Universiteit Leiden
The Netherlands

Opleiding Informatica

Predicting Fake News
via Stance Detection

Frank Haasnoot

Supervisors:
Dr. S. Verberne and Dr. O. Gadyatskaya

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

31/07/2022

Abstract

Due to the uprising of social media, Fake News is becoming a bigger problem. Anyone can share anything on social media and the quantity of these posts on a daily basis is enormous so some sort of automatization in detecting Fake News is required. In order to stimulate developing these kind of techniques, the Fake News Challenge was created in 2017 which had numerous universities across the world compete in it. The goal of the challenge was to predict the stance from an article towards a headline. The aim of this thesis is to build a better machine learning model than the submissions of 2017 with newer NLP techniques. For this research, the submission of the UCL machine reading team was replicated and newer models were used to improve the score of the UCL machine reading team. This resulted in using three different BERT models. From the experiments it was clear that the basic version of BERT, finetuned on the FNC data set, was the best model. It improved the result of the UCL team with a relative FNC-1 score improvement of 4.13, concluding that is possible to improve the submission of 2017 with newer NLP-techniques.

Contents

1 Introduction

Anno 2022 social media has become one of the biggest news sources across the world [13]. With social mediums like Facebook, Instagram and even Tiktok, there are a lot of possibilities to share news besides making content for pure amusement. At first this does not look like a big problem because you will think it is good for people to read the news and knowing that the majority of users of social media are young people and that young people usually read less news, it would be a good thing. Unfortunately, there are important differences between the mainstream media and social media.

First of all every person in the world can post their news or opinions on social media anonymously which is making it very hard to find the person behind the computer. Because of this anonymity, people can post everything on social media without having to think it through. This can cause for example bullying at schools which is even addressed as a national health issue in USA in 2011 [19].As the risk on getting caught on the internet is small, people tend to spread more and possibly Fake News on the internet because there are little to zero bad outcomes to it [11]. If a mainstream medium is spreading Fake News, they can be easily tracked by looking for the author of the article and taking actions accordingly. There is also a risk for the medium that people will define the medium as a bringer of Fake News which is harmful for the company because it will result in fewer readers [14].

Besides, mainstream media and their journalists adhere to some ethics and standards whereas people on social media adhere to nothing. These ethics causes journalist "to produce content which is honestly conveyed, accurate and fair" [7]. One of these standards is the right to a fair hearing whereas social media does not adhere to these rights. For example in the Netherlands there is an uprising of gossip channels which discusses the private lives of celebrities. They post all their gossip on their channels without trying to reach out to celebrity if this gossip is even true, concluding in damages to this specific person and lawsuits following [3].

We can conclude that social media is different in their news quality compared to the mainstream media but is this really a problem, in other words is Fake News that big of a problem? In 2021 the president elections of the United States were being held between Donald Trump on the republican side and Joe Biden on the democratic side. Although Biden won by a fine margin, during the election it was expected that the democrats would win causing Trump to spread Fake News on Twitter by saying that the election was rigged and votes were added in favor of the democrats or votes were being destroyed of republicans [4]. Because Trump was spreading Fake News about the election, it resulted in many angry republicans storming into the US capitol building with as result deaths and big damages to the building [15].

Fake News causes people to be misinformed, polarization and much more [5]. But for social media companies it is impossible to look into every post for if they contain Fake News. They have divisions of workers who try to mitigate Fake News by users who flagged certain articles but that is not enough because you can't expect your personnel to check every article since there are simply too much. For example on Twitter there are in May 2022 approximately 867 million tweets sent a day.

Luckily nowadays Natural Language Processing techniques are developed which makes a computer able to process language. There are many options to do this and so in 2017 the Fake News Challenge started which had a goal to "explore how artificial intelligence technologies, particularly machine learning and natural language processing, might be leveraged to combat the Fake News problem" [2]. For the reason that detecting Fake News can be a cumbersome task for humans and computers, the process of detecting Fake News is usually done in a few steps. One of the first helpful steps you can take is called Stance Detection. Stance detection is a method which estimates the relative perspective between a certain text (body) and a headline (stance). In the case of the Fake News challenge the body news article can Agree, Disagree, Discuss the headline or it can be Unrelated. This is a supervised learning task and can be done via many different methods which will be discussed later in this paper. With Stance Detection it is possible to process two pieces of text and classify the relation in one of the four labels. In this challenge teams were encouraged to detect Fake News by classifying the relation of headlines and articles.

