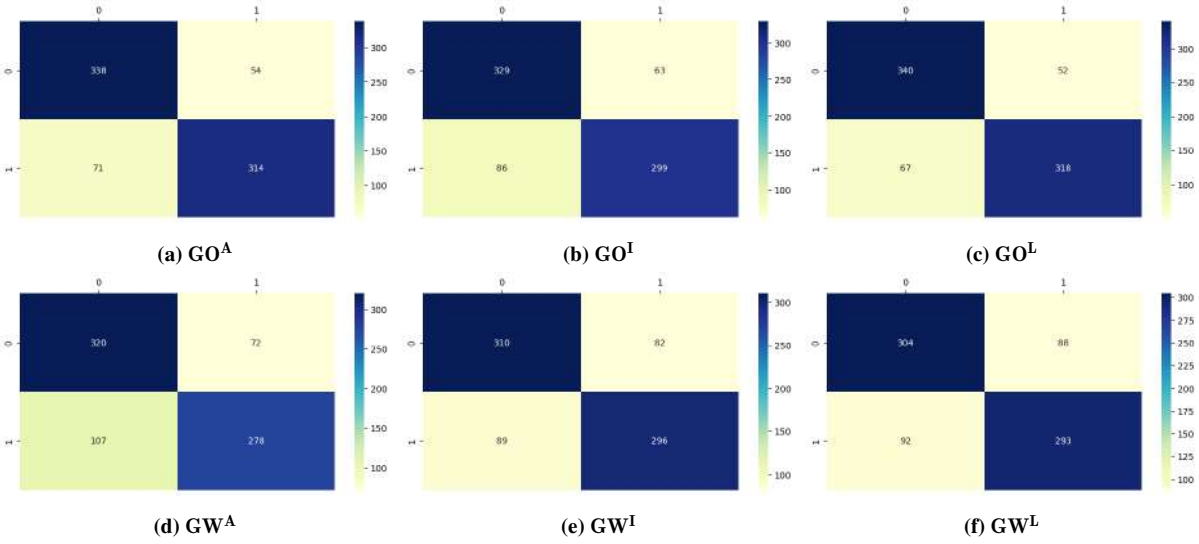
Table 4.3: Performance of GloVe and its vector descendants in news classification task.



(a) GO^A (b) GO^I (c) GO^L

(d) GW^A (e) GW^I (f) GW^L

Figure 4.7: Confusion matrix for religion news classification using hierarchical vectors generated from glove.



(a) GO^A (b) GO^I (c) GO^L

(d) GW^A (e) GW^I (f) GW^L

Figure 4.8: Confusion matrix for computer news classification using hierarchical vectors generated from glove.

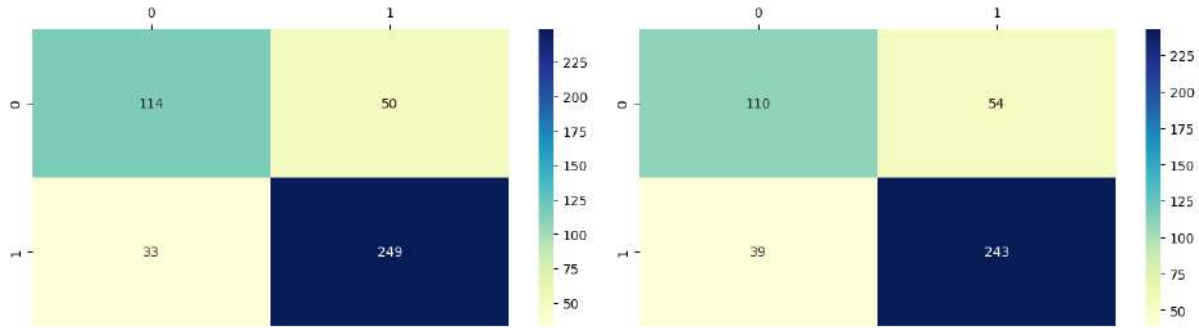
4.4.2 Noun Phrase Bracketing

The detailed results of noun phrase bracketing task have been presented in table 4.4. The results show that the hierarchical vectors generated from word2vec are unable to outperform the baseline in any given metric. However, for the hierarchical vectors generated using GloVe, vector $\mathbf{GO^L}$ has outperformed the baseline in terms of precision, recall, f1-score and overall accuracy. The confusion matrices for the mentioned vectors have been shown in figures 4.9, 4.10 and 4.11.

Vectors		Precision	Recall	f1-Score	Accuracy
Word2Vec		**77.55**	**69.51**	**73.31**	**81.39**
Hierarchical	$\mathbf{WO^A}$	74.80	56.10	64.11	76.91
Overcomplete	$\mathbf{WO^I}$	73.91	62.20	67.55	78.03
Word2Vec	$\mathbf{WO^L}$	76.43	65.24	70.39	79.82
Hierarchical	$\mathbf{WW^A}$	74.45	62.20	67.77	78.25
Weighted	$\mathbf{WW^I}$	74.26	61.59	67.33	78.03
Word2Vec	$\mathbf{WW^L}$	74.45	62.20	67.77	78.25
Glove		73.83	**67.07**	**70.29**	79.15
Hierarchical	$\mathbf{GO^A}$	69.78	59.15	64.03	75.56
Overcomplete	$\mathbf{GO^I}$	74.24	59.76	66.22	77.58
Glove	$\mathbf{GO^L}$	**76.81**	64.63	70.20	**79.82**
Hierarchical	$\mathbf{GW^A}$	69.93	60.98	65.15	76.01
Weighted	$\mathbf{GW^I}$	73.68	59.76	65.99	77.35
Glove	$\mathbf{GW^L}$	68.97	60.98	64.72	75.56

Scores are represented as %

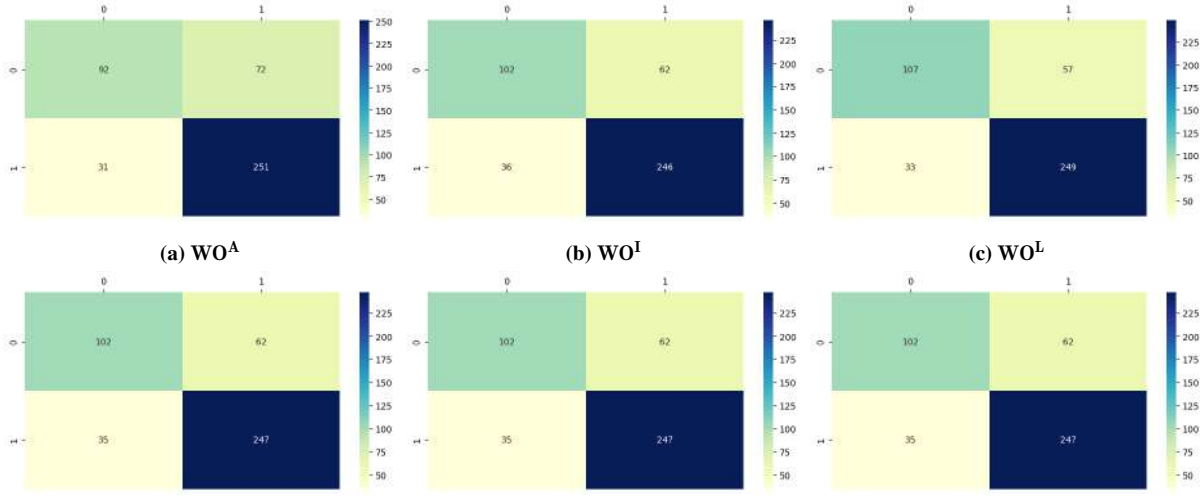
Table 4.4: Performance of base vectors and hierarchical vectors in noun phrase bracketing task.



