Zurich University
of Applied Sciences

School of
Engineering

Bachelor thesis (Computer Science & Engineering and Management)

Misinformation Detection

Author	Marcel Avesani Ardi Jasari
Main supervisor	Mark Cieliebak
Sub supervisor	Jan Milan Deriu Pius von Däniken
Date	07.06.2024

Declaration of originality

I hereby declare that I have written this thesis independently or together with the listed group members.

I have only used the sources and aids (including websites and generative AI tools) specified in the text or appendix. I am responsible for the quality of the text and the selection of all content and have ensured that information and arguments are substantiated or supported by appropriate scientific sources. Generative AI tools have been summarized by name and purpose.

Any misconduct will be dealt with according to paragraphs 39 and 40 of the General Academic Regulations for Bachelor's and Master's Degree courses at the Zurich University of Applied Sciences (Rahmenprüfungsordnung ZHAW (RPO)) and subject to the provisions for disciplinary action stipulated in the University regulations.

Place	Date	Name
Winterthur	07.06.2024	Marcel Avesani
Winterthur	07.06.2024	A. Jasoni

Mandatory Declaration AI

We hereby declare that in the preparation of this thesis, we have utilized OpenAI's ChatGPT exclusively for the purpose of enhancing the formulation and presentation of the text and for creating code used in our experiments. The use of ChatGPT was strictly limited to refining the phrasing and clarity of information already derived from our research and assisting in the development of experimental code. This tool did not contribute to the generation of new content, the development of new conceptual ideas, or the creation of contextual data beyond these specified uses.

Zusammenfassung

Diese Arbeit untersucht die Erkennung von Fehlinformationen durch die Entwicklung und Verfeinerung von Deep-Learning-Modellen. Unser Ziel ist es, die Genauigkeit von automatisierten Fact-Checking-Systemen durch den Einsatz fortschrittlicher neuronaler Netzwerkarchitekturen zu verbessern. Unsere Methoden umfassen umfassende Datenanalysen, das Feintuning von vortrainierten Sprachmodellen (Pre-trained Language Models) und das Experimentieren mit Datenaufbereitungstechniken und Ensemble-Techniken zur Optimierung der Modellleistung. Die Ergebnisse zeigen, dass unsere Modelle Fehlinformationen effektiv klassifizieren, was durch unsere wettbewerbsfähige Leistung im CheckThat! Lab 2024 demonstriert wird. Diese Studie bestätigt die Wirksamkeit von Deep-Learning-Ansätzen, die die Automatisierung der Fehlinformationserkennung erheblich vorantreiben, was für die Bewahrung der Informationsintegrität in der heutigen digitalen Landschaft unerlässlich ist.

Schlüsselbegriffe:

Misinformation Detection, Deep Learning, Neural Networks, Automated Fact-Checking, Data Augmentation, Precision and Recall, CheckThat! Lab

Abstract

This thesis explores the detection of misinformation through the development and refinement of deep learning models. Our aim is to improve the accuracy of automated fact-checking systems by utilizing advanced neural network architectures. Our methods involve comprehensive data analysis, fine-tuning Pre-trained Language Model (PLM), and experimenting with data manipulation techniques and ensemble techniques to optimize model performance. Results indicate that our models effectively classify misinformation, as demonstrated by our competitive performance in the CheckThat! Lab 2024. This study affirms the efficacy of deep learning approaches in significantly advancing the automation of misinformation detection, which is essential for preserving information integrity in today's digital landscape.

Keywords:

Misinformation Detection, Deep Learning, Neural Networks, Automated Fact-Checking, Data Augmentation, Precision and Recall, CheckThat! Lab

Contents

1 Introduction

1.1 Background and Context

The recent growth of Online Social Networks (OSNs) has transformed how information is consumed and spread globally. While OSNs offer significant benefits like real-time information sharing, they also present challenges, particularly in managing misinformation. These platforms have become a catalyst for the spread of faulty claims, complicating efforts to maintain the integrity of information available to the general public [1]. The challenge of identifying and countering fake news on social media platforms is amplified because this content is often created with the intention to look like legitimate news. This makes it hard for automated systems to discern real information from fake news without additional checks [1].

The rise of AI-generated content on these networks introduces additional challenges, as these technologies can be used not only to detect misinformation but also to create it. This bifunctional use of AI highlights the increasing complexity of the digital information landscape and the need for advanced verification techniques [1]. The spread of misinformation is not limited to OSNs but is a significant concern in a number of media, including political debates where the integrity of information can significantly influence public opinion.

In the context of political debates, fact-checking has traditionally been a manual process performed by dedicated organizations. However, the fast spread of information necessitates automated approaches to keep up with the volume of content that needs to be verified. Zhijiang Guo et al. discuss how automation has become crucial in fact-checking, employing advanced techniques from Natural Language Processing (NLP) and Machine Learning (ML) to assess the truthfulness of claims quickly and effectively [2].

However, determining which claims to verify remains a challenge. Current approaches often assess the 'check-worthiness' of claims based on public interest rather than their verifiability, potentially overlooking important but less sensational claims [3]. This method of selection introduces biases, as noted by Konstantinovskiy et al., who advocate for a shift towards assessing claims based on the availability of evidence that can substantiate or refute them.

Our research utilizes the ClaimBuster dataset, which consists of statements from U.S. presidential debates categorized into Non-factual Statements (NFSs), Unimportant Factual Statements (UFSs), and Check-worthy Factual Statements (CFSs) by a team of human coders, primarily students and some professors [4]. This dataset provides a foundation for developing computational methods to identify claims that warrant fact-checking. Nonetheless, the subjective nature of this categorization process, influenced by the coder's perceptions of what constitutes public interest, highlights the need for more objective criteria in claim selection.

1.2 Objective and Research Question

Our research aims to refine and advance disinformation detection methodologies through the utilization of deep learning models. Our primary objective involves the development and fine-tuning of a deep learning-based model specialized for this purpose. Central to our approach is an in-depth analysis of

the datasets from the CheckThat! Lab 2024, which are predominantly sourced from the ClaimBuster dataset. The ClaimBuster dataset includes a significant amount of labeled text samples extracted from all U.S. general election presidential debates between 1960 and 2016.[4].

Our focus is placed primarily on gaining a comprehensive understanding of how deep learning techniques are presently utilized in the field of fact-checking, and to refine and optimize these methods according to our research's findings. Secondly, we're experimenting with adjusting various factors to enhance our model's performance. As a result, we undertake a comprehensive examination of the dataset mentioned, analyzing its contextual properties to make it more suitable for the training of our model.

The main functionality of our model lies in its ability to discern claims worthy of being fact-checked from those that are not. It classifies their label into either 'Yes' (indicative of checkworthiness) or 'No' (indicative of non-checkworthiness). This is crucial for automating the fact-checking process, thereby streamlining the arduous task typically associated with manual verification. Given the rapid increase of disinformation across online platforms, particularly in the realm of political discourse, the need for robust automated fact-checking mechanisms has become increasingly important [2].

The performance of our model was evaluated by the F1 score metric, defined as the harmonic mean of precision and recall, which serves as a measurement of the predictors efficacy in determining labels for unseen test data [5]. Our participation in Task 1B of the CheckThat! Lab 2024 involved benchmarking our model against other participants, with rankings determined by the F1 scores achieved on a blind test dataset, inaccessible during model development. Our model secured 5th place out of 26, with an F1 score of 0.771.

Given the dataset's source, our primary focus lied in the realm of political discourse [4]. However, we posit that the insights gained from our research hold broader usability for combating disinformation across diverse genres within the online landscape.

2 Literature Review and Theory

2.1 Definition of Misinformation

As shown in Figure 1, misinformation encompasses a variety of different types of information, all of which have in common that they are false or misleading. It is spread predominantly via the internet and has become increasingly important through the rise of social media. According to Alessandro Bondielli et al., "fake news" and "rumors" are the types most commonly recognized as misinformation within public discourse. It is important to note, however, that the term also includes clickbait, social spam, and fake reviews. All of these elements can significantly impact societal decision-making by distorting the public's perception [6].

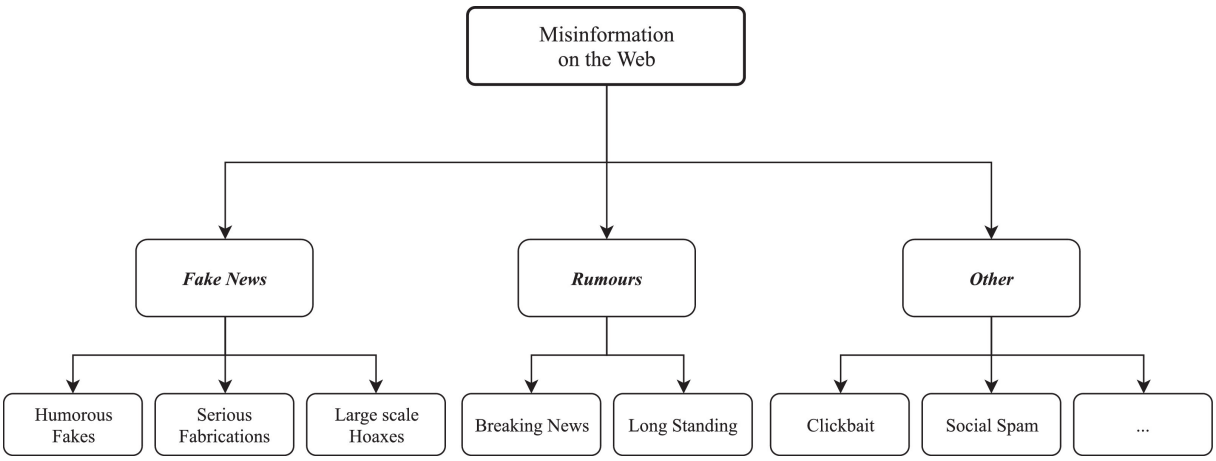

Figure 1: Different types of misinformation on the internet [6].

Misinformation is often not spread accidentally. Fake news refers to content created with the explicit intent to deceive. Sometimes, one is only capable of disproving it through arduous verification processes. Rumors, on the other hand, are unverified pieces of information that may or may not be true but spread rapidly due to their sensational nature, especially on social platforms [6].

In their study on Misinformation in Social Media, Liang Wu et al. expand this discussion by differentiating between misinformation and disinformation based on the originator's intent. According to them, disinformation is only a subset of misinformation which is created with the explicit intent to deceive. They emphasize, how in a social media environment, it can be difficult to discern if misinformation was created with intent or by accident. This makes the issue very complex for platform administrators and researchers alike. The Pizzagate incident, a conspiracy theory that ultimately led to a public shooting, is cited as an example of the potential consequences of misinformation [7].

Md Rafiqul Islam et al. emphasize the intention behind misinformation, characterizing it as a means to create distrust within communities. They call for proactive measures in order to ensure the integrity of social discourse [8].

Trough platforms like Facebook or Twitter it has become far easier to spread misinformation. Misleading

statements are frequently spread in conversations between individuals, where they are presented as factual. This can be particularly problematic in political contexts, where it may lead individuals who are misinformed to participate more actively and with more conviction in electoral processes than those who are uninformed [9].

In summary, misinformation is framed as a multifaceted problem that encompasses various types of incorrect or misleading information, each with individual origins and consequences. Addressing misinformation requires a nuanced approach that considers the intent behind the information, the medium through which it spreads, and the broader social context in which it exists [10].

2.2 Misinformation Detection: From Traditional Methods to AI

Traditionally, misinformation detection relied on manual verification by experts. However, the volume and velocity of online information are in need of an automated approach. Numerous researchers have developed systems as a means to detect and verify claims more efficiently. Most of those rely on advances made in ML and NLP [4].

There have been numerous studies on the application of ML methods on misinformation detection [11][12][13][14]. Early studies focus on more traditional ML methods, mostly feature based modelling algorithms such as Support Vector Machine (SVM), naive bayes or random forest classifiers. These are usually trained on lexical, n gram and sentiment features. Lexical features encompass measures such as number of words, average word length or the count of numbers.

2.3 The Role of Deep Learning

2.3.1 Neural Networks in Detection

While traditional ML algorithms can provide high prediction accuracy, a number of recent studies have shown that deep learning models, especially advanced PLMs, deliver superior results in the detection of misinformation [11][15]. These models, due to their extensive training on diverse datasets, can better generalize across different types of misinformation, thereby reducing the dependency on feature engineering typically required by traditional models [16].

Moreover, the adaptability of deep learning models allows for enhanced performance on complex misinformation types, which are often not well-handled by simpler, traditional algorithms. This is especially important as misinformation becomes more sophisticated. The pre-trained models' ability to process and learn from vast amounts of unstructured data also contributes to its improved performance in identifying the more subtle cues and patterns that may indicate misinformation, which traditional models might overlook due to their dependency on predefined features [2].

2.3.2 Advances in Transformer Models

The introduction of transformer-based models like BERT and RoBERTa can be regarded as a significant advance in misinformation detection. Transformer models are particularly adaptable to new and evolving datasets. By fine-tuning on specific misinformation data, they can quickly adjust to the thematic

context. Furthermore, they demonstrate robustness across diverse and challenging datasets, which is very important for handling the varied nature of misinformation as described earlier [17], [18].

2.4 Works on previous CheckThat! Lab editions

The CheckThat! Lab has already been held in previous years, and new methodologies for detecting check-worthy claims have been continually applied. Most participants based their classifiers on advanced PLMs like BERT and RoBERTa [19].

In their paper at CheckThat! 2023, Arkadiusz Modzelewski et al describe an innovative approach to fact-checking in multi-genre and multilingual content, employing GPT-3.5 for data augmentation. Arkadiusz Modzelewski et al. implemented data augmentation techniques, including generating paraphrases and translating text fragments, to enhance the performance of a pre-trained XLM-RoBERTa model. This methodology significantly improved the system's efficiency, particularly in Spanish, where their system achieved first place with an F1 score of 0.641 [20].

Several teams used ensemble methods to make the classification model more robust [21]. Martinez-Rico et al. combined the transformer model with two Feedforward Neural Networks (FFNNs) that processed inputs in the form of Term Frequency-Inverse Document Frequency (TF-IDF) vectors and Linguistic Inquiry and Word Count (LIWC) features to ensure a comprehensive analysis of the texts [22]. The use of transformer models in combination with traditional ML techniques together provided a robust system for assessing check-worthiness. Von Däniken et al. used ensembling to combine various unimodal and multimodal classifiers through a Multiple Kernel Learning process [21]. This technique allows leveraging the advantages of individual models while compensating for their individual weaknesses. Their developed system achieved a notable second place among seven teams with an F1-score of 0.708 for a multimodal check-worthiness task.

