Sentiment Classification and Opinion Mining on Airline Reviews

Peng Yuan(pengy@Stanford.edu), Yangxin Zhong(yangxin@Stanford.edu), Jian Huang(jhuang33@Stanford.edu)

Overview

Someone tweets: *@VirginAmerica Hey first time flyer next week - excited! But I'm having a hard time getting my flights added to my Elevate account. Help?*

- Is he/she happy or unhappy with the flight experience?
- If he/she is not happy with the, what went wrong?

Our project let machine solve these two questions for us.

Objective: using machine learning techniques to extract customers' feedback information from text reviews (e.g. tweets), including whether the customer likes or dislikes the services/products (*sentiment classification task*) and concrete opinions of in what aspect the user dislikes such services/products (*negative review reason task*).

Brief results: We tried several approaches including Naïve Bayes, Linear SVM, Lexicon-based Classification, and CNN with word2vec. Currently, our best solution is SVM which achieves 79.6% accuracy for sentiment task and 64.5% for the negative review reason task.

Data and Features

Data: 14640 tweets from 2/17/2015 to 2/24/2015 related to reviews of major U.S. airlines, containing *sentiment label, negative review reasons label, tweets content* and other meta information like *location*, *user ID* etc. The data is split into 80% as the training set and 20% for testing.

N-gram features: for each tweet in the dataset, we take the n-gram of the letters and the words as features and store them in sparse matrix. Eg."*having a hard*" with 3-gram letter feature will yield feature vector [hav, avi, vin, ing, nga, gah, aha, har, ard]

Word2vec[1] features: obtain word vectors by training a distributed representation, and map each word in the tweet into a dense, fixed-length vectors.

Approaches

Lexicon-based Sentiment Classification[2]:
Lexicon-based methods use a word (lexicon) list trained to tell whether each word has a positive or negative sentiment in general and aggregate the word sentiment score into the sentence/paragraph sentiment score.

Multinomial Naive Bayes (Multinomial NB):
We use the n-gram word features to feed the multinomial Nave Bayes model and use a 5-fold cross validation to select hyper-parameters.

Linear Kernel Support Vector Machine (Linear SVM):
Similar setup as the Multinomial Naive Bayes, use the n-gram word features and use a 5-fold cross validation to select hyper-parameters.

Convolutional Neural Networks (CNN):
We use word2vec features to train a convolutional neural network model and then use it to classify tweets. The convolutional layer can capture the relationship between words; the detailed network is in Figure 1.

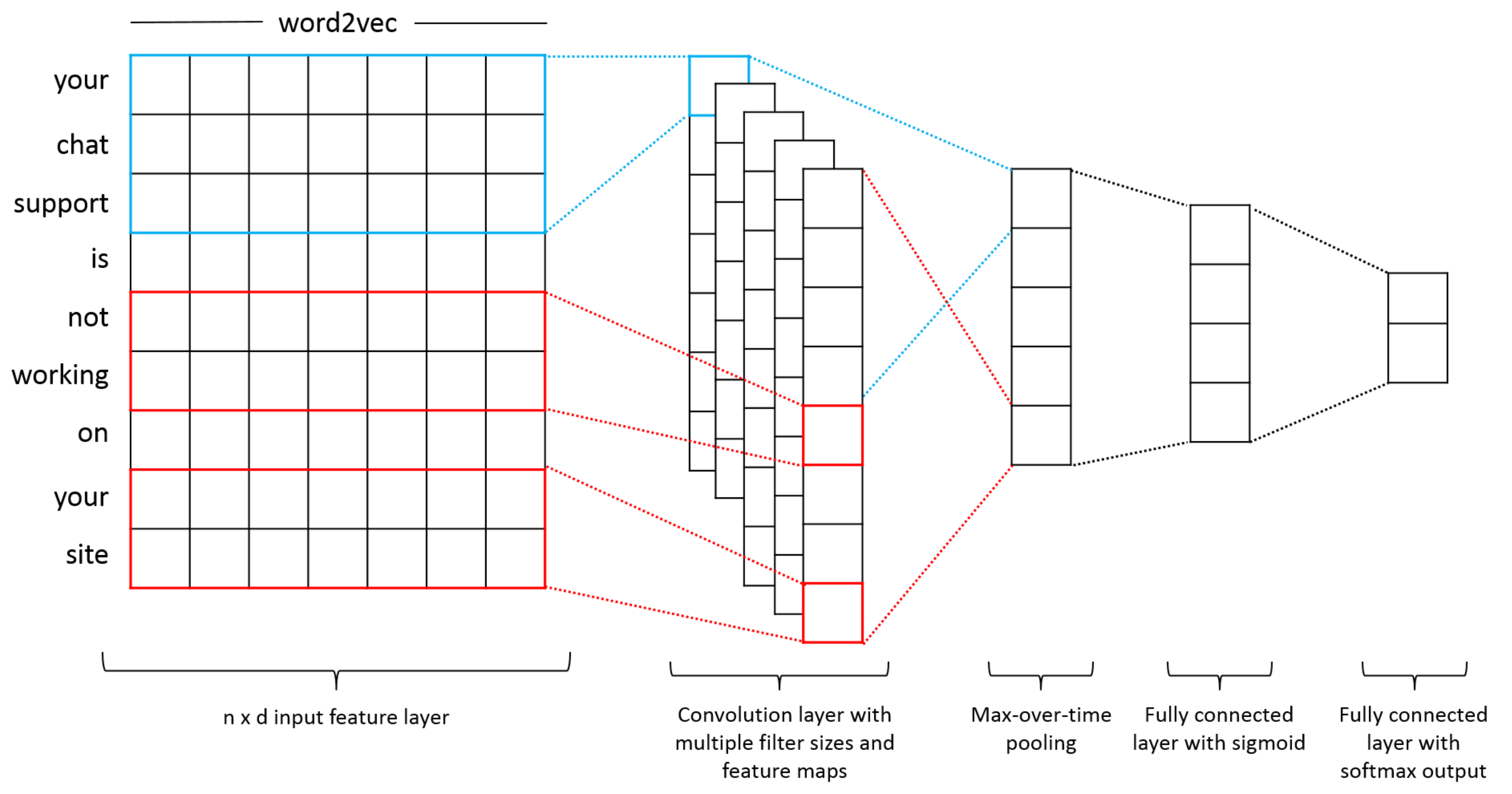
word2vec
your
chat
support
is
not
working
on
your
site
n x d input feature layer
Convolution layer with multiple filter sizes and feature maps
Max-over-time pooling
Fully connected layer with sigmoid
Fully connected layer with softmax output

