Word Sense Induction for Bengali Language

Project submitted

In partial fulfilments of the requirements for the degree of

MASTER OF COMPUTER APPLICATION

By

PARAG MITRA

Roll No: **MCA186005**

Registration No: **133667 of 2015-2016**

Under the supervision of

DR. SUDIP KUMAR NASKAR

Department of Computer Science & Engineering

Faculty of Engineering and Technology

Jadavpur University

Kolkata – 7000 032

India

Jadavpur University

Faculty of Engineering and Technology

Department of Computer Science & Engineering

CERTIFICATE

This is to clarify that the project entitled "**Word Sense Induction for Bengali Language**" has been completed by Parag Mitra. This work is carried out under the supervision of Dr. Sudip Kumar Naskar in partial fulfilment for the award of the degree of Master of Computer Application of the department of Computer Science and Engineering, Jadavpur University, during the session 2017-2018. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

_____

Dr. Sudip Kumar Naskar
Project supervisor
Computer Science & Engineering
Jadavpur University

Countersigned:

_____

Prof. Ujjwal Maulik
Head of the department
Computer Science & Engineering
Jadavpur University

_____

Prof. Chiranjib Bhattacharjee
Dean
Faculty of Engineering & Technology
Jadavpur University

Jadavpur University

Faculty of Engineering and Technology

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled **"Word Sense Induction for Bengali Language"** has been submitted by Parag Mitra in partial fulfilment of the requirements for the award of the degree of **Master of Computer Application** in the department of **Computer Science & Engineering**, Jadavpur University, during the period 2017-2018 has been carried out under my supervision and that this work has not been submitted elsewhere for obtaining a degree.

EXAMINER:

_____ _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION OF ORIGINALITY
AND
COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this project contains original work by the undersigned candidate, as part of his Master of Computer Application (MCA) studies.

All information in this document have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material results that are not original to this work.

Candidate's Name: Parag Mitra

Exam Roll No: MCA186005

Project Title: Word Sense Induction for Bengali Language

Signature with Date:

ABSTRACT

One of the most challenging research problems in natural language processing is that of unsupervised word sense disambiguation, formally known as word sense induction or discrimination. It involves discovering senses of a word given its contexts of usage without the use of a sense inventory which differentiates this task from traditional word sense disambiguation. This paper reports a work on sense induction in Bengali. The proposed method is based on K-Means clustering which uses the Bag of Words(BOW) model and it further uses context vectors made from parallel corpora. The proposed method shows a very strong performance in the word sense induction task in Bengali.

Contents

CHAPTER 1 INTRODUCTION

1.INTRODUCTION:

1.1 Word Sense Induction (WSI):

The main task of Word Sense Induction(WSI) is to determine the number of senses of a given word and also to induce the different senses of that word from several contexts of a corpus. Word sense induction (WSI) is a very important and open research problem in natural language processing (NLP) and computational linguistics. It works in the form of an unsupervised learning task with senses represented as clusters of token instances, where each cluster captures a particular sense of that word.

The manual construction of a sense inventory is a tedious and time-consuming job, and the result is highly dependent on the annotators and the domain at hand. By applying an automatic procedure, we are able to only extract the senses that are objectively present in a particular corpus, and it allows for the sense inventory to be straight forwardly adapted to a new domain.

WSI has potential to be extremely useful in downstream applications because, apart from the savings on annotation costs, it also mitigates several theoretical conflicts associated with supervised WSD tasks, which generally involve deciding on the granularity of senses.

The goal of a WSI is to computationally assign the correct sense of a word (i.e. meaning) in context (phrase, sentence, paragraph, text) also to explore a new meaning if exists from a predefined sense inventory, when the word has multiple meanings. It is a pervasive characteristic of natural language. The problem is that words often have more than one meaning, sometimes fairly similar and sometimes completely different. For example, the word bank has several senses and may refer to the edge of a river, a building, or a financial institution. The specific sense intended is determined by the textual context in which an instance of the ambiguous word appears. In “The boy leapt from the bank into the cold water.” the edge of a river is intended, whereas in “The van pulled up outside the bank and three masked men got out.” the building sense is meant, while in “The bank sent me a letter.” implies the financial institution sense.

As WSI deals solely with contexts obtained from a corpus, the exploration of the problem space gets restricted. This is because a corpus may not always contain sentences (contexts)

encapsulating all the senses a word. On the other hand, a corpus may reveal new and previously unknown senses of a word e.g., *'tablet'* which traditionally meant *'pill'* or *'stone slab'* but in modern times has also come to mean a type of *'electronic device'*.

Thus, discovering novel or new senses of a word also falls in the domain of WSI. In the word sense disambiguation (WSD) task, a sense inventory is a prerequisite, based on which the meaning of a word is disambiguated or understood in a particular context. Knowledge-based WSD algorithms, as the name suggests, uses rich knowledge bases such as WordNet, dictionaries or thesauri which provides deeper knowledge on senses such as their definitions, relationships with other senses etc.

Initially, WSI was mainly applied and developed on English texts, because of the broad availability and the prevalence of lexical resources compared to other languages. Due to the lack of lexical resources i.e. sense inventories (dictionaries, lexical databases, Wordnets, etc.) and sense-tagged corpora it is difficult to start working on WSD for under-resourced languages (Bangla, Assamese, Oriya, Kannada, etc.). To account for under-resourced languages, one can easily adopt techniques aimed at the automatic discovery of word senses from text, a task called Word Sense Induction (WSI).

Ideally, a WSI algorithm would be able to adapt to different tasks requiring different sense granularities. WSI algorithms can also be used to model the evolution of the senses of a word with time and hence can be much easier to maintain than existing fixed sense inventories like WordNet (Miller, 1995), Ontonotes (Hovy et al., 2006) etc. Automatic sense identification systems also have the potential to generalize well to large amounts of diverse data and hence be useful in various difficult domain independent tasks such as machine translation and information retrieval.

