Propaganda Detection with Machine Learning Approaches

Candidate number: 111111

MSc Data Science, University of Sussex, Brighton, UK

Abstract

Propaganda has existed and masked in the news throughout human history, raising concern about the truthfulness of information for readers. However, detecting propaganda has never been an easy task, especially with the massive increase in articles and news in this digital age. This article investigates automatic propaganda detection using machine learning techniques. We explore two approaches: Bag-of-Words (BoW) and pre-trained language models (PLMs). Our experiments demonstrate that PLMs, specifically a fine-tuned BERT model, significantly outperform BoW approaches in both tasks. The BERT model achieves high accuracy (95%) in propaganda presence detection and surpasses other models in technique classification. These results highlight the potential of PLMs for automatic propaganda detection.

1 Introduction

Propaganda, the use of information to shape opinions or promote a particular cause, has been hidden in daily news and spread widely with the advancement of the Internet. The task of detecting propaganda is difficult for inexperienced users since propaganda aims to convey biased or misleading information without being noticed (Martino et al. 2020). Moreover, in today's digital age, the development of artificial intelligence (AI) with text-generating models (Huang et al. 2023) even raises further concerns about the prevalence of fake news and its difficulty in detection. Thus, the need for automatic detection of propaganda has been emphasised, and researchers have exploited a variety of facets of this topic, from document-level or text-level approaches to using different techniques to construct the detection model.

In this article, we explore a propaganda detection problem with a provided dataset containing sentences and their propaganda labels. As our objective is to detect propaganda automatically, we aim to build a model that can deal with two key tasks:

- Task 1 - Propaganda Presence Detection: Can we determine whether a given sentence contains propaganda or not?
- Task 2 - Propaganda Techniques Classification: If a sentence is known as propaganda, can we identify the specific technique used to convey it?

With the capability of learning from data to capture patterns, machine learning approaches show great potential to address these questions by creating models that generalise the signals of propaganda and, therefore, can automatically detect them. To investigate the application of (ML) techniques, we create empirical experiments which implement two different approaches:

- Bag of words classifier: We present sentences as bags of word frequencies. These BOWs can then be treated as features in various machine-learning classifiers. We explore the performances of this technique with common classification algorithms.
- Pre-trained Language Models: We leverage the power of pre-trained large language models by using a fine-tuned BERT model for the specific classification task to apply to the propaganda detection problem.

By comparing and analysing the performance of these approaches, we have an overview of the ML efficiency in propaganda detection and how the state-of-the-art ML technique performs compared to traditional approaches. Our experiments in this article also aim to identify the most effective methods for automatic propaganda detection.

2 Related works

Since automatic propaganda detection has become an increasingly crucial research area, various approaches have been explored. The techniques have evolved significantly, transitioning from coarse methods analysing entire texts to more fine-grained approaches predicting specific locations of manipulative techniques.

The predominant approach relies on supervised learning, where annotated datasets serve as training data for ML classifiers. These datasets contain representative examples of truthful and suspicious content. The classifiers learn to identify patterns through features extracted from the text data, identifying the texts' characteristics. These features can include the presence of specific words or n-grams, Part-of-Speech (POS tagging) and Name-Entity (NE), lexical features such as special characters or punctuation, and lexicons (LIWC, for instance) to capture persuasive and biased language.

One of the early studies by Rashkin et al. 2017 demonstrated the potential of this approach. They constructed a corpus of labelled news articles and utilised lexicons to extract words associated with trusted and fake news. Using a Max-Entropy classifier, they achieved an F1 score of 65%, showcasing the promise of ML for propaganda detection.

Extending this foundation, Barrón-Cedeño et al. 2019 introduced a system called *Proppy*. *Proppy* employed a wider range of features, incorporating n-grams, lexicons, stylistic cues, vocabulary richness, readability scores, and content-based features. This comprehensive approach yielded a significant improvement, achieving an F1 score of 96.72%, exceeding Rashkin's results by over 8.51%.

Further studies, like Reis et al. 2019, explored the effectiveness of various ML classifiers using the BuzzFeed dataset, including articles related to the 2016 US election. These studies extracted a vast array of features (141 features) from news content, sources, and the surrounding environment. The results conclude that XGBoost and random forests provide the best prediction for fake news, suggesting an auxiliary tool for fact-checkers.