Sawiński et al. employed GPT and BERT models in their study and compared performances across several techniques, including zero-shot and few-shot learning, as well as fine-tuning [23]. The study found that fine-tuned BERT-based models were able to achieve performance comparable to large language models like GPT-3 in identifying check-worthy statements. This raises important questions regarding the superiority of GPT models over BERT-based models in certain applications.

3 About the ClaimBuster Dataset

3.1 Transcript Extraction and Annotation Process

The ClaimBuster dataset was created by extracting sentences from transcripts of all U.S. presidential debates from 1960 to 2016. Initially, sentences spoken by the presidential candidates were identified using both automated parsing tools and manual verification to ensure accuracy. Sentences spoken by others, such as moderators or audience members, were excluded to maintain focus on the candidates' statements. Short sentences under five words were also removed to concentrate on more substantive content. This selection process resulted in a dataset of 23,533 sentences. [4].

The following Figures 2 and 3 show an evolution of sentence volume and average sentence length in U.S. presidential debates over time. While the total number of sentences per debate has risen, the average sentence length per debate seems to have declined across the timeline. This observation aligns with the general shift in english language towards the usage of shorter sentences [24].

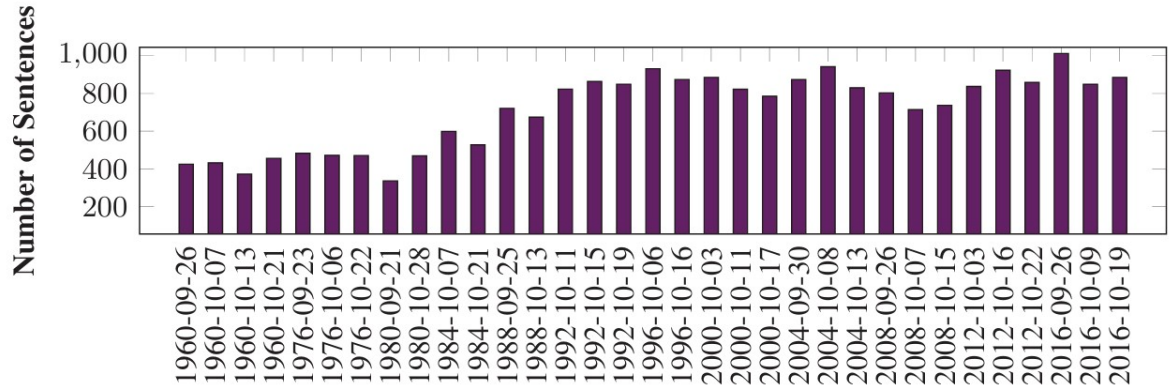

Figure 2: Sentence distribution among presidential debates [4].

Figure 3: Average sentence length in words per debate [4].

3.2 Annotation Procedure

The annotation process was an important part of developing the ClaimBuster dataset [4]. The sentences were classified into one of three categories:

NFSs: Sentences that did not contain verifiable claims.

UFSs: Factual statements that were considered trivial or not significant enough to require verification.

CFSs: Factual statements determined significant enough to require verification due to their potential impact on public discourse.

Figure 4 illustrates the distribution of these categories across all U.S. presidential debates from 1960 to 2016 from which the sentences were gathered. While there are some variations across the timeline, there is no discernible trend in the distribution of categories over time.

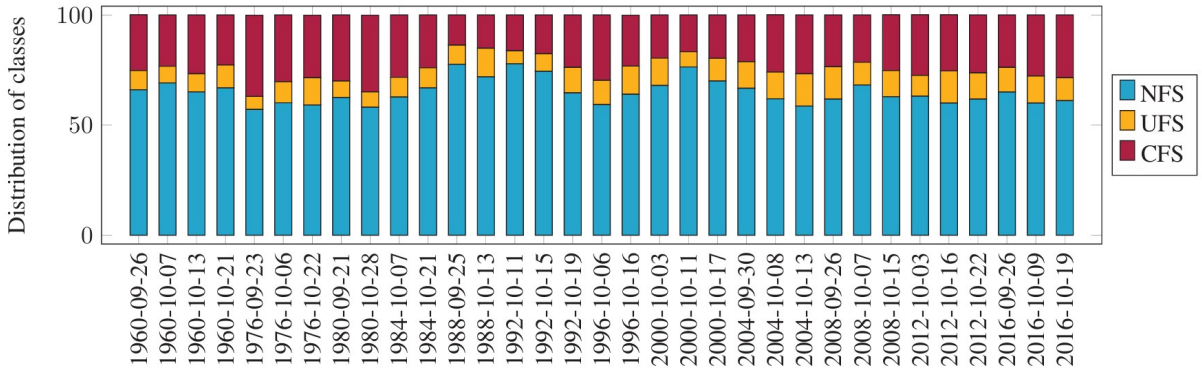

Figure 4: Category distribution per debate [4].

To ensure the objectivity and reliability of the annotations, each sentence was reviewed by multiple coders over a 26-month period. The coders were trained and provided with detailed guidelines to help them categorize the sentences with improved accuracy. The involvement of multiple coders aimed to minimize individual biases and improve the consistency of the annotations [4].

3.3 Quality Control Measures

To maintain the targeted high accuracy of annotations, a subset of the sentences was used to create a 'ground-truth' dataset. This subset was annotated by a team of three expert reviewers. On average, every tenth sentence presented to the coders for annotation was from this 'ground truth' dataset. Based on these already labeled sentences, the accuracy of the predictions made by each coder was tested. With a point system that affected the money earned per annotated sentence, coders were penalized for low annotation accuracy but rewarded for high accuracy. Their performances on these control sentences were used to assess their overall reliability and accuracy in annotation [4].

3.4 Ethical Considerations and Fairness

In the paper, the coders are described demographically as primarily students, supplemented by some professors and journalists. They were invited to participate using flyers, social media, and direct emails. There is an argument to be made about potential biases existing, mainly because most participants were students. Their views and political leanings might have a general consensus that deviates from that of the general public. However, the method of invitation through flyers and social media—tools used across all demographic groups—suggests that there were participants from different fields of study and various backgrounds. Secondly, since their performance was observed and continuously evaluated through the point system, annotations made by biased coders were actively excluded from the final evaluation [4].

4 Methodology

4.1 Data Analysis

4.1.1 Data Analysis Overview

In our initial analysis, we examined the number of samples in each of the three datasets and the ratio of positive to negative labels within them. Figure 5 illustrates the distribution of sample counts across the three datasets: the training set, the development set, and the development-test set.

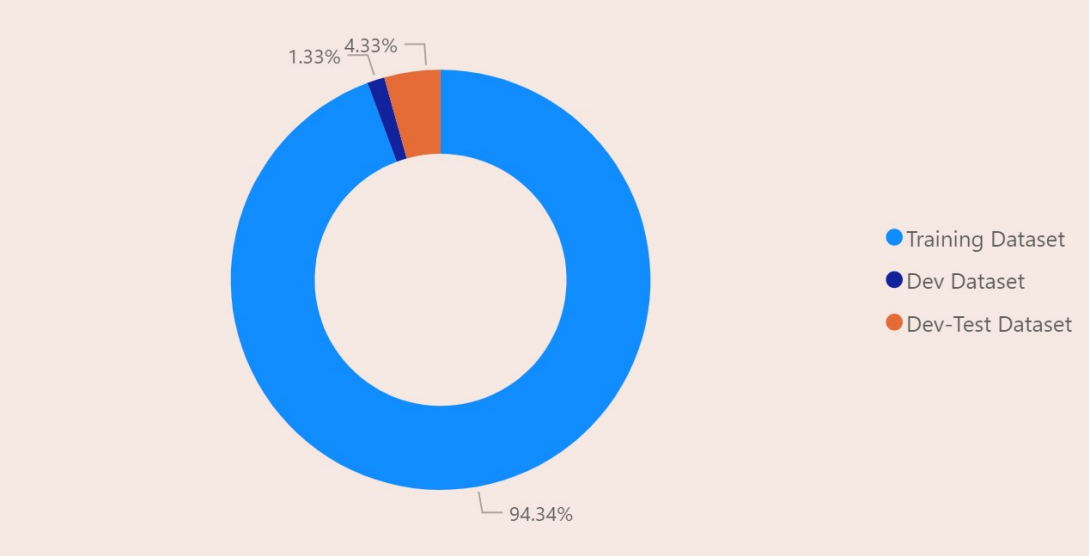

Figure 5: Distribution of sample sizes across the three datasets

The training dataset is significantly larger, containing 22,501 samples, compared to the development dataset, which includes 1,032 samples used for validation during model training. The development-test dataset, used for model testing, is smaller yet, containing only 318 samples. This notable difference in sample sizes is primarily because the training data corresponds exactly to the crowd-sourced portion of the ClaimBuster dataset, while the development data matches the ground truth dataset described in the ClaimBuster paper [4]. We assume this arrangement of data is intentional but remain open to the possibility of re-distributing or mixing the data if necessary. Moreover, the CheckThat! Lab categorized the CFSs sentences from the ClaimBuster dataset as checkworthy and the UFSs and NFSs sentences as non-checkworthy.

4.1.2 Text Sample Length Distribution Analysis

Next, we analyzed the distribution of the lengths of the text samples, both overall and segmented by class. Figures 6 to 8 depict how the lengths of the samples are distributed across the datasets.

As expected, there are significantly more shorter samples than longer ones. However, it is noteworthy that the average length of positive samples is considerably longer than that of negative samples. We also note that the dev-test set contains shorter sentences on average. This aligns with our earlier assumption

that its samples do not originate from the ClaimBuster dataset, unlike the other two sets. The source of these samples remains unidentified.

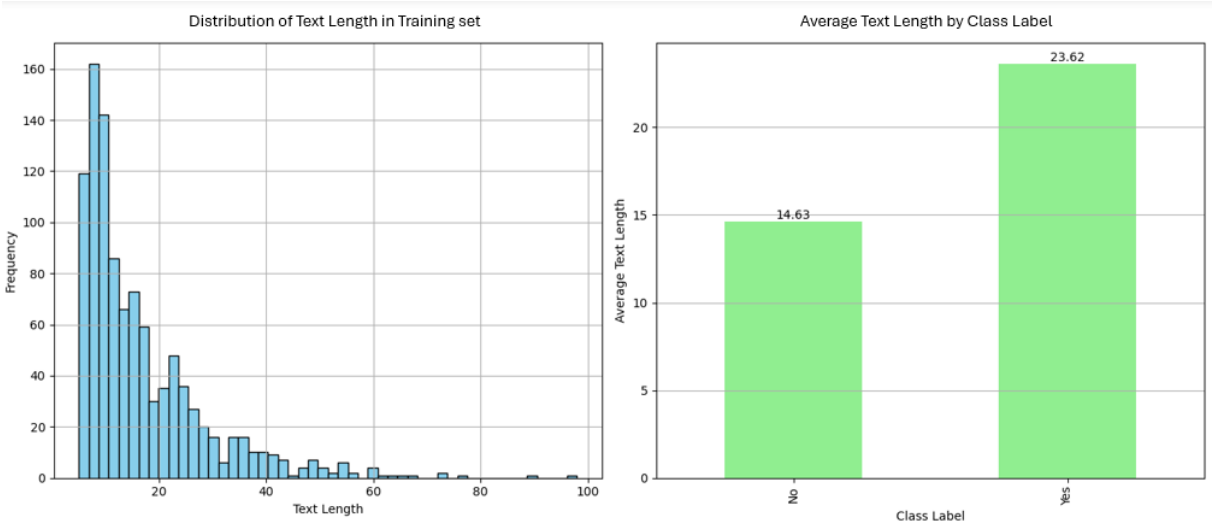

Figure 6: Distribution of text lengths in the training dataset by class label

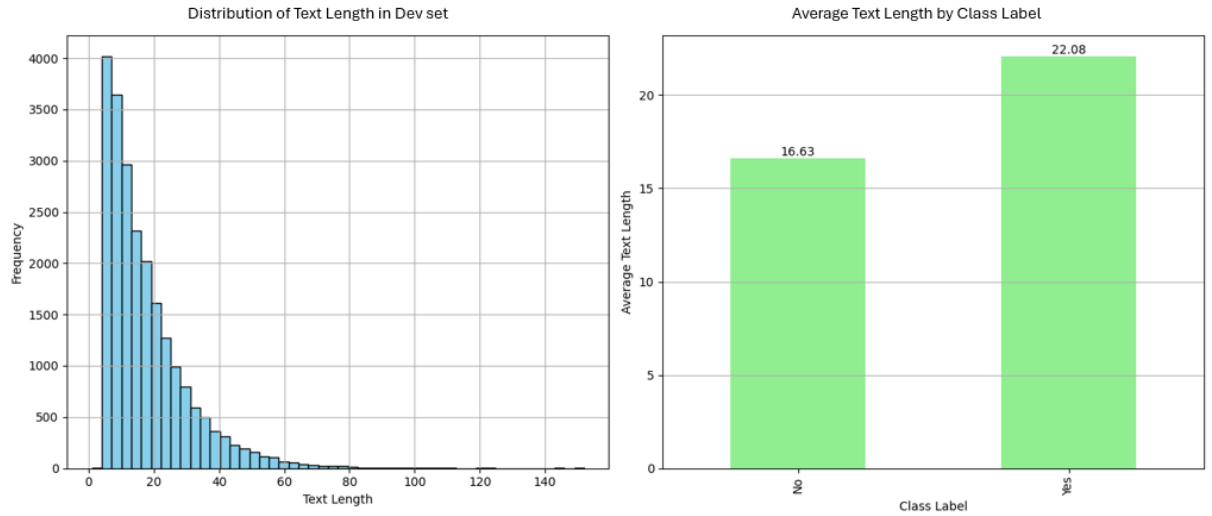

Figure 7: Distribution of text lengths in the development dataset by class label

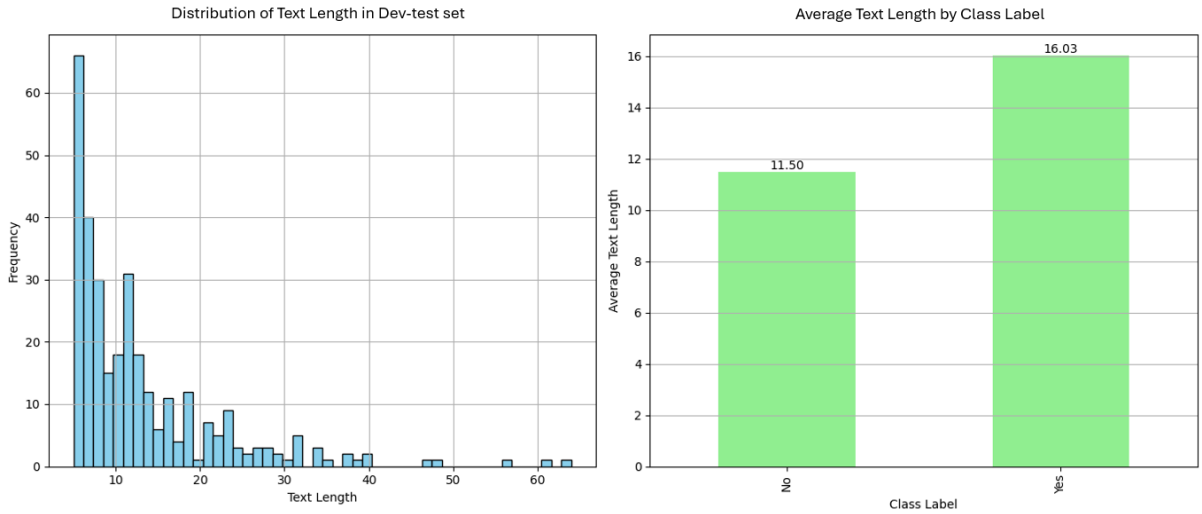

Figure 8: Distribution of text lengths in the development-test dataset by class label

According to the ClaimBuster paper, sentences shorter than five words were excluded from the dataset [4]. Nonetheless, sentences containing only five words are less likely to include a claim due to their limited length, making it challenging to formulate a complete claim within such a brief context.