1.1 Aims and research question

Since this Fake News Challenge was released and the competitors submitted research, new NLP techniques have been introduced that have become the state-of-the-art on many tasks. Because of the improvement of technology, new models were introduced and the accuracy of predictions became a lot higher. For this research I want to find out if it is possible to improve the submissions of 2017 based on this improvement of technology. If I want to improve submissions, I first have to replicate one of the results in order to understand the Fake News Challenge and ideas behind it. The submission of the UCL machine reading team [8] was the 3th best submission of the Fake News Challenge which is why I came up with the following research question:

> *"Is it possible to improve the relative score from the submission of the UCL Machine Reading team for Fake News Challenge with newer technologies?"*

I will do this by first reproducing the method of the UCL team and trying to retrieve the same results as they did. Then I will use newer different methods to see of this will improve the score. To be able to achieve this I consulted literature about newer NLP techniques and more specific the ones who are related to Stance Detection. The UCL team wrote a short paper about their submission in which the method is explained and the results are given so I am able to compare it with my reproduction [16]. In 2020 Glant et al. [8] wrote a paper about stance detection in COVID-19 tweets which is very similar to my research, that I will explain later on. BERT is nowadays the free to use state-of-the-art model for many NLP related questions. Because of its state-of-the-art performance I will investigate if it is possible to use it for this research. To discover this I will use the paper written by Devlin et al. [6] to get a better understanding of this technique.

2 Background

2.1 Fake News Challenge

Fake News can be a big problem as shown above and that is why this challenge was created. It was developed by more than 100 volunteers and 71 teams of different academia and industry around the world. The main goal of this challenge was to address the problem of Fake News and by organizing the challenge, motivating people across the world to develop anti Fake News tools by using machine learning, natural language processing and artificial intelligence. With these tools human fact checkers could make their work a lot more efficient and easier and thus making it harder for people to spread Fake News [2]. The best result of the challenge was the one of the Team SOLAT in SWEN [18]. This score will also be taken into account for the comparison since it is winner of the challenge but not for replication because it was not clear how to reproduce the code.

2.2 Stance Detection

As described above, fact checking news can be a hard task, even for trained experts. One of the possibilities to make it a bit easier is by breaking it up in steps. One of these steps is Stance Detection because it could serve as a useful building block in AI-assisted fact-checking pipeline according to the creators of the Fake News Challenge [2]. An example of this could be that you create a model with a database full of scientific articles. This can then help people in detecting Fake News by checking of a certain headline is supported by articles written about this specific subject. So for example if you want to verify if the headline "COVID-19 is not harmful to the human body' is Fake News, you can feed this headline into the model and it will check how the articles in your database relate to this headline. The model will do this by making pairs of the headline and the articles in the database and with Stance Detection, the model gives the stances of the articles to the specific headline. In this case it will probably state that many articles disagree with this headline, giving the user more information about this statement and giving a high probability that the headline is Fake News.

2.3 Models for Stance Detection

2.3.1 UCL machine reading

The UCL-model is a multi-layer perceptron (MLP) with one hidden layer. A multi-layer perceptron is an artificial neural network (ANN) where all nodes of one layer are connected to the next layer. In this case the hidden layer consists of 100 nodes. A neural network also consists of an activation function to transform the summed weighted input of the input nodes to a specific output. In this case they use rectified linear activation function (ReLu) for this MLP on the hidden layer. At the end they use a SoftMax layer to give a probability to each of the 4 possible outcomes [16].

To use the model, the body and headline texts were converted to vectors. For this model they used bag-of-words representations namely: the term frequency (TF) and the term frequency-inverse document frequency (TF-IDF) [16]. A bag-of-words representation is a way to represent a text in a vector by for example counting the word occurrences. In order to calculate the vectors, you need a collection of words which you can then relate to your document. For a TF vector this means it

counts the occurrences of the terms, which are in the collection of words, in the document. The problem with TF vectors is, it does not state how important the terms are. TF-IDF does have this weighted importance. For calculating the TF-IDF, the document frequency needs to be calculated which is how many times a certain term appears in a collection of documents. To get the inverse document frequency, the total number of documents is divided by the document frequency which then gives a low value to a word if it occurs many times in the collection of documents. The reason behind this is that if a term occurs frequently in many documents, it is usually less important to the document. For example the word "and" usually occurs many times in documents which probably means it is a general word which then means it probably does not hold much information about the document. Combining the IDF with the TF gives you a TF-IDF vector. In the model they used the following:

- The TF vector of the headline text;
- The TF vector of the body text;
- The cosine similarity between the normalized TF-IDF vectors of the headline and the body;

If all the words are taken into account for calculating the vectors, the vectors would be really long. This could make the results worse and the computing time much longer so for calculating the TF vectors, they used the 5000 most frequent words in the training set and for TF-IDF vectors, the same was done except that the most frequent words were extracted from the training and test set. In the 5000 most frequent words, they excluded stop words [16].

The goal of training the model was to minimize the cross entropy between the true labels and the SoftMax probabilities. The training was done by back-propagation over mini batches in the whole training set and the hyperparameter optimization was done by the Adam optimizer.

