Developing A Bangla WordNet: The Word Clustering Approach

Thesis · September 2018

CITATIONS
0

READS
103

3 authors:

Nafisa Nowshin
Shahjalal University of Science and Technology
3 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Zakia Ritu
Shahjalal University of Science and Technology
3 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Md Mahadi Hasan Nahid
Shahjalal University of Science and Technology
16 PUBLICATIONS **43** CITATIONS
SEE PROFILE

Shahjalal University of Science and Technology
Department of Computer Science and Engineering

Developing A Bangla WordNet: The Word Clustering Approach

Nafisa Nowshin

Reg. No.: 2013331033

4^{th} year, 2^{nd} Semester

Zakia Sultana Ritu

Reg. No.: 2013331045

4^{th} year, 2^{nd} Semester

Department of Computer Science and Engineering

Supervisor

Md Mahadi Hasan Nahid

Lecturer

Department of Computer Science and Engineering

Shahjalal University of Science and Technology

Sylhet - 3114, Bangladesh

September 8, 2018

Developing A Bangla WordNet: The Word Clustering Approach

A Thesis submitted to the

Department of Computer Science and Engineering

Shahjalal University of Science and Technology

Sylhet - 3114, Bangladesh

in partial fulfillment of the requirements for the degree of

Bachelor of Science in Computer Science and Engineering

By

Nafisa Nowshin

Reg. No.: 2013331033

4^{th} year, 2^{nd} Semester

Zakia Sultana Ritu

Reg. No.: 2013331045

4^{th} year, 2^{nd} Semester

Department of Computer Science and Engineering

Supervisor

Md Mahadi Hasan Nahid

Lecturer

Department of Computer Science and Engineering

September 8, 2018

Recommendation Letter from Thesis Supervisor

The thesis entitled

” Developing A Bangla WordNet: The Word Clustering Approach ”

submitted by the students

1. Nafisa Nowshin, 2013331033
2. Zakia Sultana Ritu, 2013331045

is a record of research work carried out under my supervision and I, hereby, approve that the report be submitted in partial fulfillment of the requirements for the award of their Bachelor Degrees.

Signature of the Supervisor:

Name of the Supervisor: Md Mahadi Hasan Nahid

Date: September 8, 2018

Certificate of Acceptance of the Thesis

The thesis entitled

" Developing A Bangla WordNet: The Word Clustering Approach"

submitted by the students

1. Nafisa Nowshin, 2013331033
2. Zakia Sultana Ritu, 2013331045

on September 8, 2018

is, hereby, accepted as the partial fulfillment of the requirements for the award of their Bachelor Degrees.

Head of the Dept.	**Chairman, Exam. Committee**	**Supervisor**
Dr Mohammad Reza Selim	Dr Mohammad Reza Selim	Md Mahadi Hasan Nahid
Professor & Head	Professor	Lecturer
Department of Computer Science and Engineering	Department of Computer Science and Engineering	Department of Computer Science and Engineering

Abstract

In this thesis report, we are proposing a method of constructing a Bangla WordNet. A WordNet can be described as a semantic network of words where all the words of a language are connected with each other through semantic relations. This database is derived from various sources. The source used by us is a Bangla corpus constructed from sources like Bangla wikipedia pages, Bangla online newspaper articles etc. Each WordNet groups word meanings in different ways depending on the construction method. The method we are proposing mainly focuses on the relationship of words having the same meaning and being used in a sentence in place of one another. WordNet has many scopes of improving and contributing to many NLP related works like search engines and information retrieval systems, word sense disambiguation, text mining, automatic text classification, automatic text summarization etc.

Keywords: Natural Language Processing(NLP) , machine learning, deep learning, neural network, word cluster, word2vec.

Acknowledgements

We would like to thank the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet 3114, Bangladesh, for supporting this research. We are also grateful to numerous authors of previous works for their cooperation and support.

We would like to express our heartiest gratitude to our advisor Md Mahadi Hasan Nahid for the constant support and inspiration he provided us for our Bachelor Thesis study and research. His patience, motivation, supervision and vast knowledge were our thorough guide till the end.

We also want to mention another name, Sabir Ismail sir, for his outstanding guide and support. He is an inspiration to us. He guided us, helped us, and mostly kept us motivated always. A very special thanks to him.

Dedication

We would like to dedicate our research to our parents. We are also grateful to anonymous authors of previous works for their co-operation and support.

Contents

Abstract I

Acknowledgement II

Dedication III

Table of Contents IV

List of Tables VI

List of Figures VII

1 Introduction **1**

1.1 Introduction 1

1.2 Motivation 4

1.3 Report Structure 5

2 Background Study **6**

2.1 Literature Review 6

2.2 WordNets In Other Languages 11

2.3 Uses of WordNet 13

3 Methodology **15**

3.1 Data Collection 16

3.2 Previous approach on word embedding 17

3.3 Our approach 18

3.3.1 Vector representation of words 18

3.3.2 Pre-processing steps 19

3.3.3 The word2vec model 20

3.3.4 FastText Model . . . 23
3.3.5 Dictionary Parsing . . . 24
3.3.6 Hierarchy Building . . . 25
3.3.7 Adding Details to Hierarchy Structure . . . 26

4 Result Analysis **28**
4.1 Experiment I: Word2vec in Tensorflow . . . 29
4.2 Experiment II: Word2vec from Gensim package (Skip-gram model) . . . 30
4.3 Experiment II: Word2vec from Gensim package (CBOW model) . . . 31
4.4 Experiment III: FastText Skip-gram model . . . 32
4.5 Experiment III: FastText CBOW model . . . 33
4.6 Training Time . . . 34
4.7 Comparing The Word Embedding Models . . . 35
4.8 Hierarchy Building . . . 35
4.9 Adding Details to Hierarchy Structure . . . 36

5 Discussion **38**
5.1 Discussion . . . 38

6 Conclusion **39**

References **40**

Appendix **42**

A Paper Published on Previous Work **43**

List of Tables

2.1 Clusters formed using N-gram approach[1] 9

3.1 Details of the Corpus 17

4.1 Parameter Tuning for Optimum Results 28
4.2 Results from Word2vec in Tensorflow 29
4.3 Results from Word2vec from Gensim package (Skip-gram model) 30
4.4 Results from Word2vec from Gensim package (CBOW model) 31
4.5 Results from FastText Skip-gram model 32
4.6 Results from FastText CBOW model 33
4.7 Training Time of the Experiments 34

List of Figures

1.1 Sample structure of a English WordNet 2
1.2 Sample structure of a Bangla WordNet 3

2.1 Block diagram of WordNet system[2] 7
2.2 Proposed method for BanglaNet[3] 8
2.3 Linked Indo WordNet structure[4] 12

3.1 Histogram of Most Frequent Words with Number of Occurrences 16
3.2 Vector representation of a text document 19
3.3 Example of Hierarchy of Word relations 25
3.4 Mapping with dictionary 26

4.1 Training Time 34
4.2 Hierarchy of Word relations along with cosine similarity 36
4.3 Hierarchy of Word relations along with cosine similarity 37

Chapter 1

Introduction

1.1 Introduction

Bangla is a major world language. And it will only grow more important in the years to come with the increase in Bangla speaking people all over the globe. Bangla is currently the 7th most spoken language[5] around the world. As the importance of this language grows so does the research works concerning Bangla language. In this era of digital development, more and more focus is being given to digital development of languages and natural language processing is given much importance in the field of computing and research. In the case of Bangla language, many research works are being conducted on various branches of natural language processing with the goal of digitalization and preservation of Bangla language. Many of these research works depend on the availability of digital resources of the language. These resources include well balanced and available monolingual and parallel corpus, dictionary etc. although Bangla is a widely spoken language, but its resources are not as rich as they should be. So much attention is now being given to the construction and development of these resources.

A WordNet can be a very powerful resource for any language. The concept of WordNet was first introduced by Princeton University. They developed the WordNet for the English language, which is now known as the Princeton WordNet[6]. WordNet is a large lexical database of English language. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept and provides short definitions and usage examples. These synsets are interlinked by means of conceptual-semantic and lexical relations. This results in

a network of meaningfully related words and concepts. WordNet's structure makes it a useful tool for computational linguistics and natural language processing. An important aspect of the WordNet is, it not only interlinks just word forms but specific senses of words. The English WordNet includes and reflects all types of relations between words. It is constructed based on both similarity and antonymity of words. We can visualize it with the help of the figure below.

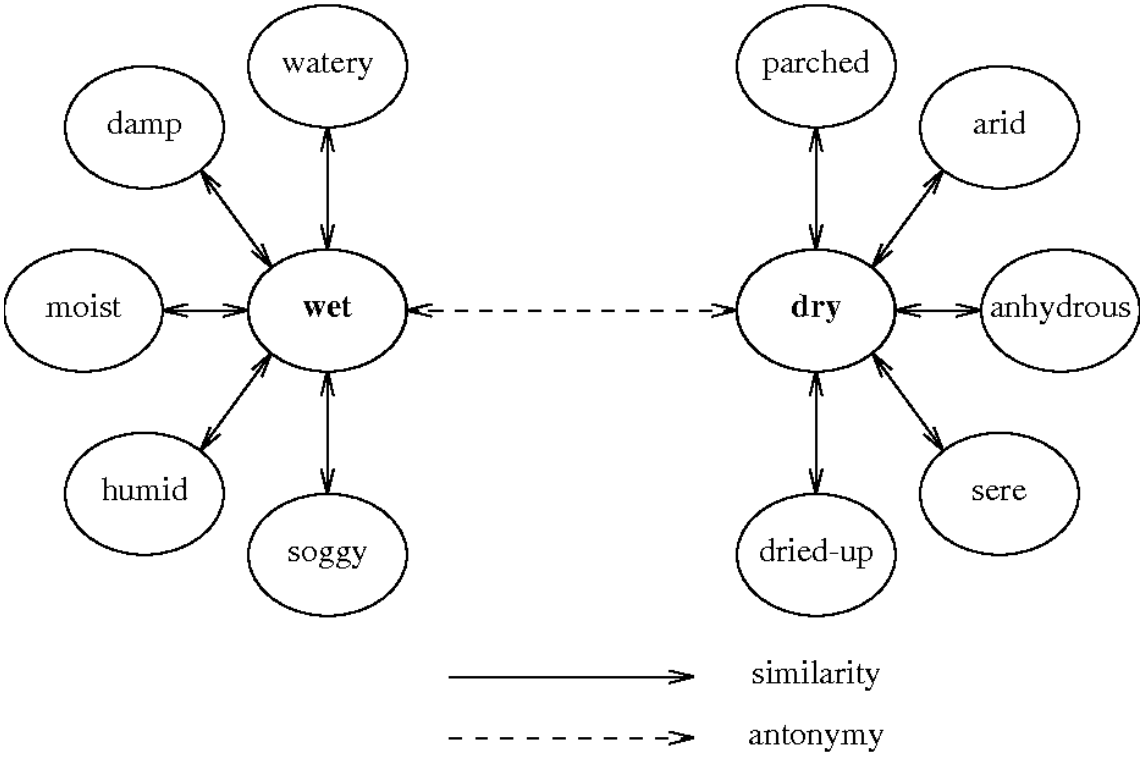

Figure 1.1: Sample structure of a English WordNet

But as the construction of a complete WordNet in a new language is a huge and challenging work, we are focusing on constructing the Bangla WordNet based on only the semantic relationship between words. We are trying to connect words with other words that have the same meaning as it and can be used in a sentence in place of one another. So, our target is to construct a WordNet that is connected via synonyms of words. There are many research works focusing on construction of WordNets on different languages all over the world. But in case of Bangla, although some attempts have been made to develop a small prototype, no complete Bangla WordNet has been built yet. The attempts made so far for this includes translating English words in the Princeton WordNet to Bangla and mapping them to construct the network. A Bangla WordNet will enrich the Bangla language

platform. The method we have proposed in this report features a new approach to Bangla WordNet construction. It basically starts with the root word, connects its variations and gradually building this way it connects the synonyms.

To build a Bangla WordNet, there is a lot of pre processing to do. First, a large dataset covering topics of vast areas is needed to find semantic relations between words. We have collected a dataset for this purpose and applied dynamic word embedding models to construct word embeddings. Different dynamic word embedding models were applied and results were compared to choose an appropriate model for constructing the word clusters that will make the WordNet structure. Then through dictionary parsing the details of the words like meaning, definition, parts of speech etc were assigned to the connected words present in the WordNet structure.

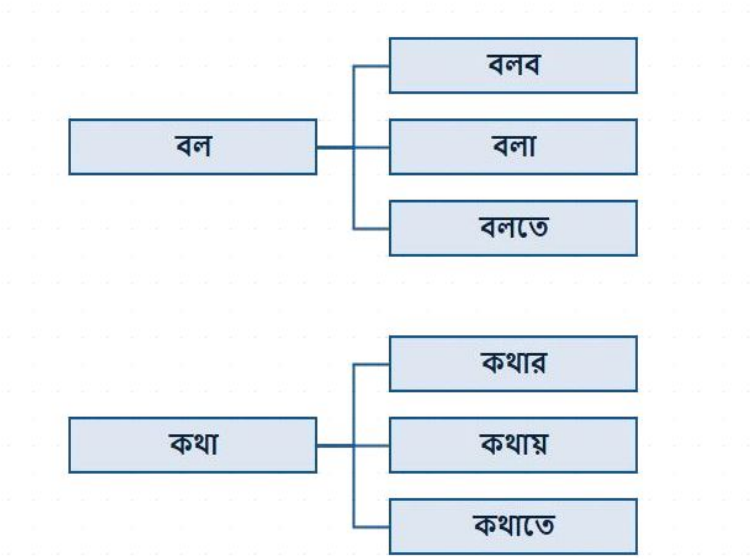

Figure 1.2: Sample structure of a Bangla WordNet

Comparing both the figures of the English and Bangla WordNet given here, some differences can be noticed. That is because the little difference in the construction methods of both WordNets. Also our WordNet does not contain antonyms of words yet and we started working with root words and built up from there but the English WordNet does not connect root words rather the synonyms of its variations.

A WordNet in itself is a huge and important resource for any language. But its importance does not end there. As a WordNet features word relations and their connections, a lot of information

can be utilized from a WordNet regarding any language. This makes it a powerful tool for research works. WordNet has use in various sectors of NLP research works. It can help and contribute in research works concerning word sense disambiguation, which is the process of determining the exact sense of a word used in natural language context. As the WordNet not only stores the synonyms but also gives idea about the context in which it is used and other words used in that same context, it is a valuable resource for word sense disambiguation researches and a complete WordNet can take us miles forward in this sector. Another use of WordNet is in search engine related works. Search engines have to predict words and synonyms based on user input and a WordNet can come very handy in doing this. It can also help in information retrieval systems by retrieving conceptual information of each word in the given query context from the WordNet. In case building an effective automatic text classification, automatic text summarization, text mining system etc can also be benefited from the data stored in a WordNet.

