Detecting Duplicate Questions

Prepared by :-

Bansi Shah - 17ce103

Darshi Shah - 17ce105

Prapti Shah - 17ce113

Suketu Shah - 17ce117

Abstract

We come across many QA forums where we repeatedly ask many questions which results in duplicate efforts and unnecessary waiting. Similar questions are the one that can have the same answers. Detecting such similar questions is quite challenging because of varying sentence structure and word choice. So, the solution for this is to automatically detect such duplicate questions. In this thesis, we will be using different approaches to detect such duplicate questions taking the dataset from QA forums like Quora. This approach will determine semantic similarities between the pair of questions on the dataset released by Quora by applying NLP(natural language processing)techniques and ML(machine learning) algorithms.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Millions of people are visiting the QA forums like Quora every month and many of them have the same questions. These questions with the same intent make writers feel they need to answer multiple versions of the same question.

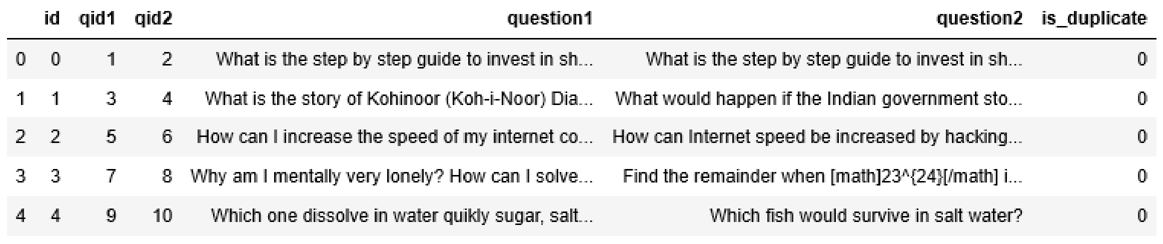
For example , consider the questions like “What are the effective weight loss plans?” and “How can a person reduce weight?” ,both these questions have the same intent and so they are called duplicates.

So it is important to find the questions that are already answered and instantly provide answers to them. For that detecting duplicate questions could be greatly beneficial. But it would be very difficult to perform this task manually looking at the quantity of the questions. So we need to have such an effective approach that would require very little human efforts.

In order to find whether the questions are duplicate or not we need to find the semantic meanings of these questions and for that we will apply Natural Language Processing techniques coupled with Machine Learning. We will use Word Embedding NLP techniques like word2vec through which we can have vector representation of all words of our questions. It may happen that the vector representation of two questions with different words may be the same. Then we will evaluate both LSTM and Bi-directional LSTM with provided keras package and pre-trained word2vec models. The approach is based on keras deep learning library.

For this, we will be using the Quora dataset consisting of over 400,000 questions duplicate pairs, each consisting of ID’s along with the pair of questions and the binary value indicating the duplicates as shown below :

Checking the first few samples of data.



	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

Fig 1.1: Sample dataset

Finding some information that can be helpful in summarizing the data.