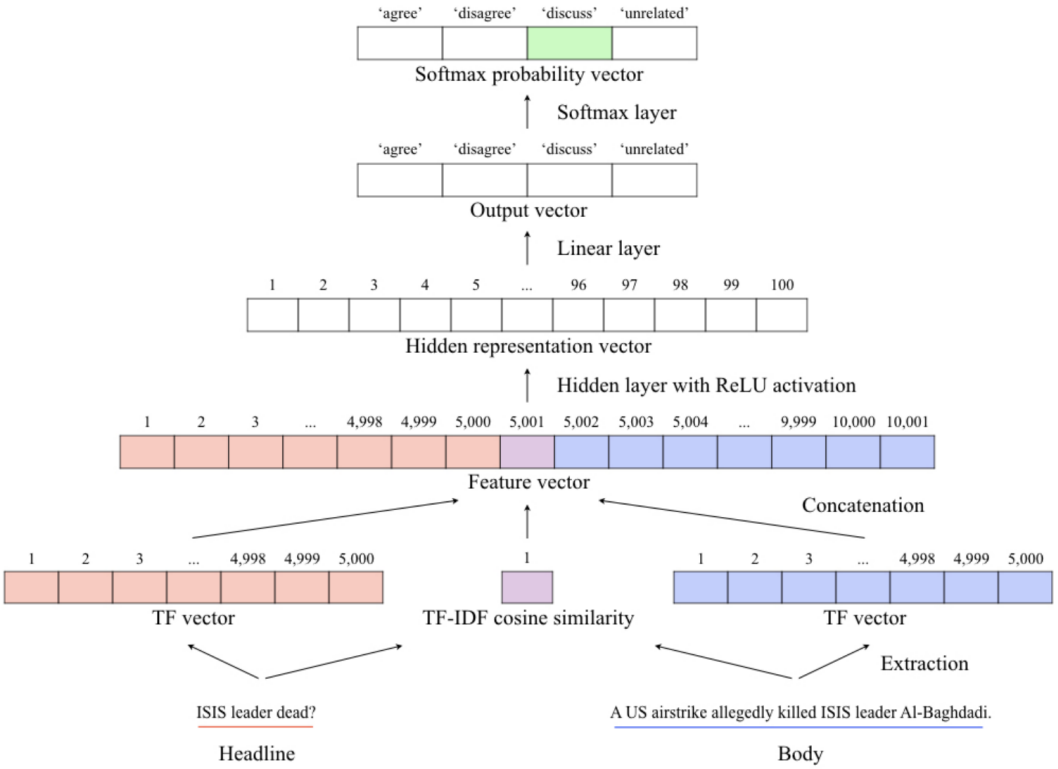
Table 1: Parameters for training

Parameter	Value
BOW vocabulary size	5000
MLP hidden layer size	100
1 - dropout on layer outputs	0.6
L2 regularisation strength	0.0001
Adam learning rate	0.01
Global norm clip ratio	5
Mini-batch size	500
Number of epochs	90

After they applied the model on the testset they got the following results:

Table 2: Confusion matrix of UCL machine reading team [16]

Figure 1: Schematic overview of UCL model [16]



Pred. / True	**Agree**	**Disagree**	**Discuss**	**Unrelated**	**FNC-1 score**
Agree	838	12	939	114	81.72
Disagree	179	46	356	116	
Discuss	523	46	3633	262	
Unrelated	53	3	330	17963	

The results led into a score of FNC-1 score of 81.72 (for explanation see 3.2) which resulted in the 3th best submission of the Fake News Challenge. In the paper Riedel et al. described the results as being good except for the average performance on the Agree and the poor performance on the Disagree label. An explanation for this could be that there are relatively few instances of the Agree and Disagree labels in data set compared to the Unrelated label, which the model defines almost perfectly. It is acceptable that model performs so well to the Unrelated label but this is not the most important label for the challenge. A better classification accuracy for the Agree and Disagree label is needed because these labels together with the Discuss label contribute the most to FNC-1 score.
[16]

