2024

# Propaganda Detection

ADVANCED NATURAL LANGUAGE PROCESSING (968G5)

CANDIDATE NO: 277238

## 1. INTRODUCTION

The detection and categorisation of propaganda, as well as propaganda techniques across various platforms have garnered considerable attention in recent times due to the rising prevalence of misinformation and manipulation in the media sphere. Propaganda employs a variety of tactics to sway public opinion, advocate biased perspectives and deceive audiences (Martino et al., 2020). To address this issue, Da San Martino et al. (2020) presented a standardised dataset for the purpose of detecting propaganda techniques in news articles as part of the SemEval 2020 Task 11.

This task encompassed two main objectives; firstly, distinguishing between propaganda and non-propaganda content through binary classification; and secondly, providing detailed classification of specific propaganda techniques. Developing effective models for these purposes is essential in countering the dissemination of propaganda and fostering media literacy. This analysis delves into two different methodologies for each objective, assesses their efficacy and delves into the insights gleaned from the conducted experiments.

## 2. RELATED WORK

The detection of propaganda has become increasingly popular in the field of natural language processing (NLP). Various methods have been utilized, from advanced deep learning models to more traditional machine learning techniques. The BERT model known as Bidirectional Encoder Representations from Transformers stands out as a notable advancement in this area. BERT's ability to grasp context from both ends of a sentence has proven highly effective in a range of NLP tasks, particularly text classification.

Alongside deep learning models like BERT, Convolutional Neural Networks (CNNs) have also gained traction for text categorization. CNNs excel at identifying local patterns and extracting key information from sentences, making them a favoured choice for such tasks. Additionally, pretrained word embeddings like GloVe (Global Vectors for Word Representation) have enhanced the efficiency of CNN models by providing meaningful word representations.

In 2019, Barrón Cedeno and team introduced Proppy, a system aimed at detecting propaganda in online news using an analysis of stylistic and linguistic features.

Moreover, researchers have leveraged networks incorporating Long-Short-Term Memory (LSTM) units for their ability to capture long term relationships and patterns within text. LSTMs prove valuable when examining propaganda strategies that span multiple sentences due to their proficiency in identifying such intricate connections.

The SemEval 2020 Task 11 enabled researchers to explore different methods for identifying propaganda. Organizers of the event shared a collection of news articles that were segmented and marked with propaganda tactics. This dataset has supported the creation and assessment of various models, such as specific CNN and BERT based approaches, which have shown encouraging outcomes in identifying propaganda tactics.

## 3. DATA EXPLORATION AND PREPROCESSING

The project's dataset includes a training set (propaganda_train.tsv) and a test set (propaganda_val.tsv). Each file comprises two columns; "label" and "tagged_in_context." The "label" column specifies the propaganda technique or indicates "not_propaganda" if there is no propaganda, while the "tagged_in_context" column presents the text snippet with "<BOS>" and "<EOS>" tokens marking the beginning and end of the propaganda within the context.

To understand the dataset better, we start by examining the sizes of the training and test sets. The training set contains 2,414 examples, with 580 instances in the test set. This information helps us gauge our data volume and ensure we have enough samples for training and assessment.

Next, we investigate how labels are distributed in the training set. By looking at each label's count, we notice an imbalance where "not_propaganda" appears most frequently at 1,191 occurrences. The other propaganda techniques vary in frequency from 144 instances for "doubt" to 164 instances for "exaggeration,minimisation."

Understanding how the labels are distributed is key to dealing with class imbalances and making sure our models can effectively handle the varying frequencies of different propaganda techniques.

To better visualize the distribution of labels, we calculate the percentage of each label within the training set. The "not_propaganda" category makes up 49.34% of the instances, while the various propaganda techniques make up the remaining 50.66%. This data underscores the importance of implementing strategies to address the skewed nature of the dataset, such as using class weighting or oversampling methods, to prevent biased predictions towards the more prevalent class.

Additionally, we investigate whether there are any missing or inconsistent data points present. Detecting and managing missing values or abnormalities is crucial for maintaining data integrity and avoiding potential issues during model training and assessment.