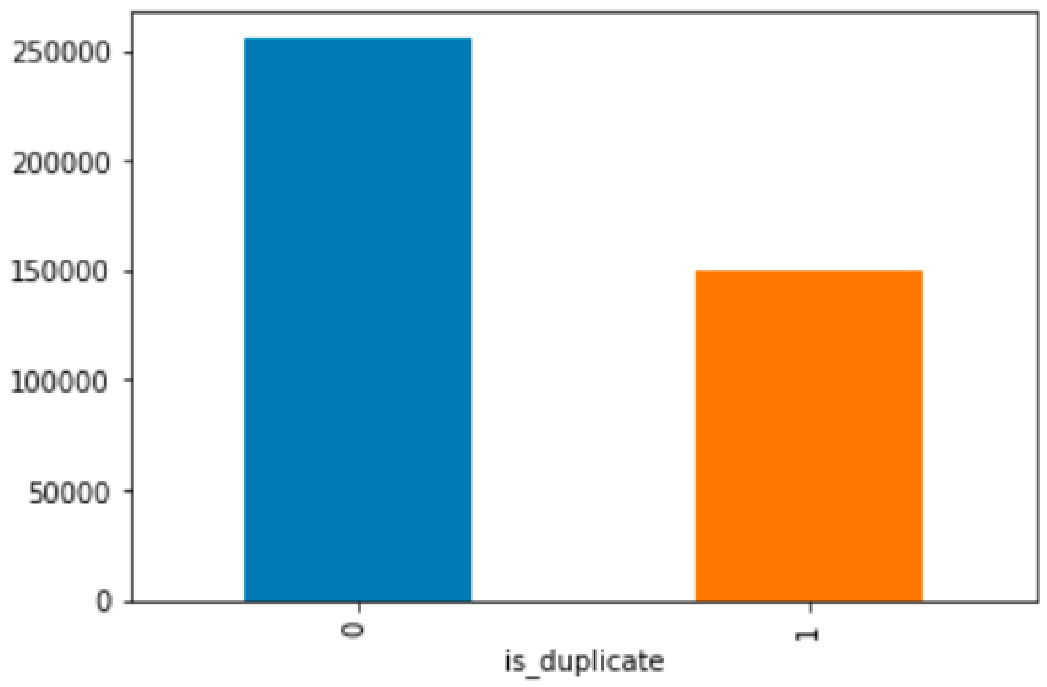


Fig 1.2: A bar graph is plotted to visualize the class distribution.

Chapter 2

Literature Review

This section presents the survey and depiction of works of literature pertinent to the objective of this paper. Later we present an outline of the performance baseline chosen from the pre-existing studies.

2.1 Different Approaches

Most deep learning techniques for recognizing semantic equality depend on a "Siamese" neural network structure that takes two sentences as input and encodes them separately utilizing the same neural system. A distance metric is then used to compare the two subsequent output vectors. Bogdanova et al. and Sanborn-Skryzalin, both were successful in using the above mentioned strategy.

Bogdanova et al. combined a convolutional neural network (CNN) with a cosine-similarity distance metric. CNN acquires the vectorial portrayal of the words in the two input segments, and the following convolutional layer builds a vectorial portrayal for every single one of the two segments. At long last, the two portrayals are thought about utilizing cosine-similarity. This framework was accounted for to score over 92% precision, turning to 30k data taken from the Meta forum in StackExchange and AskUbuntu forum, with an 80%/20% train/test split. Therefore, He established that coupling these two techniques was more efficient than traditional approaches of using Jaccard similarity or Support Vector Machines (SVMs) in detecting duplicate questions in a StackExchange dataset.

Sanborn and Skryzalin analyzed the use of Recurrent Neural Networks (RNNs) and recursive neural networks with conventional machine learning strategies and discovered that RNNs worked the best on the SemEval-2015 dataset.

A deep neural network technique to detect duplicate questions got the best accuracy in the SemEval-2016 Task 1 "Question-Question" subtask, in particular 0.73035 in terms of Pearson correlation coefficient, which had the aim of deciding the level of similitude, on a 0–5 scale, between two questions.

In the pertinent literature, *“Detection of Duplicates in Quora and Twitter Corpus”*, the writers have tested with six conventional ML classifiers. They applied a basic way to deal with retrieval of six basic features, for example, word counts, common words, and term frequencies on question pairs to train their models. The best efficiency seen right now is 72.2% and 71.9% acquired from binary classifiers random forest and KNN, respectively.

In the previous work “Duplicate Question Pair Detection with Deep Learning”, they applied deep learning strategies such as CNN, LSTM, and a hybrid model of CNN and LSTM layers. It indicates that deep learning methods were more accurate than traditional NLP methods. With an accuracy of 81.07%, their best model is LSTM network.

In their experiments, they used a GloVe word vector of 200 dimensions trained using 27 billion Twitter words.In the Quora dataset, there is an unpublished paper regarding duplicate questions detection titled “Bilateral multi-perspective matching for natural language sentences”. It proposes a BiMPM model under the “matching aggregation” framework i.e. the model compares sentences X and Y in two directions ($X \rightarrow Y$ and $X \leftarrow Y$), resulting in an accuracy of 88.17%.

2.2 Summary

Table 2.1 Summary of literature review

Paper	**Model**	**Technique**	**Accuracy**
Detection of Duplicates in Quora and Twitter Corpus	Logistic Regression Decision Tree SVM KNN Naïve Bayes Random Forest	Machine Learning	0.671 0.693 0.600 0.719 0.637 **0.722**
Duplicate Question Pair Detection with Deep Learning	LSTM (twitter word embedding 200d)	Deep learning	**0.8107**
Detecting semantically equivalent questions in online user forums	CNN (Askubuntu word vectors)	Deep learning	**0.924**
Bilateral multi-perspective matching for natural language sentences	BiMPM	Deep learning	**0.8817**

Chapter 3

Proposed Work / Research Methodology

In this chapter, we have described a general approach of training our machine learning classifiers, the process flow for feature importance analysis and we have used two types of models for this project which are LSTM and bidirectional LSTM.

3.1 Experimental and Research Design

Influenced by literature and previous study, we started this machine learning project with gaining theoretical knowledge about machine learning, various preprocessing techniques, word embeddings and it's training models. To understand the practical implementation of such techniques, we implemented some of the techniques which we were going to use in this project individually.

