Detecting Semantically Equivalent Questions Using Transformer Based Encodings

Abstract --- Due to the increasing influx of users on various Q &A forums like Quora, Stack-overflow, etc, the probability of redundancy in questions asked has drastically increased. Answers get fragmented across different versions of the same question due to this redundancy of questions.This results in unsatisfactory search results due to the dearth of answers for a particular question. However, if questions which are lexically or semantically identical could be grouped together, the search results for a given question could yield the assimilation of answers provided for all versions of a question. Dataset of more than 400,000 questions pairs provided by Quora are preprocessed through and used for the feature extraction. In this paper, we create an ensemble of 4 separate models using 4 different word-embedding techniques in each. The vectorized data from each of the embedding layers is passed through a custom made architecture using Transformer based encodings. This architecture is used to determine if a pair of questions have the same semantic meaning or not. The semantic equivalence of question pairs can further help in their assimilation in order to provide better and search results for answers. Finally, on testing the model on a subset of the provided data, the experiments show that the proposed model achieves an F1 score of 84.63%

Keywords --- Artificial Intelligence, Deep Learning, Transformer, Duplicate Text Detection, Semantic Equivalence

I. INTRODUCTION

In Natural Language Processing (NLP), semantic similarity plays an important role in many NLP applications like sentiment analysis, neural machine translation, natural language understanding and many others as mentioned by Goutam Majumder et al [1]. It is the process of encompassing the underlying meaning of two pieces of text and then comparing how similar they are. This technique is used in various Q&A forums to tackle the problem of Question Duplication.

However, due to a massive increase of users on such forums, it is quite likely for a subset of these users to ask questions which semantically have the same meaning. According to Bogdonova et al. [2], two questions can be called semantically equivalent if they can be answered by the same question. These Answers get fragmented across different versions of the same question due to the redundancy of questions in these forums. Thus, at platforms like Quora, identifying duplicate or semantically similar questions and merging them makes knowledge sharing more efficient and effective in many ways. This way, users can get answers to all the questions on a single thread and writers do not need to write the same answer in different locations for the same question. At the time, Quora used the Random Forest model to identify duplicate questions. Due to the inaccuracy of this method attributed to its simplistic architecture which lead to an unsatisfactory performance for the given task and the rapid advancements in the field of Machine Learning and Deep Learning, Quora released a dataset of more than 400,000 pairs of questions with labels indicating if a given pair could be considered duplicate or no. This dataset was then set out to be harnessed in a Kaggle competition. Participants were challenged to tackle this natural language processing problem by applying advanced techniques to classify whether question pairs are duplicates or not.

In order to tackle the problem of detecting text equivalence, a bunch of methods were experimented with. The most successful architecture included the creation of a setup similar to a Siamese network, using Transformer Based Encodings. An ensemble of this architecture was created which used the combination of 4 different types of word2vec models, ie. FastText Crawl, FastText Crawl Subword, Google News Vector and FastText Wiki News Embeddings. This ensemble was then harnessed to predict the semantic similarity between a pair of questions.

It is assumed that questions marked as duplicates in the Quora dataset are semantically equivalent since Quora's duplicate question policy concurs with this definition of semantic equivalence [4].

The rest of the paper is structured as follows. Section 2 describes the current research done related to this work. In Section 3, the dataset and the pre-processing techniques applied to it are explained. Section 4 introduces the proposed architecture of the proposed model and the underlying deep learning concepts used in the design of it. Section 5 presents the results obtained in this research using the proposed model. The paper ends with conclusion and direction for future work.

II. RELATED WORK

The task of detecting duplicate texts on a semantic rather than lexical level involves understanding the underlying ideas the pairs of texts convey. The complete meaning of a text can change the insertion or deletion of a few words so even though a pair of text may be lexically similar, their semantic meaning might be very different. In order to understand the meaning of the text, various sequence models were adopted. Multiple methods were researched [5] in order to determine the similarity of texts by calculating the distance between the vectorised representations of the pair of texts. The initial methods for detecting duplicate questions included the use of a siamese network [6]. This method included using parallely stacked character-based BI-LSTM. Cosine-similarity was used as a distance metric for job-title normalisation. E Dadashov et al in [10] used a Siamese Ma-LSTM architecture was created for the task of detecting duplicate questions on the Quora Question Pairs datasets. This model achieved an F-1 score of 79.5% and accuracy of 83.8% .

