MiniProject2: IMDb Sentiment Analysis

Bo Dang, Andi Dai, and Jingyuan Wang

February 23, 2019

Abstract

In this study, we predicted the classification of sentiment of IMDb movie reviews by using text features and classifiers. The Internet Movie Database, known as IMDb, is one of the most popular movie information online databases and we used 25,000 labeled movie reviews as training and validation set. For this binary classification task, we did the tests on three different text feature extraction pipelines, including sentiment lexicon occurrence, N-gram tf-idf, and word embedding. Then we implemented several different classifiers and used 5-fold validation pipeline to evaluate their performance. Based on our validation results, tf-idf has the best performance among all features, while the classifiers of linear-svm, nbsvm, logistic regression and neural networks performed well in this classification task. Our best test result was got from nbsvm with tf-idf features, which obtained a submitted test accuracy of 91.96%. Overall, our method showed an effective sentiment classification ability based on the language words they contain.

1 Introduction

The purpose of this project is to predict classification of sentiment of IMDb movie reviews using different text features and classifiers. IMDb is a popular website and database of movie information and reviews. This is a classical NLP problem. We used 25,000 labeled movie reviews as training and validation set, and another 25000 unlabeled reviews as test set. The label is either positive or negative. For this binary classification task, we did the tests on three different text features, including N-gram tf-idf, sentiment lexicon occurrence, and word embedding. Then we implemented several different classifier models and used 5-fold validation pipeline to evaluate their performance. Based our validation results, tf-idf feature showed best performance, while the classification models of linear-svm, nbsvm, logistic regression and neural networks performed well in this task. They achieved nearly 90% accuracy on validation test. We submitted their results on the test set and all of them achieved an accuracy of over 89%. Our best test result was gotten from nbsvm with tf-idf features, which obtained a test accuracy of 91.96%. For contrast, the Bernoulli Naive Bayes model has been implemented and reached only 81% accuracy in cross-validation test. We also tried stacking and voting method to synthesize different results and got expected improvement.

2 Related work

The sentiment classification, which identifies the sentiment polarity of short ,informal text, has attracted increasing research interest during the past years.

Generally, traditional sentiment analysis approaches adopt lexicon-based approaches [Pang et al., 2002a, Taboada et al., 2011], which mostly use a dictionary of sentiment words with their associated sentiment polarity. The learning based methods for texts sentiment classification follow [Pang et al., 2002b] 's work, which uses bag-of-word representation, representing each word as a one-hot vector. Under this assumption, many feature learning algorithms are proposed to obtain better classification performance [Pang et al., 2008, Liu et al., 2012, Feldman, 2013].

With the rivival of interest in deep learning [Bengio et al., 2013], incorporating the continuous representation of a word as features has been proving effective in the field of sentiment analysis. [Bespalov et al., 2011] initialize the word embedding by Latent Semantic Analysis and further represent each document as the linear weighted of ngram vectors for sentiment classification. Socher et al. propose Recursive Neural Network (RNN), matrix-vector RNN [Socher et al., 2012] and Recursive Neural Tensor Network(RNTN) [Socher et al., 2013]

3 Dataset and setup

Two sets of 12,500 movie reviews are gathered in the text form from the IMDb Sentiment Classification Dataset, with each labeled as either positive or negative review. Another 25,000 reviews are given as the test data and prediction of our models are evaluated through the competition community Kaggle. In this project, we actually used 3 different tokenizers for different model implementations: Scikit learn package for tf-idf weighting, Keras tokenizer, and nltk tokenizer. In each tokenizer, all words are transformed to lowercase.

4 Proposed Approach

4.1 Text Features

First text feature we used is N-gram tf-idf weighting. Initially, the reviews are loaded as a list and then we use *CountVectorizer* in sklearn to convert the collection of text documents to a matrix of token counts. Here we set parameters *ngram_range = (1, 2)* and *analyzer = 'word'*, which mean the feature should be made of word unigram and bigram. Then the *TfidfVectorizer* is adopted to convert the word n-gram features to a matrix of TF-IDF features. Finally the *Normalizer* with parameter *norm = 'l2'*, is used to normalize the TF-IDF features of individual samples.