Through comprehensive exploration of the data, we gather valuable insights into the dataset's characteristics that guide our pre-processing choices, model design strategies and approaches to handling class imbalances.

## 4. TASK 1: PROPAGANDA VS NOT PROPAGANDA CLASSIFICATION

The first task focuses on classifying sentences as either containing propaganda or not. Two approaches were investigated: fine-tuned BERT and CNN with GloVe embeddings.

### 4.1. Approach 1: Fine -tuned BERT

The first approach utilises BERT (Devlin et al., 2018), a powerful pre-trained language model that has been implemented on various NLP tasks and produced remarkable results. The model is fine-tuned on the propaganda dataset to gain its knowledge and be able to detect as needed.

#### 4.1.1. Pre-processing

The initial steps for this method involve breaking down the input text using the BERT tokenizer found in the Transformers library. This tool changes the text into a series of tokens that work well with the BERT model. These sequences are then made to be a fixed length of 64 tokens by adding or removing padding as needed. The labels are transformed into binary values, where 1 stands for propaganda and 0 signifies non propaganda.

#### 4.1.2. Model Architecture

In this method, we use the BertForSequenceClassification model for the BERT architecture. This model includes a linear classification layer placed above the pre trained BERT model. It is configured to have 2 output classes for the binary classification task. The model is adjusted further using the AdamW optimizer with a learning rate of 2e-5 and a batch size of 32. The training process lasts for 3 epochs, with validation taking place after each epoch to check the model's performance on unseen data.

#### 4.1.3. Training

Throughout the process of training, we keep track of the model's progress using a tqdm progress bar that gives us real time updates on training loss and validation accuracy. Validation accuracy is determined by comparing the model's predicted values and the actual values. Metrics like recall, accuracy, precision, and F1-score are used to assess the model's performance.

#### 4.1.4. Validation

The fine-tuned BERT method attains an accuracy of 93.37% during validation after undergoing 3 training cycles. The precision, recall and F1 score for the propaganda category stand at 0.93, 0.94 and 0.93 correspondingly, showcasing a robust capability in identifying propaganda content.

To further assess the model's effectiveness, it undergoes testing on a distinct test dataset. The model achieves a test accuracy of 91.03%. The precision, recall and F1 score for the propaganda category on this test dataset are recorded at 0.88, 0.94 and 0.91 respectively. These outcomes underscore the model's proficiency in generalizing to new data sets and accurately categorizing propaganda text.

### 4.2. Approach 2: CNN with GloVe Embeddings

The second approach for Task 1 utilizes a Convolutional Neural Network (CNN) architecture in combination with pre-trained GloVe word embeddings. CNNs have been widely used for text classification tasks due to their ability to capture local patterns and extract relevant features from text (Chen, 2015).

### 4.2.1. Pre-processing

To prepare the data for this method, the text is first converted to lowercase using a custom preprocess_text function. Next, the text data is tokenized using the Tokenizer tool from the Keras pre-processing library. The tokenizer is trained on the training data to create a vocabulary and assign unique integer indices to each word. The texts_to_sequences function is then used to convert the text data into sequences of integer indices. These sequences are padded to a fixed length of 100 using the pad_sequences function. The labels are encoded as binary values, with "1" denoting propaganda and "0" denoting not_propaganda.

The GloVe word embeddings are loaded and utilized to generate an embedding matrix, where each word in the vocabulary is matched with its corresponding embedding vector. This embedding matrix is employed to initialize the embedding layer of the CNN model.

### 4.2.2. Model Architecture

The CNN model architecture comprises an embedding layer followed by a 1D convolutional layer with 128 filters and a kernel size of 5.

The CNN model employs a convolutional layer followed by a global max pooling layer to capture key features from the convolved data. These extracted features then undergo processing in a dense layer with 64 units and ReLU activation, accompanied by a dropout layer with a rate of 0.2 to prevent overfitting. Subsequently, a dense layer with one unit and sigmoid activation is utilized for binary classification.

