A Comparative Analysis of Predicting the Sentiment of Steam Game Reviews Using Deep Neural Networks

Daniel Lee
School of Computing
Queen's University
Kingston, Canada
18dil@queensu.ca

I. Introduction

The video game market is growing faster than ever with a projected revenue of approximately $180 billion in 2021. As the game market continues to evolve, user requirements rapidly increase, making game development more difficult. Game development is becoming more complex and requires a larger team to satisfy the increasing needs of gamers. Thus, it is important for game developers to know how gamers really feel about their games, so game developers can dig deeper into the problems.

One way that game developers can see how gamers feel about their game is through game reviews. Game reviews are an indicator of a gamer's reception towards the game. Prior work [18] has studied the sentiment of end-users from mobile app reviews. However, game development is vastly different from traditional software development, which could mean that game reviews are likely to be different from traditional mobile app reviews [21].

Furthermore, game reviews may be more susceptible to sarcasm due to the sarcastic culture that Steam is built upon [1]. Sarcasm is known to make it difficult for sentiment analysis techniques that use words of the text as indicators of the sentiment [7]. As a result, traditional techniques in sentiment analysis are not always accurate, due to the lack of context, sentiment ambiguity, identifying sarcasm, comparatives, regional differences and degree of agreement between humans and machines [5]

We believe that traditional sentiment analysis techniques are not well-suited for game reviews. Hence, we want to propose that game developers should use a deep learning approach to better predict the sentiment of gamers. We want to mitigate false-positive results caused by sarcasm as much as possible by learning abstractions of features from the reviews. In short, our intuition is that deep neural networks can learn these complex problems much better than traditional techniques. Although, there has been some prior work [1], [30] that utilized traditional sentiment analysis or machine learning techniques to classify game reviews, we purely use a deep learning approach.

In our paper, we use game reviews from Steam, one of the largest online digital distribution platforms to perform sentiment analysis using a deep learning approach [16]. In addition, we compare our results to an existing out-of-box sentiment analysis technique. We conjecture that our deep learning approach out-performs the traditional sentiment analysis technique, with respect to accuracy and AUC. Ultimately, we want to provide game developers a better approach in determining how gamers truly feel about their games, so they can dive deeper into understanding the reasons of why people dislike it.

II. Steam Game Reviews

Steam was developed by Valve Corporation and is one of the largest online digital platforms available for PC games [16]. Steam provides users, who have played the game, the option to leave a *Steam review*[1] for the game. In addition, the user can label the Steam game review as "Recommended" or "Not Recommended" (i.e., positive or negative review'). The Steam Community utilizes the"Recommended" and "Not Recommended" ratings to understand the gamer's overall feelings about a game. To the best of our knowledge, there is limited research available on text classification of Steam game reviews.

III. Related Work

A. Sentiment Analysis of Users from Video Game and Mobile App Reviews

Some prior works have used sentiment analysis techniques on game reviews to classify the gamer's review.

Stråat and Vergagen [27] studied the game reviews from *Metacritic.com* to propose an aspect based sentiment analysis to elicit and evaluate user opinions on prior released games. They found that the sentiment of an aspect in a game review impacts its rating. Bais et al. [1] used traditional machine learning approaches (i.e., SVM, Logistic Regression, Multinomial Naive Bayes, Turney's unsupervised phrase-labeling algorithm, and a lexicon-based baseline) to classify Steam

[1]https://store.steampowered.com/reviews/

reviews as positive or negative, finding that SVM with TF-IDF out-performed the others with respect to accuracy, precision, and F1-score after adding hours played and proper weighting of words. Kiran et al. [10] classified positive and negative tweets on the latest review of games using machine learning approaches, and found that maximum entropy has a higher accuracy, precision, and recall than Bayes and SVM. Sirbu et al [26] studied 9,500 game reviews from Amazon and used Principal Component Analysis (PCI) to find five components that could classify game reviews as positive, negative or neutral with a 55% accuracy. Zuo [30] used Steam game reviews with additional features to train Naive Bayes and Decision Tree classifiers for sentiment analysis on the reviews, finding that the Decision Tree classifier outperformed Naive Bayes with approximately a 75% accuracy.

