Patronizing and Condescending Language Detection SemEval 2022 Task 4, Subtask 1: Binary classification

Tuguluke Abulitibu
DHI Group Inc. / 6465 Greenwood Plaza Blvd, Greenwood Village, CO 80111
University of Colorado, Boulder / Boulder, CO 80309
abulitibu.tuguluke@dhigroupinc.com
abulitibu.tuguluke@colroado.edu

Abstract

In this paper, we present the systme description and the main results of SemEval 2022 Task 4 on Patronizing and Condescending Language Detection (PCL). The task included 2 subtasks corresponding to the Binary classification and Multi-label classification, we will approch the subtask 1 with a simple CNN logistic regression(as a baseline), and with BERT if the results bench mark the CNN approach. We will conlude with the main findings and future work.

1 Introduction

Sentimental analysis has long been studied in natural language processing(Pang, 2004). Recently, there is a growing communities studying condescending languages with NLP(Wang and Potts, 2019), Almendros's work on patronizing and condesending languages towards vulnerable communities(2020) brings a new perspective on how we can apply the machine learning applications to this old problem . Our work is completed base on their annotated dataset of 10, 637 paragraphs which were selected from general news stories in 20 English speaking countries(Australia, Bangladesh, Canada, Ghana, Hong Kong, Ireland, India, Jamaica, Kenya, Sri Lanka, Malaysia, Nigeria, New Zealand, Phillipines, Pakistan, SIngapore, Tanzania, UK, United States, South Africa).

This paper is on subtask 1 from Task 4 on Patronizing and Condescending Language Detection (SemEval 2022) with binary classification, that is, given a paragraph, a system must predict whether or not it contains any form of PCL. Our system is applied a simple CNN model on Logistic regression, with a special word embedding technique using the combinations of 'tagger', 'parser', and 'ner' (We will experiment with all combination along with seperations).

Our finding in this project tells us that a simple architecture would gives us a reasonable result, what is more important is how we struture the training data.

The remainder of this paper is organized as follows: Section 2 describes the background of the prolem with simple data analysis. Section 3 presents the system that we used in the competition. Sections 4-5 present the results and discuss the approaches taken by the diffrent training data set splits, along with error find and real life examples to validate the model. Finally, Section 6-7 concludes and suggests some possible directions for future work.

2 Background

The Oxford definition of Condescending is: 'having or showing a feeling of patronizing superiority'. Followed by Patronizing: 'apparently kind or helpful but betraying a feeling of superiority; condescending' (Google). In order to encourage more research on PLC languge detections, Almendros provided ACL with their own annotated data with 2 human annotator (ann1, ann2) plus a referee (ann3). They categorized the level of PCL as a 5-point scale (0, 1 being non-PCL and 2, 3, 4 being PCL). Their criterias include:

- The saviour: The article authors may protrait themselves as saviour for the vunarable subjects, where there tends to be 'unbalanced power relations' between subjects and objects, there may also seems to be 'shallow solution' through out the essay.
- The expert: The article authorstends to know better with a lot of 'presuppositions' and 'authority voice'.
- The poet: The article authors tends to immerse with over rhetorical and literary style., where there are 'metaphors' and 'compassions'. If the focus group is poorer, then PCL scores tend to be higher('merrier').

The ten key words they seletcted as vulnarable comunities are 'disabled, homeless, hopeless, immigrant, in need, migrant, poor families, refugee, vulnerable and women.' Next, we look at how the data was structured.

2.1 Data

The data provided is dontpatronizeme_pcl.tsv, which contains paragraphs annotated with a label from 0 (not containing PCL) to 4 (being highly patronizing or condescending) towards vulnerable communities, where each columns represents:

- par_id is a unique id for each one of the paragraphs in the corpus. (We disregard this column)
- art_id is the document id in the original NOW corpus. (We disregard this column)
- keyword is the search term used to retrieve texts about a target community. (We disregard this column in training)
- country_code is a two-letter ISO Alpha-2 country code for the source media outlet. (We disregard this column in CNN training but not in LSTM)
- text is the paragraph containing the keyword.
- label is an integer between 0 and 4.

We also added a strong column for 'PCL', and 'NonPCL' just to made sure the label was correct (turned out it is) and for easier reading.
Table 1 is how each country, each word label, each PCL label and binary label counted. We can see the country and keyword are evenly distribution among each. THe only issue we can see is the unbalanced data, 90/10 ratio for NonPCL and PCL data. A simple labeling system of just printing PCL label would return a 'decent' result. So data manipulation is a process in this system design.