Several factors make the problem of word sense induction very challenging. Most importantly, it is not clear what should be the 'true' senses of a word. The semantic continuum makes it always possible to break a sense into finer grained sub-senses.

Thus, the problem is one of finding the optimal granularity for any given task. Even in a semi-supervised setting, it is unknown which sense inventories are most suited as starting points in a sense bootstrapping procedure.

Supervised approaches to WSD require significant amount of manually annotated training data for correctly predicting the sense of an ambiguous word. These methods are very proficient and generally obtain better precision than their knowledge-based

counterparts. However, supervised approaches suffer from a major bottleneck, i.e. they are dependence on large amount of sense annotated training data making these methods unsuitable for under-resourced languages. This is where WSI comes to the rescue. WSI aims to induce or discover senses whereas WSD aims to assign senses. Owing to WSI's unsupervised nature, it has potential application to sense discovery tasks in low-resourced languages where sense inventories are either poorly developed or non-existent. It is this drawback of WSD, i.e. reliance on manually annotated data, which WSI aims to overcome by employing advanced clustering techniques and exploiting raw lexical data that are available in corpus. WSI has many applications in several other NLP task such as information retrieval, machine translation, novel sense detection, etc.

1.2 Motivation:

In this project we are going to do the task of Word sense Induction (WSI) that means to find the number of senses of a given word present in the contexts of a corpus. Also we will find the accuracy of our method. Our approach is an unsupervised way of learning, that means we will use clustering method for this job using translation based features. Number of cluster will determine the number of senses for the targeted polysemous word.

Languages with large amounts of data, or funding, or political interests can be interpreted as 'well-resourced' languages, whereas, a lot of languages in the world do not enjoy this status, which is referred to in this article as 'under-resourced' languages. This paper presents the state of the art of WSI in an under-resourced language perspective.

Moreover, we are going to do this job for the Bengali polysemous words. Bengali is not a high resource language like English, so we have to face some challenges with this approach. So, we are interested to analyse the result for Bengali polysemous words.

Our data set will contain a large number of contexts that means sentences in which our targeted polysemous word will be present. The corresponding Bengali contexts have been collected from a parallel corpus.

1.3 Introduction to later Chapters:

In Chapter 2 we will discuss on the Literature Survey i.e. the Previous works on WSI.

Then in the next chapter i.e. **In chapter 3** first we will discuss about the proposed methodology in the **article 3.1.** Then in **article 3.2** we will provide our data sets for this job. Next in **article 3.3** we will show our experimental results and finally in **article 3.4** we will do a detailed analysis on the experimental result.

In chapter 4 we will do a brief and generalise analysis of the experimental results in the **article 4.1**. And the in **article 4.2** we will discuss on the conclusions we have drawn from our analysis.

CHAPTER 2 LITERATURE SURVEY

2. Literature Survey:

Much of the work on word sense induction has been quite recent following the Semeval tasks on WSI in 2007(Agirre and Soroa, 2007) and 2010, but the task was recognized much earlier and various semi-supervised and unsupervised efforts were directed towards the problem. Yarowsky (1995) proposed a semi-supervised approach, which required humans to specify seed words for every ambiguous word and assumed one sense per discourse for an ambiguous word. The unsupervised approaches mainly focus on clustering the instances of the target words in a corpus, using first-order vectors, second-order vectors (Purandare and Pedersen, 2004) (Schutze, 1998) etc. Co-occurrence graph-based approaches (Veronis, 2004) have also been used, which represent the words co-occurring with the tar- ´ get words as nodes and then identify the highly dense subgraphs or 'hubs' within this co-occurrence graph. Brody and Lapata (2009) and Lau et al. (2012) proposed bayesian WSI systems which cluster the instances by applying Latent Dirichlet Allocation (LDA)(Blei et al., 2003), Hierarchical Dirichlet Processes (HDP)(Teh et al., 2006) etc. wherein each occurence of a target word is represented as a 'document' and its surrounding context as the 'observable content'. Choe and Charniak (2013) propose a 'naive bayes' model for WSI which assumes one sense per discourse and uses Expectation Maximization(EM) to estimate model parameters like the probability of generating an instance feature like a word in the context, given the sense of the target word in a particular instance. Reisinger and Mooney (2010) and Huang et al. (2012) have proposed sense dependent multiple prototypes for a word instead of the conventional one vector representation per word and have shown that this sense differentiation improves semantic similarity measurements between words.

Firth (1954) paved the way for modern day WSI with his famous quote *"You shall know a word by the company it keeps"* which today is referred to as the *"Distributional Hypotheses"*. WSI algorithms are of two types viz. local and global. Local algorithms find the senses of a word on a per-word basis. They can be further classified as context-clustering algorithms and graph-based algorithms. Schütze first proposed the idea of context-group discrimination in 1998 and later many other researchers applied a similar approach to sense induction. In graph based methods the co-occurrence graph is created where each node represents a word and edges connect the words that are present in the same context (neighboring words within a context window or connected by dependency relations). Word senses are then calculated using graph clustering methods(Widdows & Dorow), Page-rank

algorithm (Agrrie et al., 2006), etc. Bordag(2006) proposed a word sense induction model using word triplets, based on the assumption of 'one sense per collocation' and performed clustering on the co-occurrence triplets.

Global WSI algorithms use the full-blown word space model to determine the different senses of a polysemous word by comparing them to the senses of other words. Pantel and Lin(2002) presented a global clustering algorithm that automatically extracts the word senses from text using their clustering by committee (CBC) technique. Here the concept is to find a set of unambiguous clusters (called committees) to which ambiguous word may be assigned. The centroid of a committee is taken to be its feature vector. When a word has been allocated to a cluster, the features common to both (centroid and assigned word) vectors are removed from the assigned word's vector. In this way, infrequent word senses are discovered.