Recent research has shifted focus towards a more fine-grained analysis. Da San Martino et al. 2019 proposed a model for not only detecting propaganda but also identifying the specific techniques used within the text. This subtask of technique classification was further explored by Martino et al. 2020. Top performers in this challenge leveraged fine-tuned BERT models and other pre-trained language models (PLMs). The combined F1 score reached 63.63 using a majority voting approach across the top 20 systems, which is way better than the baseline logistic regression classifier using only the length of the fragment.

3 Methodology

3.1 Dataset

The training and evaluation of propaganda classification models rely on a well-defined labeled sentence dataset. In this study, the provided dataset includes two separate files for the training and testing stages, respectively. The data contains sentences with span identification tags and the corresponding propaganda technique labels used in the sentences. Here are example instances from the given dataset:

Table 1: Example instances from dataset.

label	**sentence**
not_propaganda	This declassification effort <BOS> won't make things any worse than they are for President Trump. <EOS>
flag_waving	"The Obama administration misled the <BOS> American people <EOS> and Congress because they were desperate to get a deal with Iran," said Sen.

We use 2560 instances in the training set to create models to predict the label of 640 sentences in the testing set. There are 9 available values of labels representing texts with *not_propaganda* and 8 different propaganda techniques, including:

- flag_waving
- appeal_to_fear_prejudice
- causal_simplification
- doubt
- exaggeration,minimisation
- loaded_language
- name_calling,labeling
- repetition

These labels are a subset of propaganda techniques identified in the Propaganda Techniques Corpus (Martino et al. 2020), with the distribution of labels illustrated below.

Table 2: Distribution of labels in the training set.

label	**# of sentences**	**% of sentences**
not_propaganda	1269	49.57%
appeal_to_fear_prejudice	157	6.13%
causal_oversimplification	165	6.45%
doubt	157	6.13%
exaggeration,minimisation	170	6.64%
flag_waving	155	6.05%
loaded_language	161	6.29%
name_calling,labeling	166	6.48%

In the training set, the proportion of texts with no propaganda is approximately half of the dataset, which is ideally appropriate for the task of detecting propaganda presence. For task 2, each propaganda technique shares a similar number of instances, and therefore, we do not need to perform further data cleaning since there is no imbalance problem between classes.

Another noteworthy point is sentences that contain propaganda tend to have longer texts. On the other hand, while there isn't a significant difference in the average text lengths between propaganda techniques, Figure 2, 3 shows that the spans of some classes, such as *loaded_language*, *name_calling,labeling*, and *repetition*, are shorter than others. This reflects the fact that these techniques usually need to use shorter words/phrases to emphasise or repeat a piece of information, and the difference in the span lengths might contribute to the ability technique classification. We compare our models' predictions using sentences and spans to investigate the improvement in performance.

3.2 Approaches

3.2.1 Bag-of-Words (BoW)

BoW is a fundamental method for text representation used in linguistic statistic models or other language models. It represents sentences or text documents as a collection of word frequencies, usually in a dictionary format. Each document is transformed into a feature vector where each element corresponds to a word or a combination of words (n-grams) in the vocabulary, and its value represents the frequency of that word/n-grams within the document. The pros of this approach are its simplicity and interpretability, where BoW offers a straightforward method for computers to consume text characteristics as input for further tasks such as text classification or text generation. BoW also benefits computational efficiency, making it suitable for real-time applications (for instance, Proppy uses tf.idf-weighted n-grams, a metric derived from BoW, as a primary feature - Barrón-Cedeño et al. 2019). However, BoW ignores the word order and grammatical structure, leading to potentially missing semantic signals and complex relationships between words.

In the context of propaganda detection, we can assume that the presence of specific words or combinations of words (n-grams) might indicate a certain propaganda technique. For example, frequent occurrences of words like "enemy," "threat," or "danger" might suggest the appeal_to_fear_prejudice technique. Therefore, we use BoW as the feature with different common ML classifiers such as Logistic Regression, Naïve Bayes, Random Forest, K-Nearest-Neighbors, and Support Vector Machines.

3.2.2 Pre-trained language models

Pre-trained Language Models (PLMs) such as BERT and GPT have revolutionized NLP tasks by capturing rich semantic representations of text. These models are trained on massive amounts of text data and learn contextual relationships between words. For instance, BERT utilizes a bidirectional architecture, allowing it to consider both the preceding and following words in a sentence when generating word embeddings. The multiple attention heads in BERT's encoder also allow the model to focus on different types of relationships and dependencies, leading to richer token representations.