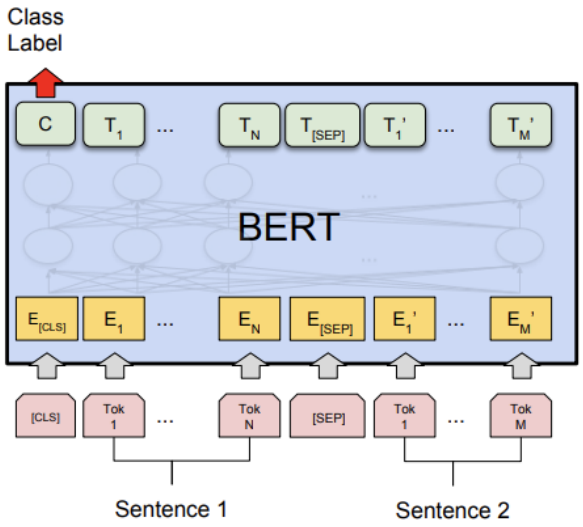
2.3.2 BERT

Before I will elaborate on the BERT method I used for this research, I will first give a bit of background about this (relatively) new state of the art NLP-technique. BERT stands for Bidirectional Encoder Representations from Transformers and is a deep learning model [6]. BERT was developed by Google in 2019 and was based on the research on Transformers which was first introduced in 2017 (also by Google). Transformers are like recurrent neural networks (RNN) but where RNN can have it difficult when processing different kind of input lengths and then especially bigger input lengths, Transformers don't have this problem. This is because Transformers use a technique called attention which computes relations between all pairs of input words. The benefit of this technique is that when the input length of your model is very long it focuses on the relative important parts and diminishes the not so important parts, causing it to be very suitable for longer input lengths. However there are still limitations to the input lengths because when it becomes too long, the computational complexity becomes too high and that is why BERT for example only accepts 512 tokens at maximum.

Following up the research on Transformers, Goolge invented BERT [6]. BERT uses two classic NLP techniques namely MLM (masked language modelling) and NSP (next sentence prediction) to pretrain itself. MLM is used because BERT trains itself in both directions and MLM prevent the words from indirectly seeing themselves. MLM masks 15% of the words in all sentences. Then the model tries to predict those random masked words to get a better understanding of the language [6].

NSP is used to get a better understanding of relationship between sentences. In traditional bag of words models the order of sentences wasn't considered in the model and with that many information got lost which is needed for tasks like Question Answering and Natural language inference. NSP is a binary task which tries to predict whether a sentence is next to another sentence and can be easily done on a monolingual text set.

Figure 2: Example of sentence classification task [6]



One of the biggest advantages of BERT, besides its state-of-the-art performance, is that it is pre-trained. This is an example of transfer learning which basically means that you don't have to start training from scratch every time you build a model. Since BERT doesn't need any labeled data, it can easily be pre-trained on lots of unlabeled data. For example, BERT is trained on the BooksCorpus (800 million words) and the English Wikipedia (2,500 million words). If you want to use BERT for your own task you only have to finetune it on your specific data, which also means you don't have to have lots of data since BERT already has an understanding of the language you need for your task. Another upside of the pre-training not needing any labeled data, is that you can easily pre-train BERT on other data to your preference. For example in the introduction I related to the US presidential election and there is already a model available which is pre-trained on tweets for stance detection about the election [9]. This is a rather specific pre-trained model but if you want to just analyze tweets, there is also a pre-trained model available specialized in social media analysis [12].

For this research I will use the following three versions of the BERT model which are all general-domain models for English, meaning that they are all pre-trained on the English language:

- The main version of the BERT model, pre-trained on Wikipedia and Bookscorpus;
- DistilBERT which is a variant of the BERT model except it works faster and is smaller than Bert and works faster [17];
- RoBERTa is a BERT model which is the same as BERT apart from the fact it masks different words in sentences in different epochs [10];

Because there is a relatively low amount of literature about stance detection on the relation between an article and a headline, I chose to apply a method analog which was used in COVID-19 tweets [8]. In this paper the researchers tried to get the stance of a tweet towards a certain target. For example the target was 'wearing a face mask' and then they tried to classify the tweets on if they were in favor of the target or against the target. The differences between that research and my research are that the COVID-19 research is a binary classification and the FNC-research is a multiclass classification. Next to that the tweet (in other research body text) and the target (in other research the headline) are shorter in the COVID-19 data because a tweet can be 280 characters at max and a published article could be much longer. But apart from these two differences the tasks are similar which is why I chose to use the BERT model and its adaptations based on this article [8]. The models are retrieved from Huggingface [1].

3 Data

3.1 Descriptive statistics

For this research the data set was used which was provided by the organizers of the Fake News Challenge [2]. The data consists of pairs of headlines and body texts with a provided label which states agree, disagree, discuss or unrelated. The data is provided in different csv files, splitting the stances from the body texts and respectively put the corresponding body/stance id to it. During the challenge there were also unlabeled bodies and stances test sets released for the submission, but because the challenge has been finished, a labeled version was released so I can calculate the score metrics of my research on the test set and can compare the results. The data has the following statistics:

Table 3: Descriptive statistics of the FNC-data [2]

	Total Rows	**Unrelated**	**Discuss**	**Agree**	**Disagree**
Number	49972	36545	8909	3678	840
Percentage	100%	73%	18%	7%	2%

If you look at the data, it is noticeable that the data set is highly imbalanced. 73% of the data consists of pairs which are Unrelated and only 2% of the pairs Disagree. This could possibly have an effect on the results of the models because there is not much training possible on the other labels, more specifically on the Agree and Disagree labels.

