Sentiment Analysis to Model Polarity of Amazon Reviews

Anup Bottu
Yale University
anup.bottu@yale.edu

Evan Gerritz
Yale University
evan.gerritz@yale.edu

Siddharth Jain
Yale University
siddharth.jain@yale.edu

Abstract

Sentiment analysis is a classic problem in the field of computational linguistics. While significant progress has been made in sentiment analysis, we have attempted to replicate various models taught in LING 227 (Yale University) to test how model accuracies differ from one another. We began with video game review data on Amazon. Our goal was to predict the given rating using the review's attached text. Our logistic regression model and Naive Bayes models set our baseline accuracy. We then developed a 2 layer neural network, which slightly improved our model. Finally, we used a gated recurrent network and transformer, which significantly improved our accuracy.

After developing these models for the video game data, we look to see how we can continue to improve the model. We look to change the size of our training data, the models' hyperparameters and more. We also look to see how well our model generalizes by testing our model on Amazon reviews for different categories of objects. Our results suggest that as the complexity of the model architecture increases, the ability of the model to accurately extract sentiment information increases on data similar to the training data. Additionally, these more complex architectures generalize better to new types of data, suggesting that these models are capturing something about the English language itself, and are not simply exploiting sentiment statistics specific to the training data.

1 Introduction

Sentiment analysis is the process of extracting the subjective attitude of an author from text data they have written. It's commonly used to determine whether the tone in the text is positive, negative, or neutral. Sentiment analysis can be used for a variety of tasks including customer reviews, social media, and surveys. Firms may use sentiment analysis to understand where products can be improved or to improve marketing techniques. Sentiment analysis is not a trivial problem, however, as there many nuances in correctly parsing natural language. For example, one might look for the presence of words such as "love" that are associated with positive sentiment, but this will cause false positives when the text is "I do not love," and there are even thornier cases like "It is hard for me to say that I love it." The architecture one chooses to model sentiment will greatly influence its ability to capture linguistic structure, and therefore also its performance. In our project, we considered several common approaches to modeling sentiment, comparing both their theoretical expressiveness, as well as their practical results.

2 Methodology

2.1 Data

First, we found a large corpus of information containing possible sentiment information. For this we used large Amazon review datasets, separated by product category. Amazon reviews are a useful corpus to use because they come labeled with the actual sentiment of the author towards the product via a 1-5 star rating. This allows us to use supervised learning techniques. We decided to train the model on one of the smaller datasets containing reviews of video games. To use the data, we had to first remove duplicate reviews, remove reviews missing text, and then take a subset of 16,000 rows in order to make the training of the models more tractable. As can be seen in Figure 1, the video games dataset, in particular, was very left-skewed. This can become a challenge for our models, as the 1-3 star ratings would be hard to distinguish due to their sparsity. Another consequence of these ratings being so scarce in the dataset is that the model can remain highly accurate, while being unable to classify the 1-3 star ratings much better than guessing. To address the first concern, we consid-

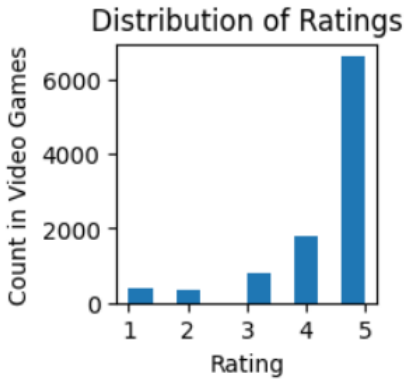

Figure 1: Distribution of Ratings in the video games dataset. Notice that it is heavily left-skewed, which will impact how we train the model.

ered the relaxed problem of binary classification to determine positive and negative sentiment; we combined the 1-3 star ratings into one class expressing generally negative sentiment, and the 4-5 star ratings into a class expressing positive sentiment. By simplifying the problem, the number of training data for each class has increased, increasing our confidence in the trained model's prediction. Additionally, the distinction between positive and negative sentiment seems more well-defined than Amazon ratings which are more subjective, as their usages likely vary between users. To solve the second issue, we chose to focus our evaluation of models based on precision and recall for each class, or rather their harmonic mean through the F-score.