(a) Bracketing using Word2Vec

(b) Bracketing using gloVe

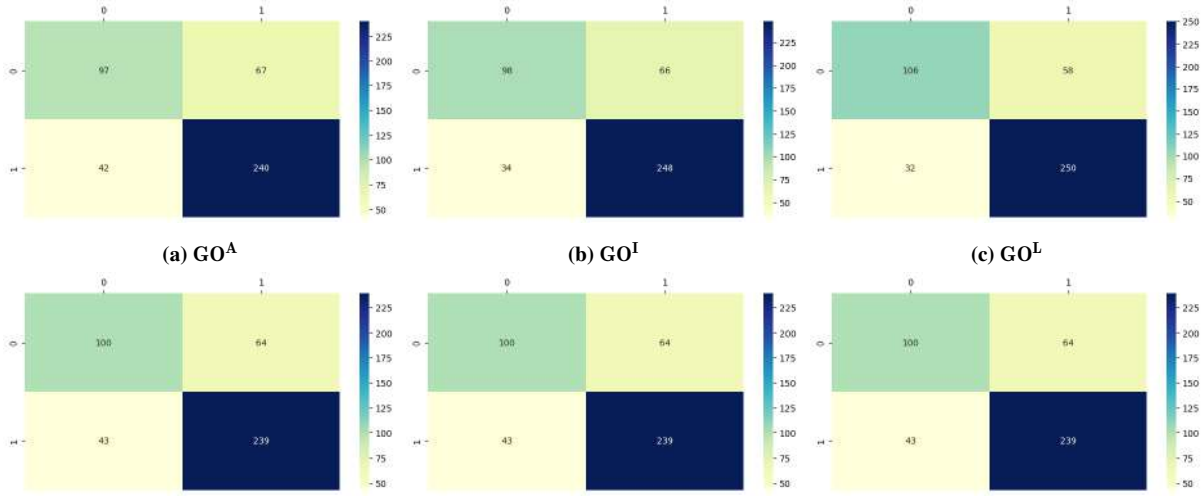
Figure 4.9: Confusion matrix for noun phrase bracketing task using base vectors.



(a) WO^A (b) WO^I (c) WO^L

(d) WW^A (e) WW^I (f) WW^L

Figure 4.10: Confusion matrix for noun phrase bracketing using hierarchical vectors generated from word2vec.



(a) GO^A (b) GO^I (c) GO^L

(d) GW^A (e) GW^I (f) GW^L

Figure 4.11: Confusion matrix for noun phrase bracketing using hierarchical vectors generated from glove.

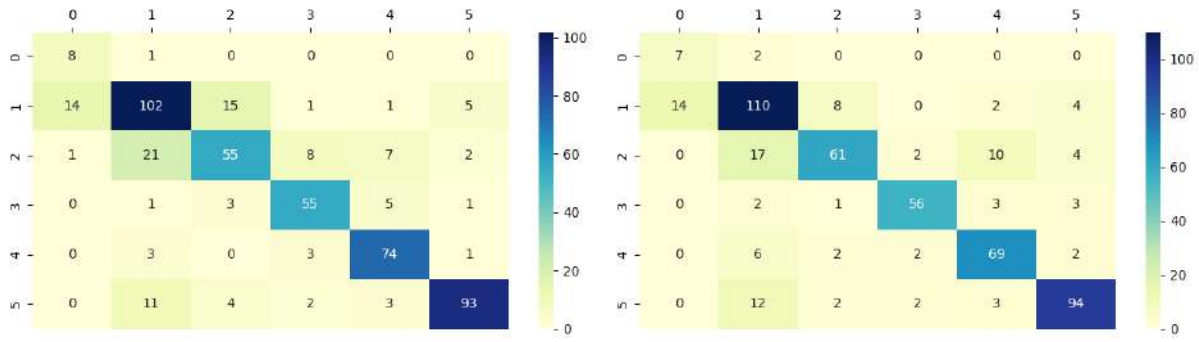
4.4.3 Question Classification (TREC)

The detailed results of question classification task have been presented in table 4.5. The results show that the hierarchical vectors generated from word2vec consistently outperform the baseline in all of the given metrics. It was observed that vector $\mathbf{WO^L}$ displayed the best performance out of all the word2vec descendants. For the hierarchical vectors generated using GloVe, only vector $\mathbf{GO^L}$ was able to outperform the baseline in terms of precision, recall, f1-score and overall accuracy. The confusion matrices for the mentioned vectors have been shown in figures 4.12, 4.13 and 4.14.

Vectors		Precision	Recall	f1-Score	Accuracy
Word2Vec		72.0	80.0	74.0	77.0
Hierarchical	$\mathbf{WO^A}$	78.0	84.0	80.0	83.0
Overcomplete	$\mathbf{WO^I}$	73.0	80.0	75.5	78.0
Word2Vec	$\mathbf{WO^L}$	**81.0**	**85.0**	**82.0**	**84.0**
Hierarchical	$\mathbf{WW^A}$	74.5	82.0	76.0	80.0
Weighted	$\mathbf{WW^I}$	70.5	76.0	71.0	73.0
Word2Vec	$\mathbf{WW^L}$	72.0	80.0	72.5	76.0
Glove		75.0	79.0	75.0	79.0
Hierarchical	$\mathbf{GO^A}$	71.0	75.0	72.0	78.0
Overcomplete	$\mathbf{GO^I}$	74.0	79.0	75.0	79.0
Glove	$\mathbf{GO^L}$	**77.0**	**83.0**	**79.0**	**83.0**
Hierarchical	$\mathbf{GW^A}$	70.0	75.0	71.0	75.0
Weighted	$\mathbf{GW^I}$	70.0	75.0	71.5	76.0
Glove	$\mathbf{GW^L}$	74.0	80.0	75.0	79.0

Scores are represented as %

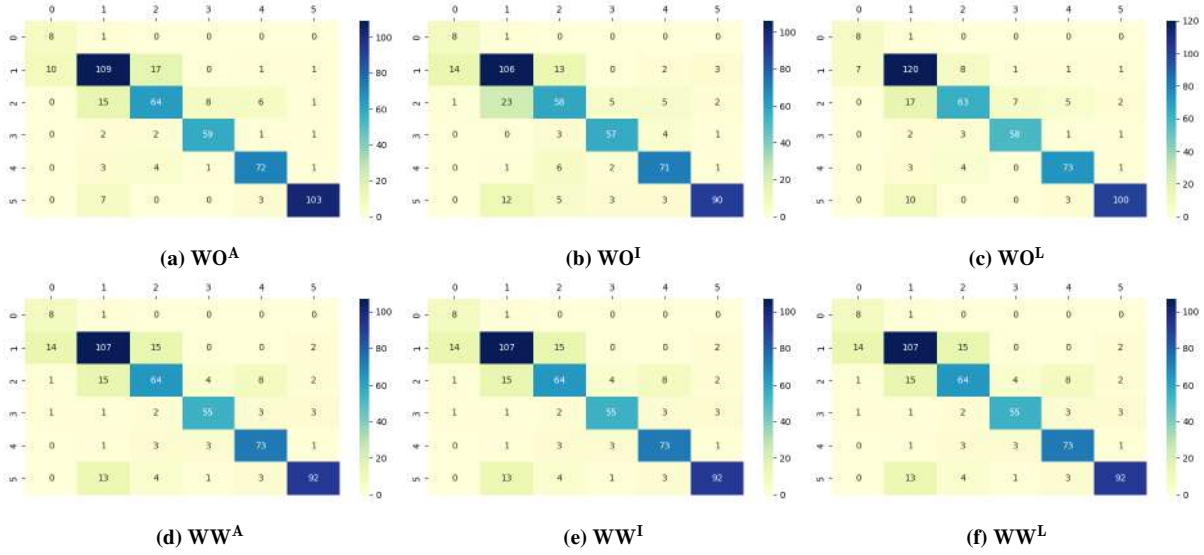
Table 4.5: Performance of base vectors and hierarchical vectors in question classification task.



