Improving Sentiment Analysis with Data Augmentation

Hang Jiang
hjian42@stanford.edu

Laura Sun
lasun@stanford.edu

Abstract

In this project, we use a novel data augmentation technique to augment the IMDb movie review dataset for sentiment analysis. This data augmentation technique conducts synonym replacement, random insertion, random swaps, and random deletion on the movie reviews while conserving the original labels. Our study shows that this technique is effective in preventing the models from overfitting the training set by introducing some noise. We achieve consistent improvements with classifiers including Logistics Regression and Support Vector Machine.

1 Introduction

Sentiment analysis is a common Natural Language Processing (NLP) task to identify the polarity of a given text. This task has been widely applied to marketing, customer service, and healthcare. However, the data annotation process is usually lengthy and expensive. Hence, creating an effective data augmentation technique prior to training is important. In this project, we are introducing a novel data augmentation technique to sentiment analysis. The input to our algorithm is IMDb movie reviews. We then augment the training data and apply Logistic Regression and SVM to output positive or negative sentiment predictions. As shown on Github[1], we first train classifiers on the original dataset and on our augmented dataset. Our objective is to improve the performance of models on sentiment analysis by using data augmentation.

2 Related Work

2.1 Sentiment Analysis

In the past, researchers have come up with different models, features, and document representations to tackle the sentiment analysis problem. Pang et al.[13][12] found hand-crafted features such as n-gram features are useful to improving the performance of standard machine learning algorithms such as Logistics Regression, Naive Bayes, and SVM on sentiment analysis. However, these features are sparse and discrete, which make it hard to extract contextual information such as sentence relations. Le and Mikolov [6] exploit neural networks to learn dense and continuous word representations from huge corpora. Both word2vec[6] and GloVe[14] are widely used to build document representations for sentiment analysis[8][5][7]. Recently, many contextual word embeddings such as BERT[3] and ULMFiT[4] are used and achieve the state-of-art performance on both IMDb[10] and Stanford Sentiment Treebank (SST)[18] datasets.

2.2 Data Augmentation Techniques

Nevertheless, it is usually hard to apply those models directly because there are not enough annotated data in some specific domains. Therefore, it is necessary to create an effective data augmentation technique, which will allow these models to perform well. Wei and Zou[20] introduced Easy Data Augmentation (EDA) method to augment data for text classification problems and gained consistent improvement on various models. Shleifer[17] and Yu el al.[21] have used back-translation to augment data for text classification and question answering respectively. To our knowledge, we are the first to apply EDA, the simple but effective data augmentation method, to IMDb Movie Reviews dataset.

[1] https://github.com/emoryjianghang/augmentSentimentAnalysis

3 Dataset

We use the Internet Movie Database (IMDb)[11] as our dataset for this project. The IMDb dataset consists of 50,000 movie reviews labeled as positive or negative. The dataset includes 25,000 training data and 25,000 for testing. We pick this dataset because its training set is relatively small compared to its test set. Obtaining more training data with data augmentation is likely to help improve model performance. We further split the original training set into 20,000 for training and 5,000 for validation. We ensure sufficient amount of representation of each class in all sets. We conduct data augmentation only on the training set. Please refer to Table 1 for data distribution for each trial.

augNum	Training Data Size (augmented except for baseline)	Validation Data Size (un-augmented)	Test Data Size (un-augmented)
Baseline	20,000	5,000	25,000
1	40,000	5,000	25,000
2	60,000	5,000	25,000
3	80,000	5,000	25,000
4	100,000	5,000	25,000
5	120,000	5,000	25,000
6	140,000	5,000	25,000
7	160,000	5,000	25,000
8	180,000	5,000	25,000
9	200,000	5,000	25,000

Table 1: Data distribution across training, validation, and test sets. Data augmentation is applied to training sets only for augNum = 1 to 9.

We extract linguistic features and apply pre-trained word embeddings. Table 2 shows the number of these features:

Data	Unigram	Bigram	Trigram	Glove	OPN	SWN	MPQA	Total
20K	70,663	1,343,826	3,004,389	50	2	1	2	4,418,933
180K	81,323	4,728,986	13,336,622	50	2	1	2	18,146,986

Table 2: This table shows the number of features for the original (20K) and augmented training data (180K) respectively with $N_{aug} = 8$. OPN refers to Opinion Lexicon. SWN refers to SentiWordNet.

