Topic Modelling and Sentiment Analysis with the Bangla Language: A Deep Learning Approach Combined with the Latent Dirichlet Allocation

by

Mustakim Al Helal

A thesis
Submitted to the Faculty of Graduate Studies and Research
In Partial Fulfillment of the Requirements
For the Degree of

Master of Science
in
Computer Science

University of Regina
Regina, Saskatchewan

September, 2018

UNIVERSITY OF REGINA

FACULTY OF GRADUATE STUDIES AND RESEARCH

SUPERVISORY AND EXAMINING COMMITTEE

Mustakim Al Helal, candidate for the degree of Master of Science in Computer Science**,** has presented a thesis titled, ***Topic Modelling and Sentiment Analysis with the Bangla Language: A Deep Learning Approach Combined with the Latent Dirichlet Allocation,*** in an oral examination held on August 28, 2018. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner:	*Dr. Yllias Chali, University of Lethbridge
Supervisor:	Dr. Malek Mouhoub, Department of Computer Science
Committee Member:	Dr. Samira Sadaoui, Department of Computer Science
Committee Member:	Dr. David Gerhard, Department of Computer Science
Chair of Defense:	Dr. Maria Velez-Caicedo, Department of Geology

*via SKYPE

Abstract

In this thesis, the Bangla language topic modelling and sentiment analysis has been researched. It has two contributions lining up together. In this regard, we have proposed different models for both the topic modelling and the sentiment analysis task. Many research exist for both of these works but they do not address the Bangla language. Topic modelling is a powerful technique for unsupervised analysis of large document collections. There are various efficient topic modelling techniques available for the English language as it is one of the most spoken languages in the whole world, but not for the other spoken languages. Bangla being the seventh most spoken native language in the world by population, it needs automation in different aspects. This thesis deals with finding the core topics of the Bangla news corpus and classifying news with a similarity measure which is one of the contributions. This is the first ever tool for Bangla topic modelling. The document models are built using LDA (Latent Dirichlet Allocation) with Bigram. Over the recent years, people in Bangladesh are heavily getting involved in social media with Bangla texts. Among this involvement, people post their opinion about products or businesses across different social sites and Facebook is the most weighted one. We have collected data from the Facebook Bangla comments and applied a state of the art algorithm to extract the sentiments which is another contribution. Our proposed system will demonstrate an efficient sentiment analysis. We have performed a comparison analysis with the existing sentiment analysis system in Bangla. However it is not straightforward to extract sentiments from the Bengali language due to its complex grammatical structure. A deep learning based method was applied to train the model and understand the underlying sentiment. The main idea is confined to the word level and character level encoding and in order to see the differences in terms of the model performance. So, we will explore different algorithms and techniques for topic modelling and sentiment analysis for the Bangla language.

Acknowledgements

My first debt of gratitude goes to my supervisor Dr. Malek Mouhoub, who encouraged me to follow this path and provided me with constant support during my M.Sc. studies. I would like to express my sincere thanks for his valuable guidance, financial assistance and constant encouragement. His enthusiasm, patience and diverse knowledge helped and enlightened me on many occasions. The amount of freedom in thinking I received from him helped me overcome the difficulties with my research. I feel proud to be his research student and cannot imagine better supervisor.

I acknowledge the Faculty of Graduate Studies and Research for providing me with the financial means, in the form of scholarships which contributed towards my tuition fees. I thank UR international office for helping me engage myself in different community work.

I also thank Dr. Samira Sadaoui and Dr. David Gerhard, my thesis committee members who read my thesis and provided invaluable suggestions and useful comments for improvement.

I would like to take this opportunity to thank every member of the Department of Computer Science who has helped me throughout my studies.

Last but not least, I would like to extend my deepest gratitude to my beloved parents for their unconditional love and support throughout my entire life. It would not have been possible for me to come to Canada and achieve a prestigious scholarship without my parent's never ending support. I dedicate my hard earned M.Sc. degree to my beloved parents.

POST DEFENCE ACKNOWLEDGEMENT

My thanks go to Dr. Yllias Chali of the University of Lethbridge for being the external examiner for my M.Sc thesis and for providing me with his invaluable comments and suggestions.

Contents

List of Figures

List of Tables

Abbreviations

NLP	**N**atural **L**anguage **P**rocessing
LDA	**L**atent **D**irichlet **A**llocation
LSI	**L**atent **S**emantic **I**ndexing
HDP	**H**ierarchical **D**irichlet **P**rocess
SVD	**S**ingular **V**alue **D**ecomposition
RNN	**R**ecurrent **N**eural **N**etwork
LSTM	**L**ong **S**hort **T**erm **M**emory
GRU	**G**ated **R**ecurrent **U**nit

Chapter 1

Introduction

In this chapter, the motivation and the problem which will be solved is discussed along with some general discussion to get started with the basic ideas of Natural Language Processing (NLP). The solution of the discussed problem has been defined. This chapter also talks about the main contribution for this thesis. The organization of the thesis will be discussed in section 1.3

1.1 Problem Statement and Motivations

Natural Language Processing (NLP) has been a demanding field of research under the Artificial Intelligence domain for quite a long time. Recently, NLP is placing as one of the top trending research fields due to having a large amount of available text data. Data science and machine learning are collectively working as the next big thing in Computer Science. Moreover, the World Wide Web (www) has become significant over the last 25 years which is fueling the data science and machine learning sectors. This presence made available of Bangla texts for classification and other classic natural language processing tasks. However, there has been no attempt of topic modelling with Bangla language so far in terms of the Latent Dirichlet Allocation (LDA) which is a powerful topic modelling algorithm. This worked as a motivation for this thesis

Another motivation for this work was social media like Facebook. Bangla is the native language in Bangladesh. Dhaka, the capital of Bangladesh ranked as the 2nd highest city in the world in terms of number of Facebook users [4] . Facebook adopted the Bangla language and Bangla speakers are more comfortable using Bangla for Facebook comments and status after the Bangla typing application Avro was first introduced in 2003. The goal for this thesis was to make a sentiment analysis model with Bangla in a different approach that can beat the existing results. There are some systems that performed sentiment analysis in Bangla but no research has been done to analyse sentiments with a Recurrent Neural Network (RNN).

Topic modelling and sentiment analysis with Bangla are both new concepts as far as the NLP field is concerned. With the topic modelling approach, classification of a large document collection can be achieved. However, applying topic modelling algorithms on Bangla language is a challenge since it has a completely different grammatical structure compared to English. Data pre-processing is different for Bangla. This research deals with finding the core topics in a Bangla news corpus which consists of 7,134 news articles from renowned online news portals of Bangladesh. A method is proposed for the classification of the news. Perplexity [5] is discussed. Coherence of different topics are also explored. A comparison between LDA, LSA, HDP and LSI are discussed in the subsequent sections of this thesis.

The second contribution of the thesis is a Sentient analysis. A collection of Bangla comments from Facebook was used as a dataset for this research. If someone wants to know which comment is positive, which one is negative and which of the comments are neutral then there has been no tool to identify this with the Bangla language. Previously, research has been done in English with a Naive Bayes classifier or from the lexical resources perspective. However, the problem with this is it's performance was limited to boolean models and not tested with multinominal features in [6]. An approach has been proposed in this thesis to successfully achieve accuracy better than the existing sentiment analysis model in Bangla. Sentiment analysis in Bangla has a wide range of future possibilities with deep learning. Deep learning methods are being applied successfully to natural language processing problems and achieving an acceptable accuracy. Recurrent Neural Network, a variation of Neural Network (NN) is used for processing sequential data. The classification task of the sentiments are a step by step process which will be discussed in detail in their respective chapters.

Long Short Term Memory (LSTM) is used in the proposed method. It is a popular and successful technique to handle long term dependency problems. There is a variant of LSTM which is known as Gated Recurrent Unit (GRU). GRU is also used in this research.

Research in NLP with the Bangla language is a necessity after the recent flourishing of the language in the world wide web. However, there is still no research to preprocess news articles dataset for topic modelling with the Bangla language which is also a motivation for this research. A Comparison of LDA, HDP and LSI algorithms is also a new step for Bangla which will benefit current and future researchers to understand the performance of these popular topic modelling algorithms in terms of Bangla language.

1.2 Proposed Solution and Contributions

This thesis provides several insights regarding the contribution with the Bangla language for topic modelling and sentiment analysis. Firstly, an analysis of the topic modelling techniques with graphical illustrations, tables and comparisons are provided. A deep explanation of a recurrent neural network is also discussed.

Secondly, a novel idea is proposed for sentiment analysis on the Bangla language. Both the word and the character level representations are achieved for training the model. LSTM and GRU are used after the character level encoding. Character level representation has never been used in Bangla language for sentiment analysis.

Thirdly, the major drawback of the word level representation for deep neural network model has been addressed in this research. The model performs with error if any word that comes in the test set has not been trained. So the character level encoding plays a vital role in understanding the sequence of the sentence and hence predicting the underlying sentiment.

Fourthly, an optimized model with the right hyper parameters for the sentiment analysis is proposed. In the proposed model, an embedding layer of 67 units and output layer with three units are used after understanding with trial and error that this is the optimal number of layers for this particular task to be solved.

Fifth, a model with the LDA algorithm is developed to train with Bangla language and predict the news class from the Bangla news corpus. LSA, LSI and HDP algorithms are also compared. Coherence of the topics is achieved with graphical illustrations. Individual news topics are collected to make it meaningful for a human reader.

Sixth, finding the related news groups with a similarity measure is implemented. This is a classification task with the LDA algorithm. The cosine similarity value for the news is computed. Coherence based topic optimization is employed to achieve the right number of topics for the corpus.

Finally, different experiments and analysis demonstrate the correctness and efficiency of the proposed models.

1.3 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 discusses related works and explains the background knowledge that is necessary for understanding the rest of the thesis work. Basic concepts of the Recurrent Neural Network (RNN) are introduced. Then the LSTM algorithm is discussed in detail. Further, various concepts of topic modelling are explained. Text processing requirements are also discussed in this chapter. Some examples are used for the easy understanding of the reader.

Chapter 3 introduces the first part of the contribution. Here, the basics of the sentiment are discussed. The RNN algorithm is illustrated with it mathematical explanation. How LSTM works for a sequential text stream is also discussed. The proposed model is discussed with graphical representations. The model architecture is also explained with a step by step process. The experiments for the proposed model are explained followed by the results with the graphical illustrations. Two different approaches are explained for the sentiment extraction. The result section focuses on the comparison of the two approaches.

Chapter 4 is about the second contribution of the thesis. It is a topic-modelling based algorithm that works with the Bangla language. The most famous algorithm for topic modelling, Latent Dirichlet Allocation (LDA) is discussed here. Data collection and preprocessing are also explained for Bangla language. The Bigram data model is first discussed. This chapter discusses the various experiments with the Bangla news corpus with respect to topic modelling. News data topic extraction experiment with both individual news and with the complete corpus is explained. LSI, HDP and a tweaked version of LDA is compared in the experimental section to check which one has better performance in terms of topic coherence. Finally, topic-wise categorization of the news is performed.

Chapter 5 then discusses the conclusions and future work regarding how these models can further be enhanced for more research with the Bangla language. This chapter proposes the future applications of these models across different fields.

Chapter 2

Literature Review and Background

2.1 Literature Study

Due to complex grammatical structure and fewer resources in Bangla, the language has not been well researched in the field of Natural Language Processing (NLP). Most of the Sentiment Analysis (SA) work has been carried out on the English language. For this research work, some related works are studied that deal with SA. Many papers regarding topic modelling are studied too for the second contribution towards Bangla.

The most relevant research work carried out is with romanized Bangla in [7]. In this paper, the researchers collected a dataset from different social media sources. They applied a deep neural network approach. More precisely, the LSTM algorithm was used to train the model and achieved an accuracy of 78% with categorical cross entropy. However, in this research two types of loss functions were discussed to check the performance of the model. Apart from the categorical cross entropy, binary cross entropy is also used. In this research a consolidated dataset was generated from different social media and was preprocessed to be readily used for research with SA by applying a deep neural network. The model that the experiments were done with is based on RNN or more specifically LSTM. Any deep neural network uses multiple layers as a basic architecture. Each layer receives the inputs from the previous layer and passes on the outputs to the next layer after performing its respective job. Accordingly, in this research the model has different layers. First layer of the model is the embedding layer. In this layer word to vector representation is done. The second layer in their model is the LSTM layer with dimensions of 128 units. Then comes the fully connected third layer with different activation for the classification purpose.

In the experiment, the last layer has 1, 2 or 3 neurons. Different number of neurons were employed depending on what classification task is being attempted. When only positive and negative sentiments were being classified, the final fully connected layer was configured with 1 or 2 neurons only. However, the dataset has another type of entity, annotated as ambiguous (neither positive nor negative). For those entities to be classified, 3 neurons were used. For the task of classification of only the positive and negative sentiment, binary cross entropy was used. Categorical cross entropy was employed in which case 3 neurons were involved.

The results achieved from this model scored a 78% in terms of accuracy with the categorical cross entropy loss and ambiguous entities removed. However, it scored much lower with the categorical cross entropy, modified text and ambiguous converted to 2 and the accuracy for this case was only 55%. The paper did not address the character level encoding of the dataset. However, Hassan et al. in [7] addressed a practical perspective of having romanized Bangla i.e writing Bangla with English letters which is an important fact for people in Bangladesh writing in different social medias.

The other work with SA was done with Bangla data collected from micro-blog posts [1]. The main goal for this research was to obtain a dataset from bangla micro-blogs and train the model on the opinionated data to identify the polarity as positive or negative. The dataset for this work consists of a collection of twitter posts in Bangla. 1300 instances were collected and 1000 instances were picked for training the model. Some English part along with the Bangla texts were also comprised into the dataset. Moreover, all the hashtags were included after preprocessing as they can potentially carry meaningful sentiments. Along with the general pre-processing which is done for any NLP research, Parts of Speech (POS) tagging was also performed on the dataset. POS tagging is basically used as a strong subjectivity of the respective sentence. Initially, the researchers in [1], manually developed a list of Bangla words grouping by nouns, pronouns, adjectives, verbs etc. Each word is then translated to English in order to achieve a score indicating its polarity as positive or negative.

Table 2.1: List of some examples words after POS tagging with positive and negative polarity [1]

POS	**Positive-Emotion Words**	**Negative-Emotion Words**
NOUN	Love, Happiness	Sorrow, Trouble
VERB	Enjoy, Attract	Damage, Hate
ADJ	Beautiful, Enthusiastic	Bad, Useless
ADV	Well	Unfortunately

However, one problem that might need to be addressed with a POS tagging is that when a Bangla word is translated to its English counterpart the meaning might get changed. A random Bangla word in a sentence might have a different context. So, even a positive word may sarcastically carry a negative sentiment which is a very challenging task to train as far as NLP is concerned. In this research in [1], a semi supervised technique has been applied due to the unavailability of a large amount of labeled data. Self training bootstrapping method is used. First, a small chunk of the dataset was trained on the basis of the frequency of positive and negative words. After the model acquires the knowledge based on this little data, another chunk of the data is applied to gain insight on the polarity. This process is continued until the whole dataset is labelled. Once the dataset is ready, feature extraction begins. Bigrams are generated. Stemming, the process of converting each word into its root is performed with a rule-based technique. Emoticons in the twitter posts were also considered as an important part. An emoticon polarity dictionary developed by Leebecker et al. was used to unwrap the meaning of the emoticons. Bangla and English Lexicons were used to identify the polarity.

Two state-of-the-art classifiers named SVM and MaxEnt were used in [1]. Finally, the classifiers results were compared in different feature sets. A comparison of F-measure with the different features are demonstrated in Table 2.2

From the results it was observed that SVM outperforms MaxEnt. The highest accuracy was obtained as much as 93% with SVM using unigrams and emoticons as features [1]. So, the result was satisfactory with a small amount of data and scarce resource for Bangla language itself. However, the neutral statements a.k.a ambiguous sentences were not considered in this research which is address in this thesis. The proposed model in this research addresses the neutral statements that do not convey any positive or negative sentiments.

Table 2.2: Comparison of F-measure for both the classifiers with different features [1]

Features	Positive F-Measure	Negative F-Measure	Positive F-Measure	Negative F-Measure
unigram	0.65	0.68	0.67	0.69
unigram+stemming	0.69	0.70	0.67	0.70
bigram	0.69	0.42	0.034	0.67
unigram+bigram	0.65	0.69	0.67	0.70
unigram+negation	0.66	0.68	0.67	0.69
emoticon	0.89	0.87	0.74	0.83
unigram+emoticon	0.93	0.93	0.83	0.85
unigram+negation+emoticon	0.92	0.92	0.83	0.85
lexicon(Bangla)	0.71	0.34	0.71	0.38
lexicon(English+Bangla)	0.53	0.73	0.57	0.74
unigram+lexicon(English+Bangla)	0.71	0.71	0.67	0.69
unigram+lexicon(English+Bangla)+ POS	0.71	0.47	0.71	0.46
all	0.89	0.85	0.83	0.85

A hybrid method was proposed in [8] to identify the sentiments. The authors first determined whether or not a sentence is subjective and they designed a model from the mixture of various POS features collected from the phrase level similarity and they used syntactic model to perform sentiment analysis. By doing so, they achieved overall 63.02% recall rate using SVM on the news data. The research involves a dependency graph to identify related nodes in a sentence. Here, nodes represent each word. It helps to determine intra-chunk polarity relationship. Chunk level information was used as a feature in this proposed method since it helps reducing ambiguity of the polarity. SentiWordNet [9] has been used as an important feature to get the idea of the words carrying potential sentimental meanings. For the purpose of this research, collected words from SentiWordNet are given a score according to their corresponding polarity.

After feature selection and processing, SVM is applied to train the classifier. It reaches an accuracy of 70.04% and a recall of 63.02%. Although, this research involves efficient features in their model, it does not come up with any significant result in terms of the accuracy. One reason for this could be the size of the dataset. A relatively small dataset is used for this research. Some similar research addressed data from social media [10]. Here, two different approaches were discussed to identify the polarity of a Facebook post. The first approach is a Naive Bayes classifier and the second one involves lexical resources. However, an observation was made on the results that focused on the fact that the Lexical resource works better for a domain specific sentiment analysis.

Character level Convolutional Neural Network (CNN) was explored in [11]. Several large scale datasets were used by Xiang Zhang et al. in [11] to demonstrate the efficiency of CNN in achieving the state of the art results. Different models such as TF-IDF and Bag of Words were compared in terms of performance. In this research, one dimensional temporal CNN was used. A key module for the CNN was max pooling to train deeper models. Characters were used in encoded version. One hot encoding was performed on size m of the input language. In total, 70 characters were used consisting of 26 English letters, 33 other characters and 10 digits and also the new line character. The input feature length was selected as 1024 which is the same as we did in our contribution. Logically, 1024 characters could include most of the texts of interest for this kind of corpus. An interesting part of the model is that it has data augmentation ability. Since, in many cases it has been studied that data augmentation makes the model capable of controlling the generalization error or also known as out-of-sample error .

A comparison between different models was carried out. First, the Bag of Words model is developed. The bag contained only 50,000 most frequent words from the training set. Bag of n-grams and its TF-IDF was then used. For the later one, 500,000 most frequent words were selected from the training subset for each dataset. Finally, Bag of means model was used. In this one, K-means clustering was used for classification. All the words appearing more than 5 times in the training period were considered.

