Patronizing language detection using NLP Transformer

Eder Tarifa
et1224@imperial.ac.uk

Paul Rhiner
pvr24@imperial.ac.uk

Abstract

In this report the task of identifying patronizing and condescending language (PCL) is tackled using various NLP techniques. The task is analyzed, different models are compared and data sampling and augmentation are explored. Ultimately, the final model achieves a F1 score of 0.557 on the official dev set.

1 Introduction

We developed a binary classification model to predict whether a given text contains patronizing and condescending language (PCL). PCL occurs when language conveys a sense of superiority over others while portraying them with compassion [Perez Almendros et al., 2020]. This type of language can be subtle and often unintentional. Although typically expressed with seemingly positive intentions, it reinforces stereotypes and power imbalances [Perez Almendros et al., 2020].

To study PCL directed at vulnerable communities, Perez Almendros et al., 2020 introduced the "Don't Patronize Me!" dataset, containing approximately 10,000 news article paragraphs annotated for PCL. Each paragraph is labeled with its English-speaking country and a keyword for the mentioned vulnerable community. This report develops a binary classifier for this task, optimizing F1 score via fine-tuning, sampling, and augmentation. We adjust parameters and evaluate performance across partitions based on input length.

2 Data Analysis

We use the dataset "Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities" for our analysis. This has six different features; par_id (a unique id for each one of the paragraphs in the corpus), art_id (the document id in the original NOW corpus), keyword (the search term used to retrieve texts about a target community), country_code (for 20 different countries), text (the paragraph containing the keyword) and label (patronizing or non-patronizing). Besides, the keywords used to create the dataset are "disabled", "homeless", "hopeless", "immigrant", "in-need", "migrant", "poor-families", "refugee", "vulnerable" and "women".

2.1 Qualitative assessment of the dataset

In this section we analyze our dataset to understand its complexity. The training set was annotated by three experts in communication, media, and train science. Two annotators labeled the dataset, with a third as a referee to resolve disagreements. This process ensures quality and highlights the topic's subjectivity.

We then examine examples to illustrate challenges and interpretations. In the first example, the sentence "No family shall be homeless under the watch of the municipal government here, said town Mayor Aldrin Garvida" is labeled non-patronizing. The phrase "under the watch of the municipal government" might suggest a paternalistic attitude—implying that the government assumes a parental role that casts families as incapable of self-governance. Similarly, the promise that "no family shall be homeless" could be seen as oversimplifying a complex issue. While this language may appear condescending by presenting the mayor as an all-knowing protector, an alternative reading is that it aims to instill confidence through strong language.

The next example involves the text "Arshad said that besides learning many new aspects of sports leadership he learnt how fast-developing nations were using sports as a tool of development and in this effort the disabled and the underprivileged were not left behind at any stage." labeled as patronizing, highlighting that "the disabled and the

underprivileged were not left behind at any stage." This phrasing can imply vulnerability and a paternalistic view, or simply report inclusive development strategies.
The final example is the headline: "Fast food employee who fed disabled man becomes internet sensation." It may reduce the disabled individual to a single characteristic, reinforcing stereotypes, or serve as a concise report of a viral moment.
In conclusion, our analysis shows that patronizing language is complex and subjective, with interpretations varying by perspectives, cultural contexts, and intent.

2.2 Quantitative assessment of the dataset,

To conduct a quantitative analysis of our dataset—a binary classification problem where only 10% of the data belong to the positive class—we examined various textual features to determine if any could indicate patronizing language. We focused on attributes such as text length (TL), number of unique words (NUW), keywords (K), number of uppercase characters (NUC), country code (CC), and a specific set of words (SW) including "us", "they", "must", and "help." For the K, CC, and SW features, we selected the candidate with the highest correlation—namely, "homeless," "gh" (Ghana), and "help" respectively. As shown in Table 1, even though these features exhibited a slightly higher correlation, none are strongly associated with patronizing language, thereby refuting our initial hypothesis.

tl	nuw	k	nuc	cc	sw
0.05	0.05	0.17	-0.01	0.14	0.11

Table 1: Correlation values between various text features and the patronizing language label.