(a) Question Classification using Word2Vec

(b) Question Classification using gloVe

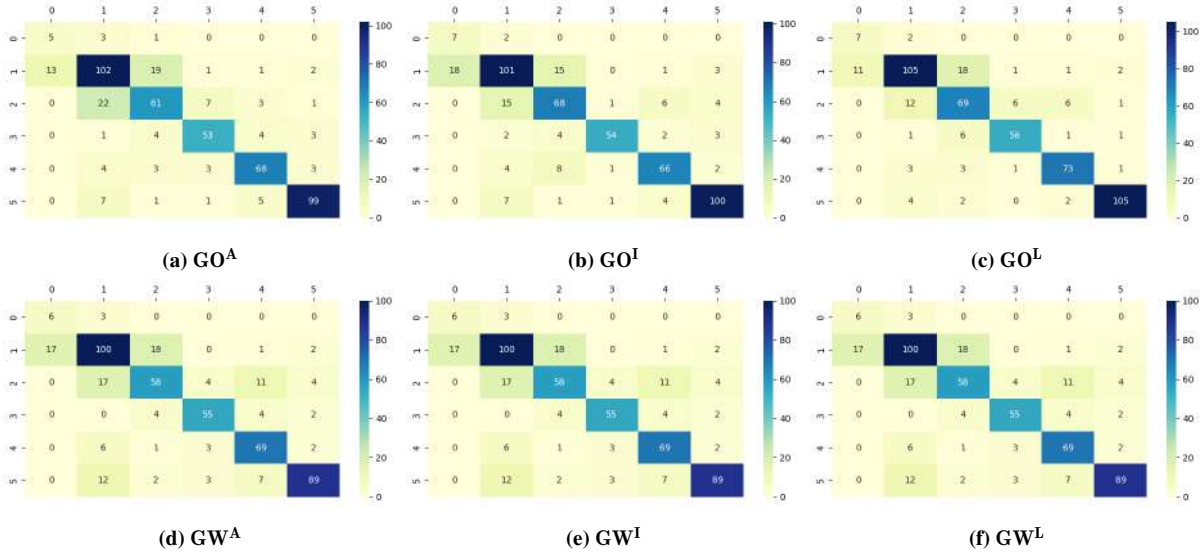
Figure 4.12: Confusion matrix for question classification task using base vectors.



(a) WO^A (b) WO^I (c) WO^L

(d) WW^A (e) WW^I (f) WW^L

Figure 4.13: Confusion matrix for question classification using hierarchical vectors generated from word2vec.



(a) GO^A (b) GO^I (c) GO^L

(d) GW^A (e) GW^I (f) GW^L

Figure 4.14: Confusion matrix for question classification using hierarchical vectors generated from glove.

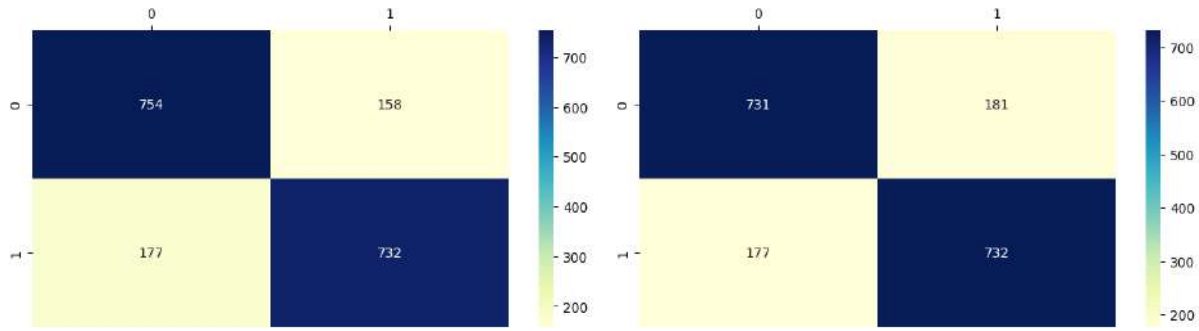
4.4.4 Sentiment Analysis

The detailed results of sentiment analysis task have been presented in table 4.6. The results show that the hierarchical vectors generated from word2vec perform rather poorly in the sentiment analysis task in terms of the given metric. It was observed that hierarchical overcomplete vectors delivered the worst performance in this task for both word2vec and glove bases. For the hierarchical vectors generated using GloVe, hierarchical weighted vectors were able to perform competitively with the baseline in terms of precision, recall, f1-score and overall accuracy. The confusion matrices for the mentioned vectors have been shown in figures 4.15, 4.16 and 4.17.

Vectors		Precision	Recall	f1-Score	Accuracy
Word2Vec		80.99	**82.68**	**81.82**	**81.60**
Hierarchical	WO^A	76.57	75.99	76.28	76.10
Overcomplete	WO^I	77.85	80.15	78.98	78.64
Word2Vec	WO^L	74.84	76.97	75.89	75.51
Hierarchical	WW^A	**81.80**	77.85	79.78	80.23
Weighted	WW^I	81.71	79.82	80.75	80.94
Word2Vec	WW^L	81.65	80.48	81.06	81.16
Glove		**80.51**	80.15	80.33	80.34
Hierarchical	GO^A	74.67	75.08	74.88	74.90
Overcomplete	GO^I	80.21	74.67	77.34	78.09
Glove	GO^L	77.47	75.77	76.61	76.83
Hierarchical	GW^A	79.70	81.36	80.52	80.29
Weighted	GW^I	80.22	81.36	**80.78**	**80.62**
Glove	GW^L	79.28	**82.24**	80.73	80.34

Scores are represented as %

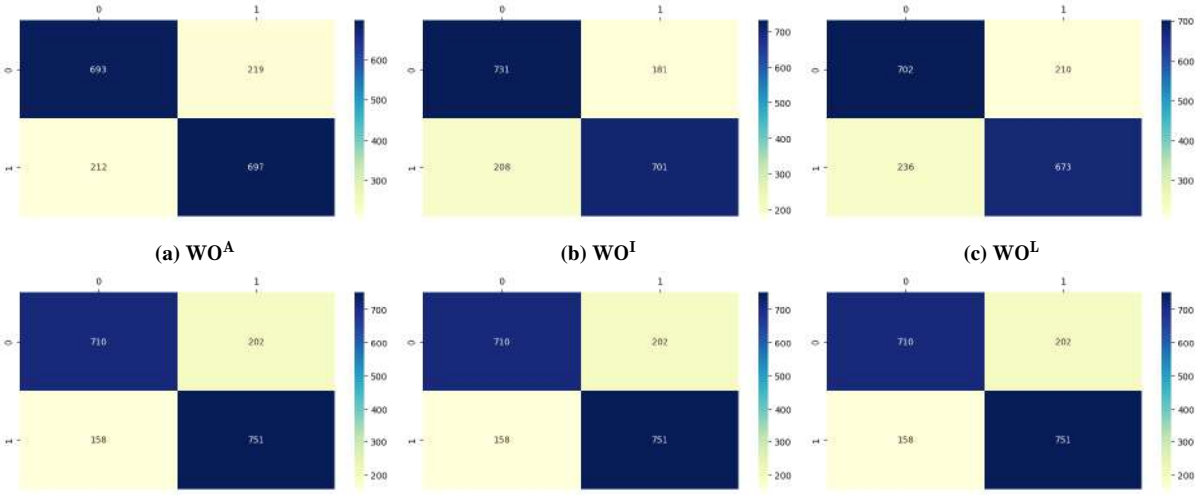
Table 4.6: Performance of base vectors and hierarchical vectors in sentiment classification task.



