Universiteit Utrecht

Classification of Propaganda on Fragment Level: Using Logistic Regression with Handcrafted Contextual Features

Author
S. (Sylvain) Maissan

Supervisor
Dong Nguyen

Second Examiner
Frans Adriaans

June 26, 2020

Abstract

Propaganda in the media has become a rising problem, especially after automation. The ease of which propaganda can be created and spread is astonishing. A way to combat this is an automated propaganda detection system. The goal of fine-grained propaganda detection is to determine whether a given sentence uses a propaganda technique, or to recognize which techniques are used on the fragment level. In this paper we try to analyze the effects of contextual features on the fragment level when training a propaganda classifier. Using a logistic regression model I created some handcrafted features that solely depend on contextual information. The results showed no significant impact on the performance. The features based on the propagandistic fragment itself prove to be the top features in this setting. In future research it is recommended to create either more complex contextual features or to create features that are able to discern whether the fragment is *Loaded Language* or *Name Calling*.

Keywords: Propaganda Classification, Fragment Level, Contextual Features,

Contents

1 Introduction

Propaganda aims at influencing people's mindset with the purpose of advancing a specific agenda. It uses psychological and rhetorical techniques to persuade the public. These techniques are intended to go unnoticed to achieve the maximum effect. For example, a commonly used technique is *Name Calling*. The goal for this technique is labelling a person, object, or political organisations as something that people either hate or adore. So, if someone calls the prime minister a 'saint', then he or she uses *Name Calling* to influence your opinion about the prime minister. Another propagandistic technique is *Loaded Language*. This technique uses emotionally 'loaded' words to influence you in a certain way. An example is a politician saying 'bureaucrats' instead of 'public servants'. The word 'bureaucrats' has a more negative annotation, compared to 'public servants'. Thus, the choice of words can influence the connotation of a sentence and the way the audience interprets it.

The modern negative use of the word 'propaganda' originated from World War I. During that time, its main use was to rally people behind the idea to go to war. Since then the issue has been raised in identifying propaganda and censoring it. The problem is that identifying propaganda is challenging to say the least. The main difficulties lie in differentiating propaganda from other types of persuasion techniques. The difference between propaganda and persuasion is that propaganda only satisfies the needs of the propagandist. Persuasion, on the other hand, is interactive and attempts to satisfy the needs of both persuader and persuade (Jowett & O'Donnell, 2018). An example of persuasion could be a businessman convincing an investor to invest in his new product or service. The needs of the businessman are fulfilled. He gets his money to produce and sell his new product and the investor gets his needs (earning money) fulfilled, because he gets a share of the profit. The digital era has only amplified the problem of differentiating between propaganda and other techniques to new heights. The rise of the Web, and a combination of freedom of expression and a low threshold for sharing information, has nurtured the number of news outlets that produce and distribute propagandistic content. These hyper partisan news outlets publish articles on social media that contain false and misleading information (Silverman, et al., 2016). Hence, the process of labelling articles as either propagandistic or non-propagandistic is not feasible without any form of automation. The solution for this is a propaganda detection model. Propaganda detection uses a model which determines whether an article is propagandistic or not by certain definitions. There has been research on how to define and recognize propagandistic techniques (Miller, 1939). There is also another issue for automated propaganda detection: the Information Overload Problem. This is the problem of a user being incapable of processing all the information that he or she receives on social media (Gomez-Rodriguez, et al., 2014). They need a system that will warn the user if the article contains propaganda. The user will then have the freedom of choice and complete information to decide if they want to read the article.