In machine learning models the words need to be converted to some numerical representation before they can be processed by the model. Traditionally BoW and TFIDF have been very popular but in recent times pre-trained word embeddings have been very instrumental in models which have produced state of the art results in nlp tasks. These embeddings have been better at capturing the meaning of words in a sentence.

Ensemble models helped in harnessing the capabilities of multiple word embeddings into a single model. Imtiaz and et al used the combination of 3 word embeddings for the task of duplicate question detection [11].

However as the sequence length increases the training time for LSTMs, RNNs and comparable models increases drastically. This kind of behaviour is also noticed with considerable increase in training data. Due to the underlying complexity LSTMs and RNNs are not considered hardware friendly and take a considerably large amount of time to generate a trained model.

With the introduction of Transformers [12] a new method to Machine Translation was created. It contains an encoder and decoder to convert sequential data from one form to another. Ostendroff et al used a Vanilla Transformer (BERT) for the task of Text Similarity [13]. This architecture also used Deep Neural Networks for computing the semantic distance between two pieces of text which provided a change from the earlier methods like Manhattan distance, Cosine Similarity, etc. Transformer encodings can be used for a multitude of applications like text classification [14], question-answering, neural machine translation, etc. Due to the ubiquity of the research done on this topic using the Quora dataset, it was decided to use it as the primary dataset for the research. In the next section, an in-depth analysis of the Quora dataset is done along with a description of how the data was preprocessed in order to suit the purpose of the proposed architecture.

III. DATASET & PREPROCESSING

The dataset used for this research is the Quora Question Pairs dataset downloaded from Kaggle [4] . Each record has a pair of questions and a target class that represents whether the questions are duplicate or not. The description of dataset columns is shown in Table 1. In the following subsection, an in-depth description of the dataset and an explanation about the preprocessing techniques used on this dataset are provided.

TABLE 1: Description of Dataset

Attribute	Description
id	ID of a question pair.
qid1	ID of question 1
qid2	ID of question 2
question 1	Full content of question two.
question 2	Full content of question two.
is_duplicate	The target label. When set to 1, it means that the pair of questions are semantically equivalent. When set to 0, it means that the pair of questions is not equivalent

A. Dataset

Quora released a public dataset that consists of 404,351 question pairs in January 2017 The question pairs are from various domains including technology, entertainment, politics, culture, and philosophy. This dataset is downloaded from Kaggle [4]. The data is split randomly into approximately 376k train examples, 11k dev examples, and 15k test examples. In the given dataset duplicated question pairs only make up 37% of the training data, indicating that the data set is heavily imbalanced with many more non-duplicated question pairs than duplicated ones. In addition, 33% of questions appear multiple times in different question pairs

B. Preprocessing

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. Various pre-processing steps were performed on the Quora dataset. NLP techniques are used such as conversion of text to lower case, lemmatization, and tokenization, with the help of freely available libraries such as Keras. Since the length of the questions was small, its preferable to abstain from using functions provided by the above mentioned libraries since it included negative words like “no” and “not” as a part of stopwords. Since these words are quite important when considering the semantic meaning of the question, stopwords are not dropped

After the preprocessing stage, the data is converted in a format suitable to be fed to the embedding layer and only contains information relevant for prediction. Using the help of tokenizers, raw textual data is converted into a vectorized representation of words. This vector of words is then assigned unique token numbers for respective words. Once the data is converted to a numeric format, a post-padding step to set the maximum length of all the text to 30 is performed. Questions with lengths larger than 30 will be cut short and the ones with length less than 30 will be zero-padded. Once the numerical data is padded, it is fed to the embedding layer where all the 4 word embeddings (ie. FastText wiki-news, FastText crawl, FastText subwords crawl, Google news vector) are applied individually and then processed further. The preprocessed data was made suitable to be usedin the architecture explained in section 4 below. The following section provides a comprehensive understanding about the various components of the setup and the architecture as a whole.