Fine-tuning a pre-trained model involves adapting it to a specific task by adjusting the final layers of the model. This process will involve freezing the pre-trained weights of the majority of the PLMs layers, which have already captured general linguistic knowledge. We then add a layer on top of the PLMs to get the desired output for specific tasks, such as Name-Entity-Recognition (NER), POS Tagging, sequence classification, or machine translation as a sequence generation task. While PLMs have been widely used recently because of their ability to represent the semantic facet of sequences with word meaning and information about word position successfully, this approach still has some downsides. One of the disadvantages is that PLMs heavily rely on the large training corpora, which can lead to expensive computational costs and potential biases inherited from the pre-training data.

In this study, we will leverage a pre-trained BERT model for the text classification task named BertForSequenceClassification. The model design includes a linear layer on top of the base BERT embedding model (Huggingface documentaion 2024). This architecture leverages the advantages of contextualized word embedding from the pre-trained BERT model, which can be crucial in identifying propaganda techniques.

3.3 Experiments

We define two tasks discussed in the introduction of this article, including propaganda presence detection, which asks if a sentence contains propaganda or not, and propaganda technique detection, which asks for what propaganda technique was used.

Both tasks are treated as a sequence classification problem, with task 1 being a binary classification question, whilst task 2 requires multi-class detection. For the two mentioned approaches, we implement the classifiers for the task 1 objective and then modify and leverage them for the technique classification goal.

3.3.1 Baselines and Evaluation Measure

As the observation of differences in the lengths of texts and spans of sentences between classes, we establish the baseline performances for our propaganda detection tasks by implementing classification models using these features. The baseline for propaganda presence detection is a logistic regression classifier with text lengths. On the other hand, task 2 will use a similar baseline system using span lengths as mentioned in the TC task of Martino et al. 2020.

In general, we primarily rely on accuracy and F1 scores to evaluate the performance of our models. Accuracy indicates the prediction ability of the model for all classes, while F1 is a harmonic mean of precision and recall, and a high F1 score often indicates a model that excels at both correctly identifying propaganda (high precision) and minimizing false negatives (high recall). However, in a particular case, it is more important to get a better recall than precision, and vice versa. For instance, a model might have high precision, meaning it rarely classifies non-propaganda sentences as propaganda but might have lower recall, indicating it misses some actual propaganda content. If this is

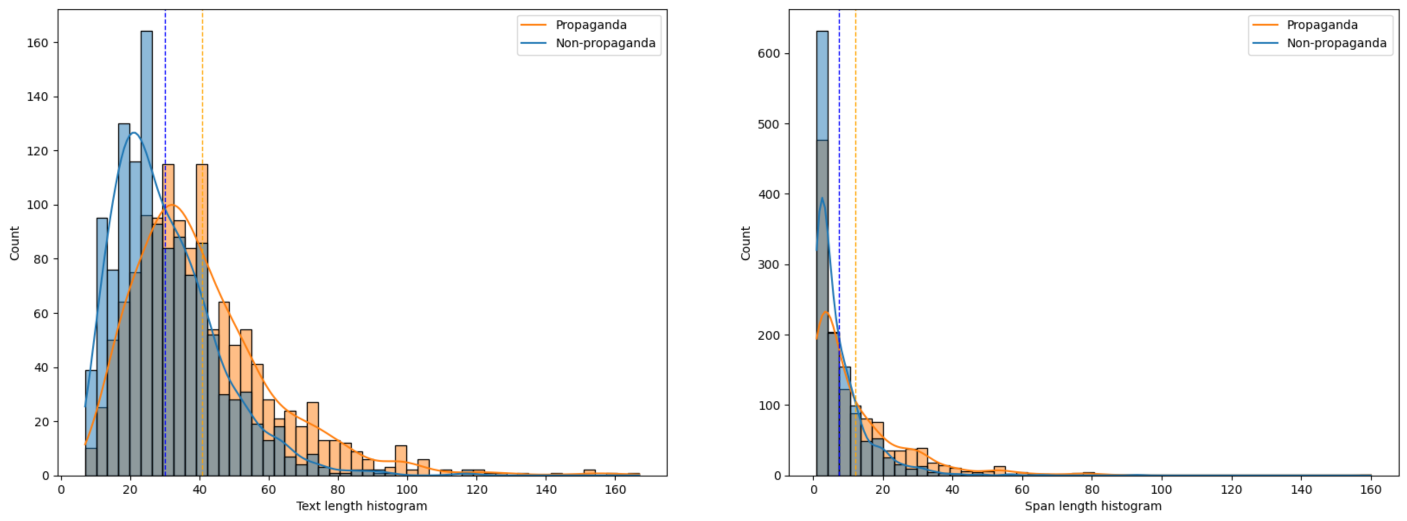

Figure 1: Histograms of text lengths and span lengths between non-propaganda and propaganda texts.

