Which Comment should I Look? A Data Driven Analysis on Reviews from the Developers' Standpoint

Shenghan Gao*
gaoshh1@shanghaitech.edu.cn
Shanghaitech University
Shanghai, China

Mingzheng Wu*
wumzh@shanghaitech.edu.cn
Shanghaitech University
Shanghai, China

ABSTRACT

Game reviews play a crucial role in providing feedback to developers during game development. Studies have emphasized the importance of consumer feedback in refining creations. Additionally, developer-audience interactions have been shown to boost consumer confidence and increase product sales. However, developers, especially smaller-scale or individual ones, often face constraints in processing and acting upon this feedback. This paper aims to tackle the question: "Which reviews are most valuable for developers?" We propose utilizing statics methods and developing a recommendation system using data from Steam, a leading platform in the gaming industry.

KEYWORDS

Recommendation system, Game developer, Game review, Steam

ACM Reference Format:
Shenghan Gao and Mingzheng Wu. 2018. Which Comment should I Look? A Data Driven Analysis on Reviews from the Developers' Standpoint. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Consumer reviews are essential for content producers, providing critical feedback. Numerous studies [6, 31] have highlighted the importance of consumer feedback, showing that it helps developers improve their products. Moreover, interactions between developers and users have been shown to boost consumer confidence and, consequently, product sales [31]. However, reviews can also be a double-edged sword for developers. Experts have found that low-quality comments can have negative impacts, including insults and other harmful content [5]. Therefore, it is crucial to identify the value of reviews, especially from the producers' perspective.

In analyzing reviews, previous researchers have proposed various indices, such as the readability index [8, 11] and the emotion index [14]. However, most of these indices focus on certain aspects of reviews. According to our literature review, there is no comprehensive index for evaluating the overall value of reviews. Additionally, reviews may receive responses from developers, and the relationship between reviews and their corresponding responses should not be overlooked. In this paper, we propose three indices to reflect the value of reviews based on their responses. Overall, we aim to answer what kind of reviews are important and how their features contribute to their value.

To address these questions, we designed several experiments. Initially, we utilized statistical methods, but the results showed little impact, probably due to the extreme complexity of reviews [4]. Consequently, we focused on deep-learning recommendation systems, which show strong potential in revealing relationships. We first built a four-layer MLP, which demonstrated a strong understanding ability, and then we utilized XAI techniques [1, 28] to decompose the DL black box.

In this paper, we take Steam[1], one of the famous game platforms, as an example to identify valuable comments. This paper contributes (1) indices extracted from responses to evaluate the value of reviews, (2) several experiments revealing the value of various review features, and (3) design suggestions based on experimental result analysis for review recommendation system design for developers.

2 BACKGROUND AND RELATED WORK

2.1 Review Analyze

In the sphere of game review analysis, scholarly efforts have delineated various investigative pathways. Various researchers[4, 12, 27] utilize multiple topic analysis methods, such as Structural Topic Model (STM) and LDA topic modeling. However, most of these studies approach the subject from the perspective of consumers. Lin et al. [18] conducted an empirical examination of 6,224 game reviews on the Steam platform and obtained valuable conclusions, including the insight that negative feedback, in particular, might offer more constructive insights for developers. However, their work does not consider various aspects of reviews such as emotions. Moreover, Nicholas et al.[8, 9, 25] summarize the evaluation criteria of reviews, such as novelty, and design a system to identify high-quality reviews. Nevertheless, their application scenario is focused on newspapers, where each review pertains to specific news articles. In contrast, game reviews are not merely textual and thus some indices, such as the relevance between reviews and news, may not be directly applicable.

*Both authors contributed equally to this research.

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

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

On Steam, a game review consists of the comment content, consumer information, other consumers' opinions on the comment, and possible responses from the developer, as shown in Figure 1

Figure 1: Comment

2.2 Analysis Techniques

For statistical methods, we use the Pearson correlation coefficient, a classical approach. However, the Pearson correlation coefficient only considers linear relationships. Therefore, we also introduce mutual information[17], which reveals the dependence between two variables, including nonlinear relationships.

Recommendation systems play an important role in our experiment designs. We introduce three major approaches: Collaborative Filtering (CF), Content-Based (CB), and Hybrid Filtering[7].

CF analyzes the similarities among users/items based on their previous interactions with items to predict a user's preference for certain items. Isinkaye et al.[15] highlight that CF demonstrates a capacity to perform effectively in scenarios where content, such as opinions, is challenging to process. However, CF encounters a cold-start problem, as it requires sufficient information to make relevant recommendations. Our observation of the current data indicates that only a limited number of creators are inclined to respond, which can further exacerbate this issue.