The aim of this paper is to develop a system for the fragment level multiple classification task (FLC). The FLC task is given a text fragment identified as propaganda, the model has to classify the applied propaganda technique in the fragment. The inspiration for the task is the SemEval 2020 competition. The task presented in the competition consists of two parts: Span Identification and Technique Classification. The experiment in this paper uses the classification task. I had to alter the dataset so that it can be used in the experiment. During the competition there was almost no usage of contextual features. I will investigate the use of contextual features for improving classification performance. The reasoning behind the choice of using contextual features is that if the system focuses only on the given text fragment identified as propaganda, then it might miss some essential information for the classification. Let us take the sentence “They act like children.” as an example. In this sentence, the word ‘children’ can have two meanings. The first meaning can be pedagogical. It could be a sentence from an article about the behaviour of children. The sentence is then not propagandistic. The second possible meaning of the sentence can be political and thus propagandistic, because the word ‘children’ is a form of *Name Calling.* So, if the system uses context as part of its input, then it could make a difference in the classification of the fragment. Contextual is defined as depending on surrounding words, phrases, and paragraphs of the writing. A feature is an individual measurable property of a phenomenon being observed. So, contextual features are the properties of the text surrounding the fragment that the system gets as input. An example is the sentence with the structure ”[A] [Propaganda technique] [B].” The system uses information from parts A and B to classify the propaganda technique.

The remainder of this paper is organized as follows: Section 2 gives background information and discusses related work. Section 3 describes the labels and the dataset that are used, and statistics about the training’s dataset. Section 4 gives the details about the used method, such as pre-processing, features, models, and evaluation. Section 5 describes the evaluation results and discusses the results that stick out. Section 7 discusses the limitations of the experiments and gives some new suggestions for the future. Finally, section 8 concludes the paper.

2 Related Work

Propaganda detection first started at the news outlet level. An organisation would check if news outlets would write articles without any propaganda. If too many articles were propagandistic, then the news outlet would receive a negative label and all the articles written by them would be labelled as propagandistic. The problem with this method of work was that not all the labelled news outlets would write propagandistic articles. Sometimes, they would write articles without any form of propaganda to increase their credibility (Horne, et al., 2018). In addition, the mainstream media would sometimes write articles containing propaganda to promote a specific agenda. So, the next step in propaganda detection would be on the article level.

Another reason for article level propaganda detection is the rise of fake news. The American presidential election in 2016 has shown the existence and persuasiveness of fake news to the public (Allcott & Gentzkow, 2017). A part of the success of fake news came from so-called hyperpartisan news publishers, which report strongly in favour of one political position. In response to this phenomena people started to fact-check the political statements and to create models to structure this process (Rashkin, et al., 2017). For example, the Long short-term memory model that was created at that time, takes a sentence as input and predicts its Politifact rating, the level of truthness of the sentence. Another method was hyperpartisan detection (Kiesel, et al., 2019). A system would rate the level of hyperpartisan content and label the article on a 5-point scale. The difference between hyperpartisan content and propaganda is that hyperpartisan content has an extreme bias in favour of one political party. An article labelled as hyperpartisan uses propaganda to manipulate people. So, propaganda detection is more about the techniques used in an article and not what political side the techniques support. With the use of hyperpartisan detection, propaganda detection did not follow too long after.

Propaganda detection has already been explored on the article level (Barrón-Cedeño, et al., 2019). However, to build models that can explain to the user why an article is propagandistic, the model should be able to detect the specific technique in a sentence or even a fragment.

The NLP4IF shared task on fine-grained propaganda detection aims to produce models capable of identifying propaganda techniques present in sentences or fragment. The task is divided into two sub-tasks. The first sub-task is sentence level classification (SLC). This is a binary task. The model classifies a given sentence with either *propaganda* or *non-propaganda*. A sentence needs to have at least one propaganda technique to be classified as propaganda. The second task is fragment level classification (FLC). Here, the model should be able to identify and classify a text-fragment containing the propaganda technique. The identification subtask is about finding the span that contains propaganda and after that the model classifies the propaganda technique used in the identified fragment.

During the NLP4IF competition the use of BERT and Long short-term memory models were a popular choice in both the FLC and SLC. Logistic regression was only chosen as the model for the SLC. The first team got 4th place out of 25 with Logistic regression. Their F_1 score was 0.623. The other team got 12th place with a score of 0.5770. The team with the highest F_1-score on the SLC used an attention transformer using BERT trained on Wikipedia and BookCorpus (Mapes Jr., White, Medury, & Dua, 2019). They had a precision of 0.6028 and a recall of 0.6648 that gave the 0.6323 F_1-score.

