An Adaptive Ensemble Machine Learning Model for Text Classification

By

ASM Shad, 011172167
MD Shayed Hasan, 011172169
Niamul Chowdhury, 011172173
Md. Alwadood Ripon, 011172180
Md. Nahid Hasan Foud, 011172181

Submitted in partial fulfilment of the requirements
of the degree of Bachelor of Science in Computer Science and Engineering

July 23, 2021

Department of Computer Science and Engineering
United International University

Abstract

In this article we depict our endeavor at creating a state-of-the-art classifier utilizing Convolutional Neural Networks (CNNs), Long Short Term Memory (LSTMs) networks and GRU. Our framework uses a lot of unlabeled information to pre-train word embeddings. We at that point utilize a subset of the unlabeled information to adjust the embeddings utilizing distant supervision. The last CNNs, LSTMs and GRU are prepared on a dataset comprises of 8 distinct classes where the embeddings are fined tuned once more. To boost performances we gather a few CNNs and LSTMs together.

Table of Contents

Chapter 1

Introduction

In this part, first, we will communicate our anxiety. By then we will push the issue with respect to certifiable conditions. Later we will present our goal and targets. We will wrap up this segment by explaining our technique quickly.

1.1 Problem Statement

Text based reports have consistently been a significant segment of extensively implied business measures. These days, increasingly more of these records are accessible in the electronic structure. There is huge measure of unlabeled data accessible on the web. Monitoring this unlabeled writings is so troublesome. Text classification is a technique that gives a general view and designs the offer. The capacity to characterize unlabeled reports would prompt simpler access for those doing explore in a particular territory. Indeed, even web indexes like Google could return better outcomes if data was better coordinated.

1.2 Motivation

There are billions of sites with endless writings on the Internet. This makes it hard to monitor them. The measure of data on the Internet is huge to the point that sifting by human specialists alone is difficult to imagine. The more data is spread on the Internet, for the most part in content structure, the more prominent the requirement for machine investigation, arranging and grouping. Models:

- News portals select their news as per branches of knowledge and different highlights. A person ideally settles on an official choice with regards to whether and where a source ought to be set in a gateway – yet man-made reasoning can likewise play out this assignment precisely.
- Vertical search engines just catch connects to a particular point – as opposed to

general web crawlers, for example, Google or Bing. Vertical search engines promote with the bit of leeway that they make it simpler for intrigued clients to discover important data quicker. This is on the grounds that the file is restricted from the start to theme explicit substance.

- Email supplier needs effective methods to separate between authentic messages and spam sends utilizing different rules. These standards incorporate the sender, yet in addition the actual content. Spam sends are portrayed by normal semantic attributes.

- Sentiment analyses are utilized for statistical surveying. These calculations are utilized to consequently identify positive or negative mentalities – for instance with respect to specific items or continuous missions. Machine upholds is a powerful guide for ordering writings. Man-made consciousness assumes an inexorably significant part here.

1.3 Goal & Objectives

Our goal is to arrange unlabeled content documentation and allot suitable class or classes to them. We have a few objectives to accomplish our goal.

- Distinguishing the unlabeled substance documentation from taught media or physically giving the substance to the model interface.

- Breaking down the content archive dependent on procedures like Tokenization, Part-of-speech Tagging, Parsing, Lemmatization and Stemming, Stopword Removal.

- Classifying the documentation depend on substance investigation.

1.4 Brief Methodology

Consistently a large number of clients look for different text substance over web. There is loads of unlabeled text substance in web which are truly elusive. Our anxiety is to arrange those unlabeled text substance. Our proposed methodology is shown in the following figure 1.1

From the outset, we scratch the unlabeled content information from web by utilizing an outsider information extraction instrument and utilize this scratched data as our dataset for the framework. We will pre-measure these information and feed them into our proposed AI model. The model will dissect the information. Later our model will arrange those settings as indicated by their classes. In conclusion, in light of past learning, our model will anticipate potential classes of unlabeled content information.

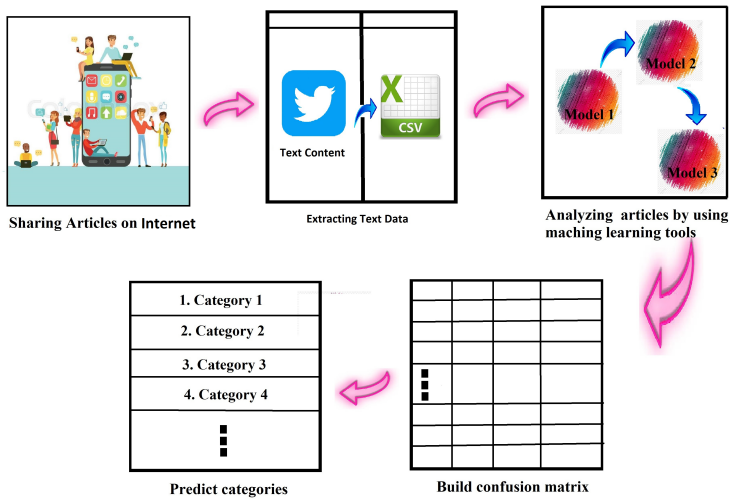

Figure 1.1: Methodology of Our System.