3.1 Linguistic Features

Traditional linguistic features are important to sentiment analysis. In our task, we explore n-gram features and sentiment lexicons to facilitate the training. First, we extract unigram, bigram, and trigrams from the training set as binary features. Besides, we use Opinion Lexicon[9], Multi-Perspective Question Answering (MPQA)[2], and SentiWordNet[1]. For both Opinion Lexicon and MPQA, we count the number of postive and negative words in each review as features. As for SentiWordNet, we use geometric weighting scheme to generate a sentiment score for each review.

3.2 Word Embeddings

Apart from using n-gram features from the textual data, we also use GloVe word embeddings[15] to represent tokens in our corpus. In particular, we use the pre-trained word vectors with 6B tokens and 400K vocab, which supports 50d, 100d, 200d, and 300d vectors. Unknown words are assigned *unk*'s word vector. To obtain each review's embeddings, we take the average of word embeddings in the review as suggested in the previous works[6]. In this project, we use 50d glove vectors.

4 Methods

4.1 Main Approaches

First of all, we preprocess the review texts, extract linguistic features, and use 50d GloVe word embeddings [2]. We run experiments on Logistics Regression and SVM on the original dataset to obtain baseline results. Ablative analysis is conducted to find the best set of features among n-grams, lexicon features and word embeddings. Secondly, we apply Easy Data Augmentation (EDA) technique to augment the IMDb training set[10]. We conduct nine trials to decide how many reviews we want to augment for each review instance, indicated by N_{aug}. We use Scikit-Learn's Logistics Regression with default settings to select the suitable amount of data augmentation.

[2]https://nlp.stanford.edu/projects/glove/

After the N_{aug} is determined, we train and tune Logistics Regression and SVM to compare the performance with the baseline. We hope to see consistent improvements in performance on the augmented datasets with both classifiers.

4.2 Models

4.2.1 Logistics Regression

We first run Logistic Regression model to obtain a baseline on prediction accuracy. We aim to minimize the cost function in regularized logistic regression given as:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y_i log(h_\theta(x_i)) + (1-y_i)log(1-h_\theta(x_i))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

where $h_\theta(x_i) = \frac{1}{1+e^{-\theta^T x_i}}$. To develop this model, we use Python scikit-learn's LogisticRegression where we tune the inverse of regularization strength C = 0.01, 0.05, 0.25, 0.5, 1. Scikit-Learn's implementation of Logistics Regression uses Stochastic Average Gradient and automatically updates the learning rate, so we do not need to tune the learning rate.

4.2.2 Support Vector Machine

A Support Vector Machine (SVM) is a large margin classifier characterized by a hyperplane separating two classes. The optimization objective is given as

$$\min\ [C\sum_{i=1}^{m}[y_i \text{cost}_1(\theta^T x_i) + (1-y_i)\text{cost}_0(1-\theta^T x_i)] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2]$$

where m is the number of examples, n is the number of features and C is the penalty parameter. We use kernels to transform input data to higher dimensional feature space so that the model can learn more complex boundaries. In our model, we use a linear SVM because LinearSVC scales better to larger datasets and requires less memory to store the data. We used Python scikit-learn's LinearSVC using a linear kernel and tune the penalty parameter C = 0.001, 0.005, 0.01, 0.05, 0.1.

4.3 Original Work

4.3.1 Data Augmentation

We adopted the Easy Data Augmentation (EDA) approach originally proposed by Wei and Zou[20]. WordNet[3] is used to find synonyms of words for the technique. In this paper, the author introduces four operations:

- **Synonym Replacement**: Randomly replace n words with their synonyms in a sentence.
- **Random Insertion**: Insert synonyms of words to a sentence at different positions.
- **Random Swap**: Randomly choose two words in a sentence and swap them.
- **Random Deletion**: Randomly remove each word in a sentence with an assigned probability.

Since some movie reviews are longer, we perform these four operations on n words per movie review with $n = \alpha l$, where α indicates the percentage of words to be changed in a review and l indicates the length of a review. Overall, we generated N_{aug} reviews for each review in the training data set. According to the original paper of EDA, the generated data have similar distribution as the original data and it is valid to conserve the labels.

