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

***Abstract*—The gaming industry continues to grow at an alarming rate. Game developers are under increasing pressure to meet the needs of gamers, but do not always knows exactly what gamers want. Game reviews are a method that gamers use to convey their sentiment about the game. For example, negative game reviews can help game developers understand the underlying problems of the game. However, identifying negative reviews can be time consuming as gamers tend to use sarcasm.**

In this paper, we propose a deep learning solution to automatically classify game reviews as either positive or negative. We use 53,764 game reviews from 3,625 Steam games that are labelled from the Steam platform to train a gated recurrent neural network model. We evaluate the performance of the model and compare it against two Naive Bayes classifiers with different resampling strategies; SMOTE and Tomek link, and under-sampling. We find that the deep learning model out-performs the Naive Bayes classifier using SMOTE and Tomek link in all performance metrics, but under-performs compared to the Naive Bayes classifier using under-sampling, with respect to the test accuracy and AUC. However, we do observe that the deep learning model could potentially be improved with more features, tuning, and epochs.

***Keywords*- Steam; Game reviews; Gated recurrent neural networks; classifier**

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 [20] 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 [23].

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 [8]. 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 [6]

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], [32] 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 the Steam platform, one of the largest online digital distribution platforms to perform sentiment analysis using a deep learning approach [17]. In addition, we compare our results to an existing sentiment analysis technique. We conjecture that our deep learning approach will out-perform the sentiment analysis technique, with respect to accuracy and AUC. We want to provide game developers a better approach in classifying game reviews, so they can quickly dive deeper into understanding the gamers needs.

II. 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 [29] 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 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. [11] 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 [28] 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 [32] 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. [7] 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. [13] 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 [26] 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 [9] 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 study, we propose a deep learning approach using a gated recurrent neural network to predict if a Steam game review is positive or negative (i.e., recommended or not recommended) based on the text of the game review.

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. [23] 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. [24] 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. [22] studied 364 survey respondents and 14 interviewees, finding that game developers often avoided automated testing, due to the lack of creativity involved.

In our study, we want to help game developers by providing them an accurate approach in classifying game reviews, so they do not waste time on noisy game reviews.

C. Mining Online Game Distribution Platforms

Several prior studies on mining online distribution platforms focus on Steam. Steam was developed by Valve Corporation and is one of the largest online digital platforms available for PC games [17]. 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.

Sifa et al. [27] studied the playtime of 6 million users on Steam to discover the fundamental principles of playtime. Lin et al. [14] studied the urgent updates of Steam to observe that developers avoid certain update cycles based on the update pattern. Lin et al. [15] 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. [16] empirically studied game reviews from Steam, finding that game reviews are different from mobile app reviews. Poretski and Arazy [25] 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. [3] 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 use sentiment analysis on the game reviews to provide support to game development.

III. Model

In our paper, we will be using a gated recurrent neural network for our deep learning model as it is primarily used for sequential data. Recurrent neural networks have an internal memory state that is capable of remembering prior time steps when propagating through the network. Gated recurrent neural networks are able to out-perform traditional recurrent neural networks in sentiment classification, and perform comparable

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

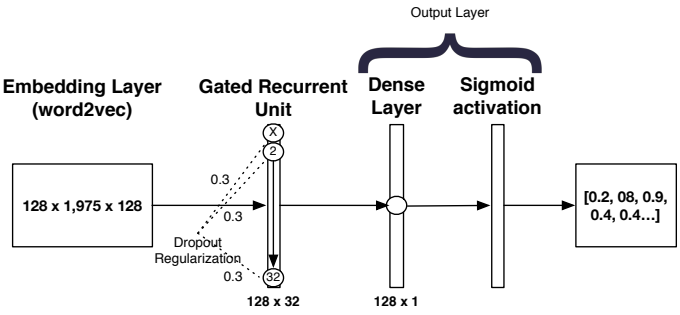

Fig. 1. The architecture of our gated recurrent neural network.

to long-short term memory networks (i.e., LSTM), while being more computationally efficient [4], [30]. Figure 1 shows an overview of our deep learning network architecture.

Originally, recurrent neural networks had a trouble time remembering long time series data, which led 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 [5], [12].

