Evaluate Helpfulness in Amazon Reviews Using Deep Learning

Bobby Nguy
Stanford University
cnguy002@stanford.edu

Abstract

Understanding how users perceive reviews' quality is crucial since it can reveal insights on what drives a user's purchasing decisions. We examine this problem by looking at Amazon review data and its helpfulness rating. We proceed by casting it as a binary classification problem using deep learning models such as RNN and LSTM. Despite presence of preferential attachment and other social biases, our results suggest that users value keywords and relevancy over sentence structure and readability. In addition, it also seem to suggest that users may have a limited "attention span" for reading each review.

1 Introduction

Consumer reviews are abundant in today's e-commerce setting. Yet not all reviews are perceived equally. Some seem to be more influential than the others, and are more relevant to a user's purchasing decision. It will be helpful to understand what constitutes a helpful review. Different from sentiment analysis, this task can be difficult even for a human agent. Two reviews may both look informative, but often one was perceived as helpful and the other was not. We would like to evaluate the performance of NLP and deep learning on such task.

In our problem setting, we examine Amazon reviews data, which provides an option for helpfulness votes. If choose to, a user can upvote or downvote a review on its helpfulness. For example $3/5$ would mean that three people thought the review was helpful and two thought otherwise. There might be many more who saw the review but did not express their views.

We approach the problem by casting it as a classification problem. Each review is labeled as "helpful" or "unhelpful"; we then predict the labels by various algorithms. Here we will use Bag-of-Words with Random Forest as a baseline. In addition, we ran simple-RNN and LSTM on the data and compare the outcome.

2 Related Work

Studies from Yun Wan et al [1] indicate that Amazon helpfulness ratings may suffer from sequential and self-selection bias, and might not be reflective of the "true quality of the review".

Danescu-Niculescu-Mizil et al [2] examined the phenomena from a social psychological perspective and considered various opinion distributions in review rating. They also confirmed that text quality is not the only explanatory factor in helpfulness rating.

Given these circumstances, compared to well-researched problems such as sentiment analysis, previous attempts to predict helpfulness ratings were only able to achieve moderate accuracy rate (approx. 70-80%). These attempts mostly use handcrafted features and apply them with Support Vector Machines. Many of these attempts however tend to discard the zero-vote population and hence cast

doubt on the generality of the results. We will further discuss this issue and our approach in the later section. Nonetheless, these attempts agree that review lengths and review texts are correlated with helpfulness perception. Little is revealed, however, as to in what ways review texts affect its perceived quality.

3 Approach

We will apply bag-of-words(BoW) with random forest, simple Recurrent Neural Network(RNN) and Long Short Term Memory (LSTM) with various hyperparameters to determine the best performing model.

The bag-of-words model is a simple model that keeps the frequency of words occurrence. In our experiment, we restrict our feature space to 5000 words that occur the most frequently. Random forest, a ensemble tree classifier, utilizes these BoW features and make subsequent predictions. On the other hand, RNN and LSTM are neural network models that are able to capture information from sequences of words.

In a RNN, each input will be fed forward sequentially, through a number of hidden states, and compute a probability output vector.

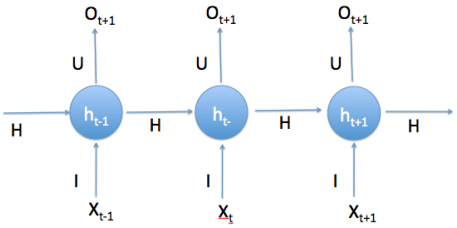

Figure 1: Flow diagram of RNN

The mathematical details are as follows [9]:

$$
\begin{aligned}
\boldsymbol{e}^{(t)} &= \boldsymbol{x}^{(t)}\boldsymbol{L} \\
\boldsymbol{h}^{(t)} &= \text{sigmoid}\left(\boldsymbol{h}^{(t-1)}\boldsymbol{H} + e^{(t)}\boldsymbol{I} + \boldsymbol{b}_1\right) \\
\hat{\boldsymbol{y}}^{(t)} &= \text{softmax}\left(\boldsymbol{h}^{(t)}\boldsymbol{U} + \boldsymbol{b}_2\right) \\
\bar{P}(\boldsymbol{x}_{t+1} = \boldsymbol{v}_j \mid \boldsymbol{x}_t, \ldots, \boldsymbol{x}_1) &= \hat{y}_j^{(t)}
\end{aligned}
$$

