Report For NLP Coursework

Junyu Chen
Imperial College London
jc522@ic.ac.uk

Yeshuai Cui
Imperial College London
yc322@ic.ac.uk

Congyun Guo
Imperial College London
cg222@ic.ac.uk

1 Introduction

While there is substantial work on modeling discriminative language, patronising and Condescending Language (PCL) is still an emergent area of study in NLP. This brings a new challenge to the community as PCL is commonly unconscious, subtler, and more subjective than the types of discourse that are typically targeted in NLP. In terms of its effect, PCL can potentially be very harmful, as it feeds stereotypes, routinizes discrimination, and drives to greater exclusion. The objective of this report is to implement different NLP models: BoW (1), RoBERTa (2), DeBERTa (3), BigBird (4), XLNet (5), BERT (6) and ALBERT (7), and compare their performance using the dataset from SemEval 2022 competition, Task 4 (8). We mainly focus on Subtask 1, a binary classification problem, i.e., Given a paragraph, a system must predict whether or not it contains any form of PCL.

2 Data analysis of the training data

2.1 Analysis of the class labels

According to the dataset, there are 10,636 paragraphs for the binary classification task. Each of these paragraphs was assigned one label ranging from 0 (not containing PCL) to 4 (being highly patronising or condescending). The table 1 summarizes the distribution of labels. Need to notice here, the label here is the original label in the dataset, and 0-1, 2-4 will be further grouped into non-PCL and PCL. Table 1 highlights that the binary classification dataset is highly unbalanced as there exist significantly more non-PCL samples (90.5%) than PCL samples (9.6%).

Label	0	1	2	3	4
Count	**8529**	947	144	**458**	391

Table 1: Label Count

Table 2 shows the relation between text length (number of words) and the distribution of classes. Despite small variations in the percentage of samples across different lengths, this work observes no strong correlation between text length and class.

Text Length	Label	Percentage
$(0, 20]$	non PCL	92.04%
	PCL	7.96%
$(20, 40]$	non-PCL	92.16%
	PCL	7.84%
$(40, 60]$	non-PCL	90.04%
	PCL	9.96%
$(60, 80]$	non-PCL	87.96%
	PCL	12.04%
$(80, 100]$	non-PCL	89.51%
	PCL	10.49%

Table 2: Text Length

There are ten different keywords in the dataset (*disabled, homeless, migrant, woman, etc.*); two extreme keywords are immigrant (with 12.8% patronising, 87.2% non-patronising samples) and homeless (with 16.5% patronising, 83.5% non-patronising samples). This difference is quite significant and will be explored in the following sections.

2.2 Qualitative assessment of the dataset

PCL classification task is hard. This is because PCL language can be subjective and context-dependent; what one person considers PCL may differ from what another person considers PCL. Additionally, PCL language can be challenging to recognize because it can be subtle and not immediately apparent, making it challenging to identify and annotate PCL for training an NLP model. Finally, PCL language can be challenging to detect because it often involves nuanced language,

such as implicit biases, presuppositions, and other forms of indirect language. The following examples show how complex and subjective this task is.

"Many donors prefer to remain anonymous, having found a more solid satisfaction comes from knowing what they are doing is truly helping those in need, than is derived from others' praise."

The statement seems to be a simple explanation of why some donors prefer to remain anonymous, and there does not seem to be any indication of condescension or patronisation. However, this example is classified as PCL.

"By learning a skill, and then passing it on to others, Otoyo is not only changing the narrative around disabled people, his work of knitting has also helped him to overcome the trauma of a decade spent fighting under one of the world's most brutal rebel groups."

The statement describes the positive impact of Otoyo's work in learning and teaching a skill, and there is no indication of condescension or patronisation towards disabled people. However, this example has label 3, which means it is PCL.

3 Modelling

3.1 Training Labels

Following the official interpretation of the training labels, we consider 0-1 as non-PCL and 2-4 as PCL. In this report, we calculate and focus on the F1 score of the positive class (PCL).

3.2 Data Pre-process

We split the official training set into training and own validation sets with a ratio of 4:1, and the official validation set is used as the own test set.

At first, we only use the text feature, whereas other features will be explored further in section 4.1. Particularly, this work preprocesses the text feature by truncating, tokenizing the sentences, and concatenating the [CLS] and [SEP] tokens.