3 Modelling Approach

3.1 Setup and Hyperparameter Tuning

To address this problem, we fine-tuned the Hugging Face model "deberta-v3-base." We selected this model because it is smaller (∼86M parameters) than alternatives like deberta-v3-large or roberta-large (+300M parameters), allowing it to be trained on a personal computer with limited resources and achieving a relatively fast training time (∼15 minutes per run). Additionally, DeBerta-V3 employs a disentangled attention mechanism that is more efficient than standard self-attention, enabling faster convergence and better generalization. We split the provided training data into separate training and development sets, using the official development set as our test set.
To adapt the model for our binary classification task, we added a classification layer and unfroze two layers of the base model. We set the batch size to 4 to ensure that all data would fit on our local GPU and initialized the learning rate at 10^{-5}. Although this lower learning rate may slow down training, we avoided increasing it further to prevent the model from failing to learn. We also set an initial weight decay of 0.01, as higher values risk undertraining the model. Although we initially planned for 20 training epochs, results in 1a indicated that the model began to overfit after 4 epochs, getting an initial F1 score of 52.12. Moreover, due to the model's erratic performance, we chose not to employ early stopping to avoid terminating training prematurely.
To further improve performance, we experimented by unfreezing different numbers of layers from the base model—intentionally overfitting first, then applying regularization. As shown in 1b, unfreezing four layers yielded the best performance. We also searched for the optimal learning rate and learning schedule; 1c and 1d demonstrate that a learning rate of 10^{-5} and a polynomial schedule provided the best results. Next, we addressed overfitting by testing higher weight decay values, ultimately finding that the initial weight decay of 0.01 performed better than all the results in 1e. We also examined whether converting all text to lowercase would affect performance, given that our model is cased, to test if our model is being able to take advantage of the semantic meaning of the capital letters. We can see the results in Table 2.

4 Further Improvements

Given the small size of our dataset, particularly the limited number of positive class examples (i.e., paragraphs featuring PCL), we hypothesized that implementing data sampling and augmentation strategies would significantly enhance our model's performance. Our reasoning is that these techniques help mitigate overfitting by providing the model with a more balanced and diverse set of training examples, ultimately leading to better generalization.

