W266: Steam Sentiment Analysis

Dean Wang and Tomas Lobo

04/20/2020

Abstract

In recent years, reviews have become important as indicators of customer sentiment towards products they purchase. Steam is one of the most important platforms for the distribution of video games as well as hosting reviews of video games, having around 100M monthly active users.[9] This study attempts to model the sentiment in Steam reviews using baseline Naive Bayes and Support Vector Machine models as well as the recent Universal Sentence Encoder (USE) Deep Averaging Network model. Various amounts of preprocessing on the reviews text are done which shows that more preprocessing improves model performance. The recent USE model outperforms the baseline models. Finally, the USE hyperparameters with the greatest impact on model performance are the choice of activation function and optimizer used.

Introduction

As more and more commerce moves online, people are also leaving reviews of products they have purchased on websites. Oftentimes, these reviews are on the same platforms that offer products. It can be important for both buyers and sellers to pay attention to reviews. Buyers might look at the overall average rating (based on aggregated review information), whether most people who have purchased a product recommend it or not, as well as information in individual reviews to inform their decision on whether or not to buy a product. For sellers, it may be useful to understand which of their products are performing well in terms of customer satisfaction.

The video games market is an example where the influence of reviews can be felt. An important part of this market is the popular Steam platform. Millions of gamers use the platform every month.[9] Tens of thousands of games can be purchased on the platform.[1] Steam is not only a place where games can be purchased and played, but they are also discussed and reviewed on the site. After purchasing a game, a person can leave a review on the game as well as whether they would recommend the game or not. In this study, we try to use the text in the review to predict their sentiment toward the game they are reviewing, which is apparent based on whether or not they recommend the game.

Background

The field of sentiment analysis has existed for some time and various approaches have been used to accomplish this task. Since the early 2000s, researchers have been using machine learning methods to classify text in terms of the opinion the writer has towards a subject. Pang et al, 2002 used Naive Bayes, max entropy, and support vector machine (SVM) models to classify movie reviews by the sentiment expressed by the author. This was largely a bag-of-words approach, with some considerations for negation.[4]

Ten years later, neural networks were becoming popular across the machine learning field. One major development in sentiment classification was to use Convolutional Neural Nets (CNNs) for this task (Zhang et al, 2015). The neural net approach was able to improve on the baseline SVM and logistic regression models by several percentage points, though in general it was slower than some of the older linear models.[11]

Recently, several transfer learning approaches have been proposed and are growing in popularity. These models are pretrained on a variety of tasks; they can then be applied to tasks that are different than what they were originally trained on. An example of this (which is used in this study) is the Universal Sentence Encoder (USE) proposed by Cer et al, 2018.[2]

An earlier study that uses the Universal Sentence Encoder for the task of sentiment classification on games reviews data is Ruseti et al, 2019. One limitation of this study is that there was limited preprocessing done; the authors suggested using spelling autocorrection to clean up spelling mistakes found in reviews text in future work.[5]

Methods

Dataset

The dataset used in this study is a collection of over 6.4 million reviews retrieved from the Steam games platform run by Valve.[8] After a user purchases a game in the Steam store, they may leave reviews describing the game as well as whether or not they would recommend the game. The columns in the dataset itself are the review text, the ID of the game the review describes, whether the review was positive or negative, and the number of users who thought the review was helpful. Each entry in the dataset represents one review.

The dataset is somewhat imbalanced, with about 86% of entries being positive reviews and the remainder being negative. We used a language detection package to determine the language of reviews and could not identify non-English reviews; therefore, we concluded that almost all of the reviews in this dataset are in English. There were some entries with null reviews. These were left in the dataset after being converted to null strings.

There were several preprocessing tasks done to clean up the data before model training and evaluation. Any HTML tags inadvertently included in reviews text was removed. Accented characters were replaced with unaccented equivalents. Any contractions were expanded to the

non-contracted equivalents. Non-alphanumeric characters were removed, and misspelled words were autocorrected where possible.
For the majority of this study, a sample of 100,000 reviews was used for training and evaluation instead of the whole dataset. The reason for this is that the spelling autocorrection preprocessing step took a great deal of time and computation resources, with the 100,000 reviews taking 9-12 hours to process in this way. Three different versions of the same 100,000 reviews with different levels of preprocessing (unprocessed, preprocessed except for spelling autocorrection, and completely preprocessed) were generated to answer the question of how important preprocessing the data was in terms of performance.
The data were then split into training, development, and testing subsets, with about 60%, 20%, and 20% of original entries allocated to each subset, respectively. Training data were used for training models, development data were used to tune model hyperparameters, and testing data were used for final evaluation of model performance to prevent overfitting to evaluation data.