1.5 Organization of Report Summary

All the data, foundation examines introduced in this paper is separated into 6 chapters including the Introduction. The venture report is coordinated as follows:

- **Chapter 1** We present our problem statement and motivation for the venture in this chapter. Later we talk about our objective and goals with brief methodology discussion.
- **Chapter 2** We show the foundation of the project with literature review. In the literature review, we examine comparative applications, insights concerning related work, and their correlation.
- **Chapter 3** We present the plan of our Context Diagram and DFD-1 Diagram.
- **Chapter 4** We discuss our process of dataset collection and preparation. Then we discuss about the methodology, implementations and the outcome of our project.
- **Chapter 5** We exhibit the designing principles we are following and the plan limitations of our task.
- **Chapter 6** We present the conclusions, rundowns of the work we have done so far, furthermore, talks about future works.

Chapter 2

Background

In this section, we will talk about our experience study. To start with, we will examine comparative application and their rundown. From that point onward, we will examine related works and their outline.

2.1 Introduction

We have perused almost 20 research papers so far as to comprehend the momentum data, past investigations, important accounts encompassing our examination theme. While perusing those applicable papers, we have accumulated some significant data concerning datasets, the various procedures they have utilized, distinctive assessment lattices. Close by our experience contemplates; we have additionally investigated the comparative applications accessible on the lookout. However, there could be no other accessible applications out there who are offering the types of assistance we need to give. Along these lines, in this section, we present the examinations about some other programming that are utilized for text classification.

2.2 Literature Review

In this part, we present the writing audit regarding comparative applications and related papers in the literature.

2.2.1 Similar Applications

In this segment, we present data about some data extraction apparatuses, as they are identified with our task.

- MonkeyLearn:
  MonkeyLearn [1] is an easy to use AI stage that allows us to jump into text classification immediately utilizing pre-trained models, this way sentiment classifier. We can

[1] `https://monkeylearn.com`

likewise fabricate our own tweaked answers for more exact bits of knowledge.
Custom models are extraordinary in the event that we need to build exactness, particularly on the off chance that we manage area explicit information. In only a couple steps (no coding or AI abilities required), we can prepare a model utilizing our own rules to consequently identify subjects, estimation, and purpose.
We can deal with our text classification models through the MonkeyLearn API, which is not difficult to utilize and accessible in all significant programming dialects. Or on the other hand, we can get to mixes and associate our information to ordinary applications like Google Sheets, Zapier, and Zendesk.

- Google Cloud NLP:
  Google Cloud NLP [2] is a set-up of text analysis devices to help us discover experiences in unstructured contents.
  Utilizing the Natural Language API, we can interface with amazing pre-trained models, intended to convey conventional outcomes with high exactness for sentiment analysis and content arrangement. The content arrangement device, for instance, permits us to classify documents into in excess of 700 classes.
  On the off chance that we need a characterization model custom-made to a particular use case, we can utilize AutoML Natural Language, which permits us to fabricate altered arrangements utilizing our own pre-defined classifications.

- Aylien:
  Aylien [3] is an AI stage that offers diverse text analysis answers for organizations. With the text classification API, we can naturally discover subjects in records or sites.
  The API applies a couple of scientific categorizations: one that utilizes 500 classes to label news and media content, and another that is more centered around promoting and permits organizations to show online advertisements in the correct spots.

- IBM Watson:
  IBM Watson [4] is a multi-cloud stage with a variety of AI devices to group business information. With the Natural Language Classifier, engineers can assemble custom order models to discover themes in information. We can prepare a model in less than 15 minutes (no AI foundation required) and effectively coordinate models into our applications utilizing the API.
  Watson additionally gives a pre-constructed answer for text analysis called Natural Language Understanding, which you can use to discover notion, feelings, and classes in content.

- Meaning Cloud:

[2] `https://cloud.google.com/natural-language`
[3] `https://aylien.com`
[4] `https://www.ibm.com/watson`

Meaning Cloud [5] offers an assortment of cloud-based APIs for text analysis. The text classification API accompanies a progression of predefined classifications to naturally sort information (for instance, we can characterize news content into in excess of 1300 subjects), or we can make custom models utilizing your own classes.
The sentiment analysis API, then again, causes us distinguish extremity and incongruity marks in content across various dialects. We can modify this arrangement too, characterizing our own word references adjusted to our area.

- Lexalytics:
  Lexalytics [6] is a particular business insight stage, including various answers for text analysis. The Semantria API permits us to perform archive arrangement and sentiment analysis utilizing Natural Language Processing and AI.
  The models uphold text in various dialects and are exceptionally adjustable: we can without much of a stretch train them to perceive industry-explicit jargon and change various boundaries to get better outcomes.

- Amazon Comprehend:
  Amazon Comprehend [7] is a NLP administration with a variety of APIs we can undoubtedly coordinate into our applications. We'll discover APIs for sentiment analysis, language detection, and a custom characterization API, that can help us construct text classification models adjusted to our business needs. We needn't bother with any AI information or broad coding abilities to make a custom model.

2.2.2 Related Work

In this segment, we present a few papers that are identified with our work.