CB primarily focuses on the metadata extracted from users or items [29]. It has the potential to address the cold-start problem inherent in CF due to its lack of necessity for previous ratings. However, it may lead to an overspecialization issue [19], which could hinder creators from exploring a diverse array of reviews.

Hybrid Filtering amalgamates multiple recommendation techniques to capitalize on their strengths and mitigate their limitations. Drawing on Burke's research[3], it is not guaranteed that combining these methods will inherently yield superior results. Consequently, the intricacies of designing an effective combination remain an open challenge that necessitates further exploration.

Last but not least, the analysis and decomposition of recommendation systems are also important. In this paper, we utilize permutation importance[1] and SHapley Additive exPlanations (SHAP)[20]. Permutation importance involves randomly ranking the values of each feature and calculating the change in model performance. If the model performance decreases significantly after ranking a feature, that feature is considered important. SHAP is a game-theoretic method that assigns a "contribution value" to each feature, indicating its contribution to each prediction.

3 DATASET

3.1 Data Collection and Cleaning

We employed the Steam API and Steam Community API to extract a comprehensive list of applications using Python scripts. Subsequently, we refined this dataset, focusing solely on standalone applications and games while excluding other types such as DLCs (Downloadable Content) and supplementary materials associated with primary game titles. Following this filtration process, we embarked on a sampling strategy to further investigate the remaining list, targeting both user reviews and detailed game information, as shown in Table 2 and Table 3. To date, our dataset encompasses reviews and details from 3,229 games. This methodical approach ensures a broad yet manageable dataset for our analysis, facilitating a comprehensive understanding of user feedback and game features.

Considering the significant differences in languages, we limited our dataset to English-language reviews. After filtering, we obtained 2,148,974 reviews, of which 17,811 had received responses.

For some reasons, developers of certain Steam games have not responded to any reviews. This means that this portion of the data is completely unlabeled. Before training the model, we will remove this portion of the data.

As shown in Table 1, we are confronted with a significant data imbalance issue. Building upon the work of Drummond et al.[10], we believe that undersampling is a viable approach to address data imbalance. We categorize comments into two classes: those with responses and those without. Through undersampling, we equalize the number of comments in both classes, and then select nine-tenths of them as the training set, with the remainder serving as the test set.

Table 1: Dataset Stat

	num
APP	192301
Game	100411
Game(crawled)	3229
Review count	4450775
Review count with response	29141
Review count after basic filtering	2148973
Review count with reponse	17811
Review count after undersample	31960
Review count with reponse	15980

3.2 Data Processing

3.2.1 Non-Text Data Processing. For boolean and categorical features, we transform them into numerical form using one-hot encoding.

3.2.2 Textual Review Data Processing. The content of reviews plays an important role as a direct reflection of consumers' opinions and suggestions. Inspired by [25], we decide to extract topics and emotions from reviews and calculate their readability and brevity indices.

Topic: There are three main ways to extract topics: Term Frequency-Inverse Document Frequency (TF-IDF), Latent Dirichlet Allocation

Table 2: Properties Collected in Review

Category	Property	Type	Description
Review Content	Review Content	String	a textual content for a review
	Comment ID	Int	an unique id to identify review
	Reviewer Attitude	Boolean	the revier's attitude to the game
	Upvote Count	Int	the count of upvotes received from other gamers
	Fun Cout	Int	the count of fun received from other gamers
	Comment Count	Int	the count of comments received from other gamers
Reviewer Info	Steam ID	Int	an unique id to identify reviewer
	Playtime	Int	the time the reviewer has played
	Owned Game Count	Int	the number of games owned by the reviewer
	Review Count	Int	the number of reviews the reviewer has made
Developer Response	Developer Response	String	a textual content of a response to a review

Table 3: Properties Collected in Detail

Property	Type	Description
Name	String	the game's name
Steam AppID	Int	an unique id to identify app(game)
Game Description	String	an description for the game
Short Game Description	String	an short description for game
Category	Set	a set of labelled tags on features
Genre	Set	a set of labelled tags on game styles

(LDA) [2], and KeyBERT [13]. TF-IDF is an effective technique for identifying key and distinctive words. The formula for TF-IDF is defined as:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

Latent Dirichlet Allocation (LDA) is used as a robust unsupervised approach in topic modeling. It posits that a paragraph can be characterized by a probabilistic distribution of latent topics. KeyBERT is a powerful keyword extraction method based on the pre-trained BERT model. This method generates a list of topic pairs, with each pair consisting of topic phrases and their corresponding weights, typically identifying up to five topics. We use LDA with 50 clusters as the final method, as it presents topic distributions comprehensively and avoids the curse of dimensionality compared to the 11,000 clusters in KeyBERT.