1.2 Motivation

There are scope of lots of research works to do on Bangla language in the huge field of Natural language processing. In current times, much importance has been given to this sector and it is now a fast developing sector. Even so, there is still no Bangla WordNet yet. There has been very little contribution to this field. But presently it has become really necessary to build a Bangla WordNet in order to provide a strong platform for computerized Bangla language. Since there is no Bangla WordNet yet, our target is to contribute in this sector as much as we can. While attempting to construct a Bangla WordNet, we can also shed light on the difficulties faced and the improvement opportunity of the methods applied. We target to present a Bangla WordNet based on the semantic relationship of words.

A big part of constructing a Bangla WordNet is Bangla word embedding. Previous works in this sector have not yielded much promising results. Many methods have been applied and many approaches have failed to increase accuracy. We want to improve the efficiency of Bangla word embedding methods. Previous methods mostly used n-gram approach for word embedding. We want to apply deep learning methods for word embedding to increase efficiency of the process. In attempting to construct the Bangla WordNet we have tried different models to produce word clustering and we can give an overview of their performance. In our work, we are trying an algorithmic

approach which the previous works have not explored.

1.3 Report Structure

The rest of the chapters is structured as follows:

- Chapter 2 reviews some of the related works on Bangla wordnet, Word embedding techniques etc. It also throws light on other approaches on WordNet construction in different languages and the uses of a WordNet.
- Chapter 3 outlines the methodology adopted for our thesis work. It discusses in full detail our implementation, the experiments done, the process followed and the steps implemented to complete our work.
- Chapter 4 deals with the results we have gotten for our implementations, their comparison and the decisions we have reached from them.
- Discussions based on the construction of Bangla WordNet will be found in chapter 5.
- We concluded in Chapter 6.

Chapter 2

Background Study

Although Bangla WordNet is a relatively new topic in the area of Bangla natural language processing, it has grabbed the attention of many researchers in recent times. Some researchers have worked with the goal of developing a complete Bangla WordNet and proposed some methods for constructing the WordNet. We will discuss some of these works in this chapter.

2.1 Literature Review

In this section we give an brief overview on some of the previous research works done on Bangla WordNet, what the current situation is in this sector and the scopes of development. We also discuss the current and previous word clustering techniques applied on various types of data and their performance.

The development of BWN[2] can be considered the first attempt in developing a Bangla WordNet. In 2008 Faruqe and Khan proposed this software framework to build and maintain a Bengali WordNet. They presented the design and implementation of the framework. Their approach can be seen in the figure below.

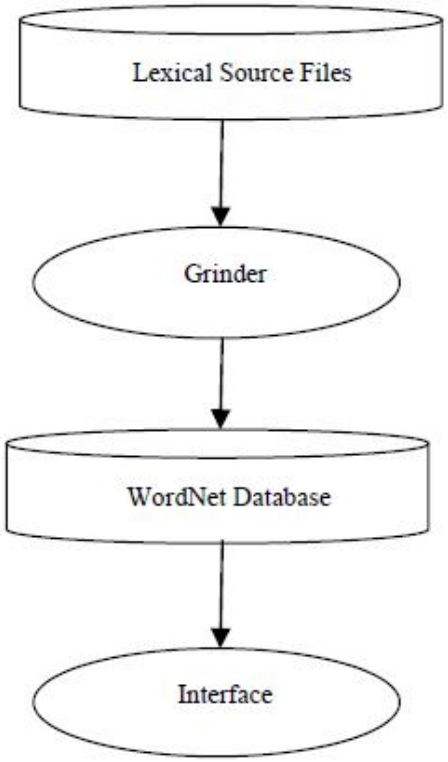

Figure 2.1: Block diagram of WordNet system[2]

With the help of a Grinder they converted lexical source files to inject them to the WordNet Database. They also developed an interface for the WordNet with key features like querying and editing the data through the interface. They also discussed how this framework can help future development of WordNet in other languages.

Another approch towards constructing a WordNet was shown by Rahit, Al-Amin, Hasan, Ahmed[3] in the BanglaNet project where they constructed a baseline for Bangla WordNet and connected it with the Princeton WordNet. They chose a semi-automatic cross lingual sense mapping approach. The Princeton WordNet synset was aligned into a bi-lingual dictionary through the English equivalent and its parts-of-speech (POS) to achieve that. Their proposed method is shown in the figure below.

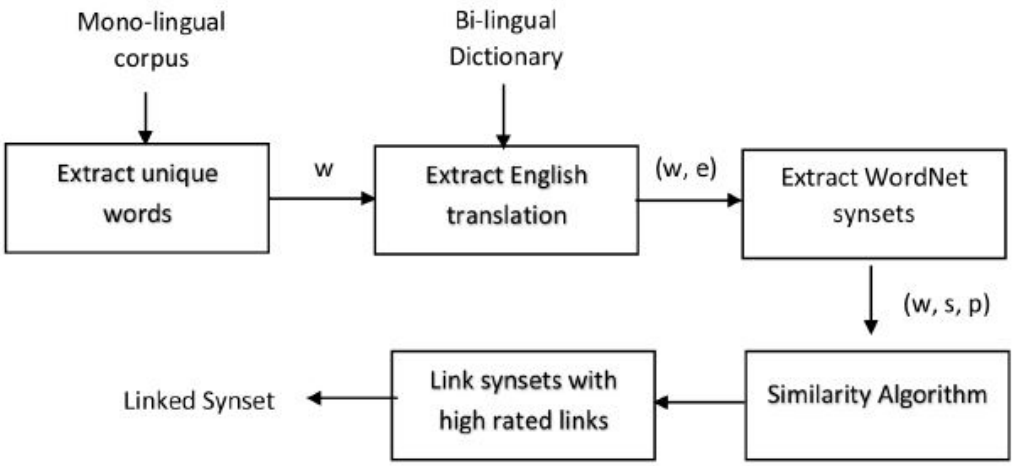

Figure 2.2: Proposed method for BanglaNet[3]

Not only the construction of a Bangla WordNet but also its practical implementations have been explored by researchers.In 2017 Pal, Saha and Naskar[7] tried an knowledge based approach to determine the exact sense of a Bengali ambiguous word with the help of Bengali WordNet. Their method was to check for overlap with the dictionary definition of an ambiguous word with its surrounding words in a sentence and their synonyms and the synonyms of the surrounding words and determine the exact meaning of that word. They reached an accuracy of 75%.

To construct the WordNet, we need to focus on the establishing the semantic relationship of the words. Also word clustering techniques will be necessary to group the related words. Many works have been done in this sector both in Bangla natural language processing and in various other languages. We discuss some of these works below.

Previous word clustering techniques mostly involved using N-gram model to construct the clusters. This can be observed in the works of Ismail and Rahman [8], who proposed a Bangla word clustering method based on N-gram Language Model. In this paper they tried to cluster bangla word using their semantic and contextual similarity. In this approach they tried to cluster the words based on the idea that, the words that have similar meaning and are used in similar context in a sentence, belong to the same cluster.

Their work was slightly upgraded later in 2016 by Urmi, Jammy and Ismail [1]. They pro-

posed a unsupervised learning approach to identify stem or root of a Bangla word from contextual similarity of words. Their object was to build a big corpus of Bangla stems along with their respective inflectional form. They worked with the assumption that if two words are similar in spelling and are used in similar context in many sentences, they have a higher chance of originating from the same root. They implemented 6-gram model for stem detection and achieved an accuracy of 40.18%. They have concluded that with big amount of text data this model will improve further. Some example of clusters they found are given in table I.

Table 2.1: Clusters formed using N-gram approach[1]

Root Word	Word in Cluster
ছোট	ছোটদের, ছোট্ট, ছোটখাট, ছোটো
এখন	এখনকার, এখনো, এখনই, এখনও
আইন	আইনী, আইনি, আইনত, আইনমন্ত্রী,
ঠিক	ঠিকমতো, ঠিকভাবে, ঠিকঠাক
গাছ	গাছগুলো, গাছটি, গাছপালার, গাছপালা

Various other approaches to produce word clustering have also been explored in other works. We can see an example of this in the works of Sinha, Dasgupta and Jana [9], who proposed to construct a Bangla semantic lexicon which is hierarchically organized. To measure semantic similarity between two Bangla words they applied a graph based edge-weighting approach. This lexical organization is represented by a graphical user interface developed by them. They have also added some details to the words like, whether it is a mythological word or not or if it a verb or not etc.

Researchers then focused on producing word clustering in dynamic approach and its performance. We get insights about this from the works of Yuan [10], who showed that word clustering technique that is based on word similarities is better than conventional greedy approach in terms of speed and performance. The basic approach of this work was to check for a certain word in the corpus, its co-occurring words for similarity. That is to say, if two words are similar, their co-occurring word pattern will also be similar. Based on this they computed word clusters and when compared with other clustering methods, this approach was found to be more efficient.

In case of Bangla language, Hadi, Khan, Sayeed [11] proposed a framework for extracting semantic relations in Bangla words. They discussed extraction of Synonyms, Hyponym, Hypernym,

Antonyms, etc as a rule based model. They used semantic analyser on nouns, adjectives and verbs to do this.

The performance of dynamic models in producing word clusters was shown by Ahmed and Amin [12]. They discussed the effect of Bangla word embedding model in document classification. They worked with a dataset prepared from Bangla newspaper documents. They applied word2vec model to generate vector representation of words for word clustering. Using this they prepared clustering of word embeddings that are found in close proximity to each other in feature space. This information was later used as features to solve Bangla document classification problem.

Upgrading the performance of word2vec in finding vector representation of words in huge datasets like a dataset containing one billion words were attempted by Rengasamy, Fu, Lee and Madduri [13]. They applied word2vec in a multi-core system and found that this approach is 3.53 times faster than original multi-threaded word2vec implementation and 1.28 times faster than recent parallel word2vec implementation.

Ma and Zhang [14] discussed the effect of word2vec in reducing the dimensionality of large datasets. They found out that, in dealing with large scale training data, word2vec helps in clustering similar data. This strategy can reduce data dimension and speed up multi-class classifications.

Robert Bamler and Stephan Mandt [15] tried to find the semantic evolution of individual words over time in time-stamped datasets. They applied Word2vec model to produce the embedding vectors. They showed experimentally that both skip-gram filtering and smoothing lead to smoothly changing embedding vectors that help predict contextual similarities at held out time stamps.

Fasttext model is a relatively new model ventured in producing word clusterings. It is a variation of skip gram model architecture of word2vce model which was proposed by Bojanowski, Grave, Joulin and Mikolov [16]. The method they followed was, each word was represented as a bag of character n-grams and vector representation was constructed from them. This allowed them to construct word clusters for words not present in the training data. They concluded that this method gives state of the art word representations for both similarity and analogy task.

Finally, we can say that rich literature is growing on the construction and implementation of Bangla wordnet and word embedding and word clustering techniques in Bangla natural language processing.

2.2 WordNets In Other Languages

Because of the importance of WordNet, many attempts have been made to develop the WordNet for many languages all over the world. As stated earlier, the Princeton WordNet[6] first introduced the concept and proposed a baseline for constructing a WordNet in any language. Following them many WordNets have been constructed so far and many are being developed in different languages. We discuss some of these works in this section.

The Arabic WordNet (AWN)[17] was constructed based on the design and contents of the universally accepted Princeton WordNet(PWN) and was mapped onto PWN 2.0. This was achieved by English to Arabic translation and for each Arabic word, finding all its senses and assigning it to its proper synset.

The Polish WordNet titled PolNet[18] only groups nouns and verbs. Vetulani and Kochanowski applied the ”merge model” approach in developing the Polish WordNet. It currently consists of 11700 synsets for nouns and 1500 synsets for verbs.

The construction of Hindi WordNet[19] first followed the construction method of PWN and categorized words into noun, adjectives, verbs and adverbs. But because of the complexity of Hindi language this did not prove to be very fruitful. After exploring other categorization they finally decided on dividing the words on two basic categories, one for words that can be categorized by the universal process and the other for specific Indian words that needs specific handling.

The insights from the Hindi WordNet plays a big role in the construction of WordNet for other Indo-aryan languages. This was reflected in the works of the Indo WordNet[4] project. It gives an overview of the construction process of WordNet for Indo-aryan languages like Gujrati, Assamese, Malaylam, Tamil, Telegu, Bangla etc. It reached the decision that the WordNet for these languages can be developed by the ”merge and expansion method” on the basis of the Hindi WordNet.

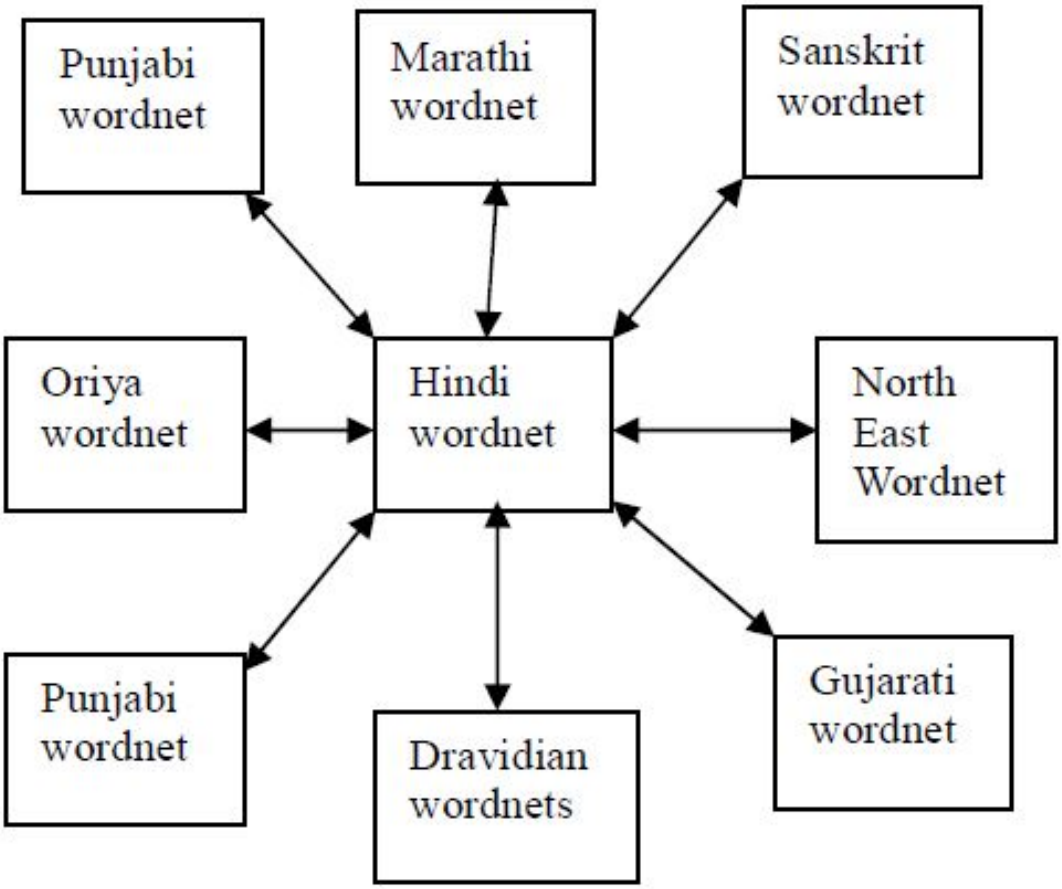

Figure 2.3: Linked Indo WordNet structure[4]

Marathi WordNet developed by Ram and Mahender[20] created a database that consists of more than 10,000 Marathi words with its noun, adjective, verb, adverb and semantic relations like synonymy, Hypernymy, Holonymy and ontology in the form of multiple tables which are in relationship. This WordNet has 36842 unique words grouped in more than 26988 Synsets.