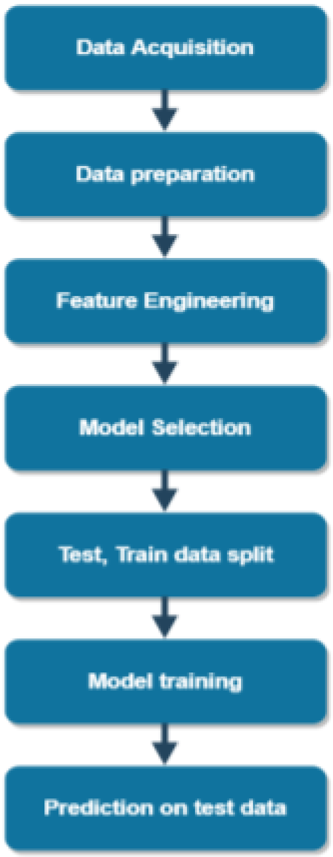


Fig 3.1: The flow of the implementation

In this project, we have used the latest Quora question pair dataset. The Quora question dataset is collected and then cleaned. Python libraries genism, nltk, numpy are used to extract features. Word2Vec features are extracted using gensim and Google pretrained vectors. Machine learning classifiers are trained on the training set and the prediction results are evaluated on the test set.

The above diagram shows the flow of our implementation where the data acquisition is the first step where the data which is to be analyzed is fetched. After the acquisition process, the data is preprocessed which is also called data preparation where the sentences are tokenized and stop words are removed for making the data smaller and more appropriate. Then comes feature set, which is a set of all the attributes that you're interested in, e.g. height and age. The data is then converted into feature sets for reducing their complexity and plotting then in matrices. After all these data processing we need to select the most important thing which is selection of model. The model is an artifact which is a way of training the machine for better accuracy (here we have used LSTM and bidirectional LSTM). When the machine is trained, tests are performed for further confirmation and for checking of the predictions of the test data.

3.2 Discussion of Approaches and Methods

The limitation of the Traditional Neural Networks is that they can only deal with the present information. We human beings don’t start their thinking from scratch at every second, i.e. you understand every word based on your previous word's understanding. Also, you don’t throw everything away and start thinking from scratch again. Thus, there is persistence in your thoughts. If we want to derive the present information with reference to previous information, traditional neural networks can’t do this. Thus, we use Recurrent Neural Networks (RNNs) to overcome this issue.

Recurrent Neural Networks have similar structure to Traditional Neural Networks but the only difference is that they are the networks with loops in them which allows the information to persist. Consider that we are trying to predict the last word in “The clouds are in the ____”. Here, it is pretty clear that the last word would be “sky” forming the sentence “The clouds are in the sky”. In such cases, where the gap between the relevant information and the place that it’s needed is small, RNNs can learn to use the past information. But if we are trying to predict the last word in the text “I grew up in France… I speak fluent French”. In this case, the previous information suggests that the next word is the name of the language. But as there are many languages, we need to narrow down by using the context “France”. Here, the gap between the previous information and the point where we need to predict the word is possible to be very large. Unfortunately, RNNs are unable to learn to connect the information as the gap grows larger and larger.

To overcome this limitation of RNN, we use Long Term Short Memory (LSTM) which is a recurrent neural network that is explicitly designed to overcome this problem of long term dependencies. Remembering information for long periods of time is practically their default behavior. They need not struggle to learn this. LSTM has been successfully used in text generation, language translation, scene text etc. Consider that we want to predict missing word in the in the I am ___ student. If we use LSTM then it will use only ‘I am’ to generate the next word and based on the examples it has seen during training it will generate a new word (it may be ‘a’, ‘very’ etc.).

To overcome this limitation of LSTM, we use Bidirectional LSTM. It has two networks, one access information in forward direction and another access in the reverse direction. Bidirectional LSTMs have access to the past as well as the future information and hence the output is generated from both the past and future context. With reference to the example discussed above, bidirectional LSTMs have information of past (I am) and future (student), so it can easily see that here it has to be ‘a’. Bidirectional LSTMs have been successfully used in image captioning, future prediction, language translation etc.

