Sentiment Analysis Using Convolutional Neural Network and Modal Distribution Removal

Xi Chen

Research Scholl of Computer Science, Australian National University

Abstract. This paper describes a method for removing noisy data according to multimodal error distribution while training neural networks. The method expands existing work from bimodal removal to multimodal removal, identifies data that contribute to the frequency peak of high error rates and removes them from the training set iteratively. We justify the effectiveness of our method by comparing it with plain convolutional neural networks and Least Trimmed Square method. We also present performance comparisons between our method and an existing study of a sentiment analysis problem. Experimental results prove that our method is efficient in terms of reducing convergence time and achieves great prediction performance.

Keywords: multimodal removal, convolutional neural network, noisy data detection, sentiment analysis

1 Introduction

Artificial Neural Networks (ANN or NN) are connectionist systems inspired by biology, which can be used to simulate brain to solve problems given learned knowledges. In recent year, there have been a vast range of application for NN in many fields such as computer vision, machine translation, etc. As more data are open to public in the era of information, quantities of involved dataset generally increase for machine learning. Also, the growth of demand contributes to high diversity and high complexity of neural network. These factors widen the domains in which NN can be employed. However, they may also result in time-consuming training processes. Along with this issue, high quality of training set extracted from large data cloud cannot be guaranteed, which also makes model generation inefficient, and even have negative influences on training result.

Such issues potentially decrease the productivity of academic study and real data analysis like marketing prediction. Researchers have developed a series of techniques to improve the training efficiency and effectiveness of NNs. One approach is to modify NN structures, which includes reducing free parameters by changing NN topology [1] and pruning trivial neurons through mathematical analysis [2]. However, these ideas present powerlessness in terms of handling data sensitivity. The tradeoff between simplification of NN structure and test performance is also hard to determine. To tackle these problems, some proposed to automatically remove noisy training data when building models. Least Trimmed Squares method (LTS) [3] iteratively remove noises based on their mean square error and have shown a good result on performance. But it misses handling with noisy testing data and halting issues. Here we apply a modal distribution removal method based on Slade and Gedeon's paper [4] which solve these subproblems ideally and show to what extent it can improve the performance of neural networks.

Above papers generally test the training data refinement approaches using simple feed forward neural networks. whereas, more complex problems and network structures have larger demand for modification as they can achieve high-level artificial intelligence programmes in more domains and are usually time consuming. Thus, we will use deep learning method to test the noisy data removal methods in this study, given a sentiment analysis problem. One of the popular approaches for solving text categorization problems is long short-term memory (LSTM), which allows for different lengths of training samples and is able to capture remote context information when learning, such as Johnson's work [5]. Nevertheless, a study [6] has proved that when key phrases are more significant than relationship among the overall text, convolutional neural network (CNN) which focus on pattern extraction usually yields better performance than LSTM.

To illustrate the feasibility of our method on complex real-world problems, we use online reviews dataset [7] collected from three different websites to predict whether sentences in reviews convey positive or negative sentiment. We firstly preprocess the original data to generate training set using natural language processing and word embedding, and train with a convolutional neural network.

We implement our proposed method based on above fundamentals, conduct experiments using our approach, LTS and pure CNN, investigate the classification performance with our method comparing with another modern study using the same dataset [8].

2 Methods

This section will explain the general method for training with noise removal, which will be divided into 3 main parts: method of data preprocessing, the architecture of NN model and the rules of modal distribution removal.

2.1 Data Preprocessing

As raw data provided are plenty of texts with labels while training data for convolution need to be numeric metrices, an appropriate data preprocessing is essential for data learning.

Firstly, sentences are tokenised into lists of words and special punctuations are removed using several regular expressions to reduce noise for training. Lists are padded to the same length which is the length of the longest sentence using '<PAD>' because CNN does not support samples of different length. An operative NN can recognise these padding symbols at most times. An NN will loss its functionality if padding symbols draw lots of attention of it.

We build a vocabulary dictionary for all the data collected, then convert the words in data into corresponding index in the dictionary. Then each training sample will become 1D array, with each element representing one word.

Another issue usually appearing is the imbalance between positive and negative samples. In order to prevent from generating bias to different categories in training model, we use a subset of this dataset, which includes same number of randomly picked samples for each category, for the training process.

After tokenisation, sentence padding, word labelling and data selection, shuffle all samples and apply word embedding for training process.

2.2 CNN Architecture

Structure. The neural network used for sentiment analysis is a convolutional neural network with 1 embedding layer, a number of convolutional layers, a number of maxpooling layers and 1 fully connected layer. The first layer (Input layer) contains neurons with the same number as that of training vector length for accepting input attributes. The output layer consists of neurons with the same number with category number, indicating the probability of a sample being classified to each class, which is normalised by Softmax. The structure and functionality of the hidden layers are shown in the Fig 1:

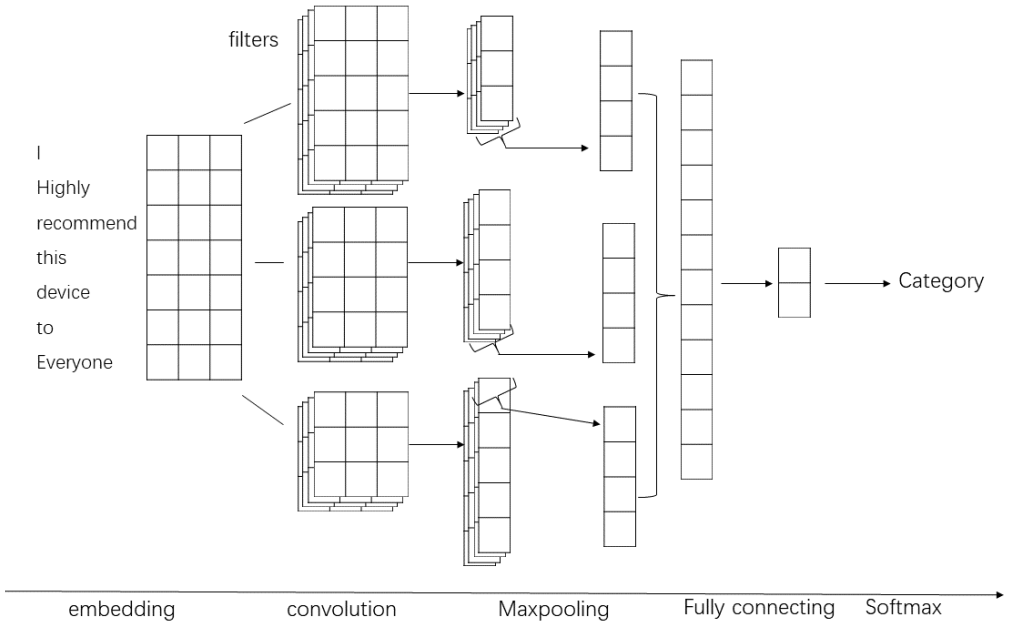

Fig 1: An example architecture of convolutional neural networks for sentiment analysis.

After converting the sentence to a matrix by text preprocessing and word embedding, several filters of different sizes will be applied to the matrix to generate the same number of feature maps. One special thing is that the width of filters is usually equal to the embedding size in text classification problems, which means each filter focus on a series of n-

grams with different weights. (In fig 1, the three kind of filters respectively extract 5-grams, 4-grams and 3-grams.) Training process will change the weights continually and finally most filters tend to extract the most significant phrases which decides the category in the sentences. Maxpooling layer retains the largest values of each feature map. Then the output of maxpooling layer will be concatenated and put into a fully connected layers to generated a positive value and a negative value, which are normalised by Softmax function to indicate the probabilities of which category the sentence belongs to. The formula of Softmax function (2) is as below:

$$\mathrm{P}(\mathrm{y} = j|x) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{1}$$

Where K refers to the number of categories, z_j means the value of the jth element in a vector. Then the category with largest probability will be considered as prediction result.

Training configuration. Training of this neural network is an iterative process with back-propagation. The loss function adopted is mean of Negative Log Likelihood (or Cross Entropy) given by formula (2):

$$H_{y'}(y) = -\sum_{i} y_i' \log(y_i) \tag{2}$$

where y_i refers to predicted category and y_i' refers to labeled category to pattern i. Comparing with other loss functions such as Mean Squared Error (MSE), Cross Entropy guarantees that optimization for our classification task is convex, which yield better convergence properties [9]. We also use Adam optimizer as it is computationally efficient and computes per-parameter learning rate for hyperparameters.

2.3 Multimodal Removal

Slade and Gedeon's paper [4] presented bimodal removal of noisy data from the original dataset. However, bimodal removal is limited to cases where error frequency distributions are perfectly bimodal. In real experiments, this distribution can be multimodal (Fig. 2). Upon that method, we use a multimodal distribution removal method (MDR) for enhancing the efficiency and effectiveness of CNNs.

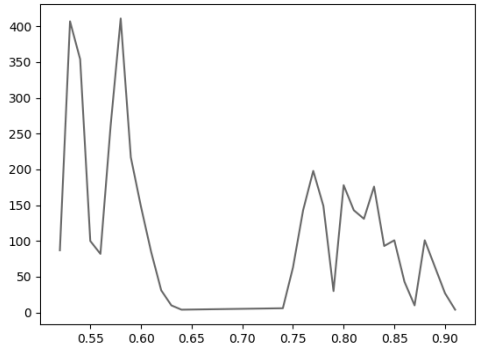

Fig 2. A distribution of Cross Entropy errors extracted from within 100 epochs

The 'peaks' in the error distribution indicate how well the NN have learned from those samples. Occurrences of samples with high error values mean those samples may be noisy data for training, and the frequency reflects how 'bad' those sample are for NN. Those samples identified as noisy data can thus be removed from the sample dataset. To implement this, for every 5 epochs (this may vary according to convergence speed), we collect losses of all current training samples and compute error frequency distribution. Then we identify the peak with highest error rate (note this peak does not necessarily have highest error rate among the whole occurrence of errors) and eliminate all samples generating this error from the training samples.

Removal criteria. If the error rate is not high enough, removal should not be performed as those samples may carry useful information. In our implementation, we use the following equation (3) to guarantee that dominant outliers are eliminated and samples carrying actual information are not accidentally removed:

$$Error \geq \bar{\delta}_{ce} + \alpha\sigma_{ce} \quad (0 \leq \alpha \leq 1) \tag{3}$$

$\bar{\delta}_{ce}$ is the mean cross entropy, σ_{ce} is the standard deviation of errors. α value is 0.8 in this experiment. The idea behind this is to measure the destructiveness of outliers by how much their errors exceed the overall level of errors.

Terminating criteria. Removals cannot be performed infinitely as the training data will shortly contain insufficient samples for training (An extreme case is all data have been eliminated). As the outliers are removed, the variance over losses will be reduced and ideally converges to 0. Thus, we intuitively use this as the terminating criteria of the removal process, halt training until the variance is lower than V. We use V =0.01 in our experiment.

3 Results

3.1 Experimental Environment

Our method is implemented in Python 3.6 with Pytorch framework (torch 0.4.0). All tests are performed in a MacOS machine with 2.5GHz Intel Core i7 CPU and 16 GBytes memory.

3.2 Dataset

We use online reviews dataset for training and testing, which are extracted from three websites to guarantee generalization of the sentiment analysis model. The goal is to predicate whether a review is positive or negative [7]. This dataset contains 3000 instances (i.e. 1000 instances for each websites). The number of instances with positive labels and negative labels are both 1500. All data were randomly divided into training, validation and testing parts, with proportion 0.8, 0.1, 0.1 respectively.

3.3 Performance Justification

This section compares the training performance by CNN with MDR, CNN with LTS and pure NN in terms of converge time and noise removal efficiency.

Fig 3 illustrates the effect of our method to the convergence by epochs. Training neural networks need optimization of a large number of parameters, so the time efficiency matters. After applying our method, the variance of losses during training dropped to below 0.01 within 150 epochs. The underlying explanation is that in the gradient descent process, we update parameters through the direction to where their derivatives drop fastest. In normal CNN model, this induces a problem that disruptive data may affects parameters so that they are not changing toward the direction that optimizes the model. After removing those noise (if they are actually noisy), the optimization process takes less time as the update of parameters are better without interference. Besides, LTS also have effect on variance reduction, whereas, the efficiency is lower because it only eliminates samples with the least error, whose quantity may be rather small. Another untrivial disadvantage of LTS is the variance of loss keep decreasing once the overall weights tend to be stable and never stop which means lots of useful data will also be discarded because there is no halting criterion for training.

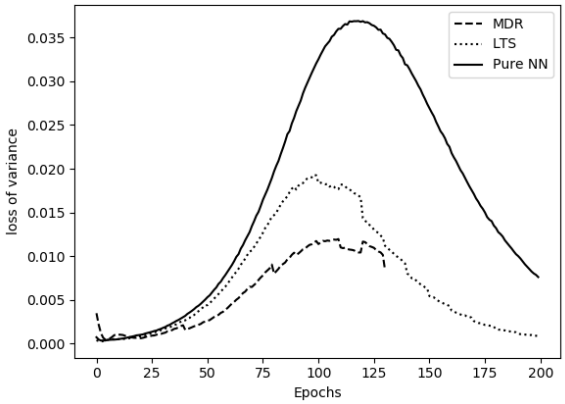

Fig 3. Change of loss of variance using MDR, LTS and pure NN

Noisy data for deep learning are similar to outliers in relatively 'shadow' machine learning problems. Outliers are identified by abnormal position in attribute space, while in our approach they are featured by contributing to high error rates. If they truly carry only disturbing and confusing information, removal makes sense as it helps the gradients descent quicker through correct direction.

However, generating a high error rate does not necessarily indicate that samples should be discard as they may carry new features at the early stages of training or some relatively rare but cannot-be-ignored information. This phenomenon is significant in real-word data, especially those with a great number of attributes. Fig 4 shows the trade-off between the converge time and accuracy.

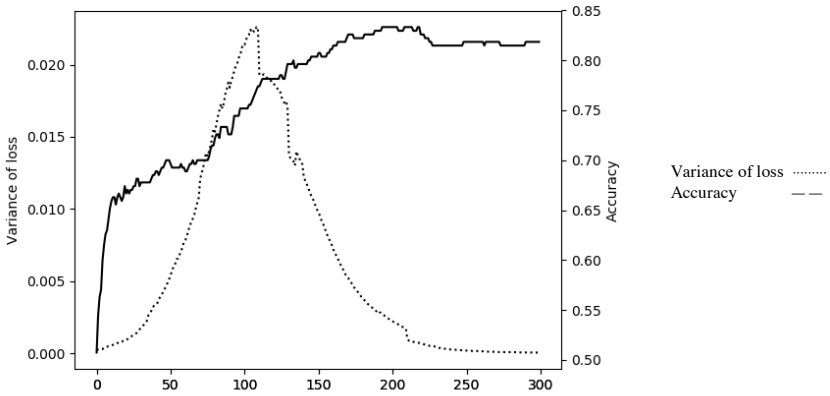

Fig 4. Comparison between change of variance of loss and accuracy

As we drop out more samples, the variance between losses tends to reduce and eventually converge. However, from the figure we can see that accuracy during training is not always positively correlated with the variance, and even decreased in some specific epochs. This result reveals the fact that we loss information when we discard data, even they are identified as noisy by our method. Thus, effective and accurate identification of noisy data is critical to methods like ours.

3.4 Result Comparison

Kotzias et al. investigated prediction of review sentiment using CNN and different vectorization method, namely embedding, BOW and a new vector representation approach considering similarity among sentences and test the performance both on instance level and group of instance level. In this section, we show that the model built by using our method yields better results measured by ROC and Accuracy than CNN with embedding and BOW method on the same dataset. As there are only 100 instances of data from each website, we combine them together for training and test them separately.

Receiver Operating Curve (ROC). ROC is generally used to evaluate the performance of a binary classifier. The ROC of our model and the compared model is shown in Fig 5. The label of x-axis true positive rate (or recall values) is a retio between number of true negatives and the number of wrong predictions, the label of y-axis false positive rate is a ratio between number of false positives and the number of wrong predictions. The graphical interpretation is that the line y = x indicates a random classifier, where the left-top corner indicates an ideal classifier. Better classifiers are supposed to be closer to the left-top corner.

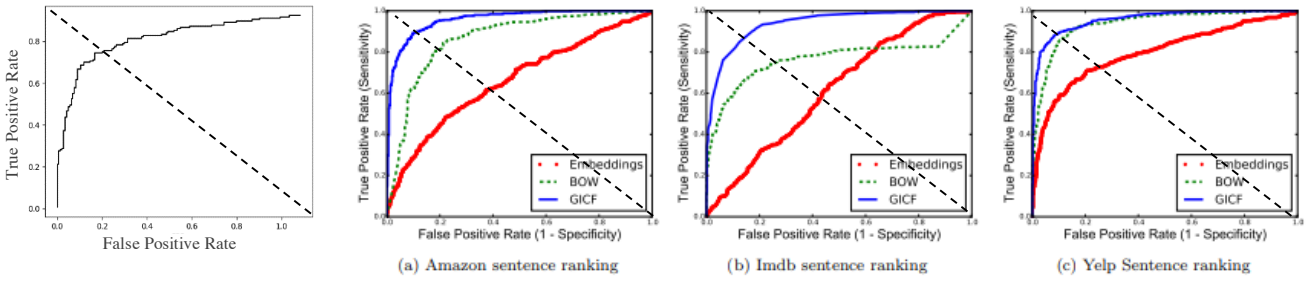

Fig 5. Comparison between ROC of testing results using two methods. (As the number of testing data of reviews for each website is limited in our study, we combine the testing prediction result together to make a smoother curve. The right three is a screenshot taken from the related paper, which shows ROC for each type of reviews)

It can be directly seen from Fig 5 that our curve yields better performance than the average performances with pure word embedding method in CNN while it still cannot outperform GICF which is the new vector representation method proposed

in related paper. As the paper of GICF has no record about comparing the convergence time using different representation method, we cannot compare more comprehensive performance between our and their works. But it is obvious that they only focus on the prediction accuracy while we both reduce the convergence time of training process and improve the accuracy using MDR, which can be seen by comparing the ROC of ours and their embedding one.

Accuracy. We also calculated a numerical evaluation metric accuracy, which is defined as a description of a combination of both random and systematic errors. High accuracy can be achieved only when both precision and recall values are high. Higher accuracy value indicates better performance on classification tasks. Table 1 compares the accuracy value obtained from the CNN using combined vectorization methods in the related paper and our CNN using noise removal method.

Table 1. The results of our method shown below are the best accuracy results computed on testing set over 20 runs.

	Amazon	**IMDb**	**Yelp**
Logistic w/ BOW	79.0%	76.2%	75.1%
Logistic w/ Embeddings	54.3%	57.9%	66.5%
GICF w/Embeddings	88.2%	86.0%	86.3%
MDR w/Embedding	81.0%	82.0%	79.0%

From above metrics, it is obvious that CNN with multimodal removal performs better than plane CNN in terms of ROC and accuracy metrics in a binary text classification task. Although the improvement of prediction performance is not as good as GICF method, which may be affected by the number and noisy level of provided data, the significant reduction of training time also makes the MDR method worth considering. This observation supports the conclusions and discussions we make in section 3.3, and validates the effectiveness of our method.

4 Conclusion and Future Work

In this report, we propose multimodal removal method for noisy data when training neural networks. We test out method over real-world datasets and compare our approach with existing methods. Our approach yields better prediction performance than pure CNN but still cannot outperform the method achieved in [8] for the same classification problem with the same dataset because MDR focus more on the comprehensive performance of both prediction accuracy and efficiency, and the limited number (only 3000) and unknown noisy level of training data may also have negative impacts on the performance. Comparing with neural networks without data removal, our approach achieved higher time efficiency in terms of convergence. However, a trade-off exists between the training efficiency and effectiveness. Above findings prove that our method can be applied to enhance practical problems.

There are plenty of works can be extended from our method. The criteria of detecting noisy data can be refined to a provable level. Our method can also be improved by combining it with the vector representation method proposed in [8] to achieve a better performance for sentiment analysis problems as word vectorization has no overlap with noisy data removal, which means implementing both approaches will achieve double improvements. Furthermore, the terminating criteria is still based on a constant value by convention, which may result in overfitting if too many samples are removed. One can find an approach to adjust this value dynamically according to concrete model or prove the robustness of the criteria.

References

1. Gedeon, T. D.: Stochastic bidirectional training. In: IEEE International Conference on System Man and Cybernetics (SMC 1998), San Diego, pp. 1968–1971 (1998)
2. Han, S., Mao, H., Dally, W.J.: Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149
3. Čížek, P. & Víšek, J.Á.: Least trimmed squares. In Härdle, W., Hlávka, Z., & Klinke, S. (eds.), *XploRe Application Guide*, pp. 49–64.Springer-Verlag (2000)
4. Slade, P., Gedeon, T. D.: Bimodal Distribution Removal, *IWANN*, pp. 249-254 (1993)
5. Johnson R, Zhang T. Supervised and semi-supervised text categorization using LSTM for region embeddings[J]. arXiv preprint arXiv:1602.02373 (2016)
6. Yin W, Kann K, Yu M, et al. Comparative study of cnn and rnn for natural language processing[J]. arXiv preprint arXiv:1702.01923 (2017)
7. UCI Machine learning Repository, https://archive.ics.uci.edu/ml/datasets/Sentiment+Labelled+Sentences
8. Kotzias D, Denil M, De Freitas N, et al. From group to individual labels using deep features[C]//Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM. 597-606 (2015)
9. Mannor, S., Peleg, D. and Rubinstein, R.: The cross entropy method for classification, Proc. ICML (2005)