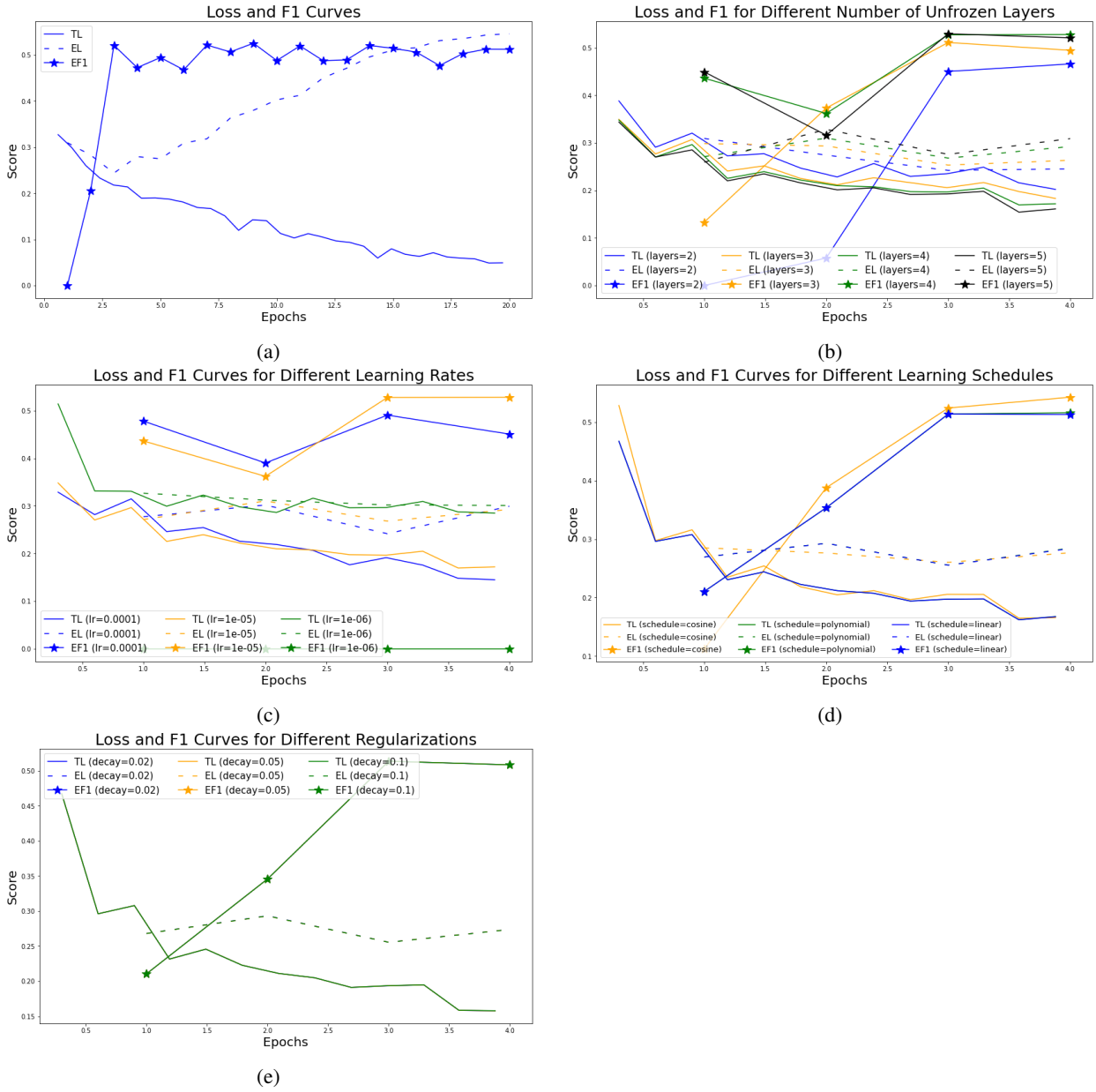

Figure 1: Hyper-parameter tuning results.

4.1 Data Sampling

To address class imbalance, we experimented with several data sampling techniques. First, we applied Random Over Sampling (ROS) to duplicate positive instances until both classes were balanced. Next, we utilized the Synthetic Minority Over-sampling Technique (SMOTE) to generate new positive samples by interpolating between existing ones. We also employed SMOTE Edited Nearest Neighbors (SMOTEENN), which combines SMOTE with a cleaning process that removes ambiguous instances between classes, thereby reducing noise. Additionally, we modified the loss function by weighting (WLF) the positive class five times more heavily than the negative class.

4.2 Data Augmentation

For data augmentation, two techniques were primarily explored: back-translation and synonym replacement.

Back-translation generates paraphrased versions of text by translating it to another language and then back to the original. The goal was to introduce variation while preserving subtle signs of PCL, helping the model generalize better. For our implementation, we used the `googletrans` API. Each paragraph was translated into a ran-

domly selected language (Spanish, German, or French) and then back to English, effectively doubling the training set with augmented versions of each paragraph.

Synonym replacement involves substituting words in the text with their synonyms to introduce lexical diversity without changing the core meaning or losing subtle PCL cues. Using WordNet from the `nltk` library, we identified synonyms for words in each paragraph and experimented with different substitution ratios. For each ratio, new data samples were created, again doubling the dataset size for each level of substitution. During training, both the original and augmented paragraphs were used, meaning the augmented data constituted 50% of the total training set for both back-translation and synonym replacement scenarios.