IV. PROPOSED METHODOLOGY

The proposed model is inspired by the transformer architecture [12] [15]. It uses the transformer encoder which consists of positional encodings, multi-head attention, and a feed-forward network which consists of Layer normalization and two fully connected layers. Unlike RNNs and LSTMs models in which parallelization is not possible because of their sequential nature, the transformer can be parallelized leading to faster training times. The function of the encoding layer is to find encoding for each word of the sentence by taking every other word of the sentence into consideration which is achieved by the self-attention mechanism which draws information from every other word and weighs their relevance to the current word before calculating the encoding for that word.

The proposed model, shown in Fig 2 consists of an embedding layer, the Transformer encoding layer and the feed forward network. The word embedding layer uses four distinct word embedding models to vectorize data. The Transformer encoding layer converts this vectorized data from both sentences into representations that are compared to each other by flattening and concatenating them before passing them through a series of fully connected layers with ReLU and dropouts. The prediction of the question pairs is found by passing the output of the fully connected layers to a softmax classifier.

Multiple models have been created using different pre-trained embeddings namely FastText wiki-news, FastText crawl, FastText subwords crawl, Google news vector [7-9] and a model which is a combination of all the models. After each of the single models make their predictions the ensemble model takes an average of the prediction probabilities of all the models which gives us the class probabilities for the ensemble model. The probabilities obtained are then used to make the final class predictions. The preprocessed data is first passed through a series of layers explained below and shown in Fig. 1. These layers are interconnected to form the Transformer Encoding layer.

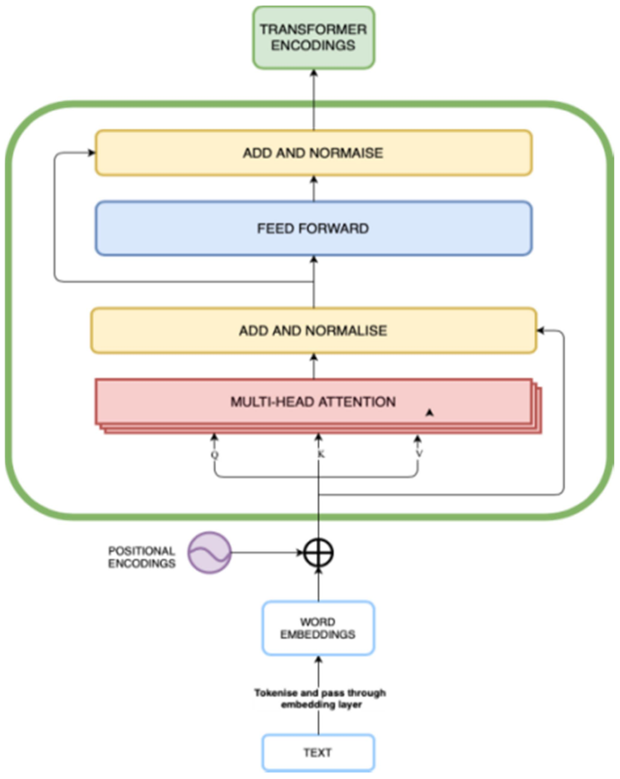


Fig. 1: Transformer Encoding Layer

1) Positional Encodings

The position and order of words in a sentence created the desired meaning which makes it very important. Since LSTMs and RNNs analyze the sentence word by word it implicitly takes the position of words into account. In transformers since there is no such mechanism, a way must found to pass this information to the model. Positional encodings [16] offer a way to solve this problem. Out of the various possible choices for positional encodings sine and cosine functions are used as used in [11].