Chapter 4

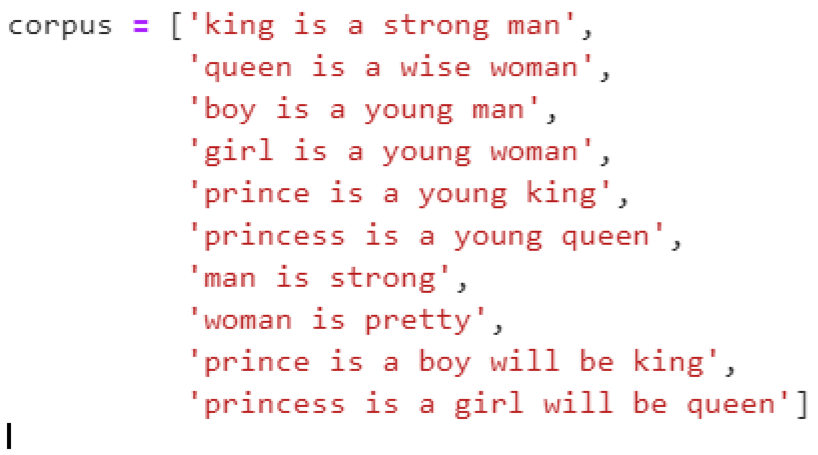
Implementation / Results

4.1 Implementation Process

4.1.1 Word2Vec:

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. We have been implementing this for detecting duplicate questions as the mapping between the target word to its context word implicitly embeds the sub-linear relationship into the vector space of words, so that relationships like “*king:man* as *queen:woman*” can be inferred by word vectors.

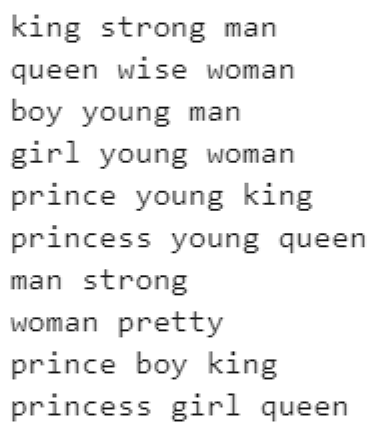
We initially implemented the knowledge that we gained about word2vec using a corpus dataset and learnt how it works. The following figure shows the corpus dataset on which we implemented word2vec initially:



```
corpus = ['king is a strong man',
          'queen is a wise woman',
          'boy is a young man',
          'girl is a young woman',
          'prince is a young king',
          'princess is a young queen',
          'man is strong',
          'woman is pretty',
          'prince is a boy will be king',
          'princess is a girl will be queen']
```

Fig 4.1 Corpus dataset used in word2vec

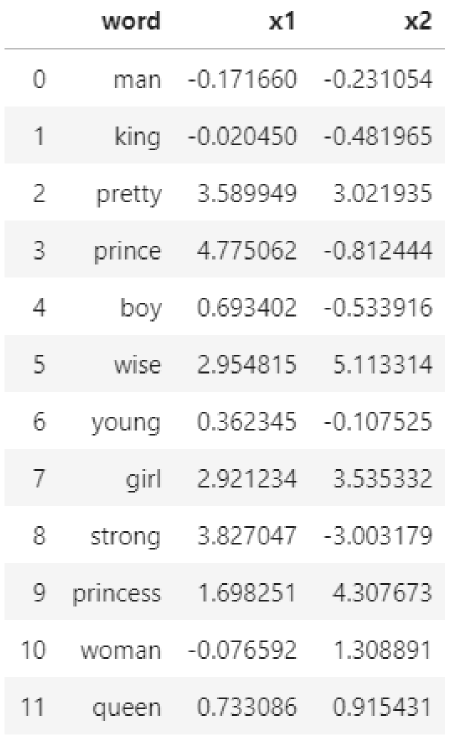
From this dataset as you can see there are many words which are unnecessary for this context, that is, the stop-words like “is”, “a”, etc. We need to remove such kind of stop-words to be able to find the similarity between the words. The following figure shows the dataset after performing data preprocessing, thats is, after removing the stop-words:



```
king strong man
queen wise woman
boy young man
girl young woman
prince young king
princess young queen
man strong
woman pretty
prince boy king
princess girl queen
```

Fig 4.2: Preprocessed corpus dataset

Basically, through word2vec we can have vector representation of all the words of our questions. It may happen that the vector representation of two questions, with different words may be the same, based on which we can say that the two questions are similar.



	word	x1	x2
0	man	-0.171660	-0.231054
1	king	-0.020450	-0.481965
2	pretty	3.589949	3.021935
3	prince	4.775062	-0.812444
4	boy	0.693402	-0.533916
5	wise	2.954815	5.113314
6	young	0.362345	-0.107525
7	girl	2.921234	3.535332
8	strong	3.827047	-3.003179
9	princess	1.698251	4.307673
10	woman	-0.076592	1.308891
11	queen	0.733086	0.915431

Fig 4.3: Vector representation of words from the dataset

As a result, we obtain the graph presenting the vector form of the words. In this graph, we can see that the words which are similar to each other or which relate to each other are seen very close to each other in the graph.