Another feature we extract is the sentiment lexicon occurrence. We used positive and negative English opinion words [Hu and Liu, 2004] as feature tokens, counted number of occurrences of each lexicon in each text and took a square root of it for a better performance.

4.2 Classifier Models

1. Bernoulli Naive Bayes (BNB), it counts the contingent probability of each word, and then calculates posterior probability by multiply occurrence or absence probability of each word. The td-idf features are converted to booleans values. And we use Laplace smoothing, i.e. count from 1 instead of 0, to avoid zero probabilities. We didn't choose the specific vocabulary for the model. This is the only one model we implemented from scratch.

2. Logistic regression, is implemented by importing *LogisticRegression* from sklearn. The *penalty* is set to *'l2'* to avoid overfitting. The stopping criteria, *tol* is set to *1e-4* and the maximum number of iteration for convergence, *max_iter* is set to *500*. In order to fully exploit the power of CPU, the parameter *n_jobs* is set to *-1*, which means using all processors.

3. Decision trees, is also imported from sklearn. The max tree depth is important for this model. More tree depth makes it easily fall into over-fitting, but fewer may cause serious under-fitting. Max Depth approximate to 15 is reasonable.

4. Support vector machines (SVM), together tested with two advanced version. When we first use normal SVC function in sklearn, there is minor problem with convergence of this model and the performance is not fairly well. So we try out LinearSVC also from sklearn, which focuses more on linear classification and scales better for large dataset, with *tol* set to *1e-8* and penalty parameter C set to 0.6. Finally NBSVM, an SVM model with Naive Bayes feature is used with external implemented model [Sida and D., 2012] in which we tune the hyper parameters of SVM and set α to 0.8, β to 0.3 and C to 10.

5. Convolutional Neural Network (CNN), is trained with Keras framework using tensorflow backend. This model is integrated and contains feature extraction as well. After tokenizing with Keras API, we pad or truncate every instance to unify their dimensions which is set to 500 and an Embedding layer is used to convert token sequences to word vectors. Then build the model with a 1D convolution layer, 1D MaxPooling layer and a hidden layer which activates parameters with ReLU function. Finally an output layer with Sigmoid activation is added and we use binary cross entropy as loss function and Adam as the optimizer to train our model. We increase embedding layer output and fix overfitting problem by adding dropout rate.

6. Recurrent Neural Network(RNN), uses identical data processing method with CNN model supported by Keras API. However, it has an LSTM layer after Convolution Network model which helps to memorize the sequence of sentiment changes captured by convolution. In this case, number of convolution kernels are reduced to shorten training runtime.

5 Result

5.1 Model Validation Pipeline

We implemented a 5-fold validation pipeline. That is, for the training matrix and the label array, we shuffled and divided them into 5 groups randomly. Then each time we use 4 of 5 groups data as training set and test it on the rest group data. The mean accuracy of 5 times results is the final validation accuracy.

5.2 Validation and Test Result

From Table 1, We got following summarized results:

1. BNB and Decision trees have poor performances on validation. BNB's performance is due to its strong i.d.d. assumption on attributes. For a sentiment classification problem, it is very likely that occurrence of words are related. Also the model would be severely influenced if the training data is incomplete,i.e. test set contains new words that the model did not learn with. And the decision tree is easy to cause over-fitting or under-fitting because the feature dimension is large. It can predict training set itself with 0.97 accuracy, which is uncommon but meaningless.

