Predicting the Helpfulness of Amazon Product Reviews With BERT

Tucker Anderson, Keane Johnson
University of California, Berkeley
Berkeley, California
tuckera@berkeley.edu
keanejohnson@berkeley.edu

Abstract

In this project, we seek to create a model to predict review helpfulness based upon a large corpus of Amazon review data. Using transfer learning of Bidirectional Encoder Representations from Transformers (BERT) we hope to jump-start our review text encoding and to perhaps capture some underlying features or positioning attributes that will improve the helpfulness of online user reviews.

1 Introduction

The rise of online commerce in the last decade has made it increasingly easy to purchase any item without leaving the comfort of one's own home. But there is a catch. Without physically interacting with a product in person, how can a consumer tell if the product's quality is worth their money?

Most consumers use the reviews from other individuals that previously bought the product to make their own buying decisions. But popular items on Amazon have thousands of reviews and no consumer will read through them all. Thus, it is in the interest of online retailers to identify and promote the most helpful product reviews to enable their customers' buying decisions.

Most online retailers have a manual system to facilitate this. Consumers rate each other's reviews as being "helpful" or "not helpful," and then the retailer displays those that are the most helpful - determined by a "helpfulness percentage" or raw number of helpful votes.

But this is a self-fulling approach, where the most helpful reviews will get the most visibility, and continue to be voted upon. As a result, retailers may miss out on promoting reviews that are helpful, but just have not gained traction.

So it is beneficial to both retailers and potential buyers to identify helpful reviews before they are evaluated by consumers. This is challenging because a review's helpfulness is influenced by a number of factors, including its scope, content and sentiment.

2 Background

Identifying helpful product reviews is not a novel problem. The large revenue implications of facilitating customer transactions have motivated previous academic and industry research into predicting the helpfulness of product reviews. However, the approaches have evolved over time as new methodologies and technology become more accessible.

The root of this problem is text classification. To that end, the first attempts at solving this challenge focused on linear regression. Kim et al. (2006) applied Support Vector Machines using a variety of different review features. These features included structural variables like length, lexical features like the TF-IDF statistic of each word, syntactic features like part-of-speech tags, and meta-data features, like the number of stars a review has. (Kim et al., 2006) identified three factors that affected helpfulness the most: the length of the review, the product rating and the TF-IDF score of words used.

Second approaches put less importance on these non-text features and focus on the semantics and context of the review. Yang et al. (2015) applies semantic features that help capture a reviewer's opinions, analyses, emotions, and personal experiences. Tang et al. (2013) takes a similar approach but focuses on modeling the content context and social context of the review.

Recent approaches continue to focus on the text of the review but apply more sophisticated models. These models apply the success of Convolutional Neural Networks (CNNs) in many natural language processing tasks. Chen et al. (2018a) applies a CNN built with word- and character-level embeddings. The character embeddings are fed through a convolutional layer and max-pooling layer, and then concatenated with the original word embedding. The review's final representation is created by concatenating all these more robust word embeddings together.

Expanding upon this work, (Chen et al., 2018b) produces a series of CNN models that incorporate character, word, and topic-level representations. The final model includes a gating layer that weights the importance of each word. All of these CNN models outperform the non-CNN baselines, indicating that CNNs are better equipped to capture the semantic meaning of reviews.

3 Methods

The key takeaway from the previous work is that in order to judge the helpfulness of a review, it is crucial to capture the meaning and context of the words within that review. Yang et al. (2015) and (Tang et al., 2013) attempted to accomplish this by capturing the semantic meaning and context of the words making up a review. Chen et al. (2018a) and (Chen et al., 2018b) attempted the same by using character-, word-, and topic-level embeddings fed through a CNN.

It is apparent that future improvements in determining the helpfulness of a review will come by developing models that can better represent the meaning and context of words. Our work pursues this goal by applying the state-of-the-art in learning contextual relations between words - BERT.

3.1 The Dataset

We use the Amazon product review dataset assembled by Julian McAuley (He and McAuley, 2016), associate professor in the Computer Science department at the University of California, San Diego. This dataset has been used by many previous studies in the review helpfulness space, including (Yang et al., 2015), (Chen et al., 2018a), and (Chen et al., 2018b).

The dataset consists of 142.8 million reviews across twenty-four product categories spanning from May 1996 to July 2014. Each row includes information about the review and product metadata. This includes ratings, text, helpfulness votes, descriptions, category information, price, brand, and image features. We focus on the text and helpfulness votes.

We focus on five product categories that were previously used in (Chen et al., 2018b): Cellphones and Accessories; Clothing, Shoes, and Jewelry; Sports and Outdoors; Home and Kitchen; and Electronics.