Neethu et al. [1] tried to work with online product analysis from tweets. Electronic media is making a huge proportion of speculation data as tweets, declarations, blog passages, etc. They try to analyze the posts of twitter about electronic products like cameras, computer parts, mobile phones etc. using Machine Learning approach. They discover a new feature for classifying the tweets which are positive and which are negative about products. They made a dataset by taking 600 negatives and 600 positives tweets utilizing Twitter posts of electronic items by gathering tweets throughout some stretch of time going from April 2013 to May 2013. There are a huge amount of misspelling and slang words on twitter. For that reason, keyword extraction is a little bit difficult in their model. To avoid this difficulty, they used a preprocessing step performed before feature extraction. Slang Words can't be simply removed. Thusly, they kept a slang word reference to supplant slang words happening in tweets with their importance. They utilized three sorts of essential classifiers (SVM, Nave Bayes, Maximum Entropy) and troupe classifiers for slant

[5] https://www.meaningcloud.com
[6] https://www.lexalytics.com/semantria
[7] https://aws.amazon.com/comprehend

order. They implemented SVM and Naive Bayes classifiers using MATLAB built functions. Greatest Entropy classifier is actualized utilizing Maxent software. In their graph it shows that all the classifiers have almost similar performance. They likewise guarantee that Naive Bayes has better exactness contrasted with the other three classifiers, yet marginally lower precision and review. SVM, Ensemble classifiers, and Maximum Entropy classifiers have comparable exactness, accuracy, and review. They got an exactness of 90& while Naive Bayes has 89.5&. Despite the fact that there are a ton of Symbolic and Machine Learning procedures used to recognize estimations from the content, they said that these strategies can be applied for Twitter feeling investigation. Portrayal exactness of the component vector is taken a stab at using different classifiers like Nave Bayes, SVM, Maximum Entropy, and Ensemble classifiers. Every one of these classifiers has practically comparative precision for the new component vector.

Bahrainian et al. [2] propose a method called "A novel hybrid method" which introduces a novel solution to Sentimental Analysis of short informal texts with a main focus on social media posts like Tweets. They compare Their proposed method against state-of-the-art SA methods. They claim that their hybrid method beats the state-of-the-art unigram baseline. There generally have formal and informal content on Twitter messages as numerous others posted on the blogosphere. The data processing is the primary distinction between handling formal and casual content. Formal text regularly needs less preprocessing than informal text. Informal text always contains slang words, emotions, bad grammar or non-dictionary-standard words etc. Their fundamental objective is preparing informal text explanations. They train the most classic and popular classifiers Naïve Bayes, SVM, and Maximum Entropy-dependent on the unigram feature set. They also compare the classifiers against their proposed novel hybrid method in a benchmark. Their test results show that their method beats all other referenced methods that mentioned before. Their dataset comprises of 940 tweets marked by a gathering of 22 human annotators. Where 470 tweets have a negative polarity and other 470 tweets have a positive polarity. Stop words in the unigram include set to test and train Support Vector Machine, NB, just as MaxEnt classifiers and they test the impact of nonattendance or attendance of emotions.

They defined the SVM unigram model has an overall accuracy of 86& and their proposed method against SVM took an accuracy of 89&. so, it is clarified that their approached method beats the unigram baseline by a huge difference.

Riyanarto et al. [3] propose a text classification method to predict a human's personality based on his text written by Twitter users. They use two languages English and Indonesian. They implemented three classifications methods NB (Naive Bayes), K-NN and SVM methods. The system will recover a group of tweets from many users. Retrieve the text from the user then preprocessed the text into a vector data. The user's text will be classified into a labeled dataset by Classification process. They use MyPersonality dataset consists of 10.000 status updates from 250 twitter users. There are 250 twitter

users having final dataset in the form of 250 documents. text data will be addressed in vector space model in text classification. They eliminate the stop words which have little or no meaning. Then they calculate Tf-Idf for each word. The event limits the quantity of words to lessen the processing time and heap, increment viability, and improve precision. In the cross-validation testing. Support Vector Machine and K-Nearest Neighbor similarly performed. Support Vector Machine performs worse due to difficulties separating a class of a word. K-Nearest Neighbor method also performs worse. Combined all the method got the best accuracy results on respondent testing with an accuracy of 65%.

Li et al. [4] said that With the rapid development of the Internet, as represented by the microblog social networks being used by a growing number of users, users in the social network to express their views or express their feelings. The traditional rules of text, for microblogging sentiment analysis task more difficult. The current text sentiment analysis of research methods mainly divided into the use of machine learning classification and the classification method based on rules. In combination with other classifiers to complete the task of sentiment classification, commonly used methods are Naïve Bayes, Support Vector Machines, and the Maximum Entropy. Experiments were carried out using microblog data. The raw dataset is extracted from the COAE2014. There are 10,000 data are annotated, which has 2740 with sentiment color, including 1608 with negative sentiment and 1132 with positive sentiment. In this paper, they proposed to use dependency parsing with sentiment relationship migration and modified distance for sentiment analysis of short text in microblogs. In future work, they would analysis the implied sentiment of short text without emotion words.

Li et al. [5] says, in this paper they propose a new approach, which is incorporates sentiment-specific word embeddings (SSWE) and a weighted text feature model (WTFM) and WTFM produces features, based on text negation, tf.idf weighting scheme, and a Rocchio text classification method. They used WTFM because WTFM is easy to build, simple and effective compare to other tweet sentiment analysis. They said, in this paper they are try to classify a tweet's sentiment polarity into three types: positive, neutral and negative. Before training and testing they preprocess their data by removing all URLs and mentions, dates are converted to symbol, all ratios are replaced by a special symbol etc. Mainly they collect their data from latest tweet sentiment analysis benchmark data set, which is from task 9 of SemEval 2014. Three different work they had done: training, development and test. They compare their propose model with Word2vec, Worf2vec+WTFM, SSWE+Word2vec, NRC, SSWE+NRC and most of this models their proposed model performed better.