While this observation may seem trivial, it has important implications for the training of our deep learning models, which we will discuss in one of the following chapters. This insight into sample length distribution could influence model performance and the strategies we employ during the training process.

4.1.3 Distribution of Class Labels

According to the paper, all statements from the debate transcripts spoken by political candidates were used, except for those under five words in length as mentioned [4]. This suggests that the distribution of check-worthy labels across the datasets may be more random than intentional.

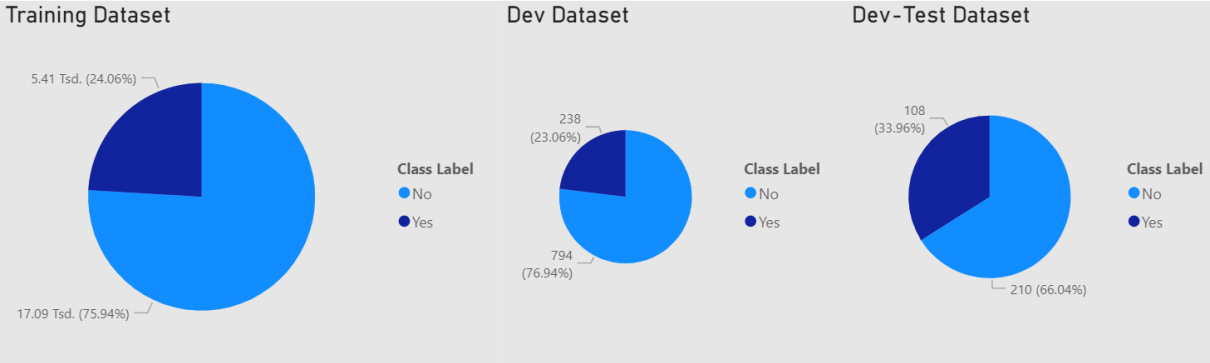

Figure 9: Distribution of class labels across the three datasets

Interestingly, as shown in Figure 9, the ratio of negative to positive labels in both the training and development datasets is approximately 3:1. In contrast, the ratio in the test dataset markedly differs, at about 2:1 in favor of negative samples. This disparity validates our previous assumption that the test sample may originate from a different source.

4.1.4 Statistical Text Analysis

To better understand the thematic content of our training dataset, we started by determining the most common words based on the classification label of the text samples in which they appear, initially by simply counting their occurrences. Figure 10 displays the words that appear most frequently in each of the two 'classes'.

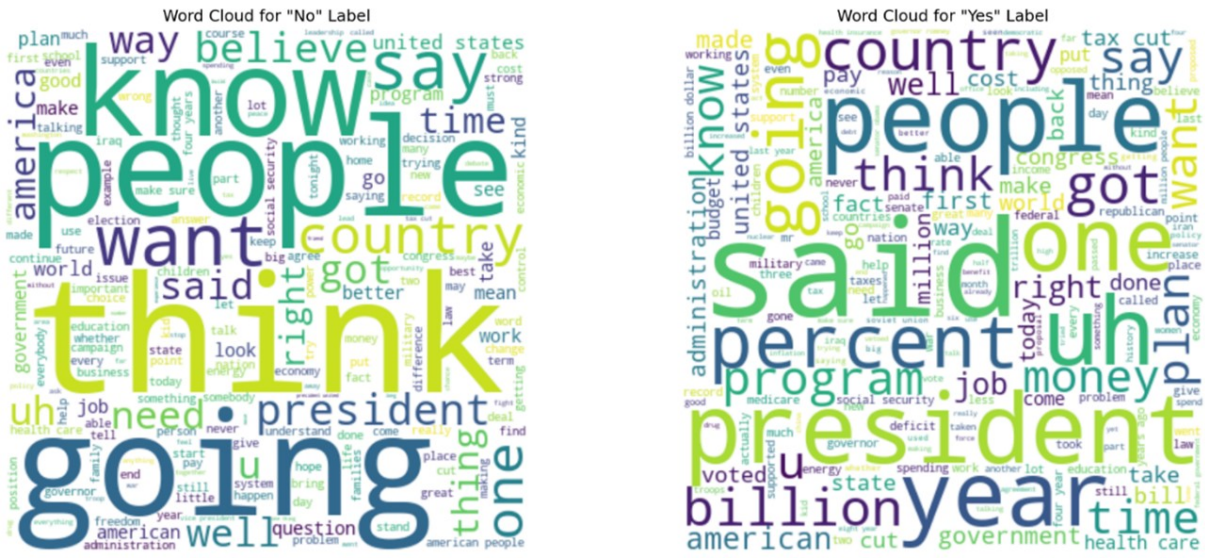

Figure 10: Most common words in the training dataset per class label

It is important to note that very short words, known as 'stop words', were excluded from this analysis because they often appear frequently but do not significantly alter the meaning of a sentence.

To further enhance this analysis, we conducted a TF-IDF analysis on the training dataset, considering both unigrams and bigrams. TF-IDF is a statistical measure used to evaluate how important a word is to a document within a collection of documents [25]. In this context, 'Term Frequency' refers to the raw count of a term in a document, which can be adjusted by the length of the document or by the raw frequency of the most frequently occurring word in the document. 'Inverse Document Frequency' measures how unique a word is across the entire dataset; it is calculated by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient. A word common across many documents will have a lower score, approaching 0, while rarer words will approach a higher value, near 1.

The product of these two metrics, the TF and the IDF, results in the TF-IDF score. A higher TF-IDF score indicates that a word is more relevant to a specific document within the dataset [25].

Following the TF-IDF analysis, we trained a SVMs classifier model on this vectorized matrix from the training dataset to predict the labels of the development and test datasets. We utilized a LinearSVC with parameters set for balanced class weight and a fixed random state for reproducibility. To determine which terms most strongly influence the classifier's decisions towards positive or negative labels, we analyzed the coefficients of the n-grams. We sorted these coefficients by their values, identifying the most positive and the most negative coefficients. We then matched these coefficients to their corresponding n-gram indices within the model to pinpoint the specific n-grams associated with the most significant influences on the classifier's decisions.

Results

The graphic in Figure 11 shows the n-grams with the highest and lowest coefficients obtained from the SVM classifier.

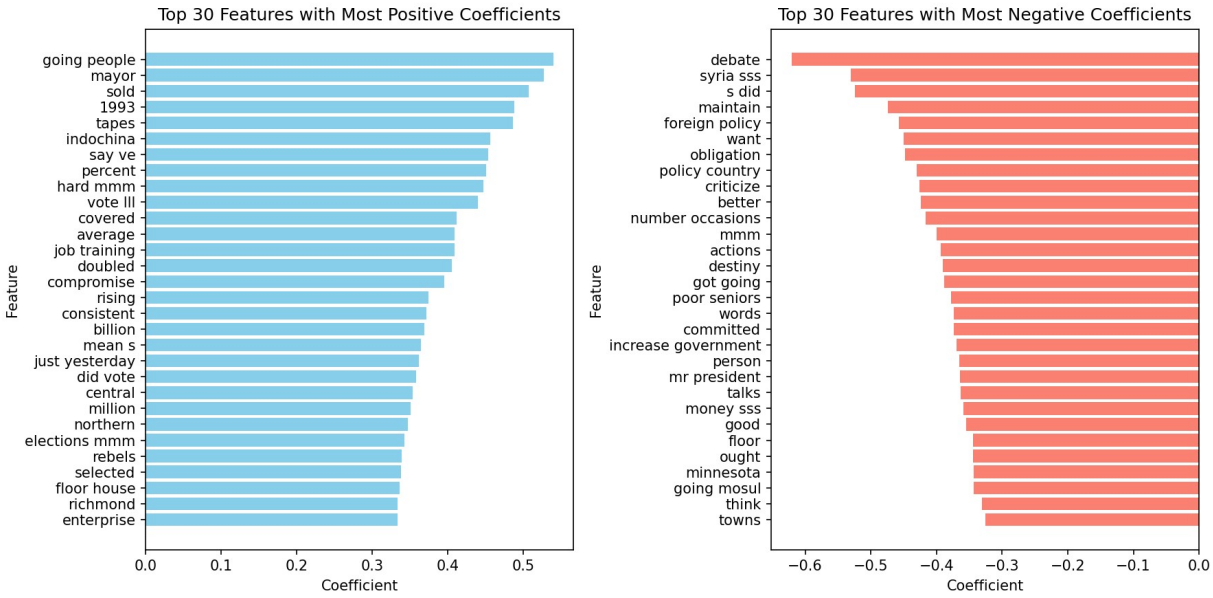

Figure 11: Influential n-grams in classification decisions of the svm model

Discussion

The n-grams with the highest positive coefficients likely represent phrases that are commonly associated with assertive, claim-like statements. These might include terms related to significant public and political issues (e.g., "million" "percent" "doubled","just yesterday") which are often discussed in debates and are typical targets for fact-checking. This suggests that the SVM classifier has effectively learned to identify features that are characteristic of statements that need further verification.

Conversely, the n-grams with the most negative coefficients represent phrases less likely to involve claims needing verification. These might include more generic, discussion-based or context-setting phrases (e.g., "debate," "mr president," "policy country") that do not assert facts that can be easily checked. This indicates that the classifier is effectively distinguishing between content that typically contains information requiring validation and general discourse.

The SVM classifier was not utilized for pre-selection of claims prior to the training of our deep learning models. However, the identifiable n-gram patterns in the dataset indicate that it possesses learnable features, enabling effective classification of samples into respective categories by the deep learning models.

4.2 Fine-tuning

Given the success of PLMs in text classification tasks [26], we decided to adopt this approach for our classification task. The first step was to select state-of-the-art PLMs and to fine-tune them with the original dataset described in 4.1. To select the PLMs, we conducted thorough literature search using keywords such as 'Text classification', 'Transfer Learning in NLP', 'BERT-based Models'. Subsequently, we prioritized models that had demonstrated strong performance on natural language understanding (NLU) benchmarks. Table 1, adapted from [27], provides an overview of the performance of the models on the MNLI matched/mismatched (m/mm) [28] and on the SQuAD v2.0 [29] development set. Another criterion taken into account is the availability of the model on HuggingFace[1]. Hugging Face provides a convenient platform for accessing PLMs. Based on these criteria, we selected the following PLMs for the fine-tuning process: $\text{RoBERTa}_{\text{base}}$[2] [30], $\text{XLNet}_{\text{base}}$[3] [31], $\text{ELECTRA}_{\text{base}}$[4] [32], $\text{DeBERTaV3}_{\text{base}}$[5] [27].

Model	MNLI-m/mm(ACC)	SQuAD 2.0(F1/EM)
$\text{RoBERTa}_{\text{base}}$	0.876/-	0.837/0.805
$\text{XLNet}_{\text{base}}$	0.868/-	-/0.802
$\text{ELECTRA}_{\text{base}}$	0.888/-	-/0.805
$\text{DeBERTa}_{\text{base}}$	0.888/0.885	0.862/0.831
$\text{DeBERTaV3}_{\text{base}}$	0.906/0.907	0.884/0.854

Table 1: Performance of state-of-the-art PLMs. Adapted from [27]

All the selected models are based on BERT, which stands for Bidirectional Encoder Representations from Transformers. BERT is a language representation model. It was pretrained using two unsupervised tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). In MLM 15% of the words in the sentence are masked with a [MASK] token and the model is trained with the objective to predict those masked words based on their left and right context. MLM enables the language representation to be conditioned on both the left and right context. Through NSP, the model learns to predict whether a given sentence B follows a given sentence A, which helps the model to understand sentence relationships. [33]. However, all the selected models omit the NSP task stating that the NSP task does not significantly improve model performance [30]–[32].

RoBERTa improves BERT by optimizing the pretraining procedure. Specifically, it removes the NSP task and uses dynamic masking instead of static masking. Additionally, RoBERTa uses larger training times, larger batches and approximately ten times more training data. Specifically, it was trained on a combination of BookCorpus, English Wikipedia, CC-News, OpenWebText, and Stories, totalling over 160 GB of data [30].

XLNet is an autoregressive pretraining model that addresses some limitations of BERT by using a permutation-based training objective, allowing it to model bidirectional context without the need for masking. In this permutation-based training, the model considers all possible arrangements of the words in the

[1] `https://huggingface.co`.
[2] `https://huggingface.co/FacebookAI/roberta-base`.
[3] `https://huggingface.co/xlnet/xlnet-base-cased`.
[4] `https://huggingface.co/google/electra-base-discriminator`.
[5] `https://huggingface.co/microsoft/deberta-v3-base`.

sentence, not just the original order [31]. $\text{XLNet}_{\text{base}}$ was pretrained on BooksCorpus, English Wikipedia, Giga5, ClueWeb 2012-B and Common Crawl, which combined give over 158 GB of data [31], [34].

ELECTRA is a discriminator model that builds upon the bidirectional modeling ideas of BERT but replaces the MLM task with a more compute-efficient task called Replaced Token Detection (RTD). In RTD, instead of masking the words, they are replaced with plausible alternatives generated by a small generator network. ELECTRA is then trained to distinguish between original and replaced tokens. This approach is more compute-efficient because the model learns from the entire input sentence rather than just the smaller masked subset [32]. The Base version of ELECTRA is pretrained on the same dataset as XLNet [32], [35].