We perform some minor data pre-processing to clean up the dataset because we want to be purposeful with the data that we are supplying to our model. We want to only analyze reviews where there are at least three total votes, or reviews where there are two total votes, but both votes are in agreement. Our rationale is that two total votes that are split does not tell us much. However, if both are an agreement, that could be indicative of a review's helpfulness.

The size of the raw and post-processing datasets by category are presented in Table 1.

Category	**Raw**	**Processed**
Cellphones and Accessories	194,439	23,534
Clothing, Shoes and Jewelry	278,677	41,057
Sports and Outdoors	296,337	57,670
Home and Kitchen	551,682	139,470[1]
Electronics	800,000[2]	216,218[3]

[1] Due to memory constraints, the size of the processed Home and Kitchen dataset was reduced to 80,000 for the baseline models.
[2] Due to memory constraints, the size of the raw Electronics dataset was reduced from 1.7 million rows to 800,000.
[3] Due to memory constraints, the size of the processed Electronics dataset was reduced to 80,000 for the baseline models.

Table 1: Size of raw and processed datasets by category.

3.2 Measuring Model Performance

In assessing the performance of our models, we think about how they would be applied by potential online retailers. With this context, we are less concerned about predicting the exact number of helpfulness votes or helpfulness percentage of a review because these will change over time as a review ages. Rather, we want to measure whether our predicted helpfulness percentage is correlated with the review's actual helpfulness percentage. We want our model to predict that the most helpful reviews will be the most helpful and the least helpful reviews will be the least helpful, masking over any slight differences in actual helpfulness percentage. The Pearson product-moment correlation coefficient measures the linear correlation between two variables and represents this relationship well. We use this statistic to measure the difference between the test sets actual labels and our models' predicted labels.

$$\rho = \frac{cov(X,Y)}{\sigma_X \cdot \sigma_Y} \tag{1}$$

where *cov(X, Y)* is the covariance between the random variables X and Y, or the actual helpfulness percentages of the test set and the predicted percentages of the test set, and σ_X and σ_Y are the standard deviations of the actual labels and predicted labels, respectively.

The Pearson product-moment correlation coefficient is bound between -1 and +1, with -1 being absolute negative correlation, 0 being no correlation, and 1 being absolute positive correlation.

4 Models

We build four different models: three baselines that have increasing complexity and the BERT model.

4.1 Helpful Reviews Are Determined Randomly

The easiest way to select helpful reviews is by random. In this case, the model does not use the review text or any other features. Rather, it generates a random number between one and zero for the predicted helpful percentage. We expect the correlation coefficient between actual helpful percentages and the randomly predicted helpful percentages to be close to zero.

4.2 Encoded Words and Linear Regression

The second model incorporates review text into the model. The reviews are tokenized, stemmed and preprocesssed with a label encoder. The encoded data is used to train a linear regression which then predicts helpful percentages. The correlation between the actual test helpful percentages and the predicted helpful percentages is then calculated to determine the model's accuracy.

4.3 TF-IDF and Linear Regression

The third model builds upon the concept of the encoded words model by using TF-IDF to determine the importance of the words within a review. The review text is stemmed and vectorized, with each review considered a document. A matrix of TF-IDF values is generated for each review, which are then fed into a linear regression. The correlation between the actual test helpful percentages and the predicted helpful percentages is then calculated to determine the model's accuracy.

4.4 BERT

The final model continues to build on the theme of the importance of the review text by incorporating BERT. The key advantage of BERT is its bidirectionality, which enables a deeper understanding of language and context. In our BERT model, we encode review summary and review text into BERT token id, segment id and masking id. From there, our baseline BERT model was built with a few fine tune layers and first pooling before emitting to another fully connected layer and then a final single dense layer. Additional hyperparameters like max sequence length, batch size, and training epochs were manually tuned as the model was developed. We used root mean square error and tested cross entropy as the loss functions. Below is the design of the proposed neural network layout.

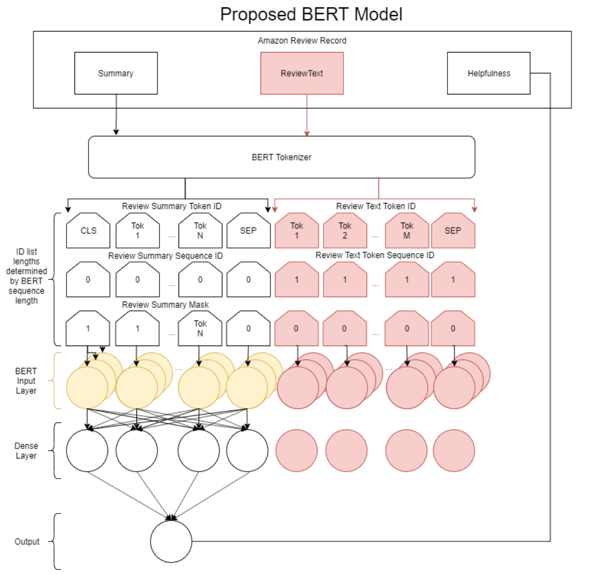

Figure 1: BERT Model Layout

Memory constraints forced us to severely limit the size of the datasets used to train the BERT models. Although previous baselines incorporated up to 80,000 rows of data, our BERT models could only train on a maximum of 10,000 rows. Even then, cloud instances needed to be upgraded to 16GB of RAM to hold the BERT model in memory.