Models

There are three main models used in this study: Naive Bayes, Support Vector Machines, and the Deep Averaging Network implementation of the Universal Sentence Encoder. The Naive Bayes and Support Vector Machine models were used as the baselines. The Universal Sentence Encoder generates sentence embedding vectors based on reviews text, and then a fully-connected neural network predicts the class (sentiment) based on the embedding vectors.[2] Models were tuned and compared based on accuracy achieved on the development and test datasets, respectively.

Results and Discussion

1) **Universal Sentence Encoder**

To run this model, Tensorflow's DNN Clasifier tf.estimator.DNNClassifier was utilized. Tensorflow provides a variety of options when it comes to assembling the hidden layers of the network, testing different optimizers, activation functions, etc.

We iterated over several different model hyperparameter combinations in the search for the best possible accuracy. After training on the training dataset, different configurations of the model were compared in terms of performance on the development dataset. The different parameters that were fine tuned during this process include:

- Hidden units: number of layers (from 2 to 4) and size
- Learning rates: 0.001, 0.003, 0.005
- Number of execution steps: 5,000 and 10,000
- Optimizer: Adagrad, Adam, Adadelta, Adamax, SGD, RMSprop, Nadam
- Activation Function: Relu, Sigmoid, Selu, Softmax, Tanh, Elu
- Batch Normalization: True, False
- Dropout: None, 0.05, 0.1, 0.2, 0.6

A custom programmatic iterator made it possible to test several scenarios by playing around with these parameters.

Additionally, during the exercise, three different versions of the same sample of 100,000 elements of the dataset were utilized, to determine how helpful the processing steps described earlier were to improve the efficiency of the algorithm:

- Unprocessed data
- Pre-processed data (eg. remove accents and special characters, expand contractions, apply lowercase, etc) **without** spell checking
- Pre-processed data **with** spell checking

These were the results obtained after testing 25 different scenarios over the three variations of the dataset. Development and test splits had a very similar performance.

Test N	Hidden Units	Learning Rate	Optimizer	Activation Function	Batch Normalization	Dropout	Unprocessed - Dev	Processed w/o spelling - Dev	Processed w/ spelling - Dev
#1	500, 100	0.003	Adagrad	Relu	None	None	83.93%	82.98%	85.72%
#2	1000, 100	0.003	Adagrad	Relu	None	None	83.75%	85.14%	85.71%
#3	1000, 100	0.005	Adagrad	Relu	None	None	84.00%	85.49%	85.99%
#4	1000, 100	0.001	Adagrad	Relu	None	None	84.29%	85.43%	85.85%
#5	20000, 100	0.001	Adagrad	Relu	None	None	84.00%	85.82%	86.08%
#6	20000, 100	0.005	Adagrad	Relu	None	None	83.27%	84.48%	85.73%
#7	1000, 100	0.005	Adadelta	Relu	None	None	84.06%	85.47%	86.02%
#8	1000, 100	0.005	Adam	Relu	None	None	82.69%	84.69%	84.66%
#9	1000, 100	0.005	Adamax	Relu	None	None	72.20%	73.27%	82.32%
#10	1000, 100	0.005	SGD	Relu	None	None	84.17%	85.69%	85.96%
#11	1000, 100	0.005	RMSprop	Relu	None	None	83.28%	83.98%	85.49%
#12	1000, 100	0.005	Nadam	Relu	None	None	83.19%	84.29%	84.93%
#13	1000, 100	0.005	Adagrad	Sigmoid	None	None	84.32%	85.49%	85.85%
#14	1000, 100	0.005	Adagrad	Selu	None	None	84.56%	85.72%	86.42%
#15	1000, 100	0.005	Adagrad	Softmax	None	None	82.18%	82.13%	82.46%
#16	1000, 100	0.005	Adagrad	Tanh	None	None	84.14%	85.52%	85.98%
#17	1000, 100	0.005	Adagrad	Elu	None	None	84.65%	85.93%	86.40%
#18	1000, 100	0.005	Adagrad	Relu	False	None	82.83%	85.33%	85.82%
#19	1000, 100	0.005	Adagrad	Relu	True	None	83.68%	85.36%	85.72%
#20	1000, 100	0.005	Adagrad	Relu	True	0.2	84.63%	86.03%	86.47%
#21	1000, 100	0.005	Adagrad	Relu	True	0.6	84.35%	85.58%	85.98%
#22	1000, 100	0.005	Adagrad	Relu	True	0.1	84.69%	85.91%	86.32%
#23	1000, 100	0.005	Adagrad	Relu	True	0.05	84.51%	85.78%	86.25%
#24	1000, 100, 100	0.005	Adagrad	Relu	True	None	83.06%	84.94%	85.50%
#25	1000, 100, 100, 100	0.005	Adagrad	Relu	True	None	82.78%	84.67%	84.80%