2.2 Models

2.2.1 Logistic Regression

The first model we used was logistic regression, which modeled the probability of positive and negative sentiment given each review. The model first maps an input string to user-defined features and then learns weights for each of the features. The dot product of the weights and the features maps each review to a probability of the review having positive sentiment or, equivalently, one minus the probability of it having negative sentiment. One can then threshold the probability to produce the class predictions for each input review.

2.2.2 Naive Bayes

We then used Naive Bayes with the same features as those used for logistic regression. Naive Bayes models both the probability of each class, as well as the probability of the review given its class, predicting an input review belongs to whichever class maximizes the product of those two probabilities. Naive Bayes is trained via MLE to learn the prior and conditional probabilities.

2.2.3 Feedforward Neural Network

Neural networks are a more general learning architecture, as there are many more parameters that can be adjusted to optimize the model. We used the same features for logistic regression and Naive Bayes as inputs to the network, and then experimented with different numbers of hidden nodes, ultimately choosing to use four hidden nodes with the $\tanh$ activation function. The output of these four hidden nodes was then combined in a single-node output layer with a sigmoid activation function, producing an estimated probability of the input review string having positive sentiment.

2.2.4 Recurrent Neural Network

In contrast to a feedforward network, in which output of nodes only cascade to the later layers in the network, a recurrent neural network allows the outputs of nodes to be connected to the inputs of nodes in previous layers. The human brain, on which neural networks were originally based, itself has a high amount of recurrence, suggesting that feedforward neural networks may be limited in their ability to perform tasks human brains find pretty easy, such as sentiment analysis. We constructed a recurrent neural network by passing the input reviews into a text vectorization layer, which converts strings to integer sequences, followed by an embedding layer, which reduces the dimensionality of the integer sequences. The sequence embeddings were then fed through a layer of 8 bidirectional (combines activations for both traversals of the layer's input) recurrent (the output also feeds directly back into its input) nodes, for an actual total of 16 hidden nodes.

2.2.5 Gated Recurrent Network

A gated recurrent network is a specific type of recurrent neural network. A GRU is similar to a long short-term memory, something we briefly mentioned in class, with a forget gate. It has fewer memories than an LSTM and is often faster. The combination of the update gate, the gate that determines which information from the previous hidden state and current input to keep, and the reset gate, which determines which information to discard, allows the GRU to retain relevant information from long sequences. Because of this, the GRU is able

to do a much better job looking back in its memory when creating a model, compared to RNNs. GRUs are often used for sequential data, such as language translation and language modeling.

For our GRU. we removed all numbers so we are only left with text strings. Using the 1000 most frequent tokens, we vectorized each review summary into a sequence 15 units long. 15 seemed good to us as it balanced retaining information in the summary, while not being too computationally heavy. We weighted the positive and negative classes using the same weights that were used in the previous models.

2.2.6 Transformer

Finally, we used a pre-trained transformer model called DistilBERT, which is a smaller, faster version of a much larger transformer called BERT. Transformers process an entire sequence of inputs at once, in contrast to the previous networks which process a sequence one token at a time. This forces the model to learn more complex interactions occurring throughout a sentence. Transformers achieve this through a mechanism called self-attention, which allows the model to differentially attend to parts of a sentence when predicting its class. This is especially useful in sentiment analysis for learning to handle more complex structures than simply word choice, such as negation.

2.3 Training Data Size

After running the models described above, we wanted to test how the model performance changed as we changed the size of our training data. We hypothesized that as the size of the training data increased, the models would improve. For the logistic regression and Naive Bayes models, we believed that the models would have a more representative probability distribution for each of the tokens in the universe. For the neural networks, we thought that the models would be able to extract more important features in determining sentiment. We were slightly concerned that as the size of the training data increased, the model may end up overfitting the training data, and have a lower accuracy score on the validation/testing data. However, the size of the datasets that we were using may be too small for this to be a genuine concern.

The different training sizes that we used were as follows: 20, 42, 88, 185, 389, 817, 1715, 3602, 7565, 15887.

