Patronizing and Condescending Language Detection in English and Mandarin

Xinyi Xiang
University of Washington
xinyix7@uw.edu

Catherine Qingyun Wang
University of Washington
qywang@uw.edu

Paula Cortés Lemos
University of Washington
mpcortes@uw.edu

Abstract

Patronizing and condescending language (PCL) is a form of harmful language that belittles and assumes pity for vulnerable groups, reinforcing negative societal stereotypes. However, unlike overt hate speech, it is often difficult to detect, even for human judges. This paper proposes a system for the automated detection of PCL. Our first approach uses static word embeddings and a convolutional neural network for classification, while our second approach uses pre-trained BERT embeddings fine-tuned for the PCL task. Our second approach was considerably more successful, and achieved an F1 score of 0.528 over the development set.

1 Introduction

Patronizing and condescending language (PCL) is often expressed unconsciously, but is nonetheless harmful towards vulnerable communities. Speaking about or generalizing entire communities in a pitying way can lead to the reinforcement of harmful stereotypes and normalize discrimination. The *Don't Patronize Me!* dataset (Perez-Almendros et al., 2020) is a large-scale dataset for PCL detection and classification. However, PCL is usually subtler and often unconscious compared to over forms of hate speech or sarcasm, making it difficult to detect even for human judges.

We propose and implement a system for detecting PCL, first using static word embeddings and convolutional neural networks (CNNs), and using a transformers-based approach using pre-trained BERT embeddings.

2 Task Description

Our primary task comes from task 4 of SemEval 2022, which focuses on the detection of Patronizing and Condescending Language (PCL). The task describes PCL as language that 'reveals a superior attitude towards others' and is 'often unconscious but harmful and discriminative' (Perez-Almendros et al., 2022). Unlike toxic or offensive language, PCL is not overtly aggressive or inflammatory, but nonetheless furthers power imbalances and harmful stereotypes towards vulnerable communities.

The data set used for this task is the *Don't Patronize Me!* data set presented in Perez-Almendros et al. (2020), an annotated corpus of PCL towards vulnerable communities. The authors define seven subcategories of PCL:

- Unbalanced power relations: the author positions themselves as a savior of those in need
- Shallow solution: the author presents an overly simple solution as though the hardship were superficial
- Presupposition: the author makes assumptions or stereotypes about a vulnerable community
- Authority voice: the author presents advice as an authority about how to overcome underprivileged situations
- Metaphor: the author describes difficult situations in softer, poetic ways
- Compassion: the author uses flowery wording to describe the group with pity
- The poorer, the merrier: the author praises the hardships as beautiful

The SemEval 2022 task has two subtasks: first, the idenfication of PCL, a binary classification task; and second, the identification of which of the seven categories a given instance of PCL belongs to, a multi-class classification task.

Our primary task will focus on subtask 1. This task involves analyzing English text to detect the affect of condescension, a binary classification task. The task will involve text data only, and the genre of said text will be excerpts from news articles.

Our adaptation task will adapt this task for another language, Mandarin Chinese. We will use the CCPC (Chinese Corpus for Patronizing and Condescending Language) corpus, a corpus of discourse from Weibo and Zhihu platforms (Wang et al., 2023). This corpus additionally annotates entries with 'PCL toxicity strength', but can still be used for the binary classification task, distinguishing between entries that are labeled as not toxic and entries that are labeled with any level of toxicity.

3 System Overview

Our initial system features two separate approaches to the PCL detection task. Our first approach tackles the classification task using static embeddings to represent the text and neural networks to perform classifications. We use pre-trained GloVe embeddings as input and, following other attempts at the task using CNNs (such as Siino et al. (2022)) and experimentation, incorporate dropout and pooling in our initial network architecture. Although the approach outperforms a random baseline, the overall accuracy over our devset is quite poor. Thus, we consider a second approach, a transformer-based approach in which we fine-tune a pretrained BERT for classification model over our data set. This approach saw considerably more success, with an F1 score of 0.53 over our devset.

Our end-to-end system includes data preprocessing by removing URLs, html tags, punctuation, emojis, and stopwords. We also balance the data, as the initial dataset is heavily skewed towards the negative class. Finally, our system evaluates outputs compared to gold standard labels over a development set, producing precision, recall, and F1 scores.

4 Approach

In this section, we review our approach and the components of our CNN and transformer models. We will first briefly overview our data preprocessing, then go into detail on the architecture of each model.

4.1 Dataset Composition

For our primary task, the data from the *Don't Patronize Me!* dataset contains 10,469 entries. To avoid tuning to the test data during our development phase, we partition this data into training and development sets, resvering 10% of data entries for the devset.

The data set labels each entry with an integer value between 0 and 4, inclusive. These represent the 'level' of PCL determined by the human annotators, with 0 being the least and 4 being the most condescending. In accordance with the methodology outlined in (Perez-Almendros et al., 2022), all labels were initially transformed into binary states. Labels originally designated as 0 and 1 were categorized as non-PCL cases and subsequently relabeled as 0. Conversely, labels originally designated as 2, 3, or 4 were identified as PCL cases and subsequently relabeled as 1.