The inspiration for this research comes from SemEval 2020 Task 11 competition. This competition is the follow up of the NLP4IF task. The sub-tasks in this competition are Span Identification and Technique Classification. Span Identification is a binary sequence tagging task. So, given a plain-text document, it identifies those specific fragments which contain at least one propaganda technique. The second task, Technique Classification, is a multilabel classification problem. Given a text fragment identified as propaganda and its document context, it identifies the applied propaganda technique in the fragment.

Only one team used context in its propaganda detection model for the NLP4IF SLC task (Hou & Chen, 2019). They used two context-aware representations based on BERT. The first one used the target sentence and the title of the article. The second representation consisted of the target sentence and its previous sentence. It is noteworthy that only one team used context. The highest score that the team got was a 0.67 F_1 score. The precision and recall were respectively 0.59 and 0.79. The model with context has the best setup of all the teams in the experiment. Considering this, it can be assumed that there could be more potential in using context as a feature for classification.

3. Dataset

The data that I am using are fragments from news articles, which are confirmed to be propagandistic. The data is annotated by a company called A Data Pro. This is a company that specialises in data annotation. The annotation task was not well suited for crowdsourcing, because of personal bias and the time required to understand all the propaganda techniques is significant. These spans are labelled with one of the 18 classifications used by (Da San Martino, et al., 2019). All the techniques are explained in figure 1.

Propaganda Technique	**Explanation**	**Example**
Loaded Language	Using strong emotional words to influence someone.	"We made tremendous progress on the project."
Name calling or labelling	Labelling the object as either something the audience fears, hates or finds undesirable, or otherwise loves or praises.	"That politician was the hero of the day."
Repetition	Repeating the same message over and over to convince the audience.	"The boy was a good footballer, because his father was a footballer, and his grandfather was a footballer."
Exaggeration or minimization	Exaggerating the object to something bigger, better, worse, etc., or reducing an object or problem to something smaller and insignificant.	"We were not arguing; we were having a heated discussion."
Doubt	Questioning the credibility of someone or something.	"Does he really have the capabilities to lead this country."
Appeal to fear/prejudice	Creating support for an idea by instilling fear in the audience towards an alternative, possibly based on preconceived judgements.	"Stop people immigrating from Syria, because they have connections with ISIS."
Flag-waving	Using nationalism or patriotism to find support for a cause.	"This will make the country the best in the world."
Causal oversimplification	Assuming one cause when there are multiple causes behind an issue. Scapegoating is included into this category.	"If France had not declared war on Germany, World War II would have never happened."
Slogans	A short and striking phrase that may include labelling and stereotyping. Slogans often have an emotional impact.	"Make America great again!"
Appeal to authority	Stating that a claim is true simply because it was said so by an expert or authority. There is no further supporting evidence to support the claim.	"These vitamins are great, because my doctor said so."

Black-and-white fallacy	Acting like there are only two options or sides, when there is a more nuanced middle ground.	“There is no alternative to war. “
Thought-terminating cliché	Words or phrases that stop critical thinking. This halts meaningful discussion and gives a lacking answer to complex questions.	“It is how it is.”
Whataboutism	Discrediting an opponent’s position by charging them with hypocrisy without directly disproving their argument.	"What about the alt-left that came charging at the, as you say, the alt-right? Do they have any semblance of guilt?"
Reductio ad Hitlerum	Persuading an audience by linking the action or idea to a hated group or organisation.	“That is something only the Nazi’s would do.”
Red Herring	Introducing an irrelevant material to the issue being discussed, so the attention is diverted from the issue.	"We need more revenue to support the programs that we have. Children are our future. Let's support children."
Bandwagon	Persuading the audience to join in and act by stating that everyone likes it that way.	Would you vote for Clinton as president? 60% says yes.”
Obfuscation, intentional vagueness, confusion	Only using unclear words and vague statements so everyone will interpret it their own way.	“If we could turn back the clock and change what happened, obviously we wouldn’t have done it. We can’t.”
Straw Man	Projecting the opponent with a certain argument even though the argument of the opponent is more nuanced, so it can be refuted.	A: “We should relax the laws on beer.” B: “No, any society with unrestricted access to intoxicants loses its work ethic and goes only for immediate gratification.”

