Quora Insincere Question Classification

Richard Yang
gyang79@wisc.edu

Junyao Wang
jwang2286@wisc.edu

Qiaochu Yu
qyu65@wisc.edu

Abstract

In this report, we would introduce our project about Quora insincere questions detection using DeepLearning. For each question in our dataset, we have text and corresponding label, and we try to figure out methods being able to achieve good classification accuracy for our data.

We use two kinds of architectures, one is CNN and the other is RNN. And because our dataset is extremely imbalance, we also apply techniques like, oversampling and weighted loss function to solve that problem.

After experiments, we come into conclusion that RNN model is better for our problem, and oversampling is a better way dealing with imbalance, though it's more time consuming. And also, we find our problem is extremely hard and cannot be achieved by a easy model

1. Introduction

Quora is a question-and-answer website where questions are asked, answered, edited, and organized by its community of users, which was co-founded by former Facebook employees Adam D'Angelo and Charlie Cheever in June 2009. Users post questions within Quora, and at the same time, other users can answer questions, upvote for or comment against others' answers. The answers will be sorted by the result of voting in its default state. These interactions naturally select the best answer for a question. In the past years, there is an exponential growth in the community of users of Quora.

Part of the appeal of Quora might be that it is more social than other specialized platforms like Stack Overflow which only accept questions about programming. When asked why Quora was choosen as the name of the website, Cheever stated, "I associated it with quorum or public congregation." As the origin of its name indicates, Quora has a social network backbone that nicely integrates its user base. Users can easily share questions and answers with their social networks through social networking sites like Twitter and Facebook to give them more exposure. Also, questions on Quora are organized by topic and by questioner. Users can meet like-minded friends under the topic they are interested in as a result.

The other part of the appeal of Quora might be that the framing of questions posted on it is not necessary to be formal[5], and the answers can just be some opinions instead of facts. The informality offers a relatively relaxed atmosphere for users to share insights but it also poses a problem necessary to solve how to handle toxic and divisive contents. How to recognize insincere questions from such a large set of questions posted by users is a crucial part of it. The kaggle competition provided some features of these insincere questions as follows[1]. First, they may have a non-neutral tone. For example, there is a question asking, "Which babies are more sweeter to their parents? Dark skin babies or light skin babies?" It is full of racial discrimination, and obviously an insincere question. Second, they may be disparaging or inflammatory, like the question "Has the United States become the largest dictatorship in the world?" Moreover, some of them are not grounded in reality, or use sexual content for shock value instead of to seek genuine answers.

The project aims to use some deep learning methods to propose a classifier to distinguish these insincere questions from others.

2. Related Work

2.1. Glove

Pennington et al.(2014)[6] proposed a global log-bilinear regression model GloVe for the unsupervised learning of word representations based on recent log-bilinear prediction-based methods like word2vec[3]. It has good performance on word analogy, word similarity, and named entity recognition tasks.

We use this as our pre-trained embedding model for classification, which would provide us with word vectors representing words correlation [figure1] .

[1] https://www.kaggle.com/c/quora-insincere-questions-classification

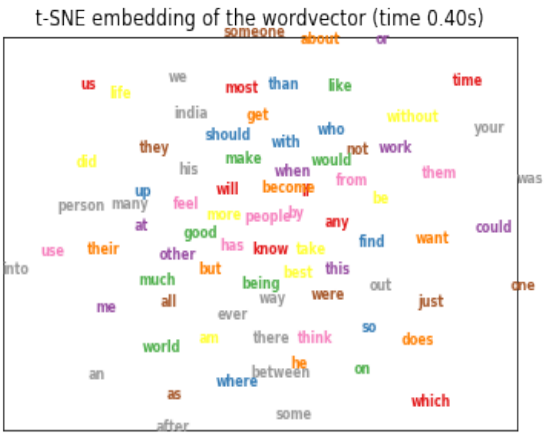

Figure 1. Embedding

2.2. CNN for text

The structure of CNN is based on "Convolutional neural networks for sentence classification".[1].

It gives us lots of insight of how to fully extract information from sentences using CNN.

2.3. RNN for text

In terms of RNN model, we use Pang B, Lee L. Opinion mining and sentiment analysis[J]. Foundations and Trends in Information Retrieval, 2008, 2(12): 1-135. And it do some basic sentimental analysis and opinion mining, and is helpful to us[4].