Figure 1: CNN network details

References

[1]. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.
[2]. M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede. Lexicon-based methods for sentiment analysis. Computational linguistics, 37(2):267–307, 2011.
[3]. Nguyen, T.H. and R. Grishman, Combining Neural Networks and Log-linear Models to Improve Relation Extraction. CoRR, 2015.

Results and Discussion

Table 1: test results for sentiment classification task

Method	Lexicon-based	Multinomial Naive Bayes	Linear SVM	Convolutional Neural Networks
Positive F-1	0.549	0.482	0.739	0.718
Negative F-1	0.769	0.818	0.873	0.861
Neutral F-1	0.294	0.367	0.608	0.520
Overall Accuracy	0.652	0.712	0.796	0.790

Table 2: test results for negative review reason task

Method	Multinomial Naive Bayes	Linear Kernel SVM	Convolutional Neural Networks
Overall Accuracy	0.576	0.648	0.601

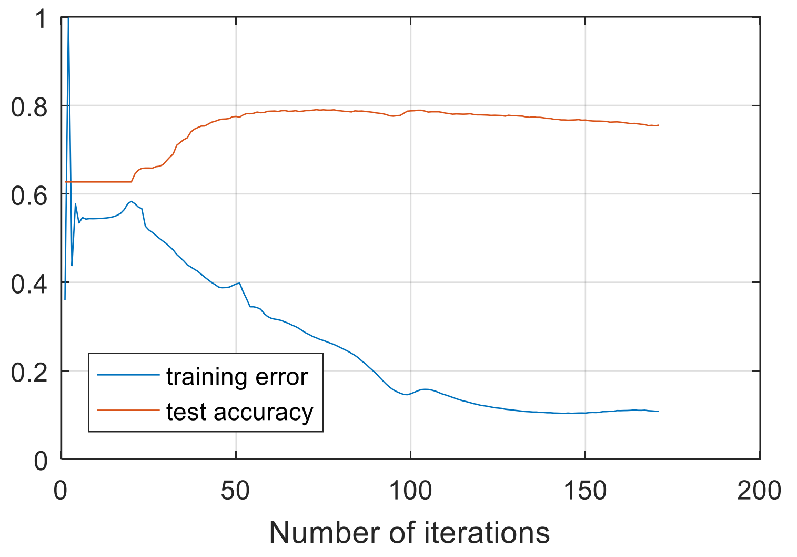

Figure 2: CNN performance while training

Test results for the sentiment classification task is shown in Table 1 and the results for negative review reason task is in Table 2. Figure 2 shows the training error and test accuracy of CNN vs. number of iterations. The flat region for test accuracy in the beginning is due to pooling layer.

Discussion:

- Hyper-parameter can influence the performance a lot, grid search on hyper-parameters using cross validation data helps a lot.
- Lexicon-based methods didn't perform well since it focus on general cases and thus didn't bring any domain specific knowledge.
- Multinomial NB runs faster than SVM due to model simplicity.
- Linear SVM performs well as expected since the data is somewhat separated by support vectors/words like "good" or "bad".
- CNN performs well on sentiment classification task as expected since it employs semantic meaning of words by word2vec, and involves n-gram relationships by the convolution layer.

Future Work:
We consider combine Recurrent neural networks (RNN) and CNN[3] in the future work since RNN can 'remember' all previous information of a tweet and CNN can well capture the inter-word information.