4.3 Categorical Features

We have also tested whether our model improves when adding the categorical features (CF) keywords and county code, since these features could have some relevant hidden information.

5 Results

After fine-tuning, our model achieved an F1 score of 54.24%, a considerable improvement over the initial configuration. Besides, applying SMOTE technique the model improved to a 56.9% F1 score. This underscores the critical importance of fine-tuning and applying further model improvement techniques, data sampling in this case, when developing a machine learning model. In contrast, lower-casing (LC) the text and incorporating categorical columns resulted in poorer performance, suggesting that the model benefits from the semantic significance of capitalization, while the categorical features do not add intrinsic value to improve predictions. Moreover, most data sampling approaches maintain performance, indicating that the model already assigns sufficient weight to the positive class—overemphasizing it further only harms performance. However, performance improved with SMOTE, suggesting that the synthetic data generated by this technique is beneficial to the model. The results are summarized in table 2

ros	smote	smoteenn	wlf	cf	lc
53.4	56.9	54.8	54.8	51.6	51.4

Table 2: Results: F1 score in percentage (on internal dev set) for each technique

Both data augmentation techniques did not improve generalization as can be seen in table 3. By inspecting the augmented data, we identified key reasons for the poor performance. When replacing words with synonyms, the process does not account for context, often altering the semantics of entire paragraphs. For back translation, the issue is more subtle. While it effectively paraphrases text, we observed that it generally euphemizes its input, leading to a loss of PCL in some paragraphs and resulting in mislabeled data. This effect was particularly pronounced in paragraphs that initially had a high level of PCL. We believe this euphemization is likely due to Google Translate's training, which may prioritize producing more neutral or less direct translations.

Syn. 5%	Syn. 10%	Syn. 20%	Trans.
51.6	52.2	51.6	53.7

Table 3: Results: F1 score in percentage (on internal dev set) for back translation and synonym replacement at different ratios

Thus, overall for our final model we use our fine-tuned hyper parameters with SMOTE and no data augmentation in training. Which achieved a F1-score of 56.9% on our internal dev set and a F1 score of 55.7% on the official dev set.

6 Comparison to two Baselines

We now compare to two baseline models. The first uses a Naive Bayes classifier with a Bag of Words (BoW) approach, optimizing feature selection via CountVectorizer. The second leverages DistilBERT embeddings fed into a Feedforward Neural Network (FFN) with LeakyReLU and Dropout, trained for 100 epochs using Adam and binary cross-entropy. Naive Bayes achieved a F1 score of 0.0297 with an accuracy of 90.63% whereas the second model achieved a F1 score of 0.3564 with an accuracy of 90.69%.

The BoW approach overwhelmingly predicts non-PCL, never classifying a non-PCL paragraph as PCL. This is likely due to the severe class imbalance and the fact that PCL operates at a conceptual level far beyond individual words. The top-weighted features are 'need' (-4.36), 'people' (-4.42), and 'poor' (-4.43). The following paragraph had the highest assigned probability among all misclassified cases: *Usually children of poor families remember [...] to success. It is the children of the rich [...] parents . The same chil-*

dren [...] fathers' sweat on the farm. This misclassification is primarily due to the repeated occurrence of the word children (ranked 6th highest in log probability) within a relatively short paragraph. This comparison reveals two key insights: first, identifying PCL is highly non-trivial - both baselines struggle, with BoW failing almost entirely and the embedding-based FFN yielding subpar results. Second, a more advanced language model with fine-tuning significantly improves performance. Even before hyperparameter tuning, our model already outperforms the embedding-based approach.

7 Analysis

This section analyzes the impact of the following metrics on our model's performance: level of patronizing content, paragraph length, and the keyword and country code associated with a paragraph. We partitioned our predictions on the official development set and calculated the F1 scores (positive class) and accuracies for all subsets.