The data consist thus of many headlines and body texts and there are multiple pairs with the same headline but different body text to make it easier for the model to classify. A formal definition can be given of the data:

Input

A headline and a body text – either from the same news article or from two different articles.

Output

There is a classification given of the relation from the headline (stance) to the body (article) which can be:

- Agree: the body text agrees with paired headline;
- Disagree: the body text disagrees with the paired headline;
- Discuss: the body text discusses the paired headline;
- Unrelated: the body text is unrelated to the paired headline;

To get a better understanding of the data and how the labels can be interpreted, I give an example with snippets from the body text which give a hint to a specific label:

Headline: “By authorities provided Jansen vaccine doesn’t work effectively against newest COVID-19 variation called Omikron”

- Body text snippet which agrees: “Research shows that the Jansen vaccine is highly ineffective against the newest Covid 19 variation.”
- Body text snippet which disagrees: “The CDC states that the Jansen vaccine has the highest protection rates against the Omikron variant.”
- Body text snippet which discusses: “New developments around the Jansen vaccine shows that is possibly more effective than we thought it was”
- Body text snippet which is unrelated: “Breaking! Joe Biden wins the 2020 presidential election!”

3.2 Evaluation

To compare the results of this research I use the same evaluation as they used in the Fake News Challenge. This metric is called the FNC-1 score. The results are calculated by summing all the correctly classified labels where every correctly classified label adds maximal 1 point to the FNC-1 score. In the FNC they make use of two different levels to calculate the results:

- Level 1: Pairs of headlines and body text classified as Related or Unrelated have a 25% score weighting;
- Level 2: Pairs of headlines and body text classified as Agree, Disagree of Discuss have 75% score weighting;

When predicting the label, there are multiple possibilities. First the model can predict if the pair is Related by predicting the label Agree, Disagree or Discuss or it can predict the pair as Unrelated by predicting the label Unrelated. If this is done correct, it adds 25% of the maximum score (1) to the FNC-1 score, which is 0.25. Then if the pair is Related, meaning it has the label Agree, Disagree or Discuss, and the model predicts the correct label, it adds another 75% of the maximum score (1) to the FNC-1 score, which is 0.75. So if the model predicts Agree while the True label is actually Disagree, it gets 0.25 points for correctly classifying it as Related but not with the correct Related label. If the model predicts Agree and the True label is Agree, it gets 0.25 points for correctly classifying it as Related and 0.75 points for predicting the right Related label. This metric scheme also means you can get only 0.25 points for predicting a pair correctly as Unrelated. If the evaluation is done for every label, all the scores are summed which gives the FNC-1 score. The maximum number of points is 11651.25. In order to compare all the results of different teams, the FNC-1 score is divided by the maximum number of points resulting in the relative FNC-1 score. This gives the following formula:

$$\text{relative FNC-1 score} = \frac{\text{maximum number of points}}{\text{FNC-1 score}}$$

There are two reasons behind this parted evaluation. The first one is that is usually easier for a model to predict if texts are unrelated/related to each other, so it must be graded differently since it is much harder to decide if the texts agree, disagree or discuss. The second reason is that

classifying related/unrelated isn't that beneficial for detecting Fake News, if you know the body text is related you still don't know the stance towards the headline [2].

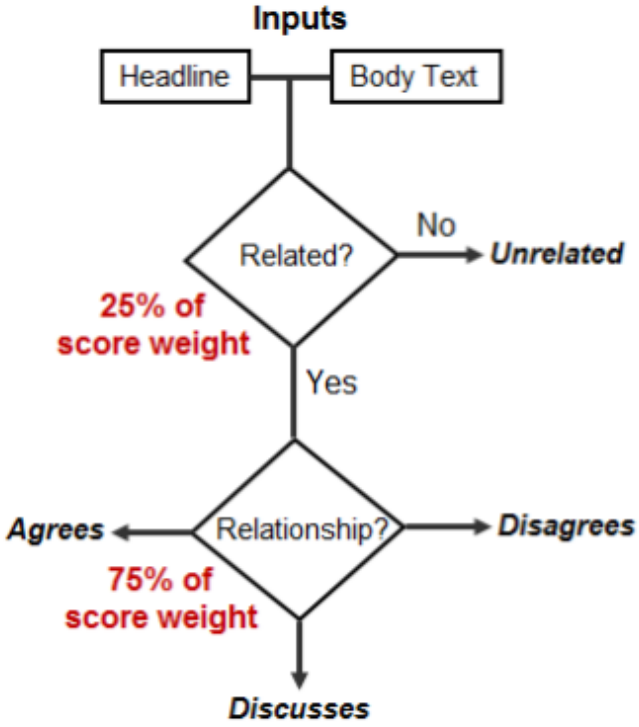
To compare how well the different models perform on specific labels, I will also calculate the Precision and Recall of every model for each label. Precision is calculated by dividing the True Positives by the sum of the True Positives and False Positives. This gives the following formula:

$$Precision = \frac{TP}{TP + FP}$$

Recall is calculated by dividing the True Positives by sum of the True Positives and the False Negatives. This gives the following formula:

$$Recall = \frac{TP}{TP + FN}$$

Figure 3: Evaluation scheme FNC [2]