3.3 Experiment Setup

We start with the pre-trained model and then apply an extra fully connected layer with a sigmoid activation function to the output of the CLS token. We use binary cross-entropy, batch size 16, and AdamW optimizer with weight decay 1e-4 as the default hyperparameter configuration for all the models.

3.4 Learning rate and Model Type

After training the DeBERTa, BigBird, XLNet, and ALBERT with learning rates ranging from 1e-3 to 1e-6[1], we found that all the models prefer learning rate 1e-5. Moreover, Figure 1 demonstrate that DeBERTa achieves the highest F1 and outperforms the RoBERTa baseline. Therefore, we focus on tuning DeBERTa in the following sections.

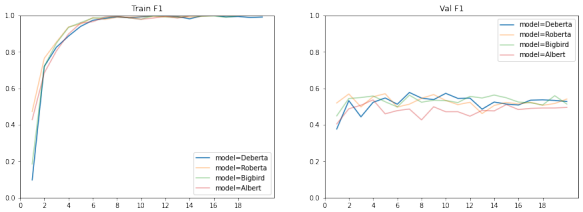

Figure 1: Learning Rate 10^{-5}

3.5 Cased vs Uncased

We trained both a cased and an uncased version of DeBERTa and BERT. Following table 3, we conclude that the case information is non-trivial, and reserving this feature is beneficial to the model performance.

	Uncased	Cased
BERT	0.5209	**0.5314**
DeBERTa-Base	0.5574	**0.5769**

Table 3: F1 Score on Own Validation Set

3.6 Hyper-parameter Tuning

3.6.1 Epochs

Because the validation curve is not always convex, early stopping may result in a poor outcome. Thus, we save the model checkpoints every two epochs and choose the one with the highest validation F1 score.

3.6.2 Learning Rate vs Learning Schedule

We compared the learning rate schedules Noam (9), CosineAnnealing, and Exponential. To align with the previous experiments, the learning rate is tuned so that all of the values produced by the schedules vary between 1e-5 to 1e-4[2].

Following figure 2, we discovered that the Noam schedule's warm-up phase improved model performance, while the other two schedules did

[1]Please refer to Appendix A
[2]Please refer to Appendix B

not significantly improve the F1 score. Furthermore, the poor performance of CosineAnnealing is perhaps a result of model underfitting, as its train F1 is significantly lower than other models.

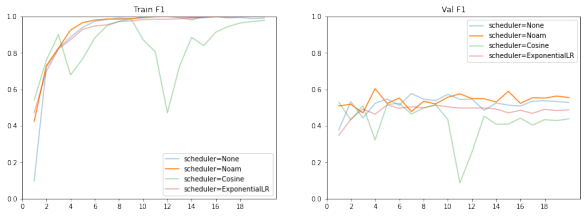

Figure 2: Learning Rate Schedules

4 Further model improvements

4.1 Keyword Pre-processing

As shown in section 2.1, there appears to be a strong correlation between the terms "immigrant" and "homeless" and the labels. Therefore, this section examines whether the model can use this feature to enhance performance.

We experimented with two keyword strategies. The first approach is to prepend the keyword to the beginning of the sentence. Another approach is to convert it to one hot vector and combine it with the CLS token, which will be the input feature for the classification layer.

We conclude that employing the one hot vector is more effective in light of the results in table 4. This phenomenon is perhaps due to giving the classifier direct access to keyword information. In comparison, when we prepend the keyword to the sentence, the keyword information may be reduced across the transformer layers.

No Keyword	Add-to-front	Onehot
0.5727	0.5844	**0.5907**

Table 4: F1 Score on Own Validation Set

4.2 Data Sampling

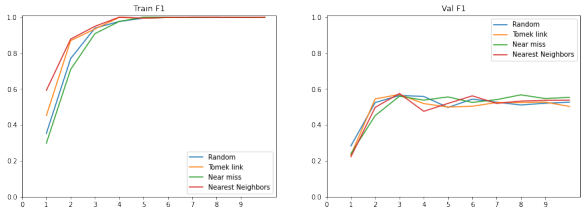

Figure 3: Under-sampling strategies

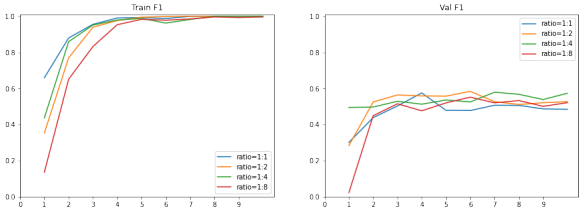

Figure 4: Under-sampling ratio

As discussed in Section 1, the data sample is unbalanced. To solve this issue, we experimented with random under-sampling, condensed nearest neighbors, Tomek link removal, and near-miss methods. Particularly, we convert input sentences into TF-IDF vectors and use functions from the imbalanced learn library to downsample the data to a 1:2 ratio.