Deep learning methods were another section of the paper. Two different types of deep learning models were used and compared. One is Word based CNN and the other one is LSTM. This research has concrete connection with the work has been done in this thesis. Two different models were compared for deep learning. One is word based CNN and the other one is simply LSTM which is the same as this thesis. For the wordbased CNN both pre-trained word2vec embedding and lookup table technique were used. In both cases the embedding size was chosen as 300 to make a fair comparison. However the techniques followed by Zhang et al. in [11] for the LSTM model is different. They combine a couple of LSTM models and generated a feature vector and then perform a multinominal logistic regression on this vector. **Vanilla architecture** which is a variation of LSTM was also used. They also used gradient clipping. The most important discussion from this research is that the character level CNN can be efficient without the need for words. This strongly indicates that languages can also be considered as a signal of any other kind. However, how well these deep learning models work depends on many parameters such as the data size, number of layers, choice of alphabets etc.

Some other works have also been studied for this research. These include neural network research topics, representation with back propagating errors and other fundamental dependency learning with the LSTM algorithm which helped carry forward this research. In [12] a learning procedure was proposed which is known as back propagation. This procedure is specific to learning in terms of neural networks. This back propagation regularly adjusts the weights of the connections in the network to minimize the difference between the actual output vector and the desired output vector. Due to this adjustment the understanding of the hidden units in the neural network becomes much clearer and they come to represent important features of the task domain. This gives an understanding as to how the weights of the CNN model should be adjusted and relate this thesis to the exploration of the neural network architecture. In [13] a research was carried out to understand the underlying concepts of the LSTM algorithm itself which also contributed towards the ideas to use this classic algorithm for the sentiment analysis task in this thesis. LSTM is a novel gradient based efficient algorithm that works well where remembering some values for an arbitrary time interval is necessary. The architecture of LSTM consists of gated units and neurons which was briefly described in [13]. Different datasets were explored to learn how many runs the LSTM takes to appropriately learn languages. Also, the learning rates were compared. A counterpart research was made by Yoshua Bengio et al. in [14] where it has been experimented that long term dependency is difficult to learn with a gradient descent approach and the algorithm becomes inefficient as the time duration of holding the memory increases. In [15] more LSTM experiments were done with gated recurrent neural network. Dropout techniques were studied in [16]. It sheds light on the techniques on how to avoid overfitting. The subsequent sections will discuss more on dropouts and also on Adam [17] for optimization. Some more research will be cited for the sentiment analysis work as and when needed in chapter 3.

For the second part of the thesis, Topic Modelling, a number of research papers and related works were studied. Most of the works are done in English due to its availability of a structured dataset. However, no paper cites the topic modelling applied on the Bangla language specifically developing models with the Latent Dirichlet Allocation (LDA) algorithm. So, that is where this thesis focuses on. Although, many experiments have been conducted over the years to extract the topics from an English corpus. This is an unsupervised learning from the classic LDA algorithm. The first paper in [2] is basically a survey on Topic Modelling. Two different aspects of topic modelling were discussed in [2]. One is about the different classic topic modelling algorithms like LDA, Latent Semantic Analysis (LSA) and the second aspect is the topic trend. The first one is experimented in this thesis. The LDA algorithm is the base for the proposed model in this thesis. The reason for LDA gaining popularity among the NLP community is that LDA can capture both the document and word perspectives from a corpus. LDA mimics the way a document is written. Given the topics it generates a document that best fits those topics and hence can understand the correlation between documents and topics. So this paper talks and discusses about each of the topic modelling algorithm and critically analyses in terms of their performances. LDA perceives a data to be a mixture of different topics and each topic containing some words. Each of the word within a topic has a probability value to belong to that topic. On the other hand LSA tries to achieve a semantic content from the vectorized representation of a document. Another goal of the LSA is to measure the similarity of documents and then picking up the most relevant word that matches the document. Apparently, these can be done with the LDA as well but the inner structure of the LSA vary from the LDA. LDA collects the related texts and divide by the number of documents. In the next section of this chapter a brief discussion will be available to understand how exactly these models work.

Discussion on topic evolution model sheds light on the pros and cons of this model in [2]. It is important to understand the topic trend over time and there are several different techniques on topic evolution finding. Word co-occurrence and time are taken into consideration to understand the topic trend. Another model is known as dynamic topic models. It generates topic distribution at different epochs. Most of the time a sequential corpus is used to understand the topic trends. It assumes that at each time slice topic distributions will change and the same topic can have different probabilities over different time slots depending on the dataset.

In this paper some comparisons of different topic modelling and topic evolution models were compared in terms of their characteristics and also their pros and cons.

Table 2.3: Comparison of characteristics of topic modelling methods [2]

Name of the methods	**Characteristics**
Latent Semantic Analysis (LSA)	1) LSA can get from the topic if there are any synonyms 2) Not robust statistical background
Probabilistic Latent Semantic Analysis (PLSA)	1) It can generate each word from a single topic;even though various words in one document may be generated from different topics 2) PLSA handles polysemy
Latent Dirichlet Allocation (LDA)	1) Need to manually remove stop words 2) It is found that the LDA can not make the representation of relationships among topics
Correlated Topic Model (CTM)	1) using of logistic normal distribution to create relations among topics 2) Allows the occurrences of words in other topics and topic words

However, CTM and PLSA will not be used in the proposed model for this thesis. But it is a good survey paper to understand the baseline models with their characteristics. Different topic evolution methods were also studied in this paper.

Blei et al. has the most fundamental contribution on LDA in his paper in [18]. This paper specifically talks about how the LDA actually works. Empirical experiments were conducted to demonstrate how the LDA algorithm can provide state of the art results across topic modelling, text classification and collaborative filtering. LDA assumes that there are K underlying latent topics by which the whole corpus can be generated. Each topic consists of all the relevant words those can belong together with a probability value. A document is generated by taking a mixture of topics and then the topics consist of word distributions. In the experiments two different corpora were tested to explore how the LDA can generate topics and their corresponding words. Perplexity was calculated of the model to test the performance in general. Perplexity is an estimate on how well a probability model

Table 2.4: Different topic evolution models and their main characteristics [2]

Models	**Main characteristics of models**
Modeling topic evolution by continuous-time model	1) "Topics over time: A non markov continuous time model of topic trends"
Modeling topic evolution by discretizing time	1) "Dynamic Topic Models" 2) "Multiscale topic tomography" 3) "A non parametric approach to pairwise dynamic topic correlation detection"
Modeling topic evolution by using citation relationship as well as discretizing time	1) "Detecting topic evolution in scientific literature: How can citations help" 2) "The web of topics: Discovering the topology of topic evolution in a corpus"

works for an unseen data. The lower the perplexity the better the prediction is. For both of the dataset it was seen in the experiment that the LDA perplexity reduces with number of topics in [18]. So, the accuracy results are validated. Among the other topic modelling algorithms LDA seems to work better and giving reasonably good results. So in this research a comparison of different models and their topic coherence will be discussed in chapter 4. Results for text classification and collaborative filtering were also combined and LDA worked better for these applications as well. So, the conclusion is that LDA being a generative model can be used as a module to different applications and extend it in different directions.

An insightful study of LDA has been done in [3]. The main goal of this paper was to classify the documents with similarity and understand the topics achieved from the LDA. So, two different types of datasets were used. The authors used the Wikipedia and Twitter datasets. Data pre-processing was applied. Tokenization, followed by stop word removal was done. Afterwards, a dictionary was made before finally training the model with the LDA. After the data was pre-processed, the two different models were individually trained for the two datasets. Topics were achieved for both the datasets.

Some experiments were carried out with these topics to understand how well they relate to the data. A document was chosen randomly and the topic distribution for that data was illustrated. Some relevant topics with high probabilities were achieved. Topic-word distribution for those topics were then experimented which showed the relevance of the words with the documents. This is a very straightforward and effective research on the LDA to understand the fundamental concepts. Apart from the topic distribution, document similarity measure was also achieved using Jensen-Shannon divergence to calculate the distance. However, in this thesis the cosine similarity is used to perform the similarity and classification tasks since it is a simplified technique in terms of the sparse matrix distance calculation and also it demonstrated effective results. The following table shows how a topic distribution is made for a particular document.

Table 2.5: Top five topic probabilities for article Light [3]

Topics	Topic47	Topic45	Topic16	Topic24	Topic30
Probabilities	0.05692600	0.45920304	0.28462998	0.01518027	0.01328273

In Table 2.5, the probability values are calculated with high precision in order to distinguish the small fractional changes. Similar research was done by Blei et al. from the application perspective of LDA. Blei proposed the LDA algorithm for topic modelling in 2003. An exploration of the underlying semantic structure of a dataset from the JSTORs archive for scientific journals were carried out in [19]. A methodology was proposed to facilitate efficient browsing of the electronic dataset. The Dynamic LDA was also implemented on this dataset to see the topic trends over a period of time. In this research, efforts were made to understand the topics of Bangladeshi newspapers.

A survey for the LDA and its variants was performed in [20]. Classification task of the LDA was also discussed. Both [19] and [20] discussed about the dynamic LDA to obtain the topic trends over time which is important for news analytics. In Bangla, no previous effort was made to understand the topic trends. The dataset that is developed for this research is organized chronologically. There are over 7000 news articles organized from the oldest to the most recent. This will help to understand the topic trends as an extension work in the future.

Out of few number of researches with the Bangla language for text summarization, a novel approach was proposed in [21]. Although no LDA based methods were discussed in [21], it is still a very insightful research done with the Bangla language. A collection of news articles were used as the dataset. Basically, it proposed a scoring method for understanding the semantic meaning of a document. Two different approaches were proposed for scoring. One is for sentence scoring and the other one is word based scoring method. However, before any scoring applied on the dataset, general preprocessing was done. Tokenization, stemming, stop words removal etc. were performed. For the word level scoring, word frequency, numeric value identification, repeated words distance, cue words count were performed. During the word frequency phase, the number of words are counted across the whole collection. This helps generating the Bag of Words model. Numeric values are considered in this research because they can effectively carry semantic meaning of any text article. These numeric values were included in the vector space representation. Repeated word distance was also calculated. Repeated words can help understand the summary. A word occurring multiple times in an article with a close distance can find the importance of that word which contributes in the summary generation. Repeated word distance further helps in identifying a cue word which may have a clear indication of the summary. For the sentence scoring, different techniques were implemented that includes identifying sentence length, sentence position and uniform sentences. Finally, words and sentences were clustered in three different ways [21]. Sentence ranking based on their similarities, Final gist analysis, Aggregate similarities were performed to combine the results and cluster different types of news together. The performance for this proposed method was evaluated against a human clustered group. So, the system-generated summary of the news was much faster and almost as relevant as the human generated ones.

2.2 Background Knowledge

In this section the required background knowledge will be discussed. It is necessary for the reader to understand the rest of the thesis. This section will briefly discuss the algorithms, concepts and the tools used for the experiments.

2.2.1 Topic Modelling

In the edge of digitization over the last decade, an enormous amount of textual data have been and still are being generated extremely rapidly on the world wide web (www). As soon as data science got fueled with big data, the importance of analyzing textual data emerged. Starting from understanding the customer's reviews across different business fields to understanding the latent meaning of a corpus (a collection of text data) gained high importance. Blei proposed a novel approach to understand the topics which eventually led to the classification of documents, understanding sentiments and opened quite a lot of analysis perspectives for textual data [18]. Topic modelling is basically understanding the topics in a corpus. It is a probabilistic method to explore the topics which is not possible or in some cases time consuming for a naked human eye. As the name suggests, topic modelling is a technique by which topics from a corpus are generated automatically that provides insight on the hidden patterns of the dataset. It is not a rule-based or a regular-expression-based technique on which some generic rules are created to find the importance of the words. Rather, it is more of a statistical and a probabilistic approach. An example of a topic modelling is as follows.

Let assume there is a large amount of data talking about different subjects like health, food, education and traffic. A good topic modelling would bring these topics together with a probabilistic value representing the chances of these words to be in that topic. So, a good topic modelling should generate these topics - **Topic 1:** *Body*, *Breathing*, *Burning*, *Acne*, *Headache*, *Hygiene* etc. **Topic 2:** *Eat*, *Digest*, *Delicious*, *Frozen*, *Taste*, *Spice*. **Topic 3:** *Engineering*, *Science*, *Degree*, *University*, *Job*, *Money*. **Topic 4:** *Accidents*, *Rush*, *Weather*, *Speed*, *Ticket*, *Car*. This is how a topic modelling brings about the hidden semantic structures of a data. Without reading a large number of text data, topics are automatically generated. This helps in classification, sentiment analysis, understanding trends in a business etc. Topic modelling has impact in the field of data science and Natural Language Processing. This thesis has half of its contribution regarding topic modelling with the Bangla language which has never been explored due to the scarcity of dataset as well as other resources on Bangla. Another important field of topic modelling is newspaper industry. Automatic classification plays a vital role in the virtual news world. A huge number of news articles are generated each day. Understanding the topics and trends of news are important to analyze and predict. World renowned news agency "New York Times" is using topic modelling to boost their customers' need and suggesting news genre according to customer's profile. However, Bangla being the 6th most spoken language all over the world it is a necessity to have an automation. Topic model may also work the same way as Netflix in terms of suggestions. It will suggest articles or news or may even be books for digital libraries.

2.2.2 Latent Dirichlet Allocation

There are many different approaches for obtaining topics from a set of text data. Topics can be generated from Term-Frequency and Inverse-Document frequency (TF-IDF) model, non-negative matrix factorization etc. However, Latent Dirichlet Allocation (LDA) is the most popular technique applied on topic modelling. This approach has never been explored on the Bangla language. LDA is basically a probabilistic algorithm that generates topics from a **Bag of Words** model. Before the LDA is discussed, a general overview of **Topic** and **Term** is given as follows:

A topic is a collection of words with different probabilities of occurrence in a document talking about that topic. If there are multiple documents, a topic would consist all the words initially from that collection. After the model is trained the topics will consist of words that are highly relevant.

A term is a word in a topic. Each term belongs to a topic. The LDA algorithm generates a topic-term matrix.

LDA is a matrix factorization technique [22]. At first, it generates a document-term matrix. In the matrix, assume that there are two documents $d1$ and $d2$ and two words $t1$ and $t2$. So the example matrix is as follows:

Table 2.6: An example document-term matrix

	t1	t2
d1	0	1
d2	1	0

In this matrix, the rows represent the documents and the columns represent the terms t1 and t2. 0 and 1 shows if the word belongs to the document or not. Then the LDA generates two more low level matrices representing document-topic and topic-term matrix which will be discussed in detail in chapter 3. Afterwards, a list of all the unique words are made from all the documents. The algorithm goes through each word and adjusts the topic-term matrix with a new assignment. A new topic “K” is assigned to the word “W” with a probability P [22]. How the probability is calculated will be discussed in the related section. The algorithm keeps iterating until a situation is reached when the probabilities do not change considerably which means that the probabilities will have a very small fractional change with further iterations. At this point the algorithm converges and is expected to have the topics extracted.

2.2.3 Latent Semantic Indexing

Latent Semantic Indexing (LSI) also known as Latent Semantic Analysis (LSA) is a widely used topic modelling algorithm apart from the LDA. This topic modelling algorithm tries to understand the underlying semantic meaning of words in a document. Topic modelling is not a straightforward process for a machine since there is not just a single word having the same concept or meaning. It would have been much easier to model topics if there would be each word mapping to a single concept. But in reality, there could be many words that address the same meaning and people use different words under the same context. Just like in English, Bangla has many words to describe a single context. LSA has efficient yet simple techniques to relate words. For example the word **Bank**, if used with **Mortgage**, **Loan**, **Credit** etc then it is defining a financial institution. But if the word **Bank** is used with **Fish**, **Tide** etc. then we know that it is defining a river bank.

LSA is a variation of the LDA that comes with the goal to find a relevant document by searching words. It becomes complicated to find documents by just searching words since words can have multidimensional contexts and this is not simple for a machine to learn. However, LSA came up with simple ideas to make a concept space by keeping track of both words as well as the documents. It generates a mapping between words and documents. The word vs concept relationship can essentially create noise due to the randomness of word selection. Some dramatic simplifications are added in LSA in order to solve this problem. The techniques are as follows:

1. As LDA, it uses the “Bag of Words” representation too. For this representation, the order of the words is not as important as the frequency.
2. Word patterns are considered and they are used to connect words with similar meanings. For example, “bank”, “loan”, “credit” might be used to define a financial institution.
3. To make the solutions tractable it is assumed that each word has only one meaning.

Unlike LDA, LSA/LSI is not a probabilistic model that calculates the dirichlet prior over the latent topics. LSA is rather a matrix decomposition technique on the document-term matrix. A comparison of the LDA and LSA was performed in the dissertation in [23]. In practice there is a trade off while using both of these algorithms. LSA has a much faster training time while it lacks in accuracy in comparison with LDA [23]. LSA uses Singular Value Decomposition (SVD) matrix on the training corpus. The SVD used in LSA consists of three matrices [24]. SVD represents a matrix T which is a product as the following:

$$T = KSD^T$$

Here K is a topic-keyword matrix, S is a topic-topic matrix, D^T is a document-topic matrix where K and D are orthogonal matrices and S is a diagonal matrix. The resultant matrix after multiplication, represents a document by keyword matrix. So, in T each document is tagged with a single keyword that describes the document. Matrix S is a diagonal matrix that is used to track the relevance between the terms. So, how closely the different terms are related can be identified. The SVD of the matrix is consequently followed up by dimensionality reduction to reduce the noise in the latent space that provides an efficient word relationship to better understand the semantics of the corpus.

SVD, is basically a second order Eigenvector technique and it can consider the fact of having relevance in different terms within a corpus. LSA is an application of SVD widely used for text processing in particular. Whereas, LDA takes on a different concept about modelling topics in an unsupervised learning situation. It works on the assumption and estimation of a Dirichlet prior in a Bayesian Network. So, it is more of a probabilistic model. These priors are the important part to start with the calculations because these values are needed to initially generate a probabilistic solution which gets better with time. The priors are then plugged in with the same principle as a Bayesian network. LDA however, tries to determine an appropriate prior rather than using the only one that fits all the 2nd order correlation like the LSA. In terms of accuracy LDA is better but that again depends on the data and the goal. The dataset that will be used in chapter 4 is not huge. That is why variations of the LDA will be experimented to observe the performances.

2.2.4 Hierarchical Dirichlet Process

Hierarchical Dirichlet Process (HDP) is another variation of the LDA that will be used to compare with. HDP does not need to know the number of topics apriori like the LDA and this is the reason for which HDP is used for many applications of NLP. As far as the LDA is concerned, the number of topics to be extracted from a corpus has to be declared beforehand since the LDA algorithm can not predict the right number of topics it should extract. However, for the HDP algorithm the number of topics can be learned from the data and it can generate topics accordingly. Like LDA, the HDP also uses a dirichlet allocation to capture the uncertainty in number of topics. A common base distribution is selected for identifying a set of possible topics for the corpus. After that, the base distribution is used to sample each document to find the possible topics of the corpus. Like LDA, HDP also considers the number of mixed words for each topic. However, it does not make the number of topics fixed for each document. Rather, the number of topics is also generated by a dirichlet allocation that consequently makes the topic a random variable. The name **Hierarchical** comes from the idea to add one more level on top of the probabilistic model. In a topic model, each document is organized with words and each document forms from the Bag of Words indexing. Let us assume that the indexing is j = 1,..., J and each indexed group has data items from x_{j_1} to x_{j_m}.