In the gated recurrent unit, we use dropout and recurrent 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 due to the fewer nodes.

To input the Steam game reviews into our recurrent neural network we first develop a Word2vec model to generate word embeddings. Word embeddings provide contextual vector representations of the words in our Steam game reviews. We pre-train our own Word2vec model using the Steam game reviews. 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 [21]. The CBOW model attempts to predict the word based on the context of the word [19]. 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.

The output of our network is a binary classification ("Recommended" or "Not Recommended") of the gamer's 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 [10].

In addition, we wrapped the entire above-mentioned process in a stratified and shuffled K-fold cross validation to evaluate the model's performance with the data. Due to the time constraint, we only used 5 folds for cross validation. We computed the mean and median accuracy for the training, validation, and test set, along with the mean and median AUC (i.e., area under the curve) for the test set across the folds. Accuracy is defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

TP is the true positive classes; TN is the true negative classes; FP is the false positive classes; FN is the false negative classes. Accuracy can be misleading due to the inherent sensitivity on a given threshold for correctly identified classes. Thus, we also calculated the AUC (i.e area under the curve) because it considers all possible thresholds, whereas the accuracy does not [18].

Furthermore, we also built two Naive Bayes classifiers to predict the sentiment of Steam game reviews, and used them as baselines in our study. Since, our dataset has a class imbalance, we used two different resampling strategies. Prior work [2] found that *SMOTE (i.e., synthetic minority oversampling technique) with Tomek links* out-performed *SMOTE over-sampling* with a 91.6% AUC. Hence, to account for the imbalanced classes in our Steam game reviews, we built one classifier using *SMOTE and Tomek links*, and another classifier using under-sampling.

IV. Experiment

In this paper, we used Steam game reviews as the primary dataset. Figure 2 shows an overview of our experiment. We collected 91,321 Steam game reviews from 4,080 Steam games that are labelled as a 1 or 0 (i.e., "*Recommended*" or "*Not Recommended*"). First, we removed non-English game reviews from our dataset as it could bias our result. We were left with 53,764 Steam game reviews from 3,625 games, which we used for the remainder of our study. Then, we shuffled our dataset and set aside a random sample of 60% of the dataset as the training set, 20% as the validation set, and the remaining 20% as the unseen test set. To prepare the Steam game reviews for the Word2vec model we pre-processed them with the following steps:

1) Lowercase text
2) Remove new lines
3) Remove trailing white spaces
4) Tokenization of text
5) Expand contractions

Initially, we used stop-word and punctuation removal in our pre-processing steps, but found that our accuracies dropped.

