Semantic Text Similarity System along with Explainable AI

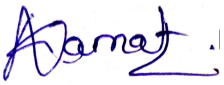
OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

ANAGHA VINAYAK KAMAT
School of Computer Science
University of Galway

Supervisor(s)
Dr. Matthias Nickles

In partial fulfillment of the requirements for the degree of
MSc in Computer Science (Data Analytics)
31st August, 2023

DECLARATION I, Anagha Vinayak Kamat, hereby declare that this thesis, titled "Semantic Text Similarity System along with Explainable AI", and the work presented in it are entirely my own except where explicitly stated otherwise in the text, and that this work has not been previously submitted, in part or whole, to any university or institution for any degree, diploma, or other qualification.

Signature: 

Abstract

This research focuses on obtaining human-interpretable explanations for obtaining semantic text similarity between 2 sentences using BERT based transformers for generating sentence level encodings and explaining the output using LIME model. The results obtained from this research are satisfactory and LIME could successfully give explanations for the decision made. Explanations are visualized at word level where each word in the sentence is highlighted in Blue and Orange color shades depending on the semantic similarity index shared between both sentences. This research could successfully achieve the problem of identifying ambiguity error in NLP tasks. However, if one needs explanations for a more advanced and complex operation such as visualizing the internal operations for obtaining embeddings from an encoder, then this approach is incapable of doing it.

Keywords: Semantic Text Similarity, Cosine Similarity, Lime, Explainanle AI, BERT, Sentence Transformer, Ambiguity Error

Contents

List of Acronyms **viii**

1 Introduction **1**

1.1 Motivation 1

1.2 Scope 3

1.3 Research Questions 3

1.4 Structure of thesis 3

2 Background Research and Literature Review **4**

2.1 Semantic Text Similarity 4

2.1.1 Transformer for embeddings 4

2.1.2 Vector Representation of Words 7

2.1.3 Deep Learning for STS 8

2.2 Explainable AI 9

3 Data **12**

4 Methodology **14**

4.1 Semantic Text Similarity 14

4.1.1 Sentence Transformer 14

4.1.2 Sentence Transformers used in this research 16

4.2 Explainability using LIME 17

5 Experiments **19**

5.1 Part 1: Experimenting with Models 19

5.2 Part 2: Public Review for Trustworthiness 20

6 Results **22**

6.1 Experimenting with models 22

6.1.1 Model 1 33

6.1.2 Model 2 34

6.1.3 Model 3 34

6.1.4 Model 4 35

6.1.5 Model 5 35

6.2 Analysis on the implementation results 36

6.3 Analysing the survey results 37

7 Conclusion **40**

7.1 Limitations 41

7.2 Future Work 41

References **47**

List of Figures

2.1 Structure of BERT input embedding [1]. 5

4.1 Structure of SBERT [2]. 15
4.2 Implementation steps for LIME algorithm 18

6.1 Result obtained from "all-MiniLM-L6-v2" for sentence pair 1 . . . 23
6.2 Result obtained from "all-MiniLM-L6-v2" for sentence pair 2 . . . 23
6.3 Result obtained from "all-MiniLM-L6-v2" for sentence pair 3 . . . 23
6.4 Result obtained from "all-MiniLM-L6-v2" for sentence pair 4 . . . 24
6.5 Result obtained from "all-MiniLM-L6-v2" for sentence pair 5 . . . 24
6.6 Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 1 24
6.7 Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 2 25
6.8 Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 3 25
6.9 Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 4 25
6.10 Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 5 26

6.11 Result obtained from "paraphrase-albert-small-v2" for sentence pair 1 26
6.12 Result obtained from "paraphrase-albert-small-v2" for sentence pair 2 27
6.13 Result obtained from "paraphrase-albert-small-v2" for sentence pair 3 27
6.14 Result obtained from "paraphrase-albert-small-v2" for sentence pair 4 27
6.15 Result obtained from "paraphrase-albert-small-v2" for sentence pair 5 28
6.16 Result obtained from "all-mpnet-base-v2" for sentence pair 1 . . . 28
6.17 Result obtained from "all-mpnet-base-v2" for sentence pair 2 . . . 28
6.18 Result obtained from "all-mpnet-base-v2" for sentence pair 3 . . . 29
6.19 Result obtained from "all-mpnet-base-v2" for sentence pair 4 . . . 29
6.20 Result obtained from "all-mpnet-base-v2" for sentence pair 5 . . . 29
6.21 Result obtained from "all-MiniLM-L12-v2" for sentence pair 1 . . 30
6.22 Result obtained from "all-MiniLM-L12-v2" for sentence pair 2 . . 30
6.23 Result obtained from "all-MiniLM-L12-v2" for sentence pair 3 . . 30
6.24 Result obtained from "all-MiniLM-L12-v2" for sentence pair 4 . . 31
6.25 Result obtained from "all-MiniLM-L12-v2" for sentence pair 5 . . 31
6.26 Survey result on incorporating AI with XAI 39
6.27 Survey result on trusting general AI and XAI 39
6.28 Survey result obtaining transparency in an AI system using XAI . 39

List of Tables

4.1 Specification of models used in the experiment when tested on semantic search and average performance of sentence embeddings for different tasks [3] 16

5.1 Table showing the example sentences tested to check the model's efficiency. The first two column are the example sentences used in this research, the third and fourth column shows whether both of them are semantically and syntactically similar or not. 20

6.1 Table showing the results of example sentences tested to check the model's efficiency. The first column is the model used refer Table 4.1, the second column is the sentence pair refer Table 5.1, the 3rd column is actual semantic similarity, the fourth column is predicted semantic similarity and the 5th column is percentage of correctly predicted sentence similarity by a model. 32

List of Acronyms

AI Artificial Intelligence. vi, 2, 9, 21, 37–39, 41

BERT Bidirectional Encoder Representations from Transformers. ii, v, 4–6, 15

biLSTM Bidirectional long-short term memory. 7, 8

bioBERT Bidirectional Encoder Representations from Transformers for biological data. 8, 9

BioSentVec Biomedical sentence embeddings. 8, 9

CNN Convolutional Neural Network. 8

GLUE General Language Understanding Evaluation. 12, 13

JD Job Description. 7

LIME Local Interpretable Model-Agnostic Explanations. ii, iv, v, 9, 11, 17, 18, 36, 40, 41

LSTM long-short term memory. 10

MRPC Microsoft Research Paraphrase Corpus. 12, 13

MSE Mean Squared Error. 9

NLP Natural Language Processing. 3, 4, 12, 36

NN Neural Network. 10

RoBERTa Robustly Optimized BERT. 6

SBERT Siamese BERT. v, 6, 7, 10, 15

SHAP SHapley Additive exPlanations. 9, 10

SOTA State-of-the-Art. 14

STS Semantic Text Similarity. iii, 2, 3, 6, 8, 12, 14–16, 19, 36

STSB Semantic Textual Similarity Benchmark. 12, 13

USE Universal Sentence Encoder. 7, 10

XAI Explainable Artificial Intelligence. vi, 1–3, 9–11, 20, 21, 36–39

XSTM Explainable Semantic Text Matching. 11

Chapter 1

Introduction

1.1 Motivation

Text/Document-based Similarity in Natural Language Processing is an evolving field that is growing day by day. At present times, it is essential not only to find similar texts or documents based on the words used but also to find the semantic similarity between any document. Semantic similarity means understanding the core meaning of any document and finding how similar documents are semantically. Still, sometimes it gets difficult to trust a machine's judgment, and thus a question rises about what resulted in a machine giving a particular output. This question can often be answered using Explainable Artificial Intelligence (XAI). XAI is basically a set of different processes or methods that allow the users to understand the decision-making process of the model and trust its decision [4].

Text Similarity has various use cases in the real world [5]. Starting from the Google Search Engine to any e-commerce website, every platform uses this method to maintain its qualitative level in the growing market. But while talking about such platforms one thing that needs to get attention is, most of the platforms are currently making use of basic text similarity techniques i.e. only

word based. Sometimes understanding meaning is much more effective than just comparing the usage of any keywords.

Semantic Text Similarity (STS) [6] is an important aspect in various fields. For instance, a sarcasm detection model can be a good example in this scenario [7]. In natural language, while passing sarcastic comments, humans use positive sentence structures that can have negative meanings. In this case, it is important the model focuses on the entire statement passed by the user instead of focusing on just particular words used. For example, if a movie review states, "The movie was fantastic! I wonder how the director could successfully achieve its target of making people bang their heads for deciding to watch this movie." In this sentence, we can see the user has given a sarcastic comment and expressed his negative feelings using positive words. Semantic Sentence similarity's job here would be focusing on terms such as bang, heads, deciding, and watch which will then sort the issue of ambiguity and give correct outputs.

But, just developing semantic similarity is not satisfying for a user. The user needs more reasons to trust a machine's decision which brings us to our main concern of using XAI in this research [4]. For instance, if we look into the medical field, if a machine learning algorithm is built to identify the medical problem behind the user's statements of expressing illness, it is also very important for the user to understand why the machine has given that output. Explainable AI can play an important role by giving human-understandable explanations for the resulting output. E.g., if a user says, "I have swelling in my knuckles and pain in my joints" and the AI model result states, "There's a possibility of you having Arthritis" then the user would require a human understandable explanation of the result to trust the system. Thus, the XAI approach will help build human trust in such systems by giving useful explanations to the user.

1.2 Scope

Semantic Text Similarity (STS) in NLP deals with finding similarities between texts or documents in terms of the meaning without focusing only on the frequency of repeated words in a sentence. Meanwhile, XAI focuses on developing algorithms that provide a clear explanation for giving a particular output promoting better human understanding. With this research, we will try to achieve more accurate results and provide insights to the users into why or why not certain texts were considered to be semantically similar to each other.

1.3 Research Questions

RQ1. How can the XAI technique improve the overall result of any existing STS algorithm and support transparency and interoperability?
RQ2. Can we use XAI to identify errors such as ambiguity(similar words with different meanings)?
RQ3. Do users trust an XAI model more than a basic AI model?

1.4 Structure of thesis

The thesis is divided into 7 chapters. The first chapter as you saw included a basic introduction about the research area along with some research questions to be considered. The second chapter will include background research in the respective field along with a detailed literature review of previous researches made in this area. Chapters 3 to 6 will discuss the data used, the methodology of the thesis, the experiments carried out, and the results obtained. Lastly, chapter 7 will give a conclusion on the entire topic and also discuss any potential future work that may arise following this research.

Chapter 2

Background Research and Literature Review

2.1 Semantic Text Similarity

Semantic similarity in NLP focuses on the degree of relatedness between texts based on their meaning. To obtain semantic text similarity we first need to find the word embeddings. This can be done using various techniques as follows.