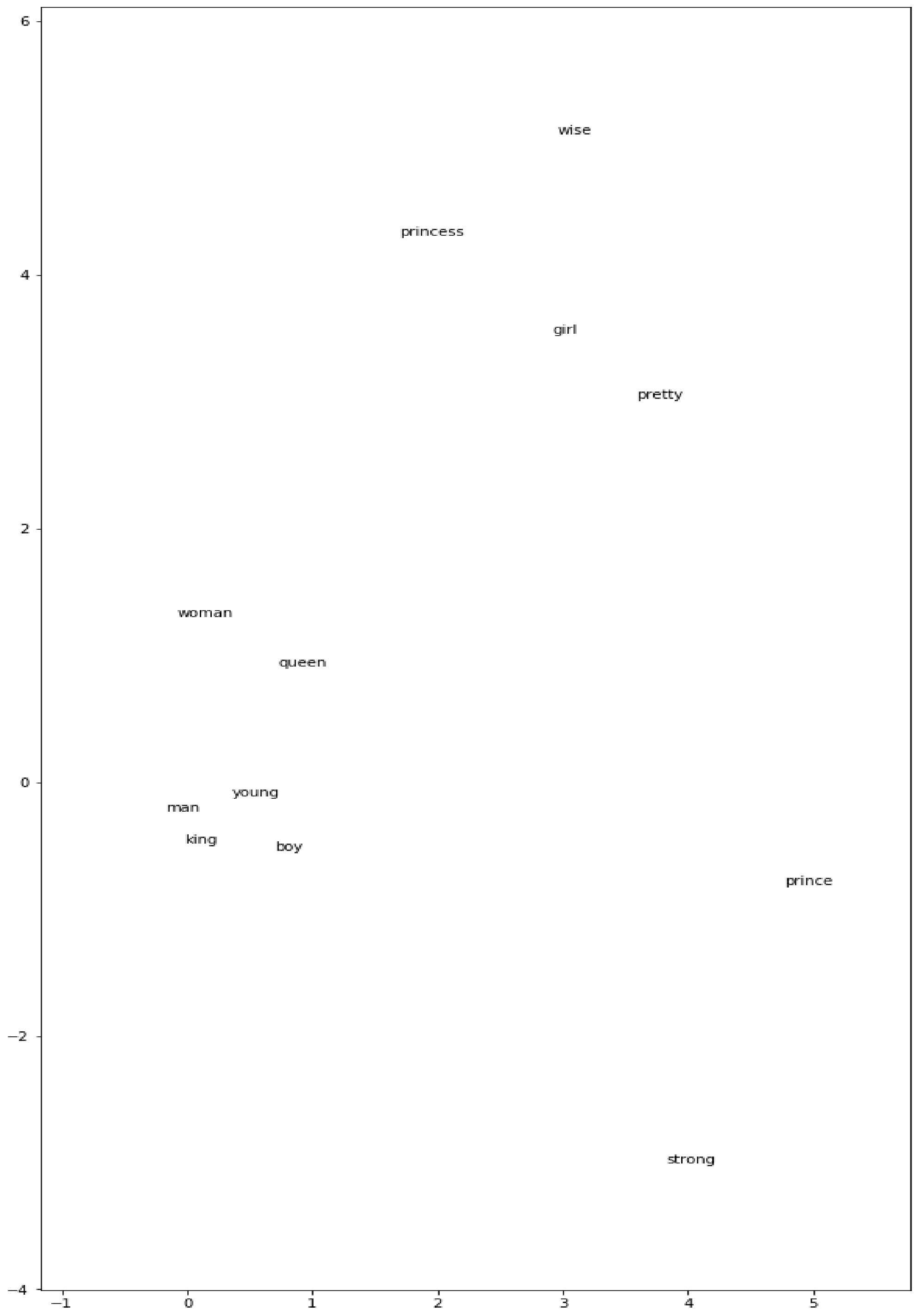


Fig 4.4 Graph representing word vectors

After exploring and implementing the word2vec model using a corpus dataset, we implemented the word2vec model on the quora dataset which was going to be used in the project. This time we implemented the word2vec with the help of the gensim library which makes its integration with the machine learning model flexible. The plot below represents the word vectors for 20 questions sampled from the quora dataset. Dimensions reduction was used to reduce the vectors from 300 to two dimensions.