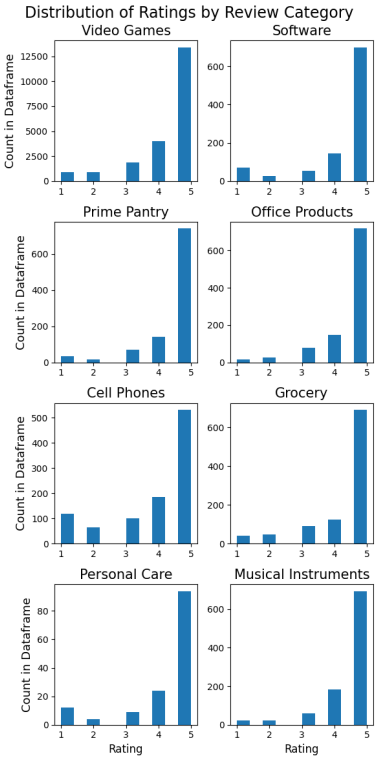

Figure 2: Distribution of ratings in each of the datasets. All of them are right-skewed, but some have more negative ratings than Video Games and others have more positive ratings.

2.4 Different Test Sets

Finally, we wanted to see how generalizable our model was. We trained it on video game data, but we wanted to see whether it could predict sentiment for other Amazon categories. The other categories that we tested the model on were software, musical instruments, prime pantry, office products, grocery, cell phones, and personal care. The distributions of ratings in these other datasets are shown in Figure 2.

3 Experiments

For our experiments, we first trained and evaluated each of the models on the video games dataset. We then considered how the size of the training set impacts each of the models, as well as how well each of the models generalizes to other review categories.

3.1 Implementation Details

For the models which require user-specified features as inputs (LR, NB, and NN), we used the following features:

1. Contains "!"

2. Number of Positive Words

3. Number of Negative Words

4. Length of Text

We could have also included n-gram features, but we decided not to in order to prevent overfitting due to the limited size of the training dataset. In particular, this likely would have hurt performance on generalization to other categories.

For all of the models except Naive Bayes (which models the priors) and the transformer (which we did not train), we needed to account for the disproportionate number of positive sentiment ratings in the training dataset. For logistic regression, this simply meant we specified that the class weight should be balanced. For the other models, we performed the LR balanced class weight calculation ourselves: $w_i = \frac{n}{2n_i}$, where n_i is the number of data points with label i in the training set. When we fit the model, we can provide these manually calculated class weights to prevent the model from always predicting positive sentiment.

For the network models, we used the binary cross-entropy loss function. Additionally, for the neural network and recurrent neural network, we used a text vectorization layer that keeps only the 1000 most common tokens in the training set and converts those to integers, followed by an embedding layer that produces a vector of length 32 for each word.

4 Results and Analysis

After training these models and evaluating them on an out-of-sample test dataset, we kept track of the following rates: accuracy, precision, recall, F-score. The results are summarized in Figure 3

4.1 Baseline Tests on Video Game Data

The baseline tests included training and testing the video game dataset. We used a sample size of 10,000.

4.1.1 Logistic Regression and Naive Bayes

The logistic regression and Naive Bayes models were there to help set our baseline. Overall, the logistic regression had an accuracy score of 72.0%. For positive metrics, it had a precision of 90.9%, recall of 75.1%, and F-score of 0.816. For negative metrics, it had a precision of 30.8%, recall of 61.0% and F-score of 0.409

The Naive Bayes had similar results with an accuracy of 74.7%, precision of 90.4%, recall of 78.2%, and F-score of 0.839, for positive metrics. For negative metrics, it had an accuracy score of 32.8%, recall of 56.2%, and F-score of 0.414.

4.1.2 Feedforward Neural Network

The feedforward neural network surprisingly had similar results. It had an accuracy of 72.9%, positive precision of 90.9%, recall of 75.2% and F-score of 0.824. It had a negative precision of 31.5%, recall of 60.2% and F-score of 0.414.

4.1.3 Recurrent Neural Network

As expected, the recurrent network outperformed the feedforward neural network, though not by a large margin, achieving an accuracy of 76%. On the positive metrics, it had a precision of 88.7%, a recall of 80.6%, and an F-score of 0.845. On the negative metrics, it had a precision of 40.3%, a recall of 56.1%, and an F-score of 0.469.

4.1.4 Gated Recurrent Network

The GRU had better results than the previous four models. We found an overall accuracy of 80.5%. For the positive metrics, it had a precision of 92.0%, recall of 84.5%, and F-score of 88.1%. For negative metrics, it had a precision of 38.7%, recall of 57.1%, and F-score of 0.461.