Figure 1

For this research *Bandwagon* and *Reductio ad Hitlerum* are combined into one class. The same holds for *Straw Men, Red Herring* and *Whataboutism*. The reason for this is that these labels have a low frequency and the algorithm will have too much difficulty learning these labels separately. The label *Obfuscation, intentional vagueness, confusion* is removed, because of the inconsistent annotation. So, there are 14 labels left for classification. These labels are the same ones that are used in the SemEval competition.

The dataset used in this experiment is from the SemEval 2020 competition training set. The reason that only the training set is used is because the other datasets are to be released in the future, so there is no way to access it. So, I divided the training set containing 371 articles. Figure 2 shows how the articles are divided into a training, development, and test set. I divided the set per article and not per span, because it is simpler to have an article to be only in one set. The ratios for the division are similar to the datasets used in previous propaganda detection tasks. The training set has 3398 technique instances in total. The statistics of the training set are listed in figure 3. Every article contains on average 14 propaganda techniques. The average propagandistic fragment has a length of 46 characters (whitespaces included). The longest fragment has 712 characters and the shortest fragment has 3 characters. The most common propagandistic techniques are *Loaded Language* and *Name Calling, Labelling*. These labels combined are 46.4% of all the occurrences.

Training Set	246 articles
Development Set	41 articles
Test Set	84 articles
Total	371 articles

Figure 2.

Category	**Frequency**	**Percentage**
Loaded Language	1128	33.2%
Name Calling, Labelling	448	13.2%
Repetition	406	11.9%
Doubt	237	7.0%
Exaggeration, Minimisation	248	7.3%
Appeal to fear-prejudice	233	6.9%
Flag-Waving	146	4.3%
Causal Oversimplification	108	3.2%
Appeal to Authority	104	3.1%
Slogans	59	1.7%
Black and White Fallacy	88	2.6%
Whataboutism, Straw Men, Red Herring	78	2.3%
Thought-terminating Cliches	59	1.7%
Bandwagon, Reductio ad hitlerum	56	1.6%
Total	3398	100%

Figure 3.

4. Methods

4.1 Preprocessing

The first step was to alter the data from the competition, so that the input is not an article, but a propagandistic fragment that is to be classified. I used the gold labels of the Span Identification task to get all the propagandistic spans from an article. Now, every article file consists of a propagandistic fragment on every line. An example is the following figure:

```
appeared
The next transmission could be more pronounced or stronger
a very, very different
He also pointed to the presence of the pneumonic version, which spreads more easily and is more
virulent, in the latest outbreak
but warned that the danger was not over
when (the plague) comes again it starts from more stock, and the magnitude in the next transmission
could be higher than the one that we saw
the magnitude in the next transmission could be higher
it could even spill over into neighbouring countries and beyond
```

fig 3.1: altered article text file 111111111

The label files consist of one propaganda technique on every line corresponding to the span in the datafile. An example is the following figure:

```
Doubt
Appeal_to_Authority
Repetition
Appeal_to_fear-prejudice
Appeal_to_fear-prejudice
Appeal_to_Authority
Appeal_to_fear-prejudice
Appeal_to_fear-prejudice
```

fig 3.2: altered article label text file 111111111

4.2 Features

The features are what the model receives as input. Every datapoint has the same set of features with different values. A quick example is letting a model recognize what a fire truck is. The features that would intuitively fit with this task are for example: Is it a vehicle? Is it red? Does it have a ladder? etc. So features are the properties that the model will focus on. You can see in figure 4 the list of features that are used, with an explanation.