4.2 Text Preprocessing

During our text preprocessing, we utilized NLTK tokenizer and BERT's default tokenizer to perform tokenization. Subsequently, we replaced any characters that are not English letters or punctuation with a blank space character. Furthermore, we eliminated any URLs and HTML tags to enhance clarity. Lastly, we removed punctuation characters from a predefined string encompassing common English punctuation symbols. The removal of these features in sentiment analysis has been proven to generally have a positive impact on results, as it allows the data to be less noisy, so the system can focus on high-impact features (Alam and Yao, 2019).

Additionally, for our BERT model, we also pad our sentences with `[CLS]` and `[SEP]` tokens to each sentence for encoding purposes. Specifically, `[CLS]` token is prepended to the start and `[SEP]` token is appended to the end. This then creates attention masks to differentiate between padding and non-padding tokens.

4.3 Data balancing

The training set for our primary task is an unbalanced data set, with 90.5% of data entries being negative samples. This posed problems for our convolutional model, as it is able to achieve high validation accuracy by merely predicting 0 (not PCL) for every entry, resulting in the model weights quickly zeroing out over training epochs. To combat this, our initial model balances the data by undersampling the negative samples to create a dataset that has equal positive and negative samples. However, this dataset is much smaller, with just over 1,000 samples rather than over 10,000, and as a result, we lose a lot of the information in those negative samples. In future iterations, we intend to continue experimenting with data augmentation methods, such as a synonym substitution method used by

(Vázquez Ramos et al., 2022).

4.4 System Architecture

4.4.1 CNN with static embeddings

For our initial model, we use static GloVe embeddings to represent the text data. Specifically, we use the average vector sum of the tokens in each data entry to create an input vector for each data entry, following the Sentence2Vec method. This allows the representation for each entry to be comprised of its token embeddings while being normalized for sentence length.

We use PyTorch dataloaders to create tensors for our training and validation set, which are randomly split 80-20. After experimenting with different model architectures, our most recent model uses cross-entropy loss and the Adam optimizer. Following the findings from (Siino et al., 2022), which demonstrated the success of convolutional layers involving different kernel sizes as well as dropout and pooling, we experimented with the kernel size for the *Conv1D* layer filters and applied dropout and max pooling. With some experimentation, we also found 0.0001 to be the best learning rate for our model. The following graphs show the loss and training and validation accuracy of this model over 50 epochs.

4.4.2 BERT model

Condescending language is often subtle, context-dependent, and can be challenging to detect accurately. The low baseline scores for this task and the poor performance of our initial model suggest that static embeddings may not be enough to capture the dependent relationships between words and the phrasal subtleties of condescending language. For these reasons, we found BERT (Bidirectional Encoder Representations from Transformers) suitable for the PCL detection task as they offer advantages in terms of contextual understandings. In addition, BERT's ability to represent the semantics of language helps in identifying condescending expressions even when they are not overtly stated.

Given the similarity of PCL detection to other subtle affect detection tasks such as hate speech detection, and the popularity of transformer models such as BERT for these tasks, we chose to adapt a transfer learning approach by fine-tuning a pre-trained BERT model on the PCL detection dataset. In particular, this model is trained with self-supervision over unlabeled text with the masked language model and next sentence prediction tasks (Devlin et al., 2019).

Specifically, we employed BERT-base-uncased, which was pre-trained in English with 12 layers, 768 as its hidden size, 12 heads, and 110M parameters. BERT utilizes almost the same architectures for pre-training and fine-tuning for different tasks; the only difference is the output layer, which can vary depending on the goal of the model. We fine-tuned our system for binary classification with a batch size of 16 and 4 epochs, within the range recommended by Devlin et al. (2019). However, after training, we noticed that for the last two epochs, loss was continuing to drop with little change in validation accuracy. Thus, to avoid possible over-fitting, we decided to retrain the model with only 2 epochs, and found that we achieved better results over our devset.

For optimization, we employed the AdamW optimizer with a default epsilon value set at 1×10^{-8}. Additionally, we used linear learning rate scheduling to explicitly set the learning rate between iterations.

5 Results

This section will present the initial results of our system over the aforementioned development set that we created. The development set has 1,046 entries, with 105 positive samples, thus having a similar proportion of negative to positive samples as in the complete training set.

Our system is set up to output predictions in the form of a text file with one prediction, 0 or 1, on each line. We use our own script, which is adapted from the official scorer of the SemEval-2022 task, to perform our evaluation (we are unable to use the official scorer for now, as it compares the predictions to the test set, while we are currently evaluating over our development set).

The official scorer produces three scores for each subtask: precision, recall, and F1 score, comparing the predicted labels to the ground truth labels, with 1 being the positive class. Table 1 shows the performance of our system with each of the following model specifications. (The results of our CNN model without undersampling are not included; because the model predicted the negative class for every single entry, the precision and recall are both 0.)