2.1.1 Transformer for embeddings

There are various transformers that we can use to obtain embeddings for a semantic text similarity model. BERT based models are commonly used transformers for performing State-Of-The-Art NLP tasks. Bidirectional Encoder Representations from Transformers (BERT) [1] is a bi-directional encoder language model. A bi-directional transformers takes information from both sides of the tokens. For e.g "I am going to bank for depositing money" and "I am sitting at a beautiful river bank" have totally different meanings with same word "bank". BERT will thus learn from the surrounding tokens as well while training and then make pre-

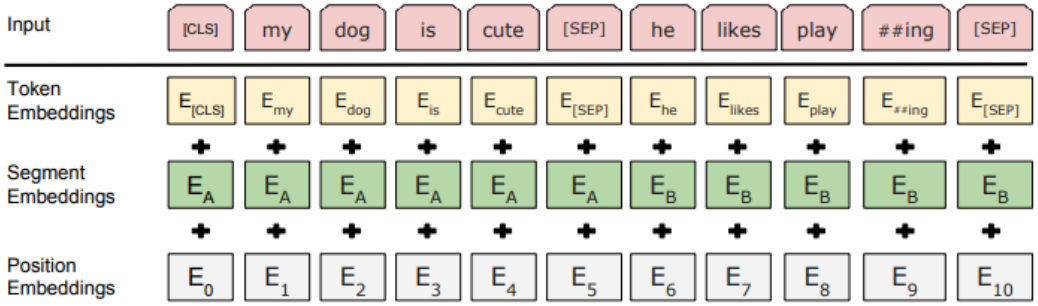


Figure 2.1: Structure of BERT input embedding [1].

diction. BERT only has encoder part and the embeddings generated are used for other different tasks as and when required. Before pre-processing, BERT follows certain steps.

- Token embeddings: A [CLS] token is added at the beginning of every sentence which checks whether the 2nd part of the sentence is directly related to the 1st part of the sentence or not. A [SEP] token is added at the end of the sentence.
- Segment embeddings: It denotes that sentences are added to each token which enables the encoder to distinguish between sentences.
- Position embeddings: A locator is added to each token that points to the location of a token in that sentence.

Transformers maps the layer in a sequential manner such that the input and output tokens are mapped together sequentially in a 1:1 format. BERT is trained based on 2 strategies. First it performs Masked Language Model (MLM) which randomly replaces 15% of the input tokens with [MASK] token. Later, it runs the sequence through BERT attention based encoder and predict [MASK] tokens based on the context of non-masked words in a sentence. Second strategy is Next Sentence Prediction (NSP) where in it will first take a number of input sentences

and predict which sentence follows the first sentence in the original paragraph as well. The next input sentence is then labelled as next or not after prediction.

BERT has mainly 2 types of architecture viz [1]. BERT-Base and BERT-Large. The BERT-Base is a 12 layer architecture with 768 hidden layers, 12 attention mechanism, and 110M parameters. On the otherhand, BERT-Large is a 24 layer architecture with 21024 hidden nodes, 16 attention mechanism and 340M parameters. The BERT model is trained on unlabelled plain text English Wikipedia corpus and the Brown corpus. Due to this, it took 4 days for BERT-Base to train on four TPUs. On the other hand, BERT-Large was trained for 4 days on 16 TPUs. Read [1] for more details.

Siamese BERT (SBERT) [2] generates sentence-level embeddings for a document using siamese or triplet network models. Detailed development of SBERT is described in. The authors of this paper developed the SBERT transformer by adding a pooling operation to the output of BERT and Robustly Optimized BERT (RoBERTa) [8] to obtain a specific-sized sentence embedding. They carried out semantic similarity for STS-specific supervised and unsupervised data and used sentEval [9] to evaluate the quality of the embeddings generated. The computational speed of SBERT was compared with GloVE [10], InferSent [11], and Universal Sentence Encoder (USE) [12] and it seemed that SBERT was 9% faster than InferSent on GPU, and around 55% faster than Universal Sentence Encoder. Although, GloVe seemed to be the fastest of all on both CPU and GPU.

The authors of paper [13] have experimented with clinical STS using 3 transformers viz. BERT, XLNet [14], and RoBERTa. The model was tested on both the general English corpus and the clinical corpus. It seemed that RoBERTa-large outperformed the test result with a Pearson correlation score of up to 0.905. Although, it occurred that the model could perform better for general English STS as compared to clinical STS.

The paper [15] discusses a new approach for developing sentence embeddings for unlabelled data. The authors have developed a Transformer-based Sequential Denoising Auto-Encoder(TSDAE) that uses an encoder-decoder architecture that overpowers previous methods such as InferSent, SBERT, and Universal Sentence Encoder (USE) [12] for sentence embeddings. The authors have also mentioned that this novel approach has outperformed the previous best approach by up to 6.4 points on diverse datasets.

2.1.2 Vector Representation of Words

Another method is representing words in a vector space. Various vector representation techniques mentioned are as follows.

GloVe: GloVe [10] is an unsupervised algorithm for obtaining vector representations of words. It is trained on global word-word co-occurrences statistics from a corpus. It shows the linear substructures of words in vector space. The model is built on word-word co-occurrences that focus on how frequently a particular word occurs with the other word. See [10] for more details.

Word2Vec [16]: The Word2Vec is an Artificial Neural Network that converts texts from a large corpus where each unique word from the corpus is represented in a vector space.

The authors of [17], have used biLSTM [18] and Siamese networks [19] to carry out semantic text similarity between a target and the text that needs to be matched. The model consists of 4 layers: input, embedding, biLSTM, and the Siamese layer. The dataset used is a job description and resumes of 70 candidates belonging to 3 fields: Data Scientist, Web Developer, and Software Developer. The embeddings are obtained by Word2Vec. Further, they used the TextRank algorithm to find the most similar or the most important sentence in the document. Later on, they shortlisted matching sentences between resumes and JD and then compared those

texts to other texts in the resume. With this step, they obtained 9 texts from resumes which they used as human-interpretable explanations for their algorithm.
FastText [20]: Another method similar to Word2Vec and GloVe is FastText which again is used to obtain vector representations of a word. The difference is, this method captures important information about a language such as analogies or semantics. FastText offers 2 models for computing embeddings. The first model is the skip-gram model that predicts a word depending on the neighboring words. The second model is a CBOW model that predicts a word depending on the context where the context is a bag of words of a fixed size. Refer to [20] for more information.
A research was carried out in [21], where FastText was compared with GloVe and Word2Vec models for obtaining better word embeddings. The comparative study showed that GloVe and fastText could perform better with fastText being the best performer. The research stated that fastText became the best-performing algorithm because of the limited number of Out Of Vocabulary words found during the experiments conducted for fastText.

2.1.3 Deep Learning for STS

Deep Learning techniques can also be used for performing semantic matching of sentences. Models such as biLSTM [18] and Convolutional Neural Network (CNN) [22] are common approaches for carrying out STS. Authors of paper [23] performed a comparative analysis for semantic text matching in the clinical domain. They divided their research into 3 sections, deep learning-based (CNN), sentence vector-based (BioSentVec [24]), and Transformer based bioBERT [25], Blue BERT [26], and Clinical BERT [27]). Furthermore, they also carried out research on general machine learning models viz. Random Forest as an additional baseline. Evaluations were made using Pearson correlation, Spearman correlation,

R^2, and MSE [28]. The top-performing models were BioSentVec and bioBERT while RandomForest showed poor performance. Although the rest of the models had almost the same similarity index. The computation time for the BERT models was relatively slow than deep learning and BioSentVec models.

2.2 Explainable AI

Explainable Artificial Intelligence (XAI) is a latest field in AI and Machine Learning. It focuses on giving visual explanations for the results produced by different machine learning models. Two most commonly used XAI model are SHapley Additive exPlanations (SHAP) [29] and Local Interpretable Model-Agnostic Explanations (LIME) [30]. XAI is needed when users demand model transparency. There are various usecases of XAI at present times [31]. To name a few:

- **Transparency and trustability**: XAI offers transparency with the system by giving human-interpretable explanations for a particular result produced by a system. For instance,let's say in legal domain, there is a model built to classify a person as guilty or non-guilty depending on different variable aspects. When the system will give a particular result taking different aspects as input, the user would also need explanations to trust the machine. Using Explainable AI, the user will grow more trust and the machine will become transparent.

- **Error and Bias Detection**: XAI can help detect errors occurring in a machine. For e.g, if a model predicts incorrect decision, XAI will explain the model's behavior and learning pattern. With this, we can fix issues and improve the training process. If a model is creating biases, XAI can show the model's learning which can help understand that the model is being biased.

- **Adapting to new situations**: As XAI allows us to look into the learning process of an AI system, it enables us to understand the complex decision making process. We can then use this knowledge in customizing the model and making it adaptable to even more diverse situations.

Since XAI is still in its research and development stage, even after having so many usecases there are a lot of struggles involved while working with these systems. Few of the challenges are listed ahead [32].

- **Model complexity:** XAI cannot handle complex models such as Deep Neural Networks or Convolution Neural Networks. There are chances that explanations produced for these models can be high-level which are difficult for humans to understand.

- **Explanation Technique:** Multiple explanations can be produced by using different XAI models. This can result in confusions and inaccurate interpretations.

- **Easy understandability:** It is not necessary that explanations provided are always understandable for a non-technical users. There are chances that explanations provided by such models would make sense for machine's working but humans would not relate.

- **Data privacy:** XAI models can release sensitive data that will affect the organizational privacy.

In [33], the authors have used USE and Sentence Bidirectional Encoder Representations from Transformers (SBERT) to obtain embeddings on the Twitter emoji data. Later, they used these embeddings to train the standard Neural Network (NN) and LSTM NN model to classify the sentiment in the text. Further, the SHAP [29] algorithm of XAI was used to generate explanations.

The [34] authors have proposed a new model XSTM inspired by Local Interpretable Model-Agnostic Explanations (LIME) method to gain Explainable Semantic Text Matching. They obtained words from texts and carried out a sensitivity analysis to obtain the effect of individual words in semantic text matching which then self-explained the model.

Lastly, one more novel approach to semantic matching with XAI has already been discussed previously in Section 2.2 [17].

Chapter 3

Data

The dataset used for this experiment is Microsoft Research Paraphrase Corpus (MRPC) dataset that can be found in General Language Understanding Evaluation (GLUE). GLUE is a repository that has access to various resources for training, evaluating, and analyzing NLP systems with high quality data [5] [35].

The MRPC dataset designed by Microsoft is a collection of sentence pairs which are first automatically collected from online news resources and then manually annotated as semantically similar or not. The training set has 3,668 annotated sentence pairs, the test set has 1,725 pairs, and the validation set has 408 pairs [36] [37].

In this research, we have used only the training set to obtain embeddings that can be further used for obtaining semantic similarity scores and finding explanations.

There are 2 datasets provided by GLUE developed for STS tasks viz. STSB and MRPC. When I manually went through some of the sentence pairs, it was found the quality of mrpc dataset was better than that of the other one because in the second dataset the sentence pairs were semantically and syntactically similar in majority cases. While in MRPC dataset, the sentences are semantically similar

and syntactically dissimilar.