Using word representations form by distributional semantic models (DSM) (M. Sahlgren) and using word embedding by neural net language models (NNLM) (Bengio et al., 2003) sense induction has been performed. Here the key idea is that words with similar distributions have similar meanings. Hope and Keller introduced a Max-Max soft clustering algorithm for WSI which is a linear time graph-based algorithm which showed results comparable with those of existing systems. Latent semantic analysis (LSA) (Landauer et al.,1998) (Landauer & Duamis.,1997) is also a popular model in this domain. In LSA, term document matrix is created for frequency counting of each word and singular value decomposition (SVD) is used to represent latent semantic dimensions.

The above mentioned WSI algorithms and techniques are performed on monolingual data. Apidianaki (2008) used bilingual data for WSI and devised a translation-oriented approach. It involved augmenting the source language context with target language equivalents. The process used bilingual corpora which was word aligned for the construction of two bilingual dictionaries where every word type is linked with its translation equivalents. Recently (Albano et al., 2014) proposed a cross-lingual sense induction model where up to five languages were used. The idea was novel and showed brilliant F scores for sense induction in English. They observed that as the number of languages used increased, so did the F score.

CHAPTER
3
WSI FOR BENGALI

3.WSI for Bengali Language:

3.1 Proposed Methodology:

Word Sense Induction(WSI) problem can be generalized as a clustering problem wherein either the contexts in which a polysemous target word occurs are clustered or a set of words related to it are clustered. There are 3 main approaches to WSI, such as- context clustering, word clustering and using co-occurrence graph. The proposed work follows the idea of context clustering.

A context is a small unit of neighbouring text, typically modelled by $\pm N$ surrounding words, or the sentence or paragraph containing the target word, although it can be longer in length. In the proposed work, we make use of sentential contexts i.e. the size of the contexts used is that of a single sentence. There can be two kinds of contexts for a word, namely headed contexts (which contain the target word) and headless contexts (which do not). The following examples for the target word *'bank'* describe the 2 types of contexts.

- "*John went to the **bank** on Saturday to cash his cheque*" (Headed Context)
- "*The **financial institution** visited by John is under heavy renovation!*" (Headless Context)

Both of the examples are talking about the '*building*' sense of the word '*bank*' but their representations are completely different. The first representation directly uses the word '*bank*' in a sentence having hints ('*cash*' and '*cheque*') to understand that the 'building' sense is being referred to while the second representation coveys the same idea ('*financial institution*') but without explicitly using the target word. While dealing with headed contexts, attention is directed towards the target word and clustering is performed based on its surrounding contexts whereas with headless ones, the goal is to concentrate on the context as a whole and cluster them based on their similarity to one another. In order to build a model for context clustering, vector representations for contexts must be defined. A context vector is a vector of the context where its dimensions are associated with features. These features maybe simple features like unigrams (single word), bigrams (ordered pair of words), etc., or more complex features like translations, sense embedding, etc. In the

proposed model for sense induction, we create context vectors of two types- unigram vectors and augmented vectors formed by taking union (combination) of both Bengali and their corresponding English context vectors.

3.2 The Difficulty of Characterizing word meaning:

There has been an enormous amount of work in the fields of WSD and WSI relying on a fixed inventory of senses and on the assumption of a single best sense for a given instance (for example, see the large body of work described in Navigli [2009]) though doubts have been expressed about this methodology when looking at the linguistic data (Kilgarriff 1998; Hanks 2000; Kilgarriff 2006).

One major issue arises from the fact that there is a spectrum of word meaning phenomena (Tuggy 1993) from clear-cut cases of ambiguity where meanings are distinct and separable, to cases where meanings are intertwined (highly interrelated) (Cruse 2000; Kilgarriff 1998), to cases of vagueness at the other extreme where meanings are underspecified.

For example, at the ambiguous end of the spectrum are words like bank (noun) with the distinct senses of financial institution and side of a river. In such cases, it is relatively straightforward to differentiate corpus examples and come up with clear definitions for a dictionary or other lexical resource.

These clearly ambiguous words are commonplace in articles promoting WSD because the ambiguity is evident and the need to resolve it is compelling. On the other end of the spectrum are cases where meaning is unspecified (vague); for example, Tuggy gives the example that aunt can be father's sister or mother's sister. There may be no contextual evidence to determine the intended reading and this does not trouble hearers and should not trouble computers (the exact meaning can be left unspecified).

Cases of polysemy are somewhere in between. Examples from Tuggy include the noun set (a chess set, a set in tennis, a set of dishes, and a set in logic) and the verb break (a stick, a law, a horse, water, ranks, a code, and a record), each having many connections between the related senses.

Although it is assumed in many cases that one meaning has spawned the other by a metaphorical process (Lakoff 1987)—for example, the mouth of a river from the mouth of a person—the process is not always transparent and neither is the point at which the spawned meaning takes an independent existence.

3.3. Bag of Words Model:

In this model, each word is assumed to be independent and the order in which they occur is immaterial. Each unique word is the same as another dimension in the new vector space and the component of a vector along this dimension is the frequency of the word. Hence, we can represent each document as this vector with each component containing the frequencies on each dimension.

The Problem with Text

A problem with modelling text is that it is messy, and techniques like machine learning algorithms prefer well defined fixed-length inputs and outputs. Machine learning algorithms cannot work with raw text directly; the text must be converted into numbers. Specifically, vectors of numbers.

In language processing, the vectors x are derived from textual data, in order to reflect various linguistic properties of the text.

— Page 65, Neural Network Methods in Natural Language Processing, 2017.

This is called feature extraction or feature encoding.

A popular and simple method of feature extraction with text data is called the bag-of-words model of text.

What is a Bag-of-Words?

A bag-of-words model, or BoW for short, is a way of extracting features from text for use in modelling, such as with machine learning algorithms. The approach is very simple and flexible, and can be used in a myriad of ways for extracting features from documents.