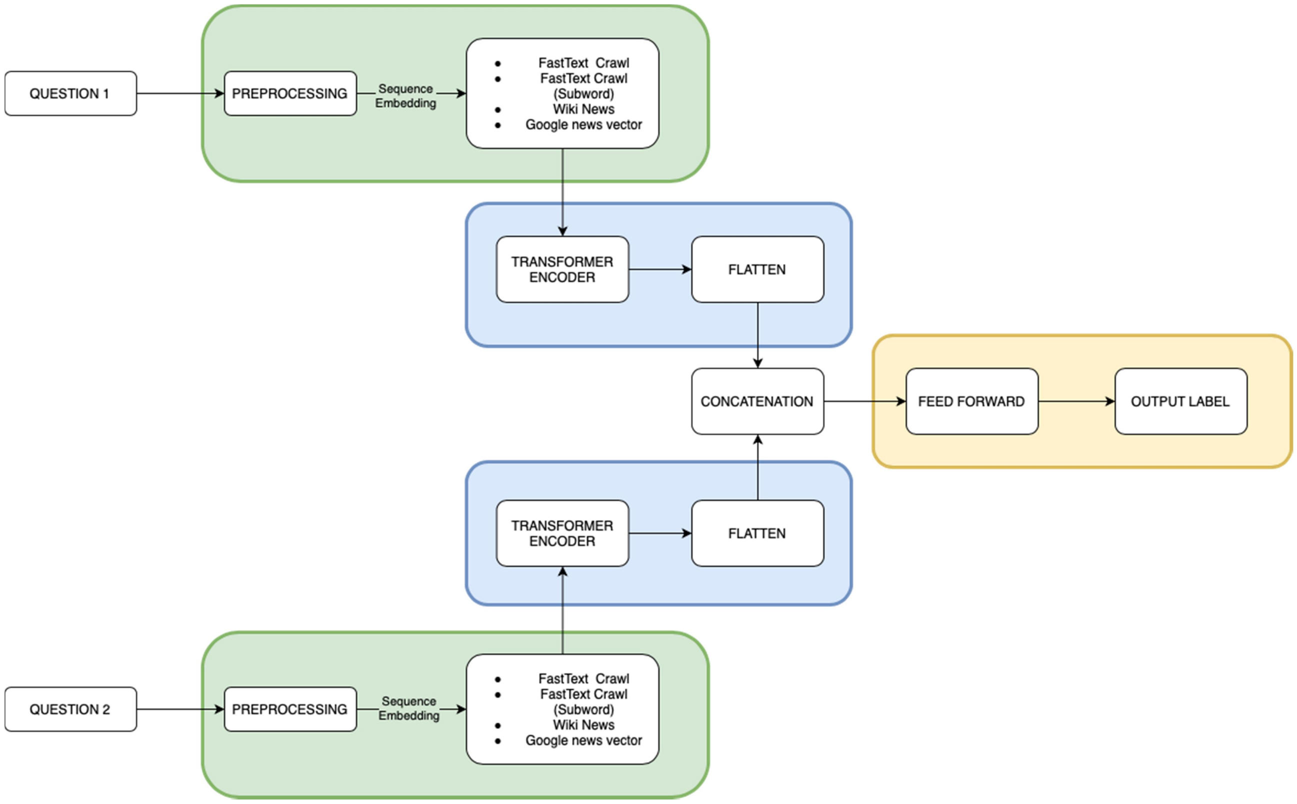


Fig. 2: Model Architecture Diagram

$$PE_{(pos,2i)}=\sin(pos/10000^{2i/dmodel})$$

$$PE_{(pos,2i+1)}=\cos(pos/10000^{2i/dmodel})\ )$$

2) Scaled Dot Product Attention

In the proposed model, scaled dot product attention is used in which 3 vectors from each of the input vectors are created, it creates the query, keys, and values vectors each having the same dimension *d*. The dot product of the query and key vectors is computed before scaling it by dividing it by $\sqrt{d}$ and passing it to a softmax function to get the self-attention weights. These self-attention weights are then multiplied by the value vectors to get the scaled dot product attention.