Emotion: In this paper, we utilize VADER [14], a rule-based model for sentiment analysis. With this method, we obtain three indices to describe the emotion of reviews: negative, neutral, and positive.

Readability: We use the Flesch-Kincaid Reading Ease [16] as the index of readability. The Flesch-Kincaid Reading Ease is widely used by the U.S. Department of Defense.

Brevity: We split each comment into words by spaces and count the unique resulting tokens [8].

Length: We tokenize each comment and calculate its length directly. However, based on our observation of its distribution (see Figure 2), we decide to apply a logarithmic transformation to make the distribution more similar to a Gaussian distribution.

3.2.3 Textual Response Data Processing. The content of reviews plays an important role, indirectly reflecting the value of the reviews. Hence, we proposed three indices to evaluate the relationship between reviews and their corresponding responses.

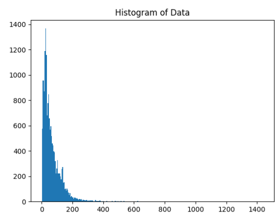

Figure 2: The Distribution of Response Length

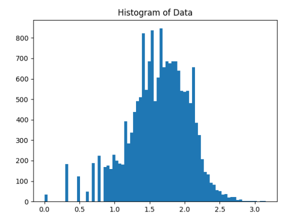

Figure 3: The Distribution of Response Length after Log

Length: We apply similar operations as for the length of reviews.

SimilarityBetween: This index denotes the similarity between a response and its corresponding review. If a response exhibits high similarity with its corresponding review or shares common keywords, it suggests targeted addressing by the developer rather than mere gratitude, further indicating the importance attributed to the review. The calculation of SimilarityBetween involves computing TF-IDF for all reviews and responses and then summing the weighted word vectors of the top 10 words to form vectors for each text segment. The cosine similarity between these vectors is used to measure their similarity.

UniquenessInner: This index represents the uniqueness of a response compared to other responses. A lower similarity between a response and others implies its uniqueness. Analyzing this uniqueness aids in understanding developers' response patterns. We calculate text segment vectors [21, 22]. Due to time complexity considerations, we compute the similarity between the specified response and 200 randomly selected distinct responses. The average of the top 50 similarity results is used as SimilarityInner. Then, we take its negative to obtain UniquenessInner.

4 EXPERIMENT I: STATISTICAL METHODS

In this experiment, we only consider reviews with responses and view three response text features as target values to calculate their correlation coefficients and mutual information indices. The results are shown in Table 5. However, it is difficult to obtain valuable insights from these statistical methods alone, so we need to utilize the power of deep learning recommendation systems.

5 EXPERIMENT II: RECOMMENDATION SYSTEM DESIGN AND EXPLANATION

5.1 Recommendation System Design

In this experiment, we consider all reviews. We will employ the user and comment features extracted using the previous method as inputs to the model, aiming to predict whether developers have responded, thus providing a more realistic outcome.

We employ different machine learning methods to predict whether a comment r has been responded to by developers, denoted as y_r. y_r is a boolean variable, where 1 indicates that comment r has been responded to, and 0 indicates no response. Initially, we attempt Support Vector Machine (SVM) with a Gaussian kernel and Multilayer Perceptron (MLP), with SVM serving as our baseline. Inspired by FuseRec [24], we recognize that comment features may vary significantly across different categories of games, and developers' tendencies towards comments may also differ. Therefore, we design a Categorized MLP(CMLP), the main structure of which is illustrated in Figure 4. On the left side, the number of output features after passing through the MLP layer is nearly the same as the number of input features. On the right side, the set of categories of the certain game g_r, denoted as $C(g_r)$, and the set of other categories, denoted as $C'(g_r)$, are input into the embedding layer E. The output results are then multiplied by learnable parameters α and β respectively, and summed up. This sum is then multiplied as the weights W of the left-side features. The formula for the weight w_i of feature f_i can be defined as:

$$w_i = \alpha \sum_{j \in C(g_r)} E_{ji} + \beta \sum_{j \in C'(g_r)} E_{ji}$$

Finally, the result goes through an MLP layer to obtain two scores $S = \{s_0, s_1\}$. The predicted y_r is then given by $\arg\max_i s_i \in \{s_0, s_1\}$.