Figure 3 shows that different sampling strategies do not significantly affect DeBERTa's performance. The best Tomek link method only outperforms random undersampling by 0.21 F1. Therefore, we believe undersampling methods do not achieve significant performance improvement to justify their computation cost, and random sampling is used for the following experiments.

Additionally, we also attempted different sampling ratios. We finetuned DeBERTa with 1:1, 1:2, 1:4, and 1:8 ratios and reported the result in Figure 4. Comparing the F1 of different sampling ratios, we observe that the 1:2 ratio between positive and negative data achieves the highest F1 score. Despite this, the F1 of 1:2 and 1:8 is not very different, highlighting that the pre-trained model is robust and unbalanced data does not cause a substantial bias in the model's prediction.

4.3 Ensembling

We also attempted bagging and voting ensembling methods to improve the model performance. For the bagging strategy, we bootstrap the dataset into five subsets (half of the size of the original training set), train the model on the bootstrapped dataset, and aggregate their prediction with majority voting. Figure 5 demonstrates the F1 score on the development dataset when bagging five different models. A significant improvement can be observed when bagging CNN models. While the performance of bagging BERT-like models is not considerably enhanced.

In addition to bagging, we also attempted a voting ensembling strategy, where the prediction of five models trained with different seeds, learning

rates, and dropouts are combined with majority voting. Following figure 6, the voting strategy leads to a lower F1 score for all of the BERT base models. This phenomenon underlines that changing the hyperparameter does not create sufficiently distinctive models, resulting in high prediction bias. We conclude that strong ensembling classifiers, such as BERT-like models, do not always lead to a rise in model performance. Therefore, ensembling strategies are not used for the final model.

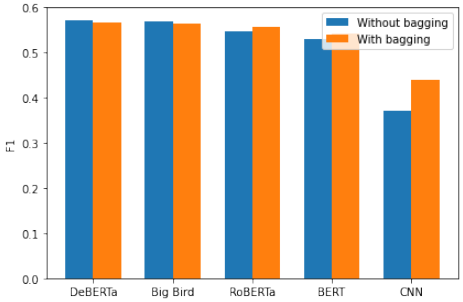

Figure 5: Models' F1 with bagging

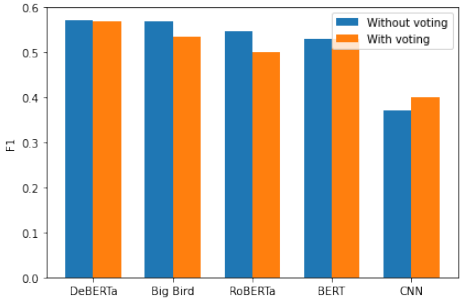

Figure 6: Models' F1 with voting

5 Experiment Results

Finally, we trained DeBERTa-Base with Noam learning schedule, using one-hot keyword vectors and random undersampling. The result is shown in Table 5.

	F1	Precision	Recall
BoW	0.3776	0.3006	0.5075
CNN	0.3742	0.2986	0.4984
DeBERTa	**0.5391**	**0.4564**	**0.6583**

Table 5: Performance on Own Testset (Official Validation Set)

In Table 5, we notice that DeBERTa significantly outperforms the F1 score of CNN and BoW, highlighting the effectiveness of the deep transformer model at the text classification task.