To give some examples, the first pair of similar sentences are "The exam contains four sections and tests a students knowledge of algebra and geometry as well as probability and statistics" and "The test , in four sections , includes algebra and geometry , along with some questions on probability and statistics" which was labelled as 1 are semantically similar but syntactically different. Another example from this dataset would be, "Sen. Bob Graham , Florida Democrat , raised $ 2 million after getting a late start" and "Further back , Sen. Bob Graham of Florida reported about $ 1.7 million on hand" share a lot more similar words but actually mean different which were labelled as 0.

In case of the 2nd one which is STSB, an example sentence would be, "A man is spreading shreded cheese on a pizza." and "A man is spreading shredded cheese on an uncooked pizza." are semantically similar and even both the sentences share maximum similar words. Although, instead of labelling the data into binary format, it was ranked from 0 to 5 with 5 being completely similar and 0 being completely dissimilar. But since, I felt it was poorly labelled, as in the case of the example given above which was labelled to 3.8, I felt choosing the mrpc dataset over this one in terms of labelling quality and ease of the task.

Looking at such examples, it was concluded to use the GLUE MRPC dataset instead of STSB. For pre-processing, data has been cleaned by removing all the rows with empty sentences. The sentences are cleaned by performing punctuation removal techniques. Although, as such any references weren't seen in the output results before and after pre-processing.

Chapter 4

Methodology

The research is divided into 2 parts. The first part involves training the model on 5 different transformer models for finding the semantic similarity between the sentence pairs. The second part involves using explainable AI to obtain visual explanations for the output generated by the models that are readily understandable by a human with non-technical background.

4.1 Semantic Text Similarity

Here, different sentence transformers are used to train the data and obtain embeddings. These embeddings are further used to obtain cosine similarity between the two sentences.

4.1.1 Sentence Transformer

Sentence transformers [2] are python frameworks that can be used to obtain State-of-the-Art (SOTA) embeddings for sentences. We can use these transformers to perform multiple tasks including semantic text similarity or semantic search or paraphrase checking.

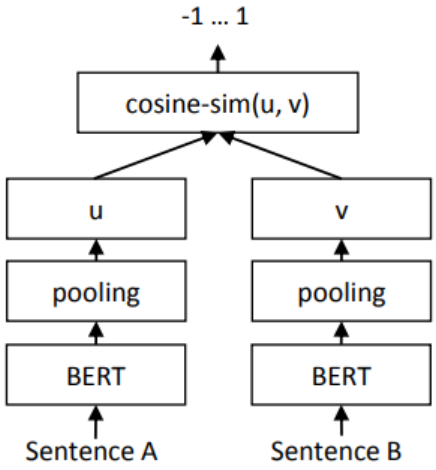


Figure 4.1: Structure of SBERT [2].

There are 2 variations of SBERT model, one is for classification related tasks and another is for basic tasks such as semantic search or STS. Figure 4.1 shows the structure of SBERT for basic semantic text similarity kind of tasks. Although, SBERT model can be used for classification tasks as well which then will have a different structure than the one in Figure 4.1. Initially, the input sentences will be passed to the BERT encoders which will generate embeddings for each sentences. Later, those embeddings will be passed to the mean pooling layer that will give the mean pooled vectors. Using those vectors we can then calculate the cosine similarity of both the sentences. For a detailed explanation on the model refer [2]. Unlike BERT, SBERT produces sentence encodings and is finetuned on Siamese network architecture. As seen in Figure 4.1, we can see a Siamese network architecture that have 2 BERT models in parallel, that process 2 sentences individually. As discussed earlier, the BERT model is just an encoder that creates encodings which can be used for different tasks. So these encodings created by BERT encoder are passed to the pooling layer that reduces the dimensions of those embeddings which are then used for different tasks.

4.1.2 Sentence Transformers used in this research

For this research, we use five different sentence transformers and test the output result for each of them.

Model	Transformer	Avg Performance	Speed	Model Size
1	all-MiniLM-L6-v2	58.80	14200	80MB
2	paraphrase-MiniLM-L6-v2	52.56	14200	80MB
3	paraphrase-albert-small-v2	52.25	5000	43 MB
4	all-mpnet-base-v2	**63.30**	2800	420MB
5	all-MiniLM-L12-v2	59.76	7500	120MB

Table 4.1: Specification of models used in the experiment when tested on semantic search and average performance of sentence embeddings for different tasks [3]

In the Table 4.1, first column is the transformer used, second column is the average of performance of the transformer in semantic text matching and average performance in other tasks. The third column is the processing speed that shows number of sentences encoded per seconds. The last column gives the model size. The first 2 models were chosen since they have the best processing speed than the rest. Nonetheless, as the name suggest "paraphrase-MiniLM-L6-v2", this model is particularly aimed for paraphrased texts analysis which makes it a right choice for using it in this research. Since, our focus was implementing explainable semantic similarity for English language at the moment, the next model we decided was "paraphrase-albert-small-v2" since, it was the best performing model for English language. The "all-mpnet-base-v2" model was chosen as it showed the best quality output for semantic search. The 5th model "all-MiniLM-L12-v2" is the 4th best model according to the source for semantic similarity task and it could also give better results of all. Hence that model was chosen for this research. All the information about the models in the Table 4.1 is taken from [3].

4.2 Explainability using LIME

Local Interpretable Model-Agnostic Explanations (LIME) [38][39] is one of the commonly used approach for obtaining explainability for the particular AI model. LIME provides human understandable visual explanations for different machine learning tasks. For this research, we have used LimeTextExplainer() method from the lime module. This method explains text classifiers based on the words present in the sentences which is are primary goal. LIME works in following ways:

- Initially LIME identifies the important data points that resulted for a model to give a certain output and then, make slight changes in those data points.
- LIME produces predictions for the changes made into the data points. The difference obtained gives an insight on the performance of the model with respect to a specific input feature.
- Further it uses an interpretable model which provides coefficients for the impact of each feature on the model prediction.
- As an output, it presents highly important features and their corresponding impact on the users which is easily readable by humans.

Refer [38] and [39] for a more detailed explanation on LIME model.

For providing explanations, we are first calculating embeddings and predicting the similarity score. Further, this output is carry forwarded to the LIME model wherein, we are initially defining two classes or prediction "Paraphrased" and "Not Paraphrased" using LimeTextExplainer method of lime module. Later, in the explain_instance method, we pass the texts that we want to compare, pass the similarity score model, and define the maximum number of features present.

The Figure 4.2 is a code snippet for implementing LIME explanation for our research. The implementation is solely a part of this research but it is referred

```
def explain_prediction(text1, text2):
    explainer = lime_text.LimeTextExplainer(class_names=["Not Paraphrase", "Paraphrase"])

    exp = explainer.explain_instance(
        "\n".join([text1, text2]),
        obtain_cos_sim,
        num_features=10,
    )
    exp.show_in_notebook(text=True)
```

Figure 4.2: Implementation steps for LIME algorithm

from [40]. Here, a explain_prediction() function is created which takes 2 sentences as arguments. In the first step inside the function, we are creating 2 classes that check whether the 2 sentences are Paraphrased i.e. semantically similar or not. The explain_instance() method takes the two sentences to be compared, a model that calculates the similarity score of those sentences and lastly, it takes the maximum number of features that can be present in the explanation. The last line of the code snippet give the visual output of the explanation highlighting the words that resulted for the particular prediction using blue and orange colors. Since, LIME is still in it's development phase, there are no facilities for changing the visual effects produced are provided. Although, the results are understanding enough at the moment but certain improvements can be brought in it in future.

The model was implemented on Google Colab since it enables us to use T4 GPU for free. It is useful for handling heavy cloud workloads including Machine Learning, Deep Learning, Data Analytics and Visualization. Since transformer models are heavy and takes lots of time for training, we choose this platform for development.

Chapter 5

Experiments

The experiment was also carried out in 2 parts. The first part was to check the working of 5 different models with the Explainable AI results and noting the performance for each of them. The second part was a survey carried out among people from different backgrounds to obtain answer for our RQ3.

5.1 Part 1: Experimenting with Models

The STS task was carried out using 5 different models to obtain the best performing model for semantic search types of tasks. Results were tested on 5 pairs of input sentences based on their syntactic and semantic similarities. To answer RQ2, 5th sentence pair can be used to check whether the system is able to identify the word ambiguity or not.

Sentence 1	Sentence 2	Semantics	Syntax
I love to watch a lot of movies	I hate to watch movies	No	Yes
I love all animals	Dog is my favorite animal	Yes	No
I love birds	I love peacock	Yes	Yes
Rini is my childhood friend	I recently met Rini	No	No
I lost my watch	I will watch out for you	No	Yes

Table 5.1: Table showing the example sentences tested to check the model's efficiency. The first two column are the example sentences used in this research, the third and fourth column shows whether both of them are semantically and syntactically similar or not.

5.2 Part 2: Public Review for Trustworthiness

A survey was carried among a group of 27 people from different backgrounds to find the answer for RQ3. viz. do people trust Explainable AI over general AI systems? A series of questions were asked to check their knowledge and obtain views about the system. This survey was carried out on Google Forms.

- **Which domain are you working?:**
  6 domains were given as an option to the user to choose. These were **IT and Computer Science, Business, HR, Legal, Medical, Banking**. People not belonging to any of these category would choose **"other"** option.

- **Have you ever used a system that uses Artificial Intelligence?**
  This question was asked to check the user's knowledge on an AI system?

- **Do you know or worked with an AI system that uses Explainable AI to explain reasons behind the predictions made?**
  This question was asked to identify the user's knowledge on an XAI system?

- **Do you believe every AI system should have an incorporated Explainable AI technology with it to validate the end results given by the system?**

This question answers the fact that whether using XAI with a general AI is important and a necessity or not? The options given to the user were, **Every AI system should compulsorily have an explanation system with it, It is good to use Explainable AI but it is not necessary, No need of Explainable AI with a general AI system**. Furthermore, they were also asked to answer why did they go for that particular choice.

- **How much would you trust an Explainable AI technology over a general AI?**

  This question answers our RQ3. The options given to the users were in a ratio of certain percentage. There were five choices given to the users. **XAI 70-100% : General AI 40-70%, XAI 40-70% : General AI 70-40%, XAI 50% : General AI 50%, Both system ¡ 40%, Don't trust AI system** This question was also accompanied with the users reasoning their choice for this question.

- **Do you think Explainable AI can bring model transparency and inter-operability?**

  This question partially answers our RQ1. People were asked to choose between **Yes and No** and give reasons for their choices.

Chapter 6

Results

The results for our experiments has 2 parts. The first part of the results will show the results obtained from the technical research carried out with the help of our models. The second part focuses on the survey among the people.

6.1 Experimenting with models

Let's see the results obtained by all different models. The model quality was judged based on the model performance i.e. checking whether the model correctly classified the sentences as paraphrased or non-paraphrased. It is not possible to give loss and time taken to train the model since we are not training a particular model in initial stage. Instead we are calculating the embeddings by encoding the sentence pairs and then using those embeddings we are calculating semantic similarity using simple cosine similarity method.