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d}})$$

3) Multi-Head Attention

Each of the Query, Keys and, Values are split into multiple heads and each of these heads is then passed to the scaled dot product attention where they run in parallel. The outputs of these multiple heads are then concatenated and are linearly transformed to obtain the expected dimensions. The idea is to combine the knowledge from multiple heads. According to the (attention paper) multi head attention allows to combine knowledge from different representations at different positions which is not possible with a single head. For the model different values for the number of heads were tried and it was found that 10 heads worked the best.

$$MultiHead(Q,K,V) = [head_1; \dots; head_h]W^O$$

$$\text{where } head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

4) Feed-Forward Networks

The output from the multi-head attention is passed to a fully connected feed-forward network consisting of 2 linear transformations with a ReLU activation in between them. The dimension of the input and output of the model is 300 and the dimension of the feed-forward layer is 512. The output of the multi-head attention and the linear transformation is followed by layer normalization where input to it is x+sub-layer(x).

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2$$

The designed model was then trained, validated and tested on the Quora Dataset and in Section 5, the results obtained by various components of the model as well as the ensemble are discussed.

V. RESULTS & DISCUSSION

The model has been trained on various pre-trained embeddings (FastText wiki news, FastText crawl, FastText crawl subword and Google news vector) individually and tried various combinations of them. The ensemble of FastText wiki news, FastText crawl, FastText crawl subword and Google news vector provided the best results. The model was trained on 376k question pairs and tested them on 15k question pairs. For the proposed model this research used a single transformer and the dimensions of the feed-forward network that provided the best results was 1024. This research used a dropout rate of 0.1 while training the model and L2 regularization with regularization parameter of 1e-5. Additionally label smoothing with a smoothing of value 0.2 was used. Adding label smoothing increased the accuracy by around 1-2% and improved the F1 score slightly. A custom averaging layer which considered the original length of the sentence before adding any padding was also experimented with, global average pooling, global max pooling and a flatten as the layer after the transformer and observed that flattening the transformer output provided the best results. After concatenating the output from both transformers it was tried to directly pass it to the softmax, passing it to a feed-forward network with tanh activation and passing it to a feed-forward network with ReLU activation. The feed-forward network with ReLU activation provided the best results. As evident from table 2 FastText wiki news, FastText crawl, FastText crawl subword and Google news vector provided accuracies of 83.36%, 83.85%, 83.32% and 83.23% and F1 scores of 81.85%, 82.55%, 81.72% and 81.72% respectively and the ensemble model had an accuracy of 86% and a F1 score of 84.63%. The details of models with each type of word embedding as well the ensemble is shown in Table 2 where its Precision, Recall and Accuracy are shown as well. The performance of each of the models are shown diagrammatically in the form of confusion matrices in Fig 3.

To understand how different embeddings and the combination are working the results for 5 pairs for each model have been extracted as shown in Table 3. In the first question, all models have predicted the same label as the true label. In the second question, the model using google news vector has predicted the wrong label but as all other models have predicted the label correctly the combination model predicts the label correctly. In the third and fifth questions, the pattern of the results is similar to the second question but instead of the model with google news vector, the model with FastText subword crawl predicts the wrong class. In the fourth question, two models predict the wrong class and two predict the right class so the prediction of the combination model depends on the probability with which each model predicts the class.

In Fig. 4 and Fig.5, a graphical description on the variation of the accuracy and loss (in Fig. 4 and 5 respectively) is shown for all of the embeddings used in the ensemble. Each of the embedding's performance is analysed during its respective training phase.

The successful performance of this architecture opens doors to new methodologies and techniques for this use case. The future scope and concluding remarks on this research has been discussed in the next section.