LSTM is a variant of recurrent neural network with more complex units that include a memory cell [9]:

$$
\begin{aligned}
i_t &= \sigma\left(W^{(i)}x_t + U^{(i)}h_{t-1}\right) \\
f_t &= \sigma\left(W^{(f)}x_t + U^{(f)}h_{t-1}\right) \\
o_t &= \sigma\left(W^{(o)}x_t + U^{(o)}h_{t-1}\right) \\
\tilde{c}_t &= \tanh\left(W^{(c)}x_t + U^{(c)}h_{t-1}\right) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \\
h_t &= o_t \circ \tanh(c_t)
\end{aligned}
$$

where i is the input gate, f is the forget gate, o the output gate and c the new memory cell.

The rationale behind these choices of models is that we can evaluate different aspects of review text in our problem. Bag-of-words is a good baseline for keyword-only evaluation, while RNN and LSTM also captures sequence information and can disambiguate various meanings of the same words. Implementation is performed using TFLearn, a deep learning library built on top of TensorFlow.

4 Experiment

4.1 Data and setup

Our experiments use Amazon review data collected by Stanford SNAP, publicly available at:

`https://snap.stanford.edu/data/web-Amazon.html`

The data comes from reviews for 24 product categories, where users provide comments or feedback on their product experiences. Specifically, we used the reviews data from the *Digital Music*, *Musical Instruments*, and *Patio, Lawn and Garden* categories, totaling to 88239 reviews. The reviews range from 1 to 2055 words, with an average of 147.

Each review comes with a helpful rating [x, y], directly relatable to the annotation "x out of y people found this review helpful" on Amazon website. Note that there might be many more who read the review but did not vote. In many cases, reviews have never been voted and have helpful rating [0,0].

Previous literature has focused on using helpfulness ratio, i.e. x/y, in making predictions and analyses. There are several problems in this approach:

1. Many reviews ($>$70%) are never rated and cannot be accounted for without correction. A lot work simply disregards this bulk of the population in their analyses.
2. Viewing each [x, y] pair as binomial (y, p_i), variance of the helpfulness ratio is relatively high with small y.
3. Preferential attachment is likely to be at force here for the helpfulness rating. We plot the proportion of reviews with x upvotes (Figure 2) and its log-log plot in (Figure 3). As suggested from the relatively straight line (until the tail portion, where variance is not stabilized), in the log-log plot, the distribution of x follows the power law, which can be generated from a network with preferential attachment. Preferential attachment is a result of the richer-gets-richer effects, where reviews that happen to have already possessed some upvotes are much likelier to gain more upvotes faster than the otherwise.

Since our focus is to discover any hints in review text that relates to helpfulness, we want to restrict measuring the effect of preferential attachment. As a result, we decide to cast the problem as a binary classification problem rather than a regression problem. To set up binary labels for the dataset, we first perform correction on the data by adding one upvote and one downvote to each observation (i.e. 1 to x and 2 to y). Hence, all cases of [0,0] will now have a helpfulness ratio of 0.5. We then label reviews with ratio $>$0.5 as "helpful" and "unhelpful" otherwise.

We then perform NLP and deep learning on the review text and evaluate the model using cross entropy loss. The score metrics that we are using are F1 scores and Accuracy:

$$F1 = 2 \times precision \times recall \,/\, (precision + recall)$$

$$Accuracy = (truePositives + trueNegatives)/totalEntries$$

We expect the model to explain a proportion of variances in the "helpful" and "unhelpful" text. The neural network models are expected to perform better than the bag-of-words baseline since LSTM captures both keywords and sentence structure, where readability is assumed to play a role in indicating helpfulness. But because of various biases, including the effect of preferential attachment, we expect none of the models to have a very high accuracy ($>$90%).