4.1.5 Transformer

The Transformer had better results than all the other models. It had an overall accuracy of 82.1%. For the positive metrics, it had a precision of 92.5%, recall of 85.6%, and F-score of 89.1. For the negative metrics, it had a precision of 42.0%, recall of 59.3%, and F-score of 0.492.

Looking at these results, we first noticed that as the model complexity increased, the model accuracy also improved. The logistic regression and Naive Bayes were our simplest models, with the lowest accuracy scores and lowest F-scores. The feedforward neural network and recurrent neural network were more complex than the first two models, and had slightly higher scores. Finally, the GRU and Transformer are much more complicated models and had higher accuracy scores and F-scores than the previous models.

We also find that the precision, recall and F-scores are lower for the negative metrics compared to the positive metrics. This makes sense to us because there is much less negative data in general.

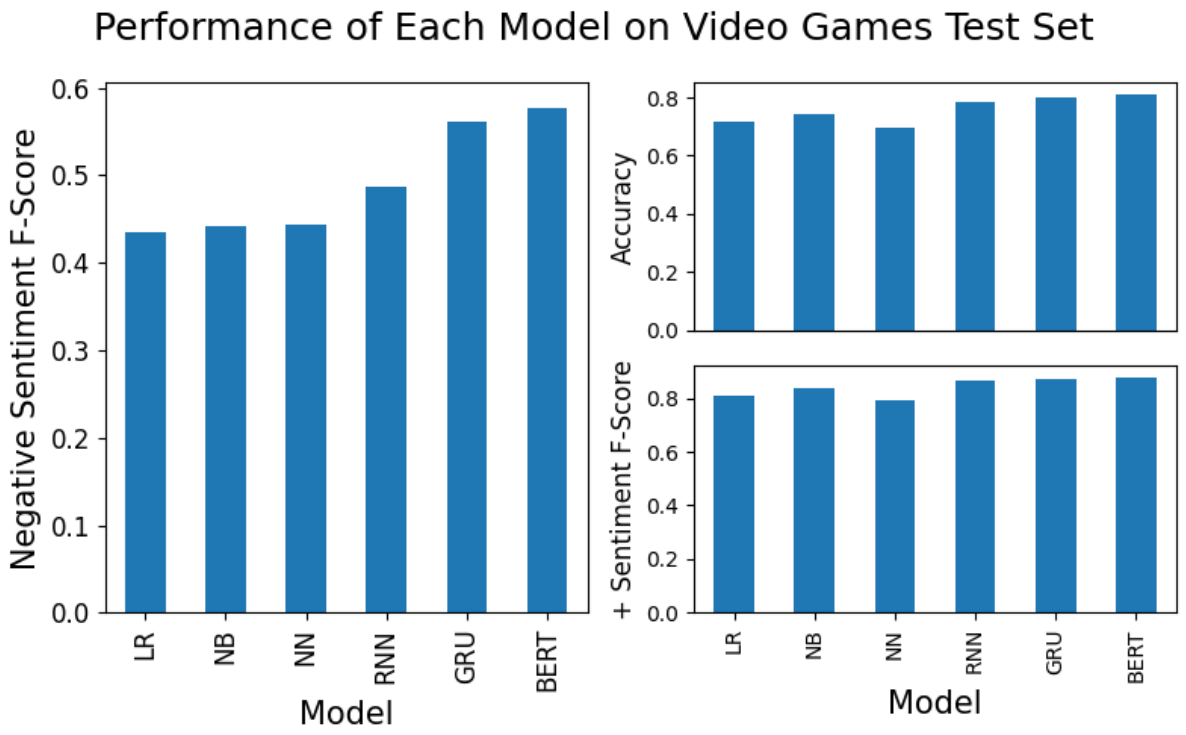

Figure 3: Performance of each model on the video games test dataset after training on the video games training dataset, evaluated via F-score and accuracy.

Therefore, it is likely difficult for the models to pick up the traits of a negative review, which results in worse performance.

We also find that the F-score improvement between the positive metric models and negative metric models is about the same. However, there are differences in precision and recall. For positive metrics, precision increased by 2% and recall improved by roughly 10%. For the negative metrics, precision improved by roughly 11%, but recall stays about the same in each model. Our initial thought was that this may occur because the negative sample size is smaller so the more complex models are able to do a better job picking up the features, however it doesn't make sense why that would apply to precision, but not recall.