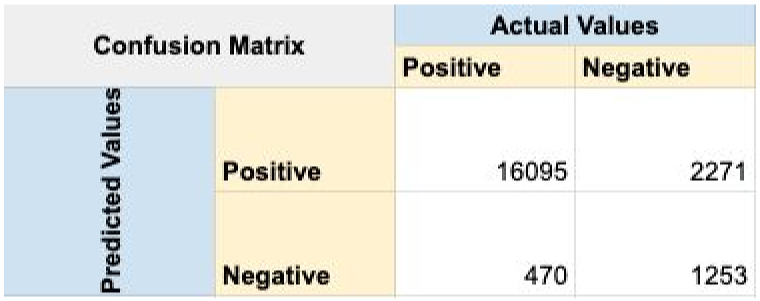
As it can be derived from the table, preprocessing the dataset had a positive impact on the accuracy of the different scenarios tested. Adding the various preprocessing steps (eg. dealing with special characters, contractions and others), increased the accuracy of each scenario by 1.3% in average. Finally, adding a last spell checking step increased the average accuracy by 0.9%.

The scenario with the best result was #20, with 2 layers, Adagrad optimizer, Relu activation function, batch normalization and a dropout of 0.1. After pre-processing and spell checking the dataset, this combination achieved an accuracy of 86.47%. Additionally, the precision in this case was 88.9% and the recall was 98.1%.

Finally, it is worth mentioning that some parameters had a bigger impact on the results that others:

- Hidden units: Increasing the number of layers from 2 to 3 and 4 did not seem to have a positive impact on accuracy. Neither did increasing the size of the first layer.
- Optimizer: This parameter had a big impact on results. Adagrad, Adadelta and SGD proved to be the top performing optimizers, more than 4 points over Adamax, the worst performing.
- Activation function: This parameter was relatively sensitive to results. Most variations had a similar performance, except by Softmax, that averaged 4 points less than the rest.
- Batch normalization: This parameter did not change results much in general.
- Dropout: A dropout of 0.2 had ~0.3-0.5% increase in accuracy compared to other parameters or the absence of the parameter.

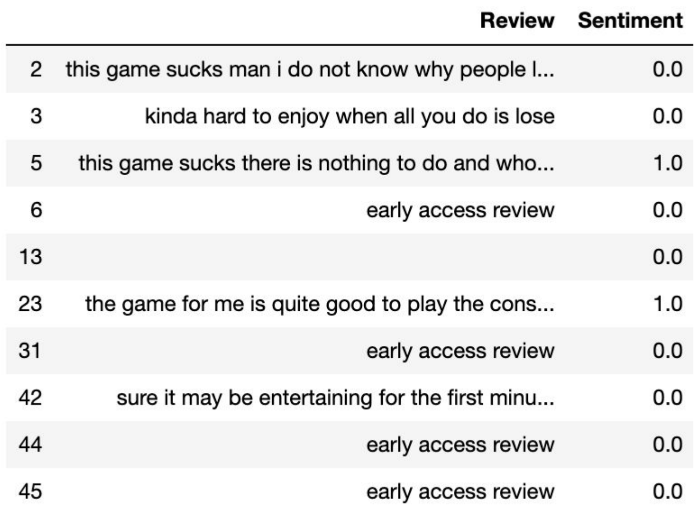
An error analysis was conducted on the best performing version of the algorithm:



Confusion Matrix		Actual Values	
		Positive	Negative
Predicted Values	Positive	16095	2271
	Negative	470	1253

As we can see, there were 1,253 (6.3%) true negatives and 16,095 (80.0%) true positives. Also, there were 2,271 (11.3%) false positives and 470 (2.4%) false negatives.

Some examples of misclassified reviews were:



	Review	Sentiment
2	this game sucks man i do not know why people l...	0.0
3	kinda hard to enjoy when all you do is lose	0.0
5	this game sucks there is nothing to do and who...	1.0
6	early access review	0.0
13		0.0
23	the game for me is quite good to play the cons...	1.0
31	early access review	0.0
42	sure it may be entertaining for the first minu...	0.0
44	early access review	0.0
45	early access review	0.0

To avoid misclassifications, there is some further clean up that we could do. For example, removing all the “early access reviews” reviews, that could be confusing the algorithm.

2) Naive Bayes:

To run this model, the GaussianNB function from sklearn was utilized. All three versions of the dataset were tested: unprocessed, processed without spell checking, and processed with spell checking.

Surprisingly, the best results in terms of accuracy were obtained in the unprocessed dataset, with a slight (negligible) increase compared to the other two versions. This is how the results looked like:

Naive Bayes	**Unprocessed**	**Processed w/o spelling**	**Processed w/ spelling**
Accuracy	74.9%	74.7%	74.4%
Precision	92.6%	92.6%	92.5%
Recall	75.5%	75.3%	75.1%
F1	83.2%	83.0%	82.9%

3) Support Vector Machines:

To run this model, the SVC function from sklearn was utilized. All three versions of the dataset were tested: unprocessed, processed without spell checking, and processed with spell checking.

