NLP Coursework | Gitlab repository

Alejandro Ayuso García
CID: 00940554
aa11414@ic.ac.uk

Alan Picucci
CID: 02482490
ap3423@ic.ac.uk

Ivaylo Stoyanov
CID: 02515358
iis23@ic.ac.uk

Abstract

This report presents a novel binary classification model trained on the Don't Patronize Me! dataset to identify patronizing and condescending language directed at vulnerable communities. Achieving an F1-score of 0.56 on the official dev set, the model surpasses RoBERTa-base baseline model's 0.48. It utilizes an ensemble approach, averaging predictions from the top three model configurations found through Bayesian hyperparameter optimization. Built upon a pre-trained RoBERTa-base cased language model, the proposed model integrates insights gleaned from exploring multiple architectures as well as data sampling and data augmentation techniques.

1 Introduction

Patronizing and condescending language (PCL), is a form of discourse that asserts superiority over others and poses a subtle yet significant challenge in Natural Language Processing (NLP). This report addresses the task of developing a binary classification model to detect PCL in text. Our goal is to surpass the RoBERTa-base baseline F1-score of 0.48 on the official dev set and 0.49 on the official test set of the Don't Patronize Me! (DPM) dataset (Perez Almendros et al., 2020).

The DPM dataset contains over 10,000 paragraphs from English language news stories across 20 countries covering 10 vulnerable groups. The dataset has been annotated to indicate the presence of PCL at the text span level.

2 Data Analysis

2.1 Class Labels Analysis

The DPM training set consists of 10,469 paragraphs, where the vast majority (91%) do not contain PCL. Figure 1 shows that annotators largely agreed on the presence of PCL in paragraphs, assigning the same score in 87% of cases (labels 0, 2 & 4). However, despite the consistent criterion, a significant proportion (15%) of paragraphs posed ambiguity regarding PCL presence (labels 1, 2 & 3), underscoring the complexity and subjectivity of the task.

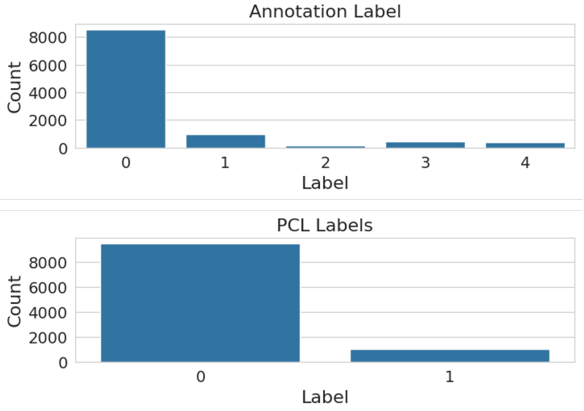


Figure 1: Frequency of annotator score and PCL labels

The average paragraph length is 49.7 tokens or 268 characters, with PCL texts being longer by 12% in tokens and 9% in characters (Table 4, Appendix). There is a weak correlation between PCL presence and both tokens (0.06) and characters (0.04), indicating longer or more complex PCL texts. Analysis of the 20 most frequent unique tokens indicates that PCL texts often use abstract, emotionally charged words like ”help”, ”life”, or ”hopeless”, whereas non-PCL texts typically feature concrete terms such as ”government”, ”year”, ”immigrants”. Country-wise, Ghana, Nigeria, and the Philippines show the highest PCL frequencies (12-14%), while Hong Kong, Australia, and Singapore have the lowest (6-7%) (Table 5, Appendix). By keyword, the homeless, poor families, and those in need are most associated with PCL (16-17%), contrasting with lower PCL incidences in texts about immigrants, migrants, and women (3-5%) (Table 6, Appendix).