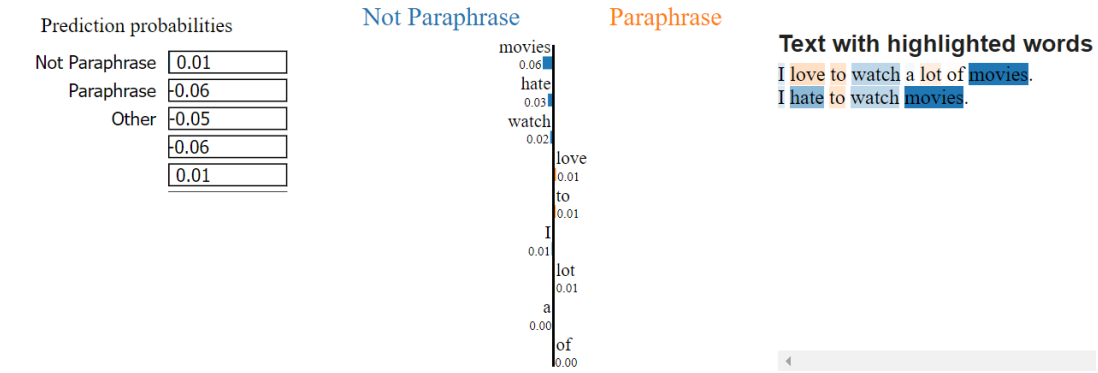


Figure 6.1: Result obtained from "all-MiniLM-L6-v2" for sentence pair 1

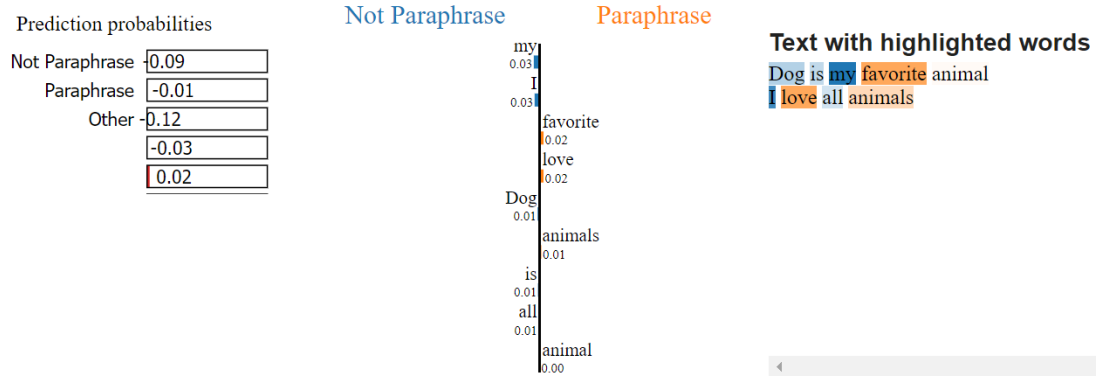


Figure 6.2: Result obtained from "all-MiniLM-L6-v2" for sentence pair 2

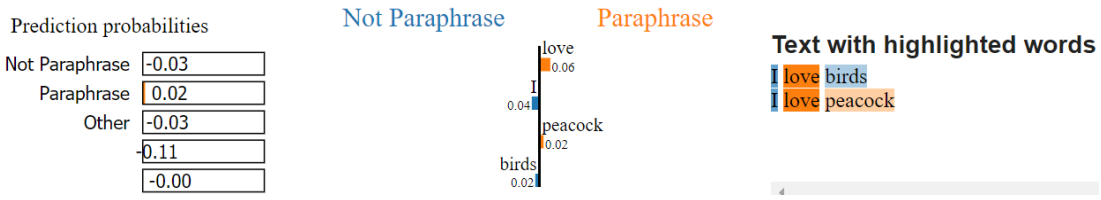


Figure 6.3: Result obtained from "all-MiniLM-L6-v2" for sentence pair 3

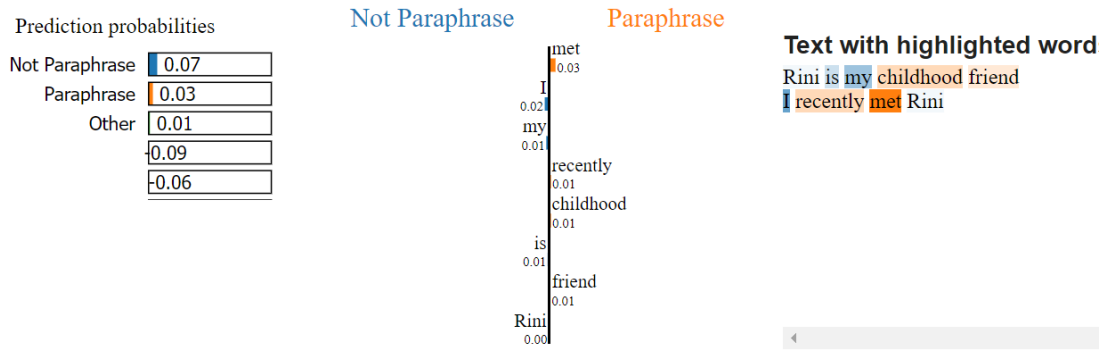


Figure 6.4: Result obtained from "all-MiniLM-L6-v2" for sentence pair 4

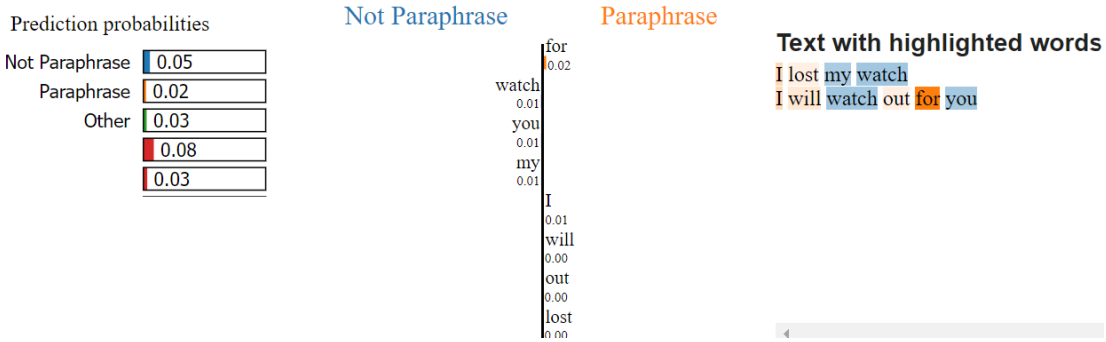


Figure 6.5: Result obtained from "all-MiniLM-L6-v2" for sentence pair 5

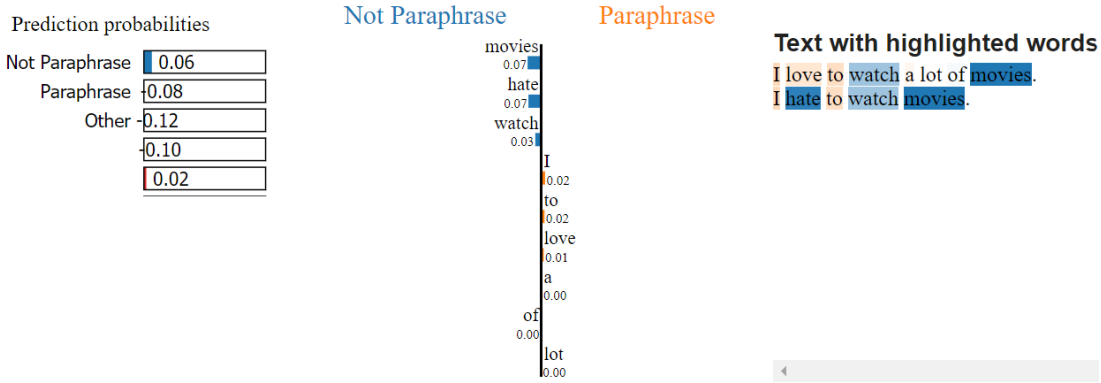


Figure 6.6: Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 1

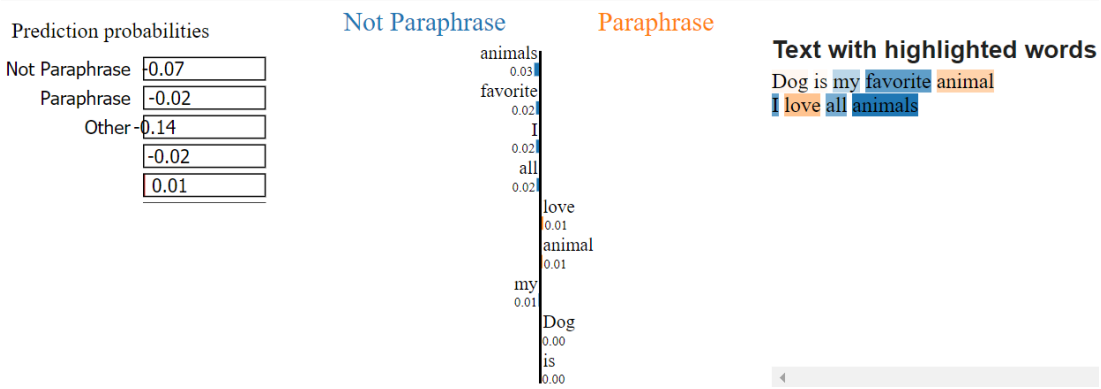


Figure 6.7: Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 2

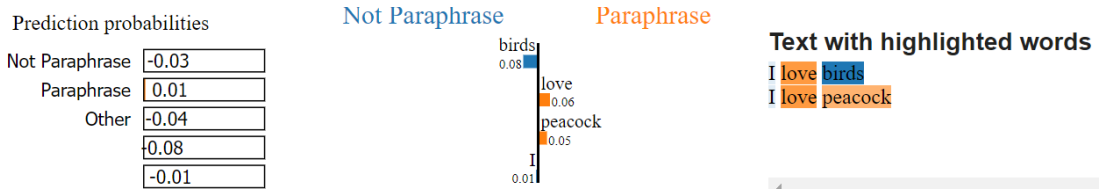


Figure 6.8: Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 3