The Sanskrit WordNet[21] is also developed following the merge and expansion method where they considered the Hindi WordNet as the source resource. This WordNet includes features like verbal concepts and gender but does not include semantic relation like ontology.

Discussing the above mentioned works we get an overall idea about the construction of a new WordNet for any languages and the approaches applied so far for doing so. This also gives us an idea about the approach of this task for an Indo-aryan language like Bangla. We can also have an idea about the challenges faced in this task. This will help us in choosing our approach to

accomplish this task.

2.3 Uses of WordNet

WordNet not only serves as a lexical resource for any language but it can also contribute in many other NLP related research works as well as give insights and help take decisions in many language based applications. We shed light on some of the works concerning the use of WordNet in this section.

As WordNet is structured based on word similarity it can be used to measure the Relatedness of Concepts. This was discussed by Pedersen, Patwardhan and Michelizzi[22]. They proposed that WordNet can prove to be really useful in measuring similarities between concepts as it organizes noun and verbs in hierarchies according to their relation. As WordNet stores information about both similarity and dissimilarity of words and the context they are used in, all this information can be utilized to establish similarity measurements between concepts.

We can assume from this work that WordNet can also help in categorization of documents. More specifically we can say that with the help of the informations stored in WordNet text categorization can be done. This was attempted by Elberrichi, Rahmoun, and Bentaalah[23]. They used the synonymy and the hyponymy relation of the WordNet for the text categorization process. Applying this method they found that the use of conceptual ideas from the WordNet improves text categorization process than the traditional Bag-of-words approach.

WordNet can be a very effective tool in Information Retrieval(IR) systems. As a WordNet establishes relations between words, the information from the WordNet database can help to improve the query results in IR systems. Mandala, Takenobu and Hozumi[24] attempted to develop a method of making WordNet more useful in information retrieval applications. Their experiment was done using several standard information retrieval test collections. They showed that using a broadened coverage of WordNet and weighting method, their experiment results in significant improvement of information retrieval performance.

Gharat and Gadge[25] proposed a new method for web information retrieval. They applied a new term weighting technique called concept-based term weighting (CBW) to give a weight for each query term to determine its significance by using WordNet Ontology. This experiment was tested using a web dataset consisting of random web pages. They reached the conclusion that this

method gives better performance than traditional TF-IDF term weighting approach.

WordNet can also prove to be a big contributor in resolving the senses of ambiguous words. A method for Word Sense Disambiguation based on domain information and WordNet hierarchy was proposed by Kolte and Bhirud[26]. They used a unsupervised approach to determine the domain of a random word as target word in the WordNet domain and the sense of that domain was taken as the sense of the target word.

From the discussions above we can perceive that the construction of a WordNet will not only be a lexical database for any language but also can contribute in many other natural language related research works and help achieve better performances.

Chapter 3

Methodology

Our thesis work is to construct a Bangla WordNet. A wordnet basically constructs a relational architecture for words where all the words are connected to each other through relations like synonyms, hyponyms, hypernyms, antonyms etc. The English WordNet is connected based on all these relationships. But as the construction of a wordnet in a new language is a huge and challenging work, we are focusing on constructing the Bangla wordnet based on only the semantic relationship between words. We are trying to connect words with other words that have the same meaning as it and can be used in a sentence in place of one another.

To establish this relationship first we need to group the words having similar meaning. For this we need word embedding method. Word embedding is done on the basis of the concept that, words having similar meaning tend to occur in similar context. So, the first step we have implemented is grouping the words based on their contextual similarity.

The previous approaches on word embedding in case of Bangla language mostly involved N-gram models. But as seen from the discussions in chapter 2, dynamic models can prove to be better than N-gram models to produce the word clusters. So in the first part of this chapter we discuss the previous approaches and our reasons behind choosing the word2vec model and give an overview on how this model works. Then we discuss in detail the steps we have followed in our implementation.

3.1 Data Collection

The data used for constructing a wordnet is usually collected from various sources. We have worked with a large Bangla text corpus in this work to construct the word clusters. We used three separate corpus, and merge them. First corpus is, SUMono [27] which contain available online and offline Bangla text data. We also use a news corpus, which contain news data from Bangla news websites. We also use Bangla wiki data from wikipedia. The detail is given in the table I. Accuracy of any model largely depends on the dataset it is applied on. If a word is used in various kind of sentences, then the trained model can be more accurate as it covers a large area of variety. The more frequent the words are, the more accurate the model will be.

Figure I represents some of the most frequent words from the corpus.

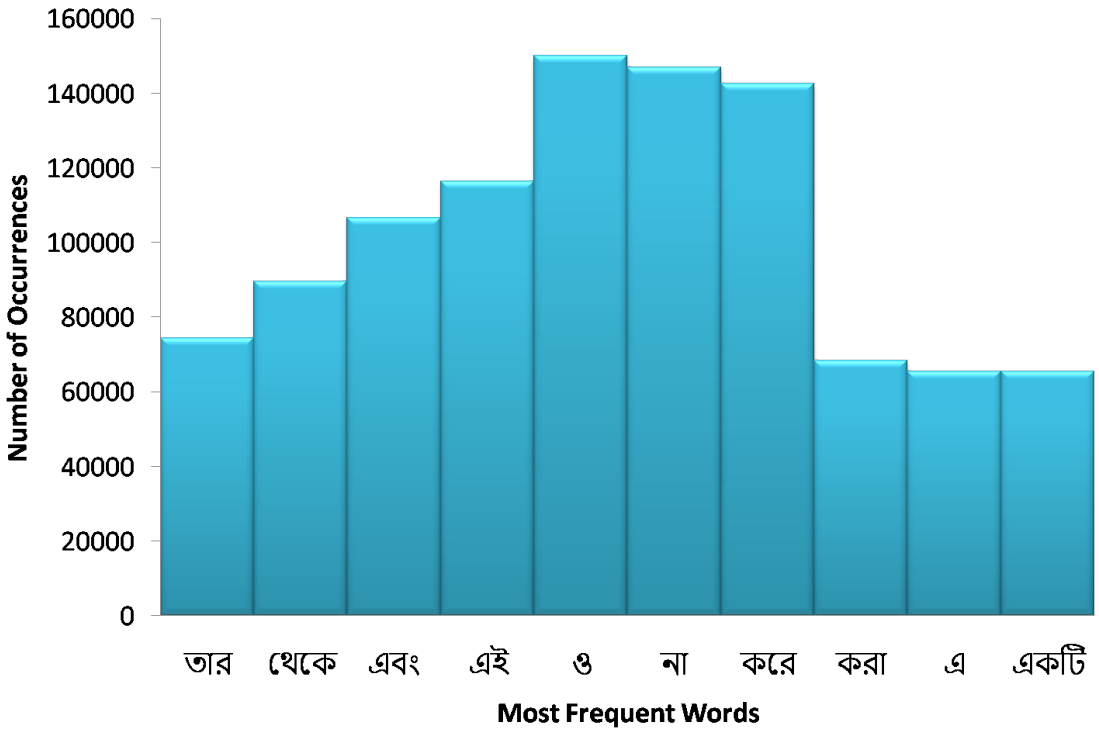

Figure 3.1: Histogram of Most Frequent Words with Number of Occurrences

This corpus consists of around 5,00,000 unique Bangla words. This corpus contains Bangla text data on various topics. This corpus was built taking contents from various sources like Bangla

articles from wikipedia, Bangla news portals articles and from writings of different renowned Bengali writers. As the text is collected from various sources it covers topics of different kinds and sectors as well as the language structure of Bangla used in day to day life. This is an important aspect of the data collected, because to get better and accurate clusters dynamically we need data that covers vast areas in which any specific word can be used. We are constructing the clusters based on the context they are used in so large data covering varied topics is necessary to get better results. Accuracy of this model largely depends on the dataset it is applied on.

Below is a detailed information about the corpus-

Table 3.1: Details of the Corpus

Total sentences	1,593,398
Total words	2,51,89,733
Unique words	5,21,391

3.2 Previous approach on word embedding

Previously there has been many works on word embedding in Bangla and many has tried to find a good method for word embedding. Most of the previous work techniques have used N-gram model. We can take the work of Ismail and Rahman [8] as an example. They used a tri-gram language model where two list of words were generated. One contained the preceding two words of a certain word and the other contained following two words of that same word.

Then for each pair of words in the corpus they then checked total number of matches in the preceding and following word list and based on that a similarity score was counted. If that score crossed a predefined threshold value, then the pair of words are said to be in the same cluster.

But later in the works of Ahmed and Amin [12] it was shown that word clusters produced by dynamic models show better performance than clusters produced using N-gram models.

They compared each word in the corpus with all the other words in the corpus. 5 lakh by 5 lakh word comparing process was really time consuming and not memory efficient. Another problem was, as there is no standard dataset they could not compute an actual accuracy, they only gave a

subjective score of the accuracy, not accuracy with the actual dataset.

3.3 Our approach

Over the years many techniques have been introduced for word clustering. Most of the approaches give high accuracy for clustering English words. As Bangla is a more complicated language, it is hard to gain high accuracy. We attempted three different approaches to determine which approach works better for Bangla word clustering. Nowadays vector representation of words have become the most popular approach for building word clusters. We applied three machine learning approaches which are based on vector representation of Words. The full methodology of our work is discussed in detail below.

3.3.1 Vector representation of words

Text documents are traditionally represented as the term ”Bag of words” in natural language processing(NLP). This means each document is represented as a fixed length vector where the vector size is equal to the vocabulary size of the data. In each index of this vector, the count or number of occurrence of a specific unique word is stored. This process effectively reduces a variable length document to a fixed length vector. This vector makes it easier to work with in various machine learning models like clustering, classification, topic segmentation etc. The process is shown graphically in the figure below.

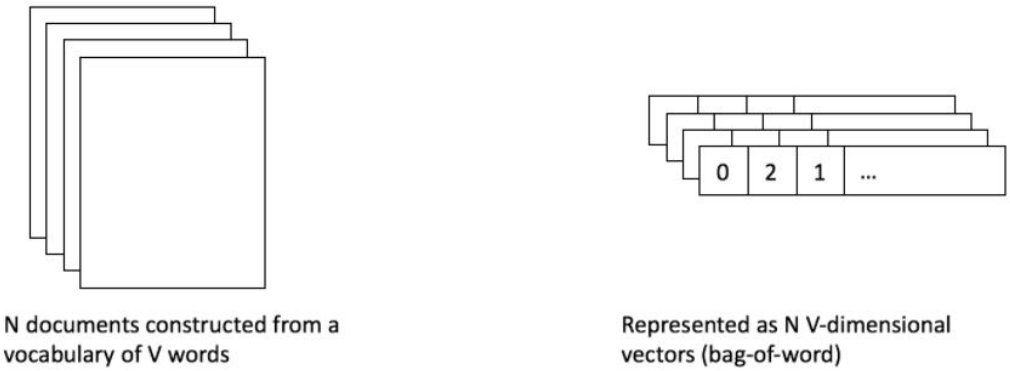

Figure 3.2: Vector representation of a text document

3.3.1.1 Word Embedding

Word embedding is a word representation method that prepares similar representation for words having similar meaning. It is based on the concept that, words that have similar meaning occur in similar context in a text document. Word embedding gives us the semantic relationship of words in a text document. In word embedding methods, words are represented as fixed length vectors. Word embedding organizes them such a way that words that have similarity in context are represented by vectors that are in close proximity to each other. Word embedding process enables us to leverage information from large corpus by constructing their vector representation.

Producing word embeddings from vector representation of words is now a very popular method as it is more efficient than traditional N-gram approach. It reduces memory efficiency and decreases processing time. So, we can see that utilizing this method we can resolve the difficulties discussed in the previous section. In our work we have applied different variations of word embedding models using vector representation of words. But some pre-processing was needed to be done on the data before feeding it to the dynamic word embedding models. We discuss this in the next section.

3.3.2 Pre-processing steps

We couldn't feed a word just as a text string to the word embedding models, we needed a way to represent the words to the network. For this, we followed some pre-processing steps to prepare

the data to be fed to the word embedding models for producing the word clusters. These steps are discussed below.

- **Corpus:** The corpus is stored as a text file. In the text file the data constitutes of Bangla sentences. They can be treated as strings. But We can not feed the strings directly to this model, so some pre-processing was done on the dataset.
- **Tokenizing:** We had to pre-process the corpus in order to use them as our proper input. Firstly, we couldn't feed a word just as a text string to a model. For this, we tokenized each word. Example:

আমার সাথে বাংলায় কথা বল => 'আমার', 'সাথে', 'বাংলায়', 'কথা', 'বল'

- **Training:** We used the tokenized dataset for training. These tokenized words were fed to the model as input. The model then builds the vocabulary from the input. This vocabulary is then used to generate the word vectors by the different models and construct the word clusters from them.

After following these basic pre-processing steps the data is fed to a dynamic word embedding model to produce the clusters. Now there are various dynamic models to produce the word embeddings and they each vary in result, performance and accuracy depending on the dataset they are applied on. Also there isn't much insight on the performance of different dynamic word embedding models in case of Bangla language. So instead of choosing one specific dynamic models, we applied different variations of dynamic word embedding models on our dataset to find the model most appropriate for building the WordNet. We discuss these models in the next section.

3.3.3 The word2vec model

Word2vec is a dynamic word embedding model that utilizes vector representation of word and produces word embeddings from them. In the first step of our implementation, we have applied three variations of the word2vec model to produce the word embeddings and compare the results. We discuss the whole process in detail in this section.

In this model, neural network is trained with a single hidden layer for performing a task. But the network is not actually used to perform that task; instead we learn the weights of the hidden layer which are actually the "word vectors". In order to do so, the neural network must be trained. For example, suppose we are going to train the neural network to do the following task-

A specific word, called the pivot word is given in the middle of a sentence; if we look at the words nearby and randomly pick one, the network will tell us for every word in our vocabulary the probable nearby word that has been chosen. Here nearby actually means a "window size", which is a parameter to the algorithm. If a window size is 3, it means 3 words behind and 3 words ahead, that is 6 in total.