The HDP model is controlled by a base distribution H from which the apriori over data items are selected. Some other parameters control the number of clusters from the unsupervised corpus. The base distribution is denoted as G_0. The probability of the J^{th} cluster is given with the following formalization:

$$G_j|G_0 \approx DP(\alpha, G_0)$$

Here G_j is the probability for the j^{th} topic. DP means the dirichlet process. G_0 is further formalized according to the following:

$$G_0 \approx DP(\alpha, H)$$

Here, H is the base distribution that has been discussed above. α is the concentration parameter. Now, as the concentration parameter α and the base distribution

H are achieved, each data item can be associated with a latent parameter θ as follows:

$$\theta_{j_i}|G_j \approx G_j$$
$$x_{j_i}|\theta_{j_i} \approx F(\theta_{j_i})$$

The above two equations describe how each parameter and each data item are associated with their dependency variable. The first equation shows how each parameter for each data item is dependent on the prior distribution given by G_j. The second equation shows how each data item is associated with individual parameter and has a distribution of $F(\theta_{j_i})$. This is known as the HDP mixture model. This is hierarchically linked to a set of Dirichlet process. AS mentioned in [25] to understand the HDP model, we need to understand that it implements the clustering and all the clusters are further shared across groups of data. The following equation can be considered:

$$G_0 = \sum_{k=1}^{\infty} \pi_{0_k} \delta_{\theta_k^*}$$

It is assumed that there are infinite number of atoms supported by the base distribution H. The atoms are denoted by δ and have masses denoted by π. The masses need to sum up to one since G_0 is a probability measure. However as there is no free lunch, HDP models have their limitations too. One major limitation with the HDP is that it does not consider the relationship among the documents. It calculates the mixture of all the documents and achieves the topics from the mixture model. This can be inefficient with a document collection where inter document relationship is important. Also, during the assignment of topics to the documents, probabilities of the neighborhood should be more than the ones far apart. Since the HDP model mix all the documents and then assign the topics, it can not keep track of the topics that are assigned to the relevant documents.

2.2.5 Singular Value Decomposition

In linear algebra Singular Value Decomposition (SVD), means the factorization of a real or a complex matrix [26]. In a more formal way, a singular value decomposition of a real or a complex matrix having dimension of m by n and the matrix known as M can be factorized with the form $U\sum V^*$ where U is a m by n real unitary matrix. $\sum$ is a m by n rectangular diagonal matrix with non negative real numbers on the diagonal and V is a m by n complex unitary matrix. Let us have an example of SVD. Assume a 4 by 5 matrix M as follows:

$$M = \begin{Vmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \end{Vmatrix}$$

So the SVD of this matrix according to the formula $U\sum V^*$ is as follows:

$$U = \begin{Vmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{Vmatrix}$$

$$\Sigma = \begin{Vmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{Vmatrix}$$

$$V^* = \begin{Vmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & -\sqrt{0.2} \end{Vmatrix}$$

The same thing happens on a LSA algorithm. After we achieve term-document matrix, SVD is applied on that matrix to infer the topics. What the SVD does is it reduces the dimensionality of the matrix extracted from document-terms matrix. Dimensionality reduction is implemented by ignoring the small singular values in the diagonal matrix. Regardless of the number of singular values set to zero, the resulting matrix which is the matrix achieved after the calculation retains its original shape. It does not drop any columns or rows from the original matrix. Apparently, the effects of the dimensionality reduction only reflects in the decomposed version.

Consider for example a large matrix consisting of many rows and columns. If the matrix is of rank 1 that means the columns and rows will only span a one-dimensional space. From this matrix, only non zero singular matrix will be obtained after the decomposition takes place. So, instead of storing this large matrix, two vectors and one real number can be stored. This results in a reduction of one order in magnitude. So, SVD can be considered as an efficient storage mechanism or can be extended as literal dimensionality reduction if the reduced matrix is fed into the algorithm. This however, depends on what kind of predictions are aimed. Let us now focus on some formalization of the SVD. Assume that a document collection which is a D by n matrix where D is the document size and n is the number of documents. Here the columns are document BOW vectors. When the LSI is applied as SVD to the document it only keeps the largest $d << m = min(D, n)$ minimum singular values that is, let U'_{D*_d} be the first d columns of U, S'_{D*_d} be the submatrix of S, $V'_N{*_d}$ be the first d columns of V. Then the reduced version of the matrix will be as follows:

$$U'_D *_d S'_D *_d V'_N{*_d}$$

This is the rank-d approximation in a least square sense. However, the question that arises is what happens to the data instances not in the training set but will occur in the test set. The answer to the question is, for the new test document it will be added to the existing data points and compute the SVD on the n+1 documents. This is computationally expensive. If the document, however, coincide with an existing document it will have the same new coordinates [27].

2.2.6 Evaluation of Topics

Evaluation of the topic modelling is an important task as any other machine learning model. With the evaluation process it can be determined how well the model is working. Topic coherence is a scientific method that measures the human interpretability of a topic model. Traditionally perplexity has been used often but it was found that perplexity does not correlate with human annotations at times [28]. On the other hand topic coherence is a topic evaluation method with higher guarantee on human interpretability [29]. So, it can be used to compare different topic models along with individual topics. How well correlated the topics are can be understood with the topic coherence score. Topic coherence measures the similarity between each two pair of terms under a topic. Assume that a topic $T1$

has N words from w_1 to w_n. So the topic coherence computes the sum with the following formula:

$$Coherence = \sum i < jScore(w_i, w_j)$$

Topic coherence computes the sum of pairwise score of the words $w_1 ... w_n$ that belong to the topic. It computes all the possible pairs of the words inside a topic. In this research topic coherence is used to evaluate the topics and check how well they correlate human annotations. However, human interpretation is still needed to tag the topics.

From a mathematical point of view, coherence is the log likelihood of a word w_i given the other word w_j. That is how it understands the correlation between a pair of words. The Umass measure of coherence introduced by Mimno et al. in [30] is used in this thesis. A pairwise score function is used as follows:

$$\text{SCORE}UMass(w_i, w_j) = logD\frac{(w_i, w_j) + 1}{D(w_i)}$$

Here, $D(w_i, w_j)$ is the count of documents containing both the words w_i and w_j. $D(w_i)$ is the count of the documents containing only w_i.

2.2.7 Recurrent Neural Network

Recurrent Neural Network (RNN) is a variation of Neural Network that is widely used in the natural language processing due to its capability to “remember” previous calculations along the process. The idea behind a RNN is to involve all the previous calculations to find out the output for the current layer. Unlike a traditional neural network where all the inputs are independent of each other, in a RNN each input carries forward to the next layer and an output at any layer is dependent on the previous layers. The name **Recurrent** comes from the idea of performing the same task for every element in a sequence. However, the RNN can also be considered as a memory saving tool. It is a neural network that can remember the result of past calculations and use it to infer the prediction for the next element in the sequence. Theoretically, it can make use of information in arbitrarily long sequences but in practice, they are limited to looking at only a couple of steps back [31]. A typical RNN is illustrated in Figure 2.1.

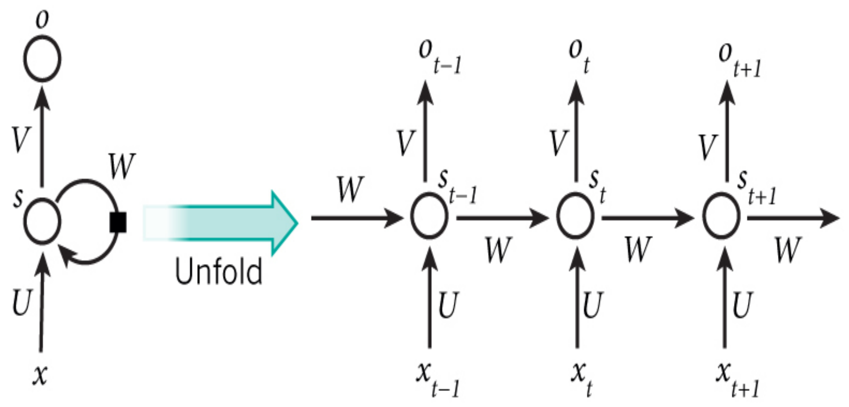


FIGURE 2.1: A typical RNN [31]

The diagram shows how the unfolding of the network takes place. Unfolding means the expansion of the network for the full sequence. For example if there is a sequence of 5 words, the network would have 5 layers after the expansion or unfolding takes place. The formulas that control the calculations are as follows:

- X_t is the input to a layer t. That means X_n could be a vector corresponding to the n^{th} word in the sequence that the RNN is working on.
- S_t is a hidden value at layer t. It can be considered as the "memory" of the network which has all the previous calculations involved. S_t is calculated based on the previous S value that is $S_{t_{-1}}$ and the current input value $S_t = f(UX_t + WS_{t_{-1}})$ where W is a weight applied to the input. The function f is a nonlinear function usually tanh or Relu.
- O_t is the output at step t. It is usually calculated with a Softmax function. $O_t = Softmax(VS_t)$.

RNNs are widely used in predicting the semantic meaning of a sentence and thus can be extensively used in sentiment analysis which is the case in this thesis. A variation of RNN will be used for sentiment analysis with the Bangla language which is the second contribution and covers a full chapter. LSTM is used which is described in the next subsection.

2.2.8 Long Short Term Memory

Long Short Term Memory (LSTM) is a variation of the RNN. LSTM is basically a combination of various RNNs with the objective to memorize certain part of the input sequence. A lot of times it is important to memorize a certain block of the data but not all and that needs to be controlled with some techniques. For example, if there is a need to predict the sales rate of a particular product during the christmas season then it needs to look at the sales data around the month of December of the previous years, considering the data around the other months makes no sense in this case. For this kind of task LSTM is used which remembers certain calculations. However, how much it should remember can be controlled by calculations. Theoretically, RNN can also remember certain part of the previous input but it suffers from two problems. *Vanishing Gradient* and *Exploding Gradient* [32]. To solve this problem, LSTM was later introduced that incorporates a memory unit known as cell into the network. A typical LSTM network is illustrated in Figure 2.2.

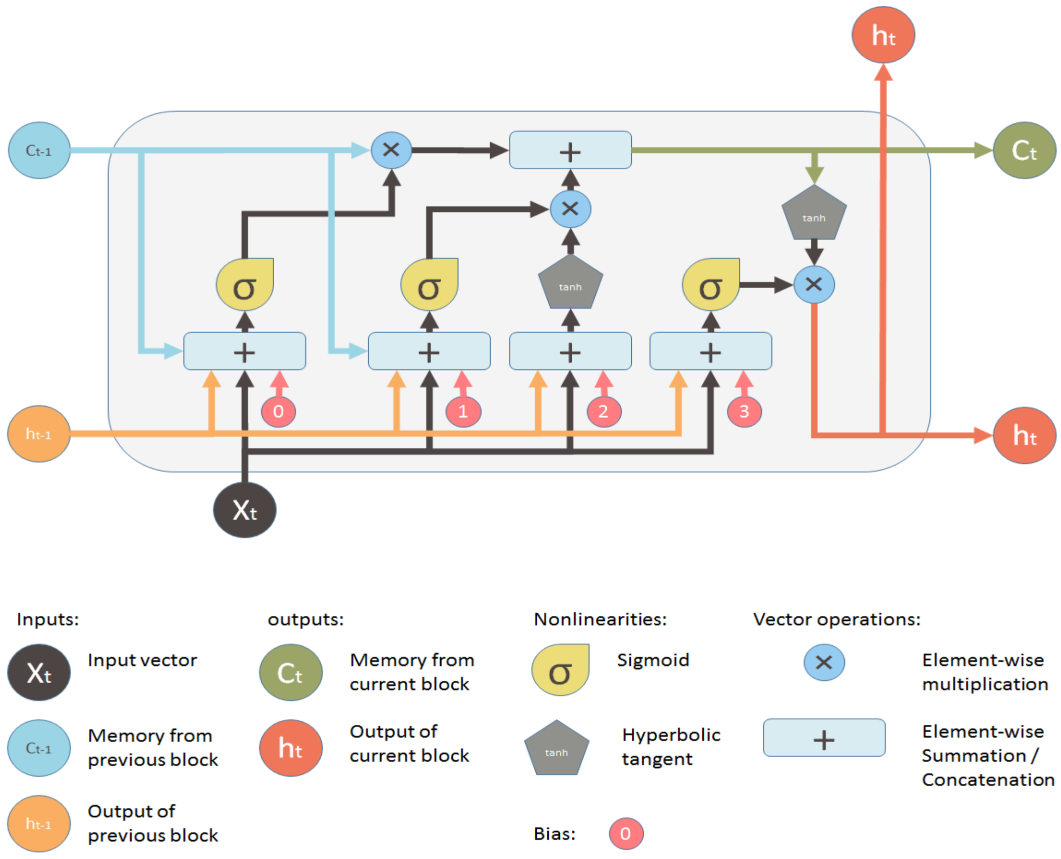

ht
Ct-1
Ct
ht-1
ht
Xt
σ
tanh
0
1
2
3
Inputs:
Xt Input vector
Ct-1 Memory from previous block
ht-1 Output of previous block
outputs:
Ct Memory from current block
ht Output of current block
Nonlinearities:
σ Sigmoid
tanh Hyperbolic tangent
Vector operations:
× Element-wise multiplication
+ Element-wise Summation / Concatenation
Bias: 0

FIGURE 2.2: A typical LSTM [32]

Initially LSTM might look like a complicated neural network but as it is explained step by step the diagram will start to make sense. For the time being, let us only focus on the input and output of the LSTM unit. Three inputs are taken by this network. X_t is the input taken for the current time step. $H_{t_{-1}}$ is the output from the previous LSTM unit that merges with the current input. C_{t-1} is the memory coming in from the previous time step and a needed portion of this memory is then combined with the current input and the previous output. Then the next output, next memory is formed and the layers continue like this.

In LSTM, two more important things to consider is the "Forget" gate and "Memory" gate denoted by the $\times$ and $+$ symbol respectively in diagram in Figure 2.2. These two parts of the LSTM unit acts like a water valve. If assumed that the memory that passes along the LSTM chain units is the water then this water needs to be controlled with the forget valve. What the forget valve does is that it controls how much of the previous information to pass through the next unit. Mathematically, this is an element wise vector multiplication. If the previous memory C_{t-1} is multiplied with a vector that is close to zero that means forgetting the most of the previous memory. Next, is the memory valve. With this, new input comes into the flow and will merge with the old memory. This is known as the memory unit. Mathematically, this is a element wise summation. After this operation, the old memory $C_{t_{-1}}$ is transformed to C_t. The mathematical calculation that takes place inside those valves will be discussed in the chapter where a model for the sentiment analysis is proposed. How the output is shaped along the path will also be discussed.

2.2.9 Gated Recurrent Unit

In this section the Gated Recurrent Unit (GRU) is explained which is a special kind of neural network and is used in this thesis for the sentiment analysis. This neural network was introduced by Cho et al. in 2014 in [33] with the objective to solve the vanishing gradient problem which is pretty obvious with a normal neural network. GRU is basically a variation of the LSTM with slight changes in the architecture of the gates and the functionalities. What makes the GRU special from the standard RNNs is that they use update gates and reset gates to control the flow of information along the layers from one unit to the other. It can handle the long term dependency problem well enough and is trained to keep information for relatively longer amount of time compared to LSTM. To understand the GRU better a single unit example is taken and the maths behind the screen is explained.

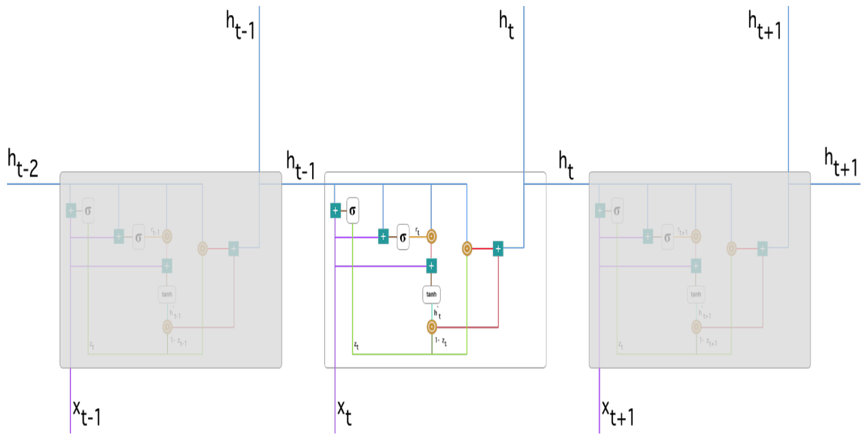


Figure 2.3: A recurrent neural network with a gated recurrent unit [32]

As can be seen from Figure in 2.3, this is a recurrent neural network with a gated recurrent unit. A more detailed diagram is shown in Figure 2.4.

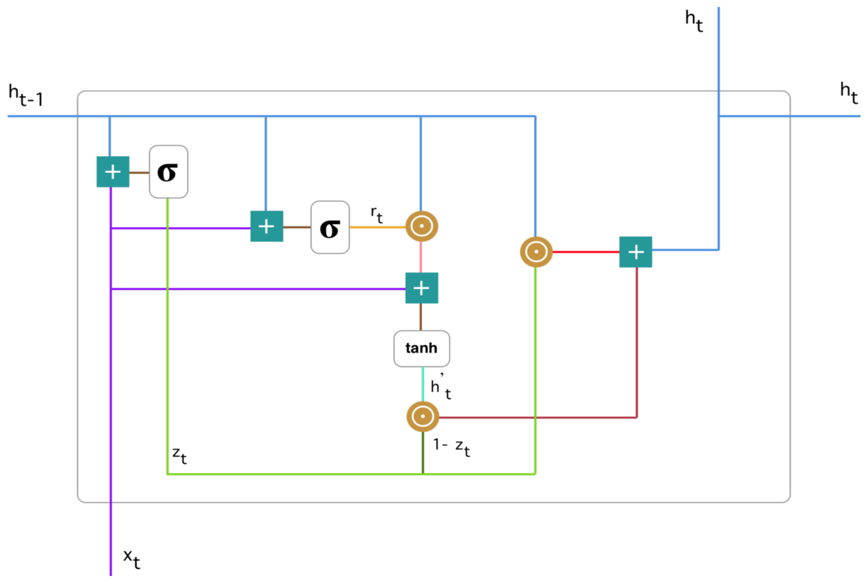


Figure 2.4: The GRU unit [32]

Let us first get familiarized with the symbols in the diagram in Figure 2.4. The $+$ symbol means the addition operation, σ symbol signifies the Sigmoid function that takes place inside the GRU. Then the $\odot$ notation signifies "hadamard product" meaning element wise vector multiplication. The *tanh* symbol means the *tanh* function which takes place inside the model. The work flow of the GRU is explained in a step by step process.

2.2.9.1 Update Gate

The update gate starts with the following equation:

$$z_t = \sigma \left\{ W^{(z)} x_t + U^{(z)} h_{t_{-1}} \right\}$$

In this formula x_t is the input that is coming into the unit at time stamp t and $h_{t_{-1}}$ holds the information from the previous unit. What the update gate does is it goes through a merge technique. However, when merging the current input with the previous output an important question that arises is, how much of each information to take along and pass through the next phase. To control the amount of information, weights are applied to them. As can be seen from the above equation, w^z is the weight applied to the input x_t and U_z is the weight applied to the output of the previous unit $h_{t_{-1}}$. Once these two vectors are simply added which is considered as "merging", they are transformed to a value between 0 and 1 by the *sigmoid* function.