The best results were obtained with the fully processed dataset, with an accuracy of 82.5%. This was still 4 points below what was accomplished with the Universal Sentence Encoder method, although about 8 points over Naive Bayes. These were the results obtained:

SVM	**Unprocessed**	**Processed w/o spelling**	**Processed w/ spelling**
Accuracy	82.2%	82.1%	82.5%
Precision	82.2%	82.1%	82.5%
Recall	100.0%	100.0%	100.0%
F1	90.2%	90.2%	90.4%

Conclusion

In our study, we used machine learning methods to predict the sentiment expressed in reviews of games on the Steam games platform. Some of the key learnings from our work were that cleaning up the reviews text by preprocessing improved performance for our main model; the most important hyperparameters to tune were the choice of activation function and optimizer; and the recently developed Universal Sentence Encoder outperformed the older baseline Naive Bayes and support vector machine models.

One of the challenges we encountered when working on our study was that one of the key steps in preprocessing, the spelling autocorrection of reviews text, took a long time and used computational resources intensely. This limited us to using a subset of 100,000 entries out of the millions we actually had available to ensure our study could be completed in a short enough amount of time. A future study could use parallelization to preprocess the entire available corpus and potentially improve model performance with more training data.

Another area that could be investigated in the future is a more nuanced classification of sentiment. In the current study, the sentiment classes predicted were only negative or positive, with no degrees of sentiment (for example neutral or slightly positive/negative sentiment in contrast to fully positive/negative sentiment). A potential future study could use manual labor to label training data with degrees of sentiment expressed, which could then be used to train models to predict degrees of sentiment.

Another research area that could be of particular interest to games companies is understanding the reasons behind positive or negative reviews. In our current study, reviews for games were only classified as positive or negative with no attention given to why a reviewer rated a game that way; however, reviews could be classified by topic and then topics could be used in combination with sentiment polarity to find general feedback trends for a given game.

Finally, recent research has resulted in newer transfer learning models, including pre trained transformer embeddings. These new models can potentially be used for the task of sentiment analysis as well. A future study could fit these models to compare against our study's baselines and main model.

References

[1] Calvin, A. (2020, January 2). Over 8,000 games were released on Steam in 2019, according to SteamSpy. Retrieved April 19, 2020, from https://www.pcgamesinsider.biz/news/70259/over-8000-games-were-released-on-steam-in-2019-according-to-steamspy/

[2] Cer, D., Yang, Y., Kong, S.-Y., Hua, N., Limtiaco, N., John, R. S., … Kurzweil, R. (2018). Universal Sentence Encoder for English. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. doi: 10.18653/v1/d18-2029

[3] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). doi: 10.3115/v1/d14-1181

[4] Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment Classification using Machine Learning Techniques. Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - EMNLP 02. doi: 10.3115/1118693.1118704

[5] Ruseti, S., Sirbu, M.-D., Calin, M. A., Dascalu, M., Trausan-Matu, S., & Militaru, G. (2019). Comprehensive Exploration of Game Reviews Extraction and Opinion Mining Using NLP Techniques. Advances in Intelligent Systems and Computing Fourth International Congress on Information and Communication Technology, 323–331. doi: 10.1007/978-981-15-0637-6_27

[6] Secui, A., Sirbu, M.-D., Dascalu, M., Crossley, S., Ruseti, S., & Trausan-Matu, S. (2016). Expressing Sentiments in Game Reviews. Artificial Intelligence: Methodology, Systems, and Applications Lecture Notes in Computer Science, 352–355. doi: 10.1007/978-3-319-44748-3_35

[7] Sirbu, D., Secui, A., Dascalu, M., Crossley, S. A., Ruseti, S., & Trausan-Matu, S. (2016). Extracting Gamers Opinions from Reviews. 2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC). doi: 10.1109/synasc.2016.044

[8] Sobkowicz, A. (2017, October 2). Steam Review Dataset (2017). Retrieved from https://zenodo.org/record/1000885#.XmJPM5NKhPM

[9] Soper, T. (2017, August 3). Valve reveals Steam's monthly active user count and game sales by region. Retrieved April 19, 2020, from https://www.geekwire.com/2017/valve-reveals-steams-monthly-active-user-count-game-sales-region/

[10] Zagal, J. P., Tomuro, N., & Shepitsen, A. (2011). Natural Language Processing in Game Studies Research. Simulation & Gaming, 43(3), 356–373. doi: 10.1177/1046878111422560

[11] Zhang, Y., & Wallace, B. (2015). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. Retrieved from https://arxiv.org/abs/1510.03820