Some prior studies have used sentiment analysis techniques on mobile app reviews to understand how the user feels about the app or app features.

Fu et al. [6] studied over 13 million mobile app reviews from the Google Play Store and trained a LDA (i.e. latent dirichlet allocation) model using the negative reviews (1-star or 2-star ratings), and then extracted the topics to observe what users disliked. Liang et al. [12] used a multifacet sentiment analysis approach on 79 paid and 70 free apps from the iOS app store to explore the relationship between mobile and reviews and the sales, finding that app reviews have a significant influence on the mobile app sales ranking. Sangani and Ananthanarayanan [24] studied the sentiment of mobile app reviews from the iOS App Store by creating topics of interest using WordNet, then ranking the reviews relevant to a topic. They found a number of topics that developers can use to interpret a review, an average rating that displays the general sentiment of a topic, and a representative sample of reviews that provide criticism for a topic. Guzman and Maalej [8] use the NLTK toolkit to extract features from mobile app reviews, then use SentiStrength to assign a sentiment score to the features, and then identified the topics of the reviews. In addition, they calculated the topic sentiment. They proposed an approach to elicit features form mobile app reviews with their corresponding sentiments, which generates two summaries of different scopes that can developers quantify user's feeling about a feature.

In our research, we propose a deep learning approach using a gated recurrent network to predict if a Steam game review is positive or negative (i.e., recommended or not recommended) based on the text.

B. Games Development From a Software Engineering Perspective

A few prior studies have focused on providing insights for game developers from a software engineering perspective.

Pascarella et al. [21] studied 60 open source projects to make a comparison between open source game development and traditional open source software development, finding that they are different in various aspects. Politowski et al. [22] studied the development of 20 games, finding that at least 30% of the games still applied a waterfall development paradigm, and 65% of the games applied iterative practices. Hill et al. [20] studied 364 survey respondents and 14 interviewees, finding that game developers often avoided automated testing, due to the lack of creativity involved.

In our research, we want to show that help game developers from a software engineering perspective by providing game developers the approach to understand gamers' needs, so they can improve their games.

C. Mining Online Game Distribution Platforms

Several prior studies on mining online distribution platforms focus on Steam.

Sifa et al. [25] studied the playtime of 6 million users on Steam to discover the fundamental principles of playtime. Lin et al. [13] studied the urgent updates of Steam to observe that developers avoid certain update cycles based on the update pattern. Lin et al. [14] studied the early access model on the Steam, observing that game developers use the early access model to gather early criticisms and positive feedback from newcomers. Lin et al. [15] empirically studied game reviews from Steam, finding that game reviews are different from mobile app reviews. Poretski and Arazy [23] empirically studied 45 games from Nexus Mods, finding that user-made game modifications can impact the increase in sales of an original game. Blackburn et al. [2] studied cheaters in the Steam community, observing that the player's network of friends impacted the likelihood of a player becoming a cheater.

In our study, we focus on mining the Steam platform to provide an approach that can use sentiment analysis of game reviews to support game development.

IV. Model

In our paper, we will be using a recurrent neural network as it is primarily used for sequential data. In addition, recurrent neural networks have an internal memory state that is capable of remembering prior time steps when propagating through the network.

To input the game reviews into our recurrent neural network we first create word embeddings from our game reviews. Word embeddings are contextual vector representations of the words in our game reviews. We pre-train a Word2vec model to generate word embeddings for our network. Word2vec utilizes the skip-gram or common bag of words (CBOW) models to create word embeddings. The Skip-gram model produces a probability distribution of the contexts position for each word in the game review [19]. The CBOW model attempts to predict the word based on the context of the word [17]. We use the Word2vec algorithm provided by the *Gensim library*[2]. Afterwards, the word embeddings are used in the embedding layer of the neural network. Since, the Word2vec model is pre-trained we do not have to train the embedding layer in each epoch, which saves us some computation time.