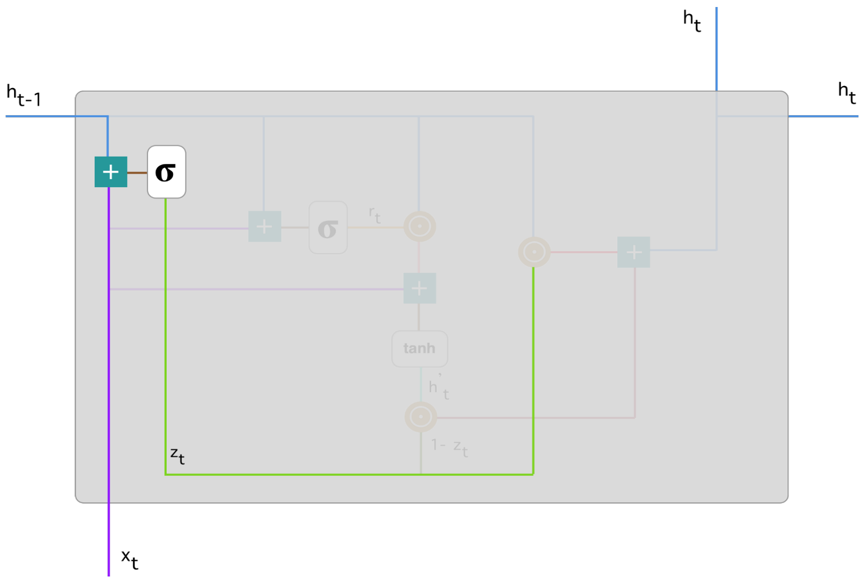


FIGURE 2.5: Diagram showing the *sigmoid* activation for merge [32]

The powerful thing about the GRU is that it can combine the exact amount of information from the previous time stamp. So, the model can even decide to keep all the past information and eliminate the vanishing gradient problem.

2.2.9.2 Reset Gate

Another important part of the GRU is the reset gate. It is essentially used to determine how much of the previous information to forget. Since the information achieved at each unit does not have the same importance to the output, it is seldom needed to have control over the information to forget. The same formula is used as the above for the update gate with different weights.

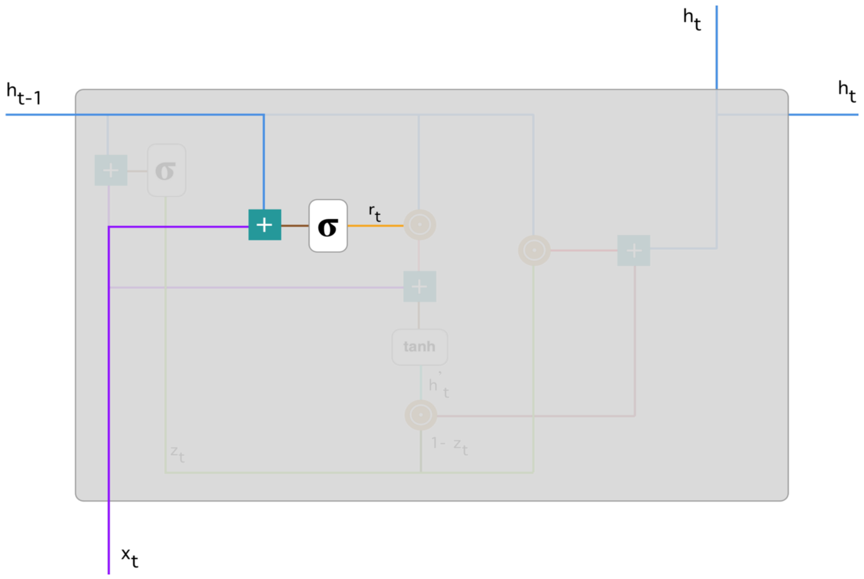


FIGURE 2.6: GRU reset function [32]

In Figure 2.6 $h_{t_{-1}}$ and r_t are added with their corresponding weights to forget the necessary amount of information and compress the added value between 0 and 1 with the *sigmoid* function.

2.2.9.3 Current Memory Content

The memory content is used to hold information for later use along the units towards the final output. The reset gate is used to store the memory information of the past and the calculation is done as follows:

$$h'_t = tanh(Wx_t + r_t.Uh_{t_{-1}})$$

As we can see from the formula, the input x_t is multiplied with its weight W. Next, the element wise vector product of the reset gate's output r_t and previous information $h_{t_{-1}}$ is calculated. This determines what to remove from the previous time step. This is where the whole GRU turns out to be interesting. For the sentiment analysis work let assume a sentence - **The book talks about science** and then there are lot more sentences in one paragraph. The paragraph finally ends with - **I did not quite like the book**. We want to estimate the sentiment from this paragraph. So, as the neural network model approaches the end of the sentence it understands that the sentiment can be understood from the last sentence by wights adjustment. As a result it does not need the information about the other part of the paragraph. So, it washes out the previous information by setting the r_t value almost close to 0 a.k.a a very low information is passed through from the previous sentences. It can only focus on the last sentence to predict the overall sentiment. After that the result from x_t and the vector multiplication are summed together. The non-liniear activation function *tanh* is applied which is reported in the Figure 2.7:

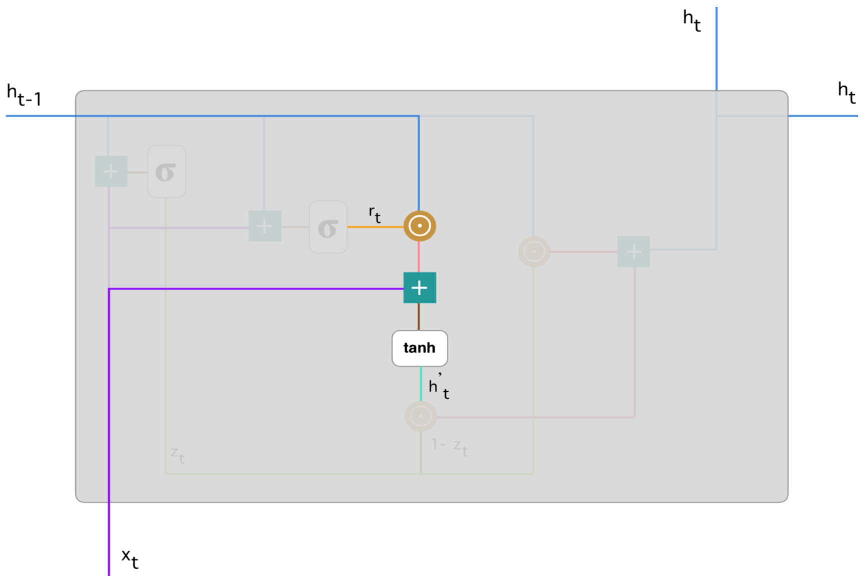


Figure 2.7: Diagram showing GRU tanh function [32]

As can be seen from this diagram the reset gate output r_t and previous information h_{t-1} is vector multiplied and summed up with the input x_t. Finally *tanh* activation function is used. That turns the new information for time-stamp t as h_t.

2.2.9.4 Final memory at Current Time-Stamp

At this final step, the GRU calculates the h_t for the current time-stamp. This is the information that has been achieved so far and it will be passed down the line to the next unit and continues till the output layer. The update gate is used for this purpose. It calculates the necessary information for the current phase h'_t and the previous phase h_{t-1} with the following formula:

$$h_t = z_t * h_{t-1} + (1 - z_t) * h'_t$$

At first, z_t from the update gate is multiplied with the previous phase information h_{t-1}. Second, $1 - z_t$ is multiplied with current phase information h'_t. These multiplications are summed together. With this technique, situation where a sentiment sits at the front of a sentence and is needed to carry along this information to the last part of a big text can be handled. By doing this it can still remember the information received at the beginning. However, when the sentiment changes over the course of the document, for example, positive at the beginning and negative at the end, it might generate incorrect result for a two class sentiment analysis. To solve this problem, more class can be added in the sentiment or perhaps weights can be applied. When the most relevant information is at the beginning then the model can learn to set the vector z_t close to one and keep most of the information. However, as z_t becomes a value close to one, $1 - z_t$ declines and approaches zero which is irrelevant for the prediction anyway. The following diagram illustrates this:

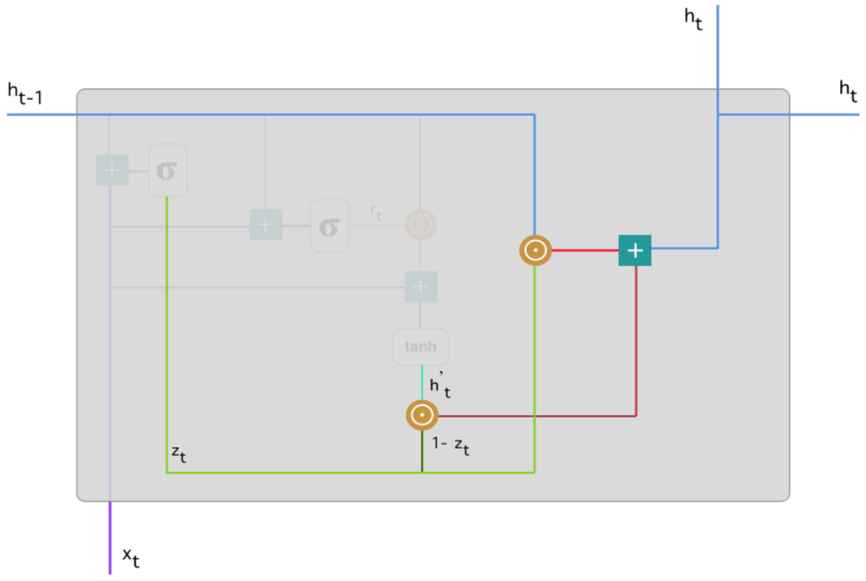


FIGURE 2.8: Diagram showing GRU function [32]

2.2.10 Evaluation of Sentiment Analysis Model

Like any other machine learning model, the model in the sentiment analysis task can be evaluated with training and testing accuracy. Comparison between the training and testing loss indicates how well the model works. The lower the loss, the better the model works. In practice, loss is reduced at each iteration of the model. The loss reduction rate is observed. How many epochs should be taken can also be determined by observing the gradual reduction in the training and testing loss. Loss is not a percentage like the accuracy. However, it is a summation of the false positive and false negative on each example during training and testing.

On the other hand, accuracy is the true positive and true negative values. For this kind of machine learning models accuracy is also compared to evaluate the model performance. For the dataset that is used for sentiment analysis it is split into training and testing sets. Finally, training and testing loss and accuracy is compared.

Chapter 3

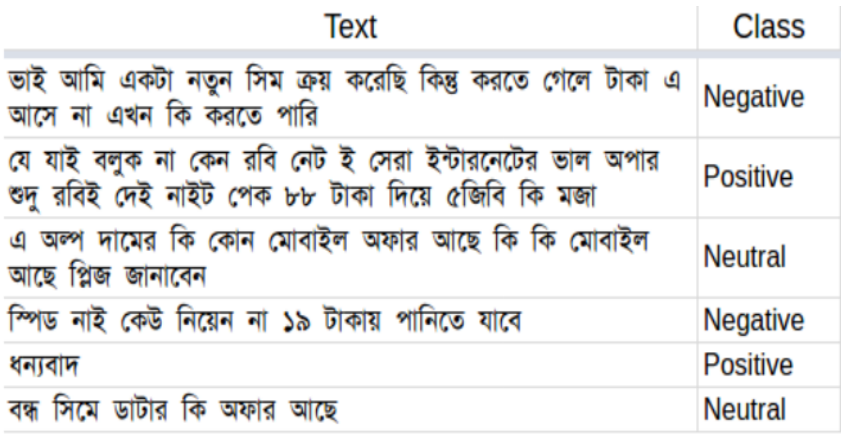
Sentiment Analysis

This chapter includes a sentiment analysis on Bangla language. A model is proposed that reflects better accuracy than the existing sentiment analysis work with the Bangla language. Data collection and the preprocessing part will be discussed with the methodology which is proposed in this thesis. All the experimental setup and their results will be discussed in the subsequent sections. A comparison of the results with the existing system will be performed. The LSTM algorithm is explained. Finally the results are discussed with graphical representations.

3.1 Data Collection and Preprocessing

Data have been collected from Facebook using Facebook graph API [34]. The data are mostly comments of the user. Reviews from pages, specifically from a mobile operator Facebook page. 34,271 thousand comments from Facebook were collected. All the unnecessary data tuples except those containing Bangla were removed. Then the data were tagged manually into Positive, Negative and Neutral class. Figure 3.1 shows how the data looks like after cleaning all the noise.

From Figure 3.1 we can see that the dataset contains columns. The “Text” column contains the texts and the “Class” column contains all the tags for the corresponding text. When the model is trained it can relate the sentences to their class which eventually leads to the prediction of a test set.



Text	Class
ভাই আমি একটা নতুন সিম ক্রয় করেছি কিন্তু করতে গেলে টাকা এ আসে না এখন কি করতে পারি	Negative
যে যাই বলুক না কেন রবি নেট ই সেরা ইন্টারনেটের ভাল অপার শুদু রবিই দেই নাইট পেক ৮৮ টাকা দিয়ে ৫জিবি কি মজা	Positive
এ অল্প দামের কি কোন মোবাইল অফার আছে কি কি মোবাইল আছে প্লিজ জানাবেন	Neutral
স্পিড নাই কেউ নিয়েন না ১৯ টাকায় পানিতে যাবে	Negative
ধন্যবাদ	Positive
বন্ধ সিমে ডাটার কি অফার আছে	Neutral

Figure 3.1: The dataset for the sentiment analysis work

In Table in 3.1 the data (sentence) counts are shown. As we can see there is not significant imblanace in the dataset. However, there are more instances of negative and neutral comments than positive comments.

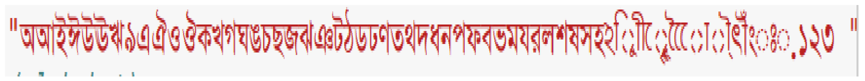
Table 3.1: Data Statistics

Class	**Count**
Positive	8271
Negative	14000
Neutral	12000
Total	**34271**

3.2 Character Encoding

Before the dataset can be used to train the model, it needs to be represented in a vector space. There are different methods to represent data into a vector space such as TF-IDF, Bag of Words, Distributed representation of words for example Word2vec, Glove etc. However, there are some drawbacks of these methods when it comes to the word level representation. The main problem with these models is that they totally rely on the words of the corpus. Since the sentiment analysis itself is an unsupervised learning, any word that has not been observed in the training period may not have contributions towards the test set. So, the results might not be complete. Consequently, the model would not know those words and can not predict it's meaning in the overall scenario. Most research involves words as the unit of a sentence. However, in [11] Xiang Zhang et al. performed an empirical study on character level text classification by using a convolutional net with an English dataset and found that this method works well on the real life data or on data generated by users. In this research the character level as well as as the word level representation is explored and two models for both of these unit encoding are proposed. A comparison of these models in terms of their accuracy for Bangla is performed and it is explained why the character level model works better than the word level model.

The accuracy of these models however depends on many other factors including the choice of alphabets, size of the dataset, hyperparameter tuning etc. In this work 67 characters of Bangla including space and some special characters are used. There are 49 letters in the Bangla language. However, the special characters are considered too. In the following figure all the 67 characters are illustrated:



"অআইঈউঊঋ৯এঐওঔকখগঘঙচছজঝঞটঠডঢণতথদধনপফবভমযরলশষসহঃিীুূৃেৈোৌ্ৎঁংঃ।১২৩ "

Figure 3.2: Characters

No numeric characters in Bangla are used. The numeric characters *"1"* ,*"2"* and *"3"* in Bangla were used instead of three other Bangla letters due to the reason that python could not recognize them. After that the characters are encoded with a unique ID from the list of the characters. The system will recognize each character with their own ID and this will further have benefits in the representations of the sentences. The encoding is illustrated in Figure 3.3. The order in the figure is just a random order for a demonstration purpose.

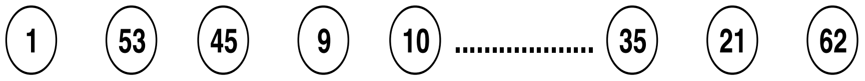


Figure 3.3: Character encoding

The length of the data instance is $l = 1024$ and it was observed from the dataset that most of the sentences are within this limit. However, sentences more than 1024 were truncated to 1024 and sentences less than 1024 were padded using 0s. Any other characters than those 67 selected characters are removed using a regular expression from python implementation.

3.3 Methodology

Deep Learning method has been applied successfully to the Natural Language Processing problems and achieved the state-of-the-art results in this field. Recurrent Neural Net [RNN] is a kind of Neural Network which is used for processing sequential data [12]. Later on, the researchers found some mathematical problems to model long sequences using RNN [13][14]. A clever idea was proposed by Hochreiter and Schmidhuber to solve this problem. The idea is to create a path and let the gradient flow over the time steps dynamically [13]. This path can be imagined as a water flowing path where different sources of water are merging together and we want to control the flow. It is known as Long Short Term Memory (LSTM) which was talked about in the background section previously in chapter 2. It is a popular and a successful technique to handle long-term dependency problem which is the domain for many NLP tasks. One of the variant of LSTM is the Gated Recurrent Unit (GRU) proposed by Cho et al. [33]. The difference between the LSTM and GRU is that it merged **Forget** and **Input** gates into an **Update** gate which means it can control the flow of information without the use of memory unit and it combines cell state and hidden state. The rest of the thing is the same as LSTM. In [15] Junyoung Chung et al. conducted an empirical study on three types of RNN and found that the Gated Recurrent Unit works better than the other two. GRU is also computationally more efficient than LSTM. The following equations explain how the GRU works from the mathematical points of view.

$$z_t = \sigma(W_z.[h_{t_{-1}}, x_t]) \tag{3.1}$$

$$r_t = \sigma(W_r.[h_{t_{-1}}, x_t]) \tag{3.2}$$

$$h`_t = tanh(W.[r_t * h_{t_{-1}}, x_t]) \tag{3.3}$$

$$h_t = (1 - z_t) * h_{t_{-1}} + z_t * h`_t \tag{3.4}$$

Here are the equations that demonstrate how the hidden state h_t is calculated in GRU. It has two gates, one is the update gate z, another one is the reset gate r. Equation (3.1) and (3.2) show how these two are calculated. The reset gate determines how to combine the new input with the previous memory, and the update gate determines how much of the previous memory to keep around. Finally, hidden state h_t is calculated as equation (3.4). However, the classification task of the sentiment is a step by step process. For example, to classify any sentence from the dataset, first it will go through the preprocessing step. Here, all the characters except the defined ones in Figure 3.2 will be filtered out from the sentence and the remaining sentence will be represented in a vector space. Every character will be given a numeric id and then it will be padded by zero to 1024 characters (any sentence with more than 1024 character will be compressed down to 1024). This vector will be fed through the model and eventually the model maps the input sentence to a sentiment class. In each hidden layer of the model, more lower level and meaningful features are extracted from the previous layer such as any usual neural network model and the output layer calculates the Softmax probability of each of the class. The class which has the highest probability is the predicted result.

3.4 Proposed Model

In this research two models are developed. One with the word level representation which is the usual case for most RNN work and then a character level model is also proposed which is the main focus of this work. Furthermore, a comparison of both the models in terms of their accuracy is performed. In this section both of these models and their architectures will be discussed. The word level is denoted as the baseline model and the second model as the character model.

3.4.1 Baseline Model

The baseline model is basically the word model. In this model each word is considered as a unit of the sentence. The architecture of the model is described with a graphical illustration. The baseline model consists of one embedding layer with 80 units and three hidden layers. Out of these three hidden layers two are LSTM layers with 128 units in each. In the third layer, a vanilla layer with 1024 unit in each is employed which is equal to the length of each sentence in the corpus. A dropout layer [16] with a probability of 0.3 is also employed between the output layer and the last hidden layer.