(a) Sentiment Analysis using Word2Vec

(b) Sentiment Analysis using gloVe

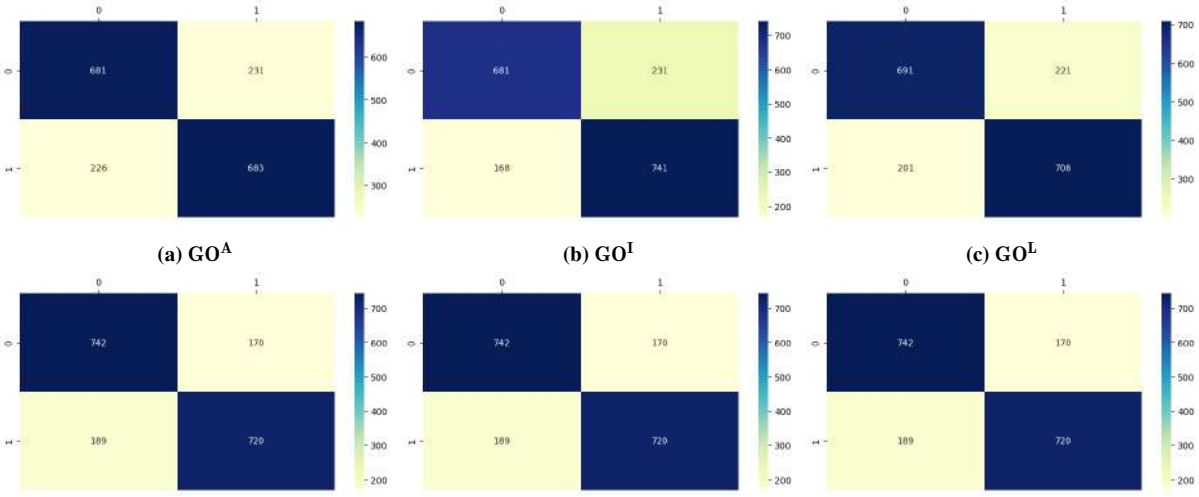
Figure 4.15: Confusion matrix for sentiment analysis task using base vectors.



(a) WO^A (b) WO^I (c) WO^L

(d) WW^A (e) WW^I (f) WW^L

Figure 4.16: Confusion matrix for sentiment analysis using hierarchical vectors generated from word2vec.



(a) GO^A (b) GO^I (c) GO^L

(d) GW^A (e) GW^I (f) GW^L

Figure 4.17: Confusion matrix for sentiment analysis using hierarchical vectors generated from glove.

4.4.5 Overall Accuracy

The results from our various downstream benchmark tasks, spanning from 4.3.1 to the 4.3.5, have been summarized in Table 4.7. Notably, it was observed that the $\mathbf{WO}^{\mathbf{A}}$ vector demonstrates a significant performance boost, particularly excelling in News Classification, outperforming the original Word2Vec vector by a notable margin. However, the Word2Vec vector has maintained its lead in Noun Phrase Bracketing and Capturing Discriminative Attribute tasks. Furthermore, the results from the POLAR framework [38] have been included as to provide an existing baseline, as the works align. The POLAR result indicates higher value between *Word2Vec w/ POLAR* and *GloVe w/ POLAR*.

Vectors		Sports Acc.	Relig. Acc.	Comp. Acc.	NP Acc.	Discr. Acc.	TREC Acc.	Senti. Acc.	Average
Word2Vec		93.97	83.33	76.19	**81.61**	**59.12**	86.8	82.43	80.49
Hierarchical	$\mathbf{WO^A}$	**96.61**	**86.89**	**82.63**	78.25	56.55	89.40	82.21	**81.79**
Overcomplete	$\mathbf{WO^I}$	93.59	85.77	76.71	79.37	59.09	87.60	**82.92**	80.72
Word2Vec	$\mathbf{WO^L}$	95.48	85.77	81.34	80.72	54.41	**89.60**	80.78	81.16
Hierarchical	$\mathbf{WO^A}$	94.22	84.10	75.16	79.60	59.12	86.20	82.48	80.12
Weighted	$\mathbf{WO^I}$	94.10	85.77	75.68	79.15	59.12	87.00	82.92	80.53
Word2Vec	$\mathbf{WO^L}$	94.22	84.10	74.65	79.6	59.12	86.20	82.48	80.05
GloVe		**96.61**	88.28	82.24	**81.39**	65.06	87.20	82.65	83.34
Hierarchical	$\mathbf{GO^A}$	95.35	87.59	**84.68**	77.13	60.96	85.40	82.10	81.89
Overcomplete	$\mathbf{GO^I}$	95.73	88.15	81.34	79.60	**65.11**	88.00	82.58	82.93
GloVe	$\mathbf{GO^L}$	95.73	**88.56**	84.17	80.27	60.96	**90.20**	**83.68**	**83.37**
Hierarchical	$\mathbf{GW^A}$	96.48	**88.56**	80.57	76.46	65.02	84.4	82.52	82.00
Weighted	$\mathbf{GW^I}$	96.23	88.01	80.57	79.82	65.06	87.20	80.36	82.46
GloVe	$\mathbf{GW^L}$	96.11	88.70	76.19	78.70	65.02	84.6	82.48	81.69
POLAR		95.10	85.20	80.20	76.10	63.80	96.40	82.10	82.7

Scores are represented as %

Table 4.7: Performance of the Hierarchical Word Vectors across different downstream tasks.

Interestingly, different vectors have excelled in different tasks. $\mathbf{WO^I}$ has performed well in Question Classification, while $\mathbf{WO^L}$ has led in Sentiment Classification, with $\mathbf{WO^A}$ consistently delivering competitive results. The weighted vectors, namely $\mathbf{WW^A}$, $\mathbf{WW^I}$, and $\mathbf{WW^L}$, have also exhibited commendable performances, comparable to the Word2Vec vectors. Overall, when assessing the collective performance across all downstream tasks, it was evident that $\mathbf{WO^A}$ consistently emerged as the top-performing vector, demonstrating its remarkable effectiveness in a diverse array of NLP tasks. When comparing to similar works, we can see that most hierarchical vectors generated from glove consistently outperform the results of the POLAR vectors.

In this evaluation, the similarity results have been distinguished from other downstream tasks, given that similarity evaluations are often considered a litmus test for the quality of word vectors. The assessment of word similarity relies on the Spearman rank correlation coefficient ρ. From the data presented in Table 4.8, it is apparent that the $\mathbf{WO^A}$, $\mathbf{WO^I}$, and $\mathbf{WO^L}$ vectors have exhibited relatively lower performance compared to the word2vec vectors across all similarity datasets. However, it is crucial to note that word similarity tests primarily serve as an indicator of vector quality. Given the exceptional performance of Hierarchical Vectors on downstream