A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things:

1. A vocabulary of known words.

2. A measure of the presence of known words.

It is called a "*bag*" of words, because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.
A very common feature extraction procedure for sentences and documents is the bag-of-words approach (BOW). In this approach, we look at the histogram of the words within the text, i.e. considering each word count as a feature.

— Page 69, Neural Network Methods in Natural Language Processing, 2017.

The intuition is that documents are similar if they have similar content. Further, that from the content alone we can learn something about the meaning of the document. The bag-of-words can be as simple or complex as you like. The complexity comes both in deciding how to design the vocabulary of known words (or tokens) and how to score the presence of known words.

We will take a closer look at both of these concerns.

Example of the Bag-of-Words Model:

Let's make the bag-of-words model concrete with a worked example.

Step 1: Collect Data

Below is a snippet of the first few lines of text from the book "**A Tale of Two Cities**" by Charles Dickens, taken from Project Gutenberg.

It was the best of times,
it was the worst of times,
it was the age of wisdom,
it was the age of foolishness,

For this small example, let's treat each line as a separate "document" and the 4 lines as our entire corpus of documents.

Step 2: Design the Vocabulary

Now we can make a list of all of the words in our model vocabulary.

The unique words here (ignoring case and punctuation) are:

- "it"
- "was"
- "the"
- "best"
- "of"
- "times"
- "worst"
- "age"
- "wisdom"
- "foolishness"

That is a vocabulary of 10 words from a corpus containing 24 words.

Step 3: Create Document Vectors

The next step is to score the words in each document.

The objective is to turn each document of free text into a vector that we can use as input or output for a machine learning model.

Because we know the vocabulary has 10 words, we can use a fixed-length document representation of 10, with one position in the vector to score each word.

The simplest scoring method is to mark the presence of words as a Boolean value, 0 for absent, 1 for present.

Using the arbitrary ordering of words listed above in our vocabulary, we can step through the first document ("*It was the best of times*") and convert it into a binary vector.
The scoring of the document would look as follows:

- "it" = 1
- "was" = 1
- "the" = 1
- "best" = 1
- "of" = 1
- "times" = 1
- "worst" = 0
- "age" = 0
- "wisdom" = 0
- "foolishness" = 0

As a binary vector, this would look as follows:

1. [1, 1, 1, 1, 1, 1, 0, 0, 0, 0]

The other three documents would look as follows:

1. "it was the worst of times" = [1, 1, 1, 0, 1, 1, 1, 0, 0, 0]
2. "it was the age of wisdom" = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0]
3. "it was the age of foolishness" = [1, 1, 1, 0, 1, 0, 0, 1, 0, 1]

All ordering of the words is nominally discarded and we have a consistent way of extracting features from any document in our corpus, ready for use in modelling. New documents that overlap with the vocabulary of known words, but may contain words outside of the vocabulary, can still be encoded, where only the occurrence of known words are scored and unknown words are ignored.

We can see how this might naturally scale to large vocabularies and larger documents.

Managing Vocabulary

As the vocabulary size increases, so does the vector representation of documents. In the previous example, the length of the document vector is equal to the number of known words.

We can imagine that for a very large corpus, such as thousands of books, that the length of the vector might be thousands or millions of positions. Further, each document may contain very few of the known words in the vocabulary.

This results in a vector with lots of zero scores, called a sparse vector or sparse representation.

Sparse vectors require more memory and computational resources when modelling and the vast number of positions or dimensions can make the modelling process very challenging for traditional algorithms.

As such, there is pressure to decrease the size of the vocabulary when using a bag-of-words model.

There are simple text cleaning techniques that can be used as a first step, such as:

- Ignoring case
- Ignoring punctuation
- Ignoring frequent words that don't contain much information, called stop words, like "a," "of," etc.
- Fixing misspelled words.
- Reducing words to their stem (e.g. "play" from "playing") using stemming algorithms.

A more sophisticated approach is to create a vocabulary of grouped words. This both changes the scope of the vocabulary and allows the bag-of-words to capture a little bit more meaning from the document.

In this approach, each word or token is called a "gram". Creating a vocabulary of two-word pairs is, in turn, called a bigram model. Again, only the bigrams that appear in the corpus are modelled, not all possible bigrams.

An N-gram is an N-token sequence of words: a 2-gram (more commonly called a bigram) is a two-word sequence of words like "please turn", "turn your", or "your homework", and a 3-gram (more commonly called a trigram) is a three-word sequence of words like "please turn your", or "turn your homework".

— Page 85, Speech and Language Processing, 2009.

For example, the bigrams in the first line of text in the previous section: "It was the best of times" are as follows:

- "it was"
- "was the"
- "the best"
- "best of"
- "of times"

A vocabulary then tracks triplets of words is called a trigram model and the general approach is called the n-gram model, where n refers to the number of grouped words.

Often a simple bigram approach is better than a 1-gram bag-of-words model for tasks like documentation classification.

a bag-of-bigrams representation is much more powerful than bag-of-words, and in many cases proves very hard to beat.

— Page 75, Neural Network Methods in Natural Language Processing, 2017.

Scoring Words

Once a vocabulary has been chosen, the occurrence of words in example documents needs to be scored. In the worked example, we have already seen one very simple approach to scoring: a binary scoring of the presence or absence of words.

Some additional simple scoring methods include:

- **Counts**. Count the number of times each word appears in a document.
- **Frequencies**. Calculate the frequency that each word appears in a document out of all the words in the document.

Word Hashing

We know from computer science that a hash function is a bit of math that maps data to a fixed size set of numbers.
For example, we use them in hash tables when programming where perhaps names are converted to numbers for fast lookup.

We can use a hash representation of known words in our vocabulary. This addresses the problem of having a very large vocabulary for a large text corpus because we can choose the size of the hash space, which is in turn the size of the vector representation of the document.

Words are hashed deterministically to the same integer index in the target hash space. A binary score or count can then be used to score the word.

This is called the "*hash trick*" or "*feature hashing* ".