Kisan et al. [6] propose that they are defining the system which gives the sentiment ib tweets by users and it's helpful to understand the opinion is the users. They also research related works which is done before. They collect their data from twitter and collecting

data from twitter they are use Twitter4j. They also fetch data from public hashtags and hashtags provides short form of a term like re-tweets as RT. They also preprocess their data. They proposed how they can use the StanfordcoreNLP libraries and twitter4j to build a application which can fetch from and performs the sentimental analysis and classify positive or negative tweets are there in any particular issue by using hashtags.

Kavitha et al. [7] propose to design and implement efficient methods for discovering events and analyze the public opinion from the real time twitter data using sentiment analysis approach. Because, In this present era, streaming data tends to collect data from live streaming to run analysis and generate reports for data prediction and this process requires skilled professional for acquiring data from live stream using complex coding and queries. The above drawback is overcome in this paper. They also discuss about some related work like: early detection of burst topics in Twitter by designing sketch based topic model, A scalable distributed event detection approach to effectively detect events in social streams etc.. This paper they proposed to extract the tweets in large scale from Twitter and then apply the sentiment analysis approach to discover public opinions for an application of any domain. The live twitter data is fetched by configuring the system with Hadoop, Hive warehouse and Apache Flume. First Hadoop is start and then Hive is start and then run the Flume amd after that register or lohin the client. The system architecture presented an overview of the real time twitter streaming process which analyzes the tweets from any source. They have used data, it's basically related to IPL and it's have been fetched to discover the public opinion and rating about IPL matches and its players. And in this paper they classify the tweets data three types: positive, negative and neutral.

Adarsh et al. [8] use Python and R tools for downloading Tweets and they are preprocessing the data for future usages in this work. Analyzing the sentiments using text analysis and Natural Language processing (NLP) methods, it's called sentiment analysis. Sentiment analysis also referred as Opinion Mining which is acclaimed approach it knows the sentiments of users and clients. Sentiment analysis also widely used in many sectors. They say, developers can import files containing a dictionary of positive and Negative words and then it compares the words to Dictionaries of positive and negative words. Then the model calculates the score by follows: if score is greater than 0 then positive, if score is less than 0 then negative and if score=0 then it's neutral.

Rathi et al. [9] talked about the diversity of social networking data and how it's always getting bigger in size. Among them they have used Twitter data one of the most renowned social network site. Assessments were gathered into three classes useful for positive, awful for negative and impartial for neutral. By using NLP and computational insight they have arranged tweets as positive, awful or unbiased. The principle emphasis of their research is on the grouping of feelings of tweets' information assembled from Twitter. As previous systems were not providing better results of sentiment classification they have

used ensemble machine learning approach. They have combined Support Vector Machine with Decision Tree and trial results demonstrate that their proposed approach is giving better arrangement brings about terms of f-measure and precision rather than singular classifiers.

Ranganathan et al. [10] propose a new optimized and more promising system, in terms of speed and efficiency, for generating meta-actions by implementing Specific Action Rule discovery based on Grabbing strategy (SARGS) algorithm. Action Rules are one of the major data mining method for gathering information from datasets. According to authors the main essence of this paper was to. In Pre-processing phase, we perform discretization on the following attributes, friends count and followers count by placing their values into intervals. The primary focus of their work is to evaluate the proposed Spark driven system implementing the Action Rule mining algorithm on twitter data, for making the users more positive, against the existing MapReduce driven system. They performed corpus based Sentimental Analysis of social networking data, and test the complete time taken by both the systems and their subcomponents for the data processing. Finally, results show quicker computational time for Spark system compared to Hadoop MapReduce for implementing the meta-action generation methods.

Kumar et al. [11] thought that our perspectives, feelings and Sentiments about others and global communication has been changed by the development of Technology of World Wide Web. These people spend enormous time on social network sharing their information all the time which leads to a huge data. This paper uses the nostalgic investigation of Twitter information utilizing R language which is useful for gathering the estimations data as either positive score, negative score or some place in the middle of them. They perform the analysis of tweets data that are having a size of TBs implies big data using R language and Rhadoop Connector. To resolve the performance problem, they extract the information from petabytes of data and they focused on the analysis of big data. Finally, this paper shows the performance estimation on two different platforms R language and Rhadoop tool.

Tabassum et al. [12] propose a classifier method that quantifies total positivity and negativity against a document or sentence. According to them sentimental Analysis is an application of NLP which deals information to look at the sentiment or assessment that can be either positive or negative. They perform the analysis on Bangla text data. They faced difficulty as there was a few previous works in this sector. This paper quantifies total positivity and negativity against a document or sentence using Random Forest Classifier to classify sentiments. As for preprocessing they followed some steps data normalization, data tokenizer and POS tagging. Feature extractor generates all unigrams that express not many information about the linguistic pattern of sentiments. Parts of speech features are also generated by first tokenizing the normalized sentences. The effectiveness of their system is evaluated using the following performance measurement techniques Confusion Matrix, Precision & Recall, F1-score, Accuracy & Error-rate. They propose combination

of unigram, POS tagging, negation handling and random forest classifier to provide more accurate result. As a future work, they plan to deal with emoticons and vast data set and also to demonstrate the neutral sentiment and compound sentences.