Feature Names	Explanation	Form	Contextual?
Span-Count Vectorizer (CV)	A vocabulary of known words in the training fragments	A 2Dvector with *n* rows and *m* columns. *n* is the number of datapoints. *m* is the number of features. Every column represents a word that was found in the training set.	No
Pre-CountVectorizer (PreCV)	A vocabulary of known words in the sentence before the propagandistic fragment	A 2D vector with *n* rows and *m* columns. *n* is the number of datapoints. *m* is the number of features. Every column represents a word that was found in the sentence before the fragment	Yes
Post-Count Vectorizer (PostCV)	A vocabulary of known words in the sentence after the propagandistic fragment	A 2D vector with *n* rows and *m* columns. *n* is the number of datapoints. *m* is the number of features. Every column represents a word that was found in the sentence after the training fragment.	Yes
Length of Span (LoS)	The relative character count of the propagandistic fragment	A vector containing real numbers between 0 and 1. The closer to 1 the bigger the span, with 1 the length of the longest span found.	No
Relative Capitation Frequency (RCF)	The percentage of capital letters relative to the total character count.	A 1D vector containing real numbers factors between 0 and 1, that represents the percentage.	Yes
Relative Punctuation Frequency (RPF)	The percentage of punctuation relative to total article size	A 1D vector containing real numbers factors between 0 and 1, that represents the percentage.	Yes

Average Sentence Length (WFPS)	Average number of words per sentence, normalized.	A 1D vector containing real numbers factors between 0 and 1. The closer to 1, the higher the average, with 1 the highest average found.	Yes
Article Size (TW)	The total number of words in the article, normalized.	A 1D-vector containing real numbers factors between 0 and 1. The closer to 1 the higher the relative total average, with 1 the highest word count of all the articles	Yes
Emotion Annotation on Article Level	Using a lexicon determines how positive or negative an article is (a negative word is -1, a positive word is +1).	A 1D vector containing real numbers factors between -1 and 1, with -1 being the most negative annotation and 1 the most positive.	Yes
Emotion Annotation on Span Level	Using a lexicon determines how positive or negative the span is (a negative word is -1, a positive word is +1).	A 1D vector containing real numbers factors between -1 and 1, with -1 being the most negative annotation and 1 the most positive. The score is normalized based on the number of words in the span (a span containing only negative words gets a -1).	No
Bias Level	Using a lexicon containing biased words to determine how biased an article is. (Every time a word is in the lexicon, then the bias level is +1.)	A 1D vector containing real numbers factors between 0 and 1. The closer to 1, the more biased the article is.	Yes
Bias Level on Span Level	Using a lexicon containing biased words to determine how biased a span is. (Every time a word is in the lexicon, then the bias level is +1.)	A 1D vector containing real numbers factors between 0 and 1. The closer to 1 the more biased the article is.	No

Subjectivity Level on Article level (Subj)	Using a lexicon containing words connected to subjectivity, to determine how subjective an article is. (Every time a word is in the lexicon, then subjectivity level +1.)	A 1D vector containing real numbers factors between 0 and 1. The closer to 1 the more subjective the article is.	Yes
Subjectivity Level on Span level (SubjSpan)	Using a lexicon containing words connected to subjectivity, to determine how subjective a fragment is. (Every time a word is in the lexicon, then subjectivity level +1.)	A 1D vector containing real numbers factors between 0 and 1. The closer to 1 the more subjective the article is.	Yes

Fig 4. Features with Explanation

The lexicon that I use for the *Emotion Annotation* feature is the NRC Sentiment Emotion lexicon (Mohammad & Turney, 2012). The source for the bias lexicon is Recasens, Danescu-Niculescu-Mizil, and Jurafsky (2013). The subjectivity lexicon originated from MPQA (Wilson, Wiebe, & Hoffmann, 2005). Further, I want to Note that the *Span-Countvectorizer* and *Length of Span* both are features that are not context related.