2.2 Qualitative Assessment

Detecting PCL poses significant challenges due to its varying degrees of subtlety and context dependence, often leading to multiple interpretations. The DPM dataset encompasses a wide spectrum of PCL content. For instance, some texts exhibit no PCL, such as "[. . .] Two of the spaces would be reserved for disabled persons and there would be five P30 spaces and eight P60 ones", while others, like "I suddenly had the urge to look for a Filipino family living on the streets and give them a helping hand[. . .]", clearly do. Borderline cases with annotator scores of 2 highlight the complexity of the task. Phrases like "Hollywood star Leo Di Caprio urges help for reuniting immigrant children with their families" may be perceived as genuine efforts to aid vulnerable groups or as attempts to portray someone as a savior, illustrating the challenge in determining the level of PCL.

Furthermore, contextual factors, including cultural variations and personal perceptions, heavily influence the perceived level of PCL. For instance, this text from Ghana "[...] the conference was organized to help young ladies uncover and successfully accomplish their dreams and become responsible women in the future" might receive a low PCL rating from a local annotator but could be viewed differently by someone with a distinct sociocultural background.

Additionally, rhetorical devices like sarcasm further complicate PCL detection, as they can obscure the true tone of a message. Challenging even for humans due to the absence of non-verbal cues, language models (LMs) often struggle with such devices. A statement like "[. . .] He is preoccupied with summoning to court people who do not write letters to Switzerland ... That is, of course, much more important than safeguarding a poor child from a mad mob" may not contain PCL but could easily be misinterpreted by a classification model due to the use of terms like "poor child".

3 Modelling

3.1 Model Description

Our custom model incorporates the pre-trained RoBERTa-base cased model and a fully-connected layer, utilizing RoBERTa's (Liu et al., 2019) encoding of the [CLS] token for classification. Leveraging the case-sensitivity of the RoBERTa model proves advantageous for PCL identification, as capitalization can convey tone or intent. To facilitate hyperparameter optimization, we prioritize flexibility in the custom model. For instance, instead of choosing between freezing or not the weights of RoBERTa during training, we use a hyperparameter to determine the number of layers to freeze, starting from the bottom embedding layer. We preserve the bottom layers, which learn general language aspects, while allowing more layers to unfreeze towards the top, enhancing task-specific performance. Notably, freezing all RoBERTa layers often resulted in an F1-score of 0, as the model only predicted a lack of PCL for each sentence. Therefore, we ensure at least the top two layers remain unfrozen.

To assess each hyperparameter configuration's performance, we employ a consistent 75%/25% split of the official training set into internal training and validation sets throughout our hyperparameter search.

3.2 Model Improvements

3.2.1 Data Sampling

Data sampling techniques were explored to address the class imbalance in the DPM dataset:

Random oversampling (RO): duplicates random samples from the minority class to match the majority class' size, balancing the training dataset.

Random undersampling (RU): removes random samples from the majority class to match the minority class' size, achieving dataset balance.

Tomek links (TL): utilizes Sentence-BERT(Reimers and Gurevych, 2019) with a distilroberta-base model to generate text embeddings of the input paragraphs, removing samples from different classes that are closest neighbors, eliminating noisy or hard-to-classify instances.

Edited Nearest Neighbours (ENN): uses Sentence-BERT embeddings like TL, but with a 3-Nearest Neighbour algorithm to remove instances where most neighbors belong to a different class, refining discrimination boundaries.

Various technique combinations were tested with a RoBERTa-base model trained for 10 epochs on the internal training set and validating on the internal dev set. Table 1 summarizes the F1-scores achieved and compares them with two RoBERTa baselines: performance with no pre-processing (1) and with downsampling of the majority class to a 1:2 ratio of PCL to non-PCL instances in line with the original RoBERTa-base baseline (2).

Technique	F1-score
RO	**0.542**
RU	0.474
TL	0.518
TL then RO	0.501
ENN	0.531
ENN then RO	0.509
RO then ENN	0.520
Baseline (1)	0.515
Baseline (2)	0.509

Table 1: F1-scores attained on the internal dev set with different data sampling techniques used with a RoBERTa base model trained for 10 epochs

RO outperformed other techniques, achieving an F1-score of 0.542, owing to its ability to generate a larger, balanced training dataset. Since RO achieved the highest F1-score, it was implemented in the final model. Conversely, RU yielded poor results due to the small dataset produced, hindering the benefits of achieving class balance. Although not matching RO's F1-score, TL, ENN, and RO then ENN showed modest performance enhancements compared to the baselines, indicating improved learning through noise elimination.