The probabilities of our output are related to how likely it is found a word nearby the given input. For example, if we gave the trained network the word 'খাই' as input, the higher probable words for output are 'ভাত', 'মাছ' etc than for unrelated words such as 'খাতা' , 'কলম' etc.

We have trained the neural network to do this task by giving it pairs of words found in our training data. The example below shows some of the training samples we would take from the sentence 'তোমার প্রতি আমার কোন অভি্যোগ নেই'. Here a small window size of 2 is used just for the example. The word highlighted in red is the input word. And the blue highlighted words are in the window.

Source Text	Training samples
"তোমার প্রতি আমার কোন অভি্যোগ নেই"	(তোমার,প্রতি) (তোমার,আমার)
"তোমার প্রতি আমার কোন অভি্যোগ নেই".	(প্রতি,তোমার) (প্রতি,আমার) (প্রতি,কোন)
"তোমার প্রতি আমার কোন অভি্যোগ নেই".	(আমার,তোমার) (আমার,প্রতি) (আমার,কোন) (আমার,অভি্যোগ)
"তোমার প্রতি আমার কোন অভি্যোগ নেই".	(কোন,প্রতি) (কোন,আমার) (কোন,অভি্যোগ) (কোন,নেই)

There are two variations of word2vec model, Continuous Bag of Words or CBOW architecture

and skip-gram architecture. In CBOW architecture the pivot word is predicted based on a set of context words. In skip-gram architecture the process is reversed. It predicts the context words using the pivot word.

The wor2vec model can be implemented by both the Tensorflow library and the Gensim library package. We applied both these methods in our implementation to produce word embedding for different words and compare the results.

- **Experiment I: Word2vec in Tensorflow**

  We used the official version of sentence embedding implementation of tensorflow. The code generates word clusters based on the features of Word2vec using the Skip-Gram model and the Negative Sampling accelerated classification algorithm. This model needs some parameter specification like the window size, vector size and number of iterations. We tried different window sizes, vector sizes and iterations to find the optimal results for the model. The results vary with the change in these parameters and fine tuning of parameters were needed to get optimal results. This is shown in detail in result analysis chapter.

- **Experiment II: Word2vec from Gensim package(Skip-gram model)**

  The python library Gensim provides Word2Vec class for producing word embeddings. It is a bulit-in class that follows the basic work process of word2vec and produces the word embeddings given the window size and vector size. First we implemented the skip-gram model architecture to produce the word embeddings. Parameter tuning was needed in this case too for getting the optimal result. So different window size and vector size was tried out to find the optimal one.

- **Experiment II: Word2vec from Gensim package(CBOW model)**

  As the next step we implemented the CBOW or Continuous Bag of Words architecture by the built-in Gensim Word2vec package. As discussed before CBOW works in the opposite way of skip-gram, so there were significant changes in the results from the ones produced by skip-gram architecture. After trying out different window sizes and vector sizes we found optimal results which are shown in result analysis chapter.

3.3.4 FastText Model

Facebook's AI Research lab created the FastText library for producing word embeddings and text classification. This model allows to create an unsupervised learning or supervised learning algorithm for obtaining vector representations for words. Then it constructs the word embeddings from them which can be later used for text classification process. They also developed pre-trained word vectors for 157 languages [28], which was trained on available Wikipedia data of those languages using fastText. FastText uses n-grams of a word and create vectors for the sum of all the n-grams of the word. A big improvement of this model over other dynamic word embedding models is that it can produce word embeddings for out of corpus words. So, even if a word is given to this model that was not present in the training data, it can still produce word embeddings for that unknown word.

Although they provide the pre-trained word vectors for Bangla also, we did not use those pre-trained vectors in our implementation. We implemented the fasttext library on our dataset to build the word vectors first and then constructed the word embeddings from them.

Like word2vec, fasttext also has both Skip gram and CBOW model architecture. It also works the same way as in word2vec. Skip-gram predicts the context words from the pivot word while CBOW uses the context words to predict the pivot word. In fasttext library we implemented both these variations to compare the word embeddings produced by them. That is discussed below.

- **Experiment III: FastText Skip-gram model**

  In this step we first implemented the FastText library Skip-gram architecture for preparing the vector representation of the words in our corpus and then constructed the word embeddings from them. The window size, vector size and iteration parameters were specified in this case. With little parameter tuning this model produced satisfactory embeddings which are shown in the result analysis chapter.

- **Experiment III: FastText CBOW model**

  As the next step we implemented the CBOW or Continuous Bag of Words architecture by the FastText library. We applied it on our dataset and built the word embeddings from them. The parameter tuning for this one was the same as the FastText Skip-gram architecture but the results varied from these two models as they are the reverse process of each other.

By implementing these five variations of dynamic word embedding models on our dataset, we constructed different word clusters for different models and compared the outputs to reach a decision about which model is the more suitable one for the construction process of a Bangla WordNet. The details and outcome of the comparison process is shown in the results analysis chapter.

3.3.5 Dictionary Parsing

The WordNet not only connects the words but also contains the meaning, definition, uses, parts of speech and other grammatical information of all the connected words. The clustering process helps us to connect the words and establish the relationship among the words. As the next step, we need to add the necessary information to the specific words. For this we need a Bangla dictionary to get the detailed informations of the connected words from it and assign to each word its own informations. For this purpose we thought of a Bangla to Bangla dictionary. We used the Bangla Ekademi Ovidhan which contains details information of Bangla words like its meaning, definition, parts of speech etc. We collected this dictionary and prepared it in text format for our use. Below is a small example of the dictionary we used.

Example:

- ইচ্ছা: বি ১ অভিলাষ; স্পৃহা; বাসনা; রুচি; প্রবৃত্তি (খাওয়ার ইচ্ছা নাই) ।
- আমি: সর্ব বক্তা নিজে । উত্তম পুরুষ; বাক্যে বক্তা ।

Here, for the first example, 'ইচ্ছা' is the target word. Next, its meanings in Bangla 'অভিলাষ' is shown. 'বি' indicates 'বিশেষ্য' or noun in English, giving the parts of speech of the word and the rest of the string gives synonyms of the target word with the example of its uses.

As for the second example, the Bangla meaning of target word 'আমি' is given by 'বক্তা নিজে' and সর্ব indicates 'সর্বনাম' meaning pronoun in English, indicating the parts of speech of the target word. 'উত্তম পুরুষ' means its a first person word and the rest of the string is the uses of the word.

We processed the collected dictionary according to our need for parsing. Parsing is the process of analyzing a string or text into logical syntactic components. In our work, the dictionary is parsed into key-value pairs, where the key denotes a specific word or target word and the value denotes all the necessary informations associated with that particular word. so, if we query using the key,

it would return the value which represents detailed information about the word we used as a key.

3.3.6 Hierarchy Building

For constructing the WordNet next we need to focus on establishing the relationship among the words and connect them according to that relation building up the WordNet structure. In WordNet, the main relation among words is the synonymy. Synonymous words having the same meaning, denoting the same concept, words that can be used in place of one another are connected to each other in WordNet. These connected words also has their definitions attached with them. These connected words make small groups and these groups in turn connect to each other via semantic relations and following this process for all words in a language the WordNet network gradually builds up. Also another aspect of WordNet is that the words contain its parts of speech information too. Noun words will connect to its synonyms which are noun too, they in turn will be connected to their synonyms and this way words belonging to the same parts of speech will ultimately belong to one big group in the WordNet structure.

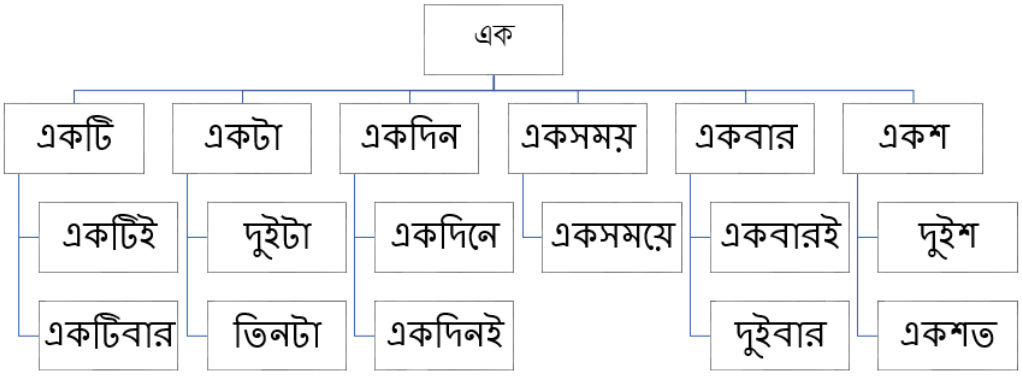

Figure 3.3: Example of Hierarchy of Word relations

Figure 3.3 shows an example structure of the WordNet hierarchy. From the root word 'এক' the network spreads connecting the similar words to the root word.

for our implementation, we are building from each word level. First we have to find the connected words of each word, establish the relation, then we have to find the words connected to those connected words present in the first level. The clusters constructed by the word embedding

models come in handy in this step. These clusters give us an idea about which words are related to whom. We can build the WordNet network from this information. We have first compared among the clusters produced by the different models to choose one appropriate model for building the hierarchical structure of the WordNet. Then using the clusters of that model we have build up the hierarchical clusters for each word, where root word is connected to its most similar ten words in the first layer. Then in the next layer each of these ten words are connected to its most similar five words. Following this process the network is built up. The details and outcomes of the process is shown in the result analysis chapter.

3.3.7 Adding Details to Hierarchy Structure

In the previous section, we have discussed building the backbone of the WordNet structure by connecting the words. As the next step, We need to add the meaning, definition and related grammatical informations of a specific word to the hierarchical WordNet structure to complete it. As discussed in the dictionary parsing section, these detailed informations are formatted as key-value pairs for processing. We will now use these key-value pairs to attach the details of the word to the WordNet structure.

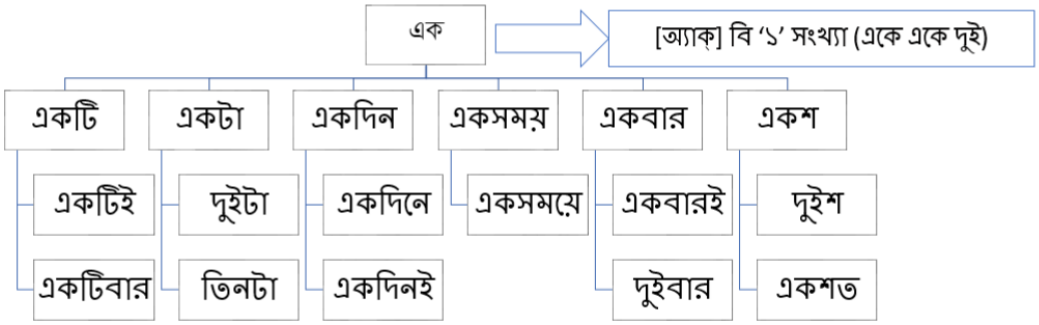

Figure 3.4: Mapping with dictionary

For doing this, we used the root word or the target word as key in the dictionary and got the value as output. That value is associated with the details of the word stored in the dictionary. That information is then mapped to the target word. Repeating this process for all the words the

hierarchy structure we get the complete WordNet Structure. A graphical representation of the process is shown in figure 3.4 where to the previously shown WordNet backbone structure, now the details of the root word is mapped from the dictionary. This process has to be repeated for all the words present in the corpus to complete the WordNet. The outcome of this step is shown in the result analysis chapter.

Chapter 4

Result Analysis

As discussed in the previous chapter, we have implemented five different variations of dynamic word embedding models and compared among the constructed word embeddings. from this comparison we attempt to find a suitable dynamic word embedding model for constructing the Bangla WordNet. In this chapter we discuss in detail the results obtained from our implementation.

First of all, parameter tuning was needed for all the word embedding models to get the optimal result of that model. We tried to get the most satisfactory results from each approach. We applied various combination of window and vector sizes and checked the results in order to tune the parameters to get the most optimum and satisfactory results. Table 4.1 shows the optimal parameter tuning for each approach.

Table 4.1: Parameter Tuning for Optimum Results

Model	Window size	Vector size	Iteration
Exp I: Word2vec in Tensorflow	4	1000	10
Exp II: Word2vec from Gensim package (Skip-gram model)	5	400	5
Exp II: Word2vec from Gensim package (CBOW model)	5	400	5
Exp III: FastText Skip-gram model	5	100	5
Exp III: FastText CBOW model	5	100	5

Coming to the constructed word embeddings, it can be seen from the results that there are similarities among the clusters from five different approaches as well as some significant differences. For some words, we got a similar set of words for a set of models. But the variety was also notable. The next section shows the constructed word embeddings of these five models.

4.1 Experiment I: Word2vec in Tensorflow

This section contains the word embedding constructed by implementing Word2vec in Tensorflow model. These are the optimal results of this model acquired by keeping the vector size at 1000 while the window size was 4 and constructed after 10 iterations. As can be seen from the examples below, the cluster contains inflection or different form of pivot word as well as the context words that tend to occur with the pivot word. Here most similar 10 words to the pivot word is shown.