DeBERTa improves upon BERT by introducing two main techniques: disentangled attention and an enhanced mask decoder. In disentangled attention, each word is represented by two vectors, one for content and one for position. This allows the model to better capture the relationships between words and their positions in a sentence. The enhanced mask decoder improves the model's ability to reconstruct masked tokens by using a more sophisticated decoder [27]. The latest version of DeBERTa, DeBERTaV3, incorporates the RTD task from ELECTRA, replacing the MLM task. DeBERTaV3 was pretrained using the same data as RoBERTa [27].

To ensure robust evaluation, multiple fine-tuning runs for each model were conducted. This approach helps to mitigate the impact of randomness in training and provides a more reliable estimate of the models' performance. The training set was used to fine-tune the models, the development set was used to evaluate performance during training and to select the best checkpoint, and the development-test set was used to evaluate the final performance of each model on unseen data. The hyper-parameters utilized are consistent with those employed by von Däniken et al. [21] in their fine-tuning of the *electra-clf* model for the CheckThat! Lab 2023. The training hyper-parameters are detailed in Table 2. Additionally, we introduced a checkpoint saving strategy every 400 steps during training. At the end of training, the checkpoint with the highest F1 score on the validation set was selected as the final model, since the official evaluation metric for Task 1 of the CheckThat! Lab 2024 is the F1 score over the positive class. We observed that this checkpoint loading strategy sometimes selected checkpoints where the evaluation loss indicated potential overfitting. This raised concerns that such checkpoints might not generalize well to unseen data. To address this, we experimented with using the evaluation loss as the criterion for the best checkpoint. Despite this adjustment, we did not observe an improvement in performance and, in some cases, resulted in worse performance on the development-test set, so we reverted to using the F1 score as the primary criterion. Additionally, we experimented with early stopping based on the F1 score, with patience set to 5 and 10 (evaluating every 100 steps) to prevent possible overfitting. However, this strategy also did not lead to improved performance and, in some cases, resulted in worse outcomes. Thus, we concluded that using the F1 score for checkpoint selection, without early stopping, provided the best results.

Hyper-parameter	Value
Epochs	10
Batch Size	16
Optimizer	AdamW[36]
Learning Rate	5e-5
Weight Decay	0.01
Warmup Steps	500
Learning Rate Decay	Linear

Table 2: Hyper-parameters for fine-tuning the PLMs. Adapted from [21]

Results

Table 3 presents the scores of each fine-tuned model on the development set and on the development-test set. For each model, we report the mean and standard deviation of the accuracy, precision, recall, and F1 score over multiple runs.

Model	F1 Score	Accuracy	Precision	Recall
Scores on development set				
RoBERTa	0.944 ± 0.010	0.974 ± 0.005	0.932 ± 0.041	0.958 ± 0.024
XLNet	0.939 ± 0.004	0.971 ± 0.001	0.930 ± 0.015	0.948 ± 0.015
ELECTRA	0.941 ± 0.011	0.973 ± 0.005	0.939 ± 0.006	0.942 ± 0.017
DeBERTaV3	0.947 ± 0.004	0.975 ± 0.002	0.933 ± 0.003	0.962 ± 0.006
Scores on development-test set				
RoBERTa	0.818 ± 0.028	0.890 ± 0.009	0.934 ± 0.056	0.732 ± 0.079
XLNet	0.821 ± 0.026	0.892 ± 0.016	0.930 ± 0.026	0.732 ± 0.026
ELECTRA	0.832 ± 0.017	0.899 ± 0.011	0.951 ± 0.034	0.741 ± 0.025
DeBERTaV3	0.849 ± 0.030	0.909 ± 0.016	0.965 ± 0.009	0.759 ± 0.052

Table 3: Performance metrics of fine-tuned models on development set and on development-test set

On the development set, DeBERTaV3 achieved the highest mean F1 Score of 0.947, closely followed by RoBERTa with a mean F1 score of 0.944. Overall, all models achieved high scores on the performance metrics. The standard deviations were relatively low across all models, indicating consistent performance during multiple runs.

For the development-test set, DeBERTaV3 again showed the best performance with a mean F1 score of 0.849. ELECTRA followed with a mean F1 score of 0.832. The standard deviations were slightly larger on the development-test set, indicating more variability in performance across different runs.

When comparing the performance across the two sets, there is a noticeable drop in the F1 score when the models are evaluated on the development-test set. The recall metric is particularly affected, e.g., for the DeBERTaV3 model, recall drops by 21.1%. Precision, on the other hand, remains stable or even improves in some cases in the development-test set.

Figure 12 illustrates the mean training and evaluation loss over multiple runs for the DeBERTaV3 and RoBERTa models during the fine-tuning process. The training loss starts high, around 0.6, and drops sharply within the first 2,000 training steps, reaching approximately 0.2. It continues to decrease but at a slower rate, dropping below 0.1 with minor fluctuations around 6,000 steps. After 8,000 steps, the training loss converges to 0. On the other hand, the evaluation loss starts around 0.3 and stabilizes around 0.1 in the first 1,000 steps. After 4,000 steps, it starts to gradually increase, reaching 0.2 at around 7,000-step mark and approximately maintaining this loss until the end of the training. Most of the other models show a similar course. In the case of the RoBERTa models, the training loss decreases more slowly compared to the other models. However, the evaluation loss for RoBERTa remains slightly below that of the other models after the 6,000-step mark. When training the RoBERTa models for more epochs, specifically 20, the training and evaluation loss follow a similar trajectory as the DeBERTaV3 model, but stretched over the longer duration.

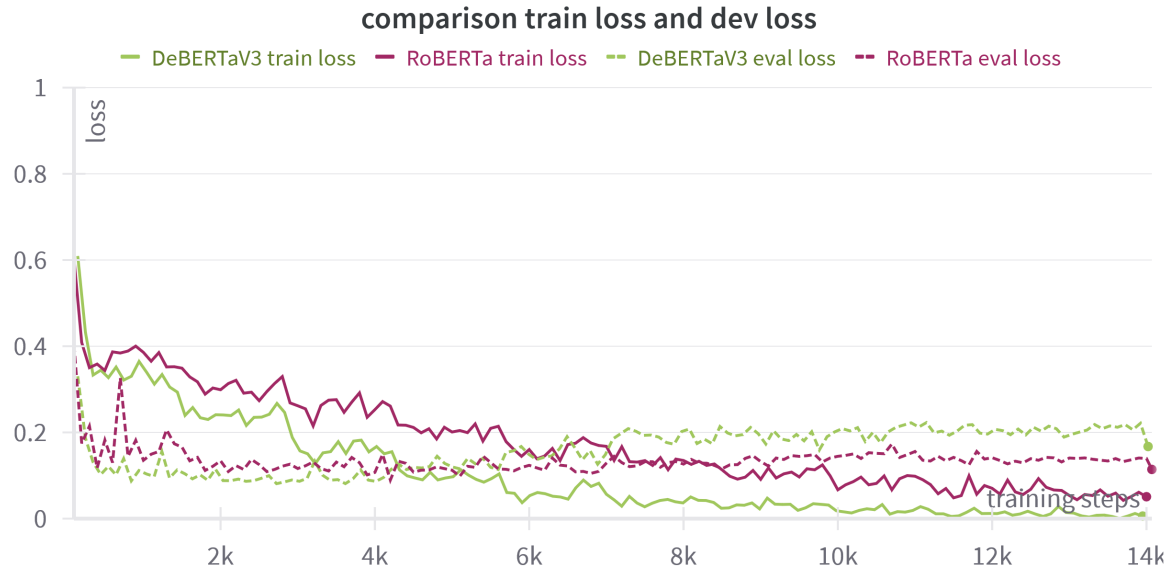

Figure 12: Loss curves on the training set and development set for the DeBERTaV3 and RoBERTa models. To maintain clarity in the plot, the loss curves for the other models, which exhibit similar patterns to the DeBERTaV3 curves, are omitted.

To further analyze the performance of our fine-tuned models, we generated the Precision-Recall (PR) curve shown in Figure 13. The PR curve provides a more insightful performance evaluation in the context of imbalanced datasets, contrary to the Receiver Operating Characteristic curve, which can give an overly optimistic view of performance. The Average Precision (AP) is a metric that summarizes the PR curve [37]. The PR curves demonstrate that ELECTRA achieved the highest AP score of 0.96, indicating the best balance between precision and recall among the models tested. Both XLNet and DeBERTaV3 followed closely with AP scores of 0.95. RoBERTa, while still performing well, had a slightly lower AP score of 0.90

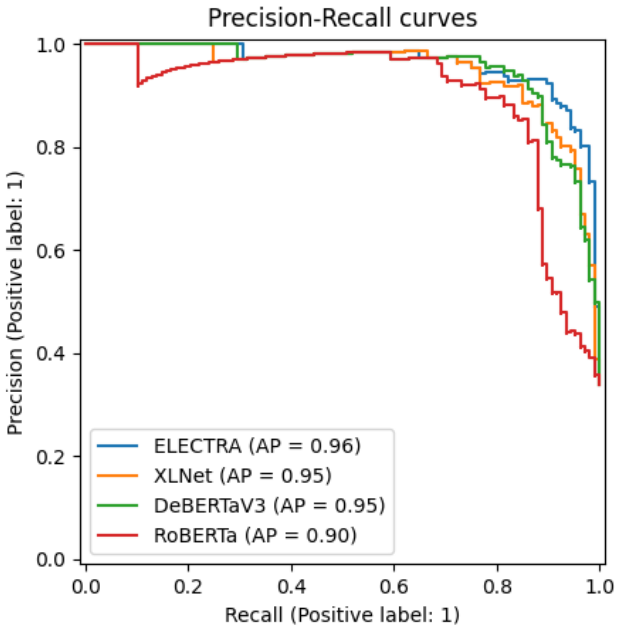

Figure 13: PR curves for fine-tuned models on the development-test set.

Discussion

The DeBERTaV3 model achieved the highest F1 scores on both sets suggesting that it is the most effective model for this task among those tested. This is reasonable since DeBERTaV3, as seen on Section 4.2, further enhanced their model based on the successful RTD technique from ELECTRA, and outperforms the other models on NLU benchmarks.

The observed drop in performance on the development-test set suggests that this set contains more challenging or diverse examples. In Section 4.1 we describe that:

1. The origin of the development-test set is unknown, and it is unclear whether the same methods used for labeling the training and development sets were applied to the development-test samples.

2. The class label distribution differs between the development-test set and the other sets.

3. The development-test set contains shorter sentences on average compared to the training and development set.

These factors could explain the significant performance gap between the development and development-test sets. Given the lack of information to address the first point, we decided to focus on the second point. The third point was discovered later in our research, which is why it was not addressed in our initial analysis.

The high precision observed indicates that when the model predicts a sentence as checkworthy, it is

usually correct. This implies that the model is good at avoiding false positives. On the other hand, The low recall means that the model is missing many checkworthy sentences. It fails to identify numerous actual positives, resulting in a high number of false negatives. Overall, having high precision and low recall suggests that the model is very cautious and selective in predicting the positive class.

To address the drop in performance observed on the development-test set and the imbalance between precision and recall, we considered the following strategies, which we describe in detail in the subsequent sections:

- Adding more diverse examples of the positive class to the training set to help the model better recognize positive instances.
- Adjusting the decision threshold to find a better trade-off between precision and recall.
- Using ensemble methods to combine the strengths of multiple models, potentially improving both precision and recall.

The rapid decrease in both training and evaluation loss during the initial phase of training indicates effective early learning, with the model quickly capturing the primary patterns in the data. The continuing decrease in training loss and the slight increase in evaluation loss hint at potential overfitting. This scenario, where the model performs better on training data but worse on unseen data, should be prevented through the checkpoint loading strategy described in Section 4.2.

4.3 Data Manipulation: Analyzing Effects on Model Performance

To further enhance the F1 score and optimize our model's predictive accuracy, we explored strategic modifications to the dataset configuration. Our primary objective was to enable the model to learn more effectively from the training data.

4.3.1 Sample Redistribution across Datasets

Our initial approach involved shuffling the three datasets—training, development, and development-test—so that all samples were evenly distributed among them. The rationale behind this strategy was inspired by a paper concerning the ClaimBuster dataset, which suggested that the development dataset arguably possesses more reliable class labels due to its 'ground truth' data [4].

By redistributing the samples, our intention was to introduce a higher proportion of reliable data into the training set, potentially enhancing overall model performance. This shuffling was performed using a random algorithm, ensuring that the distribution of class labels across each dataset remained random. However, we maintained the original size of each dataset split (22,501 in the training dataset, 1,032 in the development dataset, and 318 in the test dataset. Figure 14 illustrates the near-even distribution of class labels following the shuffling process.

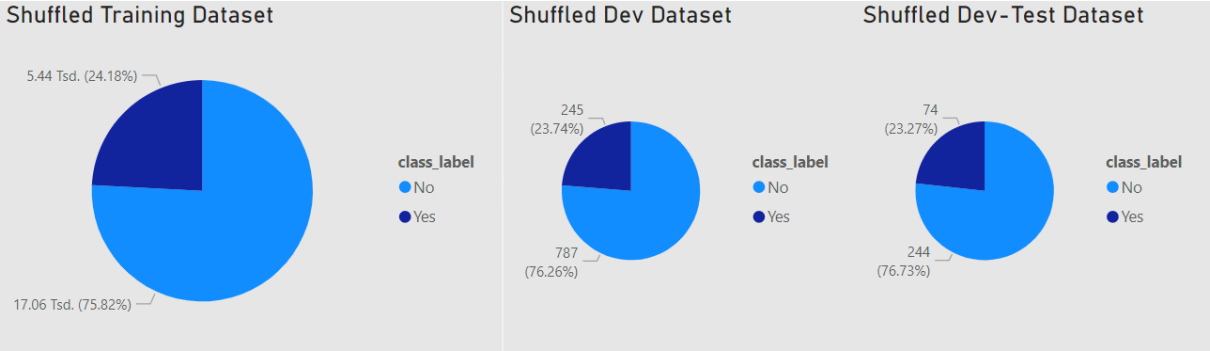

Figure 14: Distribution of class labels across the three shuffled datasets

To evaluate the impact of this redistribution, we trained three of our best-performing models (ELECTRA, DeBERTaV3 and XLNet) on the shuffled dataset.

Results

Contrary to our expectations, all three models demonstrated significantly reduced performance on the shuffled datasets compared to their performance on the original dataset configuration. Table 4 shows the mean and standard deviation of the F1 score, accuracy, precision and recall for each model across multiple runs, differentiating between training on the shuffled and original datasets.

As illustrated in Table 4, the average F1 scores for these models were approximately 0.7, indicating a strong decline in predictive efficacy.

