Success of AI Writer

Ahmet Emre Usta[1] **Huseyin Yigit Ulker**[1]

Abstract

Plagiarism detection is the process of identifying and tracking the use of content from other sources without proper attribution. In this study, we propose a plagiarism detection method using a combination of the state-of-the-art language model RoBERTa, and a bidirectional long short-term memory (Bi-LSTM) neural network. RoBERTa is a transformer-based language model that has achieved strong performance on a variety of natural language processing tasks. The Bi-LSTM is a type of recurrent neural network that is able to process input sequences in both forward and backward directions, allowing it to capture long-range dependencies in the data. We trained our plagiarism detection model using the SNLI dataset and evaluated its performance using standard metrics. Our results show that the combination of RoBERTa and Bi-LSTM is able to achieve %90 accuracies in detecting plagiarism in the SNLI dataset. The proposed method is able to handle a variety of text types and languages, making it a promising solution for detecting plagiarism in academic and other settings. But our main goal is detecting plagiarism which is done by artificial intelligence(AI).

1. Introduction

Artificial intelligence (AI) writers are gaining popularity due to their efficiency and ability to provide powerful analytics that help predict and optimize content. One popular AI writing tool is Generative Pre-trained Transformer 3 (GPT-3), which can create new sentences and paragraphs based on a few input statements and perform tasks such as rephrasing content, spell and grammar checking, researching topics, generating ideas, completing forms, and formatting documents. While AI writers can be useful for students, the possibility of plagiarism is a concern. Plagiarism detection programs such as Turnitin and Quetext aim to prevent this, but it is unclear how effective they are against AI-generated text. In an effort to address this issue, researchers plan to develop AI models for plagiarism detection using existing datasets and manually paraphrase articles with AI writers to see how existing models handle duplicate data. It is also suggested that future work could involve retraining and testing existing models using transfer learning with a larger dataset of AI-generated text.In recent years, there has been a shift towards the use of natural language processing (NLP) techniques for plagiarism detection, as they offer the potential for more accurate and efficient detection. One such NLP approach is the use of a combination of the RoBERTa language model and a BiLSTM network. RoBERTa, developed by Facebook AI, is a transformer-based language model that has achieved state-of-the-art results on a variety of NLP tasks. BiLSTM networks, on the other hand, are a type of recurrent neural network (RNN) that can process sequential data in both the forward and backward directions, allowing for the capture of contextual information from both past and future words. In this article, we will explore the use of RoBERTa and BiLSTM for plagiarism detection, discussing their individual capabilities and how they can be combined to achieve improved performance. We will also delve into the evaluation and limitations of this approach, and discuss potential future developments in the field.

2. Related Work

The need for plagiarism detection programs has increased with technological developments and easier access to data, so many searches have been carried out in this area. Some of these methods can be found in [1] and [7].

In [1] Zakiy Firdaus Alfikri, Ayu Purwarianti created a plagiarism detection model using Support Vector Machines (SVMs) and Naive Bayes methods.In the method, word similarity, fingerprint similarity, hidden semantic analysis (LSA) similarity and word pair learning features were used. Plagiarism was detected using SVM and Naive Bayes algorithms. As a result, SVM and Naive Bayes achieved a success rate of 92% and 54%, respectively.

In [2] suggested by Chien-Ying, C., Jen-Yuan, Y. , Hao-Ren, K. In this research, the use of synonyms, which is the method that people use most when plagiarism, is discussed in determining the type of plagiarism. For this in the research, they offer three solutions for synonym recognition that make use of WordNet synonyms. WordNet groups nouns, verbs, adjectives, and adverbs into cognitive synonym clusters called synsets11 [5]. These clusters are then

measured for similarity using the Jaccard's coefficient.

In [3], suggested by Mostafa Hambi and Faouzia Benabbou. In this research, 3 deep learning models were used for plagiarism detection: Doc2vec, Siamese Long Short-term Memory (SLSTM) and Convolutional Neural Network (CNN). This study also includes two text comparisons, online comparison and using an intern corpus for a comparison. In the two text comparison, the two documents will be preprocessed and converted to a list of vectors with doc2vec model [7]. The system will detect later if the input documents are similar or not using SLSTM .Using the CNN model, the probability of each type of plagiarism (copy-paste,paraphrasing,false references ,translation,ideas) is calculated.The model in this study can detect not only whether there is plagiarism, but also the probability of plagiarism types.

In [4] Ahmed Hamza Osman, Naomie Salim, Mohammed Salem Binwahlan,Rihab Alteeb and Albaraa Abuobieda introduced the plagiarism detection system using the SRL method. This research, it converts the suspect and original document into arguments based on the position of each term in the sentence using SRL [5] . Next, the proposed method compares the arguments of the doubtful sentences with the similar arguments of the original sentences. According to the similarity result, plagiarism is analyzed.