### 4.2.3. Training

For optimization, the CNN model is compiled using the binary cross entropy loss function and the Adam optimizer. Training spans 5 epochs with a batch size of 32, monitoring progress via the tqdm progress bar while evaluating performance on a validation set.

### 4.2.4. Validation

Post training, the CNN model attains an 81.16% validation accuracy. Precision, recall and F1 score values for the propaganda class on the validation set are noted as 0.91, 0.69 and 0.79 respectively.

To evaluate generalization capabilities on new data, the model undergoes testing on an independent test set yielding an accuracy of 80.34%. For propaganda text detection specifically, precision stands at 0.92 with recall at 0.64 and F1 score at 0.75—indicating effective identification of propaganda content.

### 4.3. Performance comparison and discussion

| Metric | Fine-tuned BERT | CNN with GloVe Embeddings |
|---|---|---|
| Validation Accuracy | 93.37% | 81.16% |
| Validation Precision | 0.93 | 0.91 |
| Validation Recall* | 0.94 | 0.69 |
| Validation F1-score* | 0.93 | 0.79 |
| Test Accuracy | 91.03% | 80.34% |
| Test Precision* | 0.88 | 0.92 |
| Test Recall* | 0.94 | 0.64 |
| Test F1-score* | 0.91 | 0.75 |

*Table 1: Task 1 Model Performance Comparison*

*Precision, Recall, and F1-score are reported for the propaganda class.

The data in the table clearly indicates that the fine-tuned BERT method performs better than the CNN with GloVe embeddings method across most measures on both the validation and test datasets. The fine-tuned BERT model achieves higher accuracy, recall and F1 score in comparison to the CNN model.

Although the CNN model demonstrates higher precision on the validation dataset, the fine-tuned BERT model maintains a better balance between precision and recall, resulting in a higher F1 score. This suggests that the fine-tuned BERT model is more effective at correctly detecting propaganda text while minimising false positives and false negatives.

The variance in performance can be attributed to how the BERT model is pretrained and its architecture, enabling it to capture detailed contextual information and adapt well to detecting propaganda. While computationally less intensive, the CNN model relies on fixed word embeddings and may not capture propaganda techniques as adeptly as the fine-tuned BERT model.

It's worth noting that choosing between these methods depends on factors like available computational resources, dataset size and task specific requirements.

The fine-tuned BERT model shows great results, but the CNN model remains a good choice when computational speed is crucial.

## 5. TASK 2: PROPAGANDA TECHNIQUE CLASSIFICATION

This section focuses on classifying specific propaganda techniques given a snippet or span of text known to contain propaganda. Two approaches were investigated: fine-tuned BERT with class weighting for imbalanced data and CNN-BiLSTM.

## 5.1. Fine-tuned BERT with Class Weighting for Imbalanced Data

### 5.1.1. Pre-processing

The pre-processing steps involve breaking down the input text into tokens using the BERT tokenizer. These tokens are then modified to fit a fixed length of 128 tokens. The labeling process assigns integer values to different propaganda techniques based on a mapping dictionary.

### 5.1.2. Model Architecture

This approach utilizes the BertModel architecture from the transformers library. A custom classifier named BertForTC is created by adding a linear layer (fc) with 128 units on top of the pooled output from BERT, followed by another linear layer (classifier) with output classes corresponding to the propaganda techniques.

### 5.1.3. Training

Training involves using the AdamW optimizer with a learning rate of 2e-5. To address class imbalances, weights are calculated based on the distribution of propaganda techniques in training data and incorporated into the loss function (CrossEntropyLoss) to prioritize minority classes during training (Cui et al., 2019). The model is trained over 5 epochs with a batch size set at 32.

### 5.1.4. Validation

The BERT method attains a validation accuracy of 65.22% following 5 training epochs. The assessment report displays differing results among various propaganda strategies, with the model excelling in recognizing "not_propaganda" (F1 score of 0.91) and displaying varied performance in other strategies.

To further evaluate the model's efficacy, it undergoes testing on a distinct test dataset, achieving a test accuracy of 65.17%. The assessment report for the test dataset mirrors the trends seen in the validation outcomes, with the model excelling in identifying "not_propaganda" (F1 score of 0.89) and demonstrating varying performance across other propaganda tactics.