Model	F1 Score	Accuracy	Precision	Recall
Scores on development-test set of models trained on shuffled training set				
$\text{XLNet}_{\text{shuffled}}$	0.694 ± 0.016	0.862 ± 0.002	0.714 ± 0.015	0.676 ± 0.018
$\text{ELECTRA}_{\text{shuffled}}$	0.702 ± 0.028	0.859 ± 0.014	0.689 ± 0.017	0.716 ± 0.032
$\text{DeBERTaV3}_{\text{shuffled}}$	0.698 ± 0.005	0.859 ± 0.003	0.693 ± 0.004	0.703 ± 0.007
Scores on development-test set of models trained on original training set				
XLNet	0.821 ± 0.026	0.892 ± 0.016	0.930 ± 0.026	0.732 ± 0.026
ELECTRA	0.832 ± 0.017	0.899 ± 0.011	0.951 ± 0.034	0.741 ± 0.025
DeBERTaV3	0.849 ± 0.030	0.909 ± 0.016	0.965 ± 0.009	0.759 ± 0.052

Table 4: Comparing performance metrics of fine-tuned models trained on shuffled and original training sets

Discussion

This outcome suggests that the assumed 'ground truth' quality of the development set does not automatically translate into enhanced training efficacy. The expectation was that shuffling the dataset would make the splits more similar, allowing the model to train on a consistent dataset and perform better in evaluations. However, after running the experiment three times, it became clear that this methodology did not improve model performance. It also seems unlikely that the cause would be an 'unlucky' sampling distribution given the multiple runs. The shuffling also affected the development and development-test sets, potentially reducing the reliability of these sets as indicators of model performance. Validation and test sets with mixed reliability could have led to misleading performance metrics during the training process, ultimately resulting in suboptimal model tuning and evaluation.

4.3.2 Data Augmentation with GPT

Data augmentation is a widely adopted method for generating additional training data, which is particularly advantageous when applying deep learning models [38]. This technique involves artificially expanding the dataset by creating modified versions of existing data or generating entirely new data points. In the context of text data, this can involve altering or generating text sequences to provide a richer and more diverse training set for ML models [39].

Inspired by methodologies described in research conducted by members of the Check That Lab [20], we incorporated data augmentation into our approach. For the generation of new samples, we relied on GPT-3.5 Turbo, a model from OpenAI's Generative Pre-trained Transformer series. GPT models have demonstrated exceptional capabilities in generating coherent and contextually relevant text. [40]. Additionally, they offer an optimal balance between performance and cost-effectiveness, which was a crucial factor given the volume of data we intended to generate.

The process was executed as follows: We established a connection to GPT-3.5 Turbo via the OpenAI API, which allowed for full automation of the data generation process, avoiding the need for manual input through a chat interface. Initially, we provided the model with prompts containing five samples from our training dataset. The prompt utilized is shown in Listing 1. Each prompt instructed GPT-3.5

Turbo to autonomously generate ten additional, content-wise novel samples, along with determining the classification label ('Yes' or 'No'). Furthermore, it was specified that the text samples pertained to political debates and that the class label should be assigned based on whether the general public might consider the content in the text sample to contain a claim potentially involving misinformation, thereby making it 'checkworthy'.

Listing 1: Prompt to generate new samples

```
Given these five samples from political debates:
{samples}
Generate ten additional, context-wise novel samples that pertain to
    political debates. For each sample, also determine the classification
    label 'Yes' or 'No', based on whether the general public might consider
    the content in the text sample to contain a claim potentially involving
    misinformation, making it 'checkworthy'
```

Through this automated, iterative process, we generated 22,501 new samples. These were subsequently integrated with the original training text samples, resulting in a far larger dataset which now contains 45,002 samples.

Results

We tested for a potential enhancement in prediction efficacy by training the ELECTRA Base model on the new dataset consisting of the original and augmented samples. Table 5 shows that this appoach did not improve the accuracy of our model which is reflected in the score. We ran the experiment multiple times and listed the mean and standard deviation of the F1 score, Accuracy, Precision and Recall.

Model	F1 Score	Accuracy	Precision	Recall
Scores on development-test set of ELECTRA model trained on partly augmented training set				
$\text{ELECTRA}_{\text{augm}}$	0.819 ± 0.028	0.893 ± 0.014	0.963 ± 0.017	0.713 ± 0.032
Scores on development-test set of ELECTRA model trained on original training set				
ELECTRA	0.832 ± 0.017	0.899 ± 0.011	0.951 ± 0.034	0.741 ± 0.025

Table 5: Comparing performance metrics of ELECTRA model trained on the partly augmented and original training sets

Discussion

We hypothesize that a potential limitation of our method was the reliance on GPT to assign class labels to the newly generated text samples. Although the prompts provided contextual guidance, we presume that the annotation accuracy and reliability achieved by human coders—as was the case in the development of the Claimbuster dataset—would surpass that of GPT in this specific application. As we did not have access to a comparable team of coders, there was no alternative for generating class labels. Consequently, we decided to discontinue this method and shift our focus to other data augmentation techniques, which are detailed in the subsequent chapters.

4.3.3 Filtering

The creators of the ClaimBuster dataset implemented a method to refine their training dataset by selectively reducing its size. This refinement involved removing some samples that, based on evaluations of label assignment accuracy, had received less reliable class labels compared to others. Specifically, they released a dataset named '3xNCS.json' which consists of samples compiled from the *groundtruth* and *crowdsourced* datasets using stricter criteria for label assignment. This was done to enhance the quality of the training data, which is expected to contribute to the development of more accurate models [41]. The dataset maintains a strict ratio of sentences in the non-checkworthy class to sentences in the checkworthy class, which is reflected in the file name (3x).

According to the authors, this method of filtering the dataset can aid in the training of deep learning models, leading to improved classification scores. This enhancement in model performance through the use of refined training datasets was also demonstrated in the winning paper of the Check That! Lab 2023, Task 1B [23]. Furthermore, it is well-documented in the literature that pre-trained deep learning models can achieve high scores even with smaller training datasets, an insight that has long been recognized in the field of deep learning [42]. Based on these findings, we applied this filtering process to our dataset to examine examine its impact on our models' prediction performance. Initially, only the training set was filtered, and subsequently, both the training and development sets were filtered.

Results

The filtering process resulted in two new datasets: *data_f* and *data_f_all*. In *data_f*, only the training set was filtered, resulting in a new set with 10,259 samples, with 24.6% being checkworthy and 75.4% non-checkworthy. In *data_f_all*, where both the training and development sets were filtered, the new development set contained 797 samples, with 29.7% checkworhty and 70.3% non-checkworthy. The training set remained the same as in *data_f*.

Table 6 displays the performance metrics for each fine-tuned model on the development-test set, using the filtered dataset. We report the mean and standard deviation for the accuracy, precision, recall, and F1 score across several runs for each model, and compare these metrics to the models' performances on the original dataset. We observe that filtering the training set led to varying impacts on the F1 scores across different models. For the RoBERTa model, the F1 score increased by 1.7%. In contrast, for the XLNet and ELECTRA models, the F1 scores dropped by approximately 0.6% and 0.4%, respectively. The DeBERTaV3 model's F1 score remained relatively stable at around 0.849, showing minimal change. The DeBERTaV3 model trained on the *data_f_all* dataset showed a slight increase in the F1 score from 0.849 to 0.852.

Since the filtered datasets are less than half the size of the original datasets, the training time for the models was approximately halved.

Model	F1 Score	Accuracy	Precision	Recall
Scores on development-test set of models trained on filtered dataset				
$\text{RoBERTa}_{\text{filtered}}$	0.832 ± 0.010	0.901 ± 0.005	0.981 ± 0.006	0.722 ± 0.019
$\text{XLNet}_{\text{filtered}}$	0.816 ± 0.020	0.890 ± 0.011	0.943 ± 0.015	0.719 ± 0.023
$\text{ELECTRA}_{\text{filtered}}$	0.829 ± 0.011	0.898 ± 0.006	0.968 ± 0.024	0.725 ± 0.019
$\text{DeBERTaV3}_{\text{filtered}}$	0.848 ± 0.037	0.909 ± 0.016	0.972 ± 0.004	0.753 ± 0.050
$\text{DeBERTaV3}_{\text{filtered\_all}}$	0.852 ± 0.011	0.909 ± 0.003	0.953 ± 0.041	0.772 ± 0.046
Scores on development-test set of models trained on original dataset				
RoBERTa	0.818 ± 0.028	0.890 ± 0.009	0.934 ± 0.056	0.732 ± 0.079
XLNet	0.821 ± 0.026	0.892 ± 0.016	0.930 ± 0.026	0.732 ± 0.026
ELECTRA	0.832 ± 0.017	0.899 ± 0.011	0.951 ± 0.034	0.741 ± 0.025
DeBERTaV3	0.849 ± 0.030	0.909 ± 0.016	0.965 ± 0.009	0.759 ± 0.052

Table 6: Comparing performance metrics of fine-tuned models trained on filtered and original datasets. Models marked as "filtered" were trained on *data_f*. Models marked as "filtered_all" were trained on *data_f_all*

Discussion

The small changes observed in the F1 scores across different models are not statistically significant enough to conclude that the filtering process improved or worsened performance. This conclusion is supported by the moderate variability in some models, as indicated by their standard deviations. For example, the RoBERTa model's F1 score has a standard deviation of 0.028, indicating that the scores from different runs generally fall within the range of approximately 0.79 to 0.846. This level of variability implies that the observed changes in F1 scores due to dataset filtering are withing the expected fluctuations from multiple training runs.

Despite the reduced size of the filtered datasets, which were less than half the size of the original datasets, the overall performance of the models remained similar to those trained on the original dataset. This reduction in dataset size led to a significant decrease in training time, approximately halving it. This outcome highlights the importance of dataset quality, as the filtered datasets allowed for faster training without sacrificing model performance significantly.

4.3.4 Data Augmentation through Paraphrasing

In this section we describe another data augmentation method we applied: paraphrasing. Contrary to the augmentation technique with GPT described above where the class labels quality of the synthetically generated samples rely on the capabilities of GPT, the paraphrased sentence keeps the class label of the original sentence. Paraphrasing involves generating new sentences or phrases that have the same or similar meaning as the original text, but with different wording or phrasing. B. Li et al. explain that paraphrasing techniques are subdivided in three levels: word-level, phrase-level, and sentence-level. In word-level and phrase-level paraphrasing only single words or part of the sentence are transformed. While in sentence-level techniques the entire sentence is rewritten, guaranteeing correct syntax and unchanged semantics [43]. In our task it is important to preserve the meaning of the sentences to cor-

rectly classify the class label, therefore we opted for sentence-level paraphrasing. Machine translation paraphrasing and model generation paraphrasing are two widely adopted sentence-level paraphrasing methods [43], which were also used in the 2023 edition of the CheckThat! Lab [17]. Machine translation, specifically back-translation, involves translating the original sentence to another language and then back to the original language, producing a paraphrased version. In model generation Seq2Seq models are used to directly output paraphrases of the input sentences. Machine translation creates augmentations of the data with limited diversity because of the fixed translations of the models, therefore we chose to apply model generation which generates more diverse sentences [43].

For paraphrasing through model generation, we used the paraphraser from Humarin [44], available on HuggingFace[6]. The paraphraser is based on the T5-base model [45] and is trained on the ChatGPT paraphrase dataset [46], a dataset of paraphrases generated with ChatGPT. Firstly, we augmented the training and development sets of the *data_f_all* dataset, paraphrasing each sample three times.

Additionally, we also created a balanced version of the dataset in an effort to mitigate the difference between precision and recall. We took the training and development sets of the *data_f_all* dataset and, for each sentence with a positive label, created three new paraphrased samples until the ratio between negative and positive samples was balanced.

Our initial results indicated that models could perform better on datasets with similar class ratios as the training set. The phenomenon, known as prior probability shift, occurs when the distribution of the target variable changes between the training and test sets while the conditional distribution of the input features given the target variable remains the same. Prior probability shift often negatively impacts classifier performance [47]. To address this, we created a third paraphrased version of the dataset, ensuring that the training and development sets had the same class ratio as the development-test set.

Results

The paraphrasing with the Humarin paraphraser resulted in three new datasets: *data_f_all_par*, *data_f_all_par_bal*, and *data_f_all_par_sr*. The performance metrics of the models fine-tuned on these datasets are presented in Table 7 and described in detail below.

The dataset *data_f_all_par* contains 41,036 samples in the training set and 3,188 samples in the development set, maintaining the same class label ratio as *data_f_all*. The model $\text{DeBERTaV3}_{\text{paraphrased}}$, trained on this dataset, performed worse than $\text{DeBERTaV3}_{\text{filtered\_all}}$, with an average F1 score of 0.838 on the development-test set. The F1 score on the development set also showed overall worse performance with significant fluctuations compared to $\text{DeBERTaV3}_{\text{filtered\_all}}$.

The second dataset, *data_f_all_par_bal*, has 15,307 samples in the training set and 1,119 samples in the development set. The model $\text{DeBERTaV3}_{\text{paraphrased\_b}}$, trained on this dataset, did not outperform $\text{DeBERTaV3}_{\text{filtered\_all}}$ on the test set, with an average F1 score of 0.838. However, the F1 score on the development set improved by 2.18%. Both $\text{DeBERTaV3}_{\text{paraphrased}}$ and $\text{DeBERTaV3}_{\text{paraphrased\_b}}$ exhibited mixed results on different runs, as indicated by their high standard deviation. $\text{DeBERTaV3}_{\text{paraphrased}}$ had good runs with an F1 score of 0.872 and poor runs with 0.811. Similarly, $\text{DeBERTaV3}_{\text{paraphrased\_b}}$ had

[6] `https://huggingface.co/humarin/chatgpt_paraphraser_on_T5_base`.

excellent runs with an F1 score of 0.886 and poor runs with 0.794.

The third dataset, *data_f_all_par_sr*, has 11,720 training samples and 849 development samples. The model DeBERTaV3$_{\text{paraphrased_sr}}$, trained on this dataset, improved the F1 score by 1.88% to 0.867.