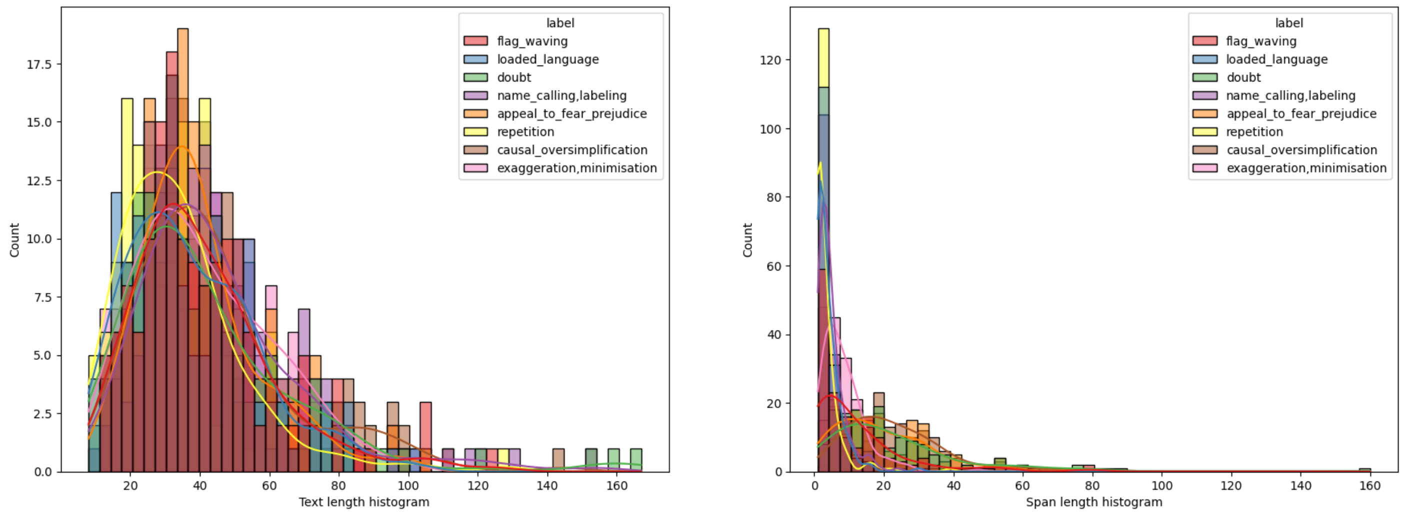

Figure 2: Histograms of text lengths and span lengths between propaganda techniques.

an auxiliary model supporting fact-checkers, it should cover all potential propaganda content for further verification.

3.3.2 Bag-of-Words Approaches

The Bag-of-Words method involves building a vocabulary of all unique elements (words/characters n-grams) from the training data and converting instances into feature vectors where values are the frequencies of corresponding elements. Different feature definitions give different vectors representing higher or lower levels of information extracted for the texts and, therefore, might affect the performance of the classifiers. For instance, compared with a unigram Bag-of-Words, a trigrams model or a 5-grams model increases vocabulary size massively and captures deeper relationships between words, such as collocations. N-grams models created from words and characters will also behave differently since characters n-grams might help in reducing the sparsity and leverage the morphological relationships.

To construct the vocabulary and perform the vectorization process, we adopt the CountVertorizer class from the scikit-learn library. The vectors are then used as features with five different classification algorithms, including Logistic Regression, Multinomial Naïve Bayes, Random Forest, K-Nearest-Neighbors, and Support Vector Machines.

In the empirical test, we try different ways of vectorizing instances by setting up 2 parameters for the CountVectorizer. These parameters are *ngram_range*, which is the n-grams used when building the vocabulary, and *analyzer*, which is how n-grams are constructed (the values 'word', 'char', 'char_wb' represent word-based, character-based, and character-based inside word boundaries approaches, respectively). We also tune the hyperparameters for each classifier to get the best results for each classification algorithm.