Khan et al. [13] propose a model to find the sentiment from the Bengali paragraph. Their system specifies is the paragraph happy or sad using various types of machine learning classification analysis algorithms. They faced some difficulties while during Bengali text processing. Their mentioned system predicts and measures the sentiment of Bengali text which collects from the Facebook social network site. Here we use six different types of machine learning classification algorithms to find the best accuracy and to detect two types of classification as Happy and Sad. They tokenized the data by using Count vectorizer. They build their dataset manually and that's why the size of their dataset is not so large. Writing English or other language words in Bengali letters is a major problem in their system as those words cannot generate the proper meaning in Bengali.

Chapter 3

Project Design

In this part, we exhibit our framework configuration identified with our undertaking. We have utilized the UML diagram for designing.

3.1 UML Diagram

The outline of our framework is appeared underneath utilizing the Context Diagram [3.1]

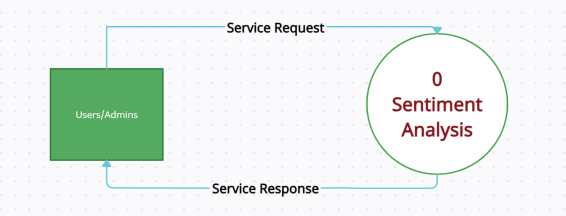

Figure 3.1: Context diagram

The Data-flow Diagram of our framework is appeared beneath utilizing the DFD level 1 [3.2]

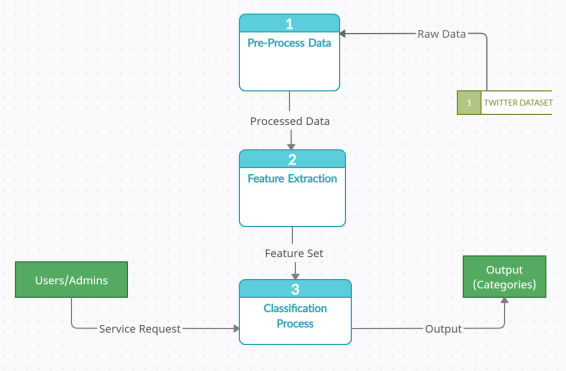

Figure 3.2: DFD level-1

3.2 UI Prototype

The UI Prototype of our framework is outlined in figures [3.3], [3.4]

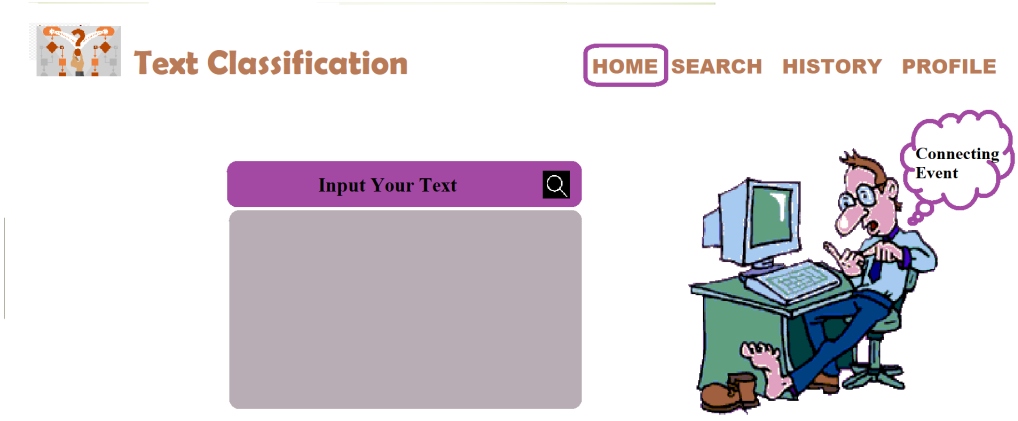

Figure 3.3: UI prototype page 1

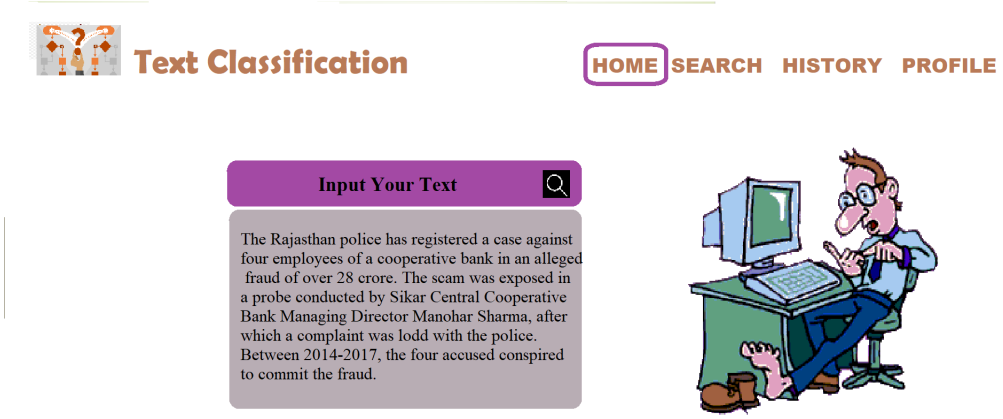

Figure 3.4: UI prototype page 2

Chapter 4

Implementation and Results

4.1 Dataset Preparation

In this Section, firstly, we present how we collect our dataset. Next, we demonstrate the guidelines for class annotation. Later, we explain our data pre-processing method.

