# University of Bucharest Team at Semeval-2022 Task4: Detection and Classification of Patronizing and Condescending Language

**Raluca Andreea Gînga, Bogdan Dobre,**
**Tudor-Andrei Dumitraşcu** and **Bogdan Radu Silviu Sielecki**
University of Bucharest
{gingaraluca, dobrebogdan98,
tudorandrei.dumitrascu, sieleckiradu}@gmail.com

## Abstract

This paper details our implementations for finding Patronizing and Condescending Language in texts, as part of the SemEval Workshop Task 4. We have used a variety of methods from simple machine learning algorithms applied on bag of words, all the way to BERT models, in order to solve the binary classification and the multi-label multi-class classification.

## 1 Introduction

The Patronizing and Condescending Language Detection Task (Pérez-Almendros et al., 2022) is based on the paper Don't Patronize Me! (P'erez-Almendros et al., 2020), which is an annotated Dataset with Patronizing and Condescending Language Towards Vulnerable Communities.

The aim of this task is to identify PCL, and to categorize the language used to express it, specifically when referring to communities identified as being vulnerable to unfair treatment in the media.

Participants were provided with sentences in context (paragraphs), extracted from news articles, in which one or several predefined vulnerable communities are mentioned. The challenge is divided into two subtasks.

1. Subtask 1: Binary classification. Given a paragraph, a system must predict whether or not it contains any form of PCL.

2. Subtask 2: Given a paragraph, a system must identify which PCL categories express the condescension. The PCL taxonomy was defined based on previous works on PCL (i.g. Unbalanced power relations, Shallow solution, Presupposition, Authority voice, Metaphor, Compassion, The poorer, the merrier. )

## 2 Background

The dataset used for this SemEval 2022 task was Don't Patronize Me! (P'erez-Almendros et al., 2020), which contains a suite of sentences that mention some vulnerable communities and published in media in a lot of English speaking countries. The paragraphs were manually annotated to show 1) whether the text contains any kind of PCL, and 2) if it contains PCL, what linguistic techniques (categories) are used to express the condescension.

The dataset for subtask 1 (binary classification) contained a number of 10.636 paragraphs and 2.792 instances were used for the categories classification subtask.

In Figure 1, it can be seen that for the first subtask, there are almost 1000 texts that contain PCL. This means that the dataset is highly imbalanced and needs to be addressed. The next three figures (2,3,4) display the distribution of the most common words, both for the full dataset and for those texts that contain / do not contain PCL.

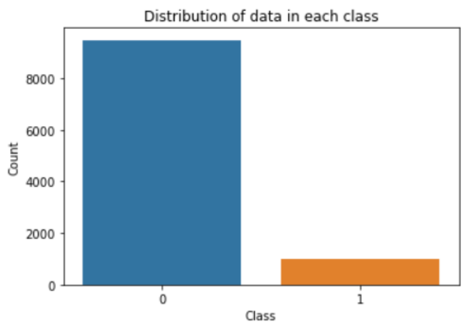


Figure 1: Classes Distribution for Binary Classification problem (Subtask 1)

For task 2, the paragraphs from task 1 are split according to the type of PCL speech category into sentences, resulting in 950 samples.

## 3 System Overview

1. Subtask 1 (Binary Classification)

   Because the dataset was very imbalanced, we tried different approaches in order to make it balanced:

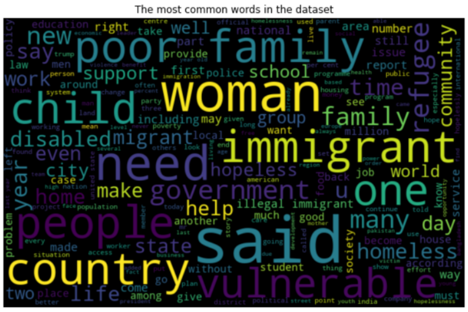


Figure 2: Most common words in the dataset (Subtask 1)


Figure 3: Most common words classified into PCL

- Adding a class weight to the models used. In this approach, we computed a metric in which we obtained a class weight according to the imbalance of the dataset. Through this method, we gave some different weights to both the majority and minority classes. This whole process had the purpose to penalize the miss classification made by the minority class by setting a higher class weight and at the same time, reducing the weight for the majority class.
- Using oversampling methods and special ensemble techniques. In this approach, we used methods like SMOTE (Synthetic Minority Over-sampling Technique), Adasyn (Adaptive Synthetic), SVM-SMOTE and self paced ensemble that performs strictly balanced undersampling in each iteration, being very efficient computationally.
- Augmenting the data. Because we notice so little data for label 1, we decided to collect hate speech datasets and add the positive texts into our dataset in order to balance the classes frequency, obtaining a total of 6372 from 795 initial texts with label 1. We will notice in the results section that this collection and generation of new dataset did not provide good results.