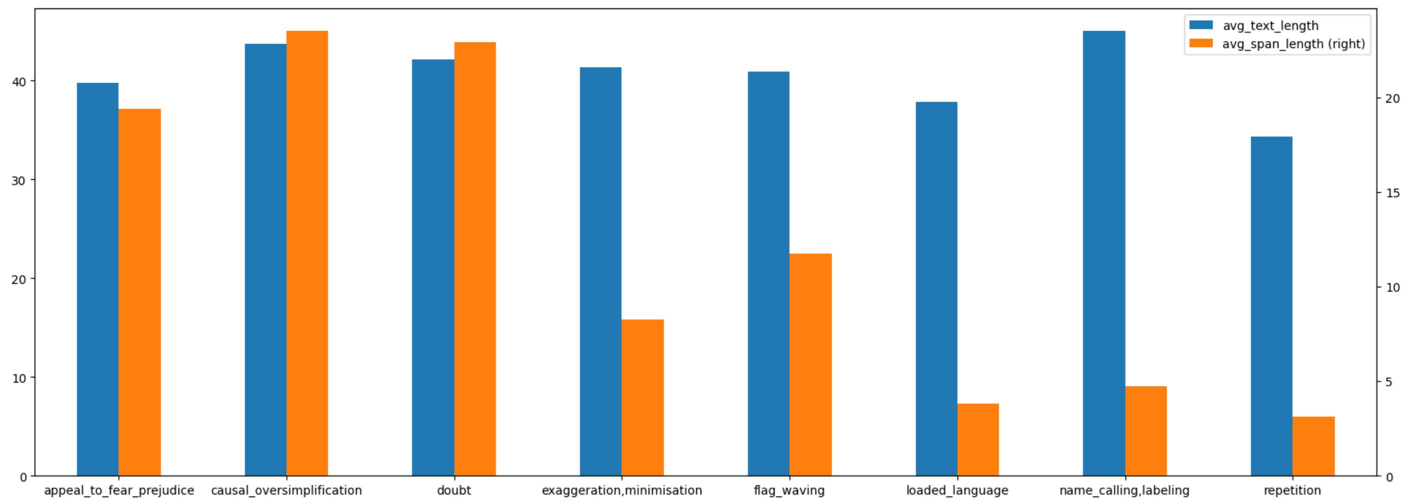

Figure 3: Comparison of the average text lengths and average span lengths between propaganda techniques.

3.3.3 Pre-trained Language Models - BertForSequenceClassification

The BertForSequenceClassification model is a fine-tuned version of BERT, which includes a pre-trained BERT model and a linear layer on the top that consumes the output from the BERT model and acts as the classifier. Overall, this system inputs tokenized sentences as vectors, passes them into the pre-trained BERT model for the embeddings, and finally computes the classification of the sentence embeddings at the top layer (Figure 4). While the output of the base BERT model includes embedding for each token, the model only pools out the embedding of the [CLS] token, which is the representation of the whole text, for the next step.

In this design, we use the pre-trained BERT for tokenizing and embedding the sentences, so only the weights of sentence embeddings used in the classification layer are updated with the backpropagation method after each batch prediction. Therefore, in the experiment, we investigate the model's performance with adjustments in the number of running epochs, learning rate, and batch size, which directly affect how the weights are updated. The values for these hyperparameters are determined empirically, and the proposed model is set to train for 10 epochs in Task 1 and 20 epochs in Task 2. To handle the overfitting loop when training the data, we implement the early stopping strategy, which stops the training when the validation loss no longer improves. The advantage of this fine-tuned model is that we leverage the rich contextualized representation of text generated by the pre-trained BERT model to feed into the classification layer. To showcase the superiority of BERT embeddings, we adopt a simple Neural Network (NN) with a very similar overall design and compare the results. The NN model includes two layers:

- Embedding layer: take vectorized sentences as input and return the sentence embedding. Instead of using the BertTokenizer for sentence tokenizing and the pre-trained BERT model for embedding, we use two different tools. First, we use the torchtext library to tokenize the texts and build the vocabulary for the vectorization process. Secondly, the vectorized sentences will be converted to embeddings by using the EmbaddingBag class from Pytorch. The EmbeddingBag computes the sum (or mean or max) of the word embeddings to get the corresponding sentence embedding for each instance. The implementations of torchtext and EmbeddingBag are from Pytorch's tutorial (Pytorch documentaion 2024) This is the main difference between our NN model and our fine-tuned BERT model.
- Linear layer: consumes the output of the Embedding layer and performs classification. This layer shares identical functionalities with the top layer of BertForSequenceClassification model.