4.3 Baseline

I used two baselines for this experiment. The first one is the majority baseline. The majority baseline will take the label that was most frequent during the training phase and use that label to predict all the development/test data points. The most frequent label in the data is *Loaded Language*.

The second model uses Logistic Regression as the baseline. The logistic model is from the scikit-learn library. The penalty that the model uses for the parameters is ‘l2’ regularization. Regularization is necessary, because otherwise the model would overfit and that will cost performance. Overfitting is the phenomenon that a model performance decreases if there are too many variables. The model loses focus on the important variables. Thus, the penalty reduces the parameters and simplifies the model when necessary. The feature that I use for this baseline is the span length. The reason is that it is simple and fast. Also, it has some prediction value that is seen in previous work and can be a lower bound for the experiments.

4.4 Evaluation

The method of evaluation is precision and recall per class. Precision counts the number of predictions and gives the percentage of correct guesses. As an example, the model predicts a label 10 times and 6 of them are correct, then the precision for that label will be 0.60. Recall is the fraction of the total amount of relevant instances that were retrieved. Another example is that if the model predicts a label eight times, but there are ten of them in the data, then the recall would be 0.80. Another evaluation method that will be used in this experiment is a confusion matrix. A confusion matrix is a matrix that shows how the model predicts. Each row of the matrix is a label that the model predicted and each column is the correct label. An example:

	A is correct	B is correct
Predict A	4	1
Predict B	2	6

The matrix gives insight about the behaviour of the model. It tells us for instance which labels get confused. So, the model predicts A often, even though it is B. The difference between the A and B is apparently not that big.

The precision and recall are used to calculate the F_1 score. The F_1 score is a metric used to measure a test's accuracy, and it balances the use of precision and recall to do it. The F_1 score is often used in information retrieval, document classification, and query classification. The formula for the F_1 score is as follows: $F_1 = 2 * \frac{precision*recall}{precision+recall}$.

Scikit learn has a different way to calculate the F_1 score. The first way is called *micro-averaged*. It counts for every label the total true positives (true positive is; Guess: A Correct: A), false negatives (false negative is; Guess: B Correct: A) and false positives (false positive is; Guess: A Correct: B), and sum these up like it is only one label. So, it calculates every instance with the same weight. The second method is called *macro-averaged.* Now we calculate for each label the F_1-score and take the unweighted mean. The advantage of this is that if a certain label does not have a lot of data points, the score still will be impactful. An example of this is Label A the model guessed 100 times and 90 were correct. For label B it guessed 10 times and only 2 were correct. The micro score would be (90 + 2) / (100 + 10) = 0.83. The macro score would be ((90 / 100) + (2 / 10)) / 2 = 0.55 .

Results

Table 1 shows the results of all the contextual and non-contextual feature experiments. All the results were produced with the Logistic Regression model (except for the Majority Baseline). We can see from the table that the Countvectoriser is the feature with the biggest impact on the F_1-score. Most of the contextual features have almost no significant impact on the scores. The problem lies in the fact that the Countvectoriser becomes a vector of 5000+ dimensions after we trained the classifier. So, all the other features get drowned out by it. The contextual features consist mostly of 1-dimensional vectors. The predictions are then made using a matrix with 5000+ rows of the Countvectoriser plus the number of other (contextual) features used, which is less than ten.

After these findings I tried to use the Countvectoriser as a contextual feature. I chose a Countvectoriser, which is solely based on the text before the propagandistic fragment. It is called the pre-Countvectorizer. In addition, I also chose a Countvectoriser, which solely focuses on the text after the propagandistic fragment. This one is called the post-Countvectorizer. The consequence of these extra features is a lower performance. This can be a result of overfitting. The training set creates a vector with 20.000+ dimensions instead of 5000+ dimensions. It outperforms micro-averaged F_1-score of the majority baseline. From the confusion matrix of preCV plus postCV we can see that a reason for the low score is that there is confusion around the *Loaded Language* label. The model guessed 193 times correct and 166 times another label was guessed, even though the correct label was *Loaded Language*, and 323 times *Loaded Language* was guessed instead of the correct label. This aspect has the most room for score improvement.