Model	F1 Score	Accuracy	Precision	Recall
Scores on development set				
DeBERTaV3$_{\text{filtered_all}}$	0.962 ± 0.005	0.977 ± 0.003	0.955 ± 0.001	0.970 ± 0.001
DeBERTaV3$_{\text{paraphrased}}$	0.940 ± 0.011	0.964 ± 0.012	0.944 ± 0.014	0.936 ± 0.004
DeBERTaV3$_{\text{paraphrased_b}}$	0.983 ± 0.003	0.983 ± 0.003	0.984 ± 0.005	0.981 ± 0.001
DeBERTaV3$_{\text{paraphrased_sr}}$	0.968 ± 0.001	0.978 ± 0.001	0.967 ± 0.006	0.967 ± 0.006
Scores on development-test set				
DeBERTaV3$_{\text{filtered_all}}$	0.852 ± 0.011	0.909 ± 0.003	0.953 ± 0.041	0.772 ± 0.046
DeBERTaV3$_{\text{paraphrased}}$	0.838 ± 0.031	0.902 ± 0.018	0.946 ± 0.028	0.753 ± 0.037
DeBERTaV3$_{\text{paraphrased_b}}$	0.838 ± 0.046	0.903 ± 0.024	0.957 ± 0.006	0.747 ± 0.074
DeBERTaV3$_{\text{paraphrased_sr}}$	0.867 ± 0.016	0.918 ± 0.008	0.958 ± 0.015	0.792 ± 0.032

Table 7: Performance metrics on development set and on development-test set of models trained on paraphrased versions of the dataset

Discussion

Here, we interpret the results and provide possible explanations for the observed outcomes.

The model DeBERTaV3$_{\text{paraphrased}}$, trained on the dataset *data_f_all_par*, did not achieve an increased average F1 score despite the significant increase in training data through the paraphrasing process. This outcome can be attributed to several factors. Firstly, while paraphrasing increased the quantity of training data, the quality of the paraphrased sentences may not have been sufficient to provide meaningful new information to the model. The paraphrased sentences, though different in wording, might not have introduced substantial new patterns. Secondly, The mixed performance, with both poor and good runs, suggests that the model might have been overfitting to noise introduced by paraphrasing.

The model DeBERTaV3$_{\text{paraphrased_b}}$, trained on the balanced dataset *data_f_all_par_bal*, also did not mitigate the difference between precision and recall as expected. While balancing the classes aimed to address the difference in class distribution, simply adding more positive samples through paraphrasing did not necessarily improve the model's understanding of the minority class. The paraphrased sentences may have not provided examples of the positive class with different meanings from the original sentence to improve recall. Additionally, the balancing of only the training and development-test sets could have aggravated the issue of prior probability shift, as the class ratio difference between the training/development sets and the development-test set increased.

The improvement observed in the DeBERTaV3$_{\text{paraphrased_sr}}$ model can be explained by the mitigation of prior probability shift achieved by aligning the class ratio of the training and development sets with that of the development-test set. This alignment could have ensured that the model's training conditions were more representative of the test conditions. Moreover, the *data_f_all_par_sr* dataset contained fewer

paraphrased sentences compared to the other two paraphrased datasets, which may have resulted in less noise being introduced into the dataset.

It is important to note that the class ratio of test samples is often unknown, as is the case for the dataset used by the CheckThat! Lab to evaluate the final submission. Therefore, aligning class ratios in training and test sets is not a practical technique for real-world applications. Instead, it is more beneficial to develop a model that generalizes well across different datasets.

Additionally, it would be interesting to evaluate a model trained on paraphrased sentences using a test set of paraphrased sentences to determine if paraphrasing can help classify the same claims when formulated differently. This could be valuable for real-world scenarios where claims might be presented in various phrasings.

4.3.5 Quantile-Based Text Length Segmentation

In our previous analysis of the datasets, a potential correlation was identified between the length of text samples and their classification labels ('Yes' and 'No'). Preliminary data suggested that samples labeled 'Yes' often consisted of longer text sequences, which is illustrated by Figure 15. Based on these insights, we hypothesized that integrating text length into our training strategy could potentially enhance the performance of our predictive models.

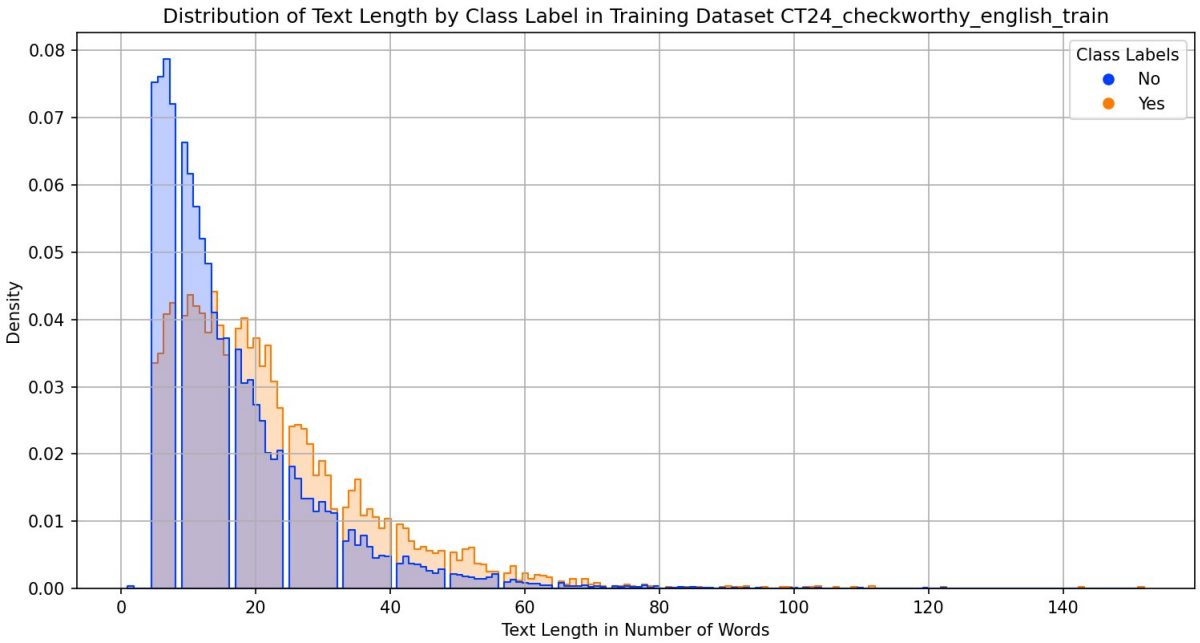

Figure 15: Distribution of text length in the training dataset by class label

To systematically incorporate text length into our model training, we adopted a method of classifying samples into three distinct categories based on their length. Each text sample was appended with a specific label indicating its length category. An example of a sample with its length category appended

is presented in Listing 2. This categorization was implemented by defining quantiles based on the distribution of text lengths within the training data. The shortest third of samples received the label 'sss', the next shortest third 'mmm', and the longest third 'lll'.

Listing 2: Sample with appended length category

Sentence_id	Text	class_label
14089	And we're in real trouble on that. sss	No

These quantiles were determined using the training dataset but were then consistently applied to the development and development-test datasets using the same threshold values. This approach ensured uniformity in how text lengths were categorized across all datasets.

We applied this quantile-based segmentation to both the original and filtered datasets, subsequently training the Microsoft DeBERTa v3 base model with them. This step was undertaken to evaluate whether the explicit consideration of text length could yield improvements in model performance, as measured by the F1 score.

Results

Table 8 presents the performance metrics of models trained on the length-segmented versions of both the original and filtered datasets. The model DeBERTaV3$_{\text{lll_q}}$, trained on the length-segmented version of the original dataset, shows an improvement of 2.71% in the F1 score compared to the DeBERTaV3 model trained on the original dataset. This improvement is particularly noticeable in the recall metric. Conversely, the DeBERTaV3$_{\text{filtered_all_lll_q}}$ model, trained on the length-segmented version of *data_f_all*, did not show the same enhancement in performance. Its F1 score is 0.842, which is lower than the 0.852 achieved by the counterpart model trained on the same dataset without length-segmentation.

Model	F1 Score	Accuracy	Precision	Recall
Scores on development-test set of models trained on length-segmented datasets				
DeBERTaV3$_{\text{lll_q}}$	0.872 ± 0.018	0.920 ± 0.011	0.956 ± 0.016	0.801 ± 0.020
DeBERTaV3$_{\text{filtered_all_lll_q}}$	0.842 ± 0.018	0.903 ± 0.009	0.943 ± 0.013	0.762 ± 0.036
Scores on development-test set of models trained on datasets without length-segmentation				
DeBERTaV3	0.849 ± 0.030	0.909 ± 0.016	0.965 ± 0.009	0.759 ± 0.052
DeBERTaV3$_{\text{filtered_all}}$	0.852 ± 0.011	0.909 ± 0.003	0.953 ± 0.041	0.772 ± 0.046

Table 8: Performance metrics on development-test set of models trained on length-segmented datasets compared with models trained on corresponding datasets without the length-segmentation.

Discussion

The improvement in the recall of the DeBERTaV3$_{\text{lll_q}}$ model indicates that the model has become more adept at identifying true positive check-worthy sentences. This could be attributed to the fact that, in the original dataset, there was a noticeable correlation between text length and class labels, with longer texts more likely to be labeled as 'Yes'. By segmenting the texts based on their length, the model could have gained a clearer understanding of this pattern, thus enhancing its ability to correctly identify check-worthy sentences that might have been missed previously.

Conversely, the same text-length segmentation approach did not yield a similar improvement when applied to the filtered dataset, as evidenced by the DeBERTaV3$_{\text{filtered_all_lll_q}}$ model's performance. One possible explanation for this is the difference in the distribution of sentence lengths between the classes in the filtered dataset compared to the original dataset.

4.4 Ensemble

To further improve the performance of our models we employed an ensemble methodology. An ensemble is a ML technique that combines multiple individual models, which solve the same task, to improve predictive performance and generalization. The fundamental idea is that the errors made by one model can be compensated by the others. Therefore, for an ensemble to be effective, it is crucial that the individual models make different and uncorrelated predictions, while still remaining consistent with the training data [48], [49]. In the literature [49]–[51], various diversity metrics are used to measure the diversity between individual models in an ensemble. These metrics are typically categorized into pairwise and non-pairwise measures. In this work, we focused on pairwise measures, which calculate the diversity between pairs of models. Then to determine the ensemble's overall diversity the average across all possible pairs is calculated. The pairwise measures utilized in this work are as follows:

- **Disagreement measure**: This metric calculates the fraction of samples on which two models disagree in their predictions. Given two classifiers i and j, the disagreement measure D is computed as:
$$D = \frac{N_{\tilde{i}j} + N_{i\tilde{j}}}{N} \tag{1}$$
where N denotes the total number of samples, $N_{\tilde{i}j}$ is the number of samples where model i is correct and model j is incorrect, and $N_{i\tilde{j}}$ the number of samples where model i is incorrect and model j is correct.

- **Double-fault measure**: This metric quantifies the proportion of instances that both models i and j misclassify. This measure is related to the ensemble accuracy, as lower values of the measure, corresponding to higher diversity, tend to favor higher ensemble accuracy. This is because when individual classifiers err on distinct instances, their errors can be mitigated by majority voting, leading to improved overall ensemble performance. The double-fault measure DF is calculated as:
$$DF = \frac{N_{\tilde{i}\tilde{j}}}{N} \tag{2}$$
where $N_{\tilde{i}\tilde{j}}$ is the number of samples in which both classifiers i and j are wrong.

It is noteworthy that previous studies [51]–[53] have highlighted that there is no definite diversity measure for classifier ensembles that fully explains the concept of diversity.

In the literature [49], [53], [54], various strategies to induce diversity among the individual models of an ensemble are described. These include:

- **Data manipulation**: Modifying the training data provided to each model.

- **Problem decomposition**: Splitting the original task into sub-tasks, each handled by a different model. For example, in a multi-class classification problem, each classifier could focus on predicting a different subset of labels.

- **Parameter manipulation**: Varying the training parameters for each model, such as learning rate, weights initialization, or batch size.

- **Structural diversity**: Using different model architectures for each model.

For our ensemble experiments, we utilized the models trained so far. This ensured diversity through data manipulation and architecture diversity of the PLMs.

To combine the predictions of the individual models majority voting is often used in classification task. Majority voting predicts the class with the most votes across the individual models. A soft voting approach, which averages the probability outputs of the individual models, can also be used. This approach allows the ensemble to incorporate the confidence levels of the individual model predictions [49], [55]. However, soft voting requires well-calibrated probabilities, which was not the case for our models. Efforts to calibrate the output probabilities using Platt scaling and isotonic regression did not yield optimal results. Therefore, we opted for majority voting.

To select the size of the ensemble (of how many individual models the ensemble consists) Sagi et al. [56] mention how different factors may influence the decision of the size of the ensemble, such as the computational cost of training many individual models, the inference time for predicting new samples in real-time systems, and the complexity of interpreting ensemble outputs. Sine we already trained the models, and the system was not intended for real-time deployment, we decided to not limit the size of our ensemble, provided they showed the best performance.

For our ensemble experiments, we selected the best-performing runs for each PLM and each dataset, provided that the run achieved a minimum F1 score of 0.82. Subsequently, we generated all possible combinations of model groups ranging from 3 models up to n models, ignoring combinations with an even number of models to avoid ties in majority voting. For each combination, we computed the mean pairwise diversity measures (disagreement and double-fault). We selected the ensembles with the highest diversity, according to the mean pairwise metric, for testing. For comparison we also show the ensembles with the lowest diversity. Additionally, for the ensemble combining all the selected models, we counted how many samples could correctly be classified by at least one member. We termed this metric the 'potential' of the ensemble.

Results

The following outlines the selected best-performing runs and their respective F1 scores on the development-test set:

1. DeBERTaV3 trained on *data_lll_quantiles*: 0.884
2. DeBERTaV3 trained on *data_f_all_paraphrased_sr*: 0.881
3. DeBERTaV3 trained on *data_f_all*: 0.864
4. DeBERTaV3 trained on the original data: 0.880
5. DeBERTaV3 trained on *data_f*: 0.890
6. DeBERTaV3 trained on *data_f_all_lll_quantiles*: 0.859
7. DeBERTaV3 trained on *data_f_all_paraphrased*: 0.872
8. DeBERTaV3 trained on *data_f_all_paraphrased_b*: 0.886
9. RoBERTa trained on *data_f*: 0.842
10. RoBERTa trained on the original data: 0.837
11. ELECTRA trained on the original data: 0.853
12. ELECTRA trained on *data_f*: 0.842
13. XLNet trained on the original data: 0.839
14. XLNet trained on *data_f*: 0.833

The pairwise disagreement between these models is illustrated in the matrix in Figure 16. Notably, RoBERTa and DeBERTaV3_f_all_paraphrased_b demonstrate the highest disagreement, differing on 9.4% of the development-test samples. On the contrary, DeBERTaV3 and DeBERTaV3_lll_quantiles show the lowest disagreement, differing on only 2.8% of predictions. DeBERTaV3 variants generally exhibit moderate disagreement among themselves (upper left 8x8 submatrix), with some pairs showing high disasgreement (above 8%) and others low disagreement (below 3%). When paired with models of different architectures, DeBERTaV3 variants show moderate disagreement without low-disagreement exceptions but with more high-disagreement exceptions.