1	za: 549	migrant: 1089	0: 8529	0: 9476
2	my: 546	in-need: 1082	1: 947	1: 993
3	ph: 545	vulnerable: 1080	3: 458	
4	pk: 545	homeless: 1077	4: 391	
5	us: 543	women: 1070	2: 144	
6	au: 541	refugee: 1068		
7	gb: 540	immigrant: 1061		
8	ke: 539	disabled: 1028		
9	ie: 537	hopeless: 1005		
10	ng: 537	poor-families: 909		
11	sg: 535			
12	in: 530			
13	ca: 530			
14	gh: 523			
15	nz: 518			
16	bd: 512			
17	lk: 504			
18	jm: 490			
19	hk: 490			
20	tz: 415			

Table 1: Country, Keyword, Orig label(5-level), Label(PCL/NonPCL) Value counts

The only inside we get from Table 2 is keyword 'immigrant' with PCL is less reprented than rest of them, we can see in the later section, it is corret to say with a WSJ opinion piece with PCL (we believe it is).

Countries	0	1	2	3	4
au	462	42	4	19	14
bd	413	55	6	20	18
ca	427	57	5	25	16
gb	428	55	8	28	21
gh	396	52	13	29	33
hk	418	43	4	14	11
ie	436	49	12	21	19
in	456	35	7	20	12
jm	390	43	8	23	26
ke	440	54	8	14	23
lk	415	40	7	19	23
my	458	45	2	16	25
ng	404	61	7	37	28
nz	427	44	9	21	17
ph	428	50	13	28	26
pk	436	56	8	22	23
sg	452	45	3	20	15
tz	346	28	8	20	13
us	449	50	5	27	12
za	448	43	7	35	16
Keywords	**0**	**1**	**2**	**3**	**4**
disabled	857	90	19	37	25
homeless	746	153	26	75	77
hopeless	761	120	8	62	54
immigrant	988	43	5	18	7
in -need	797	109	19	76	81
migrant	998	55	5	16	15
poor-families	624	135	27	72	51
refugee	901	81	16	35	35
vulnerable	923	77	7	41	32
women	934	84	12	26	14

Table 2: Countries and Keywords vs. Orig label(5-level) value counts

Table 3 are just simple grouping number between Orig label and country along with country count. We can clearly see it is evenly distributed mong candiate as well.
Unfortunaly, none of the approach we did in the project os consider novel, all has been study by previous research, we just applied what they did with a little modificaiton. The only thing worth mentioning is the Spacy tokenization process, it has 3 differrent librarcy to choose betwen small/medium/large, we can aply the **en_core_web_sm** in the develpement phase for better speed, and **en_core_web_lg** for the final process.

Countries	disabled	homeless	hopeless	immigrant	in-need	migrant	poor-families	refugee	vulnerable	women
au	47	51	52	56	56	57	53	54	60	55
bd	50	55	45	49	51	56	46	52	55	53
ca	51	53	52	51	52	47	55	56	61	52
gb	55	50	47	55	56	53	58	57	58	51
gh	62	55	57	53	51	58	25	53	54	55
hk	59	53	32	50	55	57	22	49	52	61
ie	61	50	55	58	58	58	36	58	48	55
in	30	52	62	58	57	52	52	58	59	50
jm	53	62	47	54	58	50	11	54	50	51
ke	52	51	55	55	50	54	55	49	57	61
lk	52	57	57	51	48	52	32	56	49	50
my	58	48	46	53	62	57	53	58	60	51
ng	52	60	49	52	52	55	47	56	59	55
nz	62	45	61	51	48	56	50	49	49	47
ph	61	56	56	48	54	59	53	51	55	52
pk	50	55	51	51	58	57	57	56	54	56
sg	51	56	52	56	59	58	54	45	54	50
tz	9	54	18	51	49	45	38	48	52	51
us	53	60	54	51	54	54	53	59	47	58
za	60	54	57	58	54	54	59	50	47	56

Table 3: Country vs. keywords value counts

2.2 F1 scores

For the system evaluations, we use F1 score. **sklearn.metric** library has a good f1 scoreing function which we used. Among the following 4 parameters, we picked 'average=macro' in our report.

- average=micro: compute f1 by considering total true positives, false negatives and false positives(no matter of the prediction for each label in the dataset)
- average=macro: compute f1 for each label, and returns the average without considering the proportion for each label in the dataset.
- average=weighted: compute f1 for each label, and returns the average considering the proportion for each label in the dataset.
- average=samples: compute f1 for each instance, and returns the average. Use it for multilabel classification.

More will be explain in the conclusion section.