## 5.2. Approach 2: Convolutional Neural Networks-Bidirectional Long-Short-Term Memory (CNN BiLSTM)

### 5.2.1. Pre-processing

The initial steps for preparing the CNN BiLSTM approach include breaking down the text data using the Keras Tokenizer. The tokenizer is adjusted to fit with the training data and then the texts are transformed into sequences of integer indices by using the texts_to_sequences method. These sequences are then made uniform by padding them to a fixed length of 100 through the pad_sequences function. The labels are associated with integers manually through a dictionary that allocates a distinct integer value to each propaganda technique.

### 5.2.2. Model Architecture

In terms of model architecture, the CNN BiLSTM setup starts with an embedding layer, followed by a 1D convolutional layer equipped with 64 filters and a kernel size of 3. After this convolutional layer comes a max pooling layer with a pool size of 2. The output is then fed into a bidirectional LSTM layer featuring 64 units (Zhou et al., 2016). The LSTM output leads into a dense layer with 64 units and ReLU activation, followed by a dropout layer at a rate of 0.5, culminating in another dense layer that matches the number of propaganda techniques units with SoftMax activation.

### 5.2.3. Training

The compilation process for the CNN BiLSTM model involves using the Adam optimizer and sparse categorical cross entropy loss function. To address any imbalances in class distribution, class weights are calculated based on how propaganda techniques are spread out in the training data set. During training, these class weights are inputted into the fit method to aid in handling these imbalances effectively.

The system undergoes training for 10 epochs using a batch size of 32.

### 5.2.4. Validation

Following the training phase, the CNN-BiLSTM model is assessed on the test dataset. The test accuracy is documented, together with a classification report displaying the precision, recall and F1 score for each propaganda tactic. The true and predicted labels for the validation set are then associated with their respective propaganda tactic names and saved in a Data Frame for analysis.

## 5.3. Performance Comparison and Discussion

| Metric | Fine-tuned BERT | CNN-BiLSTM |
|---|---|---|
| Validation Accuracy | 65.22% | 44.93% |
| Validation Precision | 0.47 | 0.47 |
| Validation Recall* | 0.50 | 0.46 |
| Validation F1-score* | 0.46 | 0.40 |
| Test Accuracy | 65.17% | 48.10% |
| Test Precision* | 0.44 | 0.29 |
| Test Recall* | 0.48 | 0.29 |
| Test F1-score* | 0.45 | 0.27 |

*Table 2: Task 2 Model Performance Comparison*

The fine-tuned BERT method achieves a test accuracy of 65.17%, whereas the CNN-BiLSTM method reaches a test accuracy of 48.10%. Comparing the classification report between the two, the fine-tuned BERT method demonstrates higher precision, recall and F1 scores across most propaganda techniques.

The success of the fine-tuned BERT method can be attributed to its capability to grasp contextual information and understand intricate patterns within the text data. By utilizing an attention mechanism, BERT can focus on relevant segments of the input sequence for each propaganda technique.

Conversely, although simpler and computationally more efficient, the CNN-BiLSTM approach may encounter challenges in capturing subtle details and long-range dependencies necessary for precise classification of propaganda techniques.

Both methods incorporate class weighting to address imbalanced distribution of propaganda techniques in the dataset. This technique assigns greater significance to minority classes during training, aiding models in better understanding underrepresented techniques.

I intend to investigate further enhancements that can be made to both methods by exploring various hyperparameters, model structures and feature engineering strategies to improve their performance.

## 6. Error Analysis

### 6.1. Misclassification

Upon reviewing the errors made by the models, a few trends become apparent. For instance, in Task 2 using the fine-tuned BERT method, there are instances where the model mistakes similar propaganda tactics. In one case, it predicted "causal_oversimplification" instead of the correct label "flag_waving". Likewise, in another scenario, it identified "loaded_language" instead of the actual labels "name_calling,labeling".