Technique sequence influenced performance, with RO followed by ENN outperforming the reverse order, likely due to differing dataset sizes of 11,369 and 10,856 samples, respectively.

3.2.2 Data Augmentation

We experimented with back translation as a data augmentation technique, utilizing Opus MT's open Neural Machine Translation models (Tiedemann and Thottingal, 2020). Our approach involved augmenting the internal train set by translating texts from English to an intermediate language and then back to English. This technique aimed to balance classes, akin to RO, but introducing variety to replicated data through noise, which we expected would lead to better results. We evaluated the impact of back translation on the internal dev set using a RoBERTa-base model across five languages (German, French, Spanish, Chinese, and Italian), and a Mixed approach. The Mixed approach, involving randomly sampling one of the five languages for back translation, was implemented as we observed that back translating in the same language multiple times yielded identical sentences. Through language sampling, we aimed to inject the desired variability needed to generate unique entries in the training set.

The Mixed approach led to the best results, achieving an F1-score of 0.528, which was surprisingly inferior to the performance of RO (Table 7, Appendix). This discrepancy may be attributed to a decrease in training data quality, which negatively impacted performance. Given that RO yielded superior results, we opted to utilize this method in our final model configuration.

3.2.3 Multi-class classification

We employed a RoBERTa-base multi-class classification model to investigate potential performance enhancements by learning annotator scores and converting them to binary labels. Binary and multi-class classification models were trained for 10 epochs under two different scenarios, with and without downsampling. An alternative downsampling approach achieving a 1:1 ratio of 0 to 1 annotator score instances was used for the multi-class classifier instead of the baseline downsampling used for the binary classifier.

The multi-class classifier, due to the increased granularity introduced by annotator scores, proved more sensitive to class imbalance (Table 8, Appendix). While the smallest class in the binary model's training set (label 1) accounts for 33% of the total, in the multi-class' training set, the smallest class (label 2) comprises only 1%. Balancing classes significantly improved the multi-class classifier's performance, surpassing the binary classifier by +0.05 F1-score. This improvement can be attributed to the multi-class classifier's access to more granular data, enabling better learning of dataset nuances and greater flexibility in defining decision boundaries.

Due to the performance improvement observed, the use of multi-class or binary classification was introduced as a parameter in the second stage of the hyperparameter search.

3.2.4 Hyperparameter Search

We optimized our model using a Bayesian hyperparameter search, preferring its targeted approach over grid or random search. This method efficiently navigates complex parameter spaces by learning from previous results. Our search, conducted in two phases using Weights & Biases, first adjusted epochs, learning rate, batch size, dropout, weight decay, scheduler, and frozen layers (Table 9, Appendix). Initially, we mirrored the RoBERTa baseline by downsampling our dataset. After evaluating 100 setups (Figures 2-3, Appendix), the top F1-score of 56.18 on the internal dev set was obtained using a configuration with learning rate 5e-5, batch size 32, 8 frozen layers, and cosine scheduler (Table 10, Appendix). We used early stopping and achieved this score after 5 training epochs.

In the second phase, we applied the best data sampling method (RO), refined parameter ranges, and introduced new variables: inclusion of multi-class labels, an added 256-neuron linear layer with ReLU activation, and layer-wise learning rate decay with variable group numbers. Layer-wise learning rate decay (Howard and Ruder, 2018) is a successful technique found in the literature, which we applied by dividing the RoBERTa layers into groups and applying decreasing learning rates from top to bottom layers. This technique, which can be seen as orthogonal to the variable freezing of layers from our initial search, was combined with the latter in anticipation of improved results. In particular, we split the layers into G groups, with group 1 including the embedding layer at the bottom, and group G at the top, including the linear layers. At time step t, we compute the learning rate α_t^g of the layers in group $g \in \{1, ..., G\}$ as:

$$\alpha_t^g = \gamma \alpha_t^{g+1} = \gamma^{G-g} \alpha_t^G \tag{1}$$

The final learning rate for each group was determined by a decay rate γ and the initial learning rate α_0^G, both set as hyperparameters. The number of groups, G, also a hyperparameter, was chosen from factors of 12, aligning with RoBERTa's layer count. After evaluating another 100 configurations (Figures 4-5, Appendix), the best F1-score on the internal dev set was achieved by model M1 (0.601), closely followed by M2 and M3 (Table 2).