2. Both logistic regression classifier and SVM have good performances in sentiment classification. Logistic regression is fast and stable but hard to make improvements. All SVM classifiers have

Table 1: Validation and Test Accuracy

Classifier/model	Features	Validation Accuracy	Test Accuracy
BNB	TF-IDF	0.81	N/A
Logistic Regression	TF-IDF	0.8879	0.88253
Logistic Regression[1]	Unigram+Bigram TF-IDF	0.8875	0.88946
Logistic Regression	Sentiment Lexicon	0.8317	0.8440
Decision Trees	TF-IDF	0.72	N/A
Linear SVM	TF-IDF	0.8975	N/A
Linear SVM[2]	Unigram+Bigram TF-IDF	0.9040	0.90440
NB SVM	Unigram+Bigram TF-IDF	0.9129	0.91960
CNN[3]	Word Embedding	0.8960	N/A
RNN	Word Embedding	0.8864	N/A
Stacking Model	With model 1,2,3	N/A	0.90520

Feature TF-IDF refers to general unigram tf-idf weighting

heavy dependency on hyper-parameters but perform rather well from our investigation. The experiment with NB-SVM, which is a model utilizes naive bayes as a regularization for SVM, has further improved the model by 1%. This is the best result we get from the experiments and it may have room for improvement.

3. The two neural network models have reached rather high accuracy but not excellent in this task. Difficulty towards hyper-parameter tuning has been an obstacle for refining the model. Another characteristic to note is that our training set holds only 17,500 data for a held-out validation and the model will performs better if we are given larger dataset.

4. The word tf-idf features have done an unexpectedly good work with a dimension of 75,990 in normal unigram tf-idf features and 1,516,743 with bigram term added. The performance of sentiment lexicon features is fair according to its low feature dimension and achieved 84% on test set accuracy. However, adding it to the tf-idf model has not improved its performance since it lacks recognition for word sequences and negating statement. Furthermore, the usage of bigram features enhanced the model for most classifiers since it captures modified nouns and verbs [Sida and D., 2012].

5. The majority voting approach was helpful but the improvement was rather limited possibly due to identical feature selections of the first and second model noted in Table 1. In summary, the classifier ensemble method contributes to performance but it would be good only if existing models are various and complementary with each other.

6 Discussion and Conclusion

We now have implemented different binary classification models in this task. This is a meaningful NLP practice. We got several models that performed well including Logistic Regression, Linear SVM and Neural Network. The linear SVM with unigram and bigram tf-idf weighting got the best result on the Kaggle test set. For further work, stacking methods on basis of different models may still improve the test performance, and more complicated neural nerworks like LSTM and CNN is possible to perform better. For more general applications, we need more data to validate and make trade-off among models. Another inspiration of finishing this classification is to use random forest or Multi-Grained Cascade forest [Zhihua and Feng, 2017] whose performance is stable over medium size dataset and is outstanding when doing sentiment analysis.

7 Statement of Contribution

Andi Dai implemented BNB approach and tested SciKit learning models on tf-idf features. Bo Dang tested with neural network models and implemented stacking approach of several classifiers. Jingyuan Wang implemented sentiment lexicon features and tested the model with new classifiers. All group member contributed to refining the code and writing the report.

References

[Bengio et al., 2013] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

[Bespalov et al., 2011] Bespalov, D., Bai, B., Qi, Y., and Shokoufandeh, A. (2011). Sentiment classification based on supervised latent n-gram analysis. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 375–382. ACM.

[Feldman, 2013] Feldman, R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89.

[Hu and Liu, 2004] Hu, M. and Liu, B. (2004). Mining and summarizing customer reviews. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD-2004.

[Liu et al., 2012] Liu, K.-L., Li, W.-J., and Guo, M. (2012). Emoticon smoothed language models for twitter sentiment analysis. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

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

[Pang et al., 2002a] Pang, B., Lee, L., and Vaithyanathan, S. (2002a). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

[Pang et al., 2002b] Pang, B., Lee, L., and Vaithyanathan, S. (2002b). Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 79–86, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Sida and D., 2012] Sida, W. and D., M. C. (2012). Baselines and bigrams: Simple, good sentiment and topic classification. In *ACL*.

[Socher et al., 2012] Socher, R., Huval, B., Manning, C. D., and Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for Computational Linguistics.

[Socher et al., 2013] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

[Taboada et al., 2011] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., and Stede, M. (2011). Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

[Zhihua and Feng, 2017] Zhihua, Z. and Feng, J. (2017). Deep forest: Towards an alternative to deep neural networks. In *In the International Joint Conference on Artificial Intelligence (IJCAI)*.