4 Methods

4.1 UCL Method

On the GitHub page of the UCL machine reading method there was a brief tutorial on how to reproduce their code and results. As the code files were pushed onto their Git, reproducing the code was feasible. The main challenge was that this submission was dated back in 2018 and all the required packages are updated many times since then namely: Python, NumPy, TensorFlow, Scikit-learn. First I tried to execute the scripts with the most updated versions of the respective packages but because many functions were depreciated, this didn't work. With Conda I could make a virtual environment with specific versions of the packages, but many versions were not available any more from the repositories Conda uses. So, I downloaded the specific packages and depreciated the standard Python version which made the environment ready for the reproduction.

The code consists of two scripts called pred.py and utils.py. The utils.py contains the relevant functions, classes, global variables and imports the relevant modules. After I executed this script, I could run the pred.py script which contains the actual model and the pre-processing of the data. The script gives two options, one to train the model yourself or to import the already trained model. In order to fully replicate the method of the UCl team, I trained the model myself and I would check of this would work on the data set. However for the experiments I used the already trained model since the model uses shuffling during the training process. Because this could mean the model is trained slightly different than the trained model of the UCL team, it could result in a different outcome.

4.2 BERT Method

Raw text cannot be put in a BERT model, it needs some preprocessing first by changing the raw text into numerical values. This is done by two processes called tokenization and numeral encoding. Contrary to older research on natural language processing tasks, BERT doesn't need a lot of pre-processing because the tokenizer/numeral encoding, which is required for the model, does this for you. Common preprocessing steps like de-capitalization and stemming are not needed in this case because BERT takes these aspects as extra information about the sentence [1]. Because the stances and body texts were provided separately, they first needed to be joined together with body id as the key. The tokenizer basically splits a sentence in words (tokens) and the numeral encoder gives a numerical value to the token. The encoder uses a vocabulary, based on the implementation of the encoder, which could exceed more than 500,000 words where every word has a numerical unique ID. Next to that the process gives the type of every token and the attention mask, which means on which tokens the model should focus as I explained in the model section. To clarify, here is an example of a sentence tokenized and numeral encoded:

tokenizer("Leiden University is the best university in the world!")
ínput_ids': [101, 20329, 2118, 2003, 1996, 2190, 2118, 1999, 1996, 2088, 999, 102],
'token_type_ids': [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
'attention_mask': [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]}

A BERT model takes rectangular shapes as input so every input should have equal lengths. As

the Fake News data set contains a lot of sentences of different size, this is a problem. A technique called padding solves this problem. It makes every tensor (the result of tokenized text) of equal length by adding zeroes to a sentence to equal the longest tensor. Lastly the models used in this research require a maximal input size of 512 tokens so some inputs needed to be truncated at 512 because otherwise the computional complexity would be too high as I explained in the background section. In this research I concatenated the articles and the corresponding headlines and inserted a separator between them so the model knows the sentences are distinct. I chose to put the article behind the headline because otherwise it was possible the entire headline could be truncated as some articles are very long.

Different versions of Bert use also different tokenizers. For the three models, which I elaborated on in the background section, I used the following tokenizers:

- BERT - Autotokenizer
- DistilBERT – DistilBERT-Tokenizer
- RoBERTa – RoBERTa-tokenizer

After tokenizing every input sequence pair, I removed the columns which weren't needed any more and put the data in a DataLoader. For optimization I used the AdamW optimizer, and I used a linear scheduler for training. I chose the AdamW optimizer because it is the default use for an optimization task like this because of its performance and relatively low computing time. The data set was divided in a 80% training set and a 20% test set of the total data set. For the training I used these parameters for all the three models:

Table 4: Parameters for training

Parameter	Value
Number of epochs	3
Learning rate	5e-5
Batch size	8
Dropout rate	0.1

These parameters were either default or suggested by the Huggingface website on using BERT models [1]. Now the models are pre-trained, they are ready for the experiments.

5 Experiments

For this research I conducted 4 experiments:

- Replicating the UCL machine reading method
- Applying the finetuned BERT model to the test set
- Applying the finetuned DistilBERT model to the test set
- Applying the finetuned RoBERTa model to the test set

I conducted the experiments in a Google Colab environment, which is a cloud computing environment, where I had access to a GPU which is very suitable for NLP tasks like this. This ensures the computing time is much lower than if you compute this experiments with a CPU. The computing times for training were on average 2 hours for all the models. In order to compare the UCL method I first replicated the results of the UCL method.