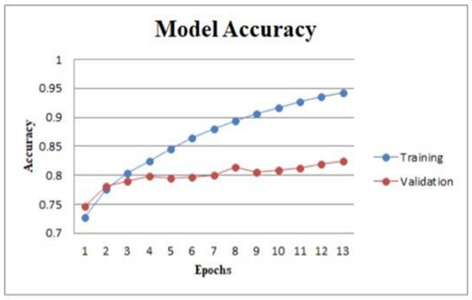
TABLE 2: Results Using Different Word Embeddings

Word Embedding	Accuracy	Precision	Recall	F1-Score
FastText Crawl (600B)	83.85 %	82.58%	82.53%	82.55%
FastText Crawl Subword (600B)	83.32%	82.28%	81.27%	81.72%
FastText Wiki News (16B)	83.36%	82.21%	81.54%	81.85%
Google News Vectors	83.23%	82.05%	81.43%	81.72%
Ensemble of 4 word embeddings	86.0%	85.31%	84.1%	84.63%

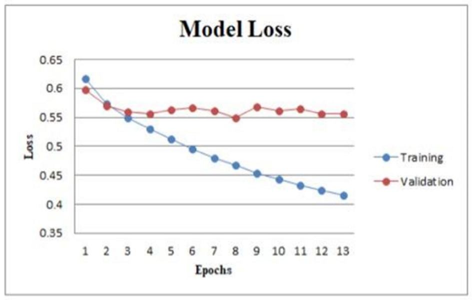
VI. CONCLUSION & FUTURE WORK

In this paper, Transformers have been used as the primary source of encoding text to extrapolate its semantic meaning (feature extraction). It is found that the performance by coupling of Transformer based encodings and deep neural networks for similarity calculation performed comparable and sometimes better results than existing methods depending on the preprocessing and word embeddings used. In order to improve performance further, a combination of the results predicted by four different word embedding models (Google News Vector, FastText crawl, FastText crawl subwords and FastText Wiki News) has been used. This combination helped us achieve a performance superior to the one achieved when their embeddings were used individually. This research managed to achieve an accuracy of 86% and an f1 of 84.63% with this ensemble model.

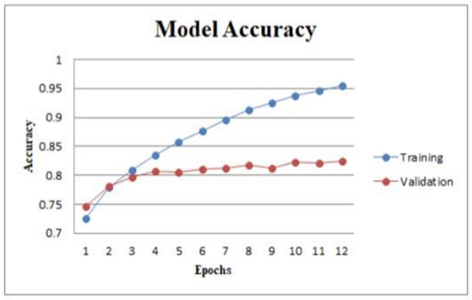
As a continuation of this research, various data augmentation techniques can be used to counter overfitting



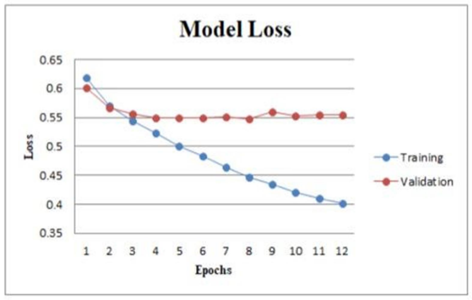
(a) Google News Vectors Embedding



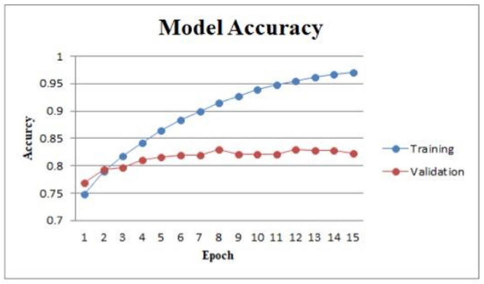
(a) Google News Vectors Embedding



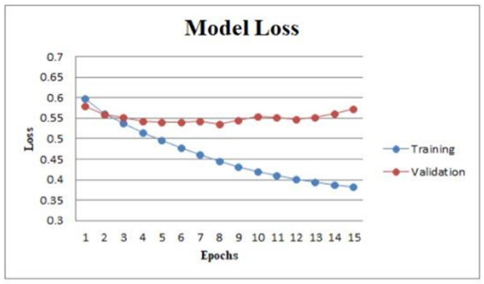
(b) FastText Wiki News (16B)