5.1 Baseline Comparison

In this section, we will dig deeper into the model of BoW. BoW uses a 2-layer neural network with an input of dimension equal to vocabulary size (23162). This gives us the convenience of inspecting the weights to understand feature importance. The first layer is a matrix of dimension 100x23162. We select the maximum absolute value of the vocabulary for each of the 100 output channels and count the occurrences.

feature (token)	occurrence
immigrant	**42**
homeless	21
life	10
help	4
three	1

Table 6: feature analysis

Table 6 shows a subset of important tokens with preserved ordering. Intuitively, the top two tokens provide important information in classifying the sample. However, other tokens, such as "three", should be irrelevant to the task but appear in the list; this highlights the limitation of BoW in extracting important features. Furthermore, consider the following example:

"Pope Francis washed and kissed the feet of Muslim, Orthodox, Hindu and Catholic refugees in a moving ceremony during Holy Thursday Mass at a refugee centre on the outskirts of Rome."

Since this example contains religious statements, BoW fails to detect this as it only counts word occurrences, misclassifying the example. On the other hand, our model uses deep transformer layers to extract features, thus considering semantic context and giving the correct prediction.

6 Analysis

Due to imbalanced data, this section calculates and reports the Micro F1 for the prediction result.

6.1 Level of patronising Content

Figure 7 demonstrates a monotonic increase of F1 score from 2 to 4 patronising level. This indicates that while the model is effective at categorizing high-level (label 4) patronising contents,

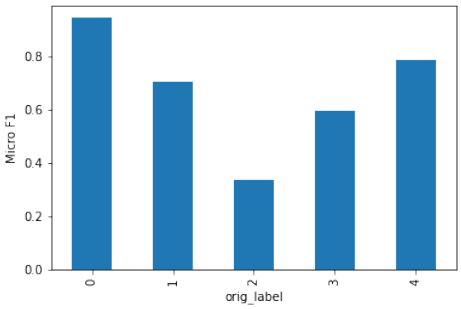

Figure 7: Level F1

while less effective at identifying low-level ones (label 2).

As shown in section 2.1, label 2 has the fewest samples, making it difficult for the model to learn. Label 2 is also at the boundary between non-patronising and patronising, further confusing the model. In comparison, it is easier for the model to classify high-level PCL content, resulting in a good F1 performance.

6.2 Length of Input Sentence

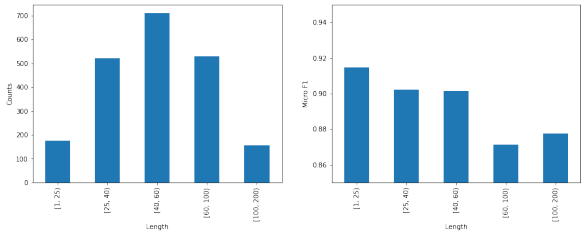

Figure 8: Length F1

We divide the word count by making the number of sentences in each partition conform to a normal distribution. As figure 8 (right) shows, the model performs well when classifying short and medium sentences ($<$ 60) but bad for long sentences ($\geq$ 60). Since a large number of data have a length less than 60, allowing the positional embedding at low indexes to be updated frequently, resulting in better F1 when classifying shorter sentences. Conversely, only a few data have lengths greater than 60, making it difficult for the model to learn the correct positional embedding, resulting in lower F1.

6.3 Keyword Type

Figure 9 shows that the model performs well on the keyword 'immigrant' while badly on 'homeless' and 'poor-families'. In general, the model performance is positively correlated with the proportion of the majority label 0. Some exceptions are 'refugee', 'vulnerable', and 'in-need', since the distribution of the test set is slightly different from our training set.

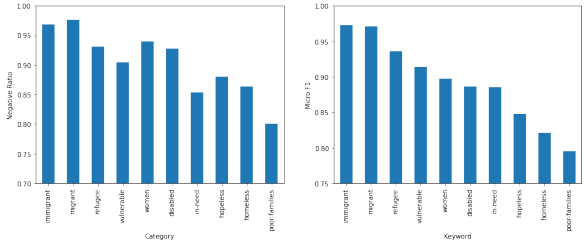

Figure 9: Keyword F1

Considering TP for the positive label. For the keyword 'homeless', our training set is more unbalanced and thus misleads the model to produce a high FN on the test set. For the keyword 'poor-families', both high FN and FP are observed, and we conclude that it is the hardest keyword for the model to classify.