Ultimately, our static embedding based models performed quite poorly, with both the linear and

Model	**Precision**	**Recall**	**F1 Score**
Linear NN	0.129	0.038	0.058
Conv1D NN (with undersampling)	0.084	0.343	0.135
BERT (4 epochs)	0.589	0.306	0.403
BERT (2 epochs)	0.565	0.495	0.528

Table 1: Precision, recall, and F1 score for each model type over the development set.

convolutional NN used for classification. The fine-tuned BERT model performed the best, with an improvement in F1 score after training for only 2 epochs. We have not tested the BERT model with undersampling due to concerns about the undersampled dataset being too small to fine-tune the BERT model; in future iterations, we plan on experimenting with other methods of data augmentation for the BERT model.

6 Discussion

Compared to the random baseline and the best-performing systems at SemEval-2022, our first implementation performs extremely poorly, while our second implementation achieved substantial improvement and performs reasonably well on the binary classification task. Nonetheless, there is still much room for improvement; this section will discuss some of the errors our system made and how the system can be improved in future iterations.

The first issue is the unbalanced data issue. Before we implemented undersampling, our CNN system predicted 'not PCL' for every entry on the dev test, suggesting that the unbalanced data was allowing the system to achieve very high validation accuracy without even considering the positive class. Of course, this defeats the purpose of the system, as the goal of PCL detection is specifically to detect positive cases when they occur. However, our random undersampling approach was not particularly successful, either. This approach resulted in a much smaller dataset than we started with, meaning that a lot of helpful data is simply lost. Because convolutional neural networks often require large amounts of data for optimal performance, we do not want to be throwing away large amounts of training data that could be useful for the inference task.

Another issue is the issue of overfitting during training. Though our CNN system was able to achieve high training and validation accuracy and low loss over 50 epochs, the F1 score over the development set was still poor. We suspect that this also somewhat has to do with the unbalanced data; even when training on the balanced undersampled data, the system will still perform poorly over the dev set because it is also unbalanced. Moreover, the small training set size as a result of the undersampling also makes it prone to overfitting. The overfitting issue could also be alleviated by cross-validating across multiple validation and development sets, rather than only using a single split.

In future iterations, we believe we can improve upon the CNN system by improving the data augmentation method and by conducting further experiments with the model architecture, including the inclusion of dense layers, or a multi-channel CNN model as seen in Siino et al. (2022). Moreover, alternate vector representations for each input data point could be considered; the current method of using static word vectors and aggregating them to create a vector representation of each data point could be improved using document vector representations or other feature sets.

However, we focus our attention on our BERT model, which is currently achieving much better performance. We have already run tests over different epoch counts to avoid overfitting, but further experimentation will be needed to tune other hyperparameters such as the learning rate scheduler.

Our preliminary results show that our model achieves a better precision than recall score over our development set. Table 2 shows some examples of errors made by our fine-tuned BERT system. As we can see, the difficulty in PCL detection lies partially in that many of the passages that contain PCL and those that do not use many of the same phrases. For example, in these examples, phrases like 'crippling', 'vulnerable', or 'anxiety' can be used either factually to denote a specific state, or in a flowery, pitying way. The challenge is in differentiating between sentences about the same concepts based on the tone with which the author describes these concepts, and that distinction is not always made by the inclusion or exclusion of specific phrases. In future iterations, we may consider ensemble methods that may help us mitigate some of the brittleness of the current model.

7 Ethical Considerations

References

Saqib Alam and Nianmin Yao. 2019. The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. *Computational and Mathematical Organization Theory*, 25.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Carla Perez-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)*, pages 298–307, Seattle, United States. Association for Computational Linguistics.

Marco Siino, Marco Cascia, and Ilenia Tinnirello. 2022. McRock at SemEval-2022 task 4: Patronizing and condescending language detection using multi-channel CNN, hybrid LSTM, DistilBERT and XLNet. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 409–417, Seattle, United States. Association for Computational Linguistics.

Laura Vázquez Ramos, Adrián Moreno Monterde, Victoria Pachón, and Jacinto Mata. 2022. I2C at SemEval-2022 task 4: Patronizing and condescending language detection using deep learning techniques. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 459–463, Seattle, United States. Association for Computational Linguistics.

Hongbo Wang, Mingda Li, Junyu Lu, Liang Yang, Hebin Xia, and Hongfei Lin. 2023. CCPC: A hierarchical Chinese corpus for patronizing and condescending language detection. In *Natural Language Processing and Chinese Computing*, pages 640–652, Cham. Springer Nature Switzerland.

Text	**Ground Truth Label**	**Predicted Label**
"They follow a crippling El Nino-triggered drought which scorched much of the region last year , hitting crop production and leaving millions in need of food aid ."	0	1
"If they couldn't , automated "" robo-debt "" letters told them to pay up , in an inversion of the usual onus of proof . It worked like extortion . Some of the victims were vulnerable , some couldn't cope ."	0	1
According to Betty-Ann Blaine , executive director of Hear the Children 's Cry , deterioration of family life ; instability/shifting households ; crippling levels of poverty ; lack of adequate social support systems and heartbreaking levels of hopelessness are key factors leading to children running away from homes .	1	0
"Homeless children are "" living like refugees in their own country "" , deprived of the ability to make choices about when they eat , and in many cases suffering high levels of anxiety about their parents ' health ."	1	0

Table 2: Some examples of errors made by our fine-tuned BERT system.