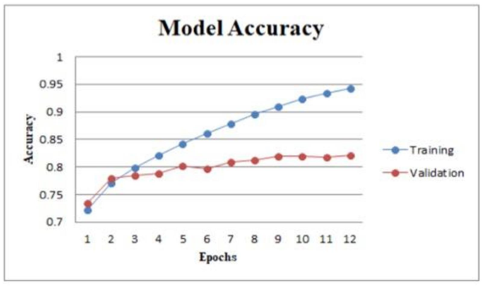
(b) FastText Wiki News (16B)



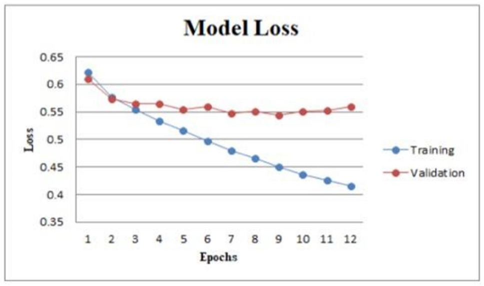
(c) FastText Crawl (600B)



(c) FastText Crawl (600B)



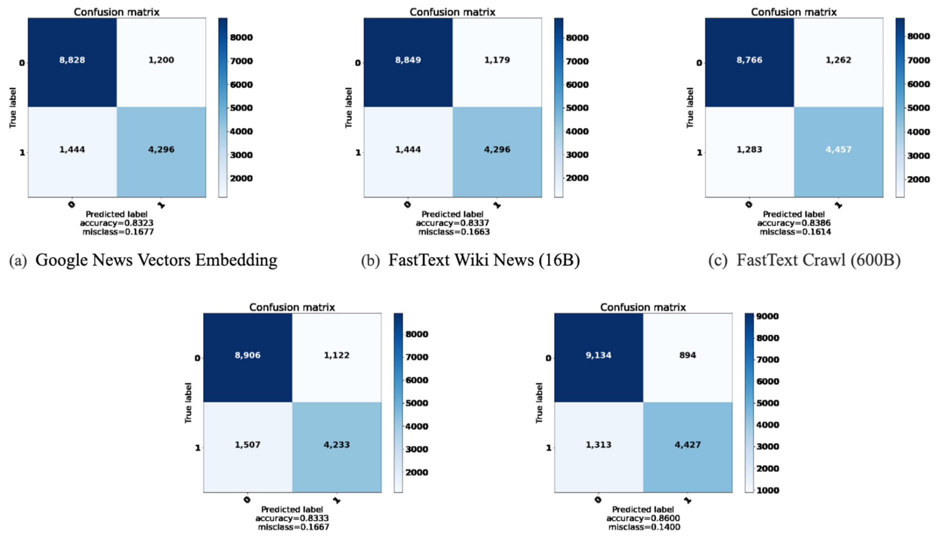
(d) FastText Crawl Subword (600B)



(d) FastText Crawl Subword (600B)

Fig. 4: Training and validation accuracies for individual word embeddings

Fig. 5 Training and validation losses for individual word embedding



(a) Google News Vectors Embedding (b) FastText Wiki News (16B) (c) FastText Crawl (600B)

(d) FastText Crawl Subword (600B) (e) Ensemble

Fig. 3: Confusion matrix of different word embeddings on test data

TABLE 3: Question Pair Classification Using Different Word Embeddings and Their Combination

Question 1	Question 2	True Label	Google News Vectors	FastText Wiki News (16B)	FastText Crawl Subword (600B)	FastText Crawl (600B)	Ensemble of 4 word embeddings
Is Kellyanne Conway annoying in your opinion?	Did Kellyanne Conway really imply that we should not pay attention to the words that come out of Donald Trump's mouth?	0	0	0	0	0	0
What are your views on the Netaji Subhash Chandra Bose's mystery death case?	What is secret of Subhash Chandra Bose's death?	1	0	1	1	1	1
How do I play Pokemon GO in Korea?	How do I play Pokemon GO in China?	0	0	0	1	0	0
What's the probability that a leap year has 53 Sundays?	What are some unknown facts about Leap Year and 29th Feb ?	0	0	1	0	1	1
Why is US mainstream media biased against Trump?	Why mainstream media hates Donald Trump?	1	1	1	0	1	1

of the proposed models during training phase after a given number of iterations. The use of different distance metrics can be used to calculate the similarity of the transformer encodings and at the same time, different transformers like BERT, ALBERT, T5-11B, which have proved most effective for the process of Text Similarity can be used in this architecture in order to attempt to produce more accurate feature extraction.