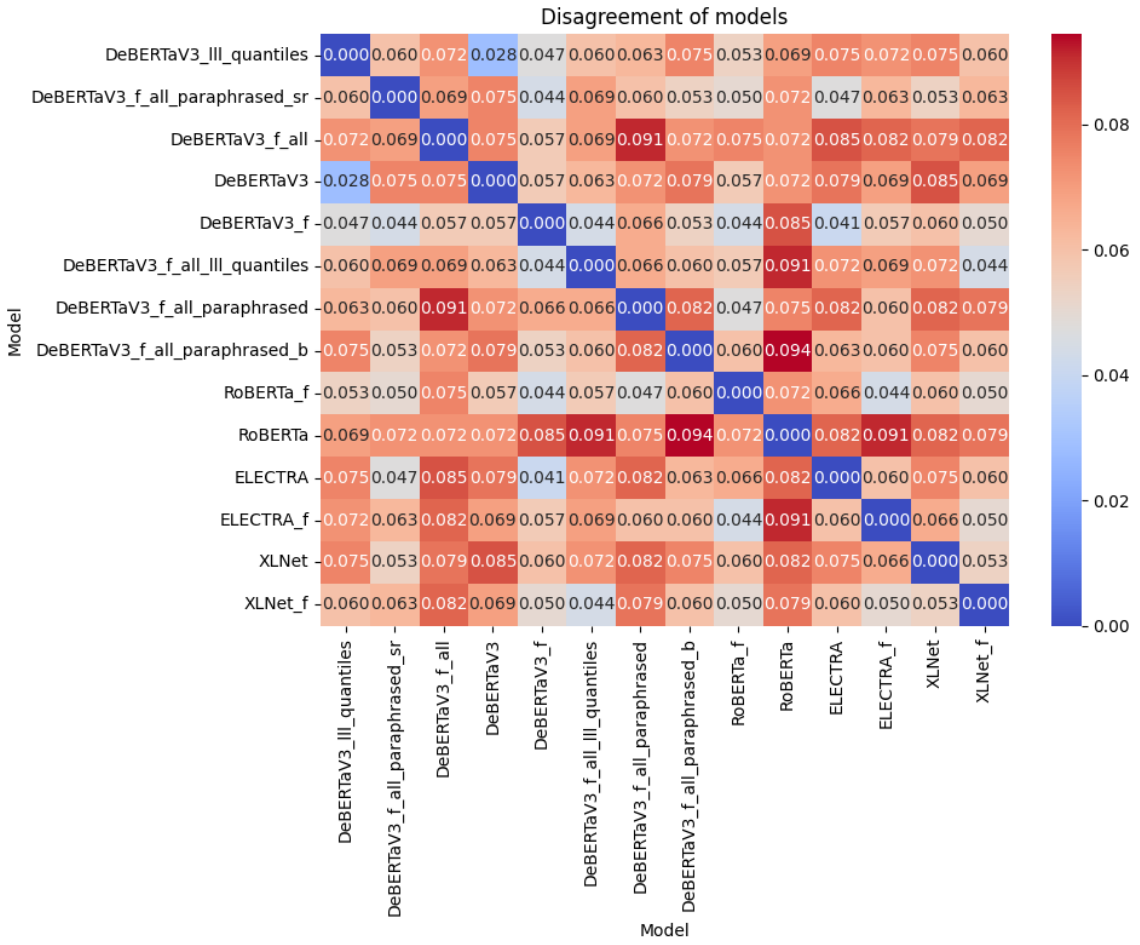

Figure 16: pairwise disagreement measure matrix

Figure 17 displays the pairwise double-fault measure matrix. DeBERTaV3 variants tend to show higher diversity (lower double-fault measure) among themselves. When paired with models of different architectures, they show lower diversity. The pairs among RoBERTa, ELECTRA, and XLNet variants (bottom right 6x6 submatrix) exhibit the lowest diversity.

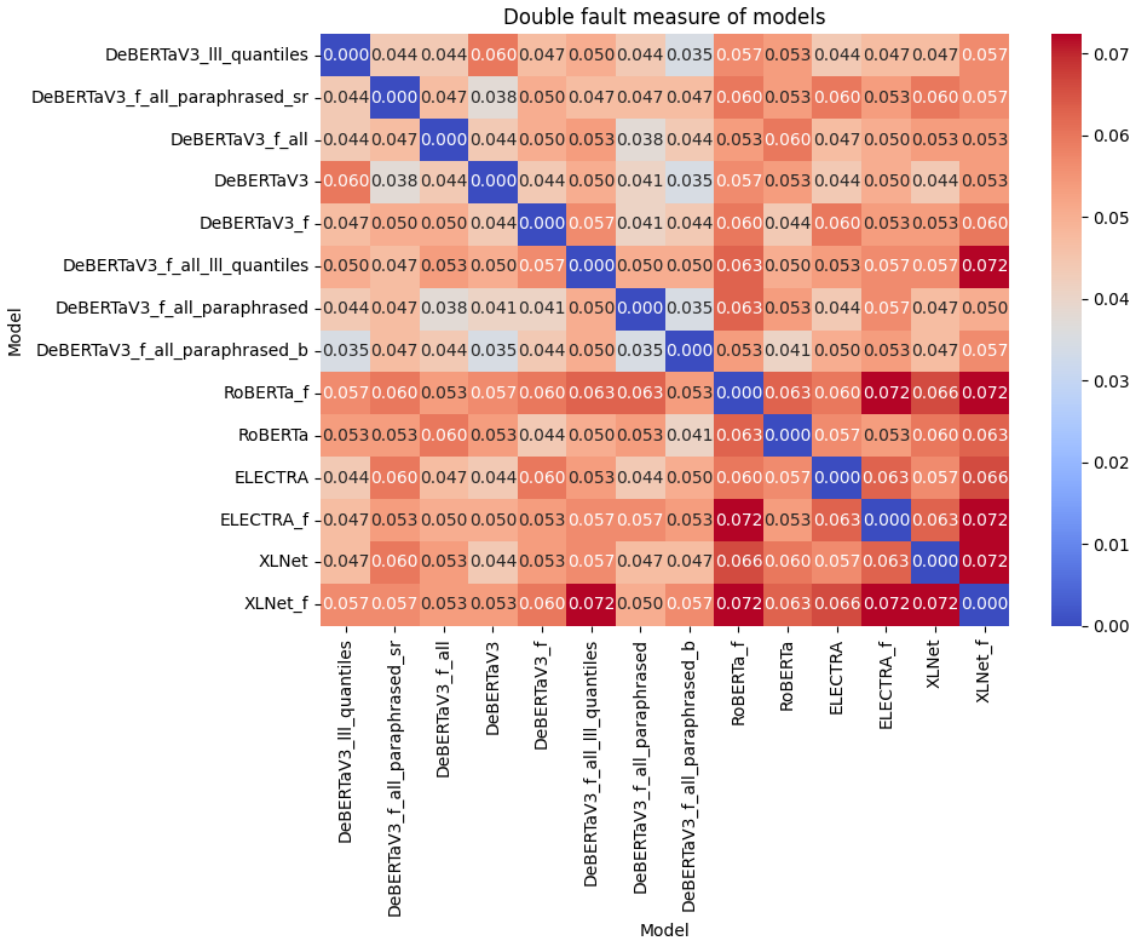

Figure 17: pairwise double-fault measure matrix

Table 9 presents the top ten ensembles with the highest and lowest mean pairwise disagreement. For each ensemble, the table lists the individual models, mean disagreement, F1 score, and improvement compared to the highest F1 score of its individual models. All ensembles consist of three models. Among the ten most diverse ensembles, five showed F1 score improvements, three showed declines, and two had unchanged scores. Contrarily, seven of the ten least diverse ensembles showed F1 score declines, one showed improvement, and two remained unchanged. Both the most and least diverse ensembles had similar average F1 scores: 0.880 (average improvement 0.27%) and 0.874 (average improvement -1.17%), respectively.

Models combination	Mean disagreement	F1 score	Improvement (%)
Top ten ensembles with the highest mean disagreement			
3, 7, 11	0.0860	0.895	2.64
3, 7, 13	0.0839	0.880	0.92
3, 7, 14	0.0839	0.884	1.38
6, 10, 12	0.0839	0.862	0.35
7, 8, 10	0.0839	0.883	-0.34
8, 10, 13	0.0839	0.862	-2.71
3, 7, 8	0.0818	0.884	-0.23
3, 10, 12	0.0818	0.870	0.69
4, 8, 10	0.0818	0.886	0.00
6, 8, 10	0.0818	0.886	0.00
Top ten ensembles with the lowest mean disagreement			
2, 5, 11	0.0440	0.890	0.00
1, 4, 5	0.0440	0.890	0.00
5, 6, 14	0.0461	0.879	-1.24
2, 5, 9	0.0461	0.867	-2.58
1, 4, 9	0.0461	0.884	-0.67
5, 9, 12	0.0482	0.872	-2.02
5, 6, 9	0.0482	0.872	-2.02
1, 5, 9	0.0482	0.878	-1.35
9, 12, 14	0.0482	0.848	0.71
5, 9, 14	0.0482	0.860	-3.37

Table 9: ensembles with the highest and lowest diversity according to the mean pairwise disagreement. The model numbers correspond to the ordering in the initial list presented in the results section

Table 10 lists the top ten ensembles with the highest and lowest diversity according to the mean double-fault measure. All ensembles consist of three models, except for three ensembles that comprise five models. Eight of the ten most diverse ensembles showed improved F1 scores, while two showed declines. Conversely, four of the ten least diverse ensembles showed F1 score declines, while the remaining six improved. The most diverse ensembles had an higher average F1 score of 0.897 with an average improvement of 1.19%, compared to 0.853 with an average improvement of 0.14% for the least diverse ensembles.

Models combination	Mean double-fault measure	F1 Score	Improvement (%)
Top ten ensembles with the lowest mean double-fault measure			
4, 7, 8	0.0367	0.895	1.02
1, 7, 8	0.0377	0.889	0.34
3, 7, 8	0.0388	0.884	-0.23
5, 7, 8	0.0398	0.888	-0.22
2, 4, 8	0.0398	0.891	0.56
4, 5, 8	0.0409	0.906	1.80
3, 4, 8	0.0409	0.906	2.26
3, 4, 7	0.0409	0.895	1.70
1, 3, 8	0.0409	0.906	2.26
3, 4, 5, 7, 8	0.0415	0.911	2.36
Top ten ensembles with the highest mean double-fault measure			
9, 12, 14	0.0723	0.848	0.71
9, 13, 14	0.0702	0.850	0.95
6, 9, 14	0.0692	0.844	-1.74
12, 13, 14	0.0692	0.844	0.24
6, 12, 14	0.0671	0.844	-1.74
6, 13, 14	0.0671	0.856	-0.35
9, 12, 13	0.0671	0.866	1.52
11, 12, 14	0.0671	0.876	2.70
6, 9, 12, 13, 14	0.0667	0.844	-1.74
9, 11, 12, 13, 14	0.0664	0.860	0.82

Table 10: Ensembles with the highest and lowest diversity according to the mean double-fault measure

For the ensemble that combines all 14 models, the potential is 314, with an F1 score of 0.878. The potential of 314 indicates that out of the 318 samples in the development-test set, 4 samples (1.26%) could not be correctly classified by any of the models.

Discussion

As expected, models that differ only in their architecture exhibit lower disagreement compared to those differing in both architecture and training dataset. This trend is evident when examining the pairwise disagreement measure, which reflects how often pairs of models disagree on their predictions. However, this trend does not hold for the double-fault measure. The DeBERTaV3 model variants, which differ only in their training datasets, exhibit the highest diversity among themselves according to the double-fault measure. This can be attributed to the relationship between the double-fault measure and the accuracy of individual models. Since DeBERTaV3 models have higher prediction accuracy, they tend to make fewer incorrect predictions. Consequently, there are fewer instances where both models in a pair make simultaneous errors, resulting in a lower double-fault measure.

The observation that both the most and least diverse ensembles based on the pairwise disagreement measure have similar average F1 scores (0.880 and 0.874, respectively) can be explained by the follow-

ing factors:

- The pairwise disagreement measure indicates how often pairs of models disagree, but this does not necessarily translate to improved ensemble performance. If disagreements occur primarily on easy samples where most models make correct predictions, the overall impact on performance is minimal.

- The disagreement measure does not account for the accuracy of individual models. For instance, models like RoBERTa and XLNet, which do not perform as well individually, can still exhibit high disagreement. Therefore, ensembles comprising such models may not show significant performance improvement despite high disagreement.

In contrast, the F1 scores for diverse ensembles are higher than those for non-diverse ensembles when evaluated using the double-fault measure. This can be explained by the following:

- The double-fault measure focuses on instances where both models in a pair make incorrect predictions. Ensembles with high diversity have models that are less likely to fail on the same samples. When one model makes an error, other models in the ensemble are more likely to correct it, leading to improved performance.

- The double-fault measure, as already explained, is related to the accuracy of the individual models. Diverse ensembles, therefore, tend to comprise models that already perform well individually. This inherent accuracy contributes to higher overall F1 scores.

Even the ensemble combining all 14 models could not achieve the full potential of 318, with four samples remaining misclassified by all models. This limitation suggests that certain samples may be difficult for all models to predict accurately. Investigating the potential on other datasets and analyzing the patterns in misclassified samples could provide further insights on the challenging samples.

5 Conclusion

This thesis aimed to refine and advance misinformation detection methodologies through the utilization of deep learning models. The primary focus was on developing and fine-tuning a deep learning-based model to identify claims that warrant fact-checking, and to participate in the Task 1B of the CheckThat! Lab 2024. The approach involved a comprehensive analysis of the dataset, fine-tuning of models, and experimentation with ensemble techniques and various data manipulation techniques to enhance model performance.

For the CheckThat! Lab 2024 Task 1B, we submitted an ensemble composed of DeBERTaV3, $\text{DeBERTaV3}_{\text{filtered}}$, and ELECTRA, which at the time of submission was the model with the highest F1 score (0.896). Our submissions placed 5th out of 26 teams with an F1 score of 0.771 on the unseen test data. The significant difference between the submission score and the scores obtained on development-test during the building process of the model, indicates room for improving model generalization across different datasets. It would be valuable to analyze the test dataset used to evaluate the submission to identify any patterns in the data that might explain the discrepancy in scores. Additionally, it would be beneficial to evaluate the different models and ensembles on the test data to determine if any model generalizes better to diverse datasets. Unfortunately, this analysis could not be performed due to time constraints and the fact that the labels of the test data were not available until one week after the submission.

