Evaluating Lexical and Translation-Based Data Augmentation for Low-Resource Sentiment Analysis

Michael Gao, Eric Lee, Andrew Park, David Zhan

Abstract

Sentiment Analysis has become a huge subdomain in the field of natural language processing. It involves parsing text to determine or quantify the sentiment or emotions of the writer. This field is relevant to a variety of fields, including but not limited to product reviews, stock analysis, and general consumer sentiment. Additionally, data augmentation has earned its spot as a vital strategy for for enhancing the performance of sentiment analysis models. This paper explores various augmentation techniques, including synonym replacement, back-translation, and random insertion, to improve model generalization and robustness. Our experiments demonstrate how these techniques influence model performance. The findings provide a comparative analysis of the effectiveness of each method, revealing that data augmentation can significantly boost sentiment classification accuracy. These insights offer a practical framework for practitioners seeking to optimize sentiment analysis models in potentially low resource environments.

1 Introduction

1.1 The task at hand

The task that we are undertaking involves training a baseline LSTM model on RateMyProfessor.com reviews and comparing the performance of the model both before and after the introduction of data augmentation techniques such as synonym replacement, random deletion, and back translation.

1.2 Illustrative Example

To illustrate, consider a review such as "The professor was very engaging and made the lectures enjoyable." Applying synonym replacement, we might alter "engaging" to "captivating," resulting in "The professor was very captivating and made the lectures enjoyable." This example highlights how small augmentations can diversify the dataset and expose the model to a broader set of linguistic patterns.

1.3 Formal Definition of the Problem

Given a dataset D of text reviews from RateMyProfessor.com X with associated sentiment labels Y, where Y takes a between 1 and 5 (inclusive, with 0.5 point intervals), our goal is to train a model $f(X;\theta)$ that predicts Y from X. We define data augmentation as a process $A(X)$, where the funciton A represents synonym replacement, random deletion, or back traslation, that transforms X into a new set X' to increase data variability. The objective is to evaluate the performance of the LSTM model with and without the application of these transformations.

1.4 Why We Selected This Task

This task was selected because sentiment analysis is a fundamental problem in Natural Language Processing with applications in customer feedback, social media monitoring, and product reviews. By focusing on data augmentation, we aim to address the challenge of data scarcity, an increasingly common problem in machine learning. Enhancing the robustness and generalization of sentiment models has practical relevance in both academia and industry. Furthermore, RateMyProfessor.com provides a rich and diverse source of text data that is representative of real-world sentiment analysis applications.

2 Literature Review

2.1 Shared Task

The shared task relevant to this project involves improving the performance of sentiment analysis models through data augmentation techniques. This challenge is common in NLP competitions and research, where participants aim to create robust models that generalize well to unseen data. The

introduction of synonym replacement, random deletion, and back-translation as augmentation methods aligns with established best practices for improving model generalization.

One example of this shared task is discussed by **(author?)** (1), who introduced a data augmentation strategy for BERT in open-domain question answering. Their approach demonstrated that augmenting data with both positive and negative examples significantly enhanced model performance. This principle can be extended to sentiment analysis, where varied examples help models learn diverse linguistic patterns, thereby improving robustness.

2.2 Summary of Related Research

We have reviewed and analyzed several papers on data augmentation in NLP. Following this, we summarize our findings.

Yang et al. (2019) presented a novel approach to data augmentation for BERT fine-tuning in open-domain question answering. They utilized a stage-wise training process where data from dissimilar sources was used initially, followed by more task-relevant data. This strategy improved generalization and demonstrated the value of diverse data. While their focus was on question answering, the general concept of multi-stage fine-tuning and exposure to diverse examples can be applied to sentiment analysis.

Lexical Substitution for Sentiment Analysis Another related method is Part-of-Speech Focused Lexical Substitution (PLSDA), which selectively replaces adjectives, nouns, and verbs in sentiment-labeled texts to generate augmented samples. By maintaining syntactic correctness and semantic consistency, this technique ensures high-quality augmentations that enhance model robustness. Compared to simpler synonym replacement, PLSDA applies linguistic constraints to ensure relevance and quality.

