QBC-CNN: A Semi-Supervied Learning Based CNN Model For Toxic Comment Classification

Siyan Cheng
University of Southern California
Los Angeles, CA 90007
siyanc@usc.edu

Laksh K. Matai
University of Southern California
Los Angeles, CA 90007
lmatai@usc.edu

Tong Liu
University of Southern California
Los Angeles, CA 90007
liu553@usc.edu

Abstract

Discussing things you care about can be difficult. The threat of abuse and harassment online means that many people stop expressing themselves and give up on seeking different opinions. One of the best ways to avoid that is by moderating the toxic comments. We aimed on Toxic comment analysis dataset on Kaggle. This is a multi-label dataset with highly imbalanced data, we tried logistic regression, random forest, gradient boosting and Latent Dirichlet Allocation(LDA) to identify and classification the toxic comment and compare their final result. We also applied deep learning approach on this problem, we implemented LSTM and vanilla CNN at first, these two models gave us a very competitve result. Finally, we built a semi-supervised CNN model inspired by the ideaQuery-by-Committee(QBC). This model gave us excellent result with only 1% of the original data and more importantly it's training time decrease significantly.

1 Introduction

Internet is an open platform for everyone to present their views in the world. A decade before, the question was who gets the chance to speak? This was because the penetration of internet was low in rural parts of the world. However, with the advent of mobile phones everyone is online. Now the question is who gets heard? A lot of Individuals are not able to enjoy their right on freedom of expression because of toxic comments used by other people. Threatening and insulting someone online could permanently hamper their online presence. One of the best ways to avoid that is by moderating the toxic comments. If we are able to classify online which comments are toxic we can remove them and give voice to every individual.

In this project, we trained multiple machine learning algorithm to classify the toxicity in the comment. We are going to use multiple methods, such as logistic regression, random forest, gradient boosting and Latent Dirichlet Allocation(LDA) to achieve the goal. This project is also the part of Kaggle competition which is hosted by Jigsaw (An Alphabet company).

2 Problem Description

Toxic comments show up in many websites, which does harms to the Internet environment and some of them should be deleted. The problem publisher has collected a large number of Wikipedia comments had been mannually labeled by human raters for toxic behavior. The types of toxicity are: *toxic, severe_toxic, obscene, threat, insult and identity_hate.* The following line is from the Kaggle dataset that we used.

"How dare you vandalize that page about the HMS Beagle! Don't vandalize again, demon!"

The above sentence is an example of a threat.Our target is trying to correctly classify all the comments in the test dataset. We will build several different classification models to classify the comments.

2.1 Dataset

The dataset used is provided by Kaggle for the competition. The dataset is of comments from Wikipedia's talk page edits. The dataset can be found at following address.

`https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data`

There are 6 types of labels and they are all one-hot encoded. Here, we compare the count of different labels vs rest of the labels by plotting a bar graph. As we can see from the graph, the dataset is hugely imbalance.

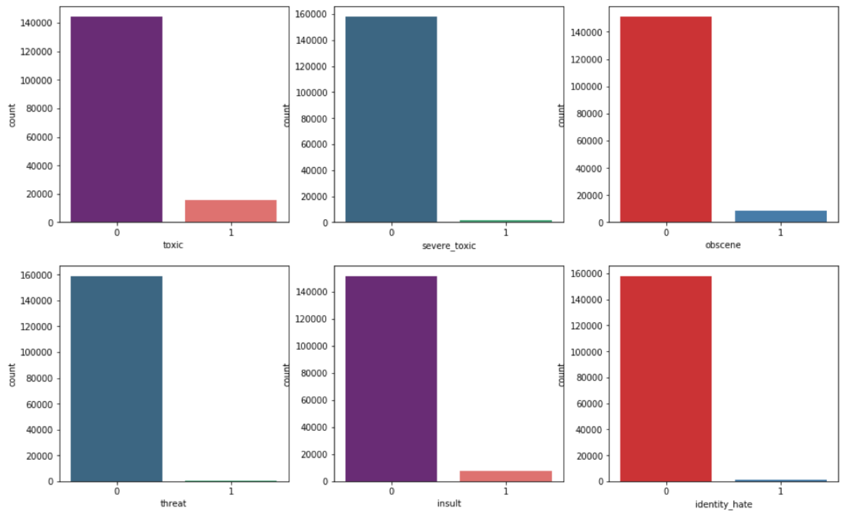

Figure 1

2.2 Multi-label Solution

In our data set, a single data point is associated with a set of labels simultaneously. Hence, we have to find a way to solve multi-label classification problem. One method is Classifier Chains, it combines the computational efficiency of the Binary Relevance method while still being able to take the label dependencies in to account for classification. Basic idea for method is Fit multi-label data to a chain of binary classifiers, "chain" means one classifier have relation to its former and continuous classifier. SVM and Naïve Bayes can be use directly to implement this method. But considering there are 1,70,000 data points, it runs so slow.