The challenge is to choose a hash space to accommodate the chosen vocabulary size to minimize the probability of collisions and trade-off sparsity.

TF-IDF

A problem with scoring word frequency is that highly frequent words start to dominate in the document (e.g. larger score), but may not contain as much "informational content" to the model as rarer but perhaps domain specific words.

One approach is to rescale the frequency of words by how often they appear in all documents, so that the scores for frequent words like "the" that are also frequent across all documents are penalized.

This approach to scoring is called Term Frequency – Inverse Document Frequency, or TF-IDF for short, where:

- **Term Frequency**: is a scoring of the frequency of the word in the current document.
- **Inverse Document Frequency**: is a scoring of how rare the word is across documents.

The scores are a weighting where not all words are equally as important or interesting. The scores have the effect of highlighting words that are distinct (contain useful information) in a given document.

Thus the idf of a rare term is high, whereas the idf of a frequent term is likely to be low.

— Page 118, An Introduction to Information Retrieval, 2008.

Limitations of Bag-of-Words

The bag-of-words model is very simple to understand and implement and offers a lot of flexibility for customization on your specific text data. It has been used with great success on prediction problems like language modelling and documentation classification.

Nevertheless, it suffers from some shortcomings, such as:

- **Vocabulary**: The vocabulary requires careful design, most specifically in order to manage the size, which impacts the sparsity of the document representations.

- **Sparsity**: Sparse representations are harder to model both for computational reasons (space and time complexity) and also for information reasons, where the challenge is for the models to harness so little information in such a large representational space.

- **Meaning**: Discarding word order ignores the context, and in turn meaning of words in the document (semantics). Context and meaning can offer a lot to the model, that if modelled could tell the difference between the same words differently arranged ("this is interesting" vs "is this interesting"), synonyms ("old bike" vs "used bike"), and much more.

3.4 Representing the Context Formally:

We represent the document(context) as an m-dimensional vector $\overrightarrow{td}$

$$\overrightarrow{td} = (tf(d, t1), \ldots, tf(d, tm))$$

where tf (d, t) denotes the frequency of the term $t \in T$ in document $d \in D$.

3.5 Pre-processing:

Since our input set contains some contexts. We cannot use them to our method directly without cleaning them. So, before going into the method we need to do some pre-processing of the data, i.e. pre-processing of the contexts. First of all, we have to do data cleaning for the contexts. Data cleaning means we have to remove the symbols like comma (,), full stop (.), double-quotes (" "), single-quotes(' '), hyphen(-), exclamatory sign(!) etc. And also we have to lemmatize each of the words of each of the contexts.

Example: Consider a context from a parallel corpus:

Before Pre-processing: 'John was sick, so he went to the doctor and the doctor said, "Alas!" and gave him a tablet.'

After Pre-processing: **'**John was sick so he went to the doctor and the doctor said Alas and gave him a tablet'

3.6 Context Representation & TF-IDF Weighting:

A document is represented by a set of keywords/terms extracted from the document. For our clustering algorithms, documents are represented using vector space model. In this model, each document, d is considered to be a vector, d, in the term-space (set of document "words"). Each document is represented by the (tf) vector:

$$dtf = (tf1, tf2, \ldots\ldots, tfn)$$

where tfi is the frequency of the i'th term in the document.

A term-document matrix can be encoded as a collection of n documents and m terms. An entry in the matrix corresponds to the "weight" of a term in the document; zero means the term has no significance in the document or simply doesn't exist in the document. The whole document collection can therefore be seen as a m x n feature matrix A (with m as the number of documents) where the element aij represents the frequency of occurrence of feature j in document i.

This way of representing the document is called term-frequency method. The most popular term weighting is the Inverse document frequency, where the term frequency is weighed with respect to the total number of times the term appears in the corpus. There is an extension of the designated the term frequency inverse document frequency (tf-idf).

The formulation of tf-idf is given as:

$$Wij = TFi,j * \log(\frac{N}{Dfi})$$

Where, Wij is the weight if the term i in the document j,

N is the total number of documents in the corpus,

tfi,j = number of occurrences of term i in document j,

dfi = the number of documents containing the term i.

In our method we have used the module "Scikit-learn" of python to find the Tf-Idf matrix. Whose columns will give us the context vectors to proceed further.

3.7. K-means:

This algorithm was proposed in the year 1957 by Stuart Lloyd. K-means is the method of partitioned cluster analysis. K means clustering algorithm is an effective algorithm to extract a given number of clusters of patterns from a training set. Once done, the cluster locations can be used to classify patterns into distinct classes.

The k-means clustering algorithm is known to be efficient in clustering large data sets. This clustering algorithm is one of the simplest and the best known unsupervised learning algorithms that solve the well-known clustering problem. The K-Means algorithm aims to partition a set of objects, based on their attributes/features, into k clusters, where k is a predefined or user-defined constant. The main idea is to define k centroids, one for each cluster. The centroid of a cluster is formed in such a way that it is closely related to all objects in that cluster.

k-means Algorithm involves clustering the given data into k groups based on the distance between the observation points and the cluster centroids. Clusters can be initialized randomly by any point from the observation data. For best results, while choosing these random points, choose points as far away from each other and the chosen points. Different values of k should

be tested before choosing the best fit. After the clusters have been initialized, points are assigned a cluster based on the closeness of that point to the cluster.

After all points have been assigned a cluster, the cluster centroids are updated and the points are reassigned to their new clusters. k needs to be chosen effectively. We have chosen k = 2 & k = 3 for our experiments.

Example:

The data set has three dimension and the cluster has two points: X= (x1, x2, x3) and Y=(y1,y2,y3).

Then the centroid Z becomes Z = (z1, z2, z3)

Where, z1=(x1+y1)/2 and z2=(x2+y2)/2 and z3=(x3+y3)/2.

In our proposed methodology we have used 'Scikit-learn', 'numpy' etc modules of python to implement the K-means algorithm.

