NLPIITR at SemEval-2021 Task 6: RoBERTa Model with Data Augmentation for Persuasion Techniques Detection

Vansh Gupta
Indian Institute of Technology, Delhi
vansh.g0108@gmail.com

Raksha Sharma
Indian Institute of Technology, Roorkee
raksha.sharma@cs.iitr.ac.in

Abstract

This paper describes and examines different systems to address Task 6 of SemEval-2021: *Detection of Persuasion Techniques In Texts And Images, Subtask 1*. The task aims to build a model for identifying rhetorical and psychological techniques (such as causal oversimplification, name-calling, smear) in the textual content of a meme which is often used in a disinformation campaign to influence the users. The paper provides an extensive comparison among various machine learning systems as a solution to the task. We elaborate on the pre-processing of the text data in favor of the task and present ways to overcome the class imbalance. The results show that fine-tuning a RoBERTa model gave the best results with an F1-Micro score of 0.51 on the development set.

1 Introduction

The purpose of this task was to identify propaganda in electoral campaigns where memes are one of the most popular types of content used to carry out disinformation campaigns. They are most effective on social media platforms since they can easily reach many users. SemEval-2021 task 6 (Dimitrov et al., 2021) refers to propaganda whenever information is purposefully shaped to foster a predetermined plan. Propaganda uses psychological and rhetorical techniques to achieve its objectives. Such techniques include the use of logical fallacies and appealing to the emotions of the audience. Logical fallacies are usually hard to spot since the argumentation, at first sight, might seem correct and objective. However, a careful analysis shows that the conclusion cannot be drawn from the premise without the misuse of logical rules. Another set of techniques uses emotional language to induce the audience to agree with the speaker only based on the emotional bond that is being created, provoking the suspension of any rational analysis of the argumentation.

The Oxford dictionary defines a meme as 'An image, video, piece of text, *etc*., typically humorous in nature, that is copied and spread rapidly by internet users, often with slight variations'. They generally consist of an image superimposed with text. The role of the image in a deceptive meme is either to reinforce/complement a technique in the text or to convey one or more persuasion techniques. A total of 3 subtasks were defined: For subtasks 1 and 2, only the meme's textual content was given. The former was a multilabel classification problem while the latter was a multilabel sequence tagging problem, each with 20 labels of techniques. For subtask 3, both textual and visual content of the memes were given. The goal was to identify which of the techniques are used for a given text/image. In this paper, we describe a system to address the subtask 1 for which, a training dataset of 600 sentences with their labels was given. The test set consists of over 200 sentences. We evaluated the performance of all the models using the Micro and Macro F1 scores.

We make use of 13 different models to approach the problem at hand, experimenting with different data preprocessing techniques to finally document 23 sets of results. We found that strong results can be achieved by fine-tuning a pre-trained RoBERTa language model. However, we also discovered that some other models, when used as part of a voting classifier, gave decent results without requiring resources and time needed to train a RoBERTa model.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 talks about the methodology. Section 4 describes the dataset for the task. Section 5 provides the experimental setup. Section 6 presents the systems implemented to address the task. Section 7 shows

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 1061–1067
Bangkok, Thailand (online), August 5–6, 2021.

the results and Section 8 concludes the paper.

2 Related Work

The closest research to persuasion detection in Computer Science is perspective detection. Lin et al., (2006) discussed different types of machine-learning classifiers such as Naïve Bayes and Support Vector Machines for the perspective detection. Another topic, closely knitted with the detection of persuasion techniques is hate speech detection. Sood et al., (2012) worked on detecting personal insults, profanity, and user posts characterized by malicious intent. Xiang et al., (2012) focused on vulgar language and profanity-related offensive content. Xu et al., (2012) further look into jokingly formulated teasing in messages that represent bullying episodes.

Burnap and Williams, (2014) specifically looked into othering language, characterized by an 'us' and 'them' dichotomy in racist communication. Approaches to detecting hate speech on Twitter using convolutional neural networks and convolution-GRU-based deep neural networks are discussed in Gamback and Sikdar, (2017) and Zhang et al., (2018) respectively.