Another useful way to do Binary Relevance(BR)[1], fit multi-label data to many binary classifiers. If there are Q labels then Q different classifiers are needed. For every instance in our dataset, we have to combine all the Q labels as one final prediction. Then calculate the hamming loss for the test dataset as final accuracy. We use logistic regression classifier to solve that problem.

3 Traditional Machine Learing Models

In this problem, we are going to use multiple methods and make the comparison between them. After having word embeddings, we can attempt different methods on them. We tried logistic regression, random forest, gradient boosting and Latent Dirichlet Allocation(LDA) to identify and classification the toxic comment and compare their final result.

3.1 Logistic Regression

We build 6 logistic regression models for the 6 labels. For every model, we use cross-validation to select best C(inverse of regularization strength). Because all labels are binary, so logistic regression is very suitable for this situation.

3.2 XGBoost

XGBoost by T Chen et al. [2] is short for "Extreme Gradient Boosting", which is a distributed gradient boosting library. It is a supervised learning based on gradient boosted trees, for every step, it just uses a subset of instances and features. Compared with SVM and Naïve Bayes, is more efficiency, which allows us to adjust the hyper-parameters by model performance.

3.3 Random Forest

Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

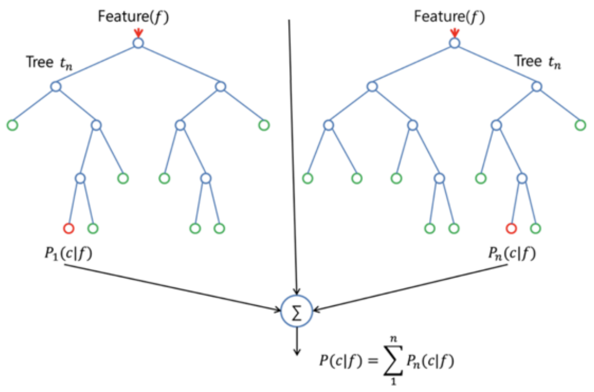

Figure 2

3.4 Latent Dirichlet Allowcation

Latent Dirichlet Allowcation by Blei D M et al. [3] is generative model that is frequently used in natural language processing field, especially in sentimental analysis. It is a unsupervised learning method that can automatically discover the topics that the data contains. We utilize this model by manually specify the number of topics(number of clusters) to perform clustering on our dataset, which gave us a decent result and a far less training time.

4 Deep Learning Models

4.1 Long Short-Term Memory Network

As a human, our thoughts have persistence. We don't start every task in our daily lives from scratch. We have the ability to use the learnings from yesterday, today and also in future. For example, when we read a book, we don't start thinking about the words from character level. We learnt about letter in kindergarten and sentences in primary school, and we use that learning even today. Unfortunately, conventional Machine Learning algorithms start thinking from beginning every time we train them;

they lack persistence. Recurrent Neural Networks is a Deep Learning model which helps us solve this problem.

Recurrent Neural Networks may perform well when the gap is small between two layers. When the gap increases the information is last, this problem is known as Vanishing Gradient.

Long Short-Term Memory (LSTM) networks are a special kind of RNN, capable of learning long-term dependencies. LSTMs have 4 types of gates Input, Output, Forget and Cell. Cell acts as a memory and helps carrying forward the information.

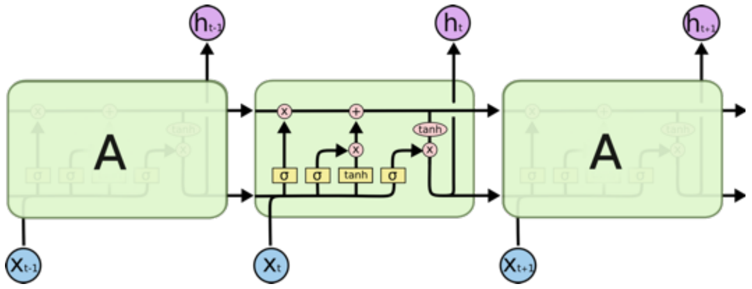

Figure 3

4.2 Convolutional Neural Network

Convolution Neural Network has brought tremendous breakthrough in both Computer Vision and Natural Language Processing in recent years. In this project, we build and train 3 different CNN models to classify toxic comments and compare their result with LSTM and traditional machine learning techniques that we mentioned before.

CNN with Binary Revelance(CNN-BR) We first build 6 different CNN models for each label in the dataset, and combine the prediction of each label as final result for evalution by Binary Revelance. For each model, we build same convolution structure for them to maintain generality.

CNN for all labels(CNN) The result that CNN with BR gave is not desirable enough, so we improve our model by change the output layer from 1 neuron to 6 neurons. By doing that, this model can train for data that with 6 labels thus remove the assumption that each label is independent to each other.