5.1.1 *Loss Function.*

We use the binary cross-entropy loss, a classical loss function for binary classification models:

$$L_\theta = -\frac{1}{|B|} \sum_{(g,r) \in B} \left[y_{gr} \log(\hat{y}_{gr}) + (1 - y_{gr}) \log(1 - \hat{y}_{gr}) \right]$$

Where θ represents the model parameters and B is the currently sampled batch. For each review r of game g, y_{gr} is the label and $\hat{y}_{gr}$ is the predicted score.

5.1.2 *Evaluation.*

Based on previous research [26], recommendation systems have various metrics to evaluate performance. In this work, we use accuracy, recall, precision, and F1 score to measure the performance of our models.

5.1.3 *Recommendation System Performance.*

As shown in Table 4, our CMLP shows the best results in terms of accuracy, recall, and F1 score. However, we also find that the inclusion of category features makes it challenging to evaluate the importance of individual features because they are modified by category embeddings. Therefore, in our explanation work, we will take the standard MLP into consideration for subsequent analysis.

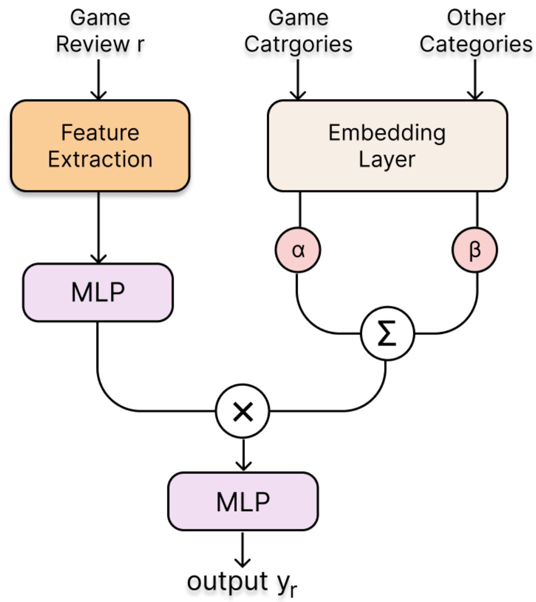

Figure 4: The structure of Categorized Multilayer Perceptron

Table 4: Results of SVM,MLP and CMLP

	SVM	MLP	CMLP
ACC	0.7531	0.7506	0.7812
Rec	0.6789	0.7685	0.8228
Pre	0.7980	0.7446	0.7601
F1	0.7336	0.7551	0.7902

5.2 Explanation of the Recommendation System

5.2.1 *Methods.*

As introduced before, we utilize permutation importance and SHAP to interpret the black box of our model.

5.2.2 *Experimental Results.*

The detailed results are shown in Appendix B.

5.3 Experimental Setting

All code was executed on an Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz, equipped with two NVIDIA GeForce RTX 3090 GPUs. The software environment was based on Python 3.12.2.

6 DISCUSSION

6.1 Findings on Permutation Importance and SHAP

As shown in Figures Figure 5 and Figure 6, the attitude of reviews plays the most important role, with negative reviews tending to receive more attention from developers, which is in line with Zhuang et al.'s findings [31]. Additionally, compared to Diakopoulos's work [8], which highlights the importance of readability, readability scores seem less impactful in game reviews. We also found that votes on reviews from other consumers (votes_up, voted_funny) show minimal impact, but their comments may attract developers. The more active the discussions are under a certain review, the more likely developers will respond.

6.2 Findings on Recommendation System

As shown by our model's performance, our CMLP exhibits stronger capabilities. This implies that different categories of developers show significant variation in their preference for reviews. However, our approach of incorporating category features also complicates the interpretation of the models. As shown in Table 7 and Table 6, the variance in feature importance in the CMLP is not obvious. We believe this is because the category embeddings modify their weights again. Such a policy might lower the confidence in our model in real scenarios because this importance performance will reduce the perceived usefulness for consumers, based on the IAM model [30]. Based on our experiments, if we incorporate category embeddings before the first layer of the MLP or treat the category as a feature, the accuracy is similar to that of the standard MLP.