3.2.5 Ensembling

Our final model was obtained by ensembling the predictions of models M1, M2, and M3 found during the hyperparameter search. We considered soft and hard voting, and ultimately opted for the former option: we average the predicted probabilities of the three models and predict the label with the highest average probability. The key advantage of soft voting in our ensemble approach is its ability to incorporate the confidence levels of individual models into its predictions. This feature is particularly advantageous if one model excels at detecting a type of PCL that others struggle to identify. The ensemble achieved an F1-score of 0.601 on the internal dev set.

Hyperparameter	**M1**	**M2**	**M3**
Number of epochs	20	20	20
Learning rate	1e-4	1e-4	5e-5
Batch size	64	64	64
Frozen layers	8	0	8
Dropout rate	0.1	0	0.1
Weight decay	0.01	0.01	0
Scheduler	Cosine	Cosine	Linear
Layer-wise decay	0.95	0.8	0.9
Layer groups	1	12	12
Multi-class labels	False	False	True
Extra linear layer	True	False	True

Table 2: Best-performing model configurations from stage 2 of the hyperparameter search

3.3 Model Performance

We evaluated our final classification model against three baseline models: binary bag of words (BoW), count BoW, and BERT-base (cased).

BoW models were trained on the official train set with the majority PCL class downsampled to a 2:1 ratio of majority to minority sample count. Minimal preprocessing was applied: converting text to lower case, tokenization, and removing stopwords and punctuation. Both BoW models use one-hot encoded vector embeddings of size equal to the vocabulary of the training set to represent input text. Each feature is the count (with count BoW) or the presence (with binary BoW) of a different word in a given input text.

The BERT-base model was trained for 1 epoch on the same official train set.

Our final ensemble model achieved 0.562 F1-score on the official dev set, outperforming M1, M2, M3 and all baseline models (Table 3). The performance gap with respect to the worst and best performing baseline models, count BoW and RoBERTa-base, is 0.241 and 0.082, respectively.

Model	F1-score
Final Ensemble Model	**0.562**
M1	0.548
M2	0.556
M3	0.559
Binary BoW	0.347
Count BoW	0.321
BERT-base cased	0.476
RoBERTa-base cased	0.480

Table 3: F1-score of final and baseline models evaluated on the official dev set

Examining a false positive of the count BoW model reveals its limitations: ”The complaint says the victims should be treated properly and the government should announce compensation for them as they mostly come from poor families”. Count BoW ignores context and solely looks at word frequency. Analysing the top 10 most common tokens in misclassified samples that are not present in the top 10 of correctly classified samples produces three tokens: ”families”, ”poor” and ”children”. These words are misinterpretable as they have a high emotional charge but are also used in formal, sober communication. Count BoW probably identified ”poor” and ”families” in the sample, and mistakenly assumed it indicated PCL.

4 Analysis

For the analysis of our final model's performance we considered how it was affected by higher levels of PCL, input length and data categories.

4.1 Level of Patronising Content

Our model is better at predicting examples with a higher level of patronising content. The high recall rates for labels '3' and '4' are contrasted with significantly lower recall for label '2' (Figure 6, Appendix). These outcomes are anticipated for two main reasons. Firstly, since label '2' texts are the least common in our training set (Figure 1), it is natural for the model performance in this class to be inferior. By having more training samples for other classes the model can learn to classify them better. Moreover, texts with label '2' likely possess inherent ambiguity, since a text is assigned this label only if both human annotators deem it ”borderline PCL”. As we noted in the Qualitative Assessment section, this classification is highly subjective and since label 2 marks the boundary between PCL and non-PCL texts, it is natural for it to display more sensitivity.