Comprehensive Survey on Data Augmentation conducted a comprehensive survey of data augmentation techniques in NLP, categorizing them into paraphrasing, noising, and sampling. For sentiment analysis, paraphrasing techniques like synonym replacement and back-translation were found to be especially effective. They also highlighted the importance of balancing augmented data to avoid overfitting, a key consideration in our approach.

These studies collectively demonstrate that data augmentation, when applied thoughtfully, can significantly improve sentiment analysis models' performance. By leveraging concepts like stage-wise training, lexical substitution, and balanced data sampling, we aim to build a sentiment model that generalizes well across unseen data.

3 Experimental Design

3.1 Data:

We used a dataset from HuggingFace composed of comments from the popular feedback website RateMyProfessor.com, consisting of 336,239 rows of training data, 72,052 rows of development data, and 72,051 rows of testing data.

We take a random subset of 50,000 rows of the training data, since the original 336k is computationally expensive for data augmentation: even augmenting 20% of our train sample (10k rows) takes approximately 4 hours for synonym replacement, and 10% of our train sample (5k rows) takes 8 hours for back translation. These procedures cannot be optimized heavily with GPU via Google Colab, and the aforementioned compute times include the usage of multiprocessing.

3.2 Evaluation Metric:

We evaluated the baselines and augmented models based on 4 criteria: Mean Squared Error, Mean Absolute Error, R-squared score, and Quadratic Weighted Kappa Score.

- **Mean Squared Error (MSE)**is the average squared error between the true labels and the predicted labels. It is calculated as MSE $= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, where y_i are true labels and $\hat{y}_i$ are predicted labels. A larger MSE value implies larger average error over our predictions.

  This metric was chosen as the model loss function. The more ideal QWK (discussed later in this section) is not differentiable and thus cannot be used as a loss function, so MSE was the best alternative.

- **Mean Absolute Error (MAE)**: is the average absolute distance difference between true labels and predicted labels. It is calculated as MAE $= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$, where y_i are true labels and $\hat{y}_i$ are predicted labels. A larger MAE value implies larger average error over our predictions.

- **R-squared Score (R^2)**: is the is the proportion of the variation in the dependent variable that is predictable from the independent variable. R-squared score is calculated as $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$, where y_i are true labels, $\hat{y}_i$ are predicted labels, and $\bar{y}$ is the mean of the true labels. R^2 ranges from $-\infty$ to 1, with 1 denoting a perfect fit with the data.

- **Quadratic Weighted Kappa (QWK)**: QWK measures the agreement between two raters (in our case, the model and ground truth), while also accounting for the magnitude of disagreement. To calculate QWK, we must create a confusion matrix O that counts the number of occurrences for each class pair. Then, we must create a weight matrix W that penalizes disagreements based on the squared difference. Finally, we must create the expected matrix E which calculates the expected agreement. Formally, it is calculated as $\kappa = 1 - \frac{\sum_{i,j} w_{ij} o_{ij}}{\sum_{i,j} w_{ij} e_{ij}}$, where w_{ij} is the weight between categories i and j, o_{ij} is the observed agreement, and e_{ij} is the expected agreement. κ values lie on the interval $[-1, 1]$, where values close to -1 represent disagreement (worse predictions than random), values close to 0 represent random levels of agreement, and values close to 1 represent perfect agreement (ground truth).

  As discussed earlier, this metric is not differentiable: the summation dependency uses discrete values, so a small change in one prediction can cause a jump discontinuity in QWK.

3.3 Simple Baseline

For our simple baseline, we implemented a majority class baseline. This baseline takes the most common class in the training set (in this case, it was a rating of 5.0), and uses it as a prediction for all inputs of the validation and test set.

With this baseline, we achieved the following results.

	Train Metrics	Val Metrics	Test Metrics
Mean Absolute Error	1.233	1.227	1.229
Mean Squared Error	3.988	3.967	3.966
R2 Score	-0.615	-0.611	-0.615
Quadratic Weighted Kappa	0.000	0.000	0.000

3.4 Strong Baseline

We decided to use a Long Short Term Model (LSTM) regressor for our strong baseline model. An LSTM model is a type of Recurrent Neural Network that use memory cells and gates (forget, input, output) to control the flow of information and memory of the model. While LSTMs are primarily used for classification tasks, LSTM regressors are used to predict continuous values, which we then round to the nearest 0.5 step for analysis.

