Sarcasm Detection with Deep-CNN Sentiment Model

Korea University COSE461 Final Project

KIM DONG HWAN
Department of Computer Science
Team 25
2016270209

Abstract

The original use of NLP in sarcasm detection has been straightly focused on context and meta-data. Since the use of sarcasm in usual dialogs are understandable when the background knowledge has common sense. The idea of isolating context features by using trainable Embedding Layers and analyzing the extracted features by using a more Deeply constructed Residual Net with transition, attention modules. Training with Headline texts and reuse as the pre-trained model testing on occasional dialog situations. The goal is to decrease the loss among the train and validation sequences and increase accuracy. This will provide the extrinsic information among general dialog.

1 Introduction

Sarcasm is another area of understanding the use of natural languages. Its linguistic characteristics intend a meaning of an utterance that is not the same literal meaning. In the voiced dialogs, people easily presume whether it is sarcastic or not with the volume of accents. But in the context, we use upper case sentences or background knowledge-based texting among resemble cultured environments. To understand these sentences, we need common background knowledge and commonsense irony. For modern NLP, it has been a challenge for effective computing in terms of learning procedures among the whole context data. The usage of Social Network Services made the understanding of sarcasm more important. So I made an adjustment using the traditional CNN model, with transitions and attention modules for an effective and simple sarcasm detection model.

2 Related Work

- Deep and Dense Sarcasm Detection by Pelser, D., Murrell, H. (2019)[1] is the reference of my model structure
- Sarcasm Detection: A Comparative Study by Yaghoobian, H., Arabnia, H. R., Rasheed, K. (2021) [2] showed recent state of art models for Sarcasm detection, which I took note of, to make an improvement.

3 Approach

I used the baseline of the model structure from the paper [1]. Adding layers of 1D-convolution, dropout, self-attention, dense transition. Adjust the feature extraction part with Norm or Activation functions, tuning hyper-parameters of optimizer and learning sequences.

- Stacked up ResNet-40 layers up (16 layers more than the original), to maintain isolated features of utterances.

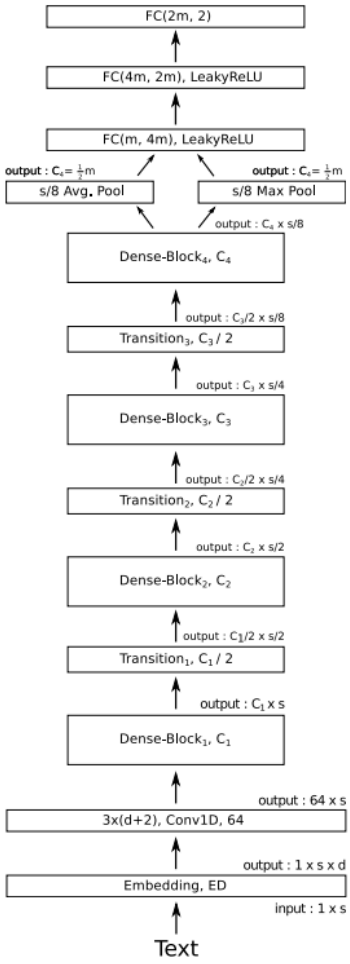

Figure 1: Model Structure

- Used LeakyReLU instead of ReLU, as the model goes deeper, it may cause dead ReLU problem which I do not want go through.
- Replaced AdaptiveAvgPool 1d to AdaptiveMaxPool 1d, so that could work better on feature characteristics extraction.
- Use the Densely connected Transition layer for the computation effectiveness and maintain the extracted features.
- Added Self-Attention module into residual layer for better feature extractions.
- Used pretrained Embedding weight during train session, make it trainable, so that could act better on whole new dataset.
- Adjust the weight initializing value for sensitive feature weight.
- Most of the structure were based on the paper [1], and the use of fastai module make it easier to implement.