4.2 Input Sequence Length

The model's performance remains relatively stable across different input sequence lengths. Slightly more variation is observed when using character length compared to token count (Figures 7-8, Appendix). The F1-score peaks in the 600-800 character range, but dips in the neighboring 400-600 character range. Similarly, the model performs best for longer texts in the 120-150 token range, and underperforms in the 60-90 token range. The truncation of token counts beyond 128 may impact these results, but this effect is likely minimal given the rarity of token counts above 128 (Figure 9, Appendix). Notably, the fact that no significant drop in performance is observed for any sequence length is an indication of model robustness.

4.3 Data Categories

Our model's performance is significantly influenced by the data categories. The F1-score for the ”in-need” category significantly exceeds that of the ”women” and ”immigrant” categories by more than threefold (Figure 10, Appendix). This indicates that the model is less proficient at accurately detecting PCL concerning certain communities. This pattern could be attributed to the different amount of training data available, since ”in-need” has one of the highest incidences of PCL in texts, while ”immigrant” and ”women” are among the categories with lowest PCL incidences (Figure 11, Appendix). Moreover, the varying levels of linguistic complexity associated with different communities could be influencing these results. This includes group-specific idiomatic expressions and unique jargon, which might affect the model's ability to detect PCL.

5 Conclusion

In this study, we presented a novel ensemble model for PCL detection, achieving an F1-score of 0.56 on the official dev set, outperforming the RoBERTa-base baseline score of 0.48. We explored data sampling, data augmentation, and multiclass classification to improve performance. We found that RO with no data augmentation offered the best performance, with multiclass labels potentially enhancing it further. Bayesian hyperparameter search revealed multiple high performing configurations, that when combined into an ensemble model achieved a higher F1-score. The final model's performance improved with higher PCL levels, and is sensitive to text category.

Future work could explore the impact on performance of incorporating additional categorical data, such as country or keyword information; new combinations of data sampling and data augmentation techniques, such as back translation followed by ENN; trying data augmentation techniques based on large language models; or the implementation of more complex architectures with the RoBERTa-base cased model.

References

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. 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.

Carla Perez 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, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

6 Appendix

6.1 Appendix A: Data Analysis

		non-PCL	PCL
Tokens	mean	49.1	55.2
	st dev	29.2	33.2
Lean Tokens	mean	26.5	28.4
	st dev	15.4	16.8
Characters	mean	265	289
	st dev	158	176
Lean characters	mean	164	171
	st dev	98	104

Table 4: Number of tokens and characters in the DPM dataset. Lean refers to counts after stopwords and punctuation is removed

Country	**Score (%)**
Ghana	14.3
Nigeria	13.4
Philippines	12.3
Jamaica	11.6
South Africa	10.6
UK	10.6
Tanzania	9.9
Pakistan	9.7
Sri Lanka	9.7
Ireland	9.7
New Zealand	9.1
Canada	8.7
Bangladesh	8.6
Kenya	8.3
United States	8.1
Malaysia	7.9
India	7.4
Singapore	7.1
Australia	6.8
Hong Kong	5.9

Table 5: Percentage of texts in the train set containing PCL broken down by country

Keyword	Incidence (%)
homeless	16.5
poor-families	16.5
in-need	16.3
hopeless	12.3
refugee	8.1
disabled	7.9
vulnerable	7.4
women	4.9
migrant	3.3
immigrant	2.8

Table 6: Percentage of texts in the train set containing PCL broken down by keyword

6.2 Appendix B: Data Augmentation

Language	F1-score
German	0.514
French	0.512
Spanish	0.517
Chinese	0.506
Italian	0.501
Mixed	0.528
Baseline (1)	0.515
Baseline (2)	0.509

Table 7: F1-scores attained on the internal dev set with different data augmentation techniques used with a RoBERTa base model trained for 10 epochs compared with two RoBERTa baselines, with no pre-processing (1) and with downsampling in line with the original RoBERTa-base baseline (2).