Persuasion technique detection is a multi-label classification problem. Many attempts have been made in the literature to make a multi-label classification system (Spyromitros et al., 2008; Elisseeff et al., 2001; Zhang and Zhou, 2006; Curram and Mingers, 1994; Huang, 1988; Crammer and Singer, 2003).

3 Method

In this paper, we make use of a one-versus-rest strategy to split a multi-label classification dataset into binary classification problems, splitting the data into one binary dataset for each class (Hastie et al., 1998), called 'pairwise coupling'. We present results with various approaches for the persuasion techniques detection in the text data. We make use of pre-defined language models and tune the hyper-parameter for best performance on the dataset. Soft voting classifier and hard voting classifier (Zhou et al., 2002) are deployed to use traditional machine learning models for the task. A soft voting classifier uses the average of probabilities for each class to classify the test instance, whereas, in hard voting, the classifier assigns the class that was voted by a majority of the classifiers. In our case, since we implemented our own voting classifier, we did not always go with the majority but experimented with different threshold values. In some cases, we gave more weightage to the models which gave more promising results. Various models, such as Logistic Regression, Naive Bayes, SVM, RNN, BiLSTM, BERT, naming a few, were implemented, which are discussed in the later sections. RoBERTa model performed the best for the persuasion techniques detection task on the text data (Section 7).

4 Dataset

In this paper, we provide systems to address subtask-1. The input data for subtask-1 is the text extracted from the memes. The training, the development, and the test sets for subtask-1 were distributed as JSON files.
An object of the JSON has the following type:

{
"id": "125"
"labels": [
"Loaded Language",
"Name calling/Labeling"
]
"text": "I HATE TRUMP \n\nMOST TERRORIST DO"
}

where

- id is the unique identifier of the example across all three tasks
- text is the textual content of the meme, as a single UTF-8 string.
- labels is a list of techniques used in the text (Written below). In this case, two techniques were spotted: Loaded Language and Name calling/Labeling.

Below are the technique names with their frequency in the training dataset.

- Appeal to authority: 13
- Appeal to fear/prejudice: 43
- Black-and-white Fallacy/Dictatorship: 18
- Causal Oversimplification: 27
- Doubt: 34
- Exaggeration/Minimisation: 52
- Flag-waving: 27
- Glittering generalities (Virtue): 32
- Loaded Language: 313
- Misrepresentation of Someone's Position: 3
- Name calling/Labeling: 188

Number of labels	Number of sentences
0	161
2	119
3	90
4	47
5	11
6	2
8	1

Table 1: Number of labels for number of sentences

- Obfuscation, Intentional Vagueness, Confusion: 4
- Presenting Irrelevant Data (Red Herring): 3
- Reductio ad hitlerum: 9
- Repetition: 8
- Slogans: 84
- Smears: 168
- Thought-terminating cliché: 20
- Whataboutism: 40
- Bandwagon: 2

Table 1 reports the number of sentences with multiple (or no) labels. For example, the first row suggests that there were 161 sentences in the dataset with no labels.

5 Experimental Setup

We train a wide range of different models for the task. As discussed in Section 3, we used both a soft voting classifier and a hard voting classifier on the traditional machine learning models for better performance. However, it should be noted that a soft voting classifier cannot be used in all the models, and accordingly, models were categorized. In order to assign different weights to different models, as per the individual performance, we used our own version of the hard voting classifier by scikit-learn (Pedregosa et al., 2011). Most of the models that we implemented support a multi-class classification where one assumes that each sample is assigned to one and only one label. However, since we were in a multi-label classification scenario, where we were required to assign a set of target labels, we wrapped these models in a OneVsRestClassifier. We converted the multi-label classification problem into a series of binary classification problems assuming that there was not any underlying correlation between any two labels and they were mutually exclusive. Below is the list of the machine learning and deep learning approaches that we implemented to build the systems to identify persuasive techniques.