Combining a vanilla layer with the LSTM layer worked better for the purpose of predicting the sequence in Bangla. Essentially, the vanilla layer is convenient for predicting the successor or generating smaller sequences that eventually is fed into the output layer. A vanilla layer of 1024 units is employed which can individually identify each of the letters. However, for the base model, embedding layer is not used in accordance with the number of letters as the character level model which will be described in the next subsection. The reason of using vanilla in combination with the LSTM layers is that the vanilla layer itself is not able to predict longer sequences. Let assume an example sentence as the following:

Sunny works at Google

This is a very simple sentence with only one subject. So it can be tagged as Sunny : Subject and Google : Organization. If the model sees any such type of sequence it can understand the semantic meaning and the dependency at the lower level. Sunny works at .... : this can be predicted that the blank space may belong to an organization.

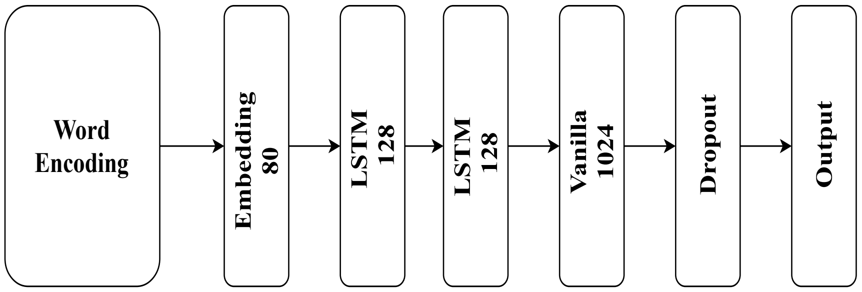


Figure 3.4: Word level model architecture

However, for long and complicated sentence structures the vanilla layer can not predict them and it does not seem to be a good idea for the corpus to use the vanilla layer at the beginning layers since LSTM can do better in understanding the complex dependencies. So the vanilla layer proved to be working better right before the output layer when the smaller sequences are already generated by previous embedding and LSTM layers. The architecture of the base model is illustrated in the Figure 3.4

3.4.2 Character Level Model

In this section the character level model is described. The proposed model consists of embedding layer with 67 units. We have 67 characters including the special characters. In this model the optimized number of hidden layers are three. Out of these 3 layers, two are with 128 GRU units and the last embedding layer is a vanilla layer same as before with 1024 units stacked up serially and the last layer is the output layer. A dropout layer of 0.3 between the output layer and the last hidden layer is employed. The model architecture is illustrated as follows:

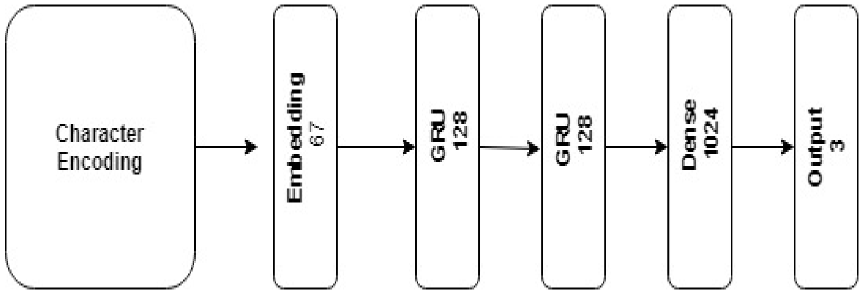


FIGURE 3.5: Character level model architecture

3.5 Experimentation

The model was run in 6 epochs with a batch size of 512 and Adam [17] was used as our optimizer. Categorical cross entropy was used as a loss function. The learning rate was set to 0.01 to train the model. Many different hyper parameters (learning rate, number of layers, layer size, optimizer) were used and this gave an optimal result.The hyper parameters were set by trial and error for the optimal value. The embedding size was kept to 67 as the dataset has 67 characters and the dropout was set to 0.3 between the output layer and the dense layer of the two models. Early stopping was used to avoid overfitting. All the experiments were done in python library named keras [35] which is a high-level neural networks API. The dataset were split into training and testing with a 80:20 ratio.

3.6 Results and Discussion

The result that was achieved from the character level model is better than the word level model. 80% accuracy on character level mode and 77% accuracy from our baseline model with word level representation was achieved. Recently, sentiment analysis achieved a highest of 78% accuracy in [7] using LSTM in Bangla with two class classification. Figure 3.6 is showing the training and testing loss of the model.

Here we can see that after a certain epoch the training loss started decreasing more than the testing loss. The testing loss on the other hand decreases at a slower rate compared to the rate of the training loss. The training was stopped at epoch 6 resulting in saving the model from overfitting. Figure 3.7 shows the training and testing accuracy of the character level model and Figure 3.8 shows the comparison between the two models. The most important observation from the experiments is that the character-level RNN could work for text classification without the need for word semantic meanings. It may also extract the information even if the word is not correct as it goes through each of the characters individually.

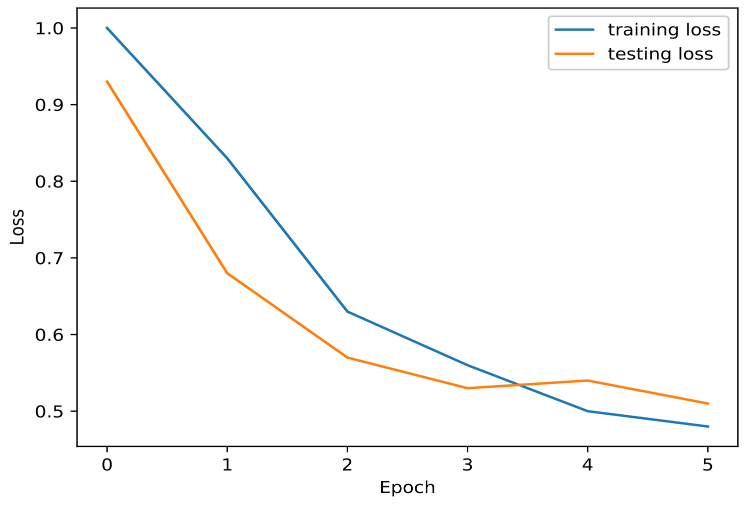


FIGURE 3.6: Training and testing loss

However, to observe the performance of this model across different datasets, more research is needed. Nevertheless, the result depends on various factors including the size of the dataset, alphabet choice, data quality etc. But the dataset in this thesis is focused on a specific telecommunication campaign domain. So this model can be helpful on some specific applications. The accuracy is calculated as a ratio of correctly classified data and a total number of data from the test set. The equation is as follows:

$$Accuracy = \frac{T_p + T_n}{(T_p + T_n + F_p + F_n)}$$

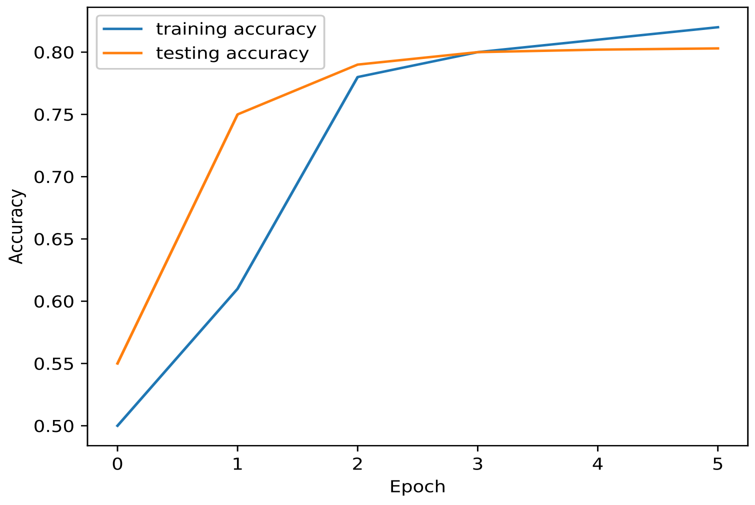


Figure 3.7: Training and testing accuracy

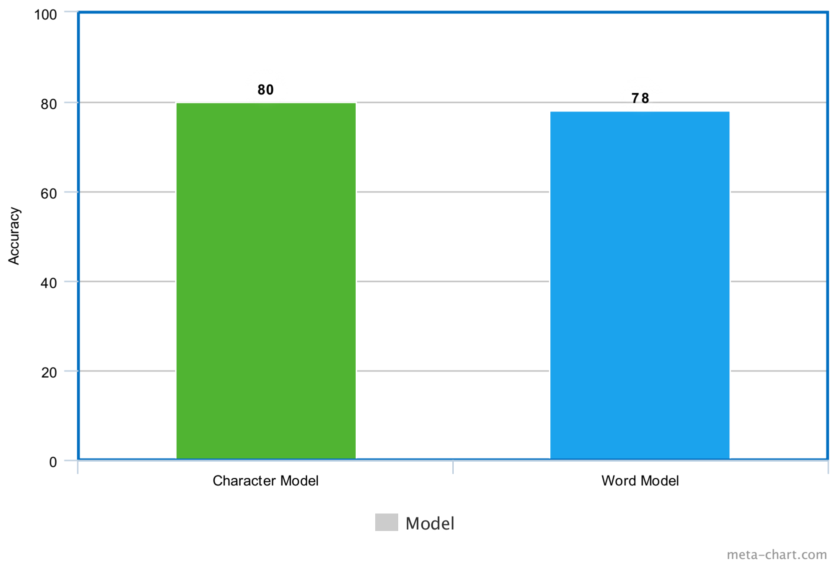


Figure 3.8: Comparison of the two models

The accuracy comes from the confusion matrix. The confusion matrix is basically a table that allows an insight to explore the exact numbers of true and false predictions from the model. It is widely used in machine learning and various artificial intelligence tasks to observe the performances of different models. From the confusion matrix table a ratio of ground truth and prediction is obtained which is eventually used as a percentage for performance. The predictions are divided into actual and predicted values. The actual values mean what is the reality and the predicted values gives the result from the model. The true value percentage can be calculated from the matrix. Let us assume some numbers as an example for better understanding the confusion matrix in the table below:

TABLE 3.2: Example confusion matrix

	predicted false	predicted true
actual false	20	10
actual true	20	150

As we can see from the table above, we have the actual false and predicted false as 20 examples from our example dataset. This means, out of 200 examples in the dataset, 20 examples which were false in reality and predicted as false by the model too. So these are the right prediction and known as True-Negative denoted as T_n. In the third row and second column there are 20 more examples which were true in reality but the model predicted them as False. So these are wrong predictions and known as False-Negative denoted as F_n. In the second row and third column the 10 examples those were false in reality but predicted as true. So, again these are wrong predictions and known as False-Positive denoted as F_p. Finally, the last 150 examples were predicted as true and were actually true. So, these are known as True-Positive denoted as T_p. For the accuracy formula, we find out the ratio of the number of correctly predicted examples to all the predictions as mentioned in the equation.

To conclude, this chapter offers a research on character-level RNN for sentiment analysis in Bangla. The model was compared with a deep learning model. Comparison of the two models across different data is one future goal of this work that will make it useable in the industries to extract sentiments from the social media reviews and comments.

Chapter 4

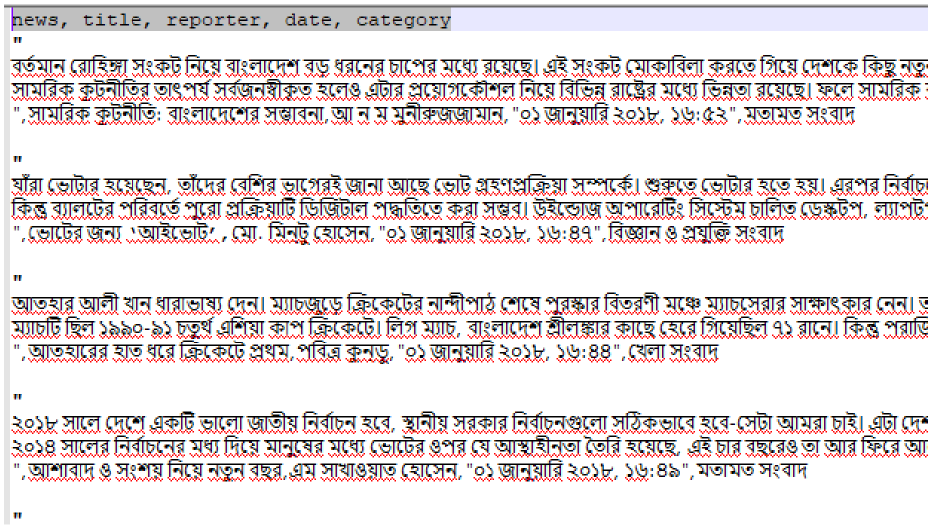
Topic Modelling

In this chapter the second contribution of the thesis will be described which is topic modelling. The problem statement, the corpus, experimental results and analysis will be reported. The basic contribution from this chapter is to develop a Bangla corpus from the most famous newspaper called "The Daily Prothom Alo", execute the preprocessing tasks and finally propose a topic model to classify news. Topic Modelling or any NLP work in general is a lot about preprocessing data. So a multi-phase preprocessing is done before designing the model. Topics are extracted and documents are classified according to their topics. Similarity measure is proposed along with the topic extraction. An evaluation method is also proposed to critically analyze the results. A comparison of the performance of different models in terms of coherence is done to see which variation of LDA works better on Bangla.

4.1 The Corpus

Collecting the data for the Bangla language has always been a challenge due to scarcity of resources and unavailability of publicly available corpora. Although various research has been going on with Bangla regarding NLP, none of those datasets are made available for the public. So, it took a while for this thesis just to understand the data and make a structured format on which topic modelling and other experiments can be performed. The dataset used for this thesis is a news corpus. It is collected from one of the most used and popular newspaper called

"The Daily Prothom Alo". A crawler is developed with python library called Beautiful Soup. The data has 7134 news articles from many different categories. All the news from January 1, 2018 to March 31st, 2018 were scraped. "The daily Prothom Alo" has an online archive section. The archive section was crawled for each day's paper over the mentioned 3 months. The news data were collected in a CSV file. The news data looks as follows:



```
news, title, reporter, date, category
"
বর্তমান রোহিঙ্গা সংকট নিয়ে বাংলাদেশ বড় ধরনের চাপের মধ্যে রয়েছে। এই সংকট মোকাবিলা করতে গিয়ে দেশকে কিছু নতু
সামরিক কূটনীতির তাৎপর্য সর্বজনস্বীকৃত হলেও এটার প্রয়োগকৌশল নিয়ে বিভিন্ন রাষ্ট্রের মধ্যে ভিন্নতা রয়েছে। ফলে সামরিক
",সামরিক কূটনীতি: বাংলাদেশের সম্ভাবনা,আ ন ম মুনীরুজ্জামান, "০১ জানুয়ারি ২০১৮, ১৬:৫২",মতামত সংবাদ

"
যাঁরা ভোটার হয়েছেন, তাঁদের বেশির ভাগেরই জানা আছে ভোট গ্রহণপ্রক্রিয়া সম্পর্কে। শুরুতে ভোটার হতে হয়। এরপর নির্বাচ
কিন্তু ব্যালটের পরিবর্তে পুরো প্রক্রিয়াটি ডিজিটাল পদ্ধতিতে করা সম্ভব। উইন্ডোজ অপারেটিং সিস্টেম চালিত ডেস্কটপ, ল্যাপট
",ভোটের জন্য 'আইভোট', মো. মিন্টু হোসেন, "০১ জানুয়ারি ২০১৮, ১৬:৪৭",বিজ্ঞান ও প্রযুক্তি সংবাদ

"
আতহার আলী খান ধারাভাষ্য দেন। ম্যাচজুড়ে ক্রিকেটের নান্দীপাঠ শেষে পুরস্কার বিতরণী মঞ্চে ম্যাচসেরার সাক্ষাৎকার নেন। ত
ম্যাচটি ছিল ১৯৯০-৯১ চতুর্থ এশিয়া কাপ ক্রিকেটে। লিগ ম্যাচ, বাংলাদেশ শ্রীলঙ্কার কাছে হেরে গিয়েছিল ৭১ রানে। কিন্তু পরাজি
",আতহারের হাত ধরে ক্রিকেটে প্রথম,পবিত্র কুনডু, "০১ জানুয়ারি ২০১৮, ১৬:৪৪",খেলা সংবাদ

"
২০১৮ সালে দেশে একটি ভালো জাতীয় নির্বাচন হবে, স্থানীয় সরকার নির্বাচনগুলো সঠিকভাবে হবে-সেটা আমরা চাই। এটা দেশ
২০১৪ সালের নির্বাচনের মধ্য দিয়ে মানুষের মধ্যে ভোটের ওপর যে আস্থাহীনতা তৈরি হয়েছে, এই চার বছরেও তা আর ফিরে আ
",আশাবাদ ও সংশয় নিয়ে নতুন বছর,এম সাখাওয়াত হোসেন, "০১ জানুয়ারি ২০১৮, ১৬:৪৯",মতামত সংবাদ

"
```

FIGURE 4.1: The news corpus: CSV file

In the dataset each row represents an instance from the newspaper. The features are **news**, **title**, **reporter**, **date**, **category**. The methods that are applied to this dataset are all unsupervised learning. However, we still collect the rest of the features so that the dataset can be applied in the future to extend as a publicly available dataset for further research with NLP. The corpus is organized chronologically from the past news to the most recent ones. This is done in order to experiment the topic trends over this period and see how they evolve in a Bangladeshi newspaper as a future work. However, the news are not ordered in any sequence in terms of category or anything else. The main objective of this chapter is to apply Topic Modelling techniques to find out the topics and then classify the news in terms of the topics the news belong to. For this, we do not need any sequence for the news. All the news articles were crawled serially on each day. News of different categories are crawled and saved into the CSV file. While saving into the CSV file, UTF-8 encoding was used. Without UTF-8 encoding python is not able to understand the characters for further processing. We removed all the hyperlinks from the text to make it more readable to the program. In the prepossessing section it will be discussed in detail.

Bangla is a popular peotic and a rich language in the subcontinent. It is the native language of Bangladesh and also widely used in the west. The language consists of 49 letters all together. Bangla language can be divided in Vowel, Vowel marks, Consonants, Consonant conjuncts, diacritical and other symbols, digits, and punctuation marks. The list of vowels and consonants are given as follows:

অ আ ই ঈ উ ঊ ঋ এ ঐ ও ঔ

Figure 4.2: Bagla Independent Vowels

The main 11 vowels in Bangla are known as independent vowels. They can be used as a standalone characters for many words. However, these vowels are sometimes conjugated with the dependent vowels to make specific sounds. The dependent vowels are used at the left, right or the base of the independent vowels. The independent vowels can also be wrapped by a dependent vowel. Sounds like **AA**, **EE**, **OO**, **OI**, **OU** etc. are made with combining the dependent and the independent vowels.

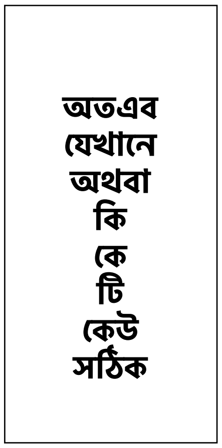
◌া ◌ি ◌ী ◌ু ◌ূ ◌ৃ ◌ে ◌ৈ ◌ো ◌ৌ

Figure 4.3: Bagla Dependent Vowels

ক খ গ ঘ ঙ চ ছ জ ঝ ঞ ট ঠ ড ঢ ণ ত থ দ ধ ন প ফ ব
ভ ম য র ল শ ষ স হ ড় ঢ় য় ৎ

Figure 4.4: Bagla Consonants

Bangla words are formed in combination of vowels, consonants and sometimes special characters. In Figure 4.5 some Bangla words are illustrated which will have their English romanized pronunciation and translation in the Table 4.1



অতএব
যেখানে
অথবা
কি
কে
টি
কেউ
সঠিক

Figure 4.5: Bagla words

4.2 The Crawler

A crawler is developed for the purpose of data collection due to the unavailability of any structured corpus on Bangla news. A python library called Beautiful Soup is used for this crawler. Open request of the url that needs to be crawled is delivered into the python code first. So it can go to that page and look for html divs or href links where it needs to grab the news from. The crawler then finds all the divs the news texts belong to. Inside each div there is a news title which is the headline. The title text is then collected at the first place. The crawler then gets into the link to get the detailed news where it has some **p** tags to fetch the texts from. At the same time we fetch the date and time, name of the reporter from that link and category from some other **p** tags around that link. After one iteration a single instance is collected. Then the iteration continues. Python lists are maintained to append each of the instances. Once all the instances are collected it is then inserted as rows into a CSV file. For debugging process each 10 days news were collected at a single run.