Hence, we propose several suggestions for designing recommendation systems: (a) Category embeddings show strong potential for improving the performance of recommendation systems. (b) If category embeddings are incorporated, the transparency of the systems shown to users should be strengthened, and some HCI (Human-Computer Interaction) knowledge might be useful.

7 LIMITATION

In this work, we faced limitations due to the poor quality of data, primarily caused by the limited response rate of developers. Additionally, the current quantified indices may not describe reviews and responses comprehensively. For example, novelty, as mentioned in *The Editor's Eye: Curation and Comment Relevance on the New York Times* [8], cannot be quantified and therefore cannot be utilized by automated computing. Moreover, as mentioned previously, the explanation results of the CMLP are affected by category embeddings, reducing its transparency.

8 CONCLUSION

In this paper, we designed three indices to quantify the value of reviews from the perspective of developer responses. We employed statistical methods and explanation techniques for recommendation systems to reveal the value of various features. Finally, we provided two design suggestions for recommendation systems based on the experimental results.

In the future, we plan to crawl more data to improve our data quality. Additionally, there are some interesting indices, such as "personal experience" from LIWC[2], which we did not utilize due to some technical problems. In the next phase, we plan to overcome these issues to provide a more comprehensive description of reviews. Moreover, we plan to introduce DiCE [23], a counterfactual explanation tool, to enhance our explanatory capabilities and consider multivariate analysis.

[2]https://www.liwc.app/

ACKNOWLEDGMENTS

We would thank Prof. Haipeng Zhang, Prof. Quan Li, Shengxin Li and all our dear classmates for their valuable response.

REFERENCES

[1] André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. 2010. Permutation importance: a corrected feature importance measure. *Bioinformatics* 26, 10 (2010), 1340–1347.

[2] David Blei, Andrew Ng, and Michael Jordan. 2001. Latent dirichlet allocation. *Advances in neural information processing systems* 14 (2001).

[3] Robin Burke. 2007. Hybrid web recommender systems. *The adaptive web: methods and strategies of web personalization* (2007), 377–408.

[4] I. Busurkina, V. Karpenko, E. Tulubenskaya, and D. Bulygin. 2020. Game Experience Evaluation. A Study of Game Reviews on the Steam Platform. In *Digital Transformation and Global Society (Communications in Computer and Information Science, Vol. 1242)*, D.A. Alexandrov, A.V. Boukhanovsky, A.V. Chugunov, Y. Kabanov, O. Koltsova, and I. Musabirov (Eds.). Springer, Cham. https://doi.org/10.1007/978-3-030-65218-0_9

[5] Lily Canter. 2013. The misconception of online comment threads: Content and control on local newspaper websites. *Journalism practice* 7, 5 (2013), 604–619.

[6] Kejia Chen, Jian Jin, and Jiayi Luo. 2022. Big consumer opinion data understanding for Kano categorization in new product development. *Journal of Ambient Intelligence and Humanized Computing* (2022), 1–20.

[7] Aminu Da'u and Naomie Salim. 2020. Recommendation system based on deep learning methods: a systematic review and new directions. *Artificial Intelligence Review* 53, 4 (2020), 2709–2748.

[8] Nicholas Diakopoulos. 2015. Picking the NYT picks: Editorial criteria and automation in the curation of online news comments. *ISOJ Journal* 6, 1 (2015), 147–166.

[9] Nicholas A Diakopoulos. 2015. The editor's eye: Curation and comment relevance on the New York times. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. 1153–1157.

[10] Chris Drummond, Robert C Holte, et al. 2003. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, Vol. 11.

[11] Rudolf Flesch. 2007. Flesch-Kincaid readability test. *Retrieved October* 26, 3 (2007), 2007.

[12] Gustavo Fortes Tondello, Deltcho Valtchanov, Adrian Reetz, Rina R Wehbe, Rita Orji, and Lennart E Nacke. 2018. Towards a trait model of video game preferences. *International Journal of Human–Computer Interaction* 34, 8 (2018), 732–748.

[13] Maarten Grootendorst. 2020. KeyBERT: Minimal keyword extraction with BERT. https://doi.org/10.5281/zenodo.4461265

[14] Clayton Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, Vol. 8. 216–225.

[15] F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh. 2015. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal* 16, 3 (2015), 261–273. https://doi.org/10.1016/j.eij.2015.06.005

[16] J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. (1975).

[17] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E* 69, 6 (2004), 066138.