Table 4.2: Results from Word2vec in Tensorflow

Random Word	Words on cluster
আমরা	আমাদের, আমি, চাই, যখন, তাই, তারা, কি, সেই, কিন্তু, সবাই
তাঁর	তার, সেই, সাথে, তাঁদের, একজন, একই, ওই, তিনি, পরে, বলে
জন্য	প্রয়োজন, সুযোগ, জন্যে, পাশাপাশি, তাই, দরকার, কারণ, কিছু, কিন্তু, তাদের
কোন	কোনো, এমন, অন্য, কারণ, তবে, সেটা, বা, নেই, তাই, এখনো
পারে	হবে, পারবে, পারি, হলে, পারেন, হতো, চাই, চায়, পারেনি, থাকে
হতে	যেতে, থাকতে, করতে, হলে, না, রাখতে, তাই, তাহলে, তবে, দিতে
বড়	সবচেয়ে, অবস্থা, খুবই, খুব, আমাদের, অনেক, কিছু, মতো, মানুষের, আছে
টাকা	হাজার, লাখ, কোটি, টাকার, খরচ, পাঁচ, মাত্র, বিক্রি, তিন, প্রায়
নতুন	মাধ্যমে, তৈরি, কাজ, জন্য, বিভিন্ন, নানা, একটি, এই, সব, একই
আমাদের	তাই, কিন্তু, যে, সেই, শুধু, এই, অনেক, সব, এখন, আমরা
একটি	একটি, একটিদুটি, যেএকটি, একট, একটিসহ, একএকটি, একটিই, একটিভ, নেইএকটি, একটিও
আমি	আমিই, আমিও, আমও, আমিআমি,আমি, আআমি, আমিষ, কীআমি, আমিতো, ওকেআমি
যায়	যায়, যায়আসে, যায়সে, যায়ই, যাঃ, যাবে, যাবেএই, যাচ্ছি, য়যায়, যাব
করেন	করেননি, করেনঃ, করেন, করেনি।, করেনও, করেতেন', করেছেন, করেণ, করেছিলেন, করেনিবরং
বছর	বছ, চবছর, বছর।, নছর, ছরছর, দশবছর, বছরকয়েক, গতবছর,
এখন	এখনৌ, এখনো, এখনকী, এখনতো, এখনই।, কীএখন, এখনএই, এখনই, এখনো, এখনি
আবার	নাআবার, তারপরআবার, র্তেআবার, আববার, আরেকবার, তারআবার, ডাকবার, বেরুবার, চকবার, ফেরবার
মতো	মতোও, মতো।, রমতো, মতোই।, জমতো, মতোনই, মতোই, এইমতো, মাপমতো, মতোন
কাজ	কাজ।, কাজও, কাজো, কাজই, কাজটাজ, ওকাজ, কাজটিই, কাজাখ, সৎকাজ, কাজটি
দেখা	অদেখা, দেখাইত, দেখাক, দেখায়ই, দেখাসহ, দেখায়া, দেখাএইসবই, দেখাত, দেখাবো, দেখাবো

4.2 Experiment II: Word2vec from Gensim package (Skip-gram model)

In this section we show the word embedding constructed by implementing Word2vec from Gensim package (Skip-gram model). These are the optimal results of this model acquired by keeping the vector size at 400 while the window size was 5 and constructed after 5 iterations. As can be seen from the examples below, the cluster contains similar types of words as the pivot word and context words occurring with the pivot word. Here most similar 10 words to the pivot word is shown.

Table 4.3: Results from Word2vec from Gensim package (Skip-gram model)

Random Word	**Words on cluster**
আমরা	আমি, তোমরা, সেটা, তাহলে, পারি, এখনো, হয়তো, এখানে, করেছি, তোমাকে
তাঁর	তাঁদের, স্বাধীনতার, যিনি, নেন, নিজ, মেডিকেল, দেন, করছিলেন, পরিবারের, যুদ্ধ
জন্য	জন্যে, সুযোগ, চেষ্টা, কাজে, উদ্দেশ্যে, মাধ্যমে, ব্যাপারে, ব্যবস্থা, করলে, পর্যায়ে
কোন	কোনো, থাকার, ছাড়া, তাতে, বসানোর, আপত্তি, প্রয়োজন, তেমন, উপায়, এমন
পারে	পারবে, চায়, পারেনি, পারেন, পারত, হতো, পারব, বাধ্য, চাই, পারবেন
হতে	পেতে, যেতে, থাকতে, রাখতে, আনতে, দোকানেও, নিতে, ঘটতে, বেশিও, লাগতে
বড়	ছোট, সবচেয়ে, শিরোনামায়, মেয়ে, গায়কেরা, সুন্দর, জোরটা, জিনিস, মধ্যবিত্তদের, ছেলে
টাকা	কোটি, হাজার, পাঁচ, টাকার, লক্ষ, বরাদ্দ, লাখ, বছর, প্রায়, গত
নতুন	নির্যাতন, প্রক্রিয়া, জাতীয়, পূর্বে, মামলা, এলাকায়, সেনাবাহিনী, অর্থনৈতিক, বিচারের, জোট
আমাদের	সবার, এখন, তোমাদের, সবাইকে, তোমার, ইয়ার্কদের, মানুষকে, এটাই, স্বাভাবিক, ওই
একটি	এটি, দুটি, অনুযায়ী, বর্তমান, সকল, আকারে, বিভিন্ন, ইতিহাসে, নতুন, নিজস্ব
আমি	বললাম, কেনে,হ্যাঁ, তোমাকে, তুমি,আমিও, তো,তোমাদের, রাজি,দেব
যায়	যাবে, যাচ্ছে, গেছে, যাক, যায়, গেলে, আসে, রূপ, ওঠে, যেত
করেন	করেছেন, করেছিলেন, দেন, করেননি, করেছিল, করবেন, হিসেবে, নেন, বিরুদ্ধে, অসহযোগে
বছর	গত, পাঁচ, চার, হাজার, মাস, বিশ, দশ, শত, সপ্তাহ, বছরের
এখন	নাকি, এখনো, নিশ্চয়ই, পেয়েছি, কথাই, এটাই, যখনই, কোনটা, স্বাভাবিক, সেটাই
আবার	পড়তে, একেবারে, সবাই, অফিসে, সবকিছু, বাড়িতে, পরিষ্কার, গুলি, করেই, মানুষটি
মতো	বেশ, লেন্সে, সুন্দর, অনেকটা, ভর্তি, পুরো, স্বচ্ছ, ফোকাস, লেন্সের,আলোতে
কাজ	কাজে, শেষে, সহায়তা, ভয়াবহ, সংগ্রহ, দিয়েই, প্রভাব, ইচ্ছা, করতো,অসাধারণ
দেখা	কমে, রয়ে, ফুসফুসে, গবেষণায়, জানা, বেড়ে, পাওয়া, রোগী, রূপ, জীবদ্দশা

4.3 Experiment II: Word2vec from Gensim package (CBOW model)

In this section we show the word embedding constructed by implementing Word2vec from Gensim package (CBOW model). These are the optimal results of this model acquired by keeping the vector size at 400 while the window size was 5 and constructed after 5 iterations. we can see from the examples below, the cluster does not contain similar words or context words of the pivot word rather it has given noisy output. Here most similar 10 words to the pivot word is shown.

Table 4.4: Results from Word2vec from Gensim package (CBOW model)

Random Word	**Words on cluster**
আমরা	ইয়ার্কদের, যতই, পৌঁছাতে, হোক, লিডাররা, শেয়ারিঙের, অবশ্যই, কথাও, চিন্তা, দুবারই
তাঁর	তাঁরা, তৈরিতে, কো, সারা, বর্জনের, সৃষ্টিতে, ভাইরাস, অপরিশোধিত, কলফনি, লিপির
জন্য	যুক্তি, চিহ্নিত, জোরদার, পরিবেশ, আশ্বাস, প্রকাশ, সমালোচনা, অন্তর্ভূক্ত, প্রত্যাখ্যান, প্রতিরোধ
কোন	কোনো, প্রত্যয়, কিছুরই, কারণ, বইতে, তেমনভাবে, আধ্যাত্মিকতা, সাবজেক্টে, ঘটেনি, ইন্ডাস্ট্রির
পারে	ঈর্ষায়, দগ্ধ, অভিবাসী, পাঠ্যবই, পারবেন, পারত, পারবে, পেরেছিল, অস্বীকৃতি, নামিজউদ্দিন
হতে	হতেই, থাকতে, দিতে, যাইতে, ফিল্মগুলো, পেতে, নিতে, সাজিয়া, ঘটাতে, জানতে
বড়	ফুটো, শ্লেষ্মা, দরজাটা, আলো, পাথরের, লেজের, চওড়া, রাস্তা, অন্ধকার, খাটো
টাকা	বছরের, 'ছয়, চলাচলের, দশ, সপ্তাহের, পূর্ববঙ্গে, লাখ, জেলার, উপলক্ষে, গান্ধীর
নতুন	গান্ধীকে,পাকিস্তান, বণ্টন, ক্রয়, মন্তব্য, শিক্ষা, পরিকল্পনার, অধিগ্রহণ, মাউন্টব্যাটেন, প্রস্তাবিত
আমাদের	বোঝার, সত্যি, রাখি, ডেথসিটিতে, তরুণী, ভাবনা, ঢুকতে, কোনটা, শুধু, কখন
একটি	বন্দরের, শাসন, আইন, হিসেবে, স্বাধীনতা, ভারতের, সংখ্যাগরিষ্ঠ, রোধ, সরকার, স্বরাজ্য
আমি	তোমাদের, জখম, আমিও, মেরেছ, রাগ, বেবী, ব্যাটা, থাকি, নাচের, আছেই
যায়	যেত, যাবে, দিত, ক্যান্টনমেন্ট, উপহারগুলো, যারাই, আধাজন্তু, জন্মাবে, থাকে, ক্রিয়াশীল
করেন	হওয়ার, বাস্তবায়ন, পদক্ষেপ, বাঘা, মনোভাবের, গভর্নর, ব্যবস্থার, বিনিয়োগে, পার্টিশন, প্রবর্তনের
বছর	জন্মের, আড়াই, পাঁচ, সাত, টাকা, দৈনিক, প্রায়, মিলিয়ন, বছরে, গোয়েন্দাকে
এখন	সত্যি, বুঝতেই, মজা, ভালো, শুধু, তারচেয়ে, শুনতে, কথাটা, বল, হয়তো
আবার	নিয়ে, আসতে, ফুল, যেন, গাড়িটা, চিন্তিত, নামতে, খানিকটা, দুর্বল, সবিতা
মতো	দিলে, পুরো, ফিলিপস, প্রচন্ড, দেখিয়ে, জলে, খেলা, প্রথমবার, সামান্য, চেহারার
কাজ	ব্যবহার, আলোচনা, রক্ষা, সেসব, সহজ, পরিহার, নিশ্চিত, প্রয়োগ, বাজেট, ব্যাখ্যা
দেখা	পাওয়া, জানা, পাহারায়, বেঁকে, কাঁচুলি, ব্লেজারেই, উর্দুকে, শহরগুলোর, ঝরেতুমি, গেয়ে

4.4 Experiment III: FastText Skip-gram model

In this section we show the word embedding constructed by implementing FastText Skip-gram model. The table below shows the optimal results of this model acquired by keeping the vector size at 100 while the window size was 5 and constructed after 5 iterations. Here optimal result was found with smaller vector size then other models. As can be seen from the examples below, the cluster contains mostly inflections or different forms of pivot word. Here most similar 10 words to the pivot word is shown.

Table 4.5: Results from FastText Skip-gram model

Random Word	**Words on cluster**
আমরা	আমরা।, কীআমরা, নয়আমরা, আমরাই, আমরাতো, হয়আমরা, কিআমরা, হোকআমরা, আমর, লেআমরা
তাঁর	তাঁর।, তাঁরই, তাঁরও, তাঁহা, তাঁরা, তাঁতীও, তাঁরাই, তাঁ, তাঁকেসহ, ওঁর
জন্য	জন্য।, জন্যেঃ, জন্যও, জন্যে।, সৌজন্য, জন্যে, এজন্য, জন্যই, জন্যি, এরজন্য
কোন	কোনস, কোন্, কোনো।, কোনোও, কোনো, কোনডা, কোন্, কোনোই, কোনও, কোনই
পারে	পারো, পারে।, পারেএ, পারেতখন, পারেঃ, পারেআর, পারেএমন, পারেনই, পারেন।, ঐপারে
হতে	ঝতে, লখনউতে, ধতে, ইইউতে, নড়তে, নতে, পেতে, চড়তে, অইতে, ওতে
বড়	বড়বড়, বড়র, হড়বড়, বড়ও, বড়ইর, বড়ছোট, ছোটবড়, ছোটছোট, ছোট, নড়বড়
টাকা	টাকা।, দশটাকা, টাকায়ও, টাকাসহ, হাজারগুণ, দুইটাকা, হাজারও, দুটাকা, হাজারদিঘী, টাকাকী
নতুন	নতুননতুন, নতুনতর, নতুন।, নতুনই, নত, নতূন, জৈতুন, নিত্যনতুন, নতুনরা, চালু
আমাদের	যেআমাদের, নাআমাদের, হবেআমাদের, দাদাআমাদের, ছিলআমাদের, সৎমাদের, তমাদের, মাদের, রোমাদের, অমাদের
একটি	একটি, একটিদুটি, যেএকটি, একট, একটিসহ, একএকটি, একটিই, একটিভ, নেইএকটি, একটিও
আমি	আমিই, আমিও, আমও, আমিআমি,আমি, আআমি, আমিষ, কীআমি, আমিতো, ওকেআমি
যায়	যায়, যায়আসে, যায়সে, যায়ই, যাঃ, যাবে, যাবেএই, যাচিছ, য়যায়, যাব
করেন	করেননি, করেনঃ, করেন, করেনি।, করেনও, করেতেন', করেছেন, করেণ, করেছিলেন, করেনিবরং
বছর	বছ, চবছর, বছর।, নছর, ছরছর, দশবছর, বছরকয়েক, গতবছর,
এখন	এখনৌ, এখনো, এখনকী, এখনতো, এখনই।, কীএখন, এখনএই, এখনই, এখনো, এখনি
আবার	নাআবার, তারপরআবার, র্তেআবার, আববার, আরেকবার, তারআবার, ডাকবার, বেরুবার, চকবার, ফেরবার
মতো	মতোও, মতো।, রমতো, মতোই।, জমতো, মতোনই, মতোই, এইমতো, মাপমতো, মতোন
কাজ	কাজ।, কাজও, কাজো, কাজই, কাজটাজ, ওকাজ, কাজটিই, কাজাখ, সৎকাজ, কাজটি
দেখা	অদেখা, দেখাইত, দেখাক, দেখায়ই, দেখাসহ, দেখায়া, দেখাএইসবই, দেখাত, দেখাবো, দেখাবো

4.5 Experiment III: FastText CBOW model

In this section we show the word embedding constructed by implementing FastText CBOW model. The table below shows the optimal results of this model acquired by keeping the vector size at 100 while the window size was 5 and constructed after 5 iterations. Here, like the fasttext skip-gram model, optimal result was found with smaller vector size then other models. As can be seen from the examples below, although the cluster contains mostly inflections or different forms of pivot word it also contains some noise. Here most similar 10 words to the pivot word is shown.

