Automated Detection and Classification of Propaganda Techniques in Text

Henry Ash Williams

May 19, 2024

1 Introduction

Propaganda is a covert technique in which one group attempts to manipulate the opinions, beliefs and attitudes of another without them realizing it. This technique has been utilized extensively throughout history and is even seen in contemporary media. For example, shortly before the death of Augustus, the first emperor of Rome, an inscription was made on what was to be his final resting place. The inscription is known as "The Deeds of the Divine Augustus", and outlines both his religious power by describing himself as 'Divine', and his importance to the Pax Romana, or the Roman Peace, by outlining his strategic ability through the anecdotes of his conquests found within the inscription. This was an attempt to shape the attitudes of his citizens towards him to be more favorable. In the years following his death, he was remembered as "The Divine Augustus", and was widely perceived with respect and reverence. However, during his reign, he routinely suppressed dissidents of the Roman Empire in an attempt to centralize his power. While it cannot be told whether the favorable view of Augustus within the Roman Empire was solely due to this text, this was not the only piece of propaganda he produced, and it likely helped to shape the view of his citizens, without them even noticing that their opinion was manufactured.

This technique is not restricted to ancient history and is often seen in our modern media, whether we notice it or not. For example, in the lead-up to and during America's invasion of Iraq, US media reported on an alliance between Saddam Hussein and al-Qaeda to justify a US-led invasion of Iraq. The two groups were, however, largely at odds with each other, and no such allegiance was ever proven. Despite this, in 2002 a majority of Americans believed that Saddam Hussein was directly involved with both al-Qaeda and the 9/11 terror attacks.

A majority of people who are exposed to propaganda do not recognize it as such, and considering the impacts of believing propaganda, both from a security and societal perspective can lead to significant consequences. However, by nature, propaganda is extremely difficult to detect, and many people struggle to recognize it even today. Considering the amount of media produced and consumed each day, a human approach to propaganda detection and classification is impossible, and new approaches are required.

This paper will explore the effectiveness of modern machine learning models in both these tasks, namely the detection of propaganda within text, and classification. Classification in this case refers to the methods of propaganda used within the text. This includes appealing to an individual's sense of fear, exaggerating facts, or the oversimplification of a complex topic. In this paper, we will cover my approaches to these tasks, using natural language processing techniques such as Term Frequency-Inverse Document Frequency, and Unigram Precision, as well as more complex, large language model approaches such as the fine-tuning of BERT models.

1.1 Propaganda Background

According to Scott [7], *"Propaganda is a major form of manipulation by symbols"*. It can be found in reports by news or governments, films, social media, radio television, just to name a few. To better understand the scope of this problem, an in-depth understanding of how propaganda is used to manipulate within the context of this task.

We were provided with a dataset of text samples and labels. The labels are a subset of the ones found in the dataset provided for Task 11 of the Semantic Evaluation 2020 workshop and include the appeal to fear prejudice, casual oversimplification, doubt, exaggeration and minimization, flag waving, loaded language, name calling and labeling, and repetition.

The appeal to fear prejudice seeks to manipulate the opinions of the target audience by provoking panic among them and is likely one of the most effective techniques to manufacture opinions [9]. This can be seen throughout history, one of the best examples of this is Joeseph Goebbels's usage of the technique to convince the German public that the Allies wanted to exterminate the German people during the reign of the Nazi party.

Casual oversimplification assumes a single cause or reason for an issue despite the reality being far more

complex [1]. This could include blaming societal issues on a single group of people.

Within propaganda, doubt is used to influence perception by encouraging its targets to question the legitimacy of others. This could take the form of an advertiser shaping the perception of a competing brand's effectiveness. Thus, encouraging the target to buy their product instead.

Exaggeration and minimization unsurprisingly, is the opposite of casual oversimplification. This is often seen in advertising, where brands exaggerate claims about their products to make them seem better than those of their competitors. Minimization is very similar to this, whereby a brand may make problems with their products seem like less of an issue than they are. They are grouped in the dataset due to their relatively low frequency, and the commonalities between them.

Flag waving utilizes strong nationalistic views to promote actions or ideas [4]. To re-use the advertiser example, it could be promoting the idea that their brand's product is better because it is manufactured in a certain country.

Name-calling and labeling are grouped as well, as they are very similar. These techniques assign the object of the propaganda campaign (the entities which the propaganda campaign seeks to manipulate individuals' perceptions) a name that the targets would find undesirable. This could be a brand calling a competitor's product an imitation, which most people would dislike.