4.2 Varying the Training Size

We continued by varying the size of the training data. In this test, we exclude the Transformer because that was pre-trained. The results are summarized in Figure 4. The GRU had the highest accuracy, and maintained that accuracy throughout, even with the smallest sample size of 20. The RNN also began with a high accuracy score of 80%. The RNN may be predicting that all the reviews are positive. Because 80% of the reviews were positive, it would result in a high accuracy score. The Naive Bayes, logistic regression, and feedforward neural net all began with a low accuracy rate and continued to increase throughout the model. The Neural Net displayed some unexpected volatility in its accuracy rate for various training data sizes. We expect that the neural net may not be reaching convergence, but have to do some more research.

For the positive F-score, as the sample size increases, all the models converge to roughly the same statistic - slightly above 0.8. The positive F-scores seemed to have a similar trajectory compared to the accuracy rate. The larger difference occurred with the negative sentiment F-scores. Here, we observed a few things. First, all the models began with a low F-score and increased over time. The simpler models, the logistic regression, and Naive Bayes reached their maximum F-score quickly and then flatted off. After a training size of 100, the models' negative F-scores didn't approve. However, for the more complex models like the GRU and RNN, the negative sentiment F-score continued to increase over all the sample sizes. We expect to see a larger increase in this F-score because there is little negative data. As the sample size increases, the number of negative

reviews increases significantly and the model has more information to train on.

4.3 Different Test Sets

After training the model using 16,000 data points of the video game data, we tested it on various categories. The results are summarized in Figure 5. We found that the accuracy scores and positive sentiment F-scores were similar across all the categories. However, there was a significant difference in negative sentiment F-scores. In decreasing order, the negative sentiment F-scores for all models were for personal care, cell phones, grocery, video games, office products, prime pantry, musical instruments, and software. Notably, video games did not have the highest accuracy of F-scores, given that the data was trained on it. This increase in performance makes sense when looking at the distributions of ratings for each category in Figure 2, as in general, the categories with increased accuracy have even more disproportionately positive reviews, and all of our models perform better on positive data. Interestingly, the differences in performance across data sets were not uniform across the models. Naive Bayes sometimes outperformed the more advanced models, indicating that their added complexity may have led to overfitting the training data.

5 Conclusion

In this study, we were looking to create various models to predict sentiment from Amazon reviews. We used six different models - logistic regression, Naive Bayes, feedforward neural network, recurrent neural network, gated recurrent network and a transformer. Our initial tests demonstrated that as the complexity of model architecture increased, model performance improved. This is logical as the more complex models can extract more features like sentence structure, context, etc to predict sentiment, that the simpler models can't. We then continued by changing the size of the training data. We found that as the size of the training data increased, the accuracy scores increased slightly, but they tapered off fast. We found a similar result with the positive F-score. However, with the negative F-score, we found that all the models improved as the sample size increased. The simpler models - logistic regression and Naive Bayes - increased slightly and then tapered off. The more complex models - recurrent neural network and gated recurrent network - both continued to increase as the sample size increased.

We believe that the lower F-scores for the negative scores and larger increase as the sample size increased is because there is less negative data in general. Therefore, as we increase the size of our total sample data, the size of our negative data set is growing much faster relative to the increase in the size of the total data set. This larger dataset allows the model to do a better job in its predictions.

As a final test, we looked to see how well our model generalizes to other categories. We trained the data on video game data and tested it on Amazon reviews of other categories. We found mixed results where some categories performed better, and others performed worse. This appears to be explained by the percentage of positive reviews the category has. The categories that had a higher percentage of positive reviews did better. The model may be skewed to predicting positive reviews and therefore do better on the models with more positive reviews.

For further studies, we would begin by improving our feature set of our logistic regression and Naive Bayes models. These are computationally cheaper methods. If we are able to achieve high accuracy with a better feature set, we may not need to use more complex and expensive models. We would continue by changing the training dataset. Video games have some very specific terminology that isn't used in other categories, such as "ffs". If we trained on a more general Amazon review category, we may have better results out-of-sample.