With an LSTM regressor, we have the following results.

	Train Metrics	Val Metrics	Test Metrics
Mean Absolute Error	0.718	0.859	0.860
Mean Squared Error	0.973	1.400	1.402
R2 Score	0.606	0.432	0.429
Quadratic Weighted Kappa	0.748	0.633	0.631

4 Experimental Results

This section should contain:

- **Published Baseline:** For the published baseline, we implemented an initial version of the data augmentation portion of our project. The technique we decided to implement was Synonym replacement. This data augmentation technique involves taking an input text and replacing a number of the words in the text with synonyms, thus generating a new training row. This allows us to artificially generate more data that retains a similar meaning to the original text, meaning the model will be trained over a more diverse training set, and thus would help the model output more accurate predictions.

  Using NLTK's .synsets() function, we generated synonyms for 3 random words within each sentence. Unfortunately, .synsets() had unpredictable behaviour regarding synonyms for shorter words (for example "Iodine" as a synonym for "I"), so we decided to set a threshold to only use words of length 3 or more as candidates for synonym replacement.

  Additionally, we undersampled our data to 20% due to the fact that the LSTM was taking extraordinarily long to run. Thus, since we only augmented 20% of the data, our dataset was 1.2 times the size of our original.

  We achieved the following results:

	Train Metrics	Val Metrics	Test Metrics
Mean Absolute Error	0.715	0.880	0.884
Mean Squared Error	0.928	1.431	1.445
R2 Score	0.624	0.419	0.412
Quadratic Weighted Kappa	0.762	0.628	0.624

The results after data augmentation only marginally improved compared to the strong baseline. This could be due to the fact that .synsets() may not produce the best synonyms or that we potentially did not augment enough of the dataset to see a meaningful difference.

- **Extensions:** We implemented more techniques as extensions to see if they improved our model performance.

  Random Deletion As a first extension, we implemented random deletion. This is the process of randomly deleting random words at random from a sentence so the model can generalize on shorter sentences with less context. We used a random threshold of 0.3 when considering each word in the sentence, meaning each word had a 30% chance of being deleted.

  Using random deletion, we augmented 20% of our dataset. So in total, our dataset was 1.4 times the size of the original (20% random deletion, 20% synonym replacement)

  We achieved the following results:

	Train Metrics	Val Metrics	Test Metrics
Mean Absolute Error	0.666	0.839	0.837
Mean Squared Error	0.924	1.448	1.451
R2 Score	0.626	0.412	0.409
Quadratic Weighted Kappa	0.771	0.636	0.636

We see that error has generally decreased and we have marginally better QWK scores compared to the strong baseline. But no change is significant.

Back-Translation As a second extension, we implemented back-translation. This is the process of translating a sentence to another language, and then translating it back to English. Through the translation, certain words will be replaced due to differences in language structure and semantics, which then we will be left with a new piece of augmented data. Using teh OPUS-MT models from the Helsinki-NLP group, we translated our data from English to French and vice versa. Unfortunately, back-translation was also taking extraordinarily long compared (7 seconds per row of data). Thus, we decided to only augment 5% of the data with this approach, leaving us with a dataset about 1.45 times the original size (5% Back-translation, 20% random deletion, 20% synonym replacement).

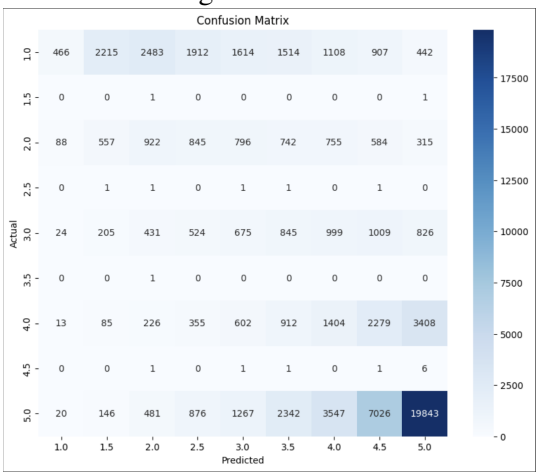
We achieved the following results:

	Train Metrics	Val Metrics	Test Metrics
Mean Absolute Error	0.708	0.869	0.866
Mean Squared Error	0.944	1.438	1.434
R2 Score	0.618	0.416	0.416
Quadratic Weighted Kappa	0.753	0.620	0.622

From the data, it seems that there is very negligible variation from the previous extension. In fact, the results across all categories are marginally worse than the original strong baseline LSTM model. Again, this may be attributed to the translation model not performing as intended, or the fact that we simply were not able to augment a significant enough portion of the dataset to make a non-negligible impact on the test metrics.