[2]https://radimrehurek.com/gensim/models/word2vec.html

We use gated recurrent units for our Recurrent neural network. Gated recurrent neural networks are able to outperform traditional recurrent neural networks in sentiment classification, and perform comparable to long-short term memory networks (i.e., LSTM), while being more computationally efficient [3], [28].

Originally, recurrent neural networks have a trouble time remembering long time series data, which can lead to the vanishing gradient problem. Gated recurrent units (GRU) solve some of the disadvantages of standard recurrent neural networks, such as the vanishing gradient problem with an update and reset gate. The update gate determines how much information from the past propagates to the future and the forget gate determines how much of the past information is forgotten [4], [11]. If needed, we will use dropout regularization to prevent overfitting of our data. Dropout regularization assigns a probability on the nodes of a layer, and deactivates the nodes that are exceeding the probability threshold, which allows for more generalization of smaller subsets of information through fewer nodes.

The output of our network is a binary classification ("Recommended" or "Not Recommended") of the user sentiment. Thus, we add a final dense layer (i.e., fully connected), with a sigmoid activation function. We use the sigmoid activation function to squeeze the outputs of the gated recurrent unit layer into a value ranging from 0 to 1 because we are performing binary classification. In addition, we use the binary cross-entropy loss function with the Adam optimizer. We use the binary cross-entropy loss function because our task is binary classification. We use the Adam (i.e., adaptive moment estimation) optimizer because it is known to be computationally efficient and uses an adaptive learning rate [9].

Furthermore, we wrap the entire above-mentioned process in a stratified K-fold cross validation to evaluate the model's performance with the data. We calculate the mean accuracy of all the folds. Due to the time constraint, we only use 5 folds.

V. Experiment

In this paper, we use game reviews as the primary dataset. We have crawled 91,321 Steam game reviews that are labelled as 1 or 0 (i.e., "Recommended" or "Not Recommended"). We set aside a random sample of 60% of the data as the training set, 20% as the validation set, and the remaining 20% as the test set. To prepare the game reviews for the Word2vec model we pre-process them with following:

1) Tokenization of text
2) Punctuation removal
3) Hyperlink removal
4) Symbol and expression removal
5) Stop-word removal
6) Lowercase text
7) Non-english text removal

After pre-processing, we convert the game review tokens into sequences. We then pad the sequences with a number of zeros until the length of the sequence vector equals the maximum sentence length across the game reviews. To build and train the model with this data we use the *Keras library*[3]

Also, we notice a class imbalance in our game review dataset. To resolve this, we use our labels to compute a dictionary of class weights with the *Sci-kit learn library*[4]. Class weights emphasize the minority class more and and the majority class less.

For the hyper-parameters of the gated recurrent network, we use an embedding dimension of 128 for the embedding layer, and we start with a probability of 0.2 in the gated recurrent unit layer for dropout regularization. If needed, we increase the probability of dropout. Although, the Adam optimizer does not require an initial learning rate, we set the learning rate at 0.0001 because prior work has shown significant improvements in performance when using an initial learning rate [29].

Moreover, due to the time constraint, we run only 25 epochs per fold. Also, we use early stopping with a patience of 10 to speed up the training of the model. Furthermore, we compare the two approaches: our deep learning approach, and an existing out-of-box sentiment analysis technique such as *SentiStrength*[5], with respect to the accuracy and AUC.

References

[1] R. Bais, P. Odek, and S. Ou, "Sentiment classification on steam reviews."

[2] J. Blackburn, N. Kourtellis, J. Skvoretz, M. Ripeanu, and A. Iamnitchi, "Cheating in online games: A social network perspective," *ACM Transactions on Internet Technology (TOIT)*, vol. 13, no. 3, p. 9, 2014.

[3] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[4] ——, "Gated feedback recurrent neural networks," in *International Conference on Machine Learning*, 2015, pp. 2067–2075.

[5] B. Donkor, "Sentiment Analysis: Why It's Never 100% Accurate," https://brnrd.me/posts/sentiment-analysis-never-accurate, 2014, (last visited:February 11, 2019).

[6] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, and N. Sadeh, "Why people hate your app: Making sense of user feedback in a mobile app store," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 1276–1284.

