Efficient Analysis of User Comments in Board Game Reviews: Topic Identification and Evaluation

Martín do Río Rico[1]

[1]Department of Computer Science, Università degli Studi di Milano, Milano, Italy.
[2]Department of Computer Science, Universidade de A Coruña, A Coruña, Spain.
[*]Address correspondence to: martin.doriorico@studenti.unimi.it

Abstract

This scientific report presents a project focused on analyzing user comments from board game and expansion reviews. The aim is to identify specific aspects of the games, such as luck dependency, bookkeeping or downtime, player interaction, advantage discrepancies, and rule complexity. The project also aims to develop a user-friendly system that minimizes human involvement for program execution. A original approach is employed to construct a selective topic identification dictionary, considering user-specific needs. Challenges related to abstract topic detection and bias towards tangible aspects are discussed. The system design involves corpus extraction, preprocessing, dictionary generation, LDA topic modeling, and manual labeling. The project provides insights into efficient topic identification and analysis of user comments in board game reviews.

1 Introduction

The objective of this project is to perform an analysis of user comments extracted from reviews pertaining to various board games and expansions. The primary aim is to ascertain whether these comments address specific aspects of the games in question. These aspects include:

- Assessment of the game's reliance on luck, determining whether it is excessively luck-dependent.
- Evaluation of the presence of substantial bookkeeping or unproductive downtime during gameplay.
- Examination of the level of interaction facilitated among players during the game.
- Identification of instances where one player attains an excessive advantage in the later stages of gameplay.
- Assessment of the complexity of the game's rules and mechanics, focusing on whether they are intricate but lack depth.
- Analysis of whether the rules and mechanics are simplistic, yet offer substantial depth to the overall gaming experience.

Beyond the specific objectives mentioned above, an additional aim of this project is to develop a system that minimizes the level of human involvement required for efficient program execution. This is achieved by reducing the program's parameters, thereby enabling users with limited programming knowledge to utilize it effectively. Due to time constraints, the system does not include a user interface and functions primarily as an analytical tool, rather than a comprehensive study of a singular dataset.

Due to the absence of a separate corpus to extract a pre-defined dictionary from or an automated method for generating a dictionary for training the LPA (Latent Dirichlet Allocation) model, I devised a novel approach to construct a dictionary that allows for selective topic identification. Unlike merely classifying the corpus into predefined topics, this approach considers the specific information requirements and needs of the user.

The developed system requires minimal intervention from the user. They can generate a dictionary by providing related words that describe the desired topics. To ensure comprehensiveness, relevant terms from the actual corpus are incorporated, mitigating the risk of overlooking common terms that the user may have unintentionally omitted.

Initially, my primary goal was to automate the process to the greatest extent possible. However, due to the inherent inconsistencies arising from the automated topic generation methods, I introduced an additional approach involving manual review and tagging of the generated topics. This allows for greater control and refinement in the identification of relevant topics.

Following multiple rounds of experimentation and meticulous refinement conducted by the program, it has reached a state where its predicted results demonstrate an acceptable level of performance. However, to optimize the quality of the outcomes, it is advisable to have a human operator oversee