Vectors		Simlex-999	WS353	WS353-S	WS353-R	MEN	MT-771
Word2Vec		**44.20**	**69.41**	**77.71**	**62.19**	**78.2**	**67.13**
Hierarchical	$\mathbf{WO^A}$	36.89	58.8	69.58	49.79	58.46	51.39
Overcomplete	$\mathbf{WO^I}$	43.94	69.13	77.39	62.00	78.03	66.99
Word2Vec	$\mathbf{WO^L}$	24.00	40.67	52.64	27.60	43.02	25.88
Hierarchical	$\mathbf{WO^A}$	**44.20**	**69.41**	**77.71**	**62.19**	**78.2**	**67.13**
Weighted	$\mathbf{WO^I}$	**44.20**	**69.41**	**77.71**	**62.19**	**78.2**	**67.13**
Word2Vec	$\mathbf{WO^L}$	**44.20**	**69.41**	**77.71**	**62.19**	**78.2**	**67.13**
GloVe		**40.83**	71.24	80.15	**64.43**	80.49	**71.53**
Hierarchical	$\mathbf{GO^A}$	29.81	58.51	71.86	46.19	68.57	56.30
Overcomplete	$\mathbf{GO^I}$	40.45	**71.27**	**80.30**	64.42	**80.52**	71.43
GloVe	$\mathbf{GO^L}$	29.81	58.51	71.86	46.19	68.57	56.30
Hierarchical	$\mathbf{GW^A}$	40.33	71.24	80.15	**64.43**	80.49	71.44
Weighted	$\mathbf{GW^I}$	**40.83**	71.24	80.15	**64.43**	80.49	**71.53**
GloVe	$\mathbf{GW^L}$	40.33	71.24	80.15	**64.43**	80.49	71.44

Scores represent ρ as %

Table 4.8: Performance of the Hierarchical Word Vectors across different word similarity tasks.

tasks, it has been contended that these similarity scores can be deprioritized in favor of the demonstrated efficacy in practical applications.

4.5 Statistical Significance of Benchmark Tests

To demonstrate that the observed results were not a case of chance and to prove the effectiveness of hierarchical vectors, a statistical significance test was performed. As suggested by previous works,[39], 100 runs of random sampling paired with the corrected paired lower-tailed student-t test [40] has been performed to test the significance of the obtained results. This method to check for statistical significance has been used as it considers the ability to replicate results as more important than Type I or Type II errors. Table 4.9 shows the statistical significance of the generated results. We observe that the obtained results are statistically significant for sports classification, religion classification, computer classification and question classification. The scores in bold in table 4.9 indicate the tests for which the drawn results are statistically significant.

The test of statistical significance has been performed to verify the superior performance of hierarchical vectors with respect to the pretrained vectors. This test was performed in two parts, first testing the absolute hierarchical overcomplete word2vec vector against word2vec and

then testing the absolute hierarchical overcomplete glove vector against glove. The null and alternative hypothesis have been described as follows:

Hypothesis:

$$\begin{aligned} \mathbf{H}_0 &: \mu_{word2vec} - \mu_{hierarchical} = 0 \\ \mathbf{H}_1 &: \mu_{word2vec} - \mu_{hierarchical} < 0 \end{aligned} \tag{4.1}$$

	Sports Accuracy	Religion Accuracy	Computer Accuracy	NP Accuracy	TREC Accuracy	Sentiment Accuracy
Word2Vec Mean	95.17	87.63	76.53	82.37	73.56	79.74
Word2Vec S.D.	1.18	2.19	2.35	1.54	1.25	0.89
Hierarchical Mean	97.82	92.09	83.02	81.33	80.90	79.94
Hierarchical S.D.	0.89	1.73	2.22	1.45	1.08	1.01
t-statistic	-2.14	-2.14	-2.34	0.69	-6.07	-0.21
p-value	**0.02**	**0.02**	**0.01**	0.75	**0.00**	0.42
Glove Mean	95.51	87.69	78.41	81.38	72.02	79.80
Glove S.D.	1.18	2.20	2.41	1.65	1.31	0.96
Hierarchical Mean	97.84	91.23	85.73	80.43	78.05	77.82
Hierarchical S.D.	0.91	2.02	1.83	1.57	1.09	0.94
t-statistic	-2.20	-1.49	-2.93	0.66	-4.54	1.94
p-value	**0.02**	**0.06**	**0.00**	0.74	**0.00**	0.97

degree of freedom = 99, level of significance (α) = 10%

Table 4.9: Test for Statistical Significance of the results of the benchmark tests.

4.6 Transformational Effects

4.6.1 Effect of Word List Size

As Syntactic Representations have been built using directions recovered form list of words, the effect of word list size on some benchmark tests have been evaluated. As the number of occurring words per each class is different i.e naturally nouns, verbs, adverbs and adjectives occur more frequently than pronouns, conjunctions, prepositions and interjections, repeated instances of words have been used to build the higher list sizes. The results in figures 4.18, 4.19 and 4.20 show the change in accuracy of respective news classification tasks with respect to word list size. The results surprisingly showed that the change of accuracy was not consistent with the change in size of word list. This phenomenon could possibly be attributed to the larger

pre-trained vector having more control on the hierarchical vector over the smaller Syntactic Representation.

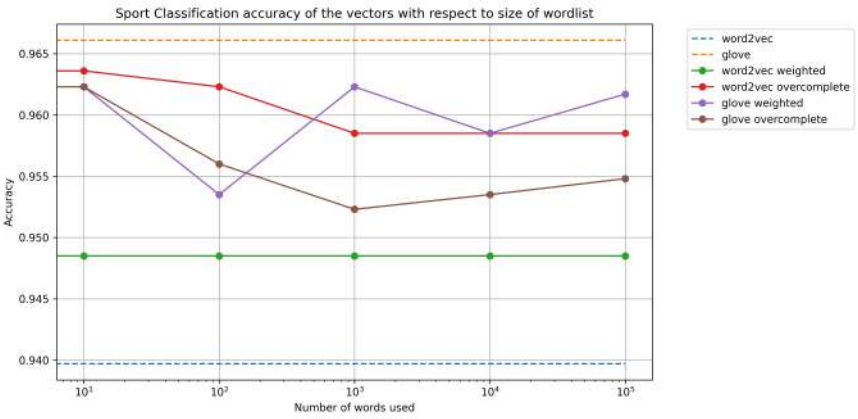


Figure 4.18: Sport Classification Accuracy per size of word list.

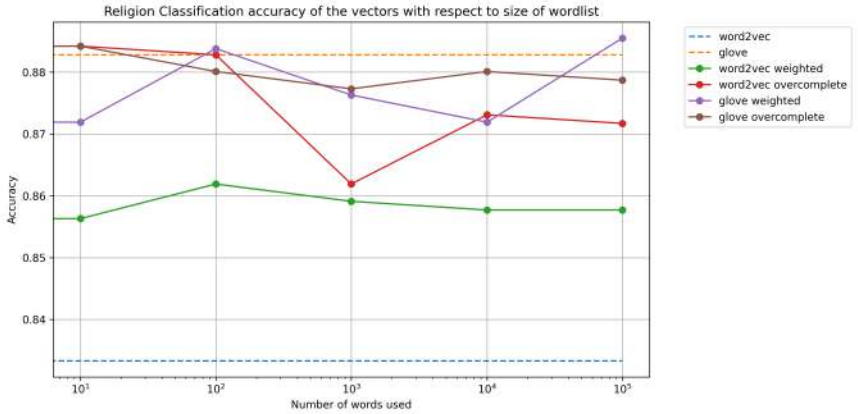


Figure 4.19: Religion Classification Accuracy per size of word list.