Repetition involves repeating a message over and over again until the targets accept it as true.

2 Methods

2.1 Data Analysis

To develop a model capable of achieving the goals outlined in the previous section, we need a thorough understanding of the data we've been provided with. The dataset of text samples, some of which contained propaganda, and others that didn't. Overall, there were 3,200 samples, 50% of which contained propaganda, and 50% of which didn't. It came in the form of two separate tab-separated value files, one for training, and another for testing. The testing dataset contains 640 samples, 48% of which contain propaganda. The training dataset contains 2,560 samples, 52% of which contained propaganda.

There are 9 labels, including one for samples without propaganda. The average word count, as well as the standard deviation, and range of each class, is displayed in Table 1. As we see, the average length of each sample across the various propaganda classes is approximately 39 words. However, some classes such as name calling and labeling, and casual oversimplification on average exhibit longer sequences. Despite this, we see little correlation between word count within the samples and the label assigned to them.

Within each of the text samples, there are tags indicating the area of interest. This indicates where the propaganda is within the sample. We extracted the text within this area of interest and calculated the same metrics as we did for Table 1. The results of this are demonstrated in Table 2. While they exhibit a correlation between the number of words in the sample, and a the correlation isn't very strong, and likely shouldn't be used to predict and classify propaganda.

In order to demonstrate the effectiveness of my approach, it is first necessary to establish a baseline for performance. In this case, the baseline would be randomly selecting a label for a given sample. For classification, the baseline is 50% and for classification, as there are 8[1] labels to choose from, the baseline accuracy is around 12.5%.

2.2 Detection

Propaganda detection is a binary classification task, meaning the output of the system will be one of two values. In this case, 1 indicates the text sample contains propaganda, and 0 indicates the lack of propaganda within the sample. There are several approaches to text-based binary classification such as bag-of-word-based approaches [6], which make use of a vocabulary of all unique words in the training data, and then a machine learning algorithm such as Naive Bayes or Support Vector Machines to perform classification.

2.2.1 Bag of Word classifier

A bag of words approach first requires us to pre-process our training data. This typically involves tokenization, where text is split into individual words, converting the tokens to lowercase, removal of stopwords (words that do not carry significant meaning such as "the", "and", "is", etc), and the lemmatization of the remaining words [6].

In Linguistics, a Lemma is the base form of a word. For example, the words "running", "ran", "runs" are all lexemes of the lemma "run". The process of converting a given lexeme to its base form, i.e. its' lemma, is known as lemmatization and is a critical step of data preprocessing within the development of bag-of-word classifiers. This is because it significantly reduces the dimensionality of the set of words, which thereby reduces the complexity of the classifier. For example, the training data has 3261 unique words that are classified as propaganda, upon performing lemmatization on these words, there are 3043 words. Furthermore, when it comes time to perform inference using our model, we don't want to miss a word within the known propaganda words dataset simply because it's in a different form. This requires us to perform the above pre-processing steps on our test data for our training data.

[1] The inputs to this model are known to contain propaganda, so we can disregard

Label		Metrics			
ID	**Description**	**Samples**	**Mean**	**Std**	**Range**
1	Appeal To Fear Prejudice	200	39.65	17.38	107
2	Name Calling, Labelling	200	44.49	23.15	144
3	Exaggeration, Minimisation	200	40.03	18.86	87
4	Not Propaganda	1600	30.02	15.25	131
5	Loaded Language	200	37.72	18.49	93
6	Casual Oversimplification	200	43.58	21.96	132
7	Doubt	200	41.65	25.45	159
8	Repetition	200	34.42	18.6	120
9	Flag Waving	200	40.0	18.63	112

Table 1: Labels found within the dataset, and metrics relating to the word count of the text samples

Label	Metrics		
ID	**Mean**	**Std**	**Range**
1	23.72	14.75	87
2	22.95	19.99	159
3	3.14	4.08	38
4	19.24	13.61	79
5	8.15	6.2	46
6	7.26	8.01	92
7	12.04	13.27	78
8	4.8	3.92	23
9	3.72	3.7	29