Based on the insights gained from this thesis, future work could focus on several areas to further enhance misinformation detection capabilities:

- Future research could continue to refine the ensemble methodology. Considering that the ClaimBuster dataset classifies sentences with three class labels NFS, CFS, UFS, future work could involve fine-tuning models to predict different subsets of these class labels. This could lead to models with complementary strengths, which could be effectively combined through an ensemble approach.

- Future work could address the difference in average sentence length between the development-test set and the training/development sets. This aspect was not explored in this thesis, and addressing it could improve model performance and generalization.

- A detailed analysis of the test dataset used in the competition could be conducted to identify patterns and characteristics that affect model performance.

6 References

[1] E. Aïmeur, S. Amri, and G. Brassard, "Fake news, disinformation and misinformation in social media: A review," *Social Network Analysis and Mining*, vol. 13, no. 30, 2023.

[2] Z. Guo, M. Schlichtkrull, and A. Vlachos, "A survey on automated fact-checking," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 178–206, 2022.

[3] L. Konstantinovskiy, O. Price, M. Babakar, and A. Zubiaga, "Toward automated factchecking: Developing an annotation schema and benchmark for consistent automated claim detection," *Digital Threats: Research and Practice*, vol. 2, no. 14, pp. 1–16, 2022. DOI: https://doi.org/10.1145/3412869.

[4] F. Arslan, N. Hassan, C. Li, and M. Tremayne, "A benchmark dataset of check-worthy factual claims," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 14, pp. 821–829, 2020. DOI: https://doi.org/10.1609/icwsm.v14i1.7346.

[5] P. Christen, D. J. Hand, and N. Kirielle, "A review of the f-measure: Its history, properties, criticism, and alternatives," *ACM Comput. Surv.*, vol. 56, no. 3, Oct. 2023, ISSN: 0360-0300. DOI: 10.1145/3606367. [Online]. Available: https://doi.org/10.1145/3606367.

[6] A. Bondielli and F. Marcelloni, "A survey on fake news and rumour detection techniques," *Information Sciences*, vol. 497, pp. 38–55, 2019, ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2019.05.035. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025519304372.

[7] L. Wu, F. Morstatter, K. M. Carley, and H. Liu, "Misinformation in social media: Definition, manipulation, and detection," *SIGKDD Explor. Newsl.*, vol. 21, no. 2, pp. 80–90, Nov. 2019, ISSN: 1931-0145. DOI: 10.1145/3373464.3373475. [Online]. Available: https://doi.org/10.1145/3373464.3373475.

[8] M. R. Islam, S. Liu, X. Wang, and G. Xu, "Deep learning for misinformation detection on online social networks: A survey and new perspectives.," *Social Network Analysis and Mining*, vol. 10, no. 82, 2020. DOI: https://doi.org/10.1007/s13278-020-00696-x.

[9] M. Fernandez and H. Alani, "Online misinformation: Challenges and future directions," in *Companion Proceedings of the The Web Conference 2018*, ser. WWW '18, Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 595–602, ISBN: 9781450356404. DOI: 10.1145/3184558.3188730. [Online]. Available: https://doi.org/10.1145/3184558.3188730.

[10] A. Peñas, J. Deriu, R. Sharma, G. Valentin, and J. Reyes-Montesinos, "Holistic analysis of organised misinformation activity in social networks," in *Disinformation in Open Online Media*, D. Ceolin, T. Caselli, and M. Tulin, Eds., Cham: Springer Nature Switzerland, 2023, pp. 132–143, ISBN: 978-3-031-47896-3.

[11] R. A. Shubha Mishra Piyush Shukla, "Analyzing machine learning enabled fake news detection techniques for diversified datasets," *Wireless Communications and Mobile Computing*, 2022. DOI: 10.1155/2022/1575365.

[12] H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," *Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, I. Traore, I. Woungang, and A. Awad, Eds., pp. 127–138, 2017.

[13] S. Gilda, "Notice of violation of ieee publication principles: Evaluating machine learning algorithms for fake news detection," pp. 110–115, 2017. DOI: 10.1109/SCORED.2017.8305411.

[14] N. K. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: Methods for finding fake news," *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1–4, 2015. DOI: https://doi.org/10.1002/pra2.2015.145052010082. [Online]. Available: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/pra2.2015.145052010082.

[15] J. Y. Khan, M. T. I. Khondaker, S. Afroz, G. Uddin, and A. Iqbal, "A benchmark study of machine learning models for online fake news detection," *Machine Learning with Applications*, vol. 4, p. 100 032, 2021, ISSN: 2666-8270. DOI: https://doi.org/10.1016/j.mlwa.2021.100032. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S266682702100013X.

[16] A. Das, H. Liu, V. Kovatchev, and M. Lease, "The state of human-centered nlp technology for fact-checking," *Information Processing & Management*, vol. 60, no. 2, p. 103 219, 2023, ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2022.103219. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030645732200320X.

[17] F. and Alam, A. Barrón-Cedeño, G. S. Cheema, *et al.*, "Overview of the CLEF-2023 CheckThat! lab task 1 on check-worthiness in multimodal and multigenre content," in *Working Notes of CLEF 2023—Conference and Labs of the Evaluation Forum*, ser. CLEF '2023, Thessaloniki, Greece, 2023.

[18] T. Pavlov and G. Mirceva, "Covid-19 fake news detection by using bert and roberta models," pp. 312–316, 2022. DOI: 10.23919/MIPRO55190.2022.9803414.

[19] H. T. Sadouk, F. Sebbak, and H. E. Zekiri, "Es-vrai at checkthat! 2023: Analyzing checkworthiness in multimodal and multigenre contents through fusion and sampling approaches," *CLEF 2023 Working Notes*, vol. 3497, pp. 430–444, 2023.

[20] A. Modzelewski, W. Sosnowski, and A. Wierzbicki, "Dshacker at checkthat! 2023: Check-worthiness in multigenre and multilingual content with gpt-3.5 data augmentation," *CLEF 2023 Working Notes*, vol. 3497, pp. 383–393, 2023.

[21] P. von Däniken, J. Deriu, and M. Cieliebak, "Zhaw-cai at checkthat! 2023: Ensembling using kernel averaging," *CLEF 2023 Working Notes*, vol. 3497, pp. 534–545, 2023.

[22] J. R. Martinez-Rico, L. Araujo, and J. Martinez-Romo, "Nlpir-uned at checkthat! 2023: Ensemble of classifiers for check-worthiness estimation," *CLEF 2023 Working Notes*, vol. 3497, pp. 372–382, 2023.

[23] M. Sawiński, K. Węcel, E. Księżniak, *et al.*, "Openfact at checkthat! 2023: Head-to-head gpt vs. bert-acomparative study of transformers language models for the detection of check-worthy claims," *CLEF 2023 Working Notes*, vol. 3497, pp. 453–472, 2023.

[24] K. Rudnicka, "Variation of sentence length across time and genre: Influence on the syntactic usage in english," *Studies in Corpus Linguistics*, Nov. 2018. DOI: 10.1075/scl.85.10rud.

[25] "Tf–idf," in *Encyclopedia of Machine Learning*, C. Sammut and G. I. Webb, Eds. Boston, MA: Springer US, 2010, pp. 986–987, ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_832. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_832.

[26] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: A comprehensive review," *ACM computing surveys (CSUR)*, vol. 54, no. 3, pp. 1–40, 2021.

[27] P. He, J. Gao, and W. Chen, "Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing," *arXiv preprint arXiv:2111.09543*, 2021.

[28] A. Williams, N. Nangia, and S. Bowman, "A broad-coverage challenge corpus for sentence understanding through inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, M. Walker, H. Ji, and A. Stent, Eds., New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1112–1122. DOI: 10.18653/v1/N18-1101. [Online]. Available: https://aclanthology.org/N18-1101.

[29] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for SQuAD," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, I. Gurevych and Y. Miyao, Eds., Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 784–789. DOI: 10.18653/v1/P18-2124. [Online]. Available: https://aclanthology.org/P18-2124.

[30] Y. Liu, M. Ott, N. Goyal, *et al.*, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[31] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[32] K. Clark, T. Luong, Q. V. Le, and C. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," in *ICLR*, 2020. [Online]. Available: https://openreview.net/pdf?id=r1xMH1BtvB.

[33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. arXiv: 1810.04805v2.

[34] Z. Dai, *Zihangdai/xlnet*, original-date: 2019-06-19T08:16:46Z, May 23, 2024. [Online]. Available: https://github.com/zihangdai/xlnet (visited on 05/26/2024).

[35] *Google-research/electra*, original-date: 2020-03-10T03:42:50Z, May 25, 2024. [Online]. Available: https://github.com/google-research/electra (visited on 05/26/2024).

[36] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7.

[37] K. Boyd, K. H. Eng, and C. D. Page, "Area under the precision-recall curve: Point estimates and confidence intervals," in *Machine Learning and Knowledge Discovery in Databases*, H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 451–466, ISBN: 978-3-642-40994-3.

[38] C. Shorten, T. M. Khoshgoftaar, and B. Furht, "Text data augmentation for deep learning," *Journal of big Data*, vol. 8, no. 1, p. 101, 2021.

[39] M. Abulaish and A. K. Sah, "A text data augmentation approach for improving the performance of cnn," in *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*, IEEE, 2019, pp. 625–630.

[40] T. B. Brown, B. Mann, N. Ryder, *et al.*, "Language models are few-shot learners," 2020. arXiv: 2005.14165 [cs.CL].

[41] K. Meng, D. Jimenez, F. Arslan, J. D. Devasier, D. Obembe, and C. Li, "Gradient-based adversarial training on transformer networks for detecting check-worthy factual claims," 2020. arXiv: 2002.07725 [cs.CL].

[42] J. Kaplan, S. McCandlish, T. Henighan, *et al.*, "Scaling laws for neural language models," 2020. arXiv: 2001.08361 [cs.LG].

[43] B. Li, Y. Hou, and W. Che, "Data augmentation approaches in natural language processing: A survey," *AI Open*, vol. 3, pp. 71–90, 2022, ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2022.03.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651022000080.

[44] M. K. Vladimir Vorobev, "A paraphrasing model based on chatgpt paraphrases," 2023. [Online]. Available: https://huggingface.co/humarin/chatgpt_paraphraser_on_T5_base.

[45] C. Raffel, N. Shazeer, A. Roberts, *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html.

[46] M. K. Vladimir Vorobev, "Chatgpt paraphrases dataset," 2023.

[47] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012, ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2011.06.019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320311002901.

[48] D. Gopika and B. Azhagusundari, "An analysis on ensemble methods in classification tasks," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 3, no. 7, 2014.

[49] L. Rokach, "Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography," *Computational Statistics & Data Analysis*, vol. 53, no. 12, pp. 4046–4072, 2009, ISSN: 0167-9473. DOI: https://doi.org/10.1016/j.csda.2009.07.017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167947309002631.

[50] L. Heidemann, A. Schwaiger, and K. Roscher, "Measuring ensemble diversity and its effects on model robustness," 2021. DOI: 10.24406/publica-fhg-411897. [Online]. Available: https://publica.fraunhofer.de/handle/publica/411897.

[51] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181–207, May 1, 2003, ISSN: 1573-0565. DOI: 10.1023/A:1022859003006. [Online]. Available: https://doi.org/10.1023/A:1022859003006.

[52] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: A survey and categorisation," *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005, ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2004.04.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253504000375.

[53] H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Comput. Surv.*, vol. 50, no. 2, Mar. 2017, ISSN: 0360-0300. DOI: 10.1145/3054925. [Online]. Available: https://doi.org/10.1145/3054925.

[54] Y. Ren, L. Zhang, and P. Suganthan, "Ensemble classification and regression-recent developments, applications and future directions [review article]," *IEEE Computational Intelligence Magazine © 2011 IEEE*, vol. 11, no. 1, pp. 41–53, 2016. DOI: 10.1109/MCI.2015.2471235.

[55] Z.-H. Zhou, "Ensemble learning," in *Encyclopedia of Biometrics*, S. Z. Li and A. Jain, Eds., Boston, MA: Springer US, 2009, pp. 270–273, ISBN: 978-0-387-73003-5. DOI: 10.1007/978-0-387-73003-5_293. [Online]. Available: https://doi.org/10.1007/978-0-387-73003-5_293.

[56] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 4, e1249, 2018. DOI: https://doi.org/10.1002/widm.1249. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1249. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249.

7 List of Figures

8 List of Tables

9 Glossary

AP Average Precision 18

CFS Check-worthy Factual Statement 1, 7, 9, 39

FFNN Feedforward Neural Network 5

LIWC Linguistic Inquiry and Word Count 5

ML Machine Learning 1, 4, 22, 31

MLM Masked Language Modeling 15, 16

NFS Non-factual Statement 1, 7, 9, 39

NLP Natural Language Processing 1, 4

NLU natural language understanding 15, 19

NSP Next Sentence Prediction 15

OSN Online Social Network 1

PLM Pre-trained Language Model 4, 5, 15, 17, 32, 46

PR Precision-Recall 18, 19, 45

RTD Replaced Token Detection 16, 19

SVM Support Vector Machine 4, 13

TF-IDF Term Frequency-Inverse Document Frequency 5, 12, 13

UFS Unimportant Factual Statement 1, 7, 9, 39

10 Appendix

10.1 Official Assignment

Titel

Misinformation Detection

Beschreibung

Diese Bachelorarbeit zielt darauf ab, ein fortschrittliches Tool zu entwickeln, das auf Deep Learning-Techniken basiert, um Desinformationen im Internet effektiv zu erkennen und zu analysieren. Im Zentrum der Arbeit steht die Erweiterung und Verfeinerung eines bestehenden Deep Learning-Modells, um es speziell für die Herausforderungen und Dynamiken des Erkennens von falschen Informationen im Web anzupassen.

Forschungsziele:

- Untersuchung der aktuellen Landschaft der Desinformation im Internet und Identifizierung spezifischer Herausforderungen und Muster.
- Analyse bestehender Deep Learning-Modelle und -Methoden, die für die Erkennung von Desinformationen relevant sind.
- Entwicklung und Anpassung eines Deep Learning-basierten Tools, das speziell auf die Erkennung von Desinformationen im Web ausgerichtet ist.
- Durchführung von Tests und Evaluationen, um die Wirksamkeit und Genauigkeit des Tools in verschiedenen Szenarien zu bewerten.

Ziel ist die Teilnahme an dem CheckThat-Lab 2024.

Voraussetzungen

- Programmierkenntnisse in Python
- Erste Erfahrungen im Deep Learning von Vorteil
- Hohe Motivation