- Logistic Regression (LR), a meta-model trained using the stochastic average gradient solver.
- Standard Naive Bayes (NB) and **Support Vector Machine (SVM)** from scikit-learn library in python.
- Linear Support Vector Classification (SVC) to return a *best fit* hyperplane that divides, or categorizes the data.
- Bernoulli Naive Bayes, a variant of Naive Bayes, using learning rate of 0.12.
- Ridge classifier with 0.01 as the tolerance for the stopping criteria.
- SGDClassifier (SDGC) with added $L2$ regularization to prevent overfitting and the hinge loss function.
- Passive Aggressive Classifier (PAC) with 50 as the cap for the number of iterations the model makes over the data.
- Random Forest Classifier (RFC) which creates a set of decision trees from a randomly selected subset of the training set.
- Multi-layer Perceptron (MLP) regressor with 1 hidden layer of size 8, followed by 2 of size 16 and finally another of size 4.
- Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997).
- Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018).
- A Robustly Optimized BERT Pretraining Approach (RoBERTa) (Liu et al., 2019).
- A Deep Neural Network model that used a vocabulary size of $10,000$; a batch size of 32; and was trained over 6 epochs. The system consisted of an embedding layer of size 128, followed by a $1D$ Convolution layer with 32 filters and *same* padding. Further, we used a max pooling layer of pool size 2 and an LSTM layer with 64 units and Dropout regularization of 0.2. Finally, 2 dense layers with 64 units and 1 unit respectively and activations *ReLU* and *sigmoid* were used.

Evaluation Measures: The given evaluation measure for the task was Micro-F1. We also report Macro-F1 in this paper as it is more favorable for multi-class classification problems.
F1-score is the harmonic mean of Precision and Recall
Precision: It is the fraction of relevant instances among all retrieved instances
Recall: It is the fraction of retrieved instances among all relevant instances
First, we independently find the F1 Score for each label, and then, to aggregate these F1-scores into a single F1-score, two ways can be used.

- Macro F1-Score: Simply average all the F1-Scores and calculate a mean F1-Score.
- Micro F1-Score: Instead of calculating each label's F1-Score, derive the F1-Score by calculating Precision and Recall by summing all the true positives, false positives, and false negatives of the system for different labels.

6 System Description

We split the training data into a training and validation set in the ratio $9:1$, *i.e.,* 90% of the corpus was used for training while the remaining 10% was part of the validation set. The preprocessing (Section 6.3) for all the models was the same and once the best model was found, it was retrained including the validation set for training as well. The trained model was then used to predict labels for the Development set. The model yielding the best accuracy on the earlier chosen validation set was used for the submission of the Development set. We followed a similar approach for the submission of the results in the task on the test set using RoBERTa model. We secured $15th$ rank in the task.

6.1 Initial Setup

The labels were in the form of a string under the column 'labels'. For easier manipulation, these were first one hot encoded into multiple labels using ',' as a separator. It should be noted that multiple columns for the same label (Owing to white space before the comma in a few cases) had to be merged.

6.2 Data Augmentation

We observed a lack of data along with an internal data bias. To overcome this issue, we incorporated different data augmentation techniques. While a few of the techniques we employed for data augmentation boosted the performance, many of them just confused the model and decreased the validation set accuracy. The techniques which were discarded for making no significant improvement on the results are as follows:

- Random Swap: Choose two words in the sentence at random, and swap their positions.
- Synonym Replacement: Choose a few words from the sentence at random, and replace them with their synonyms.
- Random Deletion: Randomly delete a word if a uniformly generated number between 0 and 1 is smaller than a pre-defined threshold.
- Random Insertion: Find a synonym of a word chosen at random from the sentence and insert it into a random position in the sentence.

However, *Back Translation* which is a classic data augmentation technique gave positive results. This method translates the text data to some language and then translates it back to the original language. This can help to generate textual data with different words while preserving the context of the text data. In our model, we made use of the Language translation API provided by Google Translate. We ran each text sentence through 5 different translations. Each series consisted of translation from English to another language of the similar scripts followed by another language that required transliteration and then back to English. We used this with 10 different languages (other than English) to create maximum diversity in the corpus. After we had generated a much larger corpus than before, there was still a serious problem of data bias in most labels since there were more zeroes (in one hot encoding) than ones. For that, apart from the original corpus, we only took those text samples out of the augmented data, in which that label was present. Compared to the previous training dataset, upon using this dataset, the performance (Micro F1 score) of the model on the development set rose for a few models while it fell for others. For the latter, we believe that the decrease in data-size was under-compensated by the decrease in the problem of bias.