3.8. Algorithmic steps for K-means clustering:

Let,

X = {x1, x2, x3,….., xn} be the set of data points and

V = {v1, v2, ……., vc} be the set of centers.

1) Randomly select 'c' cluster centers.

2) Calculate the distance between each data point and cluster centers.

3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.

4) Recalculate the new cluster center using: where, 'ci' represents the number of data points in i'th cluster.

$$Vi = (\frac{1}{Ci}) \sum_{j=1}^{Ci} Xi$$

5) Recalculate the distance between each data point and new obtained cluster center.

6) If no data point was reassigned then stop, otherwise repeat from step 3.

3.9. DATASET:

Preparation:

Three kinds of corpora were used in our work viz. a monolingual Bengali corpus, a monolingual English corpus and a Bengali–English parallel corpus. The Bengali and English corpora were obtained in the form of XML files or text files filled with noisy data. In order to use them for our proposed method, it was necessary to convert them into text files devoid of noise. The Bengali corpus was a collection of 1270 text files. All of these files were merged together and unnecessary information such as XML tags were stripped out. The English corpus on the other hand was a collection of 182 XML files which were merged together and converted to a clean text file containing only useful data.

Resources:

- **Bengali Corpus:** In order to train the Word2Vec model in Bengali the *TDIL*[1] (Technology Development for Indian Languages) corpus was used. The corpus holds knowledge spread over a total of 85 areas including commerce, mass media, translation, natural sciences etc. Table 1gives some statistics on the corpus.

- **English Corpus:** We used the *BNC*[2] (British National Corpus) – Baby Edition for training the Word2Vec model in English. This corpus covers information from 4 fields' viz. academia, newspapers, fiction and conversations between native British English speakers. Table 1 reports statistics on the corpus.

[1] http://www.isical.ac.in/~lru/downloadCorpus.html
[2] http://ota.ox.ac.uk/desc/2553

- **Parallel Corpus:** The *EILMT* (English to Indian Language Machine Translation) English−Bengali Tourism corpus[3] was used for obtaining Bengali and corresponding English contexts of a particular polysemous Bengali word. The subject matter of this corpus, as the name suggests, essentially hovers around tourism related topics such as descriptions of popular tourist destinations, the culture and civilization of those places etc. Table 1 reports a few statistics on the corpus.

  Here, in our experiment 15 polysemous Bengali words were chosen as target words from the parallel corpus. A maximum of 3 senses and 9 sentences (contexts) per word were considered for our experiments. However, most of the chosen target words had contexts describing only 2 different senses. In total, our dataset consisted of 110 sentences (contexts) for the 15 words selected for the task.

TABLE – 1

CORPUS STATISTICS

Corpus		Number of Sentences	Number of Words	Vocabulary Size
British National		333,045	4,000,000	203,367
TDIL		635,000	5,000,000	193,879
EILMT	***ENGLISH***	22,992	573,209	31,048
	BENGALI	22,992	488,462	61,398

[3] http://tdil-dc.in/index.php?option=com_download&task=showresourceDetails&toolid=1419&lang=en

Setup of Dataset:

After cleaning the noisy data, we made a csv file which contains the cleaned data. That file will contain 4 columns. The first column will contain the Bengali polysemous words, the second column will contain the Bengali context or sentence corresponding to the Bengali polysemous target word. The third column will contain the English contexts corresponding to the targeted Bengali word collected from the parallel corpus. And the last column will contain the sense id of the word within the Bengali or English context.

In the below table we have given an example of the dataset of a polysemous Bengali word 'চাল', which generally means rice in English. But, in Bengali this word has two different meaning, one of the meaning is Rice and the other meaning is the roof of a house. So, we need to find that which contexts belongs to which meaning in reality. To identify that and to compare the performance of our results we have manually assigned the senses of each of the words used in each of the contexts.

Here, we have marked the sense-id' s of that word as 1 and 2. 1 denotes that the meaning of that polysemous target word in that particular context is 'Rice' and 2 indicates that the meaning of that polysemous target word in that particular context is 'Roof'. And we have used 8 instances of this Bengali word 'চাল' for our method in which 3 of them belong to the sense-id 1 and 5 of them belong to the sense-id 2.

Next, we have considered each contexts as a single document and after saving them in text files we have used that files as our input of the program code.

The 'Table-2' will give us a glimpse of the data set we have prepared for the proposed method.

TABLE - 2

Words	Bengali sentences	English sentences	Sense id
চাল	চাল দেওয়া কুঁড়েঘরের গ্রামের মধ্যে দিয়ে যাবার সময়ে গ্রামবাসীদের বেঁচে থাকার ধরন আবিষ্কার করুন এবং রাজস্থানের মুগ্ধ করা ঢিলা উপভোগ করুন	explore the style of living of the villagers as you pass by thatched roofed hamlets and enjoy the fascinating rajasthan deserts cape	1
চাল	রুচিশীলভাবে খড়ি গুঁড়ো দিয়ে রঙ করা ওই খড়ের চাল দেওয়া মাটির ঘরগুলির একটিতে বাস করুন আর সম্পূর্ণ আধুনিক সুখসুবিধাবিহীন এক জীবনের অভিজ্ঞতা লাভ করুন	stay in one of those mud houses with thatched roofs painted elegantly with chalk powder and experience a life completely devoid of modern amenities	1
চাল	জলপান একটি মিষ্টি খাবার বিভিন্ন প্রকারের চাল ও ক্রিম দিয়ে তৈরি করা হয়	jalpan is a sweet dish made of different kinds of rice with cream milk or yoghurt with jaggery	2
চাল	প্রাকৃত পক্ষে আসমিয় খাবারে খারটি আদেশমূলক মেনু ছোট চাল দিয়ে খাওয়া হয়	in a typical assamese meal khar is a mandatory appetizer eaten with a little rice	2
চাল	যার সম্বন্ধে আমরা কথা বলছি সেটি বিশেষ সাদা চাল মূলত পাওয়া যায় লোহিত জেলায় যার নিজস্ব স্বাদ ও গন্ধ আছে	what we are talking about is the special white rice particular to lohit district which has its own special flavor and fragrance	2
চাল	চাল এবং অড়হর ডাল বাটা রোদে জারানোর জন্য ফেলে রাখা হয় এবং তারপর কড়া করে ভাজা হয়	the rice and urad dal batter is left to ferment in the sun and then deep fried	2
চাল	সাদা চুনাপাথর এবং আঠালো চাল থেকে ইট তৈরী করা হয়েছিল আর আঠালো চাল এবং ডিমের সাদা অংশ থেকে সিমেন্ট তৈরী করে হয়েছিল	the bricks were made from white lime and glutinous rice while the cement is made from glutinous rice and egg whites	2
চাল	আপনি এখনো ওই চাল দেওয়া কুটির বিশিষ্ট গ্রামগুলি দেখতে পাবেন যেগুলি বোগ্মালোর অসমতল বালুরাশিতে নির্জনভাবে বিরাজমান একটি ছোট্ট চুনকাম করা চ্যাপেল ও জঞ্জালের মাঝে চলে বেড়ানো শূকরের দলে পরিপূর্ণ	you can still see those thatched hamlets nestling solitarily on the undulating sands of bogmalo complete with a tiny white washed chapel and gangs of hogs wandering amidst the rubbish	1