We also use attention mechanism, which comes from Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems.[7]

2.4. Dealing with imbalance

We have a skew data distribution, so we need techniques to deal with such problem. We refer to Liu Y, Loh H T, Sun A. Imbalanced text classification: A term weighting approach[J]. Expert systems with Applications, 2009, 36(1): 690-701.[2]

3. Proposed Method

3.1. Preprocessing

3.1.1 Text cleaning

- Remove all punctuation.
- Replace all digits with #.
- Replace all abbreviation with their full versions.
- Pad the sentences and make equal all sentences

3.1.2 Load Embedding

After tokenizing the sentences, we use the pre-trained word embedding to convert each tokenized sentence to a time series of embedding vectors.

And during training, we set the embedding parameter fixed inside our neural network class, because it would be too expensive to compute.

3.1.3 Data Augmentation

We use spatial dropout to randomly dropout some percent of embedding values, this method would help us avoid depending too much on individual or some dimensions of embedding.

3.2. Handling imbalance data

Because our dataset is extremely imbalance, if we just use the original data, the samples with label 1 are completely masked. we may have to take some measure to deal with this issue. Here we use two methods – oversampling and weighted loss function.

3.2.1 Oversampling

Here we use method of oversampling to alleviate the imbalance problem by sampling more positive examples, the method is robust according to some reference.

Specifically, we use some kind of techniques to change the structure of our training data, the main method is to duplicate positive samples in our dataset, and make the training data almost balance in every batch.

3.2.2 Weighted loss function

Alternatively, we can use weighted loss function, in mathematics, it just gives a weight to the first term in the BCEloss.

$$loss_i = w_i y_i \log(p_i) + (1 - y_i) \log(1 - p_i)$$

This method seems equivalent with the first method in mathematics, however in DeepLearning, they would have different effect. And naturally, this method would also be faster in each epoch.

3.3. CNN

We apply different filters on our time series data and concatenate them, which outputs the multi-channels data with each channel showing the convolution results of different length of adjacent words. We also apply a filter to this output to abstract all these features. We also use batch norm and dropout to those two convolution steps. Later we flatten the output into an one-dimensional tensor and then output the logits using fully connected layer.

The computation graph is as follows [figure2]:

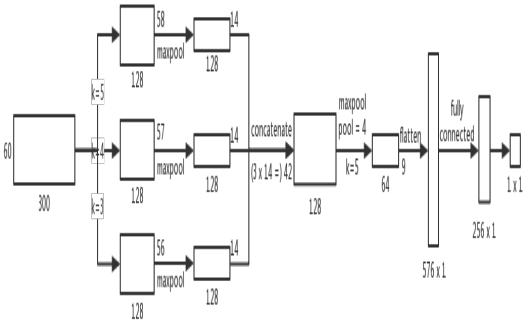

Figure 2. CNN Architecture

3.4. RNN

3.4.1 Architecture

We first use bidirectional GRU and LSTM to process our time series, where the GRU and LSTM are used sequentially. Then we use both the outputs of GRU and LSTM to apply attention mechanism , average pooling along with the maxpooling to extract both the global and local information of our questions, and then for every sample, we concatenate these four one-dimensional tensors to form a long tensor. So we can use this long tensor to compute logits. The whole architecture allows our model to read like a human.

The computation graph is as follows [figure3]:

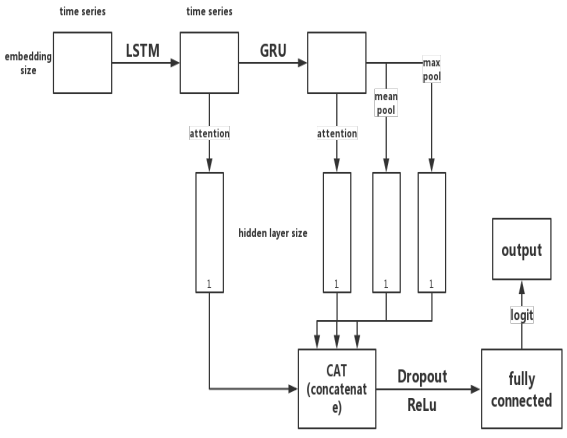

Figure 3. RNN Architecture

3.4.2 Attention mechanism

Attention mechanism is a method to give weights to every time step in our time series data. The intuition is that in a sentence, some parts are important, others are trivial. So the weights learned in attention can easily solve this problem [figure4].