4.1.1 Data Collection

Since we want to detect news classes, we searched for news articles in various websites and news portals.There are various news portals over the internet. But as we want to build a model, we decided to collect very standard news articles.We have chosen different renowned news pages as our data source just to ensure the quality of the news articles. The pages are BBC, CNN and Reuters etc. We have collected total 80,317 news articles for our dataset.

4.1.2 Data Annotation

There is several ways data annotation. To train our model, classifying each data is very important for training dataset. To classify the data, human annotators can easily detect the class of the news article.To prepare our training dataset, we used manual annotation process. While collecting the news articles, we fill-up the class column manually, so that we can get a proper training dataset to train our model.

4.1.3 Data Preprocessing

Noisy data may influence the exactness of our trial. However there isn't much futile data in a news article, yet to comprehend the class of that article we don't actually need such large articles. Commotions like exceptional characters, accentuation marks, joins, don't convey any significance in identifying classes. To get expected execution on our test we should manage such superfluous substance. For diminishing the size of data, reducing model training time, and sometimes for better precision, eliminating unique characters, punctuation marks, links, and so forth are important.We used NLTK library functions to

remove unnecessary parts of data from the dataset and to tokenize our dataset. In NLTK library there is a function called TweetTokenizer. Tokenization is the process of splitting a string into a list of tokens. In a paragraph, a sentence is a token. In a sentence, a word is a token. A token is a piece of a whole. To train our model data tokenization is important. We used TweetTokenizer to tokenize our dataset. Tough there is many other functions we could use as well, but in our case we find it convenient for our model.

4.2 Implementations

In this section of this chapter, we will introduce the coding environment of our project. Next, we manifest our method of building the training dataset. Later we explain three different Neural Network algorithm: CNN, LSTM and GRU. Additionally, we demonstrate the reasoning's for choosing various neural network algorithms.

4.2.1 Environment

The following information in the table (no) represents the languages, packages, libraries and tools that we have used during the implementation:

Software and Communications	Standards
Language	Python
Development Tool	Google Colab
Libraries	Tqdm, numpy, pandas, keras, nltk

Table 4.1: Software Standards

4.2.2 Building Training Dataset

For training the models we have used 64,253 news articles from various media sources, e.g. online news portals. Each article/news has:

- A title/headline; # Title is basically the focused words (headline) about the news/article.
- A unique ID; # for each news/article a unique ID is assigned
- Date; # the date and day of the news/article when it was published
- Classes; # each news/article is given a related predefined category e.g. political news is categorized as 'politics', game and sports related article is categorized as 'sports' and so on.

There are 8 classes in the category: 'business', 'sports', 'world', 'entertainment', 'technology', 'national', 'politics', 'science'

4.2.3 Background Models

CNN

Let us currently portray the design of the CNN we worked with. Its engineering is practically indistinguishable from the CNN of Kim (2014). The input of the network are the news events, which are tokenized into words. Each word is planned to a word vector portrayal, for example a word implanting, with the end goal that a whole event can be planned to a framework of size s × d,where s is the quantity of words in the event and d is the dimension of the embedding space (we picked d = 200). We follow Kim (2014) zero cushioning methodology with the end goal that all events have a similar framework measurement X ∈ Rs0 × d, where we picked s0 = 80. We at that point apply a few convolution activities of different sizes to this network. A solitary convolution includes a separating lattice w ∈ Rh × d where h is the size of the convolution, which means the quantity of words it ranges. The convolution activity is characterized as

$$c_i = f\left(\sum_{j,k} w_{j,k}\left(X_{[i:i+h-1]}\right)_{j,k} + b\right) \quad (1)$$

where b ∈ R is a bias term and f(x) is a nonlinear function, which we chose to be the relu function. We can utilize various sifting frameworks to learn various highlights, and moreover we can utilize numerous convolution sizes to focus on littler or bigger locales of the events. By and by, we utilized three channel sizes (either [1, 2, 3] or [3, 4, 5] relying upon the model) and we utilized a sum of 200 sifting frameworks for each channel size.

LSTM

Let us presently portray the design of the LSTM framework we worked with. Its primary structure blocks are two LSTM units. LSTMs are important for the RNN family, which are neural networks that are developed to manage successive information by sharing their inner loads over the grouping. For every component in the grouping, each word in the event, the RNN utilizes the current word embedding and its past concealed state to register the next hidden state.

GRU

We have used Gated Recurrent Units (GRUs) model in our system. GRUs are a gating instrument in Recurrent Neural Networks. The GRU resembles a LSTM with a forget door, yet has less boundaries than LSTM, as it comes up short on a yield entryway. GRUs have been appeared to display better execution on certain littler and less incessant datasets.

4.3 Model Building

In this section, we explain our experimental procedures.