5 Experiments

5.1 Evaluation Method

We use accuracy to evaluate the performance of our models: $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$.

5.2 Baseline

The first step of our project is to build a robust baseline model before doing data augmentation. We use Logistics Regression and SVM on the original IMDb dataset. Also, we want to find the best combination of features to feed into the classifiers. For each set of features, we tuned the

[3] http://www.nltk.org/howto/wordnet.html

hyper-parameters of the two models and select the best-performing model on validation set for predictions on the test set. The best-performing Logistics Regression achieves **89.59%** and the best SVM achieves **89.77%** on the test set. We will discuss details about how we choose the features in the following section.

5.3 Ablative Analysis

We have conducted ablative analysis to explain the difference between the baseline performance and current performance. As indicated in Table 3(a) and 3(b), ablative analysis shows that n-grams features are more important than others in terms of helping decreasing the misclassification rate for both Logistics Regression and SVM models. In contrast, removing lexicon features increases accuracy. This could indicate that the additional lexicon features potentially overfit the model on training set.

Component	Accuracy
LR+GloVe+ngrams+lexicon	89.12%
LR+GloVe+ngrams	89.59%
LR+ngrams	89.57%
LR+GloVe	75.02%

Component	Accuracy
SVM+GloVe+ngrams+lexicon	89.17%
SVM+GloVe+ngrams	89.77%
SVM+ngrams	89.64%
SVM+GloVe	75.12%

Table 3: (a): Accuracy when removing feature from logistic regression. (b): Accuracy when removing features from SVM.

5.4 How Much Augmentation?

In our experiment setting, we used the recommended value for $\alpha = 0.1$ since high α tends to hurt performance and changes the identity of a review, whereas low α is simply repeating the same review without many edits. Then, we augmented 9 reviews for each of the review instance in the training data set. We use Logistics Regression with $C = 1.0$ and "GloVe+n-grams" features to decide the optimal N_{aug} for our IMDb sentiment analysis task. For all trials, we achieved 100% accuracy on training data with a maximum of 1000 iterations. We use $N_{aug} = 8$ for our experiment. As shown in Figure 1, the test set accuracy is the highest at $N_{aug} = 8$, tied with $N_{aug} = 9$. Figure 2 shows that the validation loss is the lowest at $N_{aug} = 8$. This is reasonable because low N_{aug} does not introduce enough diversity and high N_{aug} generates too much noise to the data. Overall, accuracy improved from 89.54% to 90.02%.

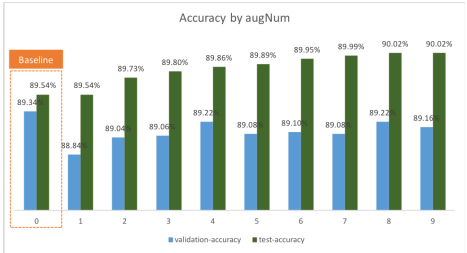

Figure 1: Accuracy on validation and test sets using Logistics Regression, with $C = 1.0$.

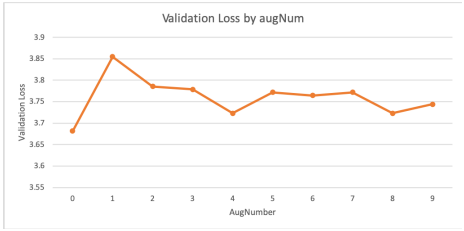

Figure 2: Validation loss versus augmentation number.

5.5 Fine Tuning

After determining the optimal $N_{aug} = 8$, we trained and tuned hyperparameters for both Logistics Regression and SVM. We achieved **90.12%** (+0.53% improvement to the baseline) with Logistics Regression and **90.21%** (+0.44% improvement to the baseline) with SVM.

5.6 Error Analysis

Our model trained on the original data set has significantly more features than the training samples. The number of features in the baseline is 4.4 million whereas the number of training samples is 20,000. As shown in the learning curves in Figure 3(a), there was a large gap between training score and validation score. The baseline model his able to achieve 100% accuracy on training set and 90% on validation set. This could potentially indicate that the model trained on the original data set suffers from high variance and low bias. Hence, we decide to perform data augmentation and increase the number of training samples to 180,000. The addition of augmented data generated helps address the overfitting problem.