Table 4.6: Results from FastText CBOW model

Random Word	Words on cluster
আমরা	আমরাযদিও, আমরাই, আপনাকেও, আমরাও, বলেছিআমরা, আপনা, কীআমরা, কীটও, আপনাকেই, কীটস
তাঁর	পুনঃআলোচনার, সুলোচনার, আলাপআলোচনার, কেস্তার, তাঁরই, কাইয়ুমআলোচনার, সৃষ্টিশীলতার, ধিক্কার, মনিকার, বিঘার
জন্য	জন্যও, জন্য।, জন্স, সৌজন্য, এজন্য, জন্যেঃ, সৌজন্যঃ, জন্যই, জন্যে।, তজ্জন্য
কোন	কোনোও, কোনো।, কোনোই, কোনো, লুকোনো, থোন, কোনস, কোন্, কোন্, কোনোটিই
পারে	পারেআর, পারেতখন, পারে।, পারেএ, পারো, পারডন, পারেঃ, পারেও, পারব্বর, পারদ
হতে	কইরতে, ঝরতে, ভরতে, কসরতে, খোরতে, শরতে, মরতে, ধরতে, ঠকতে, কুদরতে
বড়	বড়র, বড়ও, হড়বড়, বড়ইর, বড়ই, বড়সড়, গড়বড়, বড়সড়ো, নড়বড়, বড়বাড়ী
টাকা	দশটাকা, ওসাকা, শলাকা, পোঁদপাকা, গাঢাকা, টাকা।, জলধাকা, ইয়াকা, হাজারী, এলাকা
নতুন	নতুননতুন, নতুন।, 'নতুনই, ফোরামটি, অনুষ্ঠিত, পুনর্গঠিত, তিনচারটি, উৎকণ্ঠিত, পরিচালকমণ্ডলীর, ভূখণ্ডটি
আমাদের	তোমাদের, নাআমাদের, যেআমাদের, এদেরও, তাঁদেরও, খেদেরও, ইহাঁদেরও, এঁদেরই, আপনাদের, তাঁদেরই
একটি	একএকটি, ইবুকটি, ছকটি, বৈঠকটি, পোষাকটি, যুগটি, ফলকটি, সড়কটি, সূচকটি, লিংকটি
আমি	আমিআমি, আমি, আমিষ, আমিই, ওকেআমি, আমিরউল, আমিও, কীআমি, আআমি, তোমাকেআমি
যায়	যায়যায়, যায়নি, যায়মেজর, যায়এক, যায়ও, যায়, যায়গা, যায়সে, যায়এই, যায়এসব
করেন	করেনি, করেনিবরং, করেননি, করেনও, করেনঃ, করেন, করেছেন, করেননি, করেনি, করেতেন
বছর	চবছর, বছ, বছর, বছরর, বছর, নছর, ছরছর, দশবছর, বছরেরই, বছরই
এখন	এখনো, এখনই, এখনৌ, এখনকী, কীএখন, এখনএই, এখনই, কিতো, কিটস, কিষাণ
আবার	তারপরআবার, দেখবার, দেখিবার, নাআবার, তারপরসব, ধোবারজোড়, দশবারো, খুঁজবার, দেখাবার, তারপরো
মতো	জুতমতো, খুদাই, আশাও, হোতো, রুবাই, আভাই, দুকথা, গচ্ছামি, গুঁতো, খুশিমতো
কাজ	ব্যবহারটা, অকাজ, সেচেষ্টা, ব্যবহারকে, ব্যবসাকে, ধর্ষণচেষ্টা, চেষ্টাটুকু, সহজই, কষ্টও, নিশ্চেষ্ট
দেখা	অদেখা, দেখায়ই, দেখাসহ, দেখাইত, দেখাএইসবই, দেখাত, দেখাক, দেখাবো, দেখায়া, দেখাও

4.6 Training Time

An important aspect of comparing these models are the measure of training time needed for each model. In this section we show the time that was required to train each model on our dataset.

The size of our corpus is 59856174 bytes. We used a computer with 4GB RAM, core i3-3110M CPU. Training Time for each experiment is shown in table 4.7. The graphical representation is also shown in figure 4.1.

As can be seen from the table the training time varied from model to model while almost same time was needed for building the both variations of the same model.

Table 4.7: Training Time of the Experiments

Experiment	Training Time
Exp I: Word2Vec in Tensorflow	18 minutes
Exp II: Gensim Word2Vec- Skip gram Model	30 minutes
Exp II: Gensim Word2Vec- CBOW Model	32 minutes
Exp III: FastText- Skip gram Model	23 minutes
Exp III: FastText- CBOW Model	24 minutes

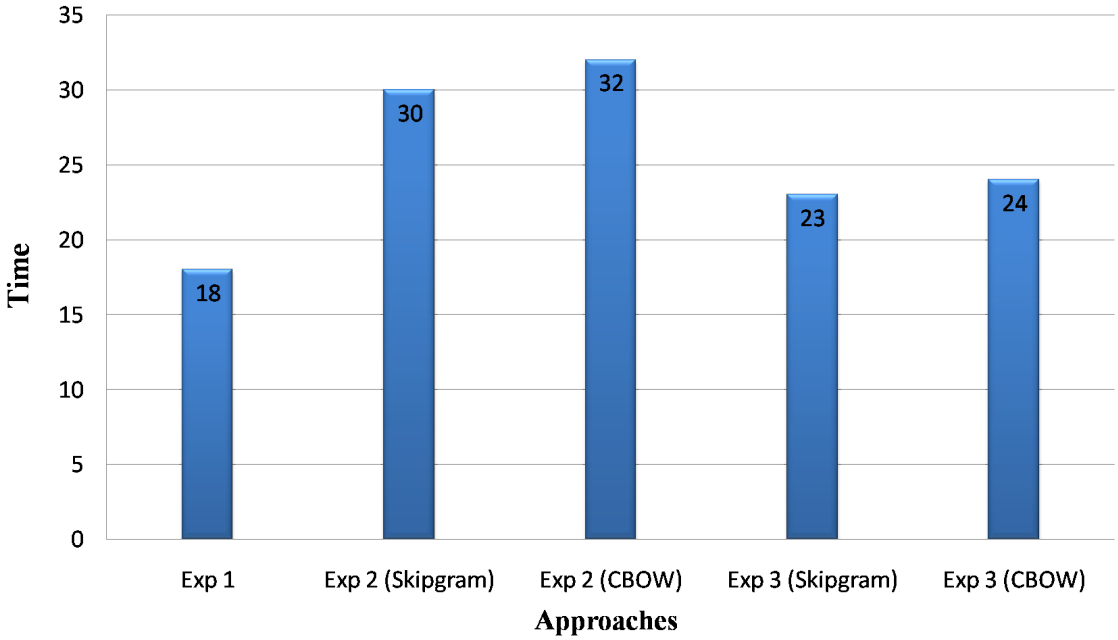

Figure 4.1: Training Time

4.7 Comparing The Word Embedding Models

From the given examples of the clusters produced by the three different models we can come to some decisions. If we consider clusters containing similar and synonymous words Word2Vec implementation of Tensorflow gives good results. Comparing the two variations of FastText model, the FastText-Skip Gram model is the best because it gives all the inflections of a specific word. The FastText-CBOW model do not produce such accurate results, rather it contains more noisy output than FastText-Skip Gram model's output. Gensim library based skip-gram model gives contextually similar words but fails to give inflection of words. The CBOW architecture of this model does not produce good clusters rather it gives noisy output. So from evaluating the results of these models, we can come to the conclusion that FastText-Skip Gram model is the more accurate and efficient model for building Bangla word clusters and consequently the more appropriate model for building the Bangla wordNet.

FastText uses n-grams of a word and create vectors for the sum of all the n-grams of the word. As a result it can produce output even if the word is not in the corpus. But the other approaches can not generate results for an unknown word. Though if we want to get cluster for a unknown word from FastText model, there was no satisfactory results but it did gave something. If the dataset can be prepared properly for the FastText skip gram model, we think it will produce really amazing and much more accurate word clusters. For our work, we proceed to build the WordNet structure with the clusters constructed by the FastText-Skip Gram model.

4.8 Hierarchy Building

The constructed word embeddings give us an idea about how the words are connected and how to establish a relationship among them. In the previous section we have seen that five different methods were applied to find a suitable method for building the hierarchical structure of the WordNet. The cluster results and the discussions in the result analysis chapter show that the FastText Skip-gram model gives the best results in case of including the inflection or different form of the target word. As we are building the WordNet structure from scratch, the words first need to be connected by their inflections. And here FastText Skip-gram model is the better suited model for that purpose. So we have chosen the embeddings produced by FastText Skip-gram model to build

the WordNet hierarchical structure. The structure is shown below.

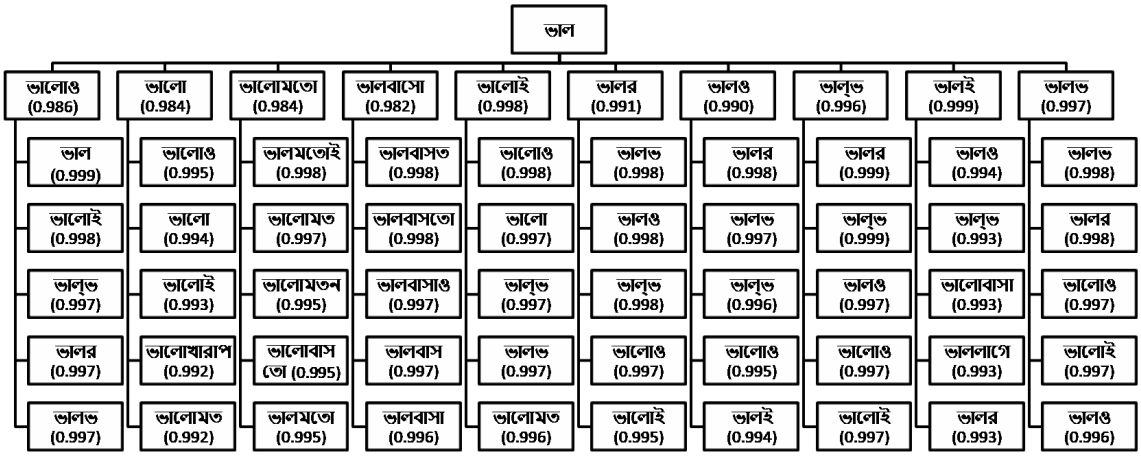

Figure 4.2: Hierarchy of Word relations along with cosine similarity

As we can see from the given structure, starting with the target word 'ভাল', its most similar ten words are connected to it making the first layer of the WordNet network. Next most similar five words of each of those ten words connect to its specific target word making the second layer of the WordNet network. The similarity is measured here by the cosine similarity between the words. This similarity was established by the dynamic word embedding models and it is used here as a measure for relativity between the words. For the ease of representation we have only shown the structure for one word here. Repeating this process for all the words in the corpus the complete WordNet structure will build up.

4.9 Adding Details to Hierarchy Structure

As discussed before, a complete WordNet contains the detailed information of the words as well as the connection among the words. The structure shown in the previous section is only the backbone of the WordNet. It has connected the words and constructed the network. But to make it a WordNet, we need to add details information like meaning, definition, uses, parts of speech and other grammatical information of all the words to the backbone structure. The process followed for doing this was discussed in the methodology chapter. Now in this section we will shed light on the outcome of that process.

We give an graphical representation of the output of this process in the following figure.

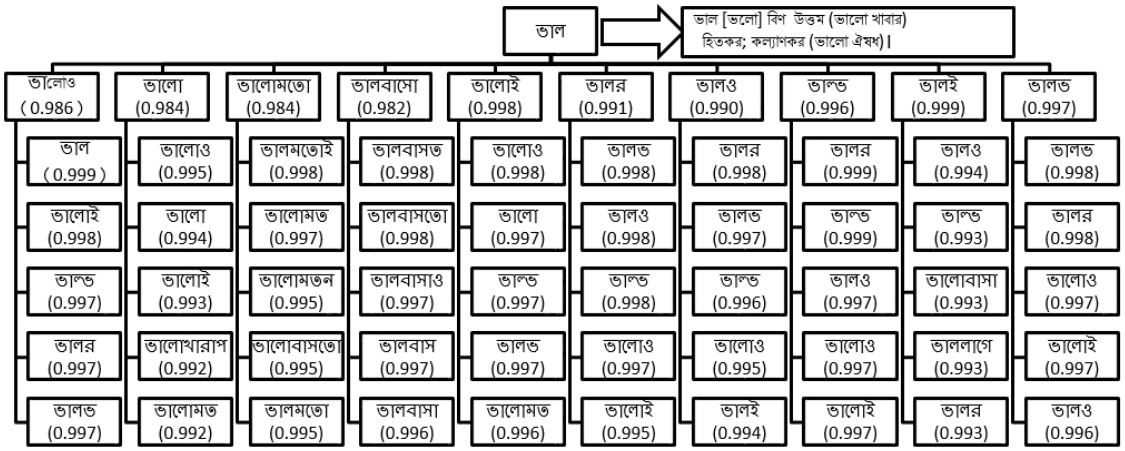

Figure 4.3: Hierarchy of Word relations along with cosine similarity

As can be seen from the figure comparing it with the previous figure, for the target word 'ভাল', its detailed information obtained via dictionary parsing is now added to the hierarchy structure. So the structure now contains the details of the word as well as the connected network. That is the WordNet structure. Here for the ease of representation only the details of the root word or target word is shown. But this process is done for all the words present in the network to complete the structure. Repeating this step for all the words present in the in the corpus, we get the complete WordNet structure.

Chapter 5

Discussion

5.1 Discussion

Complete construction of a WordNet from scratch for any language is a huge work. Our target was to construct a Bangla wordnet where the words will be related with their synonymous words and their inflection. For this we tried different dynamic word embedding models with various combination of vector size and window size to compare the constructed embeddings. As this is the basic of building the WordNet structure, finding an appropriate method for doing this was an important step of our implementation.

From observing the results for the constructed embeddings we can see that although they give good results, it can be improved by working with a bigger corpus covering vast areas of topic. A standard Bangla dataset is still scarce. If this process can be repeated with a standardized dataset better results can be obtained.

Another step was getting data from the Bangla dictionary. If the work is done with a better and more well processed dictionary, the process will improve further.

Previously WordNet building process was by machine translation of the Princeton WordNet. But our work shows that building the WordNet through word clustering process can be a promising method for WordNet construction for any language.

Chapter 6

Conclusion

A Bangla wordnet will be a big contribution in the field of Bangla natural language processing. This will open the doors to many promising contribution in many sectors of natural language processing.

We had started our work with the goal to improve efficiency from the previous works and use deep learning methods to achieve better performance in word embedding to construct the Bangla wordnet. In order to do so, We have compared different dynamic word embedding models for Bangla and received satisfactory results. We have also shown a comparison between these models which provides important insights on Bangla word embedding. This can help in further research works concerning Bangla word embedding process and the development of dynamic word embedding models for Bangla language.

The process shown in our implementation can be a promising method for WordNet construction for any language. Most of WordNet construction process follows the Princeton WordNet construction process, but this method proposes a new approach to WordNet construction and its outcome is quite satisfactory. Further work in this process will give new insights on this and can be a big step towards the digitalization of Bangla language.

References

[1] T. T. Urmi, J. J. Jammy, and S. Ismail, "A corpus based unsupervised bangla word stemming using n-gram language model," in *Informatics, Electronics and Vision (ICIEV), 2016 5th International Conference on*. IEEE, 2016, pp. 824–828.

[2] F. Faruqe and M. Khan, "Bwn-a software platform for developing bengali wordnet," in *Innovations and Advances in Computer Sciences and Engineering*. Springer, 2010, pp. 337–342.

[3] K. T. H. Rahit, M. Al-Amin, K. T. Hasan, and Z. Ahmed, "Banglanet: Towards a wordnet for bengali language."