4.3 Preprocessing and Cleaning

As the saying goes, data preprocessing and cleaning is a major part of the implementation. "Garbage in, Garbage out". So is true for this dataset. A detailed process for the data cleaning will be discussed in this section. The dataset consist of over 7,000 news collected over a 3 months of time. The whole corpus is encoded in UTF-8. Tokenization, Stop words removal, Vectorization, Bigram model development and removing over and under frequent words are all part of the data preprocessing. The steps are discussed in following subsections.

4.3.1 Tokenization

From the news article collection each of the words is tokenized and put into an array. The sentences are split to make each token. This process is known as Tokenization. A formal definition of tokenization is as follows:

"Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens and at the same time throwing away certain characters, such as punctuation" [36]. An example of bangla word tokenization for the sentence : **Aami Banglay Gaan Gai** is **ami**, **Banglay**, **gaan**, **gaai**. Each of the word in the sentence is tokenized and from the implementation point of view, these tokenized words are then appended into a python list for further processing. The main goal of this tokenization is to access each word and represent them in such a way that the computer can map them with numeric values. This is where the Bag of Words model comes into play which will be explained in the next section. Once the words are tokenized and inserted into a list it is ready for the next step which plays a vital role in data preprocessing: Stop words removal.

4.3.2 Stop Words

Stop words are the words that do not play any role in finding the semantic meaning of the corpus. These words are basically connecting words like prepositions, conjunctions etc. So, before feeding the data into the model, removing the stop words is necessary. As with English, Bangla language has a lot of stop words too. These are the connecting words just like prepositions and conjunctions etc. However, Bangla being brought up into the NLP world quite recently, there is still no fully established list of stop words. So we developed an enhanced stoplist for our program. A list of 438 stop words were used. Some of the stop words from the list with their English translation is as follows:

Table 4.1: Stop words

Bangla Stop Words in English letters	**English Meaning**
Otoab	Therefore
Jekhane	Whereas
Othoba	Or
Ki	What
Ke	Who
Ti	The
Keu	Someone
Shothik	Right

From the above table it is seen that these words have zero contribution to classify a document or provide a semantic meaning. From the news corpus all these words are removed. Words occurring frequently over 50 percent of the documents or in less than 20 documents across the whole corpus are also removed.

4.3.3 Bag of Words Model

Once the corpus is tokenized and the stop words are removed it is then ready to make the bag of words model through which the whole corpus will be vectorized. Vectorization of the corpus is an important task due to its significance in understanding the words by the machine learning algorithms. Vectorization is a way to represent the corpus with numbers and the Bag of Words model is used for this. In the construction process of the Bag of Words model there are few sub processes. The first one is to make a dictionary. From NLP perception, a dictionary consists of the tokenized words in a list with only unique words. Then each document maps that dictionary to represent its vectorized version which only has a number. The numbers only represent the frequency of the words in that document. An example is as follows:

Dictionary = [coffee, milk, sugar, spoon]

Doc = [I love coffee with milk]

Vector = [0, 0, 1, 0, 1]

In this example, we can see that the bag of words model develops the Dictionary. As the name suggests, it is a model that holds the tokenized unique words in a list. When a document needs to be vectorized it is then mapped to the dictionary. The algorithm goes through each words and maps to the dictionary to identify if it is present or not. If the word is not found in the dictionary then it is given a value 0. However, if the word is present in the dictionary then the value is equal to the frequency of the word in the document.

4.3.4 Bigram

For the task of topic modelling, Bigram creation is an important part. A bigram is a sequence of two adjacent tokens in the corpus occurring frequently [37]. So a probability is calculated for these words to occur one after another. If they have a threshold value these word pairs are combined together and put into a new token in the dictionary. Basically bigrams are n-grams with $n = 2$. A conditional probability is calculated for bigrams. Probability of W_n given $W_{n_{-1}}$ is given with the following equation:

$$P(W_n|W_{n_{-1}}) = \frac{P(W_n, W_{n_{-1}})}{P(W_{n_{-1}})} \tag{4.1}$$

An example of a Bigram for English text is “Air”, Traffic. If we have a news data about airplanes or air traffic, it is highly likely that these two tokens will occur together may be at all times. So, a new token can be inserted into the existing list as **air_traffic**. Since topic modelling extracts topics and each topic consists of relevant words, it is important that these two words combined together fall into a single topic. In our proposed model, we used the Bigram model due to the reason that many times two adjacent words occur together in the corpus. We did not need Trigrams due to the Bengali language structure not having any three consecutive words meaning a single phrase.

4.3.5 Removing Rare and Common Words

After the Bigrams are made and the dictionary is enhanced with the Bigrams, we further remove some rare and common words to make the corpus a bit more meaningful to the proposed model. Words occurring in less than 20 documents and over 50 percent of the documents are removed. This 50 percent value is a threshold which makes sense as the words occurring in more than 50 percent of the documents are some extra stop words those are not listed in the stop words list.

4.4 Proposed Model

As the news corpus is ready to train the topic modelling techniques, we formalize the proposed model on how to train the Bangla news corpus to get the best out of it in terms of topic modelling. In this subsection we describe the proposed model in a step by step process. Our main goal from this research is to find a way to extract the topics from the corpus. In this chapter a methodology is also proposed to find out the right topic a news belong to. This way each news can be classified in their right category. The proposed model is illustrated in the following diagram:

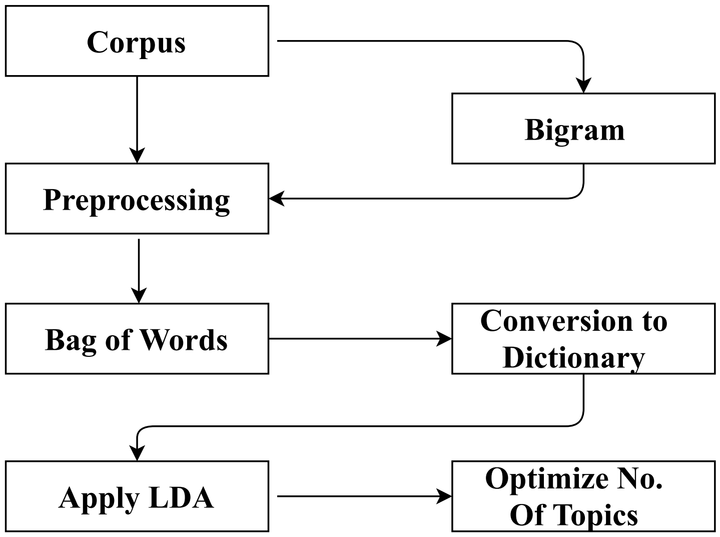


FIGURE 4.6: Proposed model for topic extraction

This is the basic structure on how the model works. Once the dictionary is ready and the preprocessing is done, the LDA algorithm is applied. The training is performed on 7134 news articles. It is an unsupervised learning. The *dictionary* needs to be assigned to the *id2word* parameter in the LDA algorithm. The dictionary is already set up in the preprocessing section. This whole dictionary goes into the model and it extracts a number of topics. However, LDA does not know how many topics it has to extract. A coherence based method is proposed to understand the optimal number of topics. From that experiment, the right number of topics are assigned as a hyperparameter during training the model. One problem with LDA is that it can get overfitted if too many topics are extracted. So, finding the right number of topic is important.

Before the model is trained and the LDA algorithm is run, a coherence-based experiment is performed. For this experiment, the number of topics are set to 200 and the Coherence VS Topic graph is monitored. We set the value to check the gradual coherence movement across different topics and found that it gets to the peak at around topic 47. So, we took that number and fed it into the algorithm. The model does not underfit or overfit. Once we get the model trained with our corpus then we want to evaluate the model with some experiments. We have performed a cosine similarity check between different news articles. Some news are similar and some of them are different. It is expected to have more similarity score between similar news and less similarity between news about different agenda. We achieved those scores for the trained LDA model. However, cosine similarity can also be achieved from the Doc2Vec model. That is why we develop the Doc2Vec model to compare the cosine similarity score between the LDA and the Doc2Vec and gain an insight on how both of these models work in terms of similarity. A comparison with other variations of LDA has also been performed.

4.5 Algorithm

In this section the LDA algorithm will be discussed. We will focus on how it extracts topics from the corpus after the algorithm has been trained. For our model we have used the Latent Dirichlet Allocation (LDA) algorithm proposed by Dr. Blei in 2003. So we will first discuss some terminology those are going to be used frequently for the LDA discussion. First we need to know what a Topic, Term and a Document mean from the LDA points of view.

Topics: A topic is a collection of words extracted from the corpus the LDA algorithm applied on. The words belonging to a topic are coherent and they collectively mean a single category. An example of a topic is as follows:

Topic 1 : Trump, U.S.A, Immigration, Immigrants, H1B
Topic 2: Traffic, Accident, Car, Death, Debris

As can be seen from these two topic examples, they both contain some words or better known as terms from the corpus. However,the words are correlated and collectively they can be interpreted by a human as a single topic. So, we can tag the topic 1 as “Immigration in the US” and topic 2 as “Traffic Accidents”. Basically LDA collects the topics from the news corpus and then we can use these topics in various ways to perform different NLP tasks such as Document classification, Sentiment Analysis, Understanding the meaning of any big corpus without going through it manually etc.

Terms: From the LDA perspective a Term is a word from the corpus. All the unique words across the complete corpus is known as terms.

Documents: A document is a collection of sentences that discuss about a single topic in a corpus. In most cases a corpus is a collection of different document. In this thesis we have 7,134 documents in our corpus talking about different topics.

LDA is a bottom up approach. It assumes that each document is a mixture of topics [22]. Each topic then generates words based on their probabilities. The probabilities sum up to one in each topic. LDA backtracks and tries to assume which topic would generate this corpus in the first place [22]. For this, LDA first generates a document-term matrix which has all the documents and all the words from the corpus. The values in this matrix represent the frequency of that term in the document. Assume we have n documents $D_1, D_2, ..., D_n$ and m terms $K_1, K_2, ..., K_m$. So the matrix will look as follows:

Table 4.2: Document-Term matrix

	K_1	K_2	K_m
D_1	1	0	2
D_2	3	1	1
D_n	0	0	4

The matrix can map each term and document with a frequency of words. Generally, this is a sparse matrix with lots of 0s since a single document can have only specific terms and not others. Later, LDA generates another matrix which maps a topic to a term. Assume we have p topics as $T_1, T_2, ..., T_p$ and as before m terms $K_1, K_2, ..., K_m$. The matrix is reported in Table 4.3.

In this matrix each term is either assigned a topic or not. So, it is a binary matrix where each topic is assigned to a term. However, the topics are not labelled in

Table 4.3: Term-Topic matrix

	K_1	K_2	K_m
T_1	1	0	0
T_2	0	1	0
T_p	0	0	1

LDA. Probability of a topic is calculated and only the top terms for a topic is considered to belong to the topic. What LDA does is it goes through each term K and assigns a topic T with a probability P. This probability comes from two other probabilities P_1 and P_2 as follows:

$$P_1 = P(Topic|Document)$$
$$P_2 = P(Term|Topic)$$

Here, probability P_1 is the proportion of terms K in the document D that are currently being assigned to topic T. Probability P_2 is the proportion of assignments to topic t over all the documents D that comes from this term K. These two probabilities are then multiplied together and that is the probability of this assignment of the topic to the term. At each run all the terms are then assigned a topic. In the next iteration the probabilities are updated with better topic-term assignments that is more coherent. At some point the model converges with no improvement in the probabilities. This is when the algorithm stops and all the terms are assigned to their right topics.

LDA algorithm has some hyperparameters. The two matrices can be controlled with Alpha and Beta hyperparameters. Alpha and Beta represent Document-Topic and Topic-Term compactness respectively. The higher the value of Alpha the Documents are composed of more topics and the higher the value of Beta each topic is composed of more words. These need to be adjusted according to the goal of the model.

Number of terms under each topic is another hyperparameter. For the purpose of this thesis it is set to 10. It makes more sense to see 10 words under each topic for the corpus of our size. LDA is one of the most popular topic modelling algorithms that gives state of the art results for English. The proposed model works good on the Bangla too. The main core parts of the LDA work as Bayesian Network.

4.6 Experimentation

The first experiment that we perform is to understand the number of topics to infer from the trained model. LDA itself does not understand the optimal number of topics. So we performed an experiment to understand the optimal number of topics. It is important to know how many topics we should infer from a trained LDA corpus. It varies on the dataset and the main goal of the research. Our purpose is to infer topics from an online newspaper with about 7,134 news article instances. When too many topics are inferred from the LDA model it may get overfitted which is not useful. On the other hand extracting too few topics does not give meaningful result. So a coherence based value is considered for understanding the right number of topics. We have experimented the model with 200 topics along with the aggregated coherence value for the topics as follows:

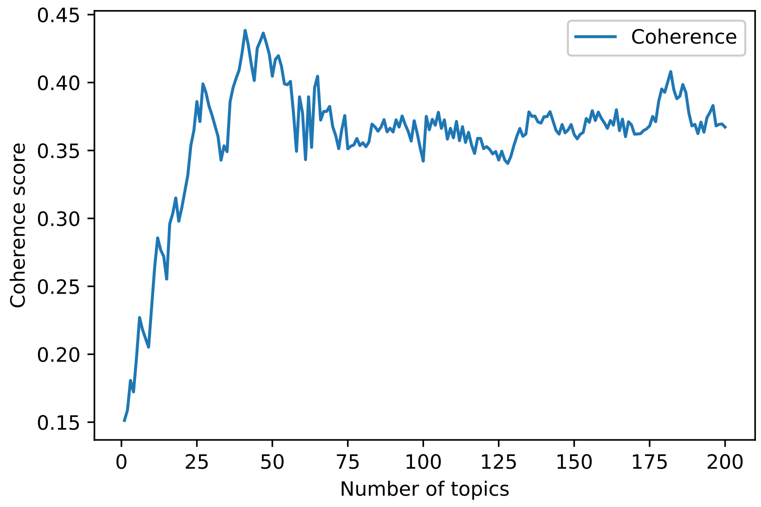


Figure 4.7: Coherence based number of topics

As can be seen from this plot, along with the increment in the number of topics, the coherence value increases rapidly during the beginning and gets to its peak when the number of topics is around 47. When we have approximately 47 topics in this case the model performs the best according to its coherence value. After that the coherence value falls off gradually. We perform this experiment for 200 topics which is more than the number of topics a newspaper is likely to have. For the purpose of this experiment the model was run 200 times and each time the coherence value is checked to understand how it changes with the number of topics. At the peak we have a coherence value somewhere around 0.45. So we took 47 as the right hyperparameter. In the coming section it will be discussed on how the coherence value makes sense for these topics.

An experiment with fewer topics is also performed. 10 and 20 topics are experimented at the beginning level before experimenting with 200 topics.

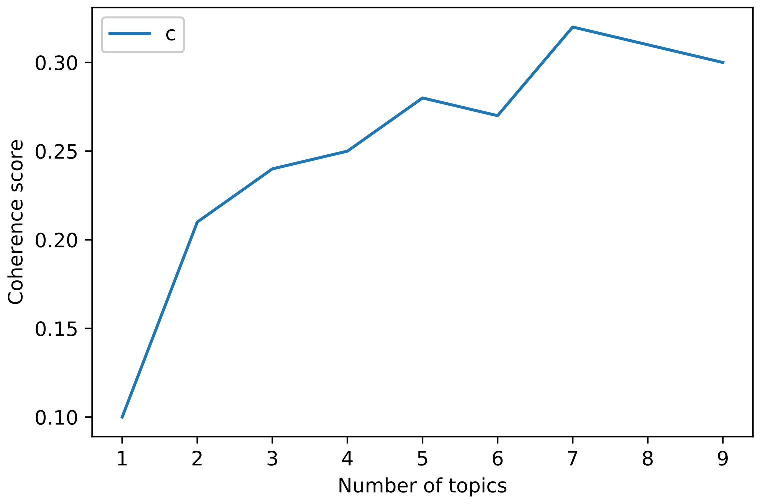


FIGURE 4.8: Coherence based number of topics (t=10)

From these experiments it is seen that none of these reach the coherence value around 0.45 which was the highest when done with 47 topics. Although in Figure 4.5 we can see that the coherence value is going up at around 0.375 but it does not still reach the maximum. So, 47 topics is the optimal number for our purpose. In the subsequent section we will see the extracted topics and their meanings.