4.2 Models

We performed classification using review text as features on the following models:

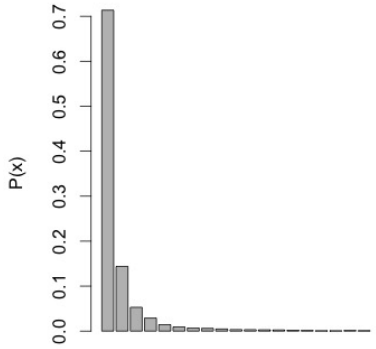

Figure 2: Proportion of reviews with upvotes = x.

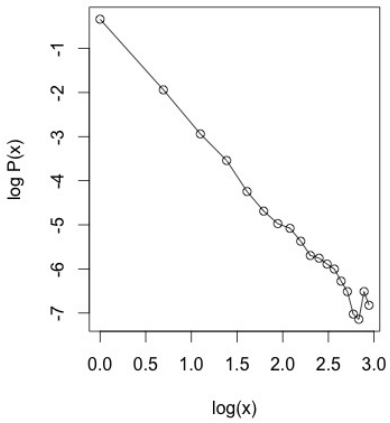

Figure 3: log-log plot of cumulative probability of reviews with upvotes = x vs x.

- Bag-of-Words (BoW) of unigrams, with max. feature dimension = 5000
- Bag-of-Words of bigrams, with max. feature dimension = 5000
- Simple RNN
- 1-Layer LSTM
- 2-Layers LSTM
- 3-Layers LSTM
- 2-Layers LSTM, with review length and rating as extra features

The neural network models are all set up with the following hyperparameters:

- Max. number words to read in review = 100
- Dropout = 0.5
- NumNeurons = 50
- Dimension of Word Embedding = 50

4.3 Glove Vectors

We compare the model performance from training our own word vectors and using pre-trained GloVe vectors. Due to our limited corpus size, our trained vectors are expected to be poor representation of the word dimensions. In fact, using GloVe vectors boosts our neural networks model performance by more than 7 percent points in accuracy. All of the following neural network models use GloVe vectors as its word vector representation.

4.4 Results and Findings

The baseline method, unigram bag-of-words with random forest, achieves a F1 score of 0.565 and an accuracy of 0.619. Bigram bag-of-words trails behind with F1 score of 0.547 and accuracy of 0.582. Both models' performances are comparable to two-layers LSTM, which achieves a F1 score of 0.549 and an accuracy of 0.65.

One-layer RNN has the worst performance: with a F1 score of 0.156 and accuracy 0.555, it borders on random guessing. It is a rather surprising result, especially considering the simplicity of the bag-of-words algorithm. Since bag-of-words capture the differences in keywords in the reviews, simple

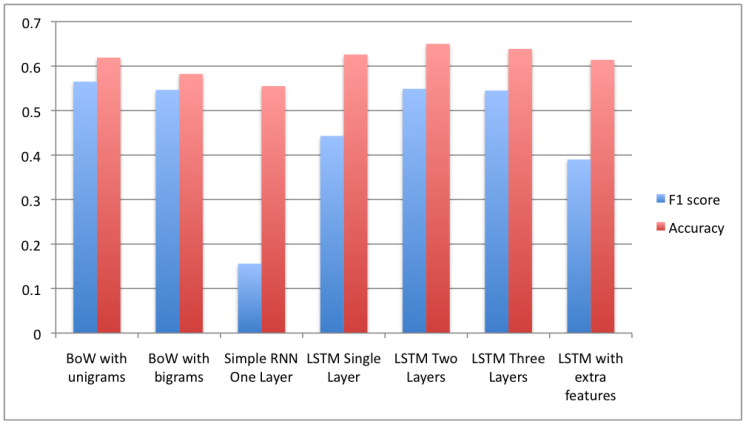

Figure 4: Experiment Results with various models

RNN the sentence structure and LSTM both of the features, it would suggest that the sentence structure and readability is of little importance when deciding whether a review is helpful or not.