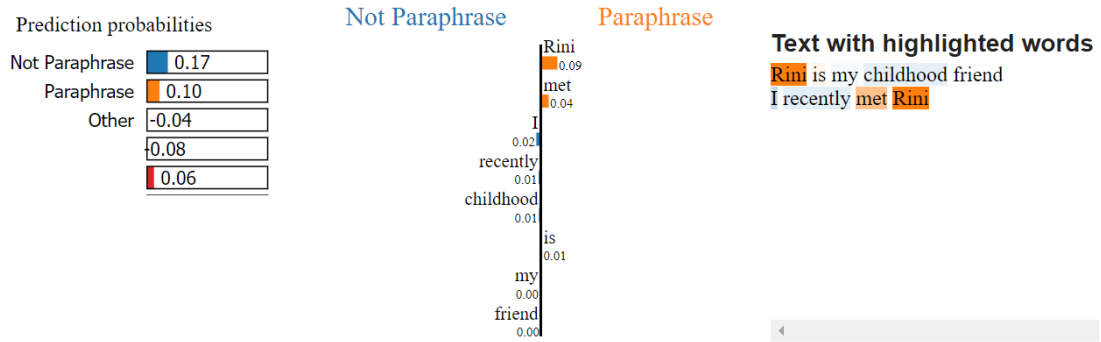


Figure 6.9: Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 4

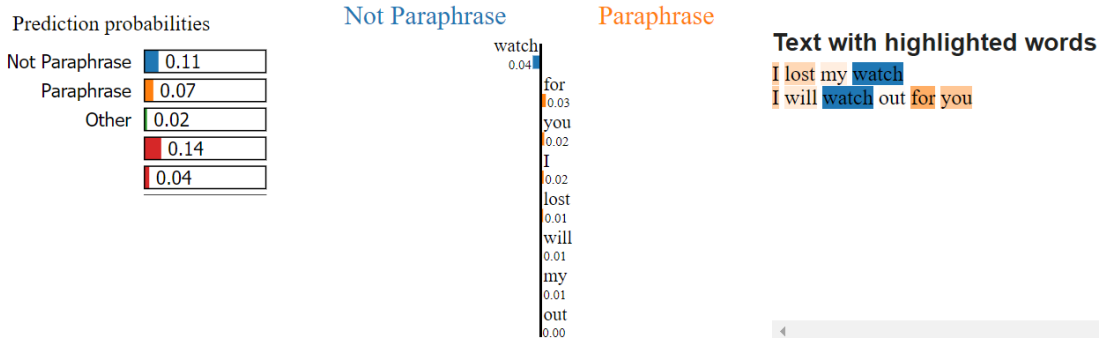


Figure 6.10: Result obtained from "paraphrase-MiniLM-L6-v2" for sentence pair 5

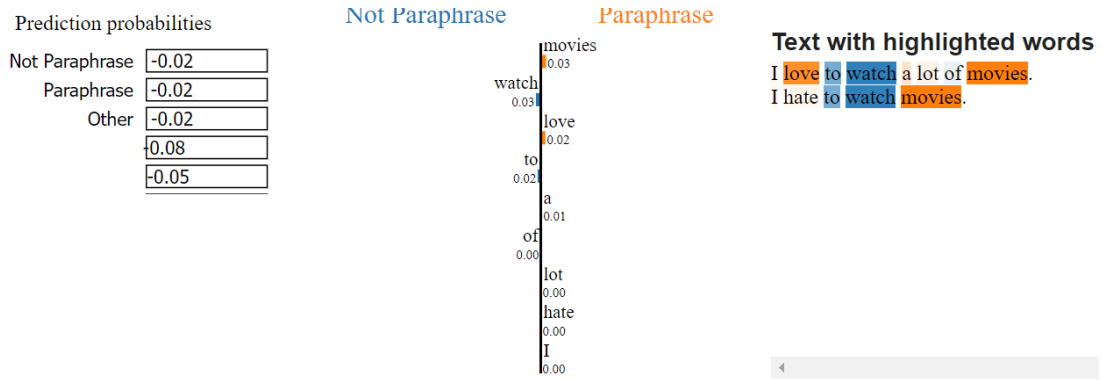


Figure 6.11: Result obtained from "paraphrase-albert-small-v2" for sentence pair 1

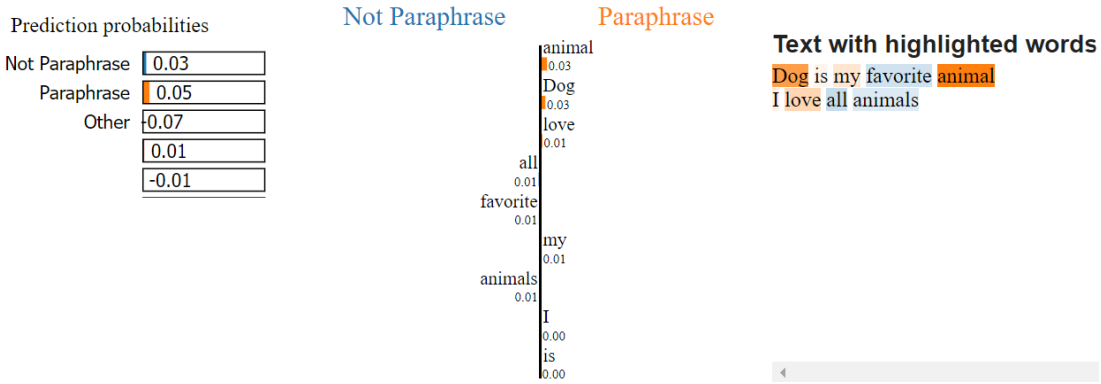


Figure 6.12: Result obtained from "paraphrase-albert-small-v2" for sentence pair 2

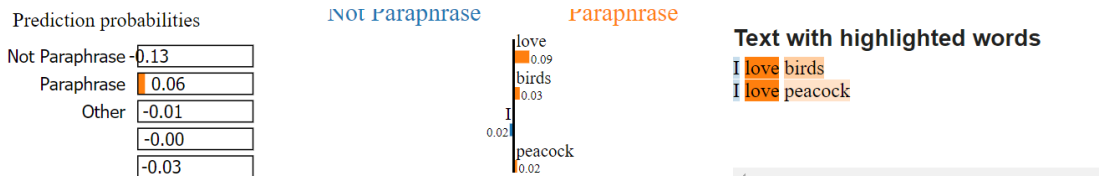


Figure 6.13: Result obtained from "paraphrase-albert-small-v2" for sentence pair 3

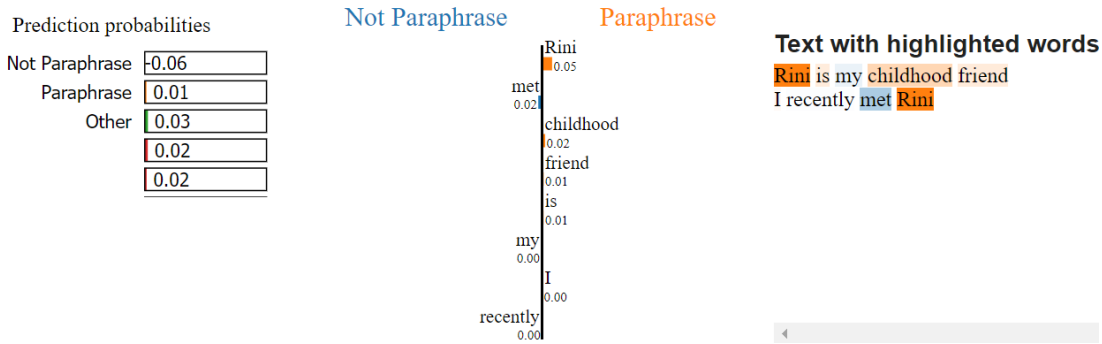


Figure 6.14: Result obtained from "paraphrase-albert-small-v2" for sentence pair 4

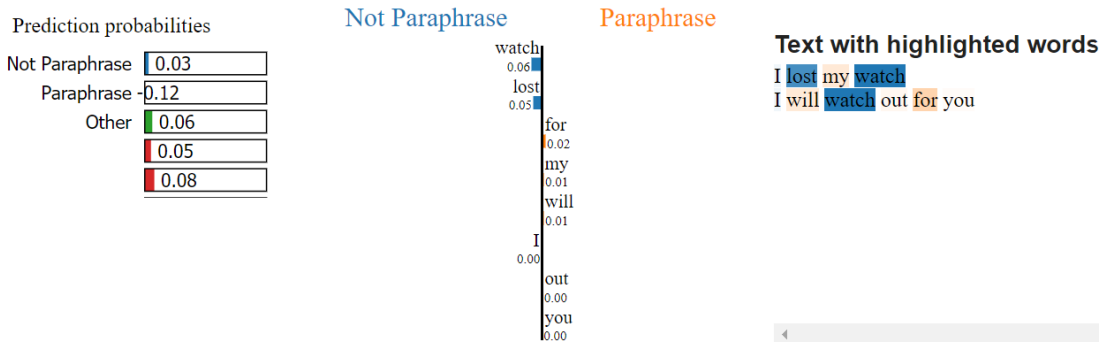


Figure 6.15: Result obtained from "paraphrase-albert-small-v2" for sentence pair 5