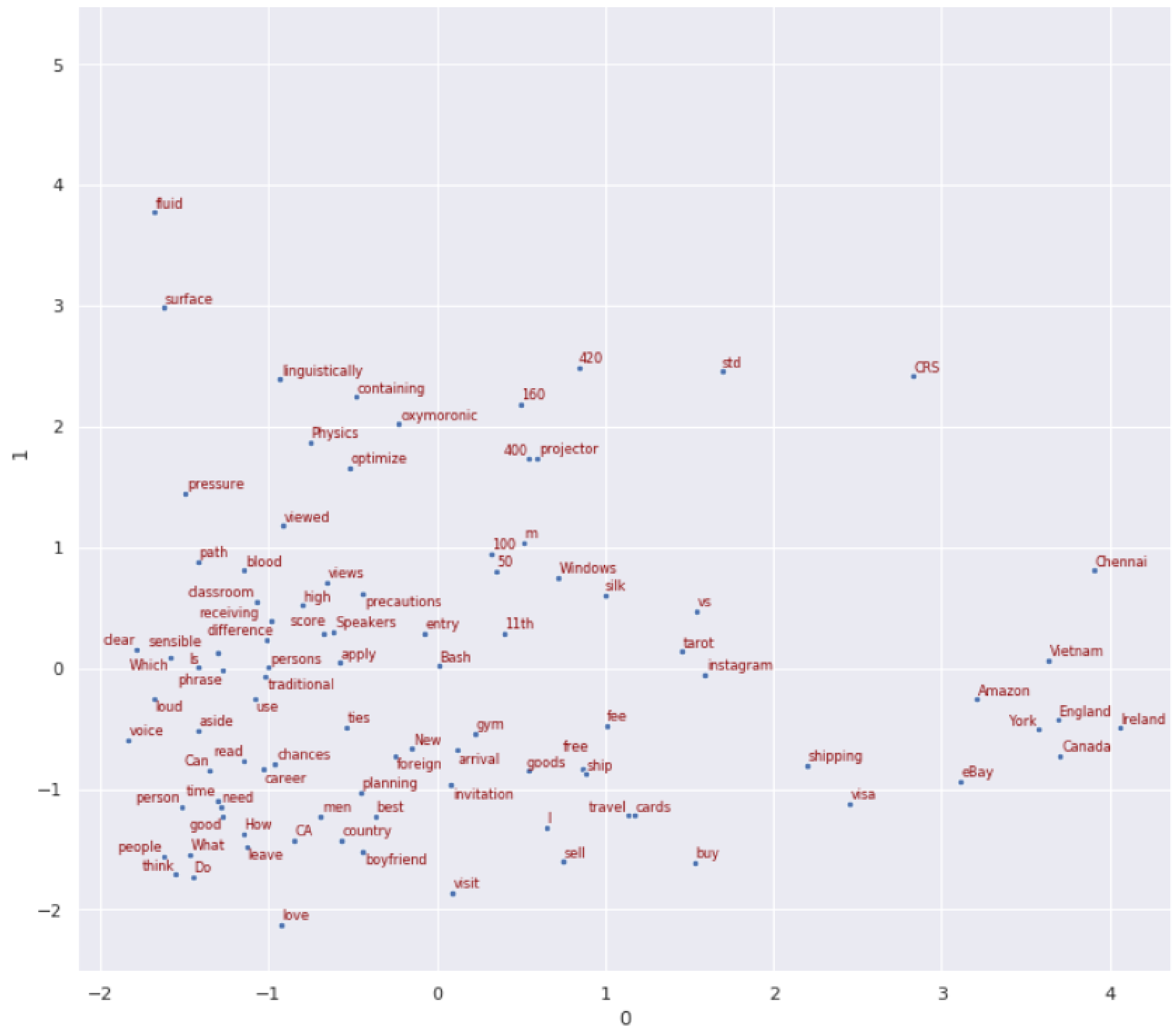


Fig 4.5 Graph representing quora word vectors

4.1.2 LSTM (Long short term memory network)

These are special types of Recurrent neural networks which have the capacity to sustain long term dependencies which are now widely used. LSTMs have a default behaviour of remembering information for a long period of time.

LSTMs have a chain like structure but when it repeats it has a different structure which has four chained structures in a special way.