Diagnosing the fitted random forest for variable importance, we get the following top predictor unigrams:

1. feature 99 : album (0.032975)
2. feature 4102 : songs (0.018699)
3. feature 2907 : music (0.015340)
4. feature 661 : cd (0.014586)
5. feature 4101 : song (0.011020)
6. feature 100 : albums (0.009514)
7. feature 3059 : one (0.008051)
8. feature 390 : best (0.007177)
9. feature 3492 : quot (0.007118)
10. feature 2533 : like (0.007077)

We repeat the same procedure and obtain the top predicting bigrams:

1. feature 1879 : hip hop (0.003301)
2. feature 2881 : one best (0.002388)
3. feature 3399 : quot quot (0.002237)
4. feature 1028 : easy use (0.001999)
5. feature 4372 : title track (0.001929)
6. feature 4854 : works well (0.001901)
7. feature 3940 : songs like (0.001882)
8. feature 1138 : every song (0.001683)
9. feature 1877 : highly recommend (0.001642)
10. feature 4847 : works great (0.001634)

The top unigram features seem to contain major keywords related to the product categories, while the top bigrams appear to convey positive sentiments. We may suggest that relevancy and sentiment reinforcement are one of the elements consumers value in reviews. Interestingly, extra features of review length and star ratings do not improve the model. This might be because the information in these two variables is already implied in the review text input.

Next, we further break down the results from the LSTM with extra features into subcategories and compare their metric scores (Figure 5). The scores for each category is similar; with the model performing slightly better on Musical Instrument. It does not appear that helpfulness evaluation is easier to identify in one product against the others.

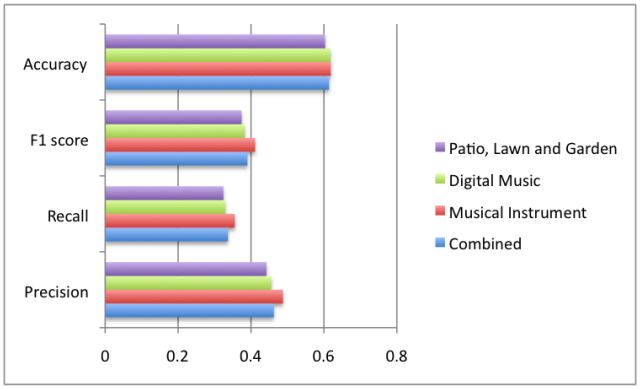

Figure 5: Performance metrics by categories

4.5 Hyperparameter Tuning

From Figure 4, while two-layers LSTM perform better than one-layer LSTM, three-layers LSTM shows no improvement over two-layers one. Very deep layer is not required in this context, as explainable variances are captured in relatively simple features such as keywords.

We also experimented limiting reviews at different lengths when being extracted from the data. Figure 6 shows the evaluation metric using maximum word lengths as 50, 100, 200 and 300. Overall it seems that various word lengths have no major impact on the model performance. As a result, word lengths of 100 are used in most of our evaluations.

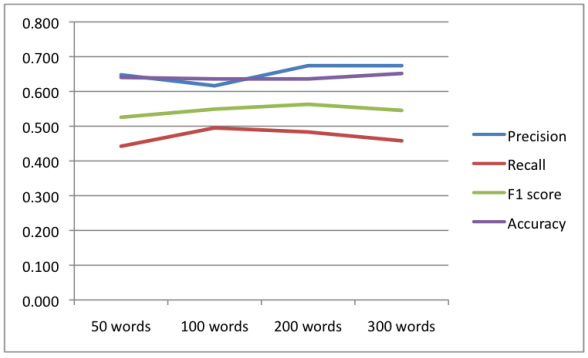

Figure 6: Performance metrics vs Review Input length

Since performance does not vary with different maximum lengths, it may suggest that users do not actually need to finish the entire review before deciding whether or not the review is helpful. We also note that in all of our models, we see significantly lower recall than precision, suggesting that many ”helpful” reviews are difficult to identify from ”unhelpful” ones.