[7] G. Ganu, N. Elhadad, and A. Marian, "Beyond the stars: improving rating predictions using review text content." in *WebDB*, vol. 9. Citeseer, 2009, pp. 1–6.

[8] E. Guzman and W. Maalej, "How do users like this feature? a fine grained sentiment analysis of app reviews," in *IEEE 22nd international requirements engineering conference (RE)*. IEEE, 2014, pp. 153–162.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[10] T. Kiran, K. G. Reddy, and J. Gopal, "Twitter sentiment analysis of game reviews using machine learning techniques."

[11] S. Kostadinov, "Understanding GRU Networks," https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be, 2017, (last visited:March 22, 2019).

[12] T.-P. Liang, X. Li, C.-T. Yang, and M. Wang, "What in consumer reviews affects the sales of mobile apps: A multifacet sentiment analysis approach," *International Journal of Electronic Commerce*, vol. 20, no. 2, pp. 236–260, 2015.

[13] D. Lin, C.-P. Bezemer, and A. E. Hassan, "Studying the urgent updates of popular games on the Steam platform," *Empirical Software Engineering*, vol. 22, no. 4, pp. 2095–2126, 2017.

[3] https://keras.io/models/model/

[4] https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html

[5] http://sentistrength.wlv.ac.uk/

[14] ——, "An empirical study of early access games on the Steam platform," *Empirical Software Engineering*, vol. 23, no. 2, pp. 771–799, 2018.
[15] D. Lin, C.-P. Bezemer, Y. Zou, and A. E. Hassan, "An empirical study of game reviews on the Steam platform," *Empirical Software Engineering*, pp. 1–38, 2018.
[16] ——, "An empirical study of game reviews on the steam platform," *Empirical Software Engineering*, vol. 24, no. 1, pp. 170–207, 2019.
[17] W. Ling, C. Dyer, A. W. Black, and I. Trancoso, "Two/too simple adaptations of word2vec for syntax problems," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1299–1304.
[18] I. Malavolta, S. Ruberto, T. Soru, and V. Terragni, "End users' perception of hybrid mobile apps in the google play store," in *2015 IEEE International Conference on Mobile Services*. IEEE, 2015, pp. 25–32.
[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[20] E. Murphy-Hill, T. Zimmermann, and N. Nagappan, "Cowboys, ankle sprains, and keepers of quality: How is video game development different from software development?" in *Proceedings of 36th International Conference on Software Engineering*. ACM, 2014, pp. 1–11.
[21] L. Pascarella, F. Palomba, M. Di Penta, and A. Bacchelli, "How is video game development different from software development in open source?" in *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*. IEEE, 2018, pp. 392–402.
[22] C. Politowski, L. Fontoura, F. Petrillo, and Y.-G. Guéhéneuc, "Are the old days gone?: A survey on actual software engineering processes in video game industry," in *Proceedings of 5th International Workshop on Games and Software Engineering*. ACM, 2016, pp. 22–28.
[23] L. Poretski and O. Arazy, "Placing value on community co-creations: A study of a video game'modding'community," in *Proceedings of ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, 2017, pp. 480–491.
[24] C. Sangani and S. Ananthanarayanan, "Sentiment analysis of app store reviews," *Methodology*, vol. 4, no. 1, pp. 153–162, 2013.
[25] R. Sifa, C. Bauckhage, and A. Drachen, "The playtime principle: Large-scale cross-games interest modeling." in *Proceedings of CIG*, 2014, pp. 1–8.
[26] D. Sirbu, A. Secui, M. Dascalu, S. A. Crossley, S. Ruseti, and S. Trausan-Matu, "Extracting gamers' opinions from reviews," in *18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. IEEE, 2016, pp. 227–232.
[27] B. Strååt and H. Verhagen, "Using user created game reviews for sentiment analysis: A method for researching user attitudes." in *GHITALY@ CHItaly*, 2017.
[28] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1422–1432.
[29] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4148–4158.
[30] Z. Zuo, "Sentiment analysis of steam review datasets using naive bayes and decision tree classifier," 2018.