Table 2: Metrics of text samples with area of interest extraction

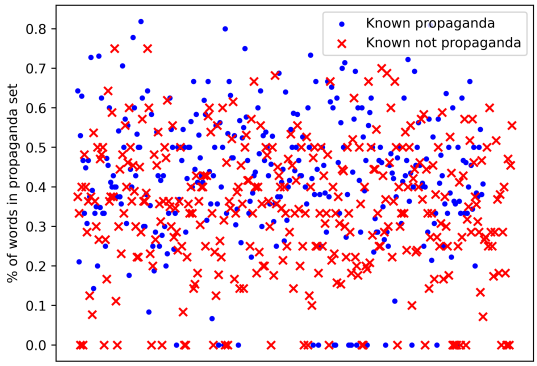

Figure 1: Correlation between percentage of words in common with propaganda set, and known classification

To determine the class of a given sample, we can simply check how many words are common between our bag of words, and the sample we want to test, divided by the total number of words in the sample. This value is fed into a simple logistic regression algorithm, which then makes a prediction. The results for this approach are demonstrated in Table 3. This approach is based on an algorithm known as BLEU [5] which is used in evaluating the quality of machine translated text.

I chose to experiment with this model, as it offers a solid baseline for further development. It also is very simple to get up and running quickly. However, its performance is slightly above our random baseline, achieving a 60% test accuracy. To understand why the accuracy was so low, I analyzed the correlation between the number of words shared between test samples, and the set of words commonly found in propaganda. This analysis is shown in Figure 1, and we see that there is very little correlation between the overlap of words found in the test sample and the words in the propaganda set, and the label assigned to the test sample explaining the low accuracy of the model.

2.2.2 Term Frequency-Inverse Document Frequency and Support Vector Machines

I then went on to enhance this model using Term Frequency-Inverse Document Frequency, TF-IDF [2]. This approach adds additional information in the form of the word importance within the corpus based on how often it occurs. By transforming the training dataset. The relative frequency of a given term t within a document d of a corpus D is calculated using the expression found in Figure 2, where $f_{t,d}$ is the number of occurrences of the term t in the document d, and $\sum_{t' \in d} f_{t',d}$ is the total number of terms in d.

The training dataset is then to its TF-IDF representation, which a support vector machine [8] was then trained on. A support vector machine is similar to the logistic regressor used in the bag of words approach as it finds a hyperplane that separates the two classes. However, it seeks to maximize the distance of the margin between them. The margin is the distance between the hyperplane and its closest data point on either side of its decision boundary. This allows support vector ma-

$$\mathrm{TF}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

$$\mathrm{IDF}(t,D) = \log\left(\frac{|D|}{1+|\{d \in D : t \in d\}|}\right)$$

$$\text{TF-IDF}(t,d,D) = \mathrm{TF}(t,d) \cdot \mathrm{IDF}(t,D)$$

Figure 2: How a term's relative frequency within a document given a corpus is calculated using TF-IDF.

chines to better generalize the task and prevents overfitting. A kernel is used to transform the input space in cases where the data is not linearly separable. To determine the best configuration of this approach, I evaluated the model's accuracy on each of them. I found that on average, sigmoid, linear, and radial basis function (RBF) kernels are optimal for this task, reaching 73% test accuracy. These results were obtained over 10 iterations to reduce the presence of outliers within the data. The results I gathered for this are demonstrated in Table 3.

2.2.3 Bidirectional Encoder Representation from Transformers

I then went on to test the effectiveness of the Bidirectional Encoder Representation from Transformers (BERT) large language model introduced by researchers at Google [3]. This model uses an "encoder only" transformer architecture. Encoders are responsible for learning and extracting relevant information from the input text and output an embedding. These embeddings include a one-hot vector of the input token, a positional encoding, and a token type. It has a contextual understanding of text, can consider words on either side of the current word, and demonstrates excellent performance on similar tasks.

When applying BERT to a new problem, an approach known as transfer learning is used. This is where a trained model is re-trained on a new task. This allows us to significantly decrease the time spent training, and the amount of data necessary for it. When applying the pre-trained model to this task, it demonstrates the worst performance out of all the models I've tested, even falling behind our baseline of randomly picking labels. However, after performing training for 10 epochs, taking approximately 10 minutes on an Apple Macbook Pro with an M2 Pro chip, and 16GB of memory, it quickly surpasses the bag of words classifier in terms of performance, see Table 3. To further enhance the performance of this model, I performed a technique known as Bayesian hyperparameter optimization [10]. Hyperparameters are first selected randomly, then either the parameters with the best performance or the point that has the highest potential to achieve a better result and the point with the highest uncertainty are selected for

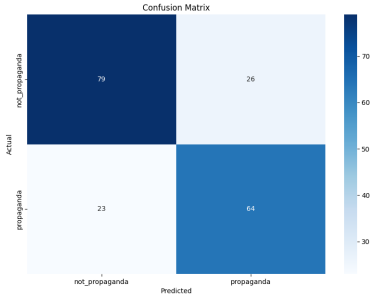

Figure 3: The confusion matrix of the propaganda detector

the next run to see if changing them improves performance.