3.10. Experimental results:

It was observed while constructing the experimental dataset that most words had on an average two or three sense usages in the parallel corpus due to the small size of the parallel corpus. This turned out to be a limitation of the resource (parallel corpus) used owing to the fact that several senses of highly polysemous words (as defined by the Bengali WordNet) could not be explored in our experiments.

On the other hand, words like ডাল (*dal*) although having 2 senses in the Bengali WordNet, i.e. *pulses* and *branch of a tree*, saw only 1 sense being represented in the corpus (*pulses*), while another corpus defined sense was discovered i.e. when the word (ডাল) is used to refer to the '*dal lake*'[4].

This in turn leads us to establish the idea that induction does not always lead to the discovery of senses of a word which are defined by the WordNet. In fact, novel (new) sense detection is a primary goal when any induction algorithm is being formulated.

In the next page we have shown the experimental results of our K-Means clustering method in a tabular form (TABLE-3).
The first column contains the polysemous Bengali words which are our target words whose senses need to be clustered.
The second column contains the number of instances that means number of contexts belong to that particular polysemous word.
The 3rd column contains the senses corresponding to each of the targeted polysemous word.

In the 4th, 5th and 6th column we have calculated the Precision, Recall and F-score of the outcome corresponding to the target word respectively.

[4] A famous tourist spot in Kashmir

TABLE - 3

			K-Means		
Word	**Instances**	**Sense**	**Precision**	**Recall**	**F Score**
চাল	8	*Rice*	0.75	0.6	0.666
		Roof	0.5	0.666	0.57
ডাল	7	*Lentil*	0.6	0.75	0.666
		Lake	0.5	0.33	0.397
রাস্তা	4	*Road*	1	1	1
অর্থ	8	*Money*	0.5	0.666	0.57
		Meaning	0.75	0.6	0.666
আচার	9	*Pickle*	1	0.6	0.75
		Ritual	0.66	1	0.8
জাল	8	*Network*	1	0.75	0.857
		Fishing Net	0.8	1	0.888
প্রণালী	6	*Recipe*	0.4	1	0.57
		Strait	1	0.2	0.33
ফল	8	*Result*	1	0.4	0.57
		Fruit	0.5	1	0.666
বোঝা	8	*Burden*	0.2	1	0.33
		Understanding	1	0.4285	0.6
তাল	9	*Rhythm*	0.5	1	0.666
		Palm Tree	1	0.714	0.833
গভীর	8	*Deep Thought*	1	0.4	0.57
		Dense	0.5	1	0.666
মিষ্টি	8	*Sweet Taste*	0.66	0.666	0.67
		Sweet Dish	0.8	0.8	0.8
লক্ষ্য	8	*Aim*	0.33	1	0.496
		Purpose	0	0	0
		Observation	1	0.5	0.666
পাতা	6	*Sole*	0.5	1	0.666
		Page	0.5	0.5	0.5
		Leaf	1	0.66	0.8
গোলা	5	*Barn*	1	1	1
		Cannon Ball	1	0.66	0.795
		Type of Kebab	0.5	1	0.666
		Average	0.70	0.71	0.64

3.11. Analysis of Results:

By looking at the experimental results we can say that, the proposed method gives us the precision as 0.7 i.e. the accuracy is 70% in average, which is quite good for a low resource language like Bengali. The average Recall (R) value is 0.71 and average F-score (F) value of our experimental result is 0.64 which are also quite satisfying.

If we observe further deeply then we can see that, the words with 2 senses gives us better accuracy in result, it gets increased up to the precision 0.72 which is almost 3% better than the overall result. Similarly, if we analyse the results of the words with more than 2 senses i.e. with 3 senses then we got precision as 0.64, which is almost 8.5% lesser than the overall result.

Similarly, if we analyse the results of the words with 2 senses and 3 senses separately for Recall then we can see that, the R value remains unchanged for the words with 2 senses as well as for the words with 3 senses.

Finally, in case of F-Score we observed that, the F value gets increased by 1% in case of the words with 2 senses. And for the words with 3 senses the F value gets decrease by approximately 2%.

The Table-4 gives us a glimpse of the differences occurred by analyse the result for the words with 2 senses and for the words with 3 senses separately.

TABLE 4

Precision, Recall and F score differences from 3 senses to 2 senses

Measure	Increased	Constant	Declined
Precision	3%	_	_
Recall	_	✓	_
F	1%	_	_

CHAPTER

4

CONCLUSIONS & FUTURE WORK

4.1. Conclusions:

The main motivation behind using translation features was to achieve better clustering. For cases like জাল, গোলা, রাস্তা, আচার etc., the *P*, *R* & *F* values were so much high. This fact indicates how well the proposed sense clustering method performed, in case of a low resource language like Bengali.