As a result, we experimented with a variety of different data subsets to try to maximize the return of our compute resources, including: (1) only reviews with at least three total votes or two total votes that were in agreement, (2) only reviews with at least five total votes, (3) only reviews with at least ten total votes, (4) only reviews with a zero helpful percentage or a one hundred helpful percentage and an equal number of each, (5) an equal number of reviews with a zero helpful percentage and a one hundred helpful percentage and all the reviews with helpful percentages in-between, (6) reviews with smaller BERT sequence lengths (126, 256, and 512), (7) only examining the brief summary text instead of combining it with the longer review text (the model above with the red data and nodes removed), (8) introduction of kernel level l1 and l2 regularization to reduce reliance on each of our nodes and (9) removal of the BERT input layer and using a simple combination layer (the model above with the yellow nodes replaced with a simple concatenation).

The final models we trained only looked at reviews that met a threshold of total reviews - between five and ten depending on the resulting size of the dataset - and that had either a zero or one hundred percent helpfulness rating.

5 Results and Discussion

The results of our models are presented in Table 2. Across all categories, randomly determining a review's helpfulness performs the worst, with a correlation coefficient close to zero. This is expected because we are not using any part of the review to determine helpfulness and so predicted labels should not be correlated with the actual labels.

Using encoded words and linear regression shows a positive but weak correlation between predicted helpfulness percentage and actual helpfulness percentage. This shows that incorporating aspects of the review text is beneficial in determining the helpfulness of the reviews.

Incorporating TF-IDF produces a much stronger correlation between the predicted and helpful percentages than using just word encodings, showing that a word's relative importance is significant in determining a review's helpfulness.

Unfortunately, our BERT model did not perform better than our TF-IDF model. The BERT model only provided a strong, positive correlation coefficient in one category: Sports and Outdoors. And even then, it did not outperform the TF-IDF baseline.

We theorize that BERT performed poorly on this task for a couple reasons. First is the quality of our data. Although the data was clean and easy to digest, it was heavily skewed to 0% and 100% helpful reviews.

	Cell.	**Cloth.**	**Out.**	**Home**	**Elec.**
DR	-0.002	0.003	-0.004	0.001	0.001
EW	0.028	0.034	0.022	0.009	0.012
TF-IDF	0.255	0.257	0.322	0.258	0.365
BERT	-0.034	-0.161	0.232	-0.176	-0.036

Table 2: Performance of different models in the following dataset categories: Cellphones and Accessories; Clothing, Shoes, and Jewelry; Sports and Outdoors; Home and Kitchen; and Electronics.

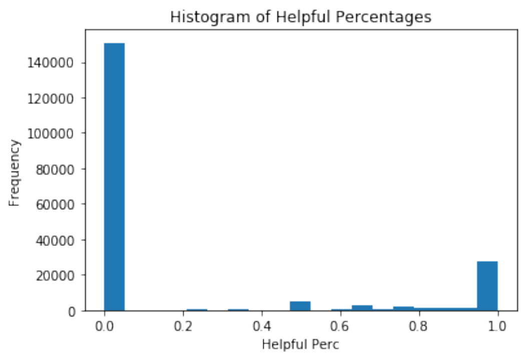

Figure 2: Histogram of cellphone helpful percentages prior to cleaning

We attempted to further clean the data such that we only fed reviews that received at least five or ten total votes depending on the category, and only reviews that were zero percent or one hundred percent helpful. Our reasoning is that these are the most indicative of helpful or non-helpful reviews. The percentage shows that all voters are in agreement, and the high number of total votes shows that those voters constituted a large group. This is admittedly a naive approach, and perhaps some more nuanced data selection could provide an increase in performance.

The losses of our training epochs illustrate that our data may have been responsible for BERT's under-performance. In most epochs, no matter the category, the validation loss is lower than the training loss. This indicates that the model is better at fitting the unknown than the known. Because this is counter-intuitive to machine learning, we theorize that our training and validation datasets are somehow substantially different, meaning that our test dataset is also likely distinct. This is likely because of the small datasets we use to train. Because of memory constraints, we use training sets of 300 reviews, validation sets of 100, and test sets of 100.

6 Conclusion and Continuing Development

From our model results, we see that our current BERT model has trouble correctly modelling the helpfulness of review, especially in comparison to our TF-IDF and bag of words models. In training the BERT model we had a significant number of issues that could have lead to improper training, and subsequently a large amount of items we had no time to address that we would have liked to.