6.3 Appendix C: Multi-class Classification

		F1-Score
No downsampling	Binary	0.515
	Multi-class	0.000
Downsampling	Binary	0.509
	Multi-class	0.561

Table 8: F1-score achieved on the internal dev set with a RoBERTa-base binary classifier and a RoBERTa-base multi-class classifier to predict annotator score followed by conversion to PCL label

6.4 Appendix D: Hyperparameter Search

Hyperparameter	Values
Number of epochs	3, 5, 10
Learning rate	1e-4, 5e-4, 1e-5, 5e-5
Batch size	16, 32, 64
Frozen layers	0, 1, 4, 8, 10
Dropout rate	0, 0.1, 0.3, 0.5
Weight decay	0, 0.01, 0.001, 0.0001
Scheduler	Linear, Cosine
Layer-wise decay	0.8, 0.85, 0.9, 0.95, 0.99
Layer groups	1, 2, 3, 4, 6, 12
Multiclass labels	True, False
Extra linear layer	True, False

Table 9: Hyperparameter search space

Hyperparameter	Value
Number of epochs	10
Learning rate	5e-5
Batch size	32
Frozen layers	8
Dropout rate	0
Weight decay	0
Scheduler	Cosine

Table 10: Stage 1 Best Model Configuration

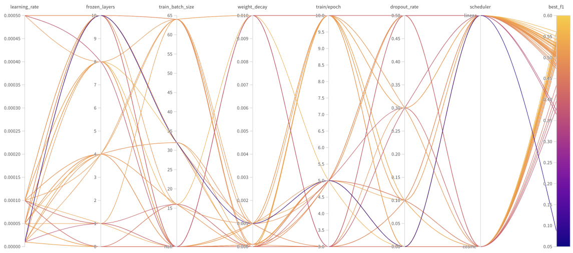
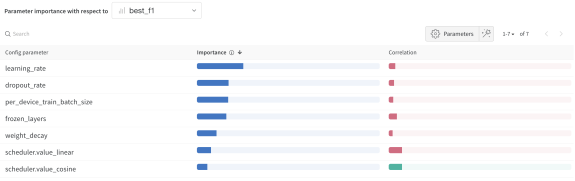


Figure 2: Parallel coordinates plot linking hyperparameter configurations from Stage 1 to the best F1-score achieved during training


Parameter importance with respect to best_f1

Config parameter	Importance	Correlation
learning_rate		
dropout_rate		
per_device_train_batch_size		
frozen_layers		
weight_decay		
scheduler.value_linear		
scheduler.value_cosine		

Figure 3: Parameter importance and correlation with the best F1-score achieved during Stage 1

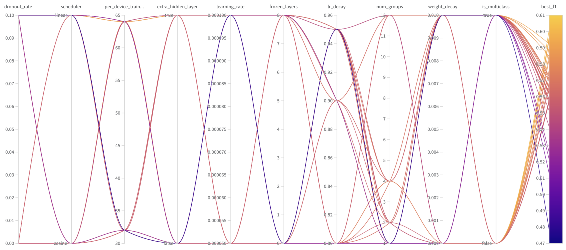
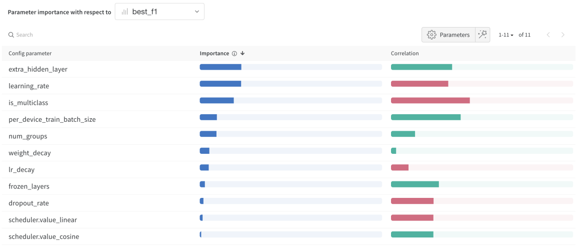


Figure 4: Parallel coordinates plot linking hyperparameter configurations from Stage 2 to the best F1-score achieved during training



Parameter importance with respect to best_f1

Config parameter	Importance	Correlation
extra_hidden_layer		
learning_rate		
is_multiclass		
per_device_train_batch_size		
num_groups		
weight_decay		
lr_decay		
frozen_layers		
dropout_rate		
scheduler.value_linear		
scheduler.value_cosine		