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

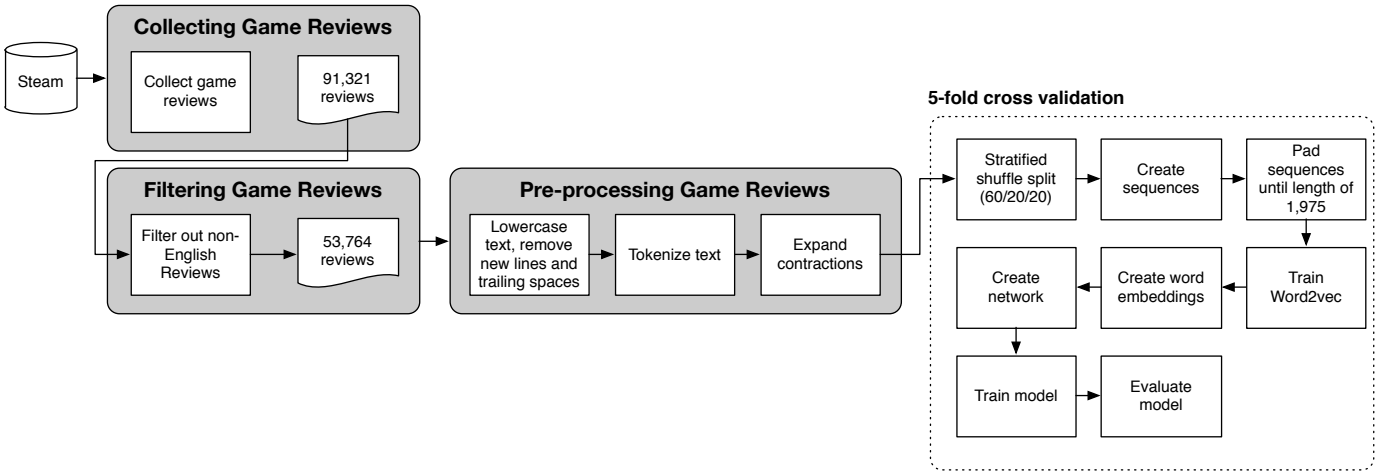

Fig. 2. An overview of our deep learning methodology.

A possible explanation for the drop in accuracies is that stop-words and punctuations emphasize certain parts of the review, which actually help the network learn.

Afterwards, we converted the Steam game review tokens into sequences. We padded the sequences with a number of zeros based on the maximum review length (1,975) across the Steam game reviews. We padded the sequences because the embedding layer in the Keras model requires a fixed input length. Thus, to build and train the deep learning model we used the *Keras library*[3]. The padded sequences of the training and validation sets were used to train the Word2vec model. The Word2vec model was re-trained on each fold.

Furthermore, we noticed a large class imbalance in our Steam game review dataset, where there were almost three times more recommended reviews (1) than not recommended reviews (0). To resolve the class imbalance, we used our class labels to compute a dictionary of class weights with the *Scikit learn library*[4]. Hence, the class weights emphasized the minority class over the majority class. We also tried over-sampling the minority class, but found our deep learning model to perform poorly.

For the hyper-parameters of the gated recurrent network, we used an embedding dimension of 128 for the embedding layer. We tried embedding dimensions of 64 and 100, but found that 128 out-performed them. We started with a dropout regularization probability threshold of 0.2 in the gated recurrent unit layer. Eventually, we ended up using a probability threshold of 0.3 because it performed better than the probability thresholds of 0.2 and 0.4.

Although the Adam optimizer does not require an initial learning rate, we tried the learning rates 0.01, 0.001 and 0.0001 because prior work has shown significant improvements in performance when using an initial learning rate [31]. We decided to use 0.001, as it out-performed the others. For the dense layer, we tried 32 and 64 nodes respectively, but found that 32 nodes out-performed the others

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

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

TABLE I
AN OVERVIEW OF THE RESULTS OF OUR DEEP LEARNING MODEL AND TWO NAIVE BAYES CLASSIFIERS.

Performance metrics	Gated recurrent neural network model	Naive Bayes classifier with SMOTE and Tomek link	Naive Bayes classifier with under-sampling
Mean training accuracy	0.84	0.47	0.68
Median training accuracy	0.85	0.47	0.68
Mean validation accuracy	0.85	-	-
Median validation accuracy	0.85	-	-
Mean test accuracy	0.60	0.46	0.65
Median test accuracy	0.62	0.46	0.65
Mean test AUC	0.59	0.51	0.65
Median test AUC	0.60	0.51	0.65

Due to the long runtime of training our network (roughly 650 seconds per epoch), we decided to only run 20 epochs per fold. We also used early stopping with a patience value of 5 to speed up the training of the model. However, we did try patience value of 8 and 10, but did not find a significant difference in prior deep learning models.

Lastly, we compared our deep learning approach with two Naive Bayes classifiers that used different resampling strategies. We assessed all the models based on the accuracy and AUC. In addition, both of the Naive Bayes classifiers used the same data pre-processing steps as our deep learning model. Also, a 5-fold cross validation was used with a stratified shuffle split (80% training and 20% test).

V. RESULTS

In this section, we discuss the results of our models on the Steam game reviews.