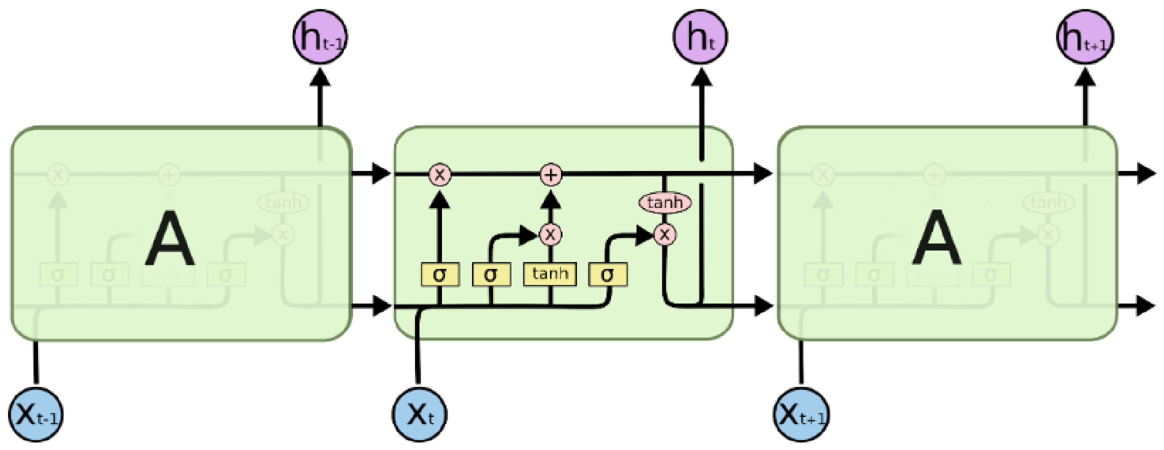


Fig 4.6: Structure of LSTM

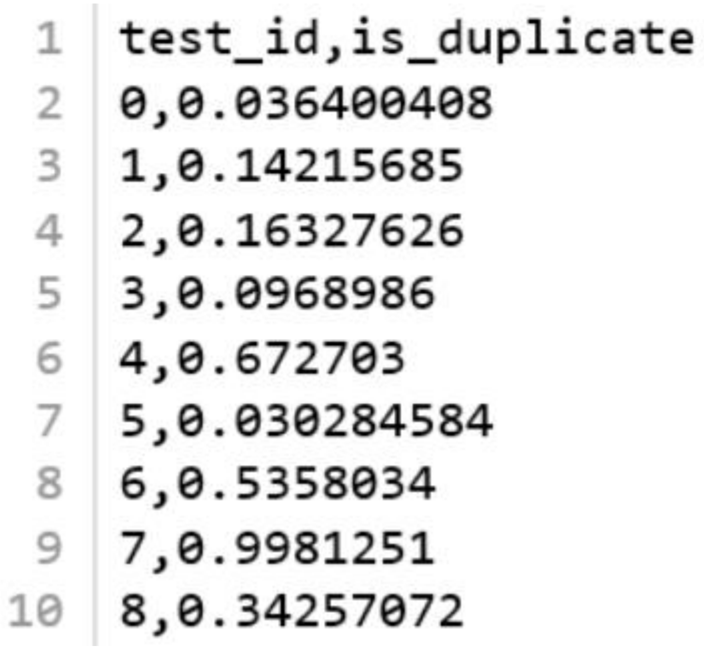
- **Core idea of LSTM**

  The Recurrent neural network suffers in vanishing gradient and exploding gradient while performing practically. This problem encouraged the invention of LSTM which introduces an explicit memory unit.

  There exist different memory blocks called cells which we can see in the above diagram. The states which have been transferred to the next state are the cell state and hidden state. This memory which is remembered by the memory blocks are manipulated through mechanisms called gates.

  LSTMs are a very promising solution to sequence and time series related problems. However, the one disadvantage is the difficulty in training them. A lot of time and system resources go into training even a simple model. But that is just a hardware constraint!

We implemented the LSTM machine learning model by using google pre-trained word2vec model. The basic step which we have performed is indexing of word vectors. After indexing the word vectors, we removed the stop-words and unnecessary characters from our dataset to obtain the processed data. After obtaining the processed data, we prepared the embedding matrix by using the word2vec model and integrated the LSTM model with the pre-trained word2vec model. Then we performed sample training and validation of data. After which we defined our model structure and trained our model. Finally as a result, we generated a .csv file which shows the duplicate occurrence of each question. In this LSTM model we have done a total of five epochs while training the model and the accuracy we get is 72.19%. The following figure gives a glimpse of .csv file that has been generated showing the duplicate occurrence of each question:



```
test_id,is_duplicate
0,0.036400408
1,0.14215685
2,0.16327626
3,0.0968986
4,0.672703
5,0.030284584
6,0.5358034
7,0.9981251
8,0.34257072
```

Fig 4.7 Glimpse of .csv file showing duplicate occurrence of each question

4.1.3 Bidirectional LSTM

The bidirectional LSTM machine learning model is almost similar to the LSTM model and is just different in a way that in the bidirectional LSTM we access the past information as well as the future information. So the implementation of the bidirectional LSTM is also almost similar to the implementation of the LSTM model. In this implementation of LSTM, the first few steps are similar to that of LSTM where we have used a pre-trained word2vec model, processed our data and created the embedding matrix. The implementation of the bidirectional LSTM differs from that of the LSTM model by the embedded version which we have created here for the left input as well as the right input. At the end, as a result we obtain the accuracy of 75.32% which is greater than the LSTM model proving that the bidirectional LSTM model is more efficient than the LSTM model.