4 Results

4.1 Propaganda presence detection

With the binary classification problem, all the developed models have promising results with the task of propaganda presence detection. Even the baseline model with only information on text lengths shows improvement compared with guessing all instances are propaganda. Table 3 shows the performance for the baseline, each classification algorithm with BoW method, and for our NN and pre-trained BertForSequenceClassification models on the propaganda presence detection task.

For Bag-of-Words approaches, interestingly, all classifiers work better with longer n-grams (4-grams or 5-grams) constructed from characters. The result indicates that logistic regression is the best model in overal (with 0.87 accuracy and 0.87 F1 score) for this method, and with simple text representations from BoW as features, the classifier outper-

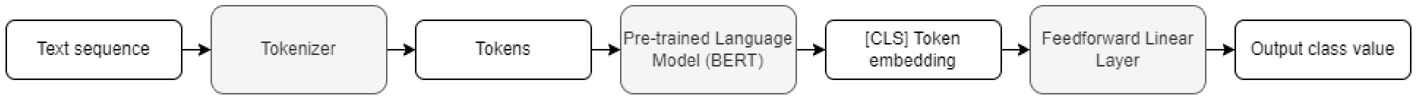

Figure 4: Overall flow of the BertForSequenceClassification model.

Table 3: An overview of the best-performing methods for the Propaganda Presence Detection task.

Models	**Accuracy**	**macro F1-Score**	**Recall for propaganda**
Baseline	0.60	0.59	0.49
MultinomialNB	0.84	0.84	0.87
LogisticRegression	0.87	0.87	0.79
RandomForest	0.86	0.86	0.80
KNN	0.72	0.70	0.48
SVC	0.86	0.86	0.82
Fine-tuned BERT	**0.95**	**0.95**	**0.95**
NN model	0.70	0.69	0.59

forms the baseline that uses only the text lengths, which only has 0.60 accuracy. In terms of the recall for the propaganda class, the Multinomial Naïve Bayes model has the best result when it covers 87% of propaganda sequences, while the KNeighborsClassifier tends to assign instances as non-propaganda and only get under 50% of true propaganda texts.

The pre-trained BertForSequenceClassification model dominates this task and shows incredibly high scores when it can accurately predict 95% of the testing instances. The precision and recall of both classes outperform that of all other models. In contradiction, the result from the NN model is far worse and only better than the baseline's performance. This demonstrates how better sentence representations using [CLS] tokens from the BERT model instead of generating sentence embeddings by a simple sum or mean operation.

4.2 Propaganda technique classification

This task is much more challenging than the previous problem, and even human experts get low agreement on span and label annotation. This might be due to the larger number of classes and the overlapping word-using between different techniques. The baseline system using span lengths can only accurately label 23% of propaganda texts, with 0 over 39 instances having loaded language being detected.

Table 4 shows the performances of SVC and Fine-tuned BERT models when they use features extracted from the whole texts with detailed precision and recall for each technique. All BoW approaches only achieve under 50% overall accuracy and show the worst predictions in the loaded_language class. Our fine-tuned BERT model witnesses a significant improvement when reaching 0.6 for both accuracy and F1 score.

Focusing on the span, we observe a slight improvement in the models' performances when using vectorized spans as inputs. Table 5 showcases the results of this approach.

In general, the pre-trained language model is the most efficient method for the technique classification task. However, the performance is still far from perfect, leaving room for further optimisation and implementation of other techniques

5 Conclusion and Future Improvements

This study explored automatic propaganda detection using two different approaches: Bag-of-Words (BoW) and pre-trained language models (PLMs). We investigated the effectiveness of these techniques in two key tasks: propaganda presence detection and propaganda technique classification.

Our experiments demonstrated that both methods can be effective in the propaganda detection problem. However, the fine-tuned BERT model significantly outperforms BoW approaches in both tasks, achieving superior accuracy and F1 scores in propaganda presence detection and showcasing its capability to capture characteristics of propaganda techniques by surpassing all other models in this classification task.