4.6.1 Topic Extraction

In this section we will explore the extracted topics and their meanings. So, we have 47 topics extracted from this experiment. The topics with their English translation are as follows:

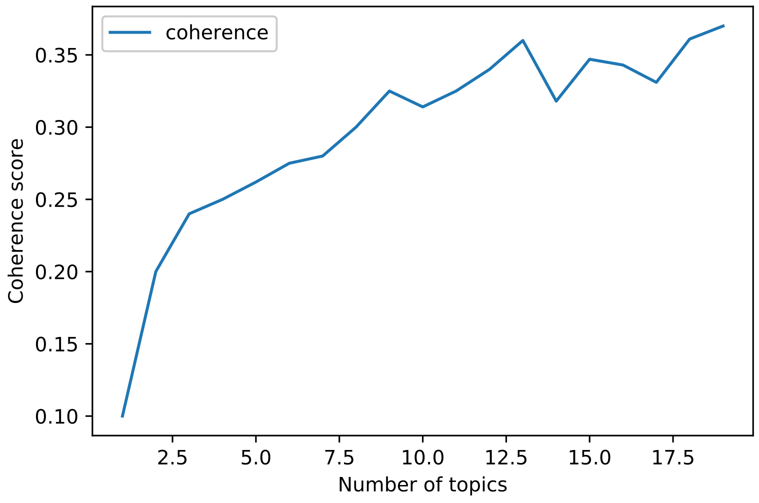


FIGURE 4.9: Coherence based number of topics (t=20)

TABLE 4.4: Extracted Topics from the optimized LDA

Tag	**Extracted Topic**
Life and Culture	(0, 0.024*“Festival” + 0.016*“Section” + 0.015*“Cultural” + 0.015*“tradition” + 0.014*“Eid” + 0.014*“End” + 0.013*“Eyes” + 0.013*“Life” + 0.011*“House” + 0.009*“Year”)
Film Festival	(1, 0.037*“Bangladesh” + 0.029*“India” + 0.018*“Festive” + 0.017*“In hand” + 0.016*“Film” + 0.015*“Bengali” + 0.014*“Sound” + 0.013*“Movie” + 0.011*“Deny” + 0.011*“Step”)
Bangladesh Election	(2, 0.023*“Development” + 0.017*“Past_year” + 0.016*“Bangladesh” + 0.015*“County” + 0.015*“This year” + 0.014*“Election” + 0.013*“Commitment” + 0.012*“Environment” + 0.012*“South_coast” + 0.012*“Protect”)
Media	(3, 0.051*“Song” + 0.033*“Seminar” + 0.030*“News” + 0.027*“Press_briefing” + 0.026*“Local” + 0.024*“Program” + 0.024*“Songs” + 0.023*“Name” + 0.023*“Political” + 0.022*“Importance”)

Continuation of Table 4.4

Tag	**Extracted Topic**
Election	(4, 0.045*“Vote” + 0.032*“Schedule” + 0.032*“Commission’s” + 0.028*“Election” + 0.027*“Election_commissioner” + 0.024*“Candidate” + 0.021*“Announce” + 0.018*“Section” + 0.015*“Postpone” + 0.015*“Head”)
Election	(5, 0.074*“Election” + 0.027*“Election’s” + 0.025*“Commission” + 0.023*“Legal” + 0.023*“Polling” + 0.018*“Government” + 0.017*“Law-order” + 0.014*“Election_commission” + 0.013*“By-law” + 0.013*“Ministry”)
Public Exams	(6, 0.035*“Power” + 0.029*“Student” + 0.023*“Exam” + 0.021*“Management” + 0.021*“Right” + 0.021*“Answer” + 0.020*“Question” + 0.019*“Delivery” + 0.018*“Ethical” + 0.017*“Politics”)
Misc	(7, 0.045*“Police” + 0.025*“heat” + 0.024*“facebook” + 0.022*“Immigration” + 0.020*“Less” + 0.019*“Achieve” + 0.018*“Computer” + 0.017*“Journalist” + 0.017*“Execute” + 0.015*“Cold”)
USA immigration	(8, 0.052*“Trump” + 0.044*“President” + 0.025*“USA” + 0.023*“Against” + 0.020*“Administration” + 0.019*“Immigration” + 0.019*“Complaint” + 0.017*“Foreign_affairs” + 0.016*“Order” + 0.016*“Direct”)
Law	(9, 0.048*“Head” + 0.031*“chief_justice” + 0.021*“Section” + 0.017*“Supreme” + 0.015*“Worker” + 0.014*“Court” + 0.012*“Justice’s” + 0.012*“Prosecution” + 0.011*“Law” + 0.011*“chief_justice”)
Cinema or Movie	(10, 0.020*“In_movie” + 0.017*“Acting” + 0.016*“Theme” + 0.016*“Death” + 0.015*“Audience” + 0.013*“Excitement” + 0.013*“Profit” + 0.012*“Gradual” + 0.012*“Super_hit” + 0.012*“In_between”)
Finance or Politics	(11, 0.045*“Money” + 0.039*“Bank” + 0.027*“League” + 0.026*“Awami_league” + 0.023*“Awami” + 0.020*“Chairman” + 0.019*“Sonali_bank” + 0.018*“Parliament” + 0.014*“Financial” + 0.013*“Institution”)

Continuation of Table 4.4	
Tag	**Extracted Topic**
Police or Crime	(12, 0.059*“Notice” + 0.024*“Yourself” + 0.023*“Police” + 0.018*“Police’s” + 0.017*“Hassan” + 0.017*“Arrest” + 0.016*“Statement” + 0.014*“Record” + 0.013*“person” + 0.013*“Presence”)
Misc	(13, 0.050*“Abdul” + 0.043*“description” + 0.030*“Life” + 0.026*“Final” + 0.022*“Bangla” + 0.019*“Form” + 0.017*“Dhaka” + 0.017*“Current” + 0.016*“Request” + 0.015*“Return”)
Movie or Cinema	(14, 0.052*“Movie” + 0.026*“Regarding” + 0.025*“good” + 0.021*“Search” + 0.019*“Interview” + 0.019*“Cinema” + 0.019*“Case” + 0.018*“Behind_the_scene” + 0.018*“Far” + 0.017*“Sensor”)
Law or Legal Issues	(15, 0.041*“Shamim” + 0.024*“Deny” + 0.021*“Weekly” + 0.020*“Issue” + 0.018*“Supreme” + 0.017*“Court” + 0.015*“Legal” + 0.014*“Influence” + 0.014*“Prosecution” + 0.014*“Statement”)
Transportat-ion	(16, 0.022*“Destination” + 0.019*“Road” + 0.018*“Maintenance” + 0.016*“Start” + 0.015*“Continuation” + 0.013*“Road_cleaning” + 0.013*“Highway” + 0.013*“Big_project” + 0.012*“Look_after” + 0.012*“Repeat”)
Accident	(17, 0.028*“Land” + 0.025*“Run” + 0.019*“leave” + 0.018*“Ahead” + 0.017*“traffic” + 0.017*“Air” + 0.014*“Crashed” + 0.014*“landing” + 0.012*“control” + 0.012*“death”)
City or government	(18, 0.040*“Mayor” + 0.038*“personal” + 0.026*“Dhaka” + 0.018*“league” + 0.016*“team” + 0.016*“Government” + 0.015*“Project” + 0.015*“Editorial” + 0.014*“city_mayor” + 0.013*“National”)
Entertainment or Media	(19, 0.085*“of_entertainment” + 0.044*“Prothom_Alo” + 0.035*“Statement” + 0.027*“Favorite” + 0.026*“Positive” + 0.019*“Movie” + 0.018*“Sentiment” + 0.017*“Support” + 0.016*“Section” + 0.015*“audience”)

Continuation of Table 4.4	
Tag	**Extracted Topic**
Water quality	(20, 0.042*“Water” + 0.021*“Plenty” + 0.020*“University” + 0.019*“Dhaka” + 0.017*“Red_color” + 0.014*“Dorm” + 0.014*“Toxic” + 0.013*“Harmful” + 0.013*“Food” + 0.012*“Drink”)
Misc	(21, 0.027*“Postpone” + 0.021*“Flow” + 0.021*“Lift” + 0.020*“To_live” + 0.019*“perception” + 0.019*“Hospital” + 0.015*“Drama” + 0.015*“Subject” + 0.014*“to_make” + 0.014*“blank”)
Students and Transportation	(22, 0.036*“Students” + 0.024*“Zone” + 0.021*“Partial” + 0.020*Transportation” + 0.020*“Methodology” + 0.019*“Buses” + 0.019*“Arrange” + 0.018*“Shortage” + 0.016*“Issue” + 0.016*”troublesome”)
Misc	(23, 0.021*“” + 0.020*“Caution” + 0.019*“Competition” + 0.016*“Good” + 0.015*“Question” + 0.015*“Shut off” + 0.014*“Knowledge” + 0.013*“Story” + 0.013*“Opportunity” + 0.012*“Protect”)
Media award	(24, 0.054*“Editor” + 0.045*“Prothom_Alo” + 0.033*“Achievement” + 0.033*“Name” + 0.030*“Warmth” + 0.027*“January” + 0.025*“Presence” + 0.025*“Nomination” + 0.023*“Best_paper” + 0.021*“award”)
Flyover project	(25, 0.043*“Foreign” + 0.037*“People” + 0.022*“Solution” + 0.018*“flyover” + 0.015*“Roads” + 0.015*“Made” + 0.014*“Invest” + 0.014*“to_make” + 0.013*“traffic_problem” + 0.013*“appreciation”)
Literature	(26, 0.021*“known” + 0.018*“Poem” + 0.018*“About_friend” + 0.017*“Story” + 0.017*“love” + 0.015*“Women” + 0.014*“Often” + 0.014*“Wife” + 0.014*“Couple” + 0.012*“Understanding”)
Bangla Language	(27, 0.034*“regarding” + 0.033*“language” + 0.024*“Bangla” + 0.023*“pain” + 0.023*“read” + 0.021*“publish” + 0.020*“in_bangla” + 0.018*“literature” + 0.018*“future” + 0.016*“generation”)

Continuation of Table 4.4	
Tag	**Extracted Topic**
Health and medicine	(28, 0.023*“health” + 0.017*“treatment” + 0.017*“child” + 0.015*“medicine” + 0.014*“bangladesh” + 0.014*“physical” + 0.013*“gain” + 0.013*“young” + 0.013*“national_health” + 0.012*“type”)
Wedding and Finance	(29, 0.027*“marriage” + 0.021*“gown” + 0.016*“wedding” + 0.016*“must” + 0.015*“loan” + 0.015*“marriage_loan” + 0.014*“privilege” + 0.014*“support” + 0.014*“states” + 0.014*“regarding_marriage”)
Administrat-ion	(30, 0.032*“meeting” + 0.022*“seminar” + 0.022*“amount” + 0.022*“statement” + 0.021*“mature” + 0.021*“funny” + 0.019*“English” + 0.016*“bring_about” + 0.014*“place” + 0.014*“year”)
Lifestyle	(31, 0.017*“opportunity” + 0.014*“family” + 0.013*“middle_class” + 0.013*“finance” + 0.012*“organization” + 0.012*“facility” + 0.012*“department” + 0.012*“bank” + 0.012*“micro_finance” + 0.011*“repay”)
Movie and Cinema	(32, 0.077*“cinema” + 0.058*“movie” + 0.019*“public” + 0.018*“combine” + 0.015*“overnight” + 0.015*“positive” + 0.015*“act” + 0.013*“girls” + 0.012*profit_share” + 0.012*“year”)
Family and Life	(33, 0.021*“Home” + 0.021*“kids” + 0.019*“winter” + 0.018*“video” + 0.015*“regarding” + 0.014*“mental” + 0.013*“entertainment” + 0.013*“digital” + 0.012*“family” + 0.012*“bonding”)
Movie	(35, 0.026*“Cinema” + 0.020*“Premier” + 0.020*“Director” + 0.019*“Announce” + 0.017*“Ending” + 0.016*“Actress” + 0.015*“Next_movie” + 0.015*“Release” + 0.014*“The_movie” + 0.012*“Sensor”)
Weather and Season	(36, 0.036*“winter” + 0.025*“season” + 0.023*“statement” + 0.017*“happens” + 0.017*“viral” + 0.016*“most” + 0.015*“team” + 0.015*“explanation” + 0.014*“good” + 0.012*“world”)

Continuation of Table 4.4

Tag	**Extracted Topic**
Education loan	(37, 0.068*“loan” + 0.043*“child” + 0.031*“repay” + 0.028*“Bank” + 0.016*“Dhaka” + 0.015*“education” + 0.014*“national_economy” + 0.014*“study” + 0.013*“money” + 0.012*“big_project”)
Information Technology	(38, 0.030*“digitization” + 0.029*“IT” + 0.027*“infrastructure” + 0.024*“authority” + 0.021*“statement” + 0.020*“technology” + 0.019*“upgrade” + 0.018*“software_market” + 0.017*“project” + 0.015*“economy”)
Foreign investment	(39, 0.043*“money” + 0.022*“finance” + 0.020*“currency” + 0.019*“approval” + 0.015*“pair” + 0.015*“law” + 0.014*“permission” + 0.013*“home_affaris” + 0.012*“america” + 0.012*“lessen”)
Project on IT	(40, 0.022*“proposal” + 0.021*“sometimes” + 0.019*“deny,” + 0.018*“mobile” + 0.017*“states” + 0.014*“done” + 0.013*“process” + 0.013*“person” + 0.012*“phone” + 0.011*“invest”)
Bangladesh Education	(41, 0.058*“education” + 0.027*“class” + 0.022*“percentage” + 0.020*“for_education” + 0.017*“average” + 0.015*“education_ministry” + 0.014*“curriculum” + 0.014*“efficient” + 0.013*“home” + 0.013*“nation_wide”)
Investment and Economy	(42, 0.031*“invest” + 0.031*“fiscal_year” + 0.023*“small_amount” + 0.019*“previous_year” + 0.015*“january” + 0.015*“” + 0.015*“million” + 0.014*“money” + 0.014*“year” + 0.012*“people”)
Foreign Affairs	(43, 0.034*“USA” + 0.033*“unstable” + 0.025*“declining_economy” + 0.022*“rise” + 0.020*“law” + 0.020*“reply_trump” + 0.019*“chairman” + 0.018*“Syria” + 0.016*“Islamic” + 0.015*“regarding”)
Stock Market	(44, 0.037*“price” + 0.024*“NewYork” + 0.024*“rapid” + 0.024*“Share_market” + 0.021*“share_price” + 0.017*“loan_issue” + 0.016*“affected” + 0.014*“type” + 0.013*“category” + 0.013*“news”)

Continuation of Table 4.4

Tag	**Extracted Topic**
Media	(45, 0.059*"book" + 0.032*"award" + 0.031*"people" + 0.022*"book_fair" + 0.022*"Prothom_alo"+ 0.019*"writer" + 0.017*"from_book" + 0.016*"Class" + 0.016*"demand" + 0.015*"february")
Power supply	(46, 0.047*"past" + 0.027*"year" + 0.027*"collection" + 0.022*"help" + 0.021*"supply" + 0.020*"past_year" + 0.019*"electricity" + 0.019*"load_shedding" + 0.015*"power" + 0.015*"achievement")

End of Table

These are all the topics extracted from the newspaper. Each topic consists of 10 words related to the topic. Some topics are a bit mixed and the meaning can be different but most the other topics makes sense and can be classified as a category from the newspaper. However, the topics are not automatically annotated. By seeing the word groups, the annotation for each topic is made manually since the LDA only provides group of words/terms known as topics. With each word, a probability is achieved in a descending order.

4.6.2 Similarity Measure

Since a trained LDA model already groups topics in terms of their keywords, we undergo through an experiment to explore the cosine similarity measure from our trained LDA model. A couple of document pairs are fed into the model and the cosine similarity value is observed. Similarity measure can also be performed with a Doc2Vec model. A comparison of the similarity scores from both the LDA and the Doc2Vec is explored. The scores are reported in the Table 4.5:

A pair of documents from the dataset are fed into both of these models to observe the similarity score. For example, doc 1 and doc2 are two highly related documents. They are both talking about a news on Myanmar Rohingya issue of Bangladesh. As a human interpreter someone will judge these two news articles as a highly related pair. LDA cosine similarity gives this pair a 97.15% similarity which would have been almost close to a human interpreter. On the other hand,

Table 4.5: Showing cosine similarity score between different models

Document Pairs	LSI	HDP	LDA	Doc2Vec
(doc5, doc9)	21.01%	0.00%	19.07%	50.91%
(doc5, doc6)	83.14%	0.00%	71.63%	72.55%
(doc271, doc272)	82.46%	0.00%	68.68%	60.61%
(doc1, doc2)	97.09%	29.21%	97.15%	68.54%
(doc1, doc513)	28.91%	0.00%	72.45%	30.31%
(doc 1916, doc 1060	89.65%	0.00%	80.99%	37.91%

Doc2Vec performed poorly and gave it only a 68.54% similarity which is not the case in reality. Document 1916 and document 1060 are talking about “Technology”. In this case LDA performs better than the Doc2Vec again. Now lets see how these models work on dissimilar news. It is expected that two dissimilar news are likely to have a low cosine similarity score. A similarity check for document 5 and document 9 is done where one is about “Sports” and the other news is about “Foreign Affairs”. LDA gives only a 19.07% match where Doc2Vec still gives it a 50.01%. So, LDA performs better than the Doc2Vec for both similarity and dissimilarity.

Now let us focus on the LSI model. our LSI model performs better than the LDA in some cases. As we can see, it has a 21.01% cosine similarity between doc1 and doc9. This pair of document is however two different topic. In this case LDA performs better to understand the dissimilarities. However, when it comes to calculate cosine similarity for similar documents, LSI performs better. The same is true for similarity score for doc1, doc2 pair and doc5, doc6 pair. These two pair of documents are closely related in terms of their topics and LSI is capturing the relevance better than the LDA model. One interesting point to note on the LSI experiment is, it failed significantly in understanding the similarity of documents those are far apart from each other in terms of their occurrence order in the corpus. In the dataset, doc1 and doc513 are two documents talking about “Technology” but LSI thinks it has only 28.91% match. The SVD based LSI captures similarities a little better than the LDA, however, it does not show a stable result due to being worse in capturing similarities of two far apart data points.

We also have similarity measure with the HDP model which performs the worst of these all. As we can see from Table in 4.5, in most of the cases HDP can not capture any similarity at all. It fails to relate the similarities for all the pairs except for doc1-doc2 pair which is a highly relevant pair in terms of their sentence and word structures. For all the other pairs it did not work at all. The mixture model of HDP does not work. As we have already discussed some of the limitations of HDP model in chapter 2, it seems that the HDP can not relate the data points together. This is where HDP fails to interrelate the documents. Also, while assigning topics to documents, neighboring documents should have a higher probability with the same topic which is not happening for HDP in the experiment and we can clearly see this with the values we have from the HDP. Let us take a look at the average value of these models to compare the similarity and dissimilarity from the graph below:

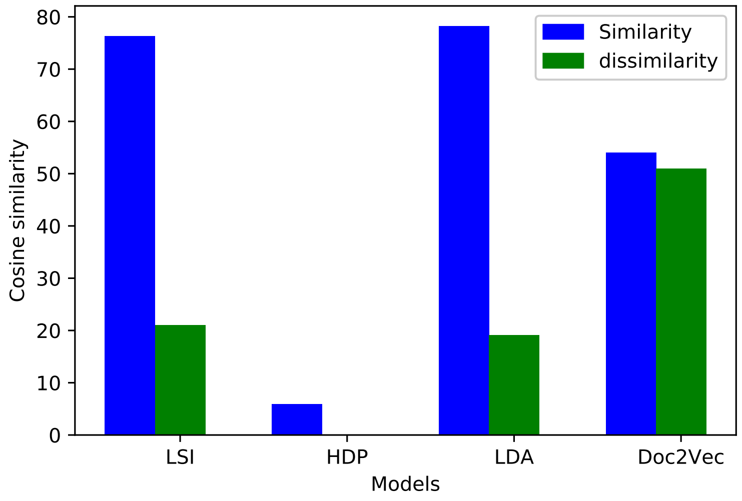


FIGURE 4.10: Similarity Dissimilarity of Cosine average

The reason that LDA performs better for Bangla lies in the way it works. Generally LDA can map documents to a vector and further works on a cosine similarity. Doc2Vec is a variation of Word2Vec with an extension of a document vector. One reason for LDA working better is that it can capture the topics already during the training time. So the topics help classifying the documents while scoring the cosine similarity. On the other hand Doc2Vec can understand the semantic meaning of the sentence inside a single document and can predict the next word as well. However, Bangla being a complex language in terms of their sentence structures and unpredictable to know the word's semantic meaning, Doc2Vec fails to understand the similarity. In this situation, LDA seems to have more capability to understand the overall semantic meaning with their topic extraction techniques.

4.6.3 Performance Comparison with Other Topic Model Algorithms

We have performed a comparative analysis between different topic models to see how well they work. The goal of this analysis is to explore on how the different topic models perform based on their coherence performance measurement. The research mainly focuses on the LDA algorithm and its different variations. So, we have compared the LSI, HDP, LDA standalone, LDA modified and LDA in combination with LSI. In this section we will illustrate the algorithm performance along with the topics from the different algorithms.