5 Conclusion

From our experiment, it is suggested that helpfulness rating depends more on keywords and relevancy and surprisingly little on sentence structure and readability. In addition, it also suggests that many users may only glimpse through reviews and not reading its entirety, as various input lengths

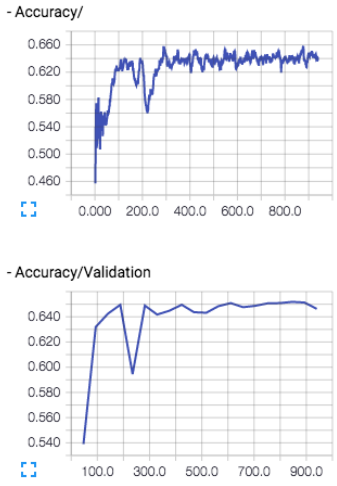

Figure 7: Accuracy vs Training Steps.

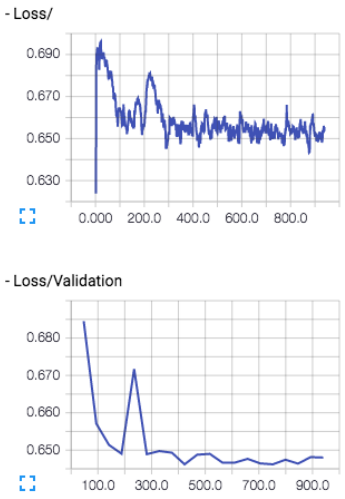

Figure 8: Loss vs Training Steps.

from the review have little effect on the performance. While preferential attachment and other biases have substantial influence, our experiment showed that review context still has impact on the perceived quality. For future works, it would be interesting to combine social network modeling technique and deep learning to understand the drivers for helpfulness ratings.

References

[1] Wan, Y., and Nakayama, M. (2012). Are Amazon.com Online Review Helpfulness Ratings Biased or Not? *Life: Web-Enabled Convergence of Commerce, Work, and Social Life* (M. J. Shaw, D. Zhang, and W. T. Yue. eds.), Springer Berlin Heidelberg, pp. 46?54.

[2] C. Danescu-Niculescu-Mizil & G. Kossinets. & J. Kleinberg & L. Lee (2009) How Opinions are Received by Online Communities: A Case Study on Amazon.com Helpfulness Votes, *Proceedings of the 18th international conference on World wide web*, pp. 141-150. New York, NY: Association for Computing Machinery.

[3] Hochreiter S. & Schmidhuber J. (1997) Long Short-Term Memory *Neural Computation Volume 9 Issue 8, November 15, 1997* pp. 1735-1780. Cambridge, MA: MIT Press

[4]Ha, Sangwook. (2015) Assessing Quality of Consumer Reviews in Mobile Application Markets: A Principal Component Analysis Approach. APA

[5] Kapoor, G.& Piramuthu, S. (2009) Sequential bias in online product reviews. *Journal of Organizational Computing and Electronic Commerce 19* pp.85-95

[6] Wang, C.& Zhang X. & Hann I. (2010) Social bias in online product ratings: a quasi-experimental analysis *WISE 2010* St Louis, MO

[7] Rodal, J. & Xiao M. & Longoria, S. (2014) Predicting Helpfulness Rating of Amazon Reviews

[8] Bhargava S. (2015) Predicting Amazon review helpfulness

[9] Stanford University CS224D: Deep Learning for Natural Language Processing, Lecture Materials, Equations copied from: http://cs224d.stanford.edu/lectures/CS224d-Lecture9.pdf