First we utilize the dataset to pre-train the word embeddings which will later be utilized in the CNN and LSTM. To do as such, we explored 3 different algorithms, Google's Word2vec (Mikolov et al., 2013a,b), Facebook's FastText (Bojanowski et al., 2016) and Stanford's GloVe (Pennington et al., 2014). Word2vec learns word vector portrayals by endeavoring to foresee setting words around an input word. FastText is fundamentally the same as Word2vec yet it additionally utilizes subword data in the expectation model. GloVe then again is a model dependent on worldwide word-word co-event insights. For every one of the three algorithms we utilized the code given by the creators accompanying their default settings.

The embeddings learned in the unaided stage contain next to no data about the sentiment polarity of the words since the setting for a positive word will in general be fundamentally the same as the setting of a negative word. To add polarity info to the embeddings, we follow the unsupervised training by a calibrating of the embeddings by means of a distant training stage. To do as such, we utilize the CNN that trains for 8 epochs. After this stage, words with totally different conclusion polarity are far separated in the embedding space.

The last preparing stage is finished by instating the embeddings in the CNN, LSTM and GRU models with the tweaked embeddings of the distant training stage. To decrease difference and lift precision, we ensemble several CNNs, LSTMs and GRUs. The models ensemble have distinctive irregular weight introductions, diverse number of epochs (from 8 to 50 altogether), various arrangement of channel sizes (either [1, 2, 3] or [3, 4, 5]) and diverse embedding pre-training algorithms (either Word2vec or FastText).

- We have used 8 epochs and 512 batch size for CNN [1, 2, 3] model.
- Then we have used 30 epochs and 512 batch size for CNN [3, 4, 5] model.
- Further we have used 30 epochs and 512 batch size for Bi-LSTM model.
- Lastly, we have used 50 epochs and 1024 batch size for GRU model

After all these steps done, ensembling all these models is a powerful technique to boost the performance of our model. We used to ensemble deep learning models in Keras in the following:

- Load individual models
- Perform prediction using "model.predict"

- Average the predictions

As we want to build an ensemble model and store it as a single model so we deployed it in this process.

4.4 Results

Let us show the outcomes got from our framework. The outcomes are summed up in the following table. This table isn't intended to be an exhaustive list of all the experiments performed, but it does illustrate the relative performances of the most important variations on the models explored here.

Model	No of epochs	Batch size	Accuracy	Val Accuracy
CNN [1,2,3]	8	512	0.77	0.81
CNN [3,4,5]	30	512	0.87	0.80
Bi-LSTM	30	512	0.70	0.83
GRU	50	1024	0.87	0.84

Table 4.2: Performance measure among different models.

Among this models, the CNN[3, 4, 5] has shown significantly better performance than CNN[1, 2, 3] and Bi-LSTM model. We have achieved an accuracy of 87% with Val. Accuracy of 80% and with Batch size of 512. On the other hand GRU model also gives us the accuracy of 87% with Val. Accuracy of 84% and with 1024 Batch Size.

As we are building an ensemble model using all this models, we can define that CNN[3, 4, 5] is doing better among all this models with accuracy of 87%, batch size of 512 and with 30 epochs.

Chapter 5

Standards and Design Constraints

In this chapter, from the start, we present the standards we have been following for Software and Communication. Then, we show the various imperatives of our Project. Later we show a few difficulties we have in this project.

5.1 Compliance with the Standards

Just notice the principles that are identified with our undertaking. This rundown isn't finished.

5.1.1 Software Standard

In this part, we present some item standards we are following for our software.

- For System Design we are using UML Design utilizing the product named Draw.io. On the other hand we could utilize the Object-Oriented Design here.
- For Version Control we are using Github. Despite the fact that we could substitute it with Bit-Pails, SourceForge, Git, GitLab, yet because of the knowledge of Github use of our colleagues, we have focused on Github.
- For Project Management we are using Trello. In spite of the fact that we could utilize different stages like Slack, Asana, ClickUp, Wrike. These are basic apparatuses to put together errands and track progress. Trello has the element of transferring files directly from dropbox, google drive account that makes it simple to utilize. It's not difficult to coordinate assignments and track progress. It additionally gives adjustable records, computerized notifications to keep educated regarding all changes. That is the reason we have picked Trello.
- For Report Writing, we are using Overleaf, a cooperative cloud-based LaTeX editorial manager.
- For coding, we are utilizing Google Colab, Jupyter Notebook, Kaggle.

- For Programming Language, we are planning to utilize Python.
- AngularJS, Django, MongoDB for web designing.

5.2 Design Constraints

Design constraints are conditions that need to occur for a project to be effective. Design limitations help thin decisions while working on a project.
Design requirements can feel like something negative here and there; however they help shape the task to fit the specific necessities of the customer.
In this part, we present a few regions where our project impacts with its result.

5.2.1 Browser Constraints

Labeling substance or items utilizing classifications as an approach to improve perusing or to distinguish related substance on our site. Stages, for example, online business, news organizations, content guardians, sites, and catalogs can utilize mechanized advances to group and label substance and items.

Text categorizing of substance on the site utilizing labels assists Google with slithering our site effectively, which eventually helps in SEO. Furthermore, robotizing the substance labels on our site and application can make the client experience better and help normalize it. Another utilization case for advertisers is investigating and breaks down labels and catchphrases utilized by contenders. Text categorization can be utilized to mechanize and accelerate this interaction.

5.2.2 Marketing Constraints

As advertising is turning out to be more focused on consistently, mechanized characterization of clients into accomplices can make the advertiser's life simple. Advertisers can screen and arrange clients dependent on how they talk about an item or brand on the web. The classifier can be prepared to distinguish advertisers or naysayers, in this way assisting brands with serving accomplices better.