I first trained my model using the following hyperparameters: a batch size of 32, a learning rate of 2e-5, a weight decay of 0.01, a dropout rate of 0.1 and the Adam optimization function. After training for 5 epochs, this model achieved approximately 70% test accuracy. However, after hyperparameter optimization, I was able to achieve 80% test accuracy. These results are shown in the Figure 3. The most significant parameter to the performance of the model was the optimization function used by the model. For example, no models trained using the RMSprop optimizer achieved better than a 0.44 test f1 score, and models trained using the Adam optimizer did not achieve better than a 0.41 test f1 score. However, models trained using stochastic gradient descent were able to reach up to a 0.80 test f1 score. Weight decay and dropout rate do not seem to have much of an impact on performance, as the best-performing models all had a wide variety of values, for example, the dropout values ranged from 0.1174 and 0.3859, and weight decay ranged from 0.002 to 0.009. Learning rate, however, had a much smaller range for these top-performing models, from 0.04 to 0. Most of the values within this range are clustered towards 0.01 leading me to believe it should be extremely small.

2.3 Classification

Classification is a more challenging task. We are given the region of interest from within a text sample. This sample is known to contain propaganda. Therefore, we only have 8 labels to consider. I began by applying the same TFIDF technique to the problem. This resulted in an accuracy and f1 score of 50%. Considering the baseline for this task would be an accuracy of approximately 12% and an F1 score of approximately 0.14, the TFIDF+SVC model provides decent performance. However, there is still room for improvement.

I chose to re-use the BERT model for the classification task, as it demonstrated excellent performance in the first task, after hyperparameter optimization. I took the same approach to pre-processing the training data

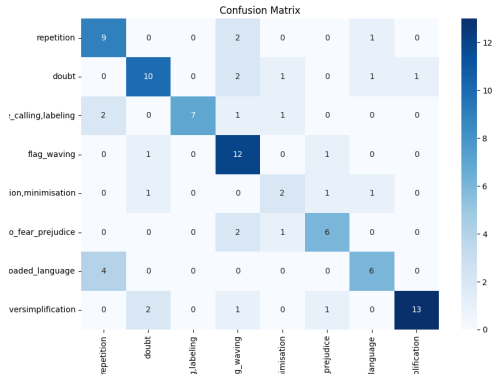

Figure 4: The confusion matrix of the propaganda classification model.

as I did for the pre-processing task, using the BERT tokenizer. However, I did remove all samples without propaganda present, as they are not considered in this task. I also chose to use the same hyperparameters to begin with. This model achieved 46% accuracy. As demonstrated in the previous task, however, there is room to improve this metric.

To improve the performance of this model, I again performed a hyperparameter sweep. I ran this sweep on an RTX 4090[2], which allowed me to train the model over 100 times in under 2 hours. The best-performing model I trained during this sweep, achieved a test accuracy of 69%, using the stochastic gradient descent optimizer, a batch size of 64, a dropout rate of 0.21, a learning rate of 0.02, and a weight decay value of 0.009. The confusion matrix for this model is displayed in Figure 4.

3 Evaluation

This section will discuss the performance of the approaches outlined in the previous section.

3.1 Detection

Detection is by far the easiest of the two tasks. It has a small solution space and requires a less in-depth understanding of what's being said. To draw from the real world, it's far easier to present an individual with two pieces of media and ask them which one contains propaganda and which doesn't, and then ask what propaganda techniques are in use. They will likely find it far easier to determine which is propaganda than they would to classify which techniques are in use within the media.

The results from these experiments suggest that this hypothesis is correct. On average, achieving far better performance than the classification approaches.

[2]Special thanks to my housemate Byron who let me use his PC for this

3.1.1 Bag of Words

Out of each of the approaches for this task, this one demonstrated the worst performance. Only improving upon our baseline by 10%. I believe this to be because there is not much correlation between the percentage of words in common with the words found in propaganda and the label assigned to it.