In the CNN-BiLSTM strategy, there are more frequent misclassifications. For example, it labelled a text snippet as "repetition" when it was actually categorized as "not_propaganda". In another case, it tagged something as "doubt" when it should have been labelled as "appeal_to_fear_prejudice".

### 6.2. Potential Reasons for Errors

There could be various reasons behind these misclassifications by the models;

Similarity between propaganda techniques; Some propaganda methods share common traits that can make differentiation challenging for the models. For instance, both "loaded_language" and "name_calling,labeling" involve using emotionally charged or biased language that may cause confusion.

Inadequate training data; The uneven distribution of propaganda tactics in the dataset can pose challenges for models to grasp nuances from underrepresented classes. This imbalance may lead to misclassifications, particularly for less represented categories.

Contextual dependencies; Propaganda techniques often rely on the context in which they are presented. It can be challenging for models to accurately classify certain techniques due to the long-range dependencies and contextual information involved.

Ambiguity in text; Some text excerpts may contain subtle or ambiguous forms of propaganda, making it difficult for models to identify the correct technique. This can result in misclassifications or false positives.

## 6.3. Suggested Improvements

To improve model performance and address misclassifications, various strategies can be considered;

Data Augmentation; Enhancing the size and diversity of training data can help models learn better representations of propaganda techniques. This can involve generating new examples through data augmentation techniques that apply transformations to existing data.

Ensemble Methods; Combining predictions from multiple models, like fine-tuned BERT and CNN BiLSTM approaches, can mitigate individual model biases and enhance overall classification accuracy.

Attention Mechanisms; Integrating attention mechanisms such as self-attention or hierarchical attention enables models to focus on relevant parts of input sequences and capture long range dependencies more efficiently. Utilising transfer learning involves making use of pre trained models that have been trained on larger datasets or similar tasks, serving as a solid starting point for identifying propaganda and its techniques. Fine tuning these models on the specific propaganda dataset can enhance performance and adaptability.

Employing contextual word embeddings ELMo (Peters et al., 2018) helps in capturing contextual information and semantic relationships among words, thereby improving representation learning.

By conducting detailed error analysis, as exemplified in this section, valuable insights into common patterns and sources of errors can be gained. Continuously refining the models based on these insights leads to ongoing enhancements in performance.

Addressing these potential causes of errors and incorporating suggested enhancements can boost the accuracy of classifying propaganda techniques with the potential to develop more dependable and resilient propaganda detection systems.

## 7. Conclusion

This research delves into examining how propaganda techniques in news articles are identified and categorized using two different methods; fine-tuned BERT and CNN with GloVe embeddings for binary classification and fine-tuned BERT with class weighting and CNN-BiLSTM for multi class classification.

The fine-tuned BERT method shows better performance across both tasks compared to the CNN based approaches. When distinguishing between propaganda and non-propaganda text, the fine-tuned BERT model achieves a test accuracy of 91.03%, surpassing the CNN with GloVe embeddings model at 80.34%. The success of the fine-tuned BERT model stems from its capability to grasp contextual information and interpret intricate patterns within the text data.

In the multi class classification task aimed at recognizing specific propaganda techniques, the fine-tuned BERT approach with class weighting attains a test accuracy of 65.17%, while the CNN BiLSTM approach reaches 48.10%. The attention mechanism embedded in the fine-tuned BERT model, along with incorporating class weights to address imbalanced classes, proves advantageous in accurately identifying propaganda techniques.

This study underscores the significance of refining pre trained language models like BERT for tasks related to detecting propaganda.

The utilization of suitable word embeddings, like GloVe and the integration of techniques that adjust class weights to tackle class imbalances also play a role in enhancing performance. Nonetheless, the research also points out the difficulties in precisely categorizing less common and subtle propaganda methods. The effectiveness of both strategies varies depending on the type of propaganda technique, with superior results seen for more prevalent techniques such as "not_propaganda" compared to rarer ones.