Table 3 shows that our deep learning model from the gated recurrent neural network out-performed the Naive Bayes classifier with SMOTE across all performance metrics. However,

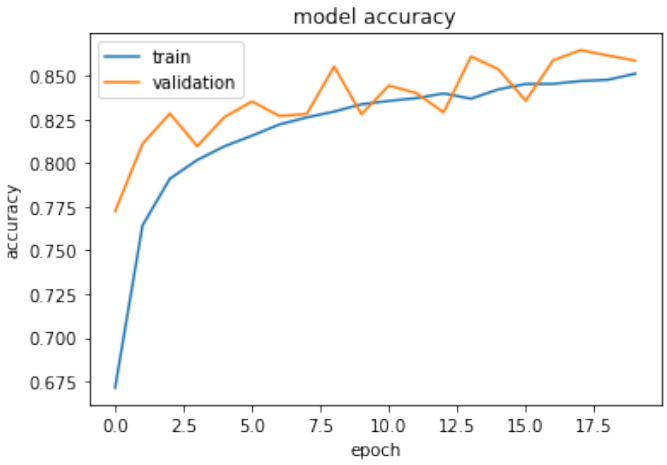

Fig. 3. The deep learning model accuracy per epoch.

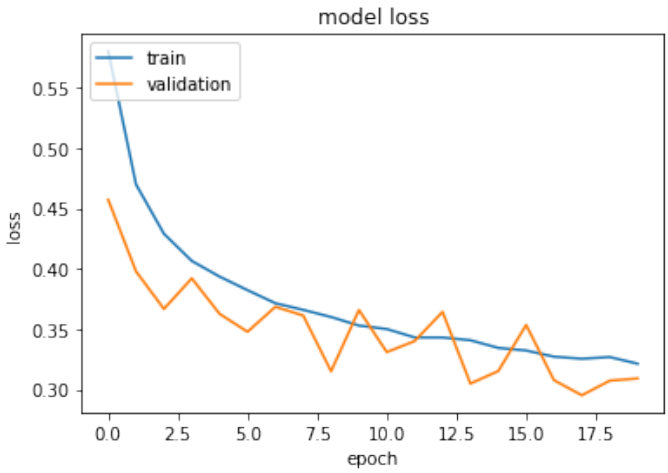

Fig. 4. The deep learning model loss per epoch.

we found that our deep learning model was out-performed by the Naive Bayes classifer that used under-sampling based on the performance metrics on the test set. A possible explanation for the poor performance of our deep learning model is that our pre-trained Word2vec model in the 5-fold cross validation model did not capture enough of the gamers vocabulary to generalize onto the test set. Another possible explanation is that we did not train the deep learning model long enough regarding epochs.

Figure 3 shows that the training accuracy is still increasing and may need more epochs to converge. In addition, Figure 4 shows that the training loss is still decreasing and may need more epochs to converge. However, Figure 4 shows that the model does overfit at certain times, which is shown from the training loss dipping under the validation loss. A possible explanation for the overfitting is because there is a class imbalance, which the class weights may not be fully resolving. It is likely that other measures are required to fix the class imbalance. Based on our results and insights, deep learning-based classifiers for sentiment analysis may not be robust against the inherent slang and sarcasm in game reviews, but does show potential in improving. We believe that more training, tuning, and features can significantly improve our deep learning model's performance.

VI. Discussion

In this section, we discuss some insights regarding the potential for our study, and some criticisms of similar work.

In our paper, we provide an approach of using deep learning to classify the sentiment of Steam game reviews. Although the Naive Bayes classifier out-performed our deep learning model, with respect to the test accuracy and AUC, we believe that adding an auxiliary input like such as hours of gameplay or compensation may help the deep learning model perform better on the test set to combat the sarcasm. For example, a game review may have sarcastic text that portrays the game in the opposite manner, but the hours of gameplay may contradict it.

To justify using the hours of gameplay as a feature for this problem, we calculated the delta days between the review post date and the initial game release date for just the top 10 games, with respect to the number of Steam game reviews. Figure 5 shows the number of days that it took a gamer to post a review after the initial game release. We wanted to see if there was any pattern in how long a gamer played the game before posting reviews based on the median delta days. We observed from Figure 5 that there are groups of games where gamers often post the review after the initial game release with a similar median delta days, which means that the hours of gameplay may have an impact on gamers posting a review. Hence, the hours of gameplay may be a significant feature in improving the deep learning model.

Furthermore, our results from Table I show that the gated recurrent neural network can learn well based on the training and validation accuracy and loss. However, the training and validation accuracy and loss also reveal that the deep learning model has room for improvement, possibly by increasing the number of epochs.