3 System overview

We picked CNN as our baseline model, since we consider it as 'go to' tool for text binary classification. There are many advantages of CNN toward other more 'cool and sexy SOTA' methods that are popular among online literature. among them:

- It is a mature techniques, easy to implement with external libraries.
- It has feature learing with a good feature extractor, and weight sharings
- It is fast (due tomemory and complexity efficiecy) to run the code than any pre-train methods. We did all the simulation on a Macbook (2.6 GHz 6-Core, 32 GB 2667 MHz DDR4 memory, each training takes less than 3 min).

But before we apply any of the deep learning iteration, we need to tokenize the text we got in our hand, this process is done using **Sapcy**'s nlp tokenizer library. The three major feature it considered are tagger, parser and ner. Later in the experiment we will try to switch off some of the feature to see it would improve the results or not.

Once the text is been vecterized, we are ready to feed it into the Logistic regression pipline, with stochastic (gradient descent) minibatching to speed up the training. We also set up the drop-off as .2 for the weight update. The other critria to stop the iteration is when we do not see major upgrade on f1 score for maximum of 5 epoch.

We set output as boolean PCL/NonPCL with a confidence score, e.g.:

```
{'PLC': 0.7293128967285156, '
    NonPLC': 0.27068713307380676}
```

We can later use those score to recategorize the validation set with diffrent threshhold to see if that would affect the F1 scores or not. e.g.: by labeling any confidence score higher than .3 as PCL other than .5.

The following is basic architecture of the system.

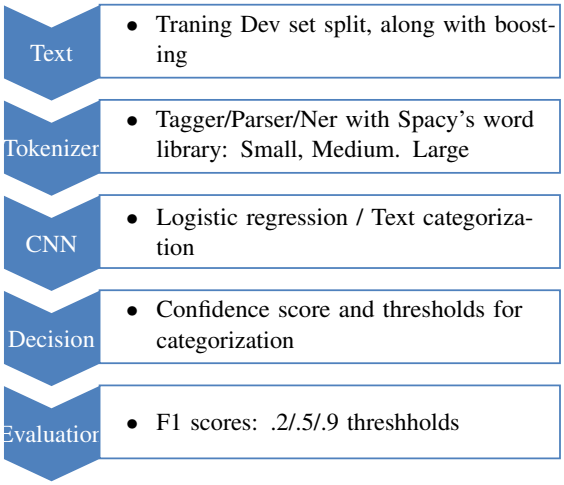

Figure 1: system Explanation

4 Experimental setup

4.1 Data

Following the BERT example, we first split the data into train, dev, and test. Dev is used for weighting training while also recording loss, f1, accuracy and recall for each epoch. Test set is used for extra F1 score calculatiton while using defferent passing thread (.5, .6, 0.7 etc).

4.2 Preprocessing

Some simple preprocesings were applied later in the experiment once the pipeline was setup. We cut down the number of the NonPCL so that to boost the PLC in a more balanced data in the processing

and record each results.
We also used the Spacy's NLP pipeline for our CNN classifer along with it's tokinizer on the text through out the process.

4.3 Evaluation mesures

Although the script has all the measure scored printed out during the training, we follow the guideline of SemEval 2020 for this particular task and only record the F1 score using Sklearn's metric library.

5 Results

5.1 Main Results

This section is mainly on hand picked numerical results. We first start with a simple 80/20 split, the printout is:

```
loss: 0.2987 f1: 0.643 accuracy: 0.888 P: 0.670 R: 0.625
```

Train/Dev/Test 60/20/20 split shows

```
loss: 0.5574 f1: 0.630 accuracy: 0.888 P: 0.673 R: 0.609
```

After setting up three threshhold for PCL, the result show no big of a difference.

PCL threshhold	F1 on test set
.2	0.65
.5	0.63
.9	0.60

5.1.1 Tuning down PCLpercentage

We then tried randomly dropout 30 percent of the PCL label, it shows

```
loss: 0.7539 f1: 0.621 accuracy: 0.888 P: 0.653 R: 0.605
```