A comparison of the average *P*, *R* and *F* scores between the words with 2 senses and for the words with 3 senses showed that all the three values i.e. Precision, recall and *F-Score* gets increased or remain unchanged for the words with 2 senses. So, we conclude that, our proposed method performs more well for the polysemous Bengali words with 2 senses. This also showcases the fact that the proposed model is capable of handling the sense induction task in Bengali quite well without having to rely on support languages.

Presence of highly polysemous support language words in the context might introduce ambiguity in the augmented vector, resulting in degraded performance; however, this phenomenon was not observed in our experiments.

We did clustering for various English and Bengali datasets collected from standard corpuses. The quality of clustering depends on the similarity measure chosen, the clustering algorithm used and the construction of the tf-idf matrix. There is no similarity measure which gave best results on every data set but in general. We also observed that there is a difference between the performance of these measures in Bengali and English. Though for Bengali, our proposed methodology gave satisfactory results as a standard algorithm should perform.

Though we have a limitation that in our proposed algorithm we need to mention the number of cluster i.e. the value of k manually, which is also a binding of the traditional k-Means clustering method. But beyond that our proposed algorithm performs up to the mark.

4.2. Future Work:

In future we can further work on other low resource languages to compare the performances in each case using this k-means clustering technique. Also we can implement other clustering algorithms for Bengali word clustering and can find the best suitable algorithm for this job. Further we can try this on multi-lingual context vectors. Also we can try this method on the polysemous words with more than 3 senses. Also, identifying appropriate number of clusters for a particular data set through some kind of evaluation instead of empirical analysis could also be a possible extension. In addition to labelling the clusters using the top terms, methods like Wikipedia cluster labelling.

References:

- *Eneko Agirre and Aitor Soroa. 2007. Semeval-2007 task 02: Evaluating word sense induction and discrimination systems. In Proceedings of the 4th International Workshop on Semantic Evaluations, pages 7–12. Association for Computational Linguistics.*

- *David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In Proceedings of the 33rd annual meeting on Association for Computational Linguistics, pages 189–196. Association for Computational Linguistics.*

- *Amruta Purandare and Ted Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. In Proceedings of the Conference on Computational Natural Language Learning, pages 41– 48. Boston.*

- *Hinrich Schutze. 1998. Automatic word sense discrimination. ¨ Computational linguistics, 24(1):97–123.*

- *Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 613–619. ACM.*

- *Jean Veronis. 2004. Hyperlex: lexical cartography for information retrieval. ´ Computer Speech & Language, 18(3):223–252.*

- *Samuel Brody and Mirella Lapata. 2009. Bayesian word sense induction. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, pages 103–111. Association for Computational Linguistics.*

- *Jey Han Lau, Paul Cook, Diana McCarthy, David Newman, and Timothy Baldwin. 2012. Word sense induction for novel sense detection. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 591–601. Association for Computational Linguistics.*

- *David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. the Journal of machine Learning research, 3:993–1022.*

- *Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. 2006. Hierarchical dirichlet processes. Journal of the american statistical association, 101(476).*

- *Do Kook Choe and Eugene Charniak. 2013. Naive bayes word sense induction. In EMNLP, pages 1433–1437.*

- *Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 109–117. Association for Computational Linguistics.*

- *Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1, pages 873–882. Association for Computational Linguistics.*

- *George A Miller. 1995. Wordnet: a lexical database for english. Communications of the ACM, 38(11):39–41.*

- *Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: the 90% solution. In Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers, pages 57–60. Association for Computational Linguistics.*

- *J.R. Firth (1957). A synopsis of linguistic theory 1930-1955. In Studies in Linguistic Analysis, pp. 1-32. Oxford: Philological Society. Reprinted in F.R. Palmer (ed.), Selected Papers of J.R. Firth 1952-1959, London: Longman (1968).*

- *D. Widdows and B. Dorow. A Graph Model for Unsupervised Lexical Acquisition. 19th International Conference on Computational Linguistics, Taipei, August 2002, pages 1093-1099.*

- *E. Agirre, D. Martínez, OL de Lacalle, and A. Soroa. 2006. Two graph-based algorithms for state-of-the-art WSD. In Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP '06). Association for Computational Linguistics, Stroudsburg, PA, USA, 585-593.*

- *S. Bordag. Word sense induction: Triplet-based clustering and automatic evaluation. In 11th Conference of the European Chapter of the Association for Computational Linguistics, 2006.*

- *M. Sahlgren, The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces. PhD thesis, Department of Linguistics, Stockholm University, 2006.*

- *Y. Bengio, R.Ducharme, P. Vincent, and C.Janvin. A neural probabilistic language model. J. Mach. Learn. Res., 3:1137–1155, March 2003. ISSN 1532-4435.*

- *D. Hope and B. Keller. MaxMax: A Graph- Based Soft Clustering Algorithm Applied to Word Sense Induction, pages 368–381. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-37247-6. doi: 10.1007/978-3-642-37247-6_30.*

- *T.K Landauer, P.W. Foltz, and D.Laham. An introduction to latent semantic analysis. Discourse Processes,25(2-3):259–284,1998.doi:10.1080/01638539809545028.*

- *T.K. Landauer and S. T. Dumais. A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. Psychological Review, 104(2):211–240, 1997.*

- *M. Apidianaki. Translation-oriented word sense induction based on parallel corpora. In Proceedings ofLanguage Resources and Evaluation Conference (LREC), 2008.*

- *L. Albano, D. Beneventano, and S. Bergamaschi. Word sense induction with multilingual features representation. In Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 02, WI-IAT '14, pages 343–349, Washington, DC, USA, 2014. IEEE Computer Society. ISBN 978-1-4799-4143-8. doi: 10.1109/WI-IAT.2014.117.*