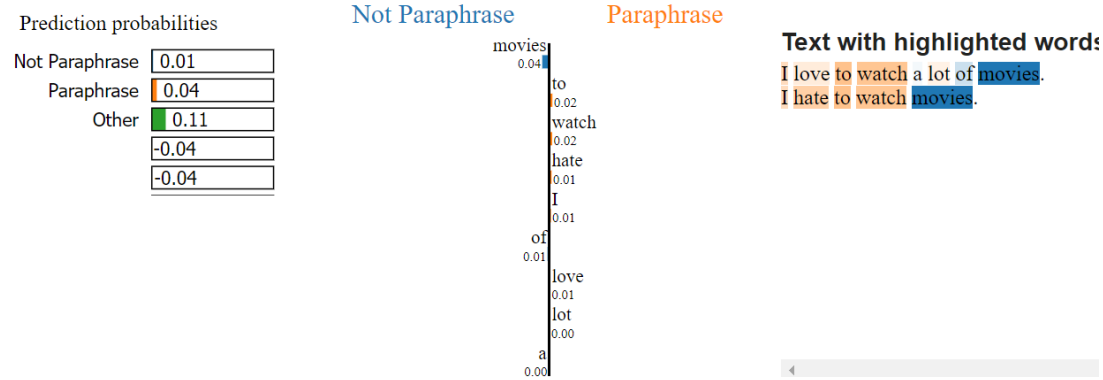


Figure 6.16: Result obtained from "all-mpnet-base-v2" for sentence pair 1

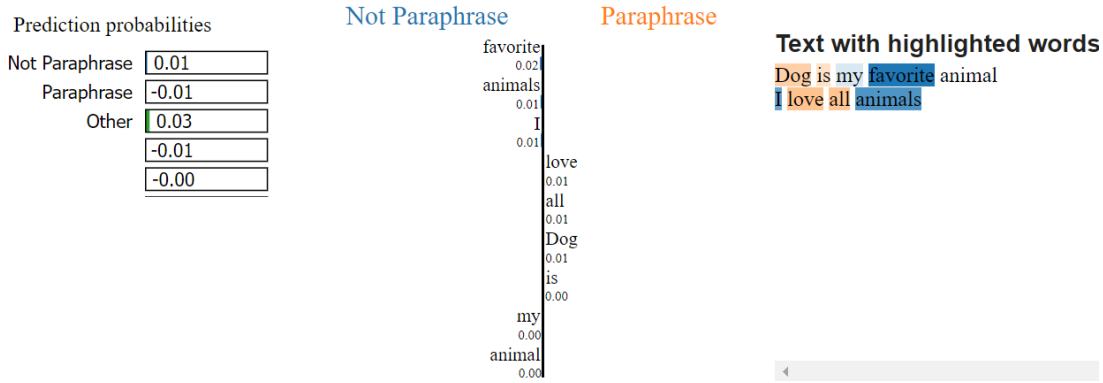


Figure 6.17: Result obtained from "all-mpnet-base-v2" for sentence pair 2

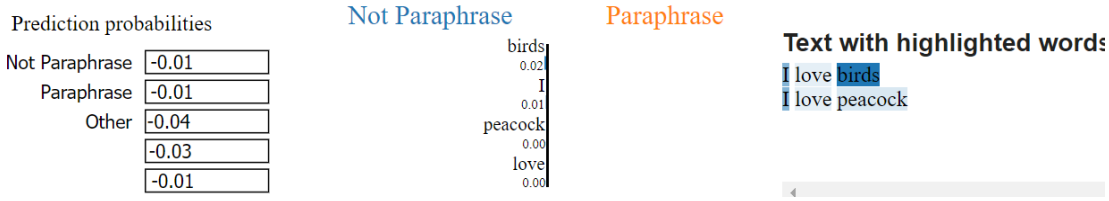


Figure 6.18: Result obtained from "all-mpnet-base-v2" for sentence pair 3

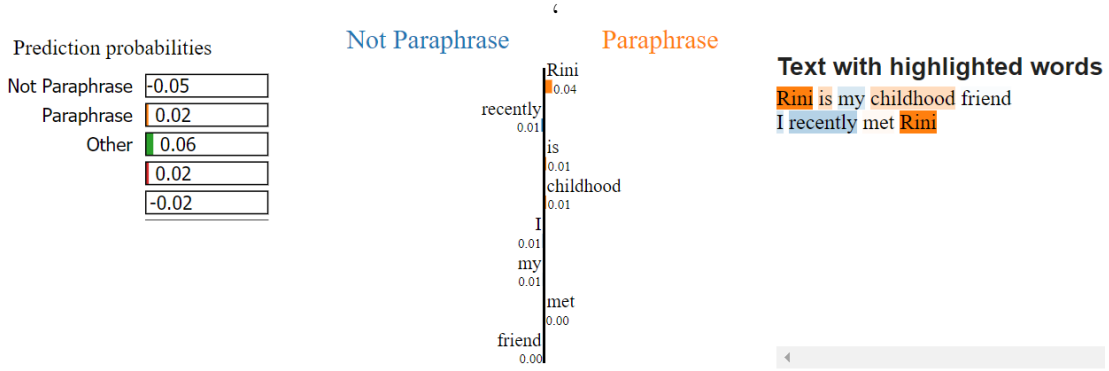


Figure 6.19: Result obtained from "all-mpnet-base-v2" for sentence pair 4

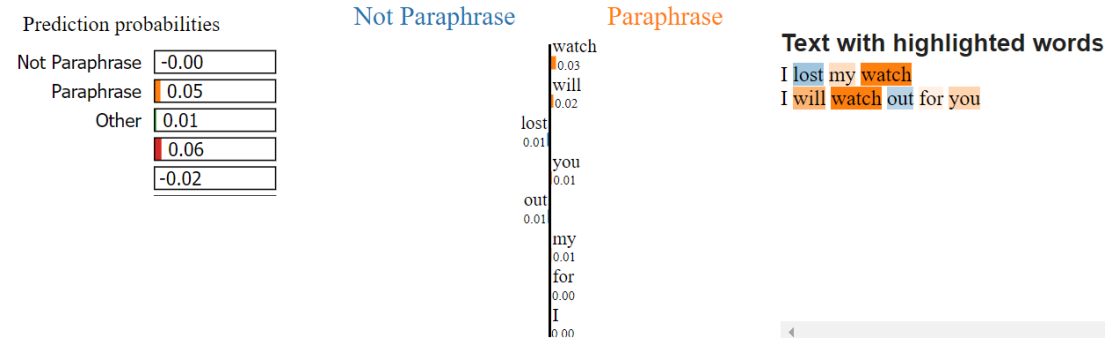


Figure 6.20: Result obtained from "all-mpnet-base-v2" for sentence pair 5

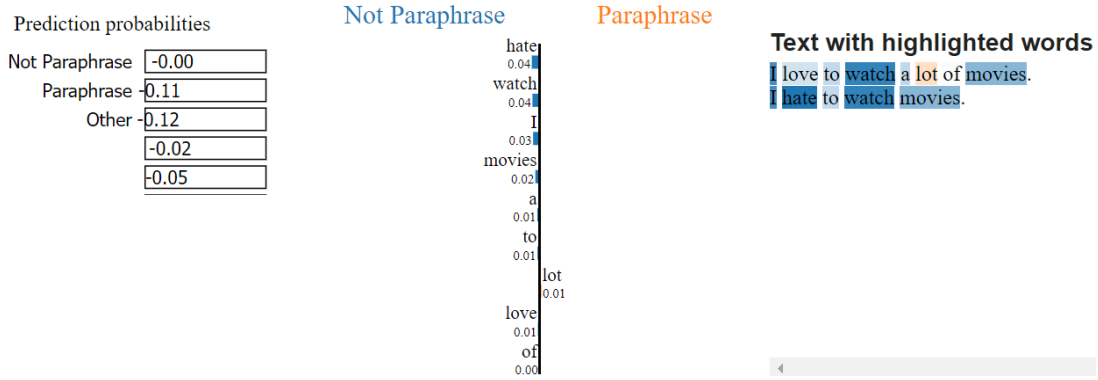


Figure 6.21: Result obtained from "all-MiniLM-L12-v2" for sentence pair 1

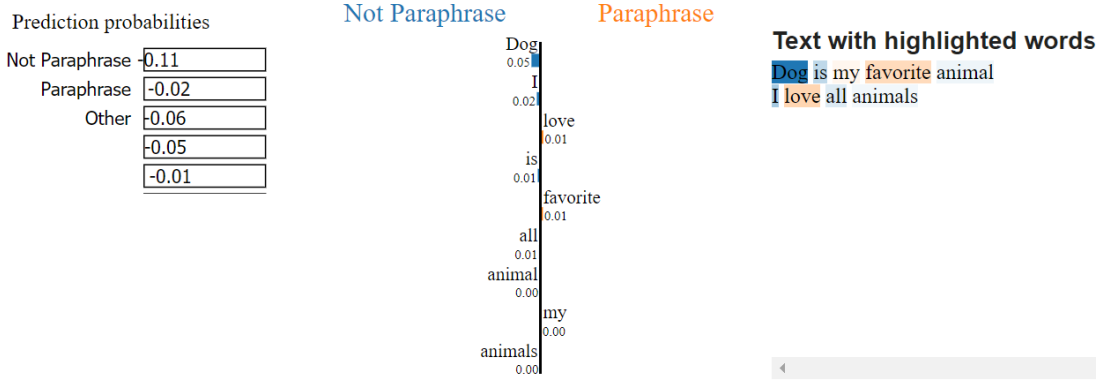


Figure 6.22: Result obtained from "all-MiniLM-L12-v2" for sentence pair 2

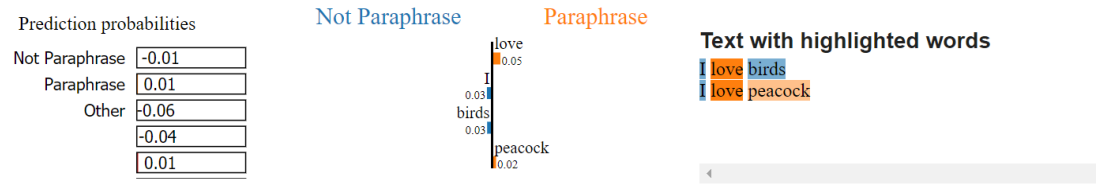


Figure 6.23: Result obtained from "all-MiniLM-L12-v2" for sentence pair 3

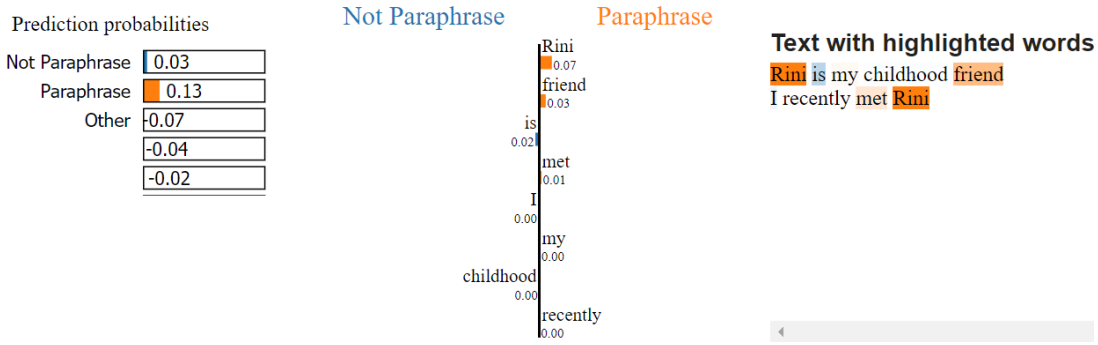


Figure 6.24: Result obtained from "all-MiniLM-L12-v2" for sentence pair 4

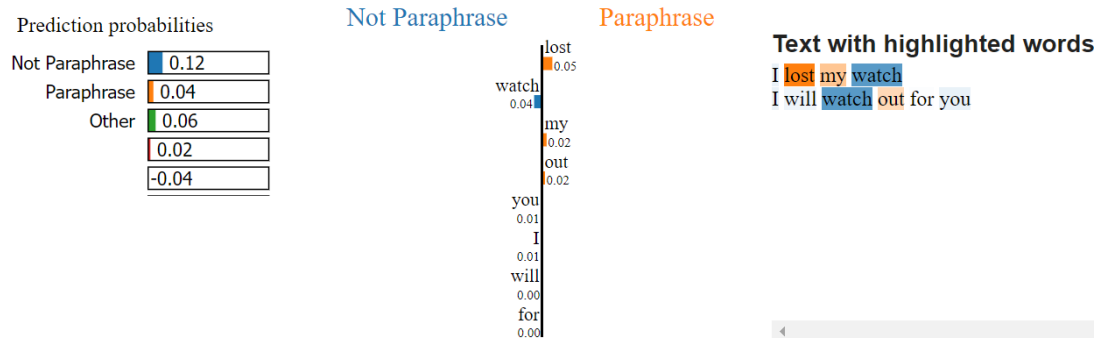


Figure 6.25: Result obtained from "all-MiniLM-L12-v2" for sentence pair 5

Model	Sentence Pair	Actual	Prediction
1	Pair 1	No	No
	Pair 2	Yes	Yes
	Pair 3	Yes	Yes
	Pair 4	No	No
	Pair 5	No	No
2	Pair 1	No	No
	Pair 2	Yes	Yes
	Pair 3	Yes	Yes
	Pair 4	No	No
	Pair 5	No	No
3	Pair 1	No	No
	Pair 2	Yes	Yes
	Pair 3	Yes	Yes
	Pair 4	No	Yes
	Pair 5	No	No
4	Pair 1	No	Yes
	Pair 2	Yes	No
	Pair 3	Yes	No
	Pair 4	No	Yes
	Pair 5	No	Yes
5	Pair 1	No	No
	Pair 2	Yes	Yes
	Pair 3	Yes	Yes
	Pair 4	No	Yes
	Pair 5	No	No

Table 6.1: Table showing the results of example sentences tested to check the model's efficiency. The first column is the model used refer Table 4.1, the second column is the sentence pair refer Table 5.1, the 3rd column is actual semantic similarity, the fourth column is predicted semantic similarity and the 5th column is percentage of correctly predicted sentence similarity by a model.