5 Error Analysis

With our data predictions, we achieve the following confusion matrix.



The confusion matrix reveals several key insights into the performance of our model. Firstly, there is strong diagonal dominance for the majority class (5.0), indicating that the model predicts this rating accurately when it is the true label. However, significant misclassifications are observed, particularly between neighboring classes. For instance, actual ratings of 4.0 are frequently misclassified as 4.5 or 5.0, and actual 3.0 ratings are often predicted as 3.5 or 4.0. This trend suggests that the model struggles with

fine-grained distinctions between adjacent classes. Additionally, there is a noticeable bias toward predicting 5.0 across multiple actual ratings, which points to an imbalance in the dataset favoring higher ratings. Sparse predictions for rare classes such as 1.5 and 4.5 further highlight the model's difficulty in handling low-frequency labels, likely due to insufficient training data for these classes. To address these issues in the future, we would probably apply techniques such as class balancing, oversampling for minority classes, and incorporating class-weighted loss functions. Moreover, using more robust data augmentation methods and exploring advanced models such as transformer-based architectures may help mitigate these misclassifications and improve overall performance.

6 Conclusions

In this project, we explored the impact of various data augmentation techniques on improving the performance of sentiment analysis models trained on RateMyProfessor.com reviews. Using a baseline LSTM regressor model, we investigated how synonym replacement, random deletion, and back-translation influenced the model's generalization and robustness.

The results showed that while data augmentation techniques provided incremental improvements over the simple baseline, the gains were not as significant as expected. Synonym replacement yielded only marginal improvements, likely due to the limitations of the synonym generation algorithm and the relatively small proportion of data augmented. Random deletion further enhanced the performance slightly by exposing the model to less context, promoting generalization. Back-translation, while theoretically the most effective, faced practical challenges due to computational inefficiency and low coverage of the dataset.

The final augmented dataset (1.45x the original size) showed minor improvements in Mean Absolute Error and Quadratic Weighted Kappa scores compared to the baseline. However, these improvements were not substantial enough to outperform the original strong baseline by a large margin. This suggests that either the amount of augmented data was insufficient or the augmentation methods used did not fully align with the data's characteristics.

Despite these limitations, our findings highlight the potential of data augmentation in low-resource NLP tasks and its ability to enhance model robustness. Future work could focus on optimizing augmentation methods, leveraging more sophisticated synonym generation techniques, and applying augmentation on a larger proportion of the data. Additionally, experimenting with transformer-based architectures like BERT or fine-tuning pre-trained models may yield better results for sentiment analysis tasks. Overall, this project provides a practical framework for integrating data augmentation into NLP pipelines and underscores its importance in improving model performance in real-world applications.

Thanks for reading!

Acknowledgements

We would like to thank Upasana Dutta for all her help and guidance during the project process.

7 Bibilography

References

[1] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Jimmy Lin, and Xiangyang Sun (2019). *Data Augmentation for BERT Fine-Tuning in Open-Domain Question Answering*. arXiv preprint arXiv:1904.06652. Available at: https://arxiv.org/abs/1904.06652.

[2] Rong Xiang, Emmanuele Chersoni, Qin Lu, Chu-Ren Huang, Wenjie Li, and Yunfei Long (2021). *Part-of-Speech Focused Lexical Substitution for Data Augmentation in Sentiment Analysis*. Journal of the Association for Information Science and Technology (JASIST). Available at: https://asistdl.onlinelibrary.wiley.com/doi/full/10.1002/asi.24493.

[3] Bohan Li and others (2022). *Data Augmentation Approaches in Natural Language Processing: A Survey*. AI Open, Vol. 3, pp. 27–41. Available at: https://doi.org/10.1016/j.aiopen.2022.03.001.

A Appendices