Acknowledgments

We would like to thank Professor Frank for his guidance and support throughout the project. In addition, we would like to thank him for teaching an incredible course. None of us came in with any prior introduction to linguistics, and we are excited to continue to explore it in future classes. This course helped us expand our understanding of the applications of various models that we have learned in other classes. This course also helped refine our understanding of how various machine-learning models work. Computational linguistics seems to be a rapidly growing field, and we're excited to continue to watch it grow.

Figure 4: Impact of varying the size of the training dataset on accuracy, as well as negative and positive F-scores. The transformer is not included here as it was pre-trained, so we were unable to change its training.

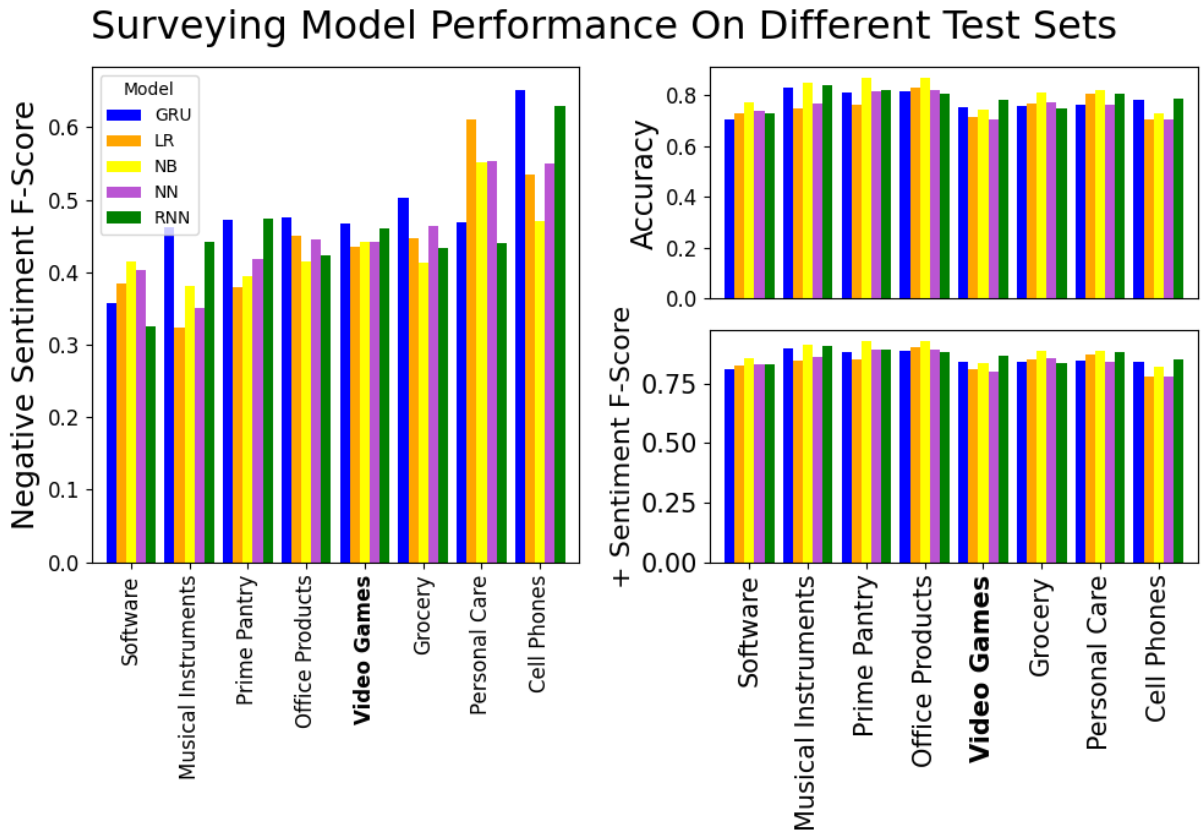

Figure 5: Performance of our models trained on the video games dataset, tested on test sets consisting of reviews from different categories.

References

Tim C. Kietzmann, Courtney J. Spoerer, Lynn K. A. Sörensen, Radoslaw M. Cichy, Olaf Hauk, and Nikolaus Kriegeskorte. 2019. Recurrence is required to capture the representational dynamics of the human visual system. *Proceedings of the National Academy of Sciences*, 116(43):21854–21863.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fined-grained aspects. *Empirical Methods in Natural Language Processing (EMNLP)*.