The images Figure 6.1 to Figure 6.20 shows the output of 25 different test cases carried out as described in the Table 6.1. These test cases show the quality of embeddings generated by the models and their visual explanations obtained through LIME.

In the LIME output, the words with orange highlights indicate similarity

while the words with blue highlight indicate semantic dissimilarity. The words with highest score of semantic similarity are given the darkest tone of orange while the words with the lowest semantic dissimilarity score is highlighted with the darkest color of blue. In the section **Produce Probabilities**, there are 3 sections,

- **Paraphrased:** It shows the probability of the sentences being paraphrased.
- **Not Paraphrased:** It shows the probability of the words not being paraphrased.
- **Other:** It includes words that are not directly responsible or important enough for a sentence to be semantically similar or not for e.g. I, a, the, etc.

It is observed the semantic similarity obtained by Model 1, 2, 3 and 5 is accurate while that of Model 4 is poor. Refer Table 4.1 to understand which models are being discussed.

6.1.1 Model 1

From Figure 6.1 to Figure 6.5, the model used is "all-MiniLM-L6-v2". For e.g in Figure 6.1, the word watch and movies should've slightly highlighted using orange and love and hate words should've been highlighted using darker tone of blue. Even if the final output and judgement given by this model is correct, but the highlights given for this sentence pair is different. In case of Figure 6.2 and Figure 6.3 the output generated is quiet better than before since here the correct words are highlighted with correct colors. For e.g favorite and love in Figure 6.2 and love and peacock in Figure 6.3 are highlighted with orange since the semantic meaning of these words align with the ones present in their paired sentences. In

Figure 6.4 correct words are highlighted in terms of words but the highlight color is wrong. In case of Figure 6.5 semantically dissimilar word i.e watch is correctly highlighted with the blue color even if it has the same syntax.

6.1.2 Model 2

From Figure 6.6 to Figure 6.10, the model used is "paraphrase-MiniLM-L6-v2". This model has given the best quality outputs of all the models used in terms of prediction probabilities values. In case of Figure 6.6, the output is slightly better than the ones generated by the model 1. Although, it is not completely correct since it is highlighting the main distinguishing element of the sentence which is love incorrectly. But again, hate is highlighted using blue tint which is a right output. Figure 6.7 and Figure 6.8 again have good output with words highlighted with correct colors except for the word favorite and birds in respective pairs. In case of Figure 6.8 correct words are highlighted but they should've been highlighted using Blue color as these words are responsible for semantic dissimilarity but they are highlighted using orange. This is the only model who has correctly colored words "recently" and "childhood" with blue tint in Figure 6.9. Similar to model 1, this time the explanations for Figure 6.10 is correct the word "watch" in the sentences is highlighted with blue. Although, lost could've been colored blue as well but instead it has orange tint.

6.1.3 Model 3

In model 3, few of the output have better quality than other models. Words in Figure 6.12, Figure 6.13 and Figure 6.15 are correctly classified. But the other 2 sentence pairs are not well differentiated.

6.1.4 Model 4

The model 4 has given the worst outputs among all others since this model is giving wrong predictions at first place. Even if the sentences are semantically similar, they are given as semantically dissimilar. It seems that similarity in this case is calculated in terms of syntax and not in terms of meaning at certain places.

6.1.5 Model 5

In model 5, Figure 6.21 has correctly highlighted all the 3 distinguishing words viz. love, watch and hate.. Figure 6.22 again have have correctly highlighted love and favorite with orange color. Although, dog and animal should also been highlighted using orange since both are same but it is highlighted using Blue. But still it can be considered since the 1st sentence says all animals and 2nd sentence says dog. This means it is not necessary that the writer like all animals instead there's a possibility of the writer liking just the dog. Figure 6.23 have correctly highlighted all three with orange and given correct output. Peacock could get a lighter tone of orange since it belongs to birds category but similar to earlier example it can be considered different. This model also has incorrectly classified Figure 6.24. In case of Figure 6.25 the word watch is correctly highlighted but lost should also get blue highlight since it is also an important distinguishing factor in the sentence.

6.2 Analysis on the implementation results

In terms of quality of the outputs generated, we can conclude that model 1 and model 2 outperformed rest of the models while model 4 became the worst suited model for our research.

Considering our first RQ from Chapter 1 we can say that using LIME, it is difficult to bring transparency and interoperability in the system for an STS task. As we can see in our explanations, we are only able to analyze which words were responsible for giving that particular result and we cannot see the explanations for internal operations and steps that resulted for a particular decision given by the system. This means, that explanations are still not transparent and it is not clear for a user to get an insight into the technical operations carried by the system. As a result, we can conclude that our approach of LIME would not be an ideal choice for obtaining transparency and interoperability of the system eventually leading to manual improvement of system's performance by try and error approach instead of XAI.

To answer our second RQ from Chapter 1, we can say that this model can identify ambiguity error. In the Figure 6.20, we can see that watch is highlighted using darker tone of orange that shows high similarity index. But this is an ambiguity error since in the first sentence watch means "a thing" and in the second sentence watch means "to look after". Looking at the figure we can easily identify where this error arises and accordingly we can solve it. Although, ambiguity error can be solved for this task but it is not a good idea to make a conclusion after testing on just one system and not trying it on different NLP use cases.

6.3 Analysing the survey results

27 people from different backgrounds were asked a series of question as mentioned in Chapter 5. Out of 27 people, 16 people belonged to IT and Computer Science background, 1 was from legal domain, 1 was from medical domain, and 3 were from banking, business, and other domains each. Among them 26 people have used AI system and only 1 person has never used it. There are high chances of the person being unaware of a system to be AI. Although, when it came to understand their knowledge on XAI systems, 16 people didn't know about it and 11 people knew that such system exists.

There were 3 more questions asked which were of high importance in terms of this thesis. In the Figure 6.26 we can see 55.6% of the people replied it is good to use Explainable AI but it is not necessary. When asked to justify there answers, a variety of answers were received. Important aspects of majority people choosing that option were,

- XAI would increase model size eventually leading to slower processing speed.
- For smaller projects it is not useful. Although, in case of big projects for e.g. in legal or medical domain such as Breast Cancer Prediction, XAI can be useful.
- Errors can be solved easily.

In Figure 6.27, 48.1% people chose 70-100% XAI system and 40-70% general AI systems. The second most chosen one was trusting both the system 50-50. When asked for justification, one most common answer was XAI is in the development phase and hence, it might not be as much efficient.

In the last Figure 6.28, majority of the population was not sure about this use case and replied as May be. One reason can be since it is in the research and

development sector, it is difficult to answer this question without any hard facts. Nonetheless, many don't have enough knowledge about XAI as well.

From this overall research, we can conclude that it is difficult for people to trust XAI over general AI but still a lot of them believe it will be a better approach since it will be more transparent. But this survey was only focused to the use case of XAI with general AI and it is not directly related to our research. Noting the responses, we can say users are ready to trust an XAI technique over general one because of its better clarity and transparent approach but still they are aware of the fact that it might not work as efficiently as required in the present times. Although further developments can bring a revolutionary development in this field. And this survey answers our 3rd RQ viz. do users trust XAI more than general AI.