REFERENCES

[1] G. Majumder, P. Pakray, A. F. Gelbukh, & D. Pinto, (2016). Semantic textual similarity methods, tools, and applications: A survey. *Computación y Sistemas*, Vol. 20, No. 4, pp. 647–665.

[2] Dasha Bogdanova, Cicero dos Santos, Luciano Barbosa, and Bianca Zadrozny. Detecting semantically equivalent questions in online user forums. *Proceedings of the 19th Conference on Computational Natural Language Learning*, 1:123–131, 2015.

[3] Quora. What's Quora's policy on merging questions? https://help.quora.com/hc/en-us/articles/360000470663-What-s-Quora-s-policy-on-merging-questions-, 2019. Online: accessed 20 May 2020.

[4] Quora. Quora question pairs, version 1. https://www.kaggle.com/c/quora-questionpairs/data, 2017, November. Online: accessed 20 May 2020.

[5] W. H. Gomma, A. A. Fahmy, "A Survey of Text Similarity Approaches" in International Journal of Computer Applications (0975 – 8887) Volume 68– No.13, April 2013, pg. 13.

[6] P. Neculoiu, M. Versteegh, and M. Rotaru, "Learning text similarity with siamese recurrent networks," in *Rep4NLP@ACL*, 2016.

[7] Marton Mihaltz. word2vec-googlenews-vectors. https://github.com/mmihaltz/word2vecGoogleNewsvectors , 2016, May Online: accessed 20 September 2019.

[8] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. CoRR, abs/1612.03651, 2016.

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. CoRR, abs/1607.04606, 2016.

[10] E Dadashov, S Sakshuwong, K Yu, Quora Question Duplication, Stanford University (2017).

[11] Z. Imtiaz, M. Umer, M. Ahmad, S. Ullah, G. S. Choi and A. Mehmood, "Duplicate Questions Pair Detection Using Siamese MaLSTM," in IEEE Access, vol. 8, pp. 21932-21942, 2020, doi: 10.1109/ACCESS.2020.2969041.

[12] Vaswani, Ashish & Shazeer, Noam & Parmar, Niki & Uszkoreit, Jakob & Jones, Llion & Gomez, Aidan & Kaiser, Lukasz & Polosukhin, Illia. (2017). Attention Is All You Need.

[13] M. Ostendorff, T. Ruas, M. Schubotz, G. Rehm, B. Gipp, "Pairwise Multi-Class Document Classification for Semantic Relations between Wikipedia Articles" in Proceedings of the ACM/IEEE Joint Conference on Digital Libraries (JCDL), 2020.

[14] AE Yüksel, YA Türkmen, A Ozgür, AB Altınel, "Turkish Tweet Classification with Transformer Encoder" in *Proceedings of Recent Advances in Natural Language Processing*, pages 1380–1387, Varna, Bulgaria, Sep 2–4, 2019. Pg 1380-1387

[15] Alammar, Jay (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/. Online: accesses 10 June 2020

[16] A. Kazemnejad.Transformer Architecture: The Positional Encoding [Blog Post]. Retrieved from https://kazemnejad.com/blog/transformer_architecture_positional_encoding/. Online: access 20 June 2020.

[17] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. A c-lstm neural network for text classification. 11 2015.

[18] Hariharan B.R. Abishek, K. and C. Valliyammai. An enhanced deep learning model for duplicate question pairs recognition. Springer, 2019.

[19] Rafael Mu¨ller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019.

[20] Łukasz Kaiser and Samy Bengio. Can active memory replace attention? In Advances in Neural Information Processing Systems, (NIPS), 2016.

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