	Features	F1 score (micro)	F1 score (macro)
	Logistic Regression Baseline (LoS feature)	0,135161	0,057701
	Majority Baseline	0,351616	0,023441
Base	Logistic Regression Baseline (CV feature)	0,457394	0,272843
	CV+LoS	0,459353	0,275312
	CV+RCF	0,439764	0,273579
	CV+RPF	0,457394	0,272843
	CV+WFPS	0,460333	0,276431
	CV+TW	0,441723	0,273369
	CV+Emotion Annotation (Article level)	0,450538	0,276293
	CV+Emotion Annotation (Span level)	0,457394	0,272843
	CV+Bias (Article level)	0,439764	0,273579
	CV+Bias (Span level)	0,454456	0,272121
	CV+LoS+WFPS	**0,463271**	0,278372
	CV+Subj	0,459353	**0,284549**
	CV+SubjSpan	0,451518	0,273108
	PreCV	0,191968	0,078585
	PostCV	0,199804	0,083652
	PreCV+PostCV	0,250734	0,080599
	CV+PreCV	0,400587	0,231157
	CV+PostCV	0,384916	0,213342
	CV+PreCV+PostCV	0,401567	0,217812
	all without CV+preCV+PostCV	0,053868	0,034611
	all without CV	0,177277	0,074512
	all	0,395691	0,222416

Tabel 1. Results of experiments per feature or feature combination

6. Discussion

The results were not completely unexpected. In the NLP4IF shared task of propaganda on fine-grained propaganda detection the highest F_1-score on the FLC task was 0.2488 (Da San Martino, et al., 2019). This task is more elaborate than the research I did, but it is according to expectations that the score did not improve much further. The reason is the difference in the models that were used. All the contestants used either BERT or LSTM, which have shown a better performance on these kinds of identification and classification tasks. Only one contestant out of eight had used contextual features in their model.

My findings of whether contextual features can be useful still has some validity. The preCounvectoriser and postCountvectoriser were not able to outperform the majority baseline. A possible cause would be overfitting, which would mean that the classifier needs more concise and focussed features. These features would have to go deeper than superficial levels of a text, in other words the syntax. They would need to focus more on the semantics in the sentences. I cannot completely reject my hypothesis that contextual features could have an important role, because the semantic features were lacking to capture deeper meanings and undertones of the words plus sentences.

Further, the training dataset was too imbalanced. Two labels, being *Loaded Language* and *Name Calling,* were almost half of all the data points. From the confusion matrix of the model with the Countvectorizer feature, we can see that the model guessed far too many times *Loaded Language*, even though it is a completely different label. In the following experiments, we should balance the data set a bit more. Especially because the more complex labels (*Thought-terminating Clichés, Black and White Fallacy, Appeal to authority,* etc.) need more data for the model to train. The precision of the model with these labels were poor. It averaged around 0.10. This is one of the aspects of the experiments that has lots of room for improvement. An example measure for improvement could be features that focus more on these more complex techniques or features that will be able to accurately discern whether the label is *Loaded Language/Name Calling* or one of the others. This could be a binary label, but this might be a temporary solution and scalable in the future.

Another limitation is Logistic Regression. We can see from the results of the NLP4IF shared task that BERT and LTSM have far better performances. In following experiments, it will be better to try out multiple algorithms, because there is a possibility that certain features mash better with certain algorithms.

For the next research, it would probably be better to experiment with fragment identification first. The model should be able to identify propagandistic fragments. It needs to be able discern from context which part of the sentence is propagandistic. There already has been research on sentence level identification (Hou & Chen, 2019), but they used BERT, which the inner workings has still not completely been explored and is not helpful for completely understanding the effect of contextual features. If we are able to understand the use of context better on the sentence level, then we will be able to create better methodology for the fragment level classification task.