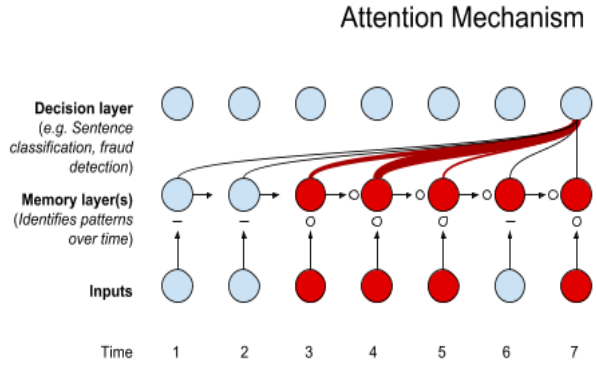

Figure 4. Attention mechanism[1]

Specific in the attention class of our model, we use one step of matrix multiplication which still keep time dimension, and then use Softmax to calculate the weights, and do a weighted sum of different time steps to output a 1-dimension tensor (Here we don't count the batch size dim)

4. Experiments

Describe the experiments you performed. You may want to create separate subsections to further structure this section.

4.1. Dataset

Our dataset is from Kaggle competition[2]. Each sample in our dataset contains id, question text and its corresponding label.

Validation We use 10000 positive and 10000 negative samples as our validation set, and we make it balance to better evaluate the performance of our model by accuracy.

Training We have training set which is extremely imbalance, the ratio between positive and negative example is about 17:1 (1,215,312:70,710).

[1] https://skymind.ai/wiki/attention-mechanism-memory-network

[2] https://www.kaggle.com/c/quora-insincere-questions-classification

Test We don't have test set, because it would make our imbalance problem even more serious. Actually in the competition, there's a test set without labels, but it's not helpful to our project.

4.2. Preprocessing

In the preprocessing:

- We set the maximum words to load from embedding 200000.
- The length of sequence of each question is 60.
- The embedding size of each word is 300.
- Batch size of training is 1024.
- In data augmentation, After some transformation, we use 2D Dropout method, and drop rate is 0.2.

4.3. Handling imbalance data

4.3.1 Oversampling

We duplicate the positive examples in training set with 17 times to make training set balance.

4.3.2 Weighted loss function

We use the parameter pos_weight in BCEloss function of Pytorch, and set it to 17, which, in mathematics, is equivalent to the previous method. So it makes the two methods comparable.

4.4. CNN

The parameter of each layer and filter is as follows:

1. Apply three kinds of filters to the inputs with kernel widths 3, 4 and 5. Make the channels' number 128.
2. Max pool three outputs with kernel widths 4. Then concatenate them by channels, forming a 128*42 dim output.
3. Apply convolution with kernel widths 5 and max pool the output with width 4 kernel.
4. Flatten the above result to one dimension tensor with 576 units.
5. Use fully connected layer to the tensor and form a one dimension tensor with 256 units.
6. Finally, output the logits with fully connection.

4.5. RNN

The detailed structure is described as follows:

1. Use bidirectional LSTM to process time series with hidden size 70.
2. Use GRU with the same hidden size on the previous output.
3. Use Attention to both output of GRU and LSTM, which gets two $2 \times 70 = 140$ length 1-dimension tensors.
4. Use maxpooling and average pooling on output of GRU.
5. Concatenate above four tensors $(4 \times 140 = 560)$ and use fully connected layer(560×40) with dropout rate 0.2, then apply relu function.
6. Finally, output the logits with fully connected layer.

4.6. Software

We use python as our only programming language, and use packages like torch, matplotlib, keras.

In terms of computation resource, we only use Google cloud platform and use Tesla P100 as our eGPU resource.

4.7. Hardware

We use regular PCs and laptops, and eGPU from Google cloud.

5. Results

5.1. Using the original data

5.1.1 Results

We first use original data to do some experiment without any method dealing with the imbalance. This part is a control group which can be compared with other group.