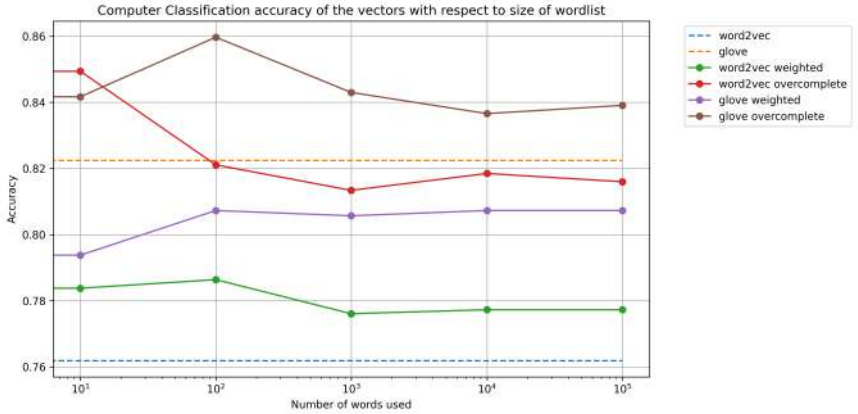
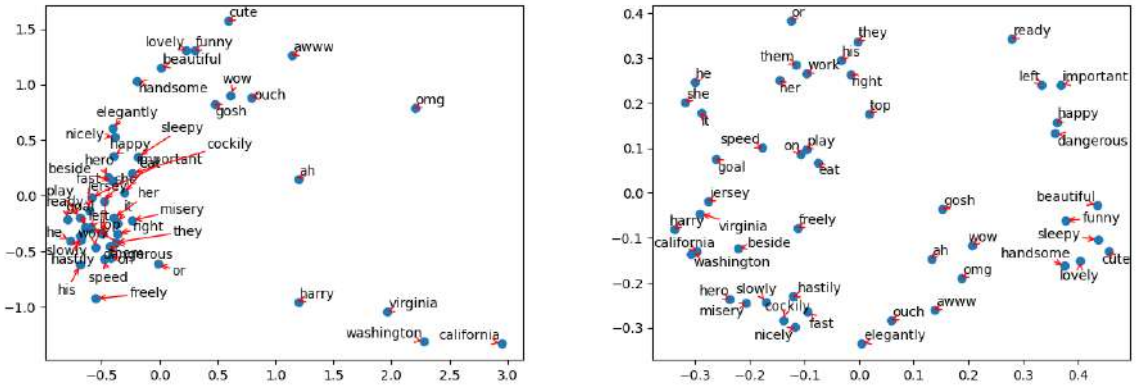


Figure 4.20: Computer Classification Accuracy per size of word list.

4.6.2 Effect on Embedding Space

The basis of syntactic representations and hierarchical vectors define the use of isolated syntactic regularities along with pretrained vectors to generate hierarchical vectors with better syntactic abilities. To verify this, the embedding spaces of both the pretrained vector and the hierarchical vector was subjected to a test of syntactic grouping. Even though, the pretrained vector space was able to group nouns fairly well, the transformed space better represents all parts of speech, as indicated clearly by the intra-group cohesion and inter-group isolation in figure 4.23.



(a) Embedding space of pretrained vector

(b) Transformed Embedding Space of Hierarchical Vector

Figure 4.21: Effect of transformation on the embedding space

4.7 Interpretability

This work has stated that the reduced Syntactic Representations allow the words to be compared along an absolute scale while the coordinates also signify the part of speech of the word. To prove this, the representations were subjected to a classification experiment based on the words in WordNet. Furthermore, following previous work [22], a qualitative analysis was conducted focusing on individual coordinates.

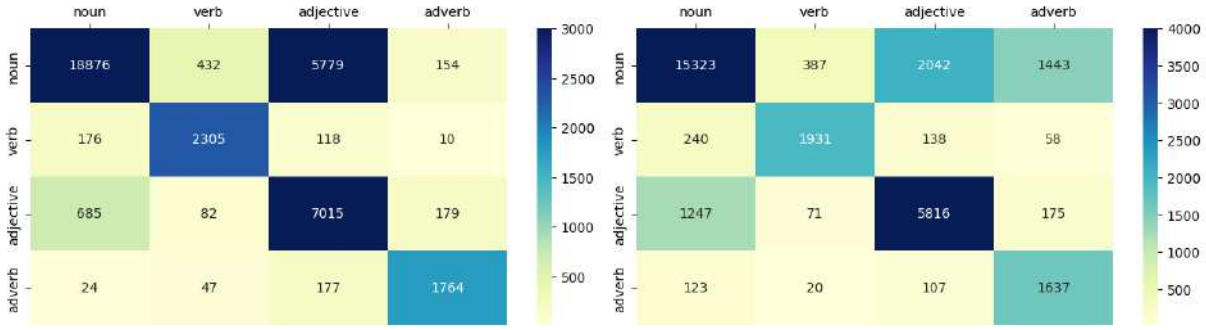
4.7.1 Word Classification

The word classification experiment seeks to examine the extent to which the generated Syntactic Representations are coherent to the word's part of speech. For this experiment, all words present in WordNet were taken and their respective *Interpretable Syntactic Representations* were generated. Following this, the predicted labels of each word was assigned as the coordinate with maximum value as shown by the visual interpretation. After assigning the predicted labels, the classification experiment were conducted in two modes: Partial and Complete. The first, as shown in figures 4.22a and 4.22b, were conducted on words from WordNet which belonged to exactly one part of speech.

It was observed that the representations were able to interpret the words in terms of their part of speech with an accuracy of **79.21% for Glove** and **80.33% for Word2vec** in the partial dataset. Following the first classification, the second classification was conducted for all words in WordNet regardless of their association to a particular part of speech. The results can be seen in figures 4.23a and 4.23b. The classification results for the complete dataset was **73.58% for Glove** and **73.00% for Word2vec**.

Dataset	Pos	Word2Vec			Glove		
		Precision	Recall	F1-score	Precision	Recall	F1-score
Partial	Noun	0.80	0.90	0.85	0.75	0.96	0.84
	Verb	0.82	0.80	0.81	0.88	0.80	0.84
	Adjective	0.80	0.72	0.75	0.88	0.54	0.67
	Adverb	0.87	0.49	0.63	0.88	0.84	0.86
Accuracy		**80.33%**			79.21%		
Complete	Noun	0.76	0.82	0.79	0.72	0.88	0.79
	Verb	0.56	0.66	0.60	0.62	0.68	0.65
	Adjective	0.74	0.67	0.70	0.81	0.52	0.63
	Adverb	0.81	0.45	0.58	0.82	0.82	0.82
Accuracy		73.00%			**73.58%**		

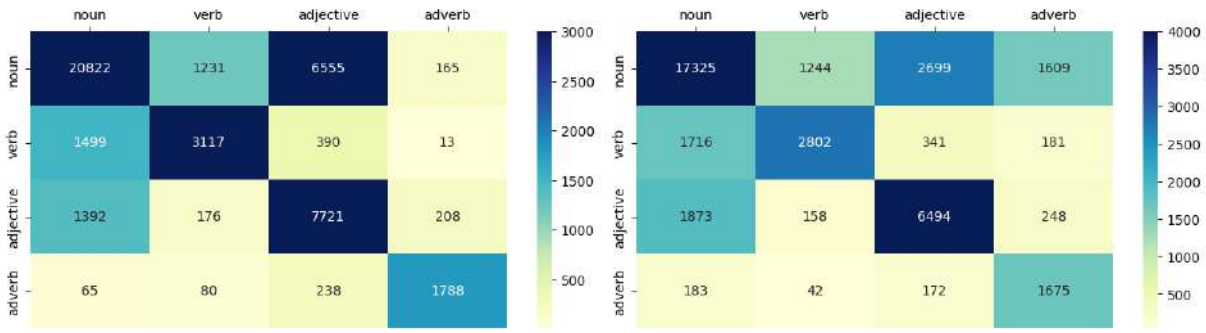
Table 4.10: Performance of the Interpretable Syntactic Representations on word classification task.