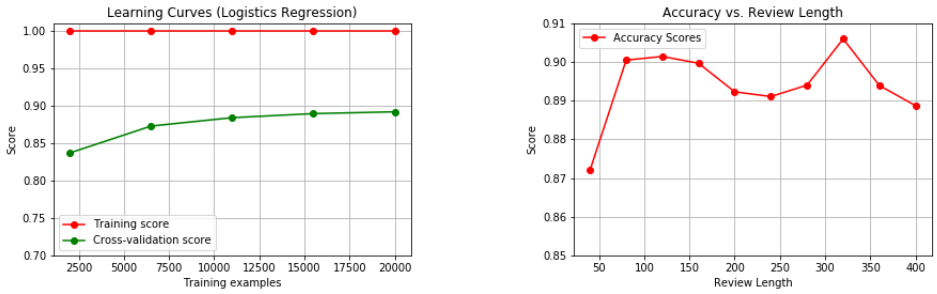

Figure 3: (a): Learning curves for the baseline model. (b): Baseline accuracy vs. review length.

We examined the predictions on test set by our model trained on both original data and on augmented data sets, and found that it suffered from the following types of errors: out of vocabulary (OOV) words, complicated sentence structure, and variable review length.

OOV Error Type: When a test instance contains OOV words for a logistics regression model, this model will not be able to recognize these words as useful features for classification. One advantage of using augmented data is to introduce new vocabulary into the training set so that our model can see more words. Many misclassified instances by the baseline model contain a few new words, which hurts the performance of the baseline model. Using data augmentation allows the model to generalize to new words which were not present in the original data set.

Complicated Structure Error Type: There are still some instances that are still incorrectly classified after we use the augmented data because of their complicated sentence structures. For instance, there is one review saying "i must admit when i first began watching this film i had no clue what was going on so the beginning was a bit confusing for me however that did not diminish my enjoyment of the movie...". This sentence contains many negative words, but the transition word "however" changes the polarity of the sentence in the end. To overcome this type of errors, we can use advanced models such as attention-based models to consider contextual information when making the predictions.

Sentence Length Error Type: The IMDb data set contains various lengths of reviews from a few words to hundreds of words per review. As shown in Figure 3(b), the accuracy decreases for reviews with less than 70 words. Our hypothesis is that there are significant amount of sparse and discrete features using n-grams on short reviews (with few words). One potential improvement is to use word embeddings only and generate dense and continuous features for shorter reviews.

6 Conclusion

In this project, we have demonstrated that using data augmentation techniques can improve the accuracy of model predictions when there is limited amount of training data. By tuning the augmentation number, we found the optimal $N_{aug} = 8$ and achieved **0.53%** improvement on Logistics Regression and **0.44%** on SVM. Our Logistic Regression model generalizes better with the augmented training data than SVM does. Data augmentation introduces noise to the training data, and helps address the overfitting problem in the baseline model. Besides, we also conduct an ablative analysis to find the best feature combination: n-gram features and GloVe word embeddings. Lexicon features are less useful in this case due to potential overlapping information as in word embeddings and n-grams.

In the future, we can experiment with more advanced models such as Attention-LSTM[19], BERT[3], and ELMo[16] on the augmented data to see whether we can see a consistent improvement with these models. Furthermore, we can combine EDA technique and backtranslation to augment the data and see whether we can achieve more improvement.

7 Appendix

References

[1] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In *Lrec*, volume 10, pages 2200–2204, 2010.

[2] Lingjia Deng and Janyce Wiebe. Mpqa 3.0: An entity/event-level sentiment corpus. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1323–1328, 2015.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[5] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[6] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

[7] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[8] Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

[9] Bing Liu, Minqing Hu, and Junsheng Cheng. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of the 14th international conference on World Wide Web*, pages 342–351. ACM, 2005.

[10] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics, 2011.

[11] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

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

[13] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 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, 2002.

[14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[15] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[16] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[17] Sam Shleifer. Low resource text classification with ulmfit and backtranslation. *arXiv preprint arXiv:1903.09244*, 2019.

[18] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 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, 2013.

[19] Yequan Wang, Minlie Huang, Li Zhao, et al. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, 2016.

[20] Jason W Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

[21] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*, 2018.

8 Acknowledgement

Every team member contributed equally in this project. We also want to acknowledge Younes Bensouda Mourri for his generous advice.