The learning curve is as follows [figure5]:

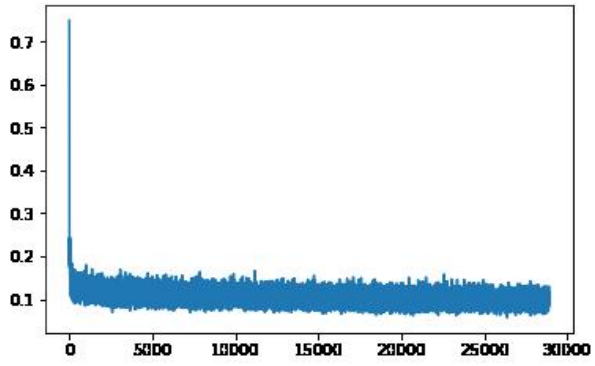

Figure 5. Without dealing with imbalance (loss - minibatch)

We can see that the training loss decreases sharply in the very beginning. Although the training accuracy is more than 95%, we found that nearly all 0s can be correctly classified, but in terms of example with label 1, the classification is not good.

5.1.2 Analysis

So we find that any kind of classifiers, even linear classification, can achieve such an accuracy (It just need to classify all examples to 0). – Because algorithm would only learn how to reduce the loss of majority (class 0), so in such a imbalanced dataset, the class 1 is completely masked.

And naturally, the accuracy in our balanced validation set is between 50% and 60%, which also validate our thought (correctly classify all 0s and doesn't consider 1s).

5.2. CNN

5.2.1 Using oversampling

Using oversampling technique, we have learning curve (loss versus minibatch) as follows [figure6]:

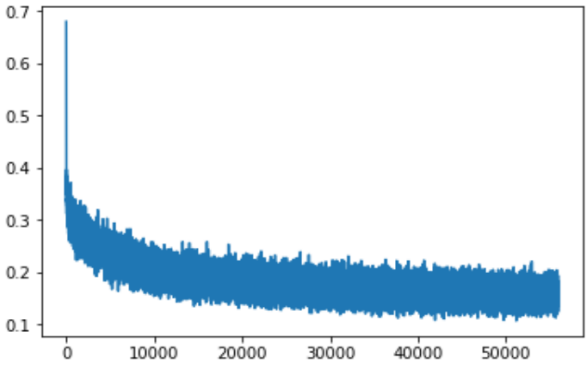

Figure 6. CNN using oversampling (loss - minibatch)

The validation accuracy is 88.02%, and the validation loss is 0.3246.

5.2.2 Using weighted loss function

Using weighted loss function technique, we have learning curve (loss versus minibatch) as follows [figure7]:

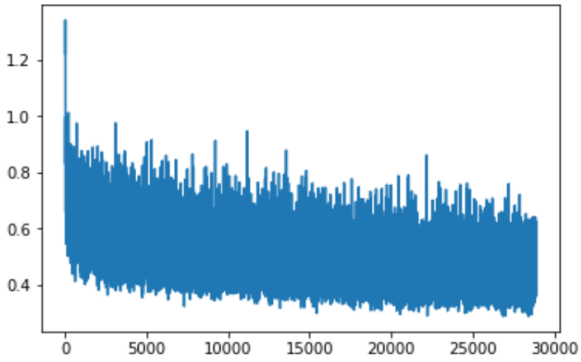

Figure 7. CNN using Weighted loss function (loss - minibatch)

The validation accuracy is 86.56%, and the validation loss is 3.3843.

5.3. RNN

5.3.1 Using oversampling

Using oversampling technique, we have learning curve (loss versus minibatch) as follows [figure8]:

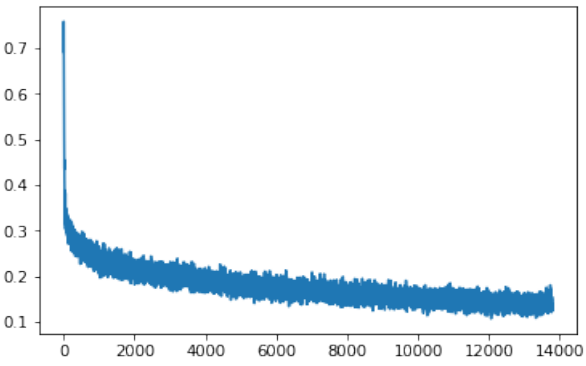

Figure 8. RNN using oversampling (loss - minibatch)

The validation accuracy is 92.1%, and the validation loss is 0.1523.