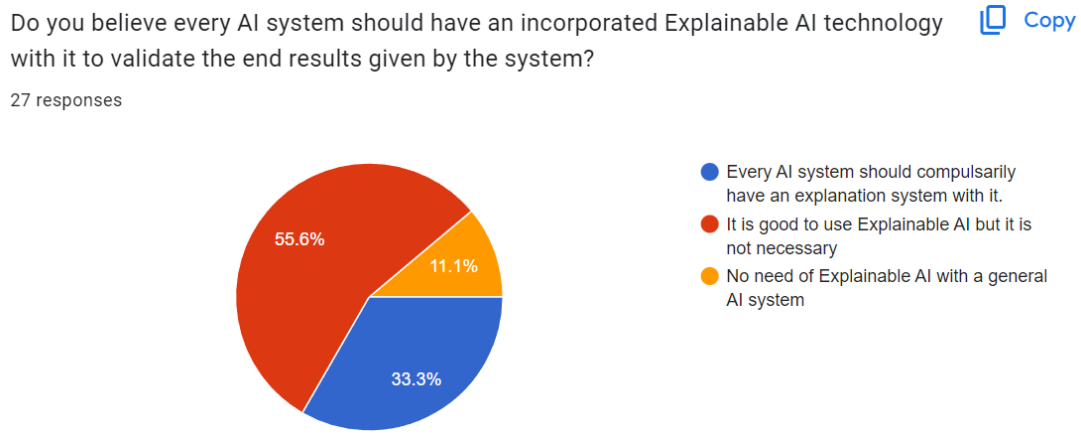


Figure 6.26: Survey result on incorporating AI with XAI

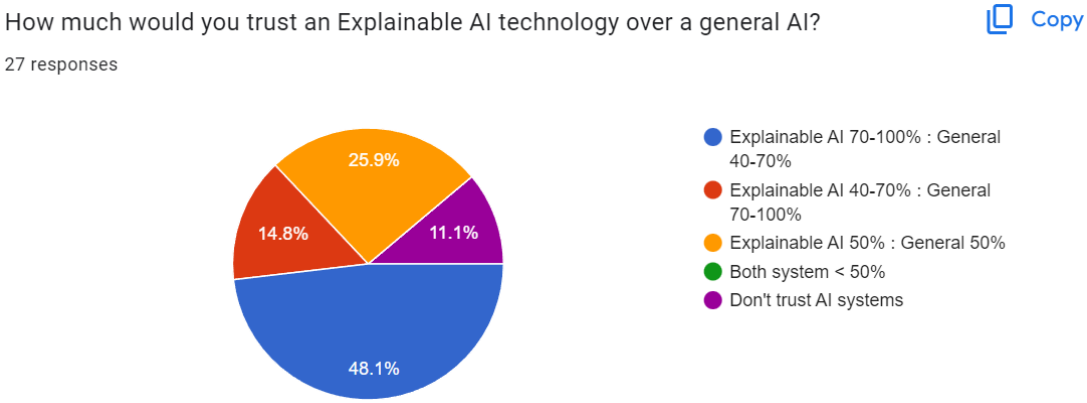


Figure 6.27: Survey result on trusting general AI and XAI

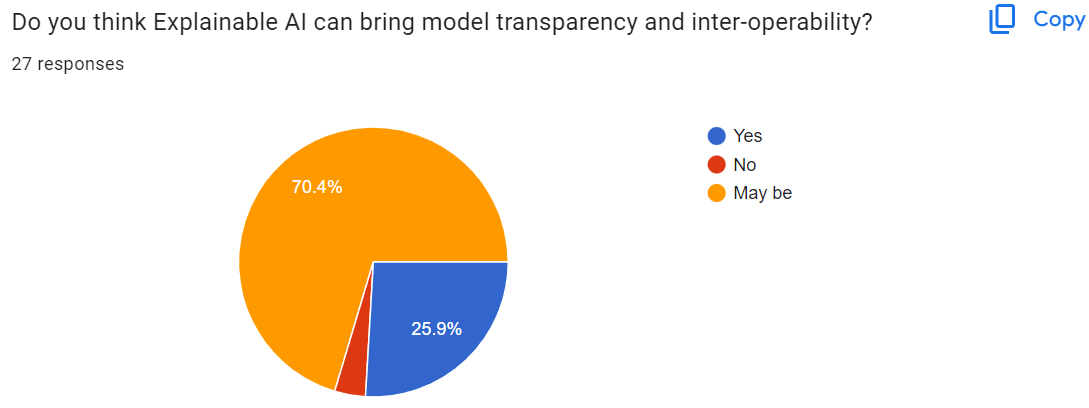


Figure 6.28: Survey result obtaining transparency in an AI system using XAI

Chapter 7

Conclusion

In this research a comparative study is carried out using 5 different Bert based sentence transformer models. These models are tested on the quality of embeddings generated which were used to obtain semantic text similarity. Furthermore, the results were accompanied with a visual explanation using LIME algorithm. The goal was to check how well Explainable AI could give explanations that were easily understandable by humans. It seemed that LIME could give accurate explanations in most cases, but it could only highlight the words with Blue or Orange shades indicating their similarity index. This explanation is useful at a basic level for e.g when a user wants to know which were the important words that resulted for a machine to give particular decision. However, ambiguity error can also be solved using this approach. But if a user wants to visualize complex tasks such as visualizing the internal operations carried by transformer than LIME might not be an ideal choice as it only give explanations at a local level. i.e. if user wants to check model behavior for all instances than LIME won't be helpful. It is only helpful in case of situation where we need explanation for a particular data point.

7.1 Limitations

One biggest drawback of this model is that the embeddings generated during the development were not of good quality due to which it affected our LIME explanations. Another limitation is that, the explanation is not sufficient at a global level as discussed earlier and the model is incapable of giving explanations for transformer working. Due to this, it is difficult to investigate the reason for obtaining poor quality of embeddings. This approach can only solve ambiguity error at the moment. LIME does not provide facility to edit the GUI of explanations provided. There are no facilities available to change the scale of the bars in the output or color of the highlights. This sometimes make it difficult for the user to understand the output because of smaller bars and overlapping values. Lastly, the visual explanations given are a bit complicated to understand especially for someone who possess less knowledge in AI field.

7.2 Future Work

Few potential developments in this research would be to obtain better quality embeddings from the transformers. As seen in the figures in Chapter 6, we can see every word is highlighted describing it's weight in a sentence. In this case, instead of highlighting any individual word, only highlighting those words which are responsible for a particular result will be a better approach as it will reduce output's complexity. For e.g. in Figure 6.5, instead of highlighting all the words with Blue and Orange shades, only highlighting the word watch would be a better choice because it is the most distinguishing factor in both the sentences irrespective of having similar sentences. Improving the visual effects of the explanations for e.g. figuring out a way to increase or decrease the scales to obtain better view on bars can be interesting.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. v, 4, 5, 6

[2] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. [Online]. Available: https://arxiv.org/abs/1908.10084 v, 6, 14, 15

[3] "Pretrained models - sbert," https://www.sbert.net/docs/pretrained_models.html, accessed: 30/08/2023. vii, 16

[4] S. Ali, T. Abuhmed, S. El-Sappagh, K. Muhammad, J. M. Alonso-Moral, R. Confalonieri, R. Guidotti, J. Del Ser, N. Díaz-Rodríguez, and F. Herrera, "Explainable artificial intelligence (xai): What we know and what is left to attain trustworthy artificial intelligence," *Information Fusion*, p. 101805, 2023. 1, 2

[5] A. Adak, B. Pradhan, N. Shukla, and A. Alamri, "Unboxing deep learning model of food delivery service reviews using explainable artificial intelligence (xai) technique," *Foods*, vol. 11, no. 14, p. 2019, 2022. 1, 12

[6] G. Majumder, P. Pakray, A. Gelbukh, and D. Pinto, "Semantic textual similarity methods, tools, and applications: A survey," *Computación y Sistemas*, vol. 20, no. 4, pp. 647–665, 2016. 2

[7] C. Wen, G. Jia, and J. Yang, "Dip: Dual incongruity perceiving network for sarcasm detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2540–2550. 2

[8] "Roberta model," https://huggingface.co/docs/transformers/model_doc/roberta, accessed: 31/08/2023. 6

[9] A. Conneau and D. Kiela, "Senteval: An evaluation toolkit for universal sentence representations," *arXiv preprint arXiv:1803.05449*, 2018. 6

[10] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://aclanthology.org/D14-1162 6, 7

[11] H. A. M. Hassan, G. Sansonetti, F. Gasparetti, A. Micarelli, and J. Beel, "Bert, elmo, use and infersent sentence encoders: The panacea for research-paper recommendation?" in *RecSys (Late-Breaking Results)*, 2019, pp. 6–10. 6

[12] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018. 6, 7

[13] X. Yang, X. He, H. Zhang, Y. Ma, J. Bian, and Y. Wu, "Measurement of semantic textual similarity in clinical texts: Comparison of transformer-

based models," *JMIR Med Inform*, vol. 8, no. 11, p. e19735, Nov 2020. [Online]. Available: http://medinform.jmir.org/2020/11/e19735/ 6

[14] "Xlnet model," https://huggingface.co/docs/transformers/model_doc/xlnet, accessed: 31/08/2023. 6

[15] K. Wang, N. Reimers, and I. Gurevych, "Tsdae: Using transformer-based sequential denoising auto-encoder for unsupervised sentence embedding learning," 2021. 7

[16] V. K. Ayyadevara and V. K. Ayyadevara, "Word2vec," *Pro Machine Learning Algorithms: A Hands-On Approach to Implementing Algorithms in Python and R*, pp. 167–178, 2018. 7

[17] T. De and D. Mukherjee, "Explainable nlp: A novel methodology to generate human-interpretable explanation for semantic text similarity," in *Advances in Signal Processing and Intelligent Recognition Systems*, S. M. Thampi, S. Krishnan, R. M. Hegde, D. Ciuonzo, T. Hanne, and J. Kannan R., Eds. Singapore: Springer Singapore, 2021, pp. 28–48. 7, 11

[18] W. Lu, J. Li, J. Wang, and L. Qin, "A cnn-bilstm-am method for stock price prediction," *Neural Computing and Applications*, vol. 33, pp. 4741–4753, 2021. 7, 8

[19] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II 14*. Springer, 2016, pp. 850–865. 7

[20] J. Bhattacharjee, *fastText Quick Start Guide: Get started with Facebook's*

library for text representation and classification. Packt Publishing Ltd, 2018. 8

[21] C. N. Tulu, "Experimental comparison of pre-trained word embedding vectors of word2vec, glove, fasttext for word level semantic text similarity measurement in turkish," *Advances in Science and Technology Research Journal*, vol. 16, no. 4, pp. 147–156, 2022. [Online]. Available: https://doi.org/10.12913/22998624/152453 8

[22] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015. 8

[23] Q. Chen, A. Rankine, Y. Peng, E. Aghaarabi, and Z. Lu, "Benchmarking effectiveness and efficiency of deep learning models for semantic textual similarity in the clinical domain: Validation study," *JMIR Med Inform*, vol. 9, no. 12, p. e27386, Dec 2021. [Online]. Available: https://medinform.jmir.org/2021/12/e27386 8

[24] Q. Chen, Y. Peng, and Z. Lu, "Biosentvec: creating sentence embeddings for biomedical texts," in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 2019, pp. 1–5. 8

[25] ——, "Biosentvec: creating sentence embeddings for biomedical texts," in *2019 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 2019, pp. 1–5. 8

[26] L. ZHAO, J. WANG, C. WANG, and M. GUO, "A cross-domain ontology semantic representation based on ncbi-bluebert embedding," *Chinese Journal of Electronics*, vol. 31, no. 5, pp. 860–869, 2022. 8

[27] B. Yan and M. Pei, "Clinical-bert: Vision-language pre-training for radiograph diagnosis and reports generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 3, 2022, pp. 2982–2990. 8

[28] A. Aziz, M. A. Hossain, and A. N. Chy, "Csecu-dsg at semeval-2021 task 1: Fusion of transformer models for lexical complexity prediction," in *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, 2021, pp. 627–631. 9

[29] "Shap," https://shap.readthedocs.io/en/latest/, accessed: 31/08/2023. 9, 10

[30] "Lime model," https://lime-ml.readthedocs.io/en/latest/index.html, accessed: 31/08/2023. 9

[31] J. Gerlings, A. Shollo, and I. Constantiou, "Reviewing the need for explainable artificial intelligence (xai)," *arXiv preprint arXiv:2012.01007*, 2020. 9

[32] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins *et al.*, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information fusion*, vol. 58, pp. 82–115, 2020. 10

[33] C. M. Sirisha Velampalli and A. Saxena, "Performance evaluation of sentiment analysis on text and emoji data using end-to-end, transfer learning, distributed and explainable ai models," *Advances in Information Technology*, vol. 13, no. 2, pp. 167–172, April 2022. 10

[34] J. Landthaler, I. Glaser, and F. Matthes, "Towards explainable semantic text matching," in *Legal Knowledge and Information Systems*. IOS Press, 2018, pp. 200–204. 11

[35] A. Warstadt, A. Singh, and S. R. Bowman, "Neural network acceptability judgments," *arXiv preprint arXiv:1805.12471*, 2018. 12

[36] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," 2019, in the Proceedings of ICLR. 12

[37] W. B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases," in *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. 12

[38] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should I trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 1135–1144. 17

[39] ——, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144. 17

[40] "Lime implementation," https://lime-ml.readthedocs.io/en/latest/lime.html#module-lime.lime_text, accessed: 30/08/2023. 18