Recurrent and recursive neural networks were used in the research in article [6]. First, words were vectorized using GloVe. Plagiarism detection was made using recurrent and recursive neural networks in vectorized sentences. As a result, he obtained F1 scores of 81% and 33% in train and test data, respectively.

3. The Approach

3.1. Dataset

SNLI (the Stanford Natural Language Inference dataset) is a large dataset of natural language sentence pairs that are annotated with labels indicating whether one sentence entails, contradicts, or is neutral with respect to the other. It was created by researchers at Stanford University and was released in 2015. The dataset consists of 570k English sentence pairs drawn from a variety of sources, including books, websites, and news articles. Each sentence pair is annotated with one of three labels: "entailment," "contradiction," or "neutral." The "-" class indicates that there is no consensus decision among the annotators, consequently, we remove them during the training and evaluation following the literature.

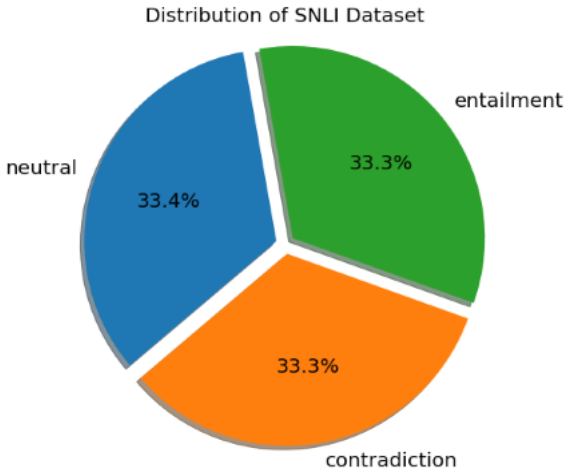

Figure 1

3.2. Model

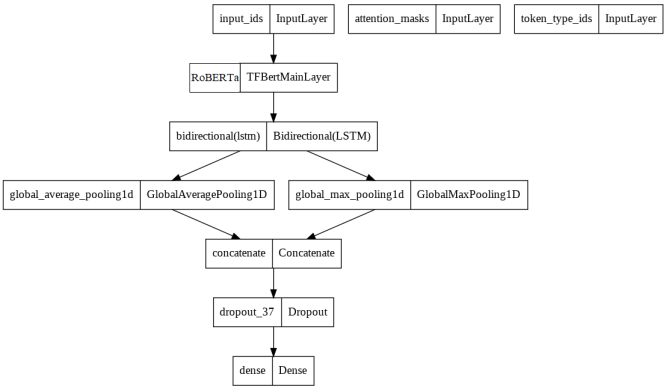

Figure 2 : Model Visualization[8]

3.2.1. Transfer Learning

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. It involves using the learned features from a previous task as the starting point for learning a new task, rather than training a model from scratch. This can significantly reduce the amount of data and computation required for the new task, and can also improve the performance of the model on the new task if the features learned for the first task are also useful for the second task. There are several ways to use transfer learning in practice, including:
Fine-tuning a pre-trained model: This involves using a pre-trained model as the starting point and then training it on a new dataset using the same architecture. This can be done by unfreezing a few of the top layers of the pre-trained

model and training those layers while keeping the rest of the layers frozen.
Using the pre-trained model as a feature extractor: In this approach, the pre-trained model is used to extract features from the new dataset, which are then fed into a new model that is trained to perform the target task.
Using a pre-trained model as a fixed feature extractor: This involves using the pre-trained model to extract features from the new dataset and then training a new model on those features. However, in this approach, the weights of the pre-trained model are not updated during training.
Transfer learning has been widely used in various fields, including natural language processing, computer vision, and speech recognition, to name a few. It has also been used to improve the performance of models on a wide range of tas ,, ks, such as image classification, object detection, and machine translation.

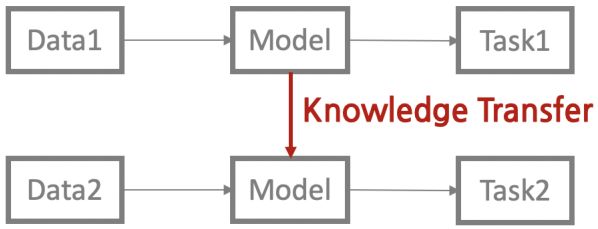

Figure 3: Transfer Learning Model[9]