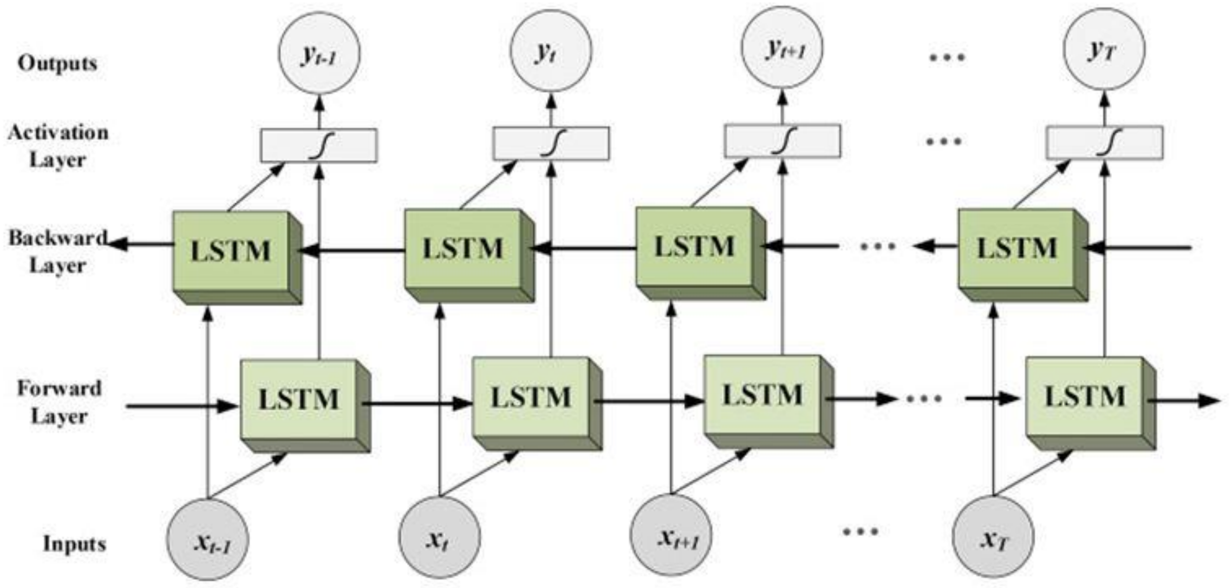


Fig 4.8: Structure of Bidirectional LSTM

Chapter 5

Conclusion and Future Enhancement

Summarizing our work,we have first presented with a study of the various approaches taken up to solve the problem of detecting duplicate questions. Further, we have presented the implementation of the word2vec NLP technique and the implementation of the LSTM and Bi-directional LSTM models with the accuracies of 72.19% and 75.32% respectively.

For future work, we aim to implement the attention model and compare its accuracy with the LSTM and Bi-directional LSTM models.

All the code for this study is available on Github:

https://github.com/Prapti1199/Detecting-Duplicate-Questions-QA-Forums-

Chapter 6

Bibliography

[1] “Detecting Duplicate Questions with Deep Learning” by Yushi Homma , Staurt Sy and Chritopher Yeh

[2] “Identifying Semantically Duplicate Questions Using Data Science Approach: A Quora Case Study” Navedanjum Ansari

[3] “Detection of Duplicates in Quora and Twitter Corpus” by Sujith Viswanathan, Nikhil Damodaran, Anson Simon, Anon George, M. Anand Kumar and K. P. Soman

[4]keras official incorporate : https://keras.io/examples/imdb_bidirectional_lstm/

[5]”Duplicate Question Pair Detection with Deep Learning” Travis Addiar

[6] “Detecting Duplicate Questions” Alfonce Nzioka

[7]”https://towardsdatascience.com/word2vec-from-scratch-with-numpy-8786ddd49e72”

[8]”Detecting Semantically Equivalent Questions in Online User Forums“ Dasha Bogdanova, C´ıcero dos Santos, Luciano Barbosa and Bianca Zadrozny https://www.aclweb.org/anthology/K15-1013/

[9]”Detecting Misflagged Duplicate Questions in Community Question-Answering Archives” Doris Hoogeveen, Andrew Bennett, Yitong Li, Karin M. Verspoor, Timothy Baldwin

[10]”https://medium.springboard.com/identifying-duplicate-questions-a-machine-learning-case-study-37117723844”

[11]”Multi-Factor Duplicate Question Detection in Stack Overflow” Yun Zhang, David Lo, Xin Xia & Jian-Ling Sun