[18] D. Lin, CP. Bezemer, Y. Zou, et al. 2019. An empirical study of game reviews on the Steam platform. *Empir Software Eng* 24 (2019), 170–207. https://doi.org/10.1007/s10664-018-9627-4

[19] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. 2015. Recommender system application developments: a survey. *Decision support systems* 74 (2015), 12–32.

[20] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 4765–4774. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

[23] Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 607–617.

[24] K. Narang, Y. Song, A. Schwing, et al. 2021. FuseRec: fusing user and item homophily modeling with temporal recommender systems. *Data Min Knowl Disc* 35 (2021), 837–862. https://doi.org/10.1007/s10618-021-00738-8

[25] Deokgun Park, Simranjit Sachar, Nicholas Diakopoulos, and Niklas Elmqvist. 2016. Supporting comment moderators in identifying high quality online news comments. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1114–1125.

[26] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. *Recommender systems handbook* (2011), 257–297.

[27] Dorinela Sirbu, Ana Secui, Mihai Dascalu, Scott Andrew Crossley, Stefan Ruseti, and Stefan Trausan-Matu. 2016. Extracting Gamers' Opinions from Reviews. In *2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. 227–232. https://doi.org/10.1109/SYNASC.2016.044

[28] Erik Štrumbelj and Igor Kononenko. 2014. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems* 41 (2014), 647–665.

[29] Xinxi Wang and Ye Wang. 2014. Improving Content-based and Hybrid Music Recommendation using Deep Learning. In *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, Florida, USA) *(MM '14)*. Association for Computing Machinery, New York, NY, USA, 627–636. https://doi.org/10.1145/2647868.2654940

[30] Yu Wang. 2016. Information adoption model, a review of the literature. *Journal of Economics, Business and Management* 4, 11 (2016), 618–622.

[31] Wei Zhuang, Qingfeng Zeng, Yu Zhang, Chunmei Liu, and Weiguo Fan. 2023. What makes user-generated content more helpful on social media platforms? Insights from creator interactivity perspective. *Information processing & management* 60, 2 (2023), 103201.

A RESULTS OF EXPERIMENT I

This section contains the results of Experiment I

Table 5: Results of Experiment I

		Corrlation Coefficience			Mutual Infomation		
		RL	SB	UI	RL	SB	UI
Reviewer	Review Count	-0.0224	-0.0101	0.0024	0.0101	0.0072	0.0114
	Game Count	-0.0309	0.0109	0.0161	0.0	0.0	0.0017
	Playtime Forever	0.0480	0.0024	-0.0118	0.0353	0.0	0.0184
	Playtime at Review	0.0473	0.0118	-0.0175	0.0428	0.0	0.0095
	Playtime Last Two Week	0.0218	0.0169	-0.0315	0.0	0.0030	0.0003
Direct Feature	Voted Up	-0.3460	-0.0072	-0.0400	0.0801	0.0051	0.0095
	Votes Up	0.0754	0.0683	-0.0157	0.0315	0.0087	0.0071
	Votes Funny	0.0042	-0.0031	0.0368	0.0084	0.0008	0.0020
	Comment Count	0.0949	0.0461	-0.0037	0.0289	0.0069	0.0067
	Is Steam Purchase	0.0119	0.0230	0.0832	0.0190	0.0107	0.0134
	Is Received for Free	-0.0140	0.0060	-0.0095	0.0054	0.0	0.0013
	Is EA	0.1140	0.0760	-0.0990	0.0428	0.0063	0.0059
Review derived	Length	0.3049	0.3383	-0.0515	0.0658	0.1244	0.0277
	Negative	0.0854	-0.0726	-0.0196	0.0399	0.0394	0.0157
	Neural	0.1927	0.0110	0.07549	0.0353	0.0517	0.0074
	Positive	-0.2769	0.0657	-0.0654	0.0671	0.0430	0.0150
	Readability Score	-0.0946	-0.0728	-0.0293	0.0397	0.0307	0.0077
	Brevity	0.2608	0.2474	-0.0823	0.0866	0.1056	0.0125

B RESULTS OF EXPERIMENT II

This section contains the results of Experiment I

B.1 Permutation importance

Table 6: Result of Permutation Importance on non-Topic Features

Feature	Importance CMLP	Importance MLP
review_count	0.0738	0.0004
game_count	0.0736	0.0009
playtime_forever	0.0746	0.0001
playtime_at_review	0.0758	-0.0007
playtime_last_two_week	0.0745	-0.0009
voted_up	0.2088	0.1534
votes_up	0.0744	0.0011
votes_funny	0.0744	-0.0004
comment_count	0.0856	0.0145
steam_purchase	0.0805	0.0036
received_for_free	0.0734	-0.0003
written_during_early_access	0.0714	0.0165
neg	0.0761	0.0013
neu	0.0756	0.0002
pos	0.0752	0.0012
length	0.0849	0.0192
readability_score	0.0740	0.0001
brevity	0.0743	-0.00007