For the level of patronizing content, we used the labels provided during the annotation process. The results can be seen in Table 4. The model performs significantly better at predicting extreme levels, which is expected. As the distance from the middle level (level 2) increases, there is greater consensus among the data labelers. Therefore, these paragraphs are more likely to exhibit clearer indicators—or the absence thereof—of PCL. Similarly, it makes sense that the F1 score increases with the patronizing level.

Patronizing Level	F1 Score	Accuracy
0	–	0.965
1	–	0.791
2	0.434	0.278
3	0.652	0.483
4	0.843	0.728

Table 4: F1 Scores and Accuracies for Different Patronizing Levels

For the length analysis, we considered both the paragraph length in words and in characters. We divided the development set into quartiles based on both word count and character length. The results are shown in Tables 5 and 6. There appears to be no significant impact on model performance. This aligns with our data analysis, which did not show any correlation between PCL and length.

Length in Characters	F1 Score	Accuracy
[0, 165]	0.578	0.934
[166, 228]	0.614	0.925
[229, 319]	0.575	0.930
[320, 1446]	0.493	0.862

Table 5: F1 Scores and Accuracies for Different Paragraph Lengths

Length in words	F1 Score	Accuracy
[0, 30]	0.585	0.939
[31, 41]	0.606	0.920
[42, 58]	0.593	0.929
[59, 272]	0.482	0.860

Table 6: F1 Scores and Accuracies for different word counts

The results for keywords and country codes are shown in Tables 8 and 7 (appendix). While country codes show little impact on performance, the keywords *women* and *refugee* stand out as outliers, with F1 scores of 0.2 and 0.32, respectively. This may suggest that texts patronizing these groups are generally more subtle, making them harder for the model to detect. The small sample size for the *women* keyword could also partly explain this result.

8 Conclusion

In this report, we developed a binary classifier to identify PCL using the DeBERTa-v3-base model, achieving an F1 score of 55.7% on the official development set. The model performed better on extreme levels of PCL, where the language is more clearly condescending, while text length, keywords, and country codes had little impact on performance. Data augmentation techniques like back-translation and synonym replacement did not improve results, but SMOTE helped address class imbalance. Fine-tuning hyperparameters was essential for achieving the best performance.

As a next step, we suggest fine-tuning larger and more advanced models to better capture the nuanced nature of PCL. Additionally, back-translation could be revisited with a focus on reducing euphemization, potentially leading to more successful augmentation. Finally, creating a larger dataset could significantly improve performance, as overfitting was a major challenge in our experiments.

A Appendices

References

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In Donia Scott, Nuria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.518. URL https://aclanthology.org/2020.coling-main.518/.

A.1 Additional tables

Country Code	F1 Score	Accuracy
au	0.461538	0.928571
bd	0.571429	0.941748
ca	0.428571	0.932203
gb	0.600000	0.937008
gh	0.666667	0.916667
hk	0.666667	0.935484
ie	0.588235	0.937500
in	0.428571	0.923077
jm	0.384615	0.846154
ke	0.606061	0.885965
lk	0.545455	0.882353
my	0.666667	0.939655
ng	0.551724	0.879630
nz	0.666667	0.940594
ph	0.516129	0.857143
pk	0.384615	0.853211
sg	0.666667	0.968421
tz	0.600000	0.914894
us	0.500000	0.894737
za	0.695652	0.936364

Table 7: F1 Scores and Accuracies for different countries

Keyword	F1 Score	Accuracy
disabled	0.500	0.928
homeless	0.594	0.877
hopeless	0.475	0.857
immigrant	0.571	0.972
in-need	0.747	0.916
migrant	0.800	0.990
poor-families	0.541	0.821
refugee	0.320	0.910
vulnerable	0.590	0.914
women	0.200	0.931

Table 8: F1 Scores and Accuracies for different keywords

A.2 Git Repository

This is the link to our GitHub repository.