QBC-CNN We also build a semi-supervised CNN model for this problem by using query-by-committee(QBC) method. We design 8 different CNN models as the committee by diversifing their feature map. In Natural Language Processing task, the convolution process is mostly conduct in 1 dimension, so by varying the feature map, we can build models with similar architecture but unsimilar data flow.

5 Experiment

5.1 Data Preprocessing

Input for the model is a sentence in English. The algorithm on the basis of words used will try to predict whether the sentence is toxic or not. If found that the sentence is toxic, it'll try to classify the toxicity. First of all, we need to transfer our sentences into numerical vectors/matrices so that we can utilize them to make computation.

TF-IDF Sparse Matrix For traditional machine learning algorithms, we feed the model with TF-IDF sparse matrix that trained on our entire training set. The reason we use TF-IDF instead of pre-trained word embedding is that traditional machine learning techniques can only accept 1 dimentional input for each training sample without explicit dimension reduction techniques. In our model, we remove the stopwords, the vocabulary size(number of features) is 20,000 and use (1,3) as n-gram range.

Word Embedding based on Word2Vec In the previous work of Mikolov et al. [4], they proposed the word2vec model that can transfer the sentences into word embeddings. Pennington et al. [5] also proposed GloVe – a brand new model for learning vector space representation of words recently based on word2vec which outperformed most related models. For deep learning techniques, we choose publicly available GloVe pre-trained word vectors to project our sentence into vector space. The dimensionality of our word vector is 300 and for those words do not appear in the pre-trained word embedding, we skip them for their low frequency of occurance. In our model, the vocabulary size is 10,000 and the maximum length of sentence is 200. We truncate those sentence with a length more than 200 and pad the sentence with 0 value when the length is less than 200 to make sure all sentences have the same size.

5.2 Models and result

Logistic Regression Here we implement and train logistic regression with L2 regularization. We choose the best penalty coefficient by conducting 5 fold cross-validation. We combine the output of each label and use Binary Revelance accuracy as the evaluation.

Random Forest We've used SkLearn's RandomForrestClassifier to create our model and train it. We also performed cross validation on 6 values of parameter 'C' which were [.01, .1, 1, 10, 100, 1000]. We received an average accuracy of 91.4% and an Area Under the Curve of 93.4.

XGBoost XGBoost provided several useful parameters. 'Eta' is step size shrinkage used in update to prevents overfitting. 'max_depth' is maximum depth of a tree. 'subsample': 0.8 is the subsample ratio for every layer. 'colsample_bytree' is subsample ratio of columns for each split. 'binary:logistic' suggests we are using logistic regression for binary classification, output probability in each level. After experiments, we set the 'max_depth' depth of the tree set as 5, 'Eta' as 0.3, 'colsample_bytree' as 0.8, 'subsample' as 0.8 to fit our model. Final accuracy rate for XGBoost is 91.92%.

LDA As the value of each label is binary, we set the number of topic(number of clusters) to 2. Then we perform clustering to each label for 6 time and combine the result they predict. Here we also use Binary Relevance as the metric of evaluation.

LSTM We started our experiment by initializing a LSTM layer with an activation function of hyperbolic tangent, with 0 dropout and 60 output neurons. Then we added a pooling layer, we used GlobalMaxPoolinf1D for performing global max pooling operation for temporal data. Global max pool doesn't take the size of pooling layer as parameter and therefore returns the global maximum value.

We then added a dropout layer, which in random fashion sets a fraction of its input to 0. We performed dropout on 10% of the inputs in this layer. Another layer after it is a densely connected NN. Here the activation function used is 'ReLu' and the number of output neurons is 50. We add one more layer of drop out and a dense layer in the similar fashion as we added above.

This is the model we constructed to achieve our goal. This model took around 4-6 hrs to train on a normal PC and around 40 mins on a cloud GPU provided by Paperspace.

CNN-BR The structure for models trained on different label is the same. The main components in sequence are: (i) 1 dimensional convolutional layer with fixed filter size(5); (ii) 1 dimensional max pooling layer with fixed filter size(5); (iii) 1 dimensional convolutional layer with fixed filter size(5); (iv) global max pooling layer to extract the information and flaten the data to feed the neural network in the following; (v) fully connected layer with 128 units; (vi) output layer with 1 neurons.

CNN We build a similar model as CNN-BR by only changing the output layer to a dense layer with 6 neurons.