Bais et al. [1] attempted to solve the same problem of classifying the sentiment of Steam game reviews, but using traditional machine learning approaches (e.g., SVM, logistic regression) instead. Although our deep learning model did not out-perform their reported classifier, we believe that the methodology they used has severe flaws that may have biased their results, which we list below:

1) They used a spell corrector to pre-process the Steam game reviews, which may have changed the complete meaning of the sentence or word.
2) They did not mention handling any class imbalance, which may have biased their accuracy to lean towards the majority class.
3) They only used 5,000 samples of Steam game reviews, which may have biased the assessment of their classifiers.

Hence, we believe that the performance of their classifiers

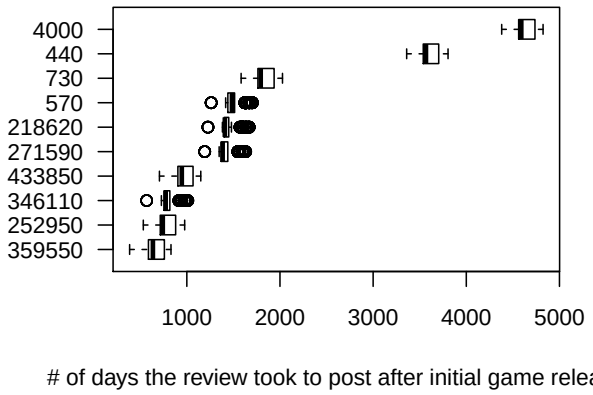

Fig. 5. The distribution of the number of days that it took a game review to be posted after the game was initially released per studied game ID.

would drastically change if they addressed the listed concerns.

VII. Conclusion

Sentiment analysis of sarcastic text is a challenging problem, even when disregarding external aspects such as context-based sentiment. However, the best solution to this challenging problem could help game developers quickly understand the needs of their gamer base. Hence, we propose a deep learning approach to help game developers accurately classify positive and negative reviews, so they can ultimately save time. We studied 53,764 game reviews from 3,625 Steam games. We built a gated recurrent neural network model, along with two Naive Bayes classifiers with different resampling strategies (i.e., SMOTE and Tomek link, and under-sampling) to account for the class imbalance.

In result, we found that our deep learning model did train well (85% median training accuracy and 85% median validation accuracy), but under-performed against the Naive Bayes classifer that used under-sampling when predicting the unseen test set. On the other hand, we observed that our deep learning model had potential to be improved with more epochs, and possibly with a broader vocabulary and additional features. Hence, there is potential that a deep learning approach can help game developers accurately classify game reviews as positive or negative.

References

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

[2] G. E. Batista, A. L. Bazzan, and M. C. Monard, "Balancing training data for automated annotation of keywords: a case study." in *WOB*, 2003, pp. 10–18.

[3] 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.

[4] 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.

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

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

[7] 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.

[8] 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.

[9] 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.

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

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

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

[13] 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.

[14] 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.

[15] ——, "An empirical study of early access games on the Steam platform," *Empirical Software Engineering*, vol. 23, no. 2, pp. 771–799, 2018.

[16] 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.

[17] ——, "An empirical study of game reviews on the steam platform," *Empirical Software Engineering*, vol. 24, no. 1, pp. 170–207, 2019.

[18] C. X. Ling, J. Huang, H. Zhang *et al.*, "Auc: a statistically consistent and more discriminating measure than accuracy," in *Ijcai*, vol. 3, 2003, pp. 519–524.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] C. Sangani and S. Ananthanarayanan, "Sentiment analysis of app store reviews," *Methodology*, vol. 4, no. 1, pp. 153–162, 2013.

[27] R. Sifa, C. Bauckhage, and A. Drachen, "The playtime principle: Large-scale cross-games interest modeling." in *Proceedings of CIG*, 2014, pp. 1–8.

[28] 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.

[29] B. Stråat and H. Verhagen, "Using user created game reviews for sentiment analysis: A method for researching user attitudes." in *GHITALY@ CHItaly*, 2017.

[30] 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.

[31] 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.

[32] Z. Zuo, "Sentiment analysis of steam review datasets using naive bayes and decision tree classifier," 2018.