These results highlight the potential of PLMs for automatic propaganda detection. However, there are still areas for improvement, particularly in propaganda technique classification. For instance, the current approach relies on sentence embeddings generated using the [CLS] token from BERT, which might not fully capture the intricate semantic relationships within sentences. Exploring alternative sentence embedding methods like Sentence-BERT (Reimers and Gurevych 2019) or BERT-flow (Li et al. 2020) could lead to better representations for propaganda detection tasks. Another direction for future work

Table 4: Performances of each approach's best model for the Propaganda Technique Classification task.

	Baseline		SVC		Fine-tuned BERT	
Techniques	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
appeal_to_fear_prejudice	0.20	0.07	0.49	0.49	0.57	0.74
causal_oversimplification	0.26	0.69	0.41	0.54	0.66	0.66
doubt	0.00	0.00	0.50	0.47	0.69	0.77
exaggeration,minimisation	0.14	0.23	0.44	0.50	0.44	0.40
flag_waving	0.11	0.04	0.71	0.67	0.70	0.71
loaded_language	0.00	0.00	0.42	0.36	0.55	0.54
name_calling,labeling	0.21	0.21	0.48	0.41	0.68	0.62
repetition	0.27	0.68	0.36	0.38	0.50	0.35

Table 5: An overview of the best-performing methods for the Propaganda Technique Classification task.

	Texts as features		Spans as features	
Models	**Accuracy**	**macro F1-Score**	**Accuracy**	**macro F1-Score**
Baseline			0.23	0.16
MultinomialNB	0.41	0.38	0.45	0.43
LogisticRegression	0.46	0.46	0.53	0.52
RandomForest	0.43	0.42	0.52	0.51
KNN	0.25	0.23	0.38	0.36
SVC	0.48	0.47	0.49	0.49
Fine-tuned BERT	**0.61**	**0.59**	**0.67**	**0.67**

is focusing on a more fine-grained method. While using features extracted from the entire sentence yielded promising results, we have seen an enhancement in classifying propaganda techniques by looking into specific spans within the text. With these improvements, we can develop a robust and accurate method for propaganda detection, supporting readers in acquiring authentic and truthful information.

References

Barrón-Cedeño, Alberto et al. (July 2019). "Proppy: A System to Unmask Propaganda in Online News". en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01. Number: 01, pp. 9847–9848. ISSN: 2374-3468. DOI: 10.1609/aaai.v33i01.33019847.

Da San Martino, Giovanni et al. (Nov. 2019). "Fine-Grained Analysis of Propaganda in News Article". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, pp. 5636–5646. DOI: 10.18653/v1/D19-1565.

Huang, Kung-Hsiang et al. (July 2023). "Faking Fake News for Real Fake News Detection: Propaganda-Loaded Training Data Generation". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, pp. 14571–14589. DOI: 10.18653/v1/2023.acl-long.815.

Huggingface documentaion (2024). *BertForSequenceClassification.* URL: https://huggingface.co/docs/transformers/v4.40.2/en/model_doc/bert#transformers.BertForSequenceClassification (visited on 05/12/2024).

Li, Bohan et al. (2020). "On the Sentence Embeddings from Pre-trained Language Models". en. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 9119–9130. DOI: 10.18653/v1/2020.emnlp-main.733.

Martino, G. Da San et al. (Sept. 2020). *SemEval-2020 Task 11: Detection of Propaganda Techniques in News Articles.* arXiv:2009.02696 [cs]. DOI: 10.48550/arXiv.2009.02696.

Pytorch documentaion (2024). *Text classification with the torchtext library — PyTorch Tutorials 2.3.0+cu121 documentation.* URL: https://pytorch.org/tutorials/beginner/text_sentiment_ngrams_tutorial.html (visited on 05/12/2024).

Rashkin, Hannah et al. (Sept. 2017). "Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Process-*

ing. Ed. by Martha Palmer, Rebecca Hwa, and Sebastian Riedel. Copenhagen, Denmark: Association for Computational Linguistics, pp. 2931–2937. DOI: `10.18653/v1/D17-1317`.

Reimers, Nils and Iryna Gurevych (Aug. 2019). *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. en.

Reis, Julio C. S. et al. (Mar. 2019). "Supervised Learning for Fake News Detection". In: *IEEE Intelligent Systems* 34.2. Conference Name: IEEE Intelligent Systems, pp. 76–81. ISSN: 1941-1294. DOI: `10.1109/MIS.2019.2899143`.