Figure 4: Most common words of texts that are not PCL

The dataset was a little bit preprocessed and split into two preprocessed types: lemmatized cleaned dataset and stemmed cleaned dataset. These two datasets were generated in order to make some comparison between those two techniques and to see which provided the best results.

To extract features from text, we have used Bag of Words, Tokenizer, Word2Vec and, finally, BertTokenizer.

We have also used a variety of models such as Neural Networks with 3 dense layers, Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) with 64 and 128 neurons with dropout as well, basic Machine Learning algorithms like Logistic Regression, Random Forest, Support Vector Machines as XGBoost. In the end, we decided to try BERT embeddings and a BERT classification model, BertForSequenceClassification, that contains a single linear classification layer on top and that provided the best results after all of the other approaches.

Another approach, called "Text shards" made use of the subtask related to multi-class classification as well. For an average text that contains PCL, only some small pieces of them are actually PCL and the rest of the text are not. The assumption is that this confuses the model, because a combination of PCL and non-PCL is labeled as PCL. To address this, the following approach is used:

- negative examples are left as they are
- each positive example is replaced with

the actual pieces of PCL inside it that we can get from the categories file
- the positive examples obtained this way are added with the negative examples to obtain a training dataset
- all the sentences are cleaned of characters that are not letters and the words in each sentence are lemmatized
- a Tensorflow Hub pretrained model called Universal Sentence Encoder is trained on it
- for each text that we want to predict, we first use the model on the whole text to get an initial label
- a window (of the size of the average length of a cleaned PCL fragment * 2) is slided through the text and the model is used to predict that particular substring. If it is labeled as PCL, then we consider the whole text as PCL.

2. Subtask 2 (Multi-label Multi-class Classification)

Considering the fact that the vocabulary of the is English is large, we have tried to leverage the power of pretrained language models. Therefore we have chosen 3 BERT-based models which were pretrained for hate speech detection and sentiment analysis. The BERT models also provided a tokenizer which split the sentences into tokens and appended the required tokens. The BERT models are used from the transformers library (Wolf et al., 2019).

- BERT (Devlin et al., 2018) Uncased
- BERT Multilingual Uncased
- BERT HateXplain (Mathew et al., 2020): This model was trained to classify text as Hate speech, Offensive or Normal. It was trained on Gab, Twitter and Humain Rationale;
- Distil BERT : This model is a version of Distilled BERT finetuned on the Twitter dataset;
- Distil BERT Multilingual Cased (Sanh et al., 2019)
- Distill RoBERTa : This model is a version of Distilled RoBERTa finetuned on the Twitter dataset;

In the paper describing the dataset (P'erez-Almendros et al., 2020), the authors group the categories into 3 General categories.

(a) The saviour: Unbalanced power relations and Shallow relations
(b) The Expert: Presupposition and Authority voice
(c) The Poet: Compassion, Metaphor and The poorer the merrier

From this idea, we tried to train the models to predict those 3 categories, and save the hidden features to a fixed latent space. Then these learned features can be used when training the model to predict the required 7 sub-classes.

Along with those BERT-based model, we also tried to implement models based on Word2Vec (Mikolov et al., 2013) trained on "Google News" and Machine Learning algorithms based on TF-IDF and BOW:

- LSTM Word2Vec Embeddings (Staudemeyer and Morris, 2019)
- biLSTM Word2Vec Embeddings (Huang et al., 2015)
- RNN Word2Vec Embeddings (Sherstinsky, 2020)
- SVM TF-IDF
- RandomForest TF-IDF

We also dabbled with the thought of training our own Word2Vec, in order to create a model specialized on hate speech. However we decided against this idea, due to the lack of usable datasets and the computational resources required for this task.

## 4 Results

1. Subtask 1 (Binary Classification)

(a) Deep Learning / Machine Learning for Imbalanced and Oversampled dataset

| | Lemmatized (F1_Score) | | | |
|---|---|---|---|---|
| Approach | Simple | SPE | SMOTE | SVMSMOTE |
| Neural Networks | 0.27 | - | 0.2823 | 0.3187 |
| Logistic Regression | 0.34 | - | 0.35 | 0.35 |
| Random Forest | 0.067 | 0.31 | 0.19 | 0.16 |
| Support Vector Machines | 0.27 | - | 0.10 | 0.14 |
| XGBoost | 0.15 | - | 0.23 | 0.24 |

| | Stemmed (F1_Score) | | | |
|---|---|---|---|---|
| Approach | Simple | SPE | SMOTE | SVMSMOTE |
| Neural Networks | 0.2698 | - | 0.289 | 0.3166 |
| Logistic Regression | 0.35 | - | 0.38 | 0.37 |
| Random Forest | 0.038 | 0.31 | 0.21 | 0.13 |
| Support Vector Machines | 0.27 | - | 0.14 | 0.20 |
| XGBoost | 0.17 | - | 0.23 | 0.24 |