During development, I originally used stemming instead of lemmatization when preprocessing both my training and testing data. This resulted in a

This indicates that this model is not suitable for this task, and alternative approaches should be chosen instead. Despite this, the ease of development these kinds of models offer makes them a good option to try to use as a baseline for performance, as randomly picking labels isn't a good means of making predictions.

3.1.2 TF-IDF + SVC

Term Frequency-Inverse Document Frequency with a support vector classifier offers surprisingly good performance considering the ease of development. The models displayed the best performance among non-deep learning-based approaches. This could indicate that these approaches could be preferable on hardware that does not have the necessary computational resources to run large language models such as BERT locally and lacks the networking hardware necessary to communicate with a remotely hosted model or the latency posed by such a solution would be undesirable. These approaches display a good level of accuracy, while still being efficient. It took 0.25 seconds to fit the Support Vector Machine, on a dataset of 2436 samples, and 0.05 seconds to make 512 predictions from the model. A large language model-based approach took 1.34 seconds to perform the same number of inferences on top-of-the-line desktop hardware, which a lot of users may not have access to.

3.1.3 BERT

The BERT-based approaches to this task demonstrate similar performance to the bag of word approach. This could indicate that more training is required, or a larger training corpus could be necessary.

However, after performing hyperparameter optimization, the model shows excellent performance. It achieved an 80% test accuracy and F1 score. This indicates this model could be an excellent choice for a real-world application, such as the automatic labeling of news articles to help inform readers and develop better media literacy. However, care should be taken to prevent bias within the dataset.

3.2 Classification

As explored in the previous section, classification is the hardest of the two tasks as it requires a in-depth understanding of each of the techniques. This is reflected in

Task	Method	Performance			
		Accuracy	F1 Score	Precision	Recall
1	Random Baseline	0.48	0.49	0.49	0.49
1	Bag-of-Words Logistic Regression	0.61	0.64	0.61	0.61
1	TF-IDF SVC (With linear kernel)	0.72	0.72	0.72	0.72
1	TF-IDF SVC (With polynomial kernel)	0.66	0.65	0.64	0.65
1	TF-IDF SVC (With rbf kernel)	0.73	0.73	0.73	0.73
1	TF-IDF SVC (With sigmoid kernel)	0.72	0.72	0.72	0.72
1	BERT Baseline	0.51	0.34	0.26	0.51
1	Fine-Tuned BERT	0.63	0.63	0.63	0.63
1	Fine-Tuned BERT With Hyperparameter Optimization	0.8	0.8	0.81	0.8
2	Random Baseline	0.12	0.14	0.3	0.12
2	TF-IDF SVC (With sigmoid kernel)	0.5	0.5	0.53	0.5
2	BERT Baseline	0.11	0.04	0.03	0.1
2	Fine-Tuned BERT	0.46	0.43	0.47	0.46
2	Fine-Tuned BERT With Hyperparameter Optimization	0.69	0.7	0.72	0.69

Table 3: The results and key metrics of my approaches for both tasks, task 1 being detection of propaganda, task 2 being its classification

the results of my approaches to this task, as the accuracy of the models I've tested is, on average, worse than the results obtained for the first task.

3.2.1 TF-IDF + SVC

The TF-IDF and SVC approach was able to achieve the correct label 50% of the time. This indicates that this model would be unsuitable for real-world applications and that another approach should be chosen.

This could be due to the high dimensionality of the training data. As TFIDF creates a high-dimensional, sparse matrix where each dimension represents a single term in the corpus, classification becomes increasingly difficult to perform accurately as the number of features increases which would lead to overfitting.

The imbalance of classes could also pose an issue to these models. However, as seen in Table 1, the classes are uniformly distributed, with each class having 200 samples. The only exception to this is for samples which do not contain propaganda. However, in this task, samples without propaganda are disregarded and as such, this is not an issue.

Furthermore, Support Vector Machines are inherently binary classifiers and as such, should not be used for multiclass classification. They are also limited in their understanding of context, as they do not capture the relationships between words.

Despite the inherent limitations of Support Vector Machines for multiclass classification, the model performs significantly above our baseline, indicating that the distribution of words could be used in further work.

3.2.2 BERT

The fine-tuned BERT model for snippet classification performed even worse than the support vector machine. However, after hyperparameter optimization, it achieved a respectable 69% accuracy.