[4] P. Bhattacharyya, "Indowordnet," in *The WordNet in Indian Languages*. Springer, 2017, pp. 1–18.

[5] "The 10 Most Spoken Languages In The World," https://www.babbel.com/en/magazine/the-10-most-spoken-languages-in-the-world/, accessed: 2018-07-01.

[6] "Princeton Wordnet," https://wordnet.princeton.edu/, accessed: 2018-07-01.

[7] S. K. N. Alok Ranjan Pal, Diganta Saha, "Word sense disambiguation in bengali: A knowledge based approach using bengali wordnet."

[8] S. Ismail and M. S. Rahman, "Bangla word clustering based on n-gram language model," in *Electrical Engineering and Information & Communication Technology (ICEEICT), 2014 International Conference on*. IEEE, 2014, pp. 1–5.

[9] M. Sinha, T. Dasgupta, A. Jana, and A. Basu, "Design and development of a bangla semantic lexicon and semantic similarity measure," *International Journal of Computer Applications*, vol. 95, no. 5, 2014.

[10] L. Yuan, “Word clustering algorithms based on word similarity,” in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2015 7th International Conference on*, vol. 1. IEEE, 2015, pp. 21–24.

[11] A. Al Hadi, M. Y. A. Khan, and M. A. Sayed, “Extracting semantic relatedness for bangla words,” in *Informatics, Electronics and Vision (ICIEV), 2016 5th International Conference on*. IEEE, 2016, pp. 10–14.

[12] A. Ahmad and M. R. Amin, “Bengali word embeddings and it’s application in solving document classification problem,” in *Computer and Information Technology (ICCIT), 2016 19th International Conference on*. IEEE, 2016, pp. 425–430.

[13] V. Rengasamy, T.-Y. Fu, W.-C. Lee, and K. Madduri, “Optimizing word2vec performance on multicore systems,” in *Proceedings of the Seventh Workshop on Irregular Applications: Architectures and Algorithms*. ACM, 2017, p. 3.

[14] L. Ma and Y. Zhang, “Using word2vec to process big text data,” in *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2895–2897.

[15] R. Bamler and S. Mandt, “Dynamic word embeddings,” *arXiv preprint arXiv:1702.08359*, 2017.

[16] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[17] W. Black, S. Elkateb, H. Rodriguez, M. Alkhalifa, P. Vossen, A. Pease, and C. Fellbaum, “Introducing the arabic wordnet project,” in *Proceedings of the third international WordNet conference*. Citeseer, 2006, pp. 295–300.

[18] Z. Vetulani and B. Kochanowski, “” polnet-polish wordnet” project: Polnet 2.0-a short description of the release,” in *Proceedings of the Seventh Global Wordnet Conference*, 2014, pp. 400–404.

[19] L. Kashyap, S. R. Joshi, and P. Bhattacharyya, “Insights on hindi wordnet coming from the indowordnet,” in *The WordNet in Indian Languages*. Springer, 2017, pp. 19–44.

[20] N. R. Ram and C. N. Mahender, “Marathi wordnet development,” *International Journal Of Engineering And Computer Science*, vol. 3, no. 08, 2014.

[21] I. K. A. N. Malhar Kulkarni, Chaitali Dangarikar and P. Bhattacharyya, “Introducing sanskrit wordnet,” 2010.

[22] T. Pedersen, S. Patwardhan, and J. Michelizzi, “Wordnet:: Similarity: measuring the relatedness of concepts,” in *Demonstration papers at HLT-NAACL 2004*. Association for Computational Linguistics, 2004, pp. 38–41.

[23] Z. Elberrichi, A. Rahmoun, and M. A. Bentaalah, “Using wordnet for text categorization.” *International Arab Journal of Information Technology (IAJIT)*, vol. 5, no. 1, 2008.

[24] R. Mandala, T. Takenobu, and T. Hozumi, “The use of wordnet in information retrieval,” *Usage of WordNet in Natural Language Processing Systems*, 1998.

[25] J. Gharat and J. Gadge, “web information retrieval using wordnet,” *International Journal of Computer Applications*, vol. 56, no. 13, 2012.

[26] S. Kolte and S. Bhirud, “Wordnet: a knowledge source for word sense disambiguation,” *International Journal of Recent Trends in Engineering*, vol. 2, no. 4, 2009.

[27] M. A. Al Mumin, A. A. M. Shoeb, M. R. Selim, and M. Z. Iqbal, “Sumono: A representative modern bengali corpus.”

[28] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, “Learning word vectors for 157 languages,” in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

Appendix A

Paper Published on Previous Work

Performance Analysis of Different Word Embedding Models on Bangla Language

Zakia Sultana Ritu
Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet, Bangladesh
zakiaritu.cse@gmail.com

Nafisa Nowshin
Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet, Bangladesh
nafisanowshin107@gmail.com

Md Mahadi Hasan Nahid
Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet, Bangladesh
nahid-cse@sust.edu

Sabir Ismail
Computer Science and Engineering
Stony Brook University
New York, United States
sabir.ismail@stonybrook.edu

***Abstract*—In this paper we discuss the performance of three-word embedding methods on Bangla corpus. Word embedding is a big part of natural language processing related research works. Many research works have focused on finding appropriate methods of word clustering process. Previously N-gram models were used for this purpose but now with the improvement of deep learning methods, dynamic word clustering models are preferred because they reduce processing time and improve memory efficiency. In this paper we discuss the performance of three word embedding models namely, word2vec in Tensorflow, word2vec from Gensim package and FastText model. We use same dataset on all the model and analyze the outcomes. These three models are applied on a Bangla dataset containing 5,21,391 unique words to produce the clusters and we evaluate their performance in terms of accuracy and efficiency.**

***Keywords*—Natural Language Processing(NLP) , machine learning, deep learning, word cluster, word embedding, Bangla word clustering, word2vec, fasttext, skip-gram, CBOW, GloVe.**

I. INTRODUCTION

Bangla is a major world language and it will increase more important in the coming years to come with the digitization of the Bangla language. As the importance of this language grows so does the research works concerning Bangla language. One of the major sectors of Bangla natural language processing involves establishing a method for producing fast and accurate word clusters. Much work is being done in this sector and continuous effort is being given to find an appropriate method for producing word clusterings. In this paper we attempt to apply three dynamic word clustering models to compare their performance in producing Bangla word clusterings.

Word clustering can be referred to as a technique for partitioning sets of words into subsets of semantically similar words. This has a far reaching effect in many NLP related works. It is increasingly becoming a major technique used in a number of NLP tasks ranging from word sense or structural disambiguation to information retrieval and filtering. To reach any decision about the performance of a model in producing word clusterings, we need to evaluate its performance by applying it to a large dataset and compare the results. But working with large dataset means more run time and consequently less efficiency. On the other hand, if it is done with smaller dataset, the clusters won't be accurate. So in choosing a word clustering method our main goal is to reduce run time and increase efficiency in producing accurate word clusters.

Word clusters include semantically similar words, meaning it will group those words that are similar in meaning and tend to occur in similar contexts in natural language. There are many approaches to compute semantic similarity between words based on their distribution in a corpus. Much research work has been done to find an efficient and accurate model for building word clusters. Although at first N-gram models were used to construct word clusters, in recent times with the improvement of deep learning methods dynamic models have become more popular in building word clusters. Dynamic models construct the vector representation of words and build clusters from them. In this paper we discuss the performance of three variations of dynamic word clustering models, which are, word2vec in Tensorflow, word2vec from Gensim package and FastText model in case of constructing Bangla word clusters.

This paper is arranged as follows, in section 2, we have shed light on some of the previous approaches to construct word clustering in Bangla and other languages and their performance. In section 3 we have discussed about the dataset that we have used in evaluating the performance of the models. In section 4 we discuss in brief the full methodology applied in our work. Then in section 5 we present an analysis of the results we got from applying these three models and evaluate their performance in producing accurate Bangla word clusterings. We conclude in section 6.

II. BACKGROUND STUDY

Word clustering is an important aspect of dealing with large datasets in research work. So, much research work has been done with a view to finding appropriate methods for constructing word clustering in Bangla and in various other languages. We will discuss some of these works below.

Previous word clustering techniques mostly involved using N-gram model to construct the clusters. This can be observed in the works of Ismail and Rahman [1], who proposed a Bangla word clustering method based on N-gram Language Model. In this paper they tried to cluster bangla word using their

semantic and contextual similarity. In this approach they tried to cluster the words based on the idea that, the words that have similar meaning and are used in similar context in a sentence, belong to the same cluster.

Their work was slightly upgraded later by Urmi, Jammy and Ismail [2]. They proposed a unsupervised learning approach to identify stem or root of a Bangla word from contextual similarity of words. Their object was to build a big corpus of Bangla stems along with their respective inflectional form. They worked with the assumption that if two words are similar in spelling and are used in similar context in many sentences, they have a higher chance of originating from the same root. They implemented 6-gram model for stem detection and achieved an accuracy of 40.18%. They have concluded that with big amount of text data this model will improve further.

Researchers then focused on producing word clustering in dynamic approach and its performance. We get insights about this from the works of Yuan [3], who showed that word clustering technique that is based on word similarities is better than conventional greedy approach in terms of speed and performance. The basic approach of this work was to check for a certain word in the corpus, its co-occurring words for similarity. That is to say, if two words are similar, their co-occurring word pattern will also be similar. Based on this they computed word clusters and when compared with other clustering methods, this approach was found to be more efficient.

The performance of dynamic models in producing Bangla word clusters was shown by Ahmed and Amin [4]. They discussed the effect of Bangla word embedding model in document classification. They worked with a dataset prepared from Bangla newspaper documents. They applied word2vec model to generate vector representation of words for word clustering. Using this they prepared clustering of word embeddings that are found in close proximity to each other in feature space. This information was later used as features to solve Bangla document classification problem.

Altszyler, Sigman and Slezak [5] tried to find out if LSA and word2vec model's capacity to identify relative dimension increases with increase in data. They found out that Word2vec can take advantage of all types of documents while LSA only gives better performance when out-of-domain documents are removed from corpus.

In case of Arabic language Soliman, Eissa and El-Beltagy [6], found that the performance measure of word2vec differs from dataset to dataset but on each dataset it shows good performance in capturing similarity among words.

Upgrading the performance of word2vec in finding vector representation of words in huge datasets like a dataset containing one billion words were attempted by Rengasamy, Fu, Lee and Madduri [7]. They applied word2vec in a multi-core system and found that this approach is 3.53 times faster than original multi-threaded word2vec implementation and 1.28 times faster than recent parallel word2vec implementation.

Ma and Zhang [8] discussed the effect of word2vec in reducing the dimensionality of large datasets. They found out that, in dealing with large scale training data, word2vec helps in clustering similar data. This strategy can reduce data dimension and speed up multi-class classifications.

With the goal of preparing vector representation of words, Naili, Chaibi, Ghezala [9], applied LSA, Word2vec and GloVe on both English and Arabic language. They reached the conclusion that although all three methods performance depend on the language used, among the three, word2vec gives the best vector representation of words.

Robert Bamler and Stephan Mandt [10] tried to find the semantic evolution of individual words over time in time-stamped datasets. They applied Word2vec model to produce the embedding vectors. They showed experimentally that both skip-gram filtering and smoothing lead to smoothly changing embedding vectors that help predict contextual similarities at held out time stamps.

Fasttext model is a relatively new model ventured in producing word clusterings. It is a variation of skip gram model architecture of word2vce model which was proposed by Bojanowski, Grave, Joulin and Mikolov [11]. The method they followed was, each word was represented as a bag of character n-grams and vector representation was constructed from them. This allowed them to construct word clusters for words not present in the training data. They concluded that this method gives state of the art word representations for both similarity and analogy task.

Finally, we can say that there is rich literature growing on word embedding techniques and there is much scope of improving in this sector.

III. DATA COLLECETION

We used three separate corpus, and merged them. First corpus is, SUMono [12] which contains available online and offline Bangla text data. We also used a news corpus, which contains news data from Bangla news websites. We also used Bangla wiki data from wikipedia. The detail is given in table I. Accuracy of any model largely depends on the dataset it is applied on. If a word is used in various kind of sentences, then the trained model can be more accurate as it covers a large area of variety. The more frequent the words are, the more accurate the model will be. This corpus contains Bangla text data on various topics. Because the corpus was built taking contents from various sources like Bangla articles from wikipedia, Bangla news portals and from writings of different renowned Bengali writers. As the text is collected from various sources it covers topics of different kinds and sectors as well as the language structure of Bangla used in day to day life. This is an important aspect of the data collected, because to get better and accurate clusters dynamically we need data that covers vast areas in which a specific word can be used.

Figure I represents some of the most frequent words from the corpus. And detailed information of the corpus is represented in table I.

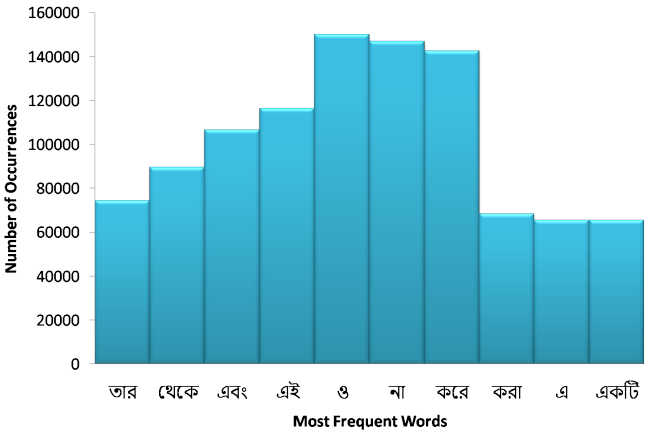

Figure 1. Histogram of Most Frequent Words with Number of Occurrences

Table I
Details of the Corpus

Total sentences	1,593,398
Total words	2,51,89,733
Unique words	5,21,391

IV. METHODOLOGY

Over the years many techniques have been introduced for word clustering. Most of the approaches give high accuracy for clustering English words. As Bangla is a more complicated language, it is hard to gain high accuracy. We attempted three different approaches to determine which approach works better for Bangla word clustering. Nowadays vector representation of words have become the most popular approach for building word clusters. We applied three machine learning approaches which are based on vector representation of Words. We followed some steps to train our datasets. these steps are discussed below.

A. The Basic Steps

- **Corpus:** The corpus is stored as a text file. In the text file the data constitutes of Bangla sentences. They can be treated as strings. But We can not feed the strings directly to this model, so some pre-processing was done on the dataset.
- **Tokenizing:** We had to pre-process the corpus in order to use them as our proper input. Firstly, we couldn't feed a word just as a text string to a model. For this, we tokenized each word. Example:

আমার সাথে বাংলায় কথা বল => 'আমার', 'সাথে', 'বাংলায়', 'কথা', 'বল'

- **Training:** We used the tokenized dataset for training. We compared the output from each model individually by using different window sizes, vector sizes and iterations over the dataset.