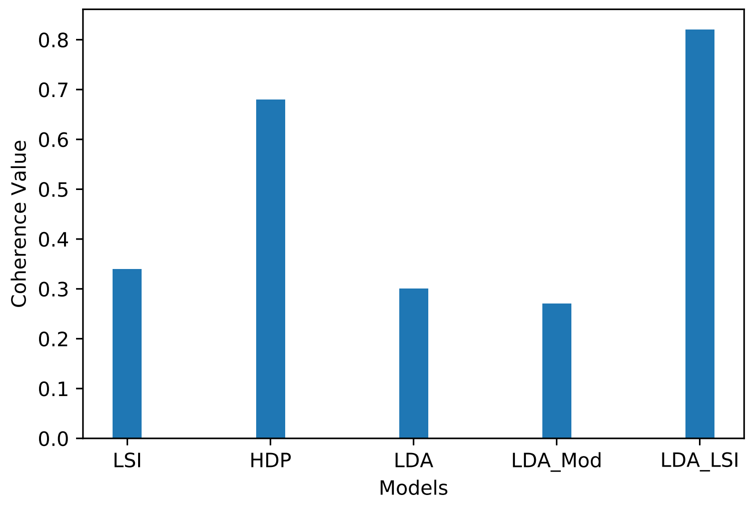


FIGURE 4.11: Model performance comparison

As we can see from the graph, we have the LSI model which is an SVD based algorithm for topic modelling. The LSI algorithm has around 0.3 coherence value. For our corpus, dimensionality reduction based LSI algorithm did not perform significantly well as can be seen from the figure. The corpus that we have is only the news texts and when converted to the bag of words and after that represented as a matrix, dimensionality reduction will not add any significant changes due to the characteristics of the dataset. The dataset is sparce and since we do remove the stop words and frequent and rare words in the preprocessing section, dimensionality reduction has no major impact on the topic modelling performance. Therefore, LSI is not an ideal topic modelling algorithm for this kind of news classification.

Next, we experiment with HDP algorithm having around 0.68 coherence value. This is indeed a significant improvement for the topic modelling. Since the news corpus has many news articles that can be grouped together, it seems from the experiment that the HDP mixture model can work better in terms of efficiency. The base distribution H having infinite number of atoms to support is the main energy behind the HDP model. The hierarchical layer being added to calculate the number of topics is also an added advantage for this model. So, overall HDP can be considered for news classification with Bangla. Although it does not have the best coherence measure, it can still be used for topic modelling for Bangla with a decent coherence.

Next, is the base LDA algorithm. LDA is a probabilistic algorithm that generally works efficiently with news classification for English. It is one of the more widely used algorithms for topic modelling. However, the base LDA model does not show any significant improvement for the corpus when the basic non-tuned hyperparameters are kept. 10 topics are extracted as in the default settings. However, this is not optimized. LDA does not know the number of topics beforehand and it needs to be provided. So, the performance achieved is even worse than the LSI model. However, with a proper hyperparameter adjustment, LDA can perform better than LSI which we will see in a bit.

After that, another experiment with LDA_mod is performed which is basically a modified version of the LDA. The LDA model was run each time with a different number of topics. Iteration takes place for ten times with ten number of topics. Each time the model is saved into a list. The top five models with the highest coherence value is extracted. Finally, we aggregate the coherence to see this model's performance. But this one performed poorly with an overall 0.27 coherence value. The major reason behind this is the lack of optimal number of topics. Since this model could not cover all the topics therefore the few extracted topics are not very coherent with each other and has overall a poor performance.

In the next experiment, we propose a new technique that is known as LDA iterative approach. In this experiment, we take the LDA model and optimize it with the correct number of topics according to our corpus through the topic optimization experiment described in the earlier section in this chapter. Once we have the optimized number of topics then we feed our model into a loop where we have set a threshold value of 0.97. So that means the iteration only stops when we start getting topics having a very high probability. LDA is a probabilistic model and each time different topics will occur as we run the algorithm for multiple times. It is important to know how good the topics are and also if they are human interpretable. The threshold value ranks the topics according to their quality. Once we get the best topics then the iteration stops. After that, we get the coherence value from this model which significantly outperforms the other ones. It achieves 0.82 coherence which is the best of all and can be used as any application for news classification. The topics are already described in the earlier section. The proposed model with the optimized number of topics works best for the Bangla news corpus.

4.7 Methodology for Classifying News Category

We have gone through a document classification according to topics. As we have a trained LDA model and also the topics, we wanted to go further with the first time news classification in Bangla language. A method is proposed for classifying news with the LDA model. At first the Document vs Topic matrix is extracted. In this matrix each of the term is tagged with a probability to belong to a topic. The idea can be illustrated with a simple example. Let assume we have a document D = **Dogs and cats are cute** and eventually become $D_{preprocessed}$ = [dogs, cats, cute]. As a human interpreter we can easily understand that this is a document with a topic - **Animal**. We may also consider that we have a topic k_1 and k_2. However the matrix for document vs term probability distribution can look like this:

$$\begin{Vmatrix} [(0, p1) & (0, p2)] \\ [(1, p3) & (1, p4)] \\ [(2, p5) & (2, p6)] \end{Vmatrix}$$

Where $p_1, p_2, ..., p_6$ are all the probabilities. 0, 1 and 2 are the word index from our example sentence. For each word

$$\sum(p_1 + p_2) = \sum(p_3 + p_4) = \sum(p_5 + p_6) = 1$$

So, in our proposed method we extract the mean of topic probability for each term. To make the example more generic let assume we have n terms and m number of topics. So our matrix is as follows:

$$\left\|\begin{matrix} [(w1, p1) & (w1, p2) & .. & (w1, p3)] \\ [(w2, p4) & (w2, p5) & . & (w2, p6)] \\ . & & & \\ . & & & \\ . & & & \\ . & & & \\ . & & & \\ [(w_n, p_{m-2}) & (w_n, p_{m-1}) & . & (w_n, p_m)] \end{matrix}\right\|$$

So the mean topic for each each word,

$$\begin{aligned} mean_1 &= \sum (p_1 ... p_3)/m \\ mean_2 &= \sum (p_4 ... p_6)/m \\ mean_m &= \sum (p_{m_{-2}} ... p_m)/m \end{aligned}$$

Finally, the document belongs to the mean score having the largest value.

The experiment is performed with two different type of news and explored how this method works to find out the best topic that fits the data. Here is the result for these two news articles:

In Figure 4.12 we have used a document about a movie and this is how the topic distribution looks like. Topic 35 is the most relevant topic for this news outperforming the other topics in terms of their probability score. Now we will take a look at how this topic looks like with the English translation:

(35, '0.026*"Cinema" + 0.020*"Premier" + 0.020*"Director" + 0.019*"Announce" + 0.017*"Ending" + 0.016*"Actress" + 0.015*"Next_movie" + 0.015*"Release" + 0.014*"The_movie" + 0.012*"Sensor"')

Since the news is about a movie, we get a very relevant answer from our model. It gives us a topic consisting about words related to movies. The first words is "Cinema" with the highest probability.

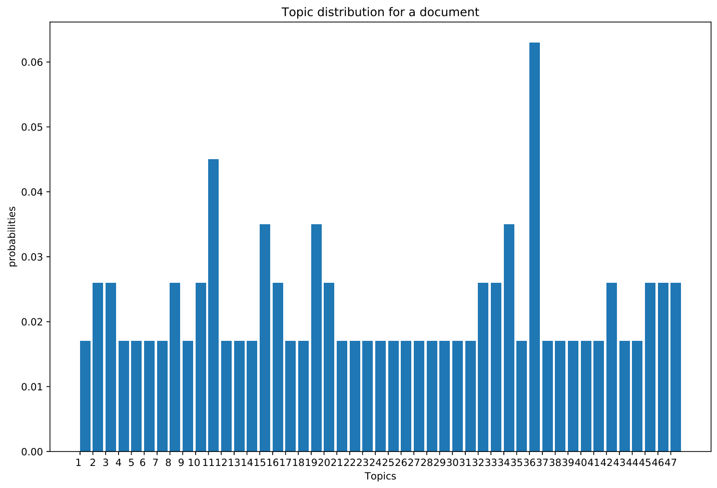


FIGURE 4.12: Document topic distribution for movie news

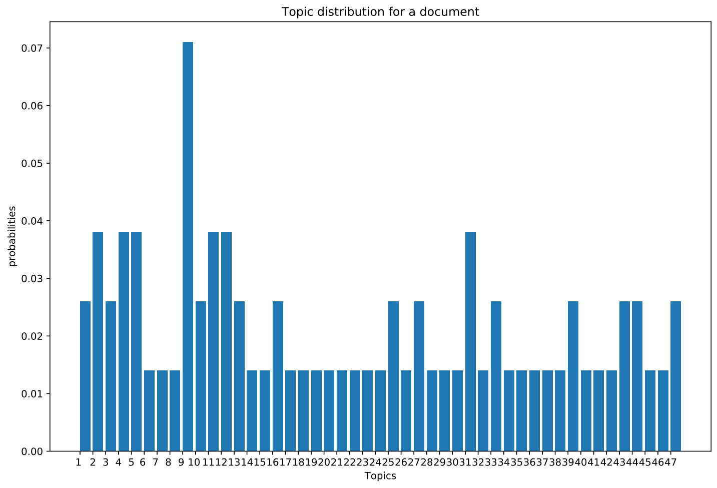


FIGURE 4.13: Document topic distribution for Trump news

(8, '0.052*"Trump" + 0.044*"President" + 0.025*"USA" + 0.023*"Against" + 0.020*"Administration" + 0.019*"Immigration" + 0.019*"Complaint" + 0.017*"Foreign_affairs" + 0.016*"Order" + 0.016*"Direct"')

In the second experiment we feed a news about Donald Trump talking about USA and immigrants facts illustrated in Figure 4.13. Then the model predicts it to be more likely to belong to this above topic consisting the word "Trump" itself with the highest probability and leading to all the other immigration and the USA related terms. So the experiment reflects the fact that we can classify Bangla news successfully with this model.

Chapter 5

Conclusion and Discussion

In this research, the scopes of the Bangla language in terms of the topic modelling and sentiment analysis have been demonstrated. Models are proposed for two different tasks that has never been applied on Bangla. Sentiment analysis and topic modelling with Bangla are both comparatively new in terms of the techniques and algorithms that have been applied in this thesis. News classification on Bangla has a greater impact on the print and news media. Bangladesh being a country of people who love internet and most of them using Bangla in the virtual world, the amount of data in Bangla text is ever growing. Over the past decade, this amount has grown dramatically. Starting from the Facebook to Twitter to other social sites there are various online businesses going on in Bangladesh. These businesses are mostly dependent on Facebook. They have Facebook pages and groups. People comment and share their opinions on these pages. The best way to keep up with a good business is to always understand and value customer's opinion. For this, sentiment analysis is important.

The state of the art NLP technologies when used in proper ways, insights of millions of various brand conversations with accuracy close to human level can be achieved. The proposed models will help to automate the topic understanding of a data collection as well as understand the sentiments. Otherwise, reading each of the document manually and understanding their semantic meanings will take forever for humans when it comes to large dataset. Eventually, Bangla textual data will turn into a big data as the amount is growing rapidly in the world wide web. So, the necessity of automation and deployment of these state of the art algorithms for Bangla language can not be ignored. This research took the lead to explore and demonstrate how these algorithms and the proposed models can be effectively used for both research and industry applications.

The first contribution of this thesis is the sentiment analysis with the Bangla language and the results are better than the existing models. The second contribution of the thesis is the first ever topic modelling with Bangla for news classification using the LDA algorithm. Topic modelling has a broad opportunity in terms of Bangla language. Bangla language is still lagging far behind due to it's scarce resources. From the implementation point of view many challenges exist for Bangla since there are no established libraries for the language. However, a model is proposed in this thesis regarding topic modelling and a comparison of the model with the other algorithms is also demonstrated. It is demonstrated that the proposed model built with the LDA algorithm works better than the other models. A news classification system with Bangla is demonstrated which has never been explored from the LDA perspective. Document similarity is also experimented with the various algorithms and compared the models. Topic modelling with Bangla has a various application starting from the news companies to the online markets etc.

In this research it is demonstrated that the topic modelling can be extended with the Bangla language. Bangla having a strong online presence over the past few years, there is a lot more to do with the language regarding topic modelling and other NLP tasks. Online Bangla libraries can use this tool as a recommender system. This work can also be extended with a trending-topic task. Topic trends is an important field in NLP which can be applied in Bangla language since there are tons of data being generated every day and these big amount of dataset can be analyzed in order to get an insight of the economy of Bangladesh which has recently been considered as the "Rising Tiger" of the south east Asia. The ready made garment (RMG) industry generates a big amount of textual data which can be used for topic modelling as well as topic trend analysis. The RMG sector is the main powerhouse of Bangladesh's economy. It is yet to be explored for the topic modelling task. Trending topics can play a vital role in predicting the corruption rate over different districts in Bangladesh by applying LDA to the local news papers from different districts and focusing on crime news. Moreover, it can be stretched to use in public sentiment analysis for prediction over diverse aspects through the print and news media.

From the technical points of view, Gensim, a python library is used for the topic modelling. Gensim is a powerful tool for any topic modelling task. The main challenge for this research was to achieve the dataset due to the lack of publicly available data. So, I crawled all the data from the online newspapers and organized them which was time consuming. Countrys largest online newspaper the daily

"Prothom Alo" was scrapped and organized the data chronologically. An extra layer of cleaning is applied to the dataset to remove some insignificant words those were not removed during the stop words removal stage. Some of the extreme rare words were considered as the stop words since they had no effect on the result. Significantly frequent words were removed too. Stemming can further be applied although LDA algorithm can extract the word to word semantic relationship for which it is not necessary for topic modelling. Bigram is used for this thesis. In the future trigrams and other n-grams can also be applied to see the effect on the results.

The other contribution of the thesis is the sentiment analysis with a RNN. Apart from a character level model, a word level RNN was proposed in the respective chapter. A comparison of these two results were shown. How the character level encoding outperforms the word level model is also discussed. In this model there are scopes to address the sentence scoring methods. There are different ways to develop a sentence scoring system and also apply the LDA algorithm for sentiment analysis. Identifying frequent words and their relation to the sentences may lead to better results in terms of sentiment analysis.

To conclude, this thesis offers a research based study on a character-level RNN for sentiment analysis in Bangla. However, the model is not a generic of it's kind since it worked well with data from a specific domain. Also, sarcastic sentence detection is not addressed in the model. So, if a positive word is used in the sentence with a negative sarcastic meaning, the model will not be able to detect this. Hence, this needs to be addressed which is a challenge due to the level of abstraction an user can create through one sentence. Intensive research is needed in this regard. Analysis from this thesis shows that the character-level RNN is an effective method to extract the sentiment from Bangla. Making the model more reliable across different data is one future goal of this thesis that will make it useable in the industries.

Bibliography

[1] Shaika Chowdhury and Wasifa Chowdhury. Performing sentiment analysis in Bangla microblog posts. In *proc. of International Conference on Informatics, Electronics & Vision (ICIEV)*, pages 1–6. IEEE, 2014.

[2] Rubayyi Alghamdi and Khalid Alfalqi. A survey of topic modeling in text mining. *International Journal of Advanced Computer Science and Applications. (IJACSA)*, Volume 6, 2015.

[3] Zhou Tong and Haiyi Zhang. A text mining research based on LDA topic modelling. *Technical report, Jodrey School of Computer Science, Acadia University, Wolfville, NS, Canada*, 10:201–210, 2016.

[4] Mhamud Murad. Dhaka ranked second in number of active facebook users, April, 2017. URL https://bdnews24.com/bangladesh/2017/04/15/dhaka-ranked-second-in-number-of-active-facebook-users.

[5] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *proc. of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. Association for Computational Linguistics, 2010.

[6] Shiry Ginosar and Avital Steinitz. Sentiment Analysis using Naive Bayes. *Technical report, University of California, Berkeley*, 2012. URL http://bid.berkeley.edu/cs294-1-spring12/images/4/4b/AvitalSteinitz_ShiryGinosar.pdf.

[7] Asif Hassan, Mohammad Rashedul Amin, Abul Kalam Al Azad, and Nabeel Mohammed. Sentiment analysis on Bangla and romanized Bangla text using deep recurrent models. In *proc. of International Workshop on Computational Intelligence (IWCI)*, pages 51–56. IEEE, 2016.

[8] Amitava Das and Sivaji Bandyopadhyay. Phrase-level polarity identification for bangla. *International Journal of Computational Linguistics and Applicatons (IJCLA)*, 1(1-2):169–182, 2010.

[9] Amitava Das and Sivaji Bandyopadhyay. Sentiwordnet for bangla. *Knowledge Sharing Event-4: Task*, 2, 2010.

[10] Sanjida Akter and Muhammad Tareq Aziz. Sentiment analysis on Facebook group using lexicon based approach. In *proc. of International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, pages 1–4. IEEE, 2016.

[11] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *proc. of Advances in neural information processing systems*, pages 649–657. Curran Associates, Inc., 2015.

[12] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[14] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[15] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[16] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003.

[19] David M. Blei and John D. Lafferty. Topic models. *Text mining: Classification, Clustering and Applications*, 10(71):34–57, 2009.

[20] ASM Ashique Mahmood. Literature survey on topic modeling. *Technical report, Dept. of CIS, University of Delaware Newark, Delaware*, 2009.

[21] Sheikh Abujar, Mahmudul Hasan, MSI Shahin, and Syed Akhter Hossain. A Heuristic Approach of Text Summarization for Bengali Documentation. In *proc. of 8th International Conference on Computing, Communication and Networking (8th ICCCNT)*, pages 1–7. IEEE, 2017.

[22] Shivam Bansal, Sunil Ray, Pranav Dar, and Tavish Srivastava. Beginners guide to topic modeling in python, Aug 2016. URL `https://www.analyticsvidhya.com/blog/2016/08/beginners-guide-to-topic-modeling-in-python/`.

[23] Leticia H. Anaya. *Comparing Latent Dirichlet Allocation and Latent Semantic Analysis as Classifiers.* ERIC, 2011.

[24] Sonia Bergamaschi and Laura Po. Comparing LDA and LSA Topic Models for Content-Based Movie Recommendation Systems. In *proc. of International Conference on Web Information Systems and Technologies*, pages 247–263. Springer, 2014.

[25] Yee W. Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Sharing clusters among related groups: Hierarchical Dirichlet processes. In *proc. of Advances in neural information processing systems*, pages 1385–1392, 2005.

[26] Michael E. Wall, Andreas Rechtsteiner, and Luis M. Rocha. Singular value decomposition and principal component analysis. In *proc. of A practical approach to microarray data analysis*, pages 91–109. Springer, 2003.

[27] Steven P. Crain, Ke Zhou, Shuang-Hong Yang, and Hongyuan Zha. Dimensionality reduction and topic modeling: From latent semantic indexing to latent dirichlet allocation and beyond. In *Mining Text Data*, pages 129–161. Springer, 2012.

[28] News classification with topic models in gensim, June, 2016. URL `https://markroxor.github.io/gensim/static/notebooks/gensim_news_classification.html#topic=0&lambda=1&term=`.

[29] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-Graber, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *proc. of Advances in neural information processing systems*, pages 288–296, 2009.

[30] David Mimno, Hanna M. Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In *proc. of the conference on empirical methods in natural language processing*, pages 262–272. Association for Computational Linguistics, 2011.

[31] Denny Britz. Recurrent neural networks tutorial, part 1 introduction to rnns, Jul 2016. URL `http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns`.

[32] Yan and Shi. Understanding lstm and its diagramsml reviewmedium, Mar 2016. URL `https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714`.

[33] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[34] Jesse Weaver and Paul Tarjan. Facebook linked data via the graph api. *Semantic Web*, 4(issue 3):245–250, 2013.

[35] Boris Kovalerchuk. Interactive visual classification, clustering and dimension reduction with glc-l. In *Visual Knowledge Discovery and Machine Learning*, pages 173–216. Springer, 2018.

[36] Tokenization, April, 2009. URL `https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html`.

[37] Michael John Collins. A new statistical parser based on bigram lexical dependencies. In *proc. of the 34th annual meeting on Association for Computational Linguistics*, pages 184–191. Association for Computational Linguistics, 1996.