Table 5: Confusion matrix of replication UCL machine reading team

True \ Pred.	**Agree**	**Disagree**	**Discuss**	**Unrelated**
Agree	838	12	939	114
Disagree	179	46	356	116
Discuss	523	46	3633	262
Unrelated	53	3	330	17963

If I compare the results of table 5 with table 2, you can see there is no difference in the classification of the labels. This means the replication was successful. This is expected because I used exactly the same trained model as they do.

Now I have the results of UCL model, I can apply the BERT models on the data to retrieve the predictions and compare them with the predictions of the UCL machine reading team.

Table 6: Confusion matrix of the BERT model

True \ Pred.	**Agree**	**Disagree**	**Discuss**	**Unrelated**
Agree	1377	51	425	50
Disagree	300	124	202	71
Discuss	744	125	3457	138
Unrelated	17	5	153	18174

Table 7: Confusion matrix of the DistillBERT model

Pred. / True	**Agree**	**Disagree**	**Discuss**	**Unrelated**
Agree	1205	86	540	72
Disagree	252	88	285	72
Discuss	491	96	3662	215
Unrelated	56	29	127	18137

Table 8: Confusion matrix of the RoBERTa model

Pred. / True	**Agree**	**Disagree**	**Discuss**	**Unrelated**
Agree	0	0	0	1903
Disagree	0	0	0	697
Discuss	0	0	0	4464
Unrelated	0	0	0	18349

If use the evaluation, provided by the FNC organization, I get the following results:

Table 9: Evaluation of models

	UCL	**BERT**	**DistilBERT**	**RoBERTa**	**SOLAT**	**Maximum score**
relative FNC-1 score	81.72	**85.85**	85.20	39.37	82.02	100,0
FNC-1 score	9521.5	9963.25	9926.75	4587.25	9556.50	11651.25

As you can see in Table 9, The BERT model has the best result on the test set by beating the score of the UCL team and the SOLAT team by a mininum of 3 points when you look at the relative score. The DistilBERT model also outperforms both the models. The RoBERTa model on the other hand performs really bad compared to all the other models. To get an understanding of the scores, confusion matrices give a lot of information but for comparison it is better to use the precision and recall for the different labels to determine which labels contribute the most to the score of the models and which labels were difficult to predict.

Table 10: Precision and recall per class of different models

	Classes							
	Agree		**Disagree**		**Discuss**		**Unrelated**	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
UCL	0.53	0.44	0.43	0.07	0.69	0.81	0.97	0.98
BERT	0.56	0.72	0.41	0.18	0.82	0.77	0.99	0.99
DistilBERT	0.6	0.63	0.29	0.13	0.79	0.82	0.98	0.99
RoBERTa	0.0	0.0	0.0	0.0	0.0	0.0	0.72	1.00

In table 3 you can see that is fairly easy for every model to predict if a pair of a headline and an article is Unrelated to each other and there is also little to no difference between the UCL model

and the other models. This is probably also why a model gets fewer points when it classifies a pair as Unrelated correct. What is also remarkable is that all the models can't predict the Disagree label very well as you can see from the low recall scores. The BERT and DistilBERT both predict the Disagree label better than the UCL model but it is still relatively low. The main difference in the scoring of the models is made in Discuss section when looking at the precision scores. The Discuss label gets an even amount of points when labeled correctly as the Agree and Disagree label, causing it to improve the score a lot. A reason why the model predicts the Discuss label better is presumably because the data set contains a lot more examples of a Discuss relation than the other two labels and also the reason why the model predicts the Unrelated label so well since 73% of the examples in the training set are unrelated. In general the BERT and DistillBERT models perform better on all labels but the outperforming is bigger when there are more examples in the training set of the specific label. The RoBERTa model on the other hand scores very low on all the labels except for unrelated label. As you can see in table 4 a possible explanation for this is that the model is over-fitted. It classifies all the relations as unrelated and due to the highly imbalanced data set (73% of the training set has the label unrelated), over-fitting could happen easily. This could have been fixed if a different learning rate or batch size was used.

6 Conclusions and Further research

The research question of this thesis was to examine it if was possible to improve the submissions of the Fake News Challenge of 2017. First I replicated the method of the UCL machine reading team and then I tested three different BERT methods on the Fake News data set. It is possible to improve the results of 2017. The DistilBERT and BERT method both outperformed the submission of the UCL machine reading team and even the teams which weren't replicated in this research, causing these results to be the number 1 submission if it was submitted in 2017. Given these results I can conclude that it is possible to outperform the submissions of the Fake News Challenge.