Table 7: Result of Permutation Importance on Topic Features

Feature	Importance		Feature	Importance	
	CMLP	MLP		CMLP	MLP
topic_0	0.0804	0.0103	topic_25	0.0750	0.0010
topic_1	0.0739	0.0002	topic_26	0.0757	0.0021
topic_2	0.0752	0.0009	topic_27	0.0739	-0.00002
topic_3	0.0774	0.0032	topic_28	0.0742	-0.00001
topic_4	0.0748	0.0005	topic_29	0.0742	-0.0007
topic_5	0.0744	-0.0007	topic_30	0.0759	0.0036
topic_6	0.0744	0.0003	topic_31	0.0732	-0.0004
topic_7	0.0745	0.0014	topic_32	0.0747	-0.00001
topic_8	0.0738	0.0003	topic_33	0.0776	0.0024
topic_9	0.0750	0.0016	topic_34	0.0757	0.0010
topic_10	0.0736	0.0004	topic_35	0.0742	0.0005
topic_11	0.0756	0.0007	topic_36	0.0733	0.0006
topic_12	0.0740	0.0011	topic_37	0.0746	-0.0003
topic_13	0.0735	0.0001	topic_38	0.0747	0.0007
topic_14	0.0738	0.0008	topic_39	0.0736	0.0007
topic_15	0.0740	0.0008	topic_40	0.0751	0.0004
topic_16	0.0736	-0.0004	topic_41	0.0742	-0.0002
topic_17	0.0737	-0.0003	topic_42	0.0744	-0.0008
topic_18	0.0733	0.0005	topic_43	0.0796	0.0041
topic_19	0.0747	-0.00004	topic_44	0.0729	-0.0002
topic_20	0.0748	0.0003	topic_45	0.0742	-0.0004
topic_21	0.0744	0.0010	topic_46	0.0745	0.0009
topic_22	0.0729	0.0001	topic_47	0.0722	-0.0007
topic_23	0.0735	-0.00008	topic_48	0.0742	0.0010
topic_24	0.0755	0.0015	topic_49	0.0734	-0.0005