7. Conclusion

I have focussed on the contextual features for fragment level propaganda classification in news articles. If we can make better use of these features, we will be able to greatly improve our performance in the future. It might have a smaller impact on the performance, than focussing solely on the fragment itself, but it still has impact. It is thus needed if we want to optimize the detection models. The research showed that simple features do not have enough impact. However there is a possibility that this is caused by the choices that were made for the model. In further research it is recommended to try more models instead of one. BERT and Long short term, which memory both have shown great promise. For the features we can delve more in the semantics. Try different lexicons or research the propagandistic properties more.

References

Allcott, H. & Gentzkow, M. (2017). Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives, 31(2)*, 211-236.

Barrón-Cedeño, A., Da San Martino, G., Jaradat, I., & Nakov, P. (2019). *Proppy: A System to Unmask Propaganda in Online News.* Qatar Computing Research Institute, HBKU, Qatar.

Da San Martino, G., Barrón-Cedeño, A., & Nakov, P. (2019). Findings of the NLP4IF-2019 Shared Task on Fine-Grained Propaganda Detection. *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda* (pp. 162-170). Hong Kong, China: Association for Computational Linguistics.

Da San Martino, G., Yu, S., Barrón-Cedeño, A., Petrov, R., & Nakov, P. (2019). Fine-Grained Analysis of Propaganda in News Article. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing* (pp. 5636-5646). Hong Kong, China: Association for Computational Linguistics.

Gomez-Rodriguez, M., Gummadi, K. P., & Schölkopf, B. (2014). Quantifying Information Overload in Social Media. *Eighth International AAAI Conference on Weblogs and Social Media*, (pp. 170-179).

Horne, B. D., Khedr, S., & Adah, S. (2018). Sampling the News Producers: A Large News and Feature Data Set for the Study. *International AAAI Conference on Web and Social Media.* Stanford, CA, USA.

Hou, W., & Chen, Y. (2019). Sentence-Level Propaganda Detection Using BERT with. *Proceedings of the 2nd Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda,* (pp. 83-86). Hong Kong, China: Association for Computational Linguistics.

Jowett, G. S., & O'Donnell, V. (2018). *Propaganda & Persuasion.* SAGE Publications.

Kiesel, J., Mestre, M., Shuukla, R., Vincent, E., Adineh, P., Corney, D., . . . Potthast, M. (2019). SemEval-2019 Task 4: Hyperpartisan News Detection. *Proceedings of the 13th International Workshop on Semantic Evaluation* (pp. 829-839). Minneapolis, Minnesota, USA: Association for Computational Linguistics.

Mapes Jr., N. J., White, A., Medury, R., & Dua, S. (2019). Divisive Language and Propaganda Detection using Multi-head. *Proceedings of the 2nd Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda* (pp. 103-106). Hong Kong, China: Association for Computational Linguistics.

Miller, C. R. (1939, February 20). How to Detect and Analyze Propaganda. Town Hall, Inc.

Mohammad, S. M., & Turney, P. D. (2012) Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, 29(3), 436-465.

Rashkin, H. C. E. Y. J. J., Volkova, S., & Choi, Y. (2017). Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (pp. 2931-2397). Copenhagen, Denmark: Association for Computational Linguistics.

Recasens M., Danescu-Niculescu-Mizil, C., & Jurafsky, D. (2013). Linguistic model for analyzing and detecting biased language

Silverman, C., Strapagiel, L., Shaban, H., Hall, E., & Singer-Vine, J. (2016, October 20). *Hyperpartisan Facebook Pages Are Publishing False And Misleading Information At An Alarming Rate*. retrieved from BuzzFeedNews: https://www.buzzfeednews.com/article/craigsilverman/partisan-fb-pages-analysis

Wilson, T., Wiebe, J., & Hoffmann P. (2005). Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP-2005)*.