4 Experiments

4.1 Data

Used News Headlines [3] dataset, for pre-training, and tested on Reddit comment dataset [4].

dweNet FastText-1M 300D	86.51
dweNet FastText-1M-subword 300D	86.15
dweNet GLoVe 50 static	85.68

Figure 2: Score of Headlines Set

	Main	
	Accuracy	*F1*
Bag-of-words	0.63	0.64
CNN	0.65	0.66
CNN-SVM [6]	0.68	0.68
CUE-CNN [4]	0.70	0.69
CASCADE [21]	0.77	0.77
CASCADE (no personality features)	0.68	0.66
AMR [7]	0.68	0.70
dweNet	0.69	0.69

Figure 3: Score of Reddit Main Set

To learn sarcastic features from more official use of words. Then test my trained model if it works on casual dialogs. The inputs are the sarcastic/nonsarcastic headlines and comment the outputs are bi-classified classes.

4.2 Evaluation method

Used loss function CrossEntropy and F1 scores for train and validation function evaluations. Comparing consumed time and criteria with the original shape of ResNet and customed net call DweNet referred in the paper. Finally check the overall performances, comparing with pre-trained GloVe weight of Reddit dataset with my Headline pre-trained model.

The assumption was about learning the model with the official use of words could improve the model functionality among most words.

4.3 Experimental details

- Stacked up the layers 3, 3, 6, 6 then trained on Headlines. Since the loss of original model was large, tried to reduce loss in following sessions.
- Put the self-attention model to residual layers. self-attention showed significant gross of accuracy and made faster learning, but also raised losses, so I thought my model has been overfitted.
- Add density to transition layer to maintain extracted feature avoiding loss during the transition [bottleneck], adding attention and dense lead the model to overfitting.
- Added dropout layers to prevent the overfitting, and it is now lack of learnings.
- While the accuracy is rising, loss also went high. So I thought the model is loosing the learned features, probably dead relu affected, replaced the relu to LeakyReLU.
- The loss of train and valid went low, now I tried on Weight initializing and Momentum value since I use Adam.

4.4 Results

The Scores had been observed Top of 5 trials.

- Headlines - Train Loss : 0.0934 || Valid Loss : 0.3312 || Acc : 0.8643 || Macro F1 : 0.8613
- Reddit - Loss : 0.667 || Acc : 0.662 || Macro F1 : 0.659

5 Analysis

The model I made was not better than the state of arts, but I think it had worked fluently since its origin was the ResNet CNN model. Since the deeper layer is easily overfitted on the training set,

longer computation time consumption even decreases the overall functionality. During the training session, I actually added Early stopping callbacks to stop the training by about 10 17 epochs. The self-attention supports residual layers extracting features of sarcasm, and dense transition also helped maintain those features since the transitions mostly cause losses.

6 Conclusion

The Traditional CNN model with recent transition, attention modules are working fluently well. Even though the training set and testing set were different, but it worked as much as the others which are using weighted embeddings on the same set. My assumption, that the official words are enough to predict casual texts, was not available in the real testing since the AI model is not really understanding natural languages, and I think most of the areas do the same. However, the result showed that the models, in terms of modern neural networks, are improving and working fluently with the variated from traditional methods. Including NLPs, plenty of NN models were originated from the traditional methods for the optimized solution and I think improving the traditional method of constructing AI models, could be the answer to AI in the future.

In the meaning of variations, I want to try even more with various types of models to the mixture of modern and antique. I think that maybe at some point, this could be another solution in some areas.

References

[1] Devin Pelser and Hugh Murrell. Deep and dense sarcasm detection. *arXiv preprint arXiv:1911.07474*, 2019.

[2] Hamed Yaghoobian, Hamid R Arabnia, and Khaled Rasheed. Sarcasm detection: A comparative study. *arXiv preprint arXiv:2107.02276*, 2021.

[3] Rishabh Misra. News headlines dataset for sarcasm detection. *Kaggle*.

[4] Dan Ofer. Sarcasm on reddit. *Kaggle*.