7 Conclusion

In summary, this work first analyzed the dataset, observing the difficulty and unbalanced nature of the dataset. Then, this work explored various models and settled with DeBERTa, which outperforms the CNN and BoW baseline. Next, this work performed hyperparameter tuning on DeBERTa and found that Noam scheduler, 1e-5 learning rate, and Cased input achieved the highest performance. This work also attempted keyword concatenation, various data undersampling strategies, bagging, and voting, to further improve our DeBERTA performance. Ultimately, our final model achieved 0.54 F1 in the official dev set, outperforming the baseline RoBERTa-Base 0.49 F1. By evaluating our final model's performance, we observe that the model excels in classifying data with high patronising levels and short lengths but gets confused when encountering long and low-level patronising samples.

To solve this issue, future work can experiment with oversampling long and low-level patronising samples. For instance, data augmentation methods can be attempted to generate artificial samples for those samples. Then, adding those artificial samples to the training set may improve model performance on long or low-patronising samples.

References

[1] Z. S. Harris, "Distributional structure," *¡i¿WORD¡/i¿*, vol. 10, no. 2-3, pp. 146–162, 1954. [Online]. Available: https://doi.org/10.1080/00437956.1954.11659520

[2] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," Jul. 2019, arXiv:1907.11692 [cs]. [Online]. Available: http://arxiv.org/abs/1907.11692

[3] P. He, X. Liu, J. Gao, and W. Chen, "DeBERTa: Decoding-enhanced BERT with Disentangled Attention," Oct. 2021, arXiv:2006.03654 [cs]. [Online]. Available: http://arxiv.org/abs/2006.03654

[4] M. Zaheer, G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed, "Big Bird: Transformers for Longer Sequences," Jan. 2021, arXiv:2007.14062 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2007.14062

[5] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding," Jan. 2020, arXiv:1906.08237 [cs]. [Online]. Available: http://arxiv.org/abs/1906.08237

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," May 2019, arXiv:1810.04805 [cs]. [Online]. Available: http://arxiv.org/abs/1810.04805

[7] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations," Feb. 2020, arXiv:1909.11942 [cs]. [Online]. Available: http://arxiv.org/abs/1909.11942

[8] C. Perez-Almendros, L. Espinosa-Anke, and S. Schockaert, "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*, 2020, pp. 5891–5902.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Appendix A

Model	Best Epoch	F1 Score	learning rate
Deberta-Base	10	**0.572707**	1e-04
Roberta-Base	2	0.569270	1e-05
Bigbird-Roberta	15	0.548315	1e-05
XLNet	16	0.548077	1e-05
Deberta-Small	20	0.543909	1e-04
XLNet	18	0.537585	1e-06
Albert-Base	4	0.536998	1e-05
Deberta-Base	20	0.533333	1e-06
Bert-Cased	2	0.531429	1e-05
Deberta-Small	14	0.529412	1e-05
Bigbird-Roberta	18	0.527897	1e-06
Bert-Uncased	6	0.520930	1e-05
Albert-Base	12	0.501099	1e-06
Deberta-Small	20	0.413919	1e-06
Albert-Base	14	0.387528	1e-04
XLNet	2	0.101167	1e-04
Deberta-Base	2	0.068858	1e-04
XLNet	12	0.038278	1e-03
Bigbird-Roberta	2	0.000000	1e-03
Deberta-Base	2	0.000000	1e-03
Albert-Base	2	0.000000	1e-03
Roberta-Base	5	0.000000	4e-05
Deberta-Small	2	0.000000	1e-03
Bigbird-Roberta	2	0.000000	1e-04

Table 7: F1 Score on Own Validation Set

Appendix B

This section desribes the scheduler configuration for our experiments

Noam

We set $lr_{init} = 10^{-4}$. This will result in $lr_t \in [10^{-5}, 10^{-4}]$

$$lr_t = lr_{init} \cdot min(t^{-0.5}, t \cdot warmup^{-1.5})$$

CosineAnnealing

We set $lr_{min} = 1-^{-6}$ and $lr_{max} = 10^{-4}$. The initial $T_{max} = 3$ and it is doubled after every restart. While T_{cur} increases with t and is reset to 0 after every restart.

$$lr_t = lr_{min} + \frac{1}{2}(lr_{max} - lr_{min})(1 + cos(\frac{T_{cur}}{T_{max}}\pi))$$

Exponential

We set $lr_{init} = 10^{-4}$ and $\gamma = 0.8$

$$lr_t = lr_{init} \cdot \gamma^t$$