In the experiments section I concluded that the BERT models performed better relative to the UCL model on specific labels when there was more data available of this label. Because this research was limited to the Fake News Challenge, I could only use the provided data set. I think if there was more data available, especially the BERT model would outperform the UCL model even more. Of course with more data the UCL model would also perform better, but looking at the results, the BERT model would benefit more of this extra data.

The main goal of this research was to beat the score of 2017, which the models did but it could be even better. Since there was not much time for finetuning and computing power, the default parameters were used for the training. Especially when looking at the results of the RoBERTa model, there is a lot of room for improvement. It always predicts the majority class due to the highly imbalanced data set. Maybe with a different learning rate or batch size the model would perform better.

Since the data set was highly imbalanced, maybe this data set was not suitable for a Fake News Challenge like this. For detection of Fake News, the main goal is to check if texts agree or disagree with each other because that ultimately decides if one of two texts is Fake or wrong. As said in the background section, even for trained experts Fake News detection could be a cumbersome task. Classifying a pair as Unrelated, is not that hard actually and is also not meant by this statement. Than it is particularly strange that 73% of the data set consist of Unrelated pairs which thus don't really help in the detection of Fake News. A better balanced data set would benefit the research a lot.

For further research, there are some things left which could be improved. The results of this thesis were good because it beat the scores of 2017 but someone could make this model work even better if they altered more with the hyper parameters or even use other variants of BERT. Also this thesis was limited to the data set of the FNC but if someone could expand the data set and make it less imbalanced, the models would work even better.

Fake News detection is a challenging task and with help of techniques like Stance Detection it could be less challenging as a helping tool for experts By defining the relation between certain headlines and articles, experts could classify texts much faster as Fake News. With a relatively low amount of time spent on the hyperparameters of the models and trying different models, there are a lot of possibilities to improve the results and thus helping experts to battle the problem what is called Fake News even more.

6.1 Practical limitations

The servers of the University of Leiden are overcrowded, so when I tried to train models on these servers I usually got kicked out so I had to start over again. A solution for this was Google Colab but also with Google Colab, you can't make unlimited use of their GPU's. The problem is that you can only execute one process at the time and if the servers are overcrowded they kick you out. Even with an subscription this happens very often. Because of overusing the GPU, according to Google, I could not use it anymore for a undisclosed period of time. Then I would try to run the experiments with a CPU but because these kind of tasks are not really suitable for a CPU, the computing times would be very long. At first hand this could be a solution but because of the long run times, Google COLAB would eventually kick you out as well and would give a runtime error.

During this research I spent a lot of time expanding my research by reading newer articles and possibly went too deep in some techniques. Fake News detection is a very comprehensive task and it is even for humans a very difficult task. Two experts for example could have a different opinion on if something is Fake News. Due to its complexity and lots of literature written about this subject, I sometimes lost the focus on my research and was merely trying to expand my research. If I defined my scope better at the beginning of this research period, I would have gotten my results much sooner and I could spent more time on adjusting the models to get better results.

References

[1] https://huggingface.co/.

[2] http://www.fakenewschallenge.org.

[3] Juicekanaal roddelpraat moet uitzending over famke louise offline halen.

[4] Trump has longstanding history of calling elections 'rigged' if he doesn't like the results.

[5] Marina Azzimonti and Marcos Fernandes. Social media networks, fake news, and polarization. *European Journal of Political Economy*, page 102256, 2022.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[7] Society of Professional Journalists Fred Brown. www.spj.org/ethicscode.asp.

[8] Kyle Glandt, Sarthak Khanal, Yingjie Li, Doina Caragea, and Cornelia Caragea. Stance detection in COVID-19 tweets. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1596–1611, Online, August 2021. Association for Computational Linguistics.

[9] Kornraphop Kawintiranon and Lisa Singh. Knowledge enhanced masked language model for stance detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4725–4735, Online, June 2021. Association for Computational Linguistics.

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[11] Xiao Ma, Jeff Hancock, and Mor Naaman. Anonymity, intimacy and self-disclosure in social media. In *Proceedings of the 2016 CHI conference on human factors in computing systems*, pages 3857–3869, 2016.

[12] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. Bertweet: A pre-trained language model for english tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020.

[13] Kirsten Eddy Nic Newman with Richard Fletcher, Craig T. Robertson and Rasmus Kleis Nielsen. Reuters institute digital news report 2022, 2022.

[14] Dana Nuccitelli. the 97% climate crisis fox news defends global warming false balance by denying the 97% consensus, 2013.

[15] Carol D. Rebecca Tan, Peter Jamison. Trump supporters storm u.s. capitol, with one woman killed and tear gas fired.

[16] Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. *CoRR*, abs/1707.03264, 2017.

[17] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[18] Sean Baird with contributions by Doug Sibley and Yuxi Pan. Talos targets disinformation with fake news challenge victory, 2017.

[19] Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666, 2012.