The analysis of errors carried out in the study offers valuable insights into the typical patterns and sources of inaccuracies in propaganda detection models. Factors like similarity between propaganda techniques, insufficient training data, contextual dependencies and textual ambiguities are identified as potential causes for misclassifications. To overcome these hurdles and enhance model performance, approaches such as data augmentation, ensemble methods, attention mechanisms, transfer learning and contextual embeddings are recommended.

To sum up, this study showcases how fine-tuned BERT models can effectively detect and classify propaganda tasks. The results underscore the significance of leveraging pretrained language models, integrating class weighting methods and conducting thorough error analyses to create more precise and dependable propaganda detection systems. However, further research and development are necessary to refine the classification of less common and nuanced propaganda techniques.

## 8. Future Work

Future research in identifying propaganda and categorizing techniques could consider exploring ways to enhance the models. This may involve using larger pre trained models such as BERT-large (Devlin et al., 2018) or employing ensemble methods for better performance. Adding in more features like linguistic cues or domain specific embeddings could offer valuable insights (Barrón Cedeno et al., 2019). It would be beneficial to test these models on larger and more varied datasets to see how well they can adapt (Ribeiro et al., 2020). Looking into few shot learning methods, such as meta learning or prototypical networks, might help tackle the issue of limited labelled data (Brown et al., 2020). Moreover, delving into interpretability techniques like attention visualization or layer wise relevance propagation could give a deeper insight into how the models make decisions. Combining propaganda detection with fact checking and credibility assessment systems could provide a thorough assessment of information reliability (Barrón Cedeno et al., 2019).

## References

Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K. T., Montavon, G., ... & Kindermans, P. J. (2019) 'iNNvestigate neural networks', Journal of Machine Learning Research, 20(93), pp. 1-8.

Barrón-Cedeño, A., Da San Martino, G., Jaradat, I. and Nakov, P. (2019) 'Proppy: Organizing the news based on their propagandistic content', Information Processing & Management, 56(5), pp. 1849-1864.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., ... & Stoyanov, V. (2020) Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116.

Da San Martino, G., Barrón-Cedeño, A., Wachsmuth, H., Petrov, R. and Nakov, P. (2020) SemEval-2020 task 11: Detection of propaganda techniques in news articles. arXiv preprint arXiv:2009.02696.

Da San Martino, G., Yu, S., Barrón-Cedeno, A., Petrov, R. and Nakov, P. (2019) 'Fine-grained analysis of propaganda in news articles', In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 5636-5646.

Deepanshi (2021) 'Text Preprocessing NLP | Text Preprocessing in NLP with Python codes', Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/06/text-preprocessing-in-nlp-with-python-codes/ (Accessed: 12 May 2024).

Devlin, J., Chang, M. W., Lee, K. and Toutanova, K. (2019) 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding', In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171-4186.

Dimitrov, D., Baran, E., Fafalios, P., Yu, R., Zhu, X., Zloch, M. and Dietze, S. (2021) TweetsCOV19--A Knowledge Base of Semantically Annotated Tweets about the COVID-19 Pandemic. arXiv preprint arXiv:2006.14492.

Hochreiter, S. and Schmidhuber, J. (1997) 'Long short-term memory', Neural computation, 9(8), pp. 1735-1780.

Li, J., Chen, X., Hovy, E. and Jurafsky, D. (2015) Visualizing and understanding neural models in NLP. arXiv preprint arXiv:1506.01066.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019) Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

Mitra, T. and Gilbert, E. (2015) 'CREDBANK: A Large-Scale Social Media Corpus with Associated Credibility Annotations', In Proceedings of the International AAAI Conference on Web and Social Media, 9(1), pp. 258-267.

Mysiak, K. (2020) 'NLP Part 2 | Preprocessing Text Data Using Python', Medium. Available at: https://towardsdatascience.com/preprocessing-text-data-using-python-576206753c28 (Accessed: 12 May 2024).

Pennington, J., Socher, R. and Manning, C. D. (2014) 'Glove: Global vectors for word representation', In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543.

Reimers, N. and Gurevych, I. (2019) Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv preprint arXiv:1908.10084.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016) '"Why should I trust you?" Explaining the predictions of any classifier', In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135-1144.