6.3 Preprocessing

Since the corpus sentences were texts extracted from memes, we expected them to contain some patois and other contractions. Consequently, we started our preprocessing by substituting conjunctions with the complete words, and some of the more commonly used lingos with their literal and

System Name	Drop	Threshold	Aug F1-Micro	F1-Micro	Aug F1-Macro	F1-Macro
RoBERTa (Dev Set)	×		**0.50909**		**0.34419**	-
All models together	×	0	0.44118	0.38450	0.20520	0.18398
	×	1	0.40495	0.34520	0.18473	0.15812
	×	2	0.37919	0.30891	0.17666	0.12938
	×	3	0.35689	0.26981	0.16338	0.11112
	×	4	0.33835	0.26818	0.13557	0.10442
All models together	✓	0	0.34404	–	0.20039	–
	✓	1	0.39575	–	0.21937	–
	✓	2	0.41728	–	0.22840	–
	✓	3	0.43426	–	0.23603	–
	✓	4	0.44021	–	0.23809	–
Ridge Classifier	×	–	0.38983	0.29870	0.14469	0.12775
Rand Forest Clf	×	–	0.37209	0.17021	0.15921	0.07692
Naive Bayes Clf	×	–	0.35805	0.27189	0.11108	0.08185
Logit Regr Clf	×	–	0.35448	0.27865	0.12454	0.07897
SGDC Classifier	×	–	0.34752	0.32150	0.17489	0.14933
BNB Classifier	×	–	0.34520	0.26957	0.15900	0.11105
SVM Classifier	×	–	0.34409	0.29857	0.17228	0.13124
PAC Classifier	×	–	0.34295	0.30435	0.17311	0.15109
MLP Regressor	×	–	0.32975	0.28200	0.14570	0.14747
Deep Neural Network	×	0.2	0.31390	0.32819	0.14713	0.12470
	×	0.3	0.32049	0.32283	0.14894	0.12299
	×	0.4	0.32137	0.31855	0.14894	0.11849
	×	0.5	0.31360	0.31919	0.13993	0.11866

Table 2: Results with various systems

legal English counterparts.

After that, we proceeded by using custom made functions for subjecting the sentences to lowercasing and removing most punctuation marks or any kind of special character which do not include any valuable information for text classification.

Next, we take care of the stopwords, which are a set of frequently used words (often short) and their removal is critical because, in their absence, we can focus on the essential words to the context of the sentence more. Therefore, we remove all the stop-words present in the text sentences using the default set of stop-words that can be downloaded from the NLTK library (Bird et al., 2009).

Next, we moved to Lemmatization. Stemming is another widely used technique for a similar purpose where one chops off its inflections and keeps what hopefully represents the main essence of the word. However, we observed that lemmatization gave better results on the provided data. Instead of chopping words off (truncating the word), lemmatization relies on a linguistic knowledge base like *WordNet* (An extensive database of English which superficially resembles a thesaurus, in that it groups words based on their meanings) to obtain the correct base forms of words.

7 Results and Analysis

Table 2 presents the F1-Micro and F1-Macro obtained with various systems on the test data for the subtask-1 of Task-6 after training on both the augmented data and just the original base data. The scores for the models trained on the augmented data use the prefix *Aug* in the column name. The models with the *Drop* column marked X, were trained on the complete augmented data. In contrast, while training the ones containing a ✓, we took only those augmented sentences that contained the label we were training the model for. This was done to cope with the data bias problem as discussed in Section 6. In the machine learning models, the *Threshold* column contains an integer value which is the threshold for the hard voting classifier *i.e.*, the minimum number of models predicting a technique

to be associated with a sentence. However, in the case of the Deep Neural Network, the *Threshold* column contains the probability of the prediction of a particular technique.

The system 'All models together' deployed an ensemble of traditional machine learning models, as listed in the Section 5. From table 2, we unsurprisingly infer that *RoBERTa* gave the best results for the persuasion techniques classification on the text data. Even so, there are quite a few takeaway points one can conclude from the same.