B.2 SHAP

The following are SHAP on non-topic features

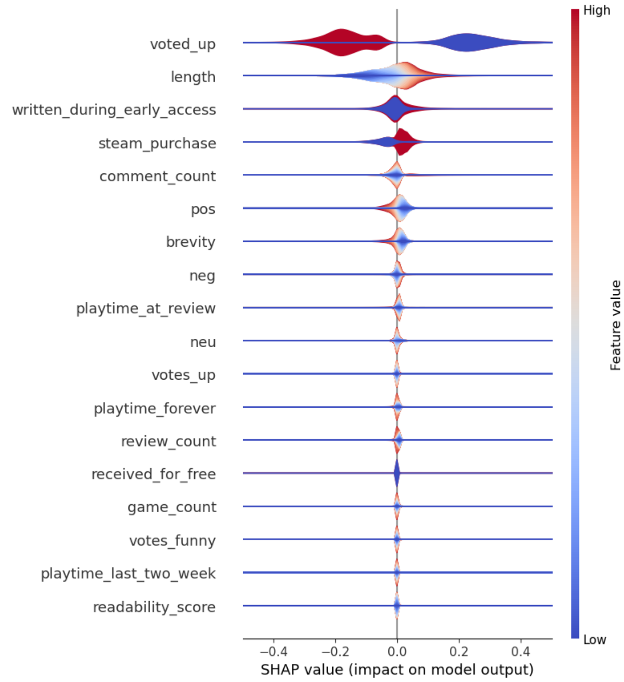

Figure 5: Results of SHAP on CMLP with non-Topic Features

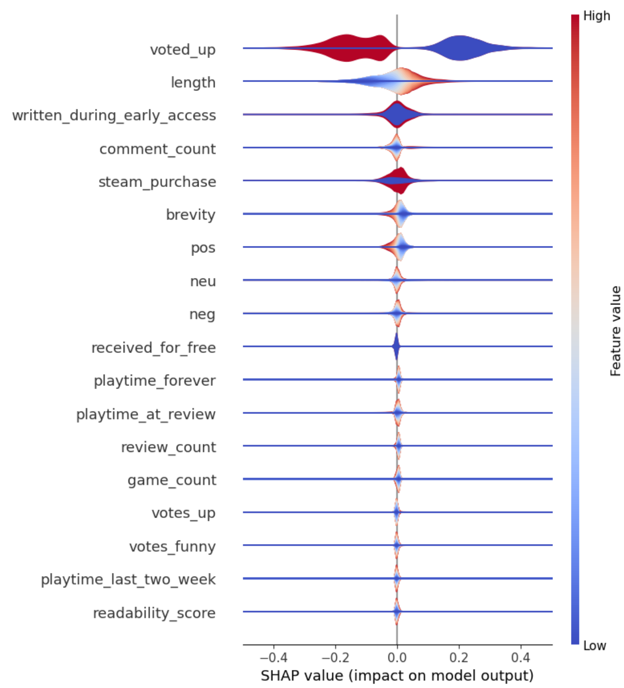

Figure 6: Results of SHAP on MLP with non-Topic Features

The following are SHAP on topic features

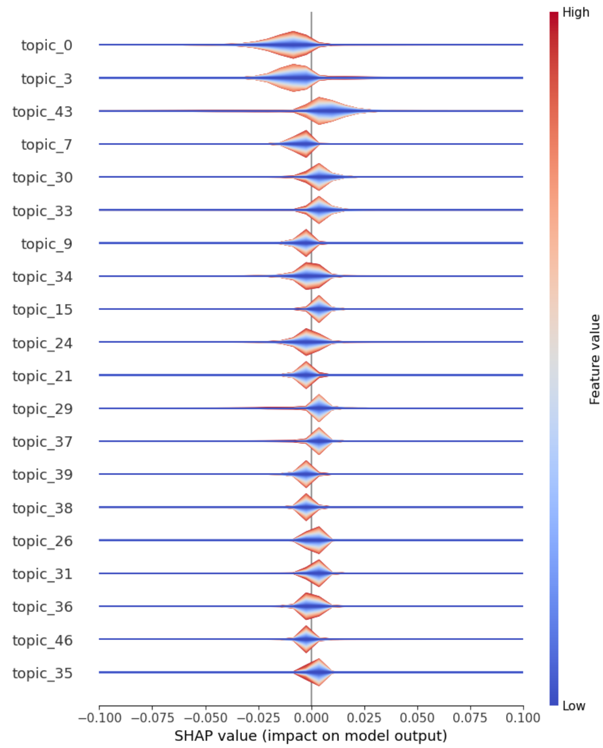

Figure 7: Results of SHAP on CMLP with Topic Features

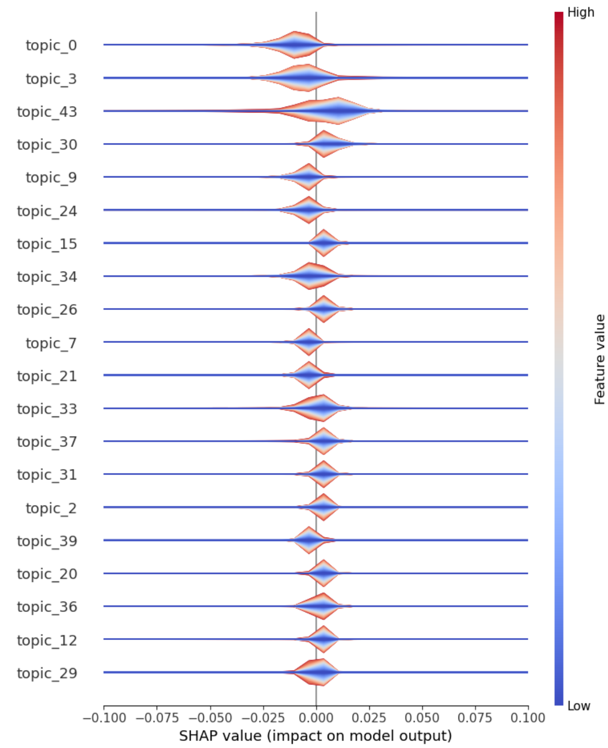

Figure 8: Results of SHAP on MLP with Topic Features

C MAP/REDUCE

Original API	Map/Reduce
32.9207	5231.4198

We try to utilize Map/Reduce on calculation of brevity index but show poor performance. This may because the index take a review as a unit and the amopunt of a review could show the power of Map/Reduce.