Figure 5: Parameter importance and correlation with the best F1-score achieved during Stage 2

6.5 Appendix E: Analysis

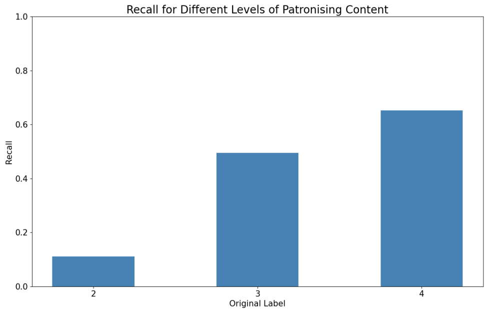


Figure 6: Model performance on the official dev set for different levels of patronising content

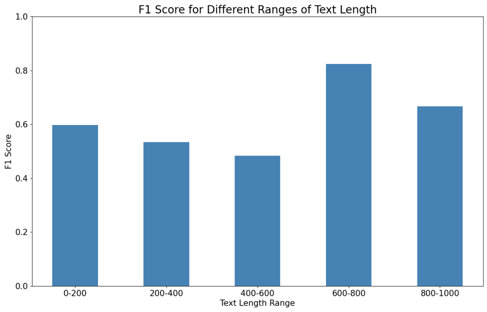


Figure 7: Model performance on the official dev set for different lengths of input sequence

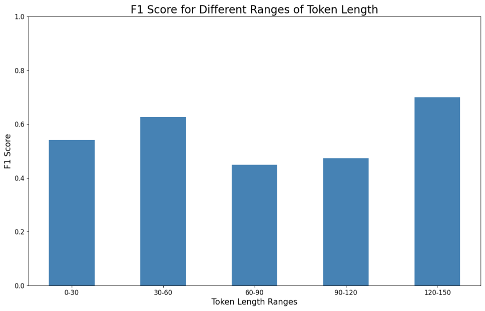


Figure 8: Model performance on the official dev set for different lengths of input tokens (truncated at 128 tokens)

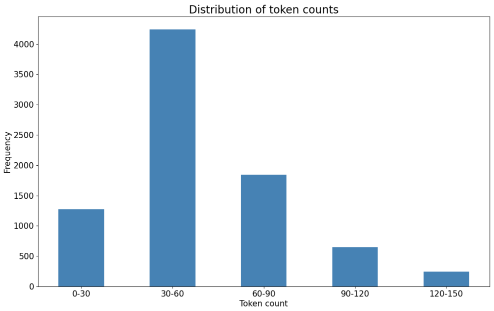


Figure 9: Distribution of token counts in the training set (truncated at 128 tokens)

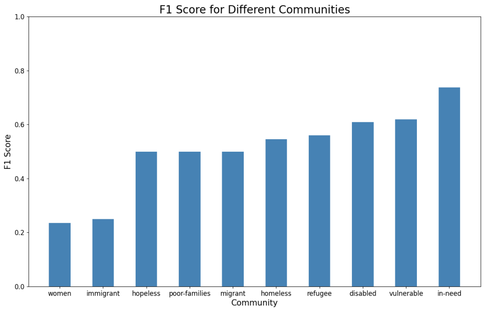


Figure 10: Model performance on the official dev set for different data categories

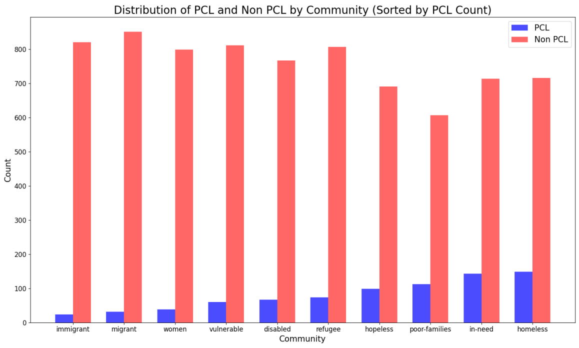


Figure 11: Distribution of PCL and non-PCL counts by community in the training set