Firstly, even though there is a visible gap between RoBERTa and the rest of the models, the ensemble of models was almost 200 times faster on a CPU compared to RoBERTa, which was trained on a GPU. This indicates that the former is a superior choice when quick NLP predictions are required using low-end systems.

Next, we observe the effect of tackling the data bias technique by choosing some specific augmented sentences for the training corpus. In 'All models together', even though a better result was reported by the system without the drop and the threshold value 0, overall, dropping the sentences gave better performance for different threshold values. One can note that the latter system even gave better F1-Macro score universally for all thresholds.

Finally, we adjudge a rather interesting observation. Most of the systems comprehensively work far better with the augmented data, but the Deep Neural Network, which contained 1.5 Million trainable parameters, showed a startling outcome. The score obtained after training on the base corpus was better than the one obtained after training on much larger augmented data. Repeated back-translation might be the reason that we fed the similar type of data instances to the system. Thus, the model was not able to learn enough, giving poor accuracy and faulty predictions. We didn't see the similar observation in traditional machine learning models or even MLP regressors because these models use fewer parameters.

8 Conclusion

Persuasion techniques detection is a multi-label classification problem. This paper presents and analyzes 13 machine learning and deep learning-based systems for persuasion techniques detection in the text data. Persuasion data has an extensive range of techniques as labels (20 in our case) with high-class imbalance. We observe that the data augmentation technique, *i.e.*, Back Translation, helps overcome class imbalance and produces more robust systems. Besides, we present the essential data preprocessing for the task. Results show that RoBERTa, a deep neural language model, outperformed other systems by a significant margin. Fine tuning the RoBERTa model on the training data captures the sensitive features for persuasion techniques identification.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Peter Burnap and Matthew Leighton Williams. 2014. Hate speech, machine classification and statistical modelling of information flows on twitter: Interpretation and communication for policy decision making.

Koby Crammer and Yoram Singer. 2003. A family of additive online algorithms for category ranking. *Journal of Machine Learning Research*, 3(Feb):1025–1058.

Stephen P Curram and John Mingers. 1994. Neural networks, decision tree induction and discriminant analysis: An empirical comparison. *Journal of the Operational Research Society*, 45(4):440–450.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dimiter Dimitrov, Bishr Bin Ali, Shaden Shaar, Firoj Alam, Fabrizio Silvestri, Hamed Firooz, Preslav Nakov, and Giovanni Da San Martino. 2021. Task 6 at semeval-2021: Detection of persuasion techniques in texts and images. In *Proceedings of the 15th International Workshop on Semantic Evaluation*, SemEval '21, Bangkok, Thailand.

André Elisseeff, Jason Weston, et al. 2001. A kernel method for multi-labelled classification. In *NIPS*, volume 14, pages 681–687.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online*, pages 85–90.

Trevor Hastie, Robert Tibshirani, et al. 1998. Classification by pairwise coupling. *Annals of statistics*, 26(2):451–471.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

William Y Huang. 1988. Neural net and traditional classifiers1 william y. huang and richard p. lippmann. In *Neural Information Processing Systems: Proceedings of a conference held in Denver, Colorado, November 1987*, volume 1, page 387. Springer Science & Business Media.

Wei-Hao Lin, Theresa Wilson, Janyce Wiebe, and Alexander Hauptmann. 2006. Which side are you on? In *Proceedings of the Tenth Conference on Computational Natural Language Learning - CoNLL-X '06*. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sara Owsley Sood, Elizabeth F Churchill, and Judd Antin. 2012. Automatic identification of personal insults on social news sites. *Journal of the American Society for Information Science and Technology*, 63(2):270–285.

Eleftherios Spyromitros, Grigorios Tsoumakas, and Ioannis Vlahavas. 2008. An empirical study of lazy multilabel classification algorithms. In *Hellenic conference on artificial intelligence*, pages 401–406. Springer.

Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1980–1984.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666.

Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, 18(10):1338–1351.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *European semantic web conference*, pages 745–760. Springer.

Zhi-Hua Zhou, Jianxin Wu, and Wei Tang. 2002. Ensembling neural networks: many could be better than all. *Artificial intelligence*, 137(1-2):239–263.