6.1 Other Features

Our model takes in information directly scraped from Amazon reviews. In our investigations, we focused mostly on the text features (summary and review text) within each review but there were a few other features included that may have proven to be beneficial in a compound model. We would have liked to explore more to determine if features such as star rating, pictures, and user history add strength to a review helpfulness prediction model. Given our issues with solely pure text data, we did not explore these possibilities in-depth.

6.2 Neural Network Hyperparameters

Our neural network model necessitated us using small batch sizes during training, but we would have liked to explore larger batch sizes and sequence lengths with better TPU and GPU access to improve model accuracy. We would also have liked to tweak the amount and depth of our fully connected layers, and possibly even add a convolutional layer as explored in (Chen et al., 2018a) and (Chen et al., 2018b). We attempted some forms of regularization during our attempts to address the memory leak issues, but we would have liked to more fully explored kernel level l1 and l2 regularization and dropout, as we feel such large volumes of data could lead to model overfitting.

6.3 Bias in our Model and Data

Considering the large amount of data manipulation we needed to conduct, mostly due to issues mentioned in the BERT memory constraints, and the way the review feedback system is set up on Amazon we believe there were some biases in the dataset that were not fully controlled for. In attempts to truncate the reviews to fit our large model, it is possible that we removed some of the most relevant features of the review text, such as the conclusions of each reviewer (we generally focused on slicing the earliest review text).

Considering the large financial incentive to provide fake reviews for products on Amazon, and to possibly increase the helpfulness score of each review, it's likely that there is bias in our review dataset. In our project we did not control or observe this possibility, but to state that the review methodology and helpfulness tagging is a "base truth" of helpfulness would not entirely be accurate. In addition, reviews that are regarded as "more helpful" are shown with a greater likelihood on Amazon, which in turn could lead to a positive feedback loop of review helpfulness score. If possible we would like to have access to a more rigorously controlled and universally observed set of reviews to ensure a more accurate representation of review helpfulness.

6.4 BERT & Memory Constraints

The BERT embedding functions were trained on 64GB cloud TPU, producing embeddings with a max sequence length of 512. This sequence length is not enough to capture full review text from all reviews, and our current resources kept us from utilizing such a memory intensive sequence length. Due to the constraints of our hardware, we attempted to utilize 256, 128 and 64 sequence lengths and smaller and smaller batch sizes to avoid memory leak issues during training of our model. These memory issues, along with our surprisingly large fully connected neural network layout yielded memory issues even with batch sizes of 1 and 64 sequence lengths. We would have liked to determine the root cause of our memory and convergence issues.

BERT embeddings are generally used to classify sentence level tokens, where as we were attempting to expand this role to a document level embedding with a final regression output of weighted review helpfulness. It's possible this deviation from the original intent of these embeddings at least partially resulted in our poor results. As a result, we might want to look into using other word embeddings such as ELMo and GloVe.

Most importantly, we would like to explore the intricacies of why our BERT tokenized model had trouble converging and leaking memory. It seems like such a complex transfer learning model may not lend itself to a relatively simple dataset with a large corpus of long document strings. If we had access to a cloud instance with a TPU with a large memory, it's possible our issues would be subdued. Ultimately, our BERT model may have been too complicated for what we set out to do with our resources.

References

Cen Chen, Yinfei Yang, Jun Zhou, Xiaolong Li, Forrest Sheng Bao 2018. Cross-Domain Review Helpfulness Prediction based on Convolutional Neural Networks with Auxiliary Domain Discriminators. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, vol 2 (Short Papers)*, vol 2. pp 602–607.

Cen Chen, Minghui Qiu, Yinfei Yang , Jun Zhou, Jun Huang, Xiaolong Li, Forrest S. Bao. 2018. Review Helpfulness Prediction with Embedding-Gated CNN. *arXiv preprint arXiv:1808.09896*

Chi Sun, Xipeng Qiu, Yige Xu, Xuanjing Huang 2019. How to finetune BERT for text classification? In *CoRR abs/1905.05583*

Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *arXiv preprint arXiv:1810.04805*

Gerardo Ocampo Diaz and Vincent Ng. 2018. Modeling and prediction of online product review helpfulness: a survey. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp 698–708.

Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. 2013. Context-aware review helpfulness rating prediction. In *Proceedings of the 7th ACM Conference on Recommender System*, pp 1-8.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering In *WWW*.

Soo-Min Kim, Patrick Pantel, Tim Chklovski, and Marco Pennacchiotti. 2006. Automatically assessing review helpfulness. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp 423-430.

Yinfei Yang, Yaowei Yan, Minghui Qiu, and Forrest Bao. 2015. Semantic analysis and helpfulness prediction of text for online product reviews. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, (Volume 2: Short Papers), pp 38–44.