5.2.3 Social Constraints

A quicker crisis reaction framework can be made by characterizing alarming discussion via web-based media. Specialists can screen and arrange crisis circumstances to make a brisk reaction if any such circumstance emerges. This is an instance of extremely specific grouping.
The scholarly world, law professionals, social specialists, government, and non-profit organizations can likewise utilize text categorization innovation. As these associations manage

a ton of unstructured content, dealing with the information would be a lot simpler in the event that it was normalized by classes/labels.

5.2.4 Customer Service Constraints

Text classification can likewise be utilized to mechanize CRM assignments. The content classifier is exceptionally adaptable and can be prepared likewise. The CRM errands can straightforwardly be allocated and examined dependent on significance and importance. This diminishes manual work and along these lines is exceptionally time-productive.

5.3 Challenges

5.4 Budget Analysis

In this part we present approximate spending plan for our initial year design project. We have demonstrated the financial plan in table [5.1].

Design	Development	Deployment	Hardware	Maintenance	Descriptions	Price
Logo	-	-	-	-	-	2500 BDT
Web	-	-	-	-	-	10000 BDT
-	Web	-	-	-	-	12000 BDT
-	-	Domain	-	-	-	950 BDT/year
-	-	Hosting	-	-	-	1500 BDT/year
-	-	-	Laptop	-	-	60500 BDT
-	-	-	-	Software Tester	2 Pers., 2 Mos.	35000 BDT
-	-	-	-	System Admin	1 Pers., 12 Mos.	120000 BDT
					Total	24450 BDT

Table 5.1: Approximate budget analysis for initial year.

Chapter 6

Conclusion

In this chapter, we present a rundown of our task regarding our so far advancement and future work.

6.1 Summary

In this paper, we have proposed a web based text analysis system that will classify text news articles into eight different classes. We have gathered data from various well known online news portals such as BBC, CNN, Al jazeera etc. and we have pre-processed the data.We also created a guideline to classify the news articles for our training dataset. Later, we have implemented four neural network models; CNN[1,2,3], CNN[3,4,5], Bi-LSTM, GRU for news classification. After that we have ensembled all this models to get the result. Finally, we have automatically labeled the news articles classes.

6.2 Future Work

We hope to design a web application, where we can put a news article and that will suggest us the class of that news. We also will look forward to increase our models prediction accuracy. In future we will take a look on our run-time as well.

References

[1] MS Neethu and R Rajasree. Sentiment analysis in twitter using machine learning techniques. In *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–5. IEEE, 2013.

[2] Seyed-Ali Bahrainian and Andreas Dengel. Sentiment analysis using sentiment features. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 3, pages 26–29. IEEE, 2013.

[3] Bayu Yudha Pratama and Riyanarto Sarno. Personality classification based on twitter text using naive bayes, knn and svm. In *2015 International Conference on Data and Software Engineering (ICoDSE)*, pages 170–174. IEEE, 2015.

[4] Jie Li and Lirong Qiu. A sentiment analysis method of short texts in microblog. In *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, volume 1, pages 776–779. IEEE, 2017.

[5] Quanzhi Li, Sameena Shah, Rui Fang, Armineh Nourbakhsh, and Xiaomo Liu. Tweet sentiment analysis by incorporating sentiment-specific word embedding and weighted text features. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 568–571. IEEE, 2016.

[6] Hase Sudeep Kisan, Hase Anand Kisan, and Aher Priyanka Suresh. Collective intelligence & sentimental analysis of twitter data by using standfordnlp libraries with software as a service (saas). In *2016 IEEE international conference on computational intelligence and computing research (ICCIC)*, pages 1–4. IEEE, 2016.

[7] G Kavitha, B Saveen, and Nomaan Imtiaz. Discovering public opinions by performing sentimental analysis on real time twitter data. In *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, pages 1–4. IEEE, 2018.

[8] MJ Adarsh and Pushpa Ravikumar. An effective method of predicting the polarity of airline tweets using sentimental analysis. In *2018 4th International Conference on Electrical Energy Systems (ICEES)*, pages 676–679. IEEE, 2018.

[9] Megha Rathi, Aditya Malik, Daksh Varshney, Rachita Sharma, and Sarthak Mendiratta. Sentiment analysis of tweets using machine learning approach. In *2018 Eleventh international conference on contemporary computing (IC3)*, pages 1–3. IEEE, 2018.

[10] Jaishree Ranganathan, Allen S Irudayaraj, and Angelina A Tzacheva. Action rules for sentiment analysis on twitter data using spark. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 51–60. IEEE, 2017.

[11] Sunny Kumar, Paramjeet Singh, and Shaveta Rani. Sentimental analysis of social media using r language and hadoop: Rhadoop. In *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pages 207–213. IEEE, 2016.

[12] Nusrath Tabassum and Muhammad Ibrahim Khan. Design an empirical framework for sentiment analysis from bangla text using machine learning. In *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pages 1–5. IEEE, 2019.

[13] Md Rafidul Hasan Khan, Umme Sunzida Afroz, Abu Kaisar Mohammad Masum, Sheikh Abujar, and Syed Akhter Hossain. Sentiment analysis from bengali depression dataset using machine learning. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–5. IEEE, 2020.