As displayed in Figure 4, the model struggled to predict the exaggeration and minimization of propaganda techniques. This may be because, to detect this form of propaganda, the model needs to know if the alleged facts within the snippet are true or not. This would be incredibly difficult for an automated system, as it requires a lot of knowledge of the world as a whole, and what is or isn't true. This is also one of the hardest forms of propaganda for humans to detect, as it requires

knowing things that may have not been made publicly available.

4 Conclusion & Further Work

In conclusion, this paper has discussed my approach to the automated detection and classification of propaganda within text.

I was able to achieve 80% test accuracy of propaganda detection after hyperparameter optimization. However, before I performed this optimization, the model achieved similar accuracy to a TF-IDF and SVM model, which may be a better choice for the task at hand due to its lower computational complexity.

Classification of propaganda was far more difficult, and I was only able to achieve approximately 70% testing accuracy after hyperparameter optimization using BERT. The most challenging aspect of this task was the classification of exaggeration and minimization as it requires a good understanding of the world, which deep learning models would not possess.

Future work could consider the impact of using state-of-the-art language models, such as OpenAI's GPT V4. However, as the model is currently a closed source, fine-tuning this model is impossible, and the sheer number of parameters in the model itself would likely make training very difficult. Other models such as Llama, or Mixtral could be tested instead as they are available for download, but they still require expensive hardware to fine-tune as they both have a very high number of parameters.

The LaTeX source code for this document, along with all the code used to test this model are publicly available on my Github page. This includes data relating to the training runs performed as part of hyperparameter optimization.

References

[1] Giovanni Da San Martino et al. "SemEval-2020 Task 11: Detection of Propaganda Techniques in News Articles". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Ed. by Aurelie Herbelot et al. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 1377–1414. DOI: 10.18653/v1/2020.semeval-1.186. URL: https://aclanthology.org/2020.semeval-1.186.

[2] Seyyed Mohammad Hossein Dadgar, Mohammad Shirzad Araghi, and Morteza Mastery Farahani. "A novel text mining approach based on TF-IDF and Support Vector Machine for news classification". In: *2016 IEEE International Conference on Engineering and Technology (ICETECH)*. 2016, pp. 112–116. DOI: 10.1109/ICETECH.2016.7569223.

[3] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[4] Renee Hobbs. "Teaching about Propaganda: An Examination of the Historical Roots of Media Literacy". In: *Journal of Media Literacy Education* 6 (Nov. 2014), pp. 56–67. DOI: 10.23860/JMLE-2016-06-02-5.

[5] Kishore Papineni et al. "Bleu: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Ed. by Pierre Isabelle, Eugene Charniak, and Dekang Lin. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://aclanthology.org/P02-1040.

[6] Wisam A. Qader, Musa M. Ameen, and Bilal I. Ahmed. "An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges". In: *2019 International Engineering Conference (IEC)*. 2019, pp. 200–204. DOI: 10.1109/IEC47844.2019.8950616.

[7] J. Scott. *Power: Critical Concepts*. Critical Concepts in Sociology Series. Routledge, 1994. ISBN: 9780415079389. URL: https://books.google.co.uk/books?id=JhOgKax0IYIC.

[8] Shan Suthaharan. "Chapter 7: Support Vector Machine". In: *Machine Learning Models and Algorithms for Big Data Classification: Thinking with Examples for Effective Learning*. 1st ed. Vol. 36. Integrated Series in Information Systems. Published: 21 October 2015, 23 August 2016, 20 October 2015. New York, NY: Springer New York, 2015. Chap. 7, pp. 207–235. ISBN: 978-1-4899-7640-6. DOI: 10.1007/978-1-4899-7641-3. URL: https://doi.org/10.1007/978-1-4899-7641-3.

[9] Melanie B Tannenbaum et al. "Appealing to fear: A meta-analysis of fear appeal effectiveness and theories". en. In: *Psychol Bull* 141.6 (Nov. 2015), pp. 1178–1204.

[10] A. Helen Victoria and G. Maragatham. "Automatic tuning of hyperparameters using Bayesian optimization". In: *Evolving Systems* 12.1 (2021), pp. 217–223. ISSN: 1868-6486. DOI: 10.1007/s12530-020-09345-2. URL: https://doi.org/10.1007/s12530-020-09345-2.