(a) Partial Word Classification for `Glove`

(b) Partial Word Classification for `word2vec`

Figure 4.22: Confusion matrix for partial word classification



(a) Complete Word Classification for `Glove`

(b) Complete Word Classification for `word2vec`

Figure 4.23: Confusion matrix for complete word classification

4.7.2 Effect of Size of Word List

The effect of size of Syntactic Representation on Interpretability have been presented in figure 4.24. It can clearly be seen that the Interpretable Syntactic Representations drawn from a larger word list preform better than their counterparts drawn from smaller subsets.

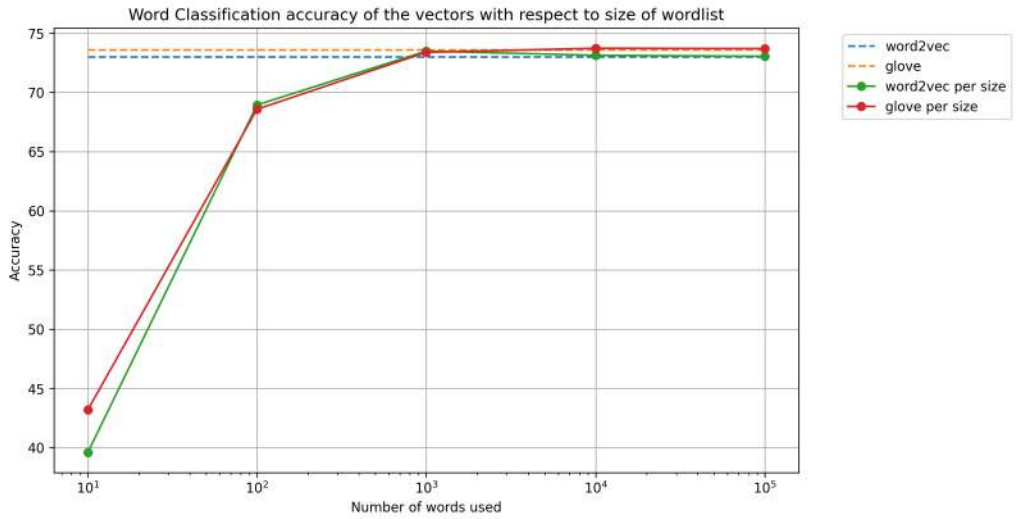


Figure 4.24: Word Classification Accuracy per size of word list.

4.7.3 Qualitative Evaluation of Interpretability

A qualitative evaluation of interpretability was performed following previous works.[19][21] Given a vector with interpretable dimensions, it is necessary that the top-ranking words for that dimension exhibit semantic or syntactic groupings. As this work has detailed isolating syntactic regularities from pre-trained representations, this evaluation was performed to test for syntactic grouping of the top-ranking words along all coordinates. The results have been shown in table 4.11.

Dimension	Top Ranking Words
Noun	vpnvpn, northbrook, florida, michigan, maryland, oregon, nj, illinois, missouri, washington, virginia, carolina, california, georgia, ohio, william, pennsylvania, indiana, ontario, myrtle, nc, colorado, arizona, arkansas, tennessee, kentucky, louisiana, chicago, wisconsin, joseph
Verb	personlise, journal, profile, repairs, reorganize, demoralize, marginalize, disassemble, retool, meshothelioma, list, decompile, obfuscate, homogenize, sensationalize, ninotchka, regroup, radicalize, dismantle, modernize, bulldoze, polarize, reprioratize, marginalize, revitalise, agitate, flatten
Adjective	sporophores, leptonic, savourest, scarious, formless, unspiritual, unipotent, unformed, infiltrative, everstarving, kidisms, scapegoatting, undeformed, unworldly, noncollinar, hyperplastic, automorphic, nonsubstantive, facefuls, ishybadboy, exophytic, nondegenerate, unmanifested, nonconceptual, undifferentiated, passionless, distichous, unauthoritative
Adverb	jacketss, agitatedly, moodily, ponderously, conspiratorially, inelegantly, cockily, somberly, primly, mournfully, expansively, indelicately, enigmatically, regally, bmtron, contemplatively, snarkily, wittily, bemusedly, pensively, lasciviously, companionably, sassily, fetchingly, jauntily, lightheartedly, loftily, coquettishly, cheesily
Pronoun	unto, hath, believeth, saith, thy, heareth, knoweth, receiveth, thou, thine, whosoever, doeth, thyself, keepeth, saidst, loveth, thee, ye, seeth, liveth, trusteth, speaketh, knowest, hast, shouldest, seeketh, sinneth, hateth, teachest, begotten, bareth, dwelleth, committeth, shalt, calleth, judgest, whomsoever, asketh, believest
Conjunction	checkour, checkthese, nevertheless, additionally, nonetheless, chymbers, lynters, however, tyrryts, chyrits, also, specialially, certainly, tends, although, appealing, often, inmb, usually, phraseactivities, consequently, therefore, generally, furthermore, enlarement, tend, definately, greatly
Preposition	across, between, toward, million, around, over, billion, from, through, approximately, onto, towards, ranging, per, into, spanning, in, including, covering, miles, during, within, acres, extending, of, atop, five, reaching, eight, annual, six, percent, erupted, amid, totaling, four, against, rolling, via
Interjection	ahhh, ahh, hahaha, awww, ohh, oooh, hahahaha, haha, ohhh, omg, hahah, gosh, ooh, awwww, yay, woah, ahhhh, hahahahaha, hehe, lmao, ohhhh, hah, hahahah, oh, hehehe, whoa, lol, ooooh, soo, aww, geez, ahahaha, yah, gah, huh, ah, sooo, ahhhhh, heh, ahaha, yeah, hooo, ahah, awwwww

Table 4.11: Top-ranked words per dimension for Interpretable Syntactic Representations.

As shown in table 4.11, it is observed that the representations were able to produce excellent syntactic groupings and the produced groups are coherent with the built dimension. Previous works were able to present qualitative analysis using semantic groupings without any labels,[19] however this work uses a coarse-grained approach to produce better interpretations based on predetermined labels. The words in the table was extracted from the absolute syntactic representation model. Apart from a few random words in the model vocabulary, the part of speech dimensions were generally able to incorporate words which aligned with human judgement of the eight parts of speech.

Chapter 5: Conclusion

5.1 Conclusion

The main objective of this thesis was to add interpretablility to otherwise dense word embeddings by introducing the notion of Syntactic Representations. Syntactic representations were presented as a metarepresentation to the underlying word embeddings, presenting a reduced form, in terms of eight parts of speech. Furthermore, this thesis has used the aforementioned Syntactic Representations to create Hierarchical vectors attempting to emulate the incremental model of representations. The results of the benchmark tests and interpretability experiments, as demonstrated, have been convincing of the usability of these representations.

With respect to the performance in benchmark tasks, the Hierarchical Vectors outperformed their respective base vectors in multiple tasks. This superior performance of these vectors was accompanied by their statistical significance. The results of the statistical significance tests proved that these vectors were able to significantly outperform their base forms. Furthermore, the effect on the embedding space visually demonstrated the enhanced syntactic abilities of the hierarchical vectors. A curious case related to the change of accuracy with respect to size of word list was also observed. The accuracy seemed to remain constant even when the size of word list was increased exponentially. This phenomenon was credited to the pretrained vector possibly having more control on the Hierarchical vector than the Syntactic Representation. At the same time, the test on Syntactic Representations demonstrated an increase in accuracy with the increase in size of the word list.

The interpretability experiments for Syntactic Representations were also able to prove that the representations could distinguish between parts of speech in a manner that aligned with human judgement. The Qualitative evaluation of Interpretability further provided supporting arguements for the effective interpretation produced by the Syntactic Representations. With these evaluations, this thesis concludes that post-processing pretrained word embeddings to produce metarepresentations was able to make the underlying vectors interpretable while outperforming the original vectors in benchmark tests.