PCL threshhold	F1 on test set
.2	0.66
.5	0.63
.9	0.62

Randomly dropout 50 percent of the NonPCL label give us an interesting log of f1 hit 0.7 at early iterations and then drop to .66.

```
loss: 7.7381 f1: 0.599 accuracy: 0.888 P: 0.687 R: 0.583
loss: 5.8414 f1: 0.675 accuracy: 0.888 P: 0.720 R: 0.652
loss: 4.5629 f1: 0.688 accuracy: 0.888 P: 0.712 R: 0.672
loss: 3.0936 f1: 0.700 accuracy: 0.888 P: 0.722 R: 0.685
loss: 4.0779 f1: 0.675 accuracy: 0.888 P: 0.689 R: 0.665
loss: 1.5275 f1: 0.665 accuracy: 0.888 P: 0.685 R: 0.652
loss: 1.0300 f1: 0.651 accuracy: 0.888 P: 0.671 R: 0.638
loss: 0.7745 f1: 0.664 accuracy: 0.888 P: 0.683 R: 0.652
```

It did not improve much with confidence threshold either.
Later, we set randomly dropout as 90 percent of the NonPCL label (PCL and NonPCL are 50/50 balanced and 'fast' training):

```
loss: 0.5543 f1: 0.672 accuracy: 0.888 P: 0.673 R: 0.672
```

and our test set agreed with the dev set F1 score.

PCL threshhold	F1 on test set
.2	0.72
.5	0.74
.9	0.75

At last, we did doubling the size of PCL onto the original data, the reult is a bit puzzling:

```
loss: 0.5925 f1: 0.839 accuracy: 0.888 P: 0.853 R: 0.829
```

PCL threshhold	F1 on test set
.2	0.86
.5	0.86
.9	0.86

5.2 Error

Like Professor Martin once said, F1 score is just a number for the manger. I found it to be more easily manipulated to suit certain agenda.
And we consider the higher F1 number as an error, since the PCL is over represented in both Dev and Test sets. Doubling the number of PCL text has the same issue. So we are ready to say the system F1 score is between .6 and .65, with no evaluation from the SemEval's result and comparisons.
two example is our system label the following as PCL, but real label is NonPCL, clearly a sign of NonPCL overfit.

```
besides , any sane person who has seen
    the insanity that is the six years
    of the aquino administration have
    much to be angry about . i certainly
     do . then again , there is much in
    our history to lament and in need of
     correction .
the staff members make these blankets in
     their free time or on their quiet
    days at the salon . they distribute
    the items to shelters , orphanages ,
     the homeless and others in need .
```

5.3 Some interesting cases

We found the openion section of the Wallstreet Journal to be pantronizing some times. Here is some of the title I considered PCL, and we can apply our methods to it. The first example say it is PCL, and the second one on immigrant labeled as NonPCL (which I think can be explained by less Immigrant topic in training than others).

Meet the Kidd Who Goes Toe to Toe With Warren Buffett ---- Patience, concentration and courage have allowed Wilmot Kidd to rack up one of the greatest long-term track records in the history of investing. He's a model for how to think about, and practice, intelligent investing.

No End in Sight for California Homeless Mess ----- Mental illness, addiction and the release of thousands of prisoners, all undoubtedly contribute to the current California homeless situation, but the core of the problem is supply and demand.

6 Conclusion

In this paper, we introduced our CNN system for the SemEval 2022 Task 4 on Patronizing and Condescending Language Detection (PLC) on binary classification. We start with data explanations, and how to set up preprocess (with Spacy tokenizer) the data for the system, our numerical results show the F1 score should be between .6 and .65, the higher one should be an error on PCL's over-representation.

7 Future work

We did succesfully ran one pretrained BERT model, but due to low F1 and excruciatingly long time for one 5 Epoch training, we did not ran a second time. That should be a priority with a better GPU resource. We did include a notebook convert to HTML with LSTM application (.47 for 5 –100 epoch, doulbing the PCL size will boose that number to .56), and that should be included for the future model comparison more scientifically. Last one can the best way to tune the data instead of the model, but that can be a philisophical one mush as technical, why would we not use majority of the hard collected data and focus on a few? which can be done in the first place.

8 Appendix

References

[1] Carla Perez-Almendros Luis Espinosa-Anke Steven Schockaert, 2020, Don't Patronize Me! An Annotated Dataset with Patronizing and Condescending Language towards Vulnerable Communities.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[3] Zijian Wang and Christopher Potts. 2019. Talkdown: A corpus for condescension detection in context. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing.

[4] Niall J Conroy, Victoria L Rubin, and Yimin Chen. 2015. Automatic deception detection: Methods for finding fake news. Proceedings of the Association for Information Science and Technology, 52(1):1–4.

[5] Bo Pang and Lillian Lee, 2004, A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts.

[6] David P. Helmbold, 2015, On the Inductive Bias of Dropout.

[7] Dan Jurafsky and James H. Martin, 2021, Speech and Language Processing (3rd ed. draft).

[8] H. Sak, A. Senior, F Beaufays, 2014, Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition