Movie Review Sentiment Analysis

Pengfei Zhang, Chao Gao, Weiwei Li
Dizzy Hyperparameters

May 2016

1 Goal:

The goal is to develop a supervised method to automatically classify movie reviews into positive or negative categories and pursuing the best accuracy.

2 Introduction:

From the goal, we got 2 hints to solve the problem, first, we need to find a way to present the input in a proper way for classification, more detail, we need to build a vector space for the sequential linguistic symbol which could keep most of the sentiment information. Second, find a good classify for the features we get.

3 Background:

Because it is about movie, we can assume the language we used in the dataset is heavily restricted, so people would have similar expression styles. To predict the result, intuitively, people could use their experience to select some words we knew that has strong emotion such as "good, like, bad, hate", and check if these words exist in the text to classify them to the corresponding category. Some earily work from Pang, Lee and Vaithyanathan showed that, based on the human selected words, under the help of statistics, the accuracy could reach 69%. Based on the similar idea what the count of the words more or less represent the meaning of a text, TF-IDF (term frequency–inverse document frequency) was invented, one thing need to notice is, the unit of the language we focused is not word, but term here, so the machine can classify not only "snow" and "ball", but also "snow ball". However, we still missed another important information embedded in language, which is sequence. That's why n-gram and neural networks become very popular in NLP nowadays (especially LSTM, RNN and CNN).

4 Approach:

Before we start the exciting machine learning works, some necessary preprocessing jobs need to be done. Considering our training set only has 25,000 articles, it is very hard, but surely we can, to learning the advanced language model from the data, then we only focused on the sequence of terms. So the main step of preprocessing include: Lowercase, remove punctuations and only keep the word stem for the training.

Without e a proper understanding of the language data, we decide to start with TF-IDF with a simple classifier, so we selected logistic classifier as our start point, then SVM with linear and nonlinear kernels, and finally CNN and LSTM to try to learn features from the raw data instead of TF-IDF, which is still a hand-craft method. It's also the same routine that we have introduced in the background how the text classification developed. The final result of this report also showed that when we are selecting the model and method, we need to think about the time and spatial scale of the problem we need to solve. Also, more advanced model might need more time and experience to fine tuning the hyper parameters and then finally to get the best result.

1. TF-IDF

The key concept of TF-IDF is evaluating the term frequency in a document and reduce the effect of the word appeared too much in all the documents such as "the", which is also the most frequent word in our dictionary in the train set, also because the IDF we applied, the result showed no significant different with and without stop words, which contain rare information, but could heavily reduce our computing work. And we have tried 1-gram, 2-gram and 3-gram to build the terms.

2. Logistic Regression

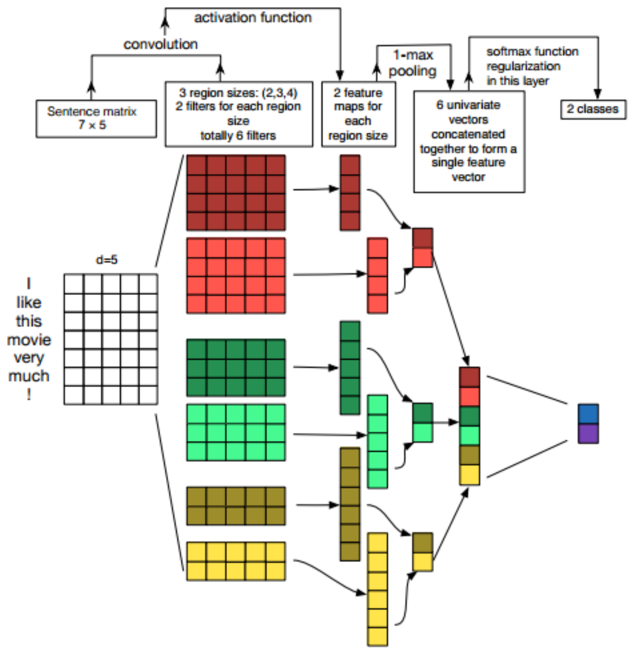
We applied Logistic Regression as our first classifier, trained the classifier with L1 and L2 regularization via the penalty parameter and compared different regularization strengths by defining a range of values for the inverse-regularization parameter C.

3. SVM

After we got a not bad result from Logistic Regression, we assume the distribution of the TF-IDF feature of our data is kind of linear in the space, so we select SVM as our second classifier, trained SVM with linear kernel, polynomial kernel, sigmoid kernel and rbf kernel.

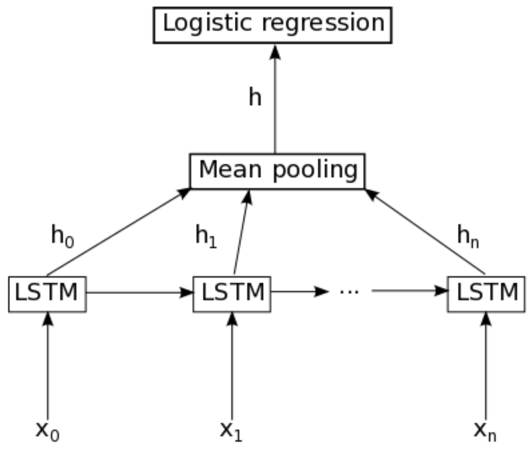
4. Convolutional Neural Networks

With CNN, we have tried the CNN designed from Zhang and Wallace (2015), which first depict three filter, which sizes are 2, 3, 4 respectively, which corresponding to the concept of 1-gram, 2-gram and 3-gram. Then 1 max-pooling layer and input the concatenated feature to the softmax function give us the output. Here, due to the input in the original implementation takes the max length of the article as the input length, however, for our data set, it is too long, so in our implementation, we applied a max-length of 100, 200 and 300 and a ratio to control the position to seperate the review, which defaultly 0.5. So if the review is longer than max-length, we could take the first length * ratio words of the review and the last length * (1 - ratio) words from the end the review to consist a new review, because we think the head and end of the article usually keeps most of the author's opionion.


activation function
convolution
Sentence matrix 7 × 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
1-max pooling
6 univariate vectors concatenated together to form a single feature vector
softmax function regularization in this layer
2 classes
d=5
I
like
this
movie
very
much
!

5. LSTM

The input of LSTM is same as CNN. We applied the tutorial level LSTM from Theano's documentation, which contains a layer of LSTM, a mean pooling layer and then get the final result through logistic regression, because the time limit, we only tried a small dataset from our trainset (2,000 used to train and 500 to test) and the accuracy on the small test set is 0.813.



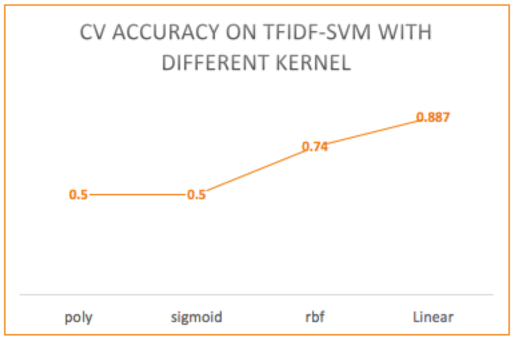
5 Dataset:

SOURCE: IMDB Large Movie Review Dataset	
Training Set	
POS: 12500	**NEG:** 12500
Test Set	
11000	
Vocabulary	
75540	

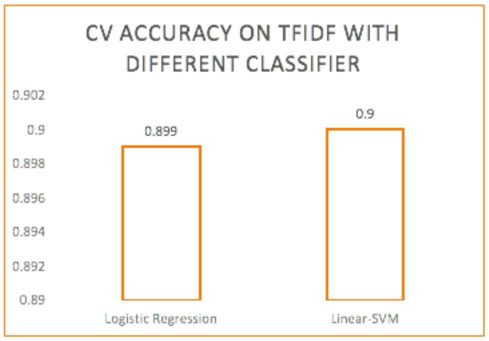
Number of Words		
	Original	**Without stop words**
Mean	238	121
Max	2498	1429
Min	10	4

6 Evaluation:

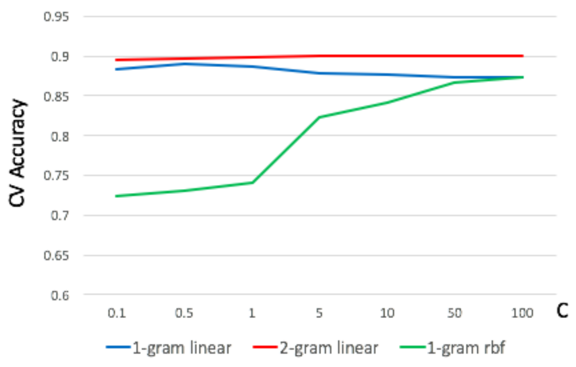
1. TFIDF
a. The CV accuracy on different kernel model in TFIDF-SVM method.



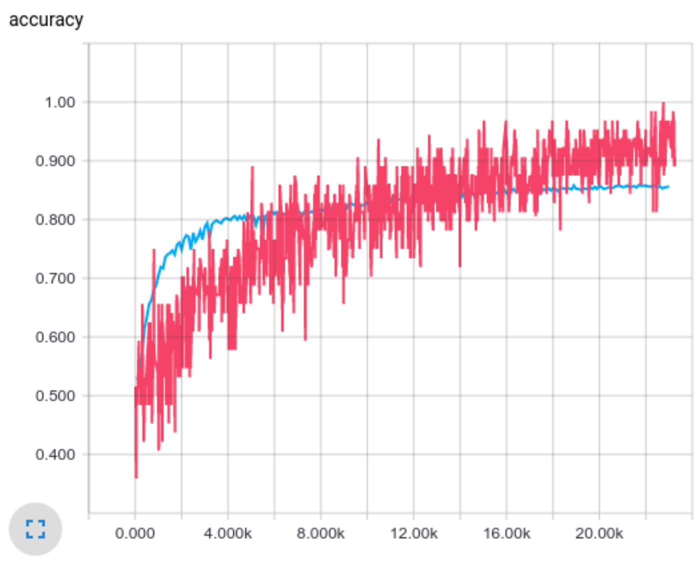
b. The CV accuracy on different classifier in TFIDF method
Based on the previous result, we choose Linear SVM model and compared it with Logistic Regression model in TFIDF. After keep tuning, the best result for different classifier is shown below:

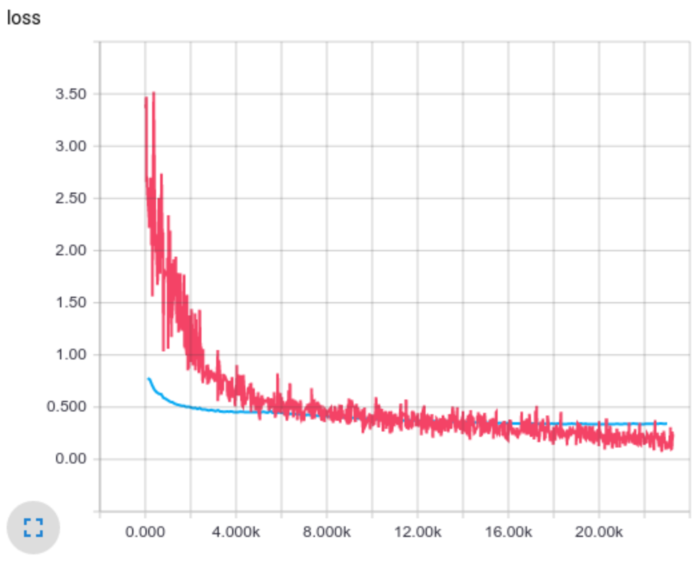


c. The CV accuracy on different parameter C and different gram in TFIDF-SVM method



d. CNN Becasue CNN is too time consuming, we having tuned the parameters well, the best result we got until now is 0.86564 on Kaggle's test set. Here is the figure of accuracy and loss with $embeddingdimension = 128, batchsize = 64$, $epoch = 100$, $max\_length = 200$ and $ratio = 0.5$.





7 Conclusion:

Although the final result through all the models are produced by TFIDF + SVM with Linear Kernel, which is 0.89709 on Kaggle, however, the CNN showed a potential to get better result if we select proper tuning. Furthermore, in the input, we ignored the information contained in the architecture of the review, so with sentence vector and paragraph vector, we could abandon the rough method that only take word from the head and the end, and get the better result.

8 Team Roles:

Pengfei Zhang, feature selection, implementation of classifiers.
Chao Gao, classifier and kernel selection, hyper-parameters tuning, data pre-processing.
Weiwei Li, classifier and kernel selection, maintenance of system.

9 Reference:

Convolutional Neural Networks for Sentence Classification, Y.Kim, 2014

A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification, Y.Zhang, B.Wallance, 2015

Learning Word Vectors for Sentiment Analysis, A.L.Maas, R.E.Daly, P.T.Pham, D.Huang, A.Y.Ng, and C.Potts, 2011

A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, B.Pang, L.Lee, 2004

Learning distributed representations of concepts, G.E.Hinton, 1986

Efficient Estimation of Word Representations in Vector Space, T.Mikolov, K,Chen, G.Corrado, J.Dean, 2013

Distributed representations of words and phrases and their compositionality, T.Mikolov, I.Sutskever, K.Chen, GS.Corrado, J.Dean, 2013

Opinion Mining and Sentiment Analysis, B.Pang, L.Lee, 2008

Thumbs up? Sentiment Classification using Machine Learning Techniques, B.Pang, L.Lee, 2002