5.2 Limitations

In the duration of the thesis, various limitations were faced. Some of them have been discussed below:

- As the oracle used for this thesis, WordNet, was constrained to only four parts of speech, i.e Noun, Verb, Adjective and Adverb, the work was done on a rough grained basis. The

words for the remaining four parts of speech were manually added and it was not feasible to extend the parts of speech to cover modals, determiners, verb forms, etc.

- The inherent lack of polysemy in the used embedding generators restricted the Hierarchical Vectors. Even though the Syntactic Representations were able to provide the probability of being a certain part of speech, the lack of polysemy led to them being static in nature.

5.3 Future Works

In accordance with the novel notion presented by this thesis, the following works can be contemplated:

- Generation of sense aware Syntactic Representations and consequently sense aware Hi-erarchical Vectors.
- For low resource languages like Nepali, parts of speech labeled dataset can be used to calculate the change of basis matrix and further generate Syntactic Representations.

References

[1] M. T. Ribeiro, S. Singh, and C. Guestrin, “" why should i trust you?" explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[2] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.

[3] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Advances in neural information processing systems*, vol. 26, 2013.

[5] M. E. Peters, M. Neumann, M. Iyyer, *et al.*, *Deep contextualized word representations*, 2018. arXiv: 1802.05365 [cs.CL].

[6] N. A. Smith, “Contextual word representations: A contextual introduction,” *arXiv preprint arXiv:1902.06006*, 2019.

[7] “Regulation (eu) 2016/679 of the european parliament and of the council,” *Regulation (eu)*, vol. 679, p. 2016, 2016.

[8] J. Turian, L. Ratinov, and Y. 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*, 2010, pp. 384–394.

[9] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.

[10] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American society for information science*, vol. 41, no. 6, pp. 391–407, 1990.

[11] K. Lund and C. Burgess, “Producing high-dimensional semantic spaces from lexical co-occurrence,” *Behavior research methods, instruments, & computers*, vol. 28, no. 2, pp. 203–208, 1996.

[12] Y. Bengio, R. Ducharme, and P. Vincent, “A neural probabilistic language model,” *Advances in neural information processing systems*, vol. 13, 2000.

[13] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, “A neural probabilistic language model,” *J. Mach. Learn. Res.*, vol. 3, no. null, 1137–1155, 2003, ISSN: 1532-4435.

[14] F. Morin and Y. Bengio, “Hierarchical probabilistic neural network language model,” in *International workshop on artificial intelligence and statistics*, PMLR, 2005, pp. 246–252.

[15] A. Mnih and G. Hinton, "Three new graphical models for statistical language modelling," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 641–648.

[16] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167.

[17] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 2013, pp. 746–751.

[18] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," *arXiv preprint arXiv:1705.00108*, 2017.

[19] M. Faruqui, Y. Tsvetkov, D. Yogatama, C. Dyer, and N. Smith, "Sparse overcomplete word vector representations," *arXiv preprint arXiv:1506.02004*, 2015.

[20] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural computation*, vol. 12, no. 2, pp. 337–365, 2000.

[21] A. Subramanian, D. Pruthi, H. Jhamtani, T. Berg-Kirkpatrick, and E. Hovy, "Spine: Sparse interpretable neural embeddings," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[22] A. Panigrahi, H. V. Simhadri, and C. Bhattacharyya, "Word2sense: Sparse interpretable word embeddings," in *Proceedings of the 57th annual meeting of the Association for Computational Linguistics*, 2019, pp. 5692–5705.

[23] M. Berger, "Visually analyzing contextualized embeddings," in *2020 IEEE Visualization Conference (VIS)*, IEEE, 2020, pp. 276–280.

[24] Z. J. Wang, F. Hohman, and D. H. Chau, "Wizmap: Scalable interactive visualization for exploring large machine learning embeddings," *arXiv preprint arXiv:2306.09328*, 2023.

[25] B. Mathew, S. Sikdar, F. Lemmerich, and M. Strohmaier, "The polar framework: Polar opposites enable interpretability of pre-trained word embeddings," in *Proceedings of The Web Conference 2020*, 2020, pp. 1548–1558.

[26] J. Engler, S. Sikdar, M. Lutz, and M. Strohmaier, "Sensepolar: Word sense aware interpretability for pre-trained contextual word embeddings," *arXiv preprint arXiv:2301.04704*, 2023.

[27] G. A. Miller, "Wordnet: A lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[28] A. Ben-Israel and T. N. Greville, *Generalized inverses: theory and applications*. Springer Science & Business Media, 2003, vol. 15.

[29] A. Lazaridou, E. M. Vecchi, and M. Baroni, "Fish transporters and miracle homes: How compositional distributional semantics can help np parsing," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1908–1913.

[30] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," 1993.

[31] A. Krebs, A. Lenci, and D. Paperno, "Semeval-2018 task 10: Capturing discriminative attributes," in *Proceedings of the 12th international workshop on semantic evaluation*, 2018, pp. 732–740.

[32] X. Li and D. Roth, "Learning question classifiers," in *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

[33] R. Socher, A. Perelygin, J. Wu, *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[34] F. Hill, R. Reichart, and A. Korhonen, "Simlex-999: Evaluating semantic models with (genuine) similarity estimation," *Computational Linguistics*, vol. 41, no. 4, pp. 665–695, 2015.

[35] L. Finkelstein, E. Gabrilovich, Y. Matias, *et al.*, "Placing search in context: The concept revisited," in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 406–414.

[36] E. Bruni, N.-K. Tran, and M. Baroni, "Multimodal distributional semantics," *Journal of artificial intelligence research*, vol. 49, pp. 1–47, 2014.

[37] G. Halawi, G. Dror, E. Gabrilovich, and Y. Koren, "Large-scale learning of word relatedness with constraints," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 1406–1414.

[38] B. Mathew, S. Sikdar, F. Lemmerich, and M. Strohmaier, "The polar framework: Polar opposites enable interpretability of pre-trained word embeddings," in *Proceedings of The Web Conference 2020*, 2020, pp. 1548–1558.

[39] R. R. Bouckaert and E. Frank, "Evaluating the replicability of significance tests for comparing learning algorithms," in *Pacific-Asia conference on knowledge discovery and data mining*, Springer, 2004, pp. 3–12.

[40] C. Nadeau and Y. Bengio, "Inference for the generalization error," *Advances in neural information processing systems*, vol. 12, 1999.

7th International Conference on
Inventive Computation Technologies
ICICT 2024
24-26, April 2024 ~ Lalitpur, Nepal
icict.conf@gmail.com
www.icicts.com/2024

Letter of Acceptance

Author Name: **Biraj Silwal, Sanjeevan Satyal, Shashidhar Ram Joshi**

Affiliation Details: **Department of Electronics and Computer Engineering**
Institute of Engineering - Pulchowk Campus, Kathmandu, Nepal

Dear Author:

It is with great pleasure that we extend our warmest congratulations to you on the acceptance of the paper titled **" Adding Interpretability to Word Embeddings using Word Metarepresentations" - PAPER ID: ICICT-046"** for presentation at the 7th International Conference on Inventive Computation Technologies, scheduled to be held in Lalitpur, Nepal, from April 24th to April 26th, 2024.

Your submission was subjected to a rigorous review process, and the result that your paper has been selected for inclusion in our conference program. We believe that your contribution will greatly enrich the discussions and knowledge exchange at our event.

Your participation will undoubtedly contribute to the success of the 7th International Conference on Inventive Computation Technologies.

Once again, congratulations on your acceptance, and we anticipate your valuable contribution to our conference.

Prof. Dr. Subarna Shakya
Conference Chair - ICICT 2024

Proceedings by
IEEE