B. Our Experiments

1) ***Experiment I: Word2vec in Tensorflow****:* We used the official version of sentence embedding implementation of tensorflow. The code generates word clusters based on the features of Word2vec using the Skip-Gram model and the Negative Sampling accelerated classification algorithm. We tried different window sizes, vector sizes and iterations.

2) ***Experiment II: FastText Model****:* Facebook's AI Research lab created the FastText library for word embedding and text classification. Both Skip gram and CBOW model can be trained with FastText. For the training of Skip Gram model, we kept the window size at 5 and vector size was 100. we trained the CBOW model as well.

3) ***Experiment III: Word2vec from Gensim package****:* The python library Gensim provides Word2Vec class for working with word embedding. We tuned the parameters and checked results both for Skip gram and CBOW model.

C. Training Time

The size of our corpus is 59856174 bytes. We used a computer with 4GB RAM, core i3-3110M CPU. Training Time for each experiment is as follows-

Table II
Training Time of the Experiments

Experiment	Training Time
Word2Vec in Tensorflow	18 minutes
FastText- Skip gram Model	23 minutes
FastText- CBOW Model	24 minutes
Gensim Word2Vec- Skip gram Model	30 minutes
Gensim Word2Vec- CBOW Model	32 minutes

V. RESULT ANALYSIS

There are similarities among the clusters from three different approaches as well as some significant differences. For some words, we got a similar set of words for a set of models. But the variety was also notable. We tried to get the most satisfactory results from each approach. We applied various combination of window and vector sizes in order to tune the parameters to get the most optimum and satisfactory results. Table III shows parameter tuning for each approach.

Table III
Parameter Tuning for Optimum Results

Experiment	Window size	Vector size	Iteration
I- Word2vec in Tensorflow	4	1000	10
II- Skip gram Model	5	100	5
II- CBOW Model	5	100	5
III- Skip gram Model	5	400	5
III- CBOW Model	5	400	5

Some sample results from the experiments are given in table IV, V, VI, VII and VIII.

Table IV
Results from Experiment I

Random Word	Words on cluster
আমরা	আমাদের, আমি, চাই, যখন, তাই, তারা, কি, সেই, কিন্তু, সবাই
তাঁর	তার, সেই, সাথে, তাঁদের, একজন, একই, ওই, তিনি, পরে, বলে
জন্য	প্রয়োজন, সুযোগ, জন্যে, পাশাপাশি, তাই, দরকার, কারণ, কিছু, কিন্তু, তাদের
কোন	কোনো, এমন, অন্য, কারণ, তবে, সেটা, বা, নেই, তাই, এখনো
পারে	হবে, পারবে, পারি, হলে, পারেন, হতো, চাই, চায়, পারেনি, থাকে
হতে	যেতে, থাকতে, করতে, হলে, না, রাখতে, তাই, তাহলে, তবে, দিতে
বড়	সবচেয়ে, অবস্থা, খুবই, খুব, আমাদের, অনেক, কিছু, মতো, মানুষের, আছে
টাকা	হাজার, লাখ, কোটি, টাকার, খরচ, পাঁচ, মাত্র, বিক্রি, তিন, প্রায়
নতুন	মাধ্যমে, তৈরি, কাজ, জন্য, বিভিন্ন, নানা, একটি, এই, সব, একই
দেখা	অদেখা, দেখাইত, দেখাক, দেখায়ই, দেখাসহ, দেখায়া, দেখাএইসবই, দেখাত, দেখাবো, দেখাবো

Table V
Results from Experiment II-Skip Gram

Random Word	Words on cluster
আমরা	আমরা।, কীআমরা, নয়আমরা, আমরাই, আমরাতো, হয়আমরা, কিআমরা, হোকআমরা, আমর, লেআমরা
তাঁর	তাঁর।, তাঁরই, তাঁরও, তাঁহা, তাঁরা, তাঁতীও, তাঁরাই, তাঁ, তাঁকেসহ, ওঁর
জন্য	জন্য।, জন্যেঃ, জন্যও, জন্যে।, সৌজন্য, জন্যে, এজন্য, জন্যই, জন্যি, এরজন্য
কোন	কোনস, কোন্, কোন্যো, কোনোও, কোনো, কোনডা, কোন্, কোনোই, কোনও, কোনই
পারে	পারো, পারে।, পারেএ, পারেতখন, পারেঃ, পারেআর, পারেএমন, পারেনই, পারেন।, ঐপারে
হতে	ঝতে, লখনউতে, ধতে, ইইউতে, নড়তে, নতে, পেতে, চড়তে, অইতে, ওতে
বড়	বড়বড়, বড়র, হড়বড়, বড়ও, বড়ইর, বড়ছোট, ছোটবড়, ছোটছোট, ছোট, নড়বড়
টাকা	টাকা।, দশটাকা, টাকায়ও, টাকাসহ, হাজারগুণ, দুইটাকা, হাজারও, দুটাকা, হাজারদিঘী, টাকাকী
নতুন	নতুননতুন, নতুনতর, নতুন।, নতুনই, নত, নতূন, জৈতুন, নিত্যনতুন, নতুনরা, চালু
দেখা	অদেখা, দেখাইত, দেখাক, দেখায়ই, দেখাসহ, দেখায়া, দেখাএইসবই, দেখাত, দেখাবো, দেখাবো

Table VI
Results from Experiment II- CBOW

Random Word	Words on cluster
আমরা	আমরাযদিও, আমরাই, আপনাকেও, আমরাও, বলেছিআমরা, আপনা, কীআমরা, কীটও, আপনাকেই, কীটস
তাঁর	পুনঃআলোচনার, সুলোচনার, আলাপআলোচনার, কেস্তার, তাঁরই, কাইয়ুমআলোচনার, সৃষ্টিশীলতার, ধিক্কার, মনিকার, বিঘার
জন্য	জন্যও, জন্য।, জন্স, সৌজন্য, এজন্য, জন্যেঃ, সৌজন্যঃ, জন্যই, জন্যে।, তজ্জন্য
কোন	কোনোও, কোন্যো, কোনোই, কোনো, লুকোনো, থোন, কোনস, কোন্, কোন্, কোনোটিই
পারে	পারেআর, পারেতখন, পারে।, পারেএ, পারো, পারডন, পারেঃ, পারেও, পারক্কর, পারদ
হতে	কইরতে, ঝরতে, ভরতে, কসরতে, খোরতে, শরতে, মরতে, ধরতে, ঠকতে, কুদরতে
বড়	বড়র, বড়ও, হড়বড়, বড়ইর, বড়ই, বড়সড়, গড়বড়, বড়সড়ো, নড়বড়, বড়বাড়ী
টাকা	দশটাকা, ওসাকা, শলাকা, পোঁদপাকা, গাঢাকা, টাকা।, জলধাকা, ইয়াকা, হাজারী, এলাকা
নতুন	নতুননতুন, নতুন।, 'নতুনই, ফোরামটি, অনুষ্ঠিত, পুনর্গঠিত, তিনচারটি, উৎকণ্ঠিত, পরিচালকমণ্ডলীর, ভূখণ্ডটি
দেখা	অদেখা, দেখায়ই, দেখাসহ, দেখাইত, দেখাএইসবই, দেখাত, দেখাক, দেখাবো, দেখায়া, দেখাও

Table VII
Results from Experiment III- Skip Gram

Random Word	Words on cluster
আমরা	আমি, তোমরা, সেটা, তাহলে, পারি, এখনো, হয়তো, এখানে, করেছি, তোমাকে
তাঁর	তাঁদের, স্বাধীনতার, যিনি, নেন, নিজ, মেডিকেল, দেন, করছিলেন, পরিবারের, যুদ্ধ
জন্য	জন্যে, সুযোগ, চেষ্টা, কাজে, উদ্দেশ্যে, মাধ্যমে, ব্যাপারে, ব্যবস্থা, করলে, পর্যায়ে
কোন	কোনো, থাকার, ছাড়া, তাতে, বসানোর, আপত্তি, প্রয়োজন, তেমন, উপায়, এমন
পারে	পারবে, চায়, পারেনি, পারেন, পারত, হতো, পারব, বাধ্য, চাই, পারবেন
হতে	পেতে, যেতে, থাকতে, রাখতে, আনতে, দোকানেও, নিতে, ঘটতে, বেশিও, লাগতে
বড়	ছোট, সবচেয়ে, শিরোনামায়, মেয়ে, গায়কেরা, সুন্দর, জোরটা, জিনিস, মধ্যবিত্তদের, ছেলে
টাকা	কোটি, হাজার, পাঁচ, টাকার, লক্ষ, বরাদ্দ, লাখ, বছর, প্রায়, গত
নতুন	নির্যাতন, প্রক্রিয়া, জাতীয়, পূর্বে, মামলা, এলাকায়, সেনাবাহিনী, অর্থনৈতিক, বিচারের, জোট
দেখা	কমে, রয়ে, ফুসফুসে, গবেষণায়, জানা, বেড়ে, পাওয়া, রোগী, রূপ, জীবদ্দশা

Table VIII
Results from Experiment III- CBOW

Random Word	Words on cluster
আমরা	ইয়ার্কদের, যতই, পৌঁছাতে, হোক, লিডাররা, শেয়ারিঙের, অবশ্যই, কথাও, চিন্তা, দুবারই
তাঁর	তাঁরা, তৈরিতে, কো, সারা, বর্জনের, সৃষ্টিতে, ভাইরাস, অপরিশোধিত, কলফনি, লিপির
জন্য	যুক্তি, চিহ্নিত, জোরদার, পরিবেশ, আশ্বাস, প্রকাশ, সমালোচনা, অন্তর্ভুক্ত, প্রত্যাখ্যান, প্রতিরোধ
কোন	কোনো, প্রত্যয়, কিছুরই, কারণ, বইতে, তেমনভাবে, আধ্যাত্মিকতা, সাবজেক্টে, ঘটেনি, ইন্ডাস্ট্রির
পারে	ঈর্ষায়, দগ্ধ, অভিবাসী, পাঠ্যবই, পারবেন, পারত, পারবে, পেরেছিল, অস্বীকৃতি, নামিজউদ্দিন
হতে	হতেই, থাকতে, দিতে, যাইতে, ফিল্মগুলো, পেতে, নিতে, সাজিয়া, ঘটাতে, জানতে
বড়	ফুটো, শ্লেষ্মা, দরজাটা, আলো, পাথরের, লেজের, চওড়া, রাস্তা, অন্ধকার, খাটো
টাকা	বছরের, 'ছয়, চলাচলের, দশ, সপ্তাহের, পূর্ববঙ্গে, লাখ, জেলার, উপলক্ষে, গান্ধীর
নতুন	গান্ধীকে,পাকিস্তান, বন্টন, ক্রয়, মন্তব্য, শিক্ষা, পরিকল্পনার, অধিগ্রহণ, মাউন্টব্যাটেন, প্রস্তাবিত
দেখা	পাওয়া, জানা, পাহারায়, বেঁকে, কাঁচুলি, ব্লেজারেই, উর্দুকে, শহরগুলোর, ঝরেতুমি, গেয়ে

From the given examples of the clusters produced by the three different models we can come to some decisions. If we consider clusters containing similar and synonymous words Word2Vec implementation of Tensorflow gives good results. Comparing the two variations of FastText model, the FastText-Skip Gram model is the best because it gives all the inflections of a specific word. The FastText-CBOW model do not produce such accurate results. Gensim library based skip-gram model gives contextually similar words but fails to give inflection of words. The CBOW architecture of this model does not produce good clusters rather it gives noisy output. So from evaluating the results of these models, we can come to the conclusion that FastText-Skip Gram model is the more accurate and efficient model for building Bangla word clusters.

FastText uses n-grams of a word and create vectors for the sum of all the n-grams of the word. As a result it can produce

output even if the word is not in the corpus. But the other approaches can not generate results for an unknown word. Though if we want to get cluster for a unknown word from FastText model, there was no satisfactory results but it did gave something. If the dataset can be prepared properly for the FastText skip gram model, we think it will produce really amazing and much more accurate word clusters.

VI. CONCLUSIONS

Bangla is a complex language with a wide range of vocabulary containing many rare words. Language structure, use of complex words, multiple meanings in different context all of these reasons makes it really difficult to choose one model as the best model for Bangla word clustering. The contents of the dataset also plays a big role in deciding this. We have tried to give some perspective on some of the dynamic approaches that have been used for Bangla word clustering. Among the models applied, we have reached the conclusion that FastText-Skip Gram model produces the best result on the given dataset. We can get more accurate results by increasing the size of the dataset.

References

[1] S. Ismail and M. S. Rahman, "Bangla word clustering based on n-gram language model," in Electrical Engineering and Information & Communication Technology (ICEEICT), 2014 International Conference on. IEEE, 2014, pp. 1–5.

[2] T. T. Urmi, J. J. Jammy, and S. Ismail, "A corpus based unsupervised bangla word stemming using n-gram language model," in Informatics, Electronics and Vision (ICIEV), 2016 5th International Conference on. IEEE, 2016, pp. 824–828.

[3] L. Yuan, "Word clustering algorithms based on word similarity," in Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2015 7th International Conference on, vol. 1. IEEE, 2015, pp. 21–24.

[4] A. Ahmad and M. R. Amin, "Bengali word embeddings and it's application in solving document classification problem," in Computer and Information Technology (ICCIT), 2016 19th International Conference on. IEEE, 2016, pp. 425–430.

[5] E. Altszyler, M. Sigman, and D. F. Slezak, "Corpus specificity in lsa and word2vec: the role of out-of-domain documents," arXiv preprint arXiv:1712.10054, 2017.

[6] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "Aravec: A set of arabic word embedding models for use in arabic nlp," Procedia Computer Science, vol. 117, pp. 256–265, 2017.

[7] V. Rengasamy, T.-Y. Fu, W.-C. Lee, and K. Madduri, "Optimizing word2vec performance on multicore systems," in Proceedings of the Seventh Workshop on Irregular Applications: Architectures and Algorithms. ACM, 2017, p. 3.

[8] L. Ma and Y. Zhang, "Using word2vec to process big text data," in Big Data (Big Data), 2015 IEEE International Conference on. IEEE, 2015, pp. 2895–2897.

[9] M. Naili, A. H. Chaibi, and H. H. B. Ghezala, "Comparative study of word embedding methods in topic segmentation," Procedia Computer Science, vol. 112, pp. 340–349, 2017.

[10] R. Bamler and S. Mandt, "Dynamic word embeddings," arXiv preprint arXiv:1702.08359, 2017.

[11] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," Transactions of the Association for Computational Linguistics, vol. 5, pp. 135–146, 2017.

[12] M. A. Al Mumin, A. A. M. Shoeb, M. R. Selim, and M. Z. Iqbal, "Sumono: A representative modern bengali corpus."