5.3.2 Using weighted loss function

Using weighted loss function technique, we have learning curve (loss versus minibatch) as follows [figure9]:

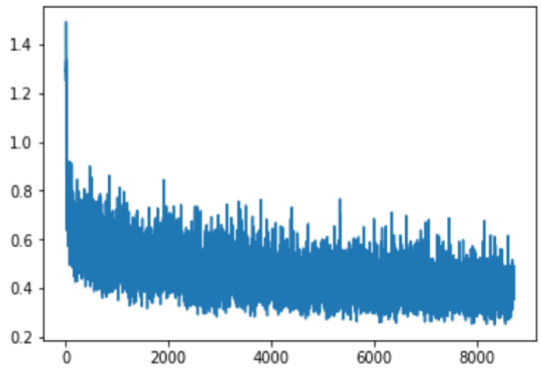

Figure 9. RNN using Weighted loss function (loss - minibatch)

The validation accuracy is 92.0%, and the validation loss is 0.4012. But we cannot compare this loss to the loss in oversampling method.

6. Discussion

6.1. Comparison

We put results together to make it clear.

	val loss	val accu
CNN+oversampling	0.3246	88.02%
CNN+weighted	3.3843	86.56%
RNN+oversampling	0.1523	92.1%
RNN+weighted	0.4012	92.0%

Then we can see that the two methods for dealing with imbalance data don't have significant difference. But RNN model is much better than CNN model.

- The first reason behind this is two methods are equivalent in mathematics, one duplicate the label 1 questions, the other gives the loss term a high weight. But during training, the oversampling method is much more stable, and perform better, but also much more time consuming..
- And reason for second statement is that RNN takes time into account, but CNN just considers the cluster of words.

6.2. Testing and reasoning

We plug some obvious insincere questions into our model, like:

1. "Is our professor in STAT479 most handsome and kind guy in the University of Wisconsin-Madsion? (predict = 0)
2. "Can anyone be my girlfriend? My phone number is 6083321444, contact me plz. (predict = 0)

But our model decides they are sincere with high confidence, so we think our model just learns something common in our dataset, but doesn't really know how to distinguish these two kinds of questions.

7. Conclusions

- After all the discussion and experiment, we finally find a good classifier for our text data, which can achieve satisfying accuracy, so we have accomplished our initial goal.
- In this project, we know that the RNN model is more suitable for text classification problem, which is significantly better than CNN architecture .
- When it comes to the techniques to dealing with imbalance data, the method of oversampling is much more stable than the method of weighted loss function but also more time-consuming.

8. Deficiency and future direction

- However, we find our model doesn't know how to truly distinguish two kinds of questions, and it actually just detects some common features of label 1 questions in our dataset (Such questions may have specific form in Quora data, but cannot generalize) , which is far from enough.
- Our problem is actually very tricky and ambiguous, because the label of a question doesn't depend on specific word or even sentence, and even human beings cannot reach agreement in certain classification results. This also tells us very simple architecture wouldn't work very well.
- So we think we cannot just extract superficial information like that, and we may need powerful tools or complex structure to truly find what a insincere question looks like. And creating such a tool or structure perhaps needs to fully combine the knowledge in language with DeepLearning theory.

9. Acknowledgements

Thanks to our professor Sebastian Raschka, we get to know how to use Google cloud platform and got free coupons, which is really important for our project.

10. Contributions

- Richard Yang is responsible for the data preprocessing, coding RNN model, and writing framework of report.
- Junyao Wang is responsible for the CNN model, and writing the CNN part in report.
- Qiaochu Yu is responsible for tuning the parameters, analyze model and write the rest part of the report.

Everyone contributes much to our project, and everyone is indispensable.

References

[1] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[2] Y. Liu, H. T. Loh, and A. Sun. Imbalanced text classification: A term weighting approach. *Expert systems with Applications*, 36(1):690–701, 2009.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[4] B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[5] S. Patil and K. Lee. Detecting experts on quora: by their activity, quality of answers, linguistic characteristics and temporal behaviors. *Social network analysis and mining*, 6(1):5, 2016.

[6] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.