Table 1: Result

Model	Accuracy	ROC-AUC
Logistic Regression	87.84	96.23
Random Forest	91.48	93.47
XGBoost	91.92	96.69
CNN-BR	81.93	50.21
CNN	97.90	96.25
LSTM	**98.15**	**97.31**
LDA	86.05	N/A

5.3 Learning with less training data

After implementing so many models, we started to think: do we need all data to train our model? Or we can get a decent result just by training our model on a subset of the data. Inspired by Query-By-Committee semi-supervised learning algorithm, we build our QBC-CNN model. In this model, we create 8 different simple CNN model as the committee. These 8 models have similar artchitecture, their main components are: (i) 1 dimensional convolutional layer; (ii) global max pooling layer; (v) fully connected layer with 128 units; (vi) output layer with 6 neurons. For each model, we vary the size of convolution kernel from 2 to 9. Intially we have 1000 data points in the labeled data pool, and repeat the algorithm by following 2 steps: (i) we train these 8 models on labeled data pool; (ii) predict the label of unlabeled data and choose 1000 data points with the most confliction by committee.

For comparison, we also build a CNN model that have same architecture as QBC-CNN. We set convolutional kernel size as 5, and just randomly choose 1000 data point in the unlabeled data and add them to labeled data pool in each iteration.

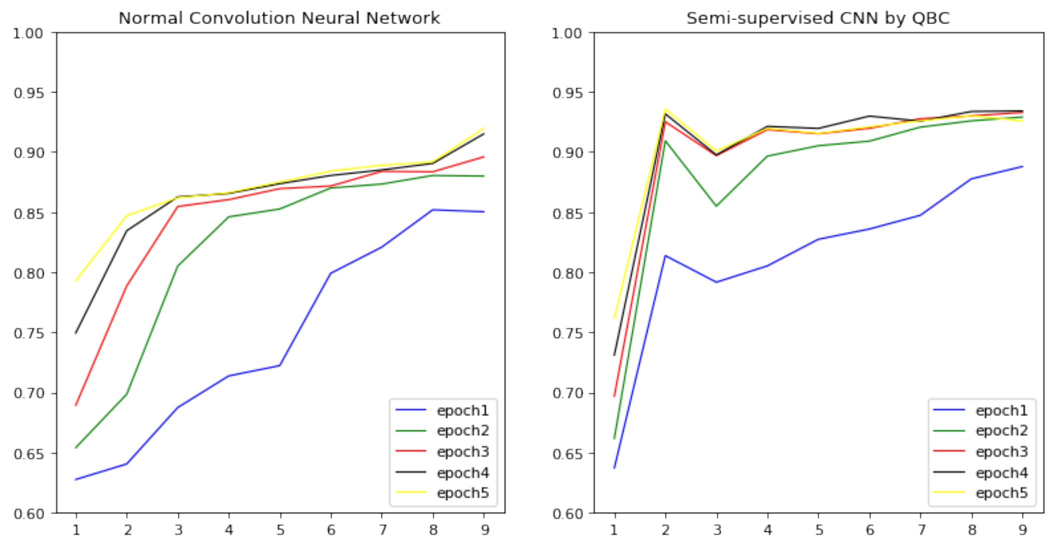

Figure 4: From this figure, we can see that from 1000 data points to 2000 data points, the ROC-AUC score of QBC-CNN increases shapely, and get an accuracy near 94%; but for normal CNN, it increses slowly and with the help of 9000 data points, it get a ROC-AUC score around 92, which is not even as good as QBC-CNN trained with 2000 data points.

6 Conclusion

We performed toxic comment analysis with the help of various models. The most important part for most of the models other than CNN was the preprocessing part, where we used Term Frequency Inverse Document Frequency. It helped us get a numerical form of data from the textual form initially provided in input. This numerical form was further used for training all the models. Once we figured out the common process of training models, we shifted our focus from just solving the problem to comparing different models. We compared 3 major aspects – Accuracy Score, ROC- AUC score, Computation Time. In the end, we found out that LSTM provided the best combination of Accuracy Score and ROC-AUC score, but with a cost of Computation Time. The timed required to train a model on an average CPU was approximately 4-6hrs, so we used an online GPU to train our model and it took us around 30 mins. While using Latent Dirichlet Allocation(LDA) we got a decent accuracy score of around 86% and a rapid computation time of 40 sec. Now, this leaves us on two paths for future work. First, make LDA more efficient and receive accuracy around 95%. Second, to create a method of ensemble learning using LDA and maybe Logistic Regression or another model, to get a high accuracy score with a smaller computation time.

References

[1] Zhang M L, Zhou Z H. A Review on Multi-Label Learning Algorithms[J]. IEEE Transactions on Knowledge & Data Engineering, 2014, 26(8):1819-1837.

[2] Chen T, Guestrin C. XGBoost:A Scalable Tree Boosting System[C]// ACM, 2016:785-794.

[3] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. J Machine Learning Research Archive, 2003, 3:993-1022.

[4] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[J]. 2013, 26:3111-3119.

[5] Pennington J, Socher R, Manning C. Glove: Global Vectors for Word Representation[C]// Conference on Empirical Methods in Natural Language Processing. 2014:1532-1543.