3.2.2. RoBERTa

In natural language processing, a tokenizer is a software tool that divides a piece of text into smaller units called tokens. These tokens can be words, punctuation marks, or other elements of the text, depending on the specific tokenization method being used. The RoBERTa tokenizer is a specific tokenization method that is used to process text data for use with the RoBERTa language model.The RoBERTa tokenizer uses a pre-defined vocabulary of around 50,000 subwords. This vocabulary is derived from a large dataset of text data, and is designed to cover a wide range of common words and word sequences that occur in the data.The RoBERTa tokenizer first divides the input text into subwords using the vocabulary. If a word is not present in the vocabulary, it is divided into subwords based on the most frequently occurring character sequences in the data. The RoBERTa tokenizer then applies BPE to further divide the subwords into smaller units called ”tokens.” BPE works by iteratively replacing the most frequent character pairs in the data with a single, unused character, until a specified vocabulary size is reached. This process helps to reduce the overall size of the vocabulary, while still preserving the most important information in the data.The RoBERTa tokenizer includes several special tokens that are used to represent specific pieces of information, such as the beginning and end of a sequence, and padding. The RoBERTa tokenizer also includes several preprocessing steps that are applied to the input text before tokenization. These steps can include lowercasing, accent removal, and handling of special characters.

3.2.3. Bidirectional Long Short-Term Memory (Bi-LSTM)

Bidirectional Long Short-Term Memory (Bi-LSTM) is a type of recurrent neural network (RNN) that processes input sequences by reading them in both forward and backward directions. It is a variant of the LSTM (Long Short-Term Memory) network, which is designed to remember information for long periods of time and is particularly effective at processing sequential data.

The Bi-LSTM network consists of two LSTM networks that are connected in opposite directions, one processing the input sequence forwards and the other processing it backwards. The output of the two LSTM networks is then concatenated and passed through one or more fully connected (dense) layers to produce the final output.

One of the key advantages of the Bi-LSTM is that it can capture contextual information from both past and future input elements, allowing it to better understand the relationships between input elements and make more accurate predictions. This is particularly useful in tasks such as language modeling, machine translation, and speech recognition, where the order of the input elements is important.

Bi-LSTM networks are trained using backpropagation through time (BPTT), which involves unrolling the network over time and updating the weights of the network at each time step using the gradients calculated by the backpropagation algorithm. They can also be trained using more efficient variants of BPTT such as truncated BPTT and teacher forcing.

4. Experimental Results

At this step, we tried two different experiment using transfer learning. In the first one, for a base model, we prefer to use bert-base model from Hugging Face library. For the second with different methods and a modified dataset, we use roberta-base model from Hugging Face again. Both of the experiments run on the Google Colab Pro subscription. For the settings, we select the high memory and premium GPU options.

4.1. Bidirectional Encoder Representations from Transformers (BERT)

We use all train set which contains approximately 550000 line provided from SNLI. At first, we run feature extraction method on 427,267 parameters with 2 epochs. This part

take 36 minutes. Then, for the fine-tuning part, we run the model on 109,909,507 parameters again with 2 epochs. This part takes the 66 minutes to done. On validation set, we get the 0.29 loss score and 0.89 accuracy score. Then for test set, we get the 0.27 loss score and 0.90 accuracy score.

4.2. RoBERTa

For this part, since the RoBERTa is a very large model compared to the Bert, train times get higher. To solve this problem, we prefer to use a part of our train dataset. To select the meaningful part of this dataset, we check the length of the sentences and then the select the longer half of the all train dataset. In addition to shortening the train time, this gave us the advantage of using a higher epoch number. At first, we run feature extraction method on 427,267 parameters with 4 epochs. This part take 32 minutes. Then, for the fine-tuning part, we run the model on 125,072,899 parameters again with 4 epochs. This part takes the 64 minutes to done. On validation set, we get the 0.22 loss score and 0.91 accuracy score. Then for test set, we get the 0.27 loss score and 0.90 accuracy score.

References

[1] ALFIKRI, Z. F., AND PURWARIANTI, A. "detailed analysis of extrinsic plagiarism detection system using machine learning approach (naive bayes and svm), 2014.

[2] CHIEN-YING, JEN-YUAN, C., AND Y.AND HAO-REN, K. Plagiarism detection using rouge and wordnet. journal of computing, 2(3), 34-44, 2010.

[3] HAMBI, E. M., AND BENABBOU, F. A new online plagiarism detection system based on deep learning in international journal of advanced computer science and applications, 2020.

[4] OSMAN, A. H., SALIM, N., BINWAHLAN, M. S., ALTEEB, R., AND ABUOBIEDA, A. An improved plagiarism detection scheme based on semantic role labeling, 2012.

[5] PAUL, M., AND JAMAL, S. An improved srl based plagiarism detection technique using sentence ranking, 2014.

[6] SANBORN, A., AND SKRYZALIN, J. Deep learning for semantic similarity., 2017.

[7] SANDILYA, N., SHARMA, R., AND MELEET, M. Plagiarism detection using natural language processing and support vector machine, 2022.

[8] RoBERTa and BiLSTM Model, "https://huggingface.co/keras-io/bert-semantic-similarity/blob/main/model.png"

[9] Transfer Learning , "https://ratsgo.github.io/nlpbook/docs/introduction/transfer/"