(b) Tokenization of the lemmatized and stemmed dataset / Word2Vec + LSTM neural network

| | Lemmatized (F1_Score) | | Stemmed (F1_Score) | |
|---|---|---|---|---|
| Approach | Simple | Word2Vec | Simple | Word2Vec |
| LSTM (64 neurons) | 0.2693 | 0.2109 | 0.3213 | 0.2093 |
| LSTM (128 neurons) | 0.2317 | 0.2308 | 0.2789 | 0.2412 |

(c) Data augmentation

| | Augmented Data (F1_Score) |
|---|---|
| NNs with 3 layers | 0.2155 |
| Logistic Regression | 0.23 |
| U.S.E. + 2 dense layers | 0.2316 |

(d) BERT Transformers + BertForSequenceClassification

| | Unprocessed data (F1_Score) |
|---|---|
| Bert Tokenizer + Classifier | 0.5074 |

(e) Text shards

| | Lemmas F1_Score |
|---|---|
| Text shards | 0.3117 |

2. Subtask 2 (Multi-label Multi-class Classification)

(a) BERT models approach for classification across 7 classes. Table X shows that the model was able to learn only two of the classes.

| | F1 Score | | | |
|---|---|---|---|---|
| Class | Unb | Sha | Pre | Aut |
| BERT | 0.82 | 0.0 | 0.0 | 0.0 |
| DistilRoBERTa | 0.83 | 0.0 | 0.0 | 0.0 |
| DistilBERT | 0.82 | 0.0 | 0.0 | 0.0 |

| | F1 Score | | | |
|---|---|---|---|---|
| Class | Met | Com | Mer | Mean Score |
| BERT | 0.0 | 0.0 | 0.64 | 0.21 |
| DistilRoBERTa | 0.0 | 0.0 | 0.59 | 0.20 |
| DistilBERT | 0.66 | 0.08 | 0.0 | 0.34 |

(b) The general class approach is detailed in the following tables. It shows that the general classes were learned, but when using the pretrained models and fine-tuning on the specific classes, some of previously learned features are lost.

| | F1 Score | | |
|---|---|---|---|
| Main/Sub-classes | Expert | Aut | Pre |
| BERT | 0.44 | 0.0 | 0.0 |
| DistilRoBERTa | 0.54 | 0.0 | 0.0 |
| DistilBERT | 0.42 | 0.0 | 0.40 |
| DistilBERTMLC | 0.36 | 0.0 | 0.0 |

| | F1 Score | | |
|---|---|---|---|
| Main/Sub-classes | Saviour | Sha | Unb |
| BERT | 0.85 | 0.0 | 0.84 |
| DistilRoBERTa | 0.85 | 0.0 | 0.84 |
| DistilBERT | 0.75 | 0.0 | 0.81 |
| DistilBERTMLC | 0.86 | 0.0 | 0.83 |

| | F1 Score | | | |
|---|---|---|---|---|
| Main/Sub-classes | Poet | Com | Mer | Met |
| BERT | 0.69 | 0.0 | 0.0 | 0.59 |
| DistilRoBERTa | 0.69 | 0.11 | 0.0 | 0.65 |
| DistilBERT | 0.61 | 0.0 | 0.0 | 0.67 |
| DistilBERTMLC | 0.60 | 0.0 | 0.0 | 0.52 |

## 5 Conclusion

In the current project, it was solved the problem posed by SemEval 2022 Task 4: Patronizing and Condescending Language Detection. There were applied various methods, including the application of Word Embedings (Bag of Words, Word2Vec, BERT), tokenization, oversampling/undersampling of the datasets.

In the binary classification problem, the approach that gave the best result on the validation dataset was BERT transformers combined with BERT for Sequence Classification, obtaining 0.50 as f1_score.

In the multi classification multi label task, the number of labels proved to be a challenge. The results overall are low and the models are only able to learn only a few classes. The general class approach also proved to be inefficient.

Some recommendations for future work could be to have a better approach and introduce more linguistic insight in the approach.


## Acknowledgements

We would like to thank our supervisor Ana-Sabina Uban for the guidance and throughout the development of the project.


## References


Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatexplain: A benchmark dataset for explainable hate speech detection. *arXiv preprint arXiv:2012.10289*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Carla P'erez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2020. Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Alex Sherstinsky. 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Ralf C. Staudemeyer and Eric Rothstein Morris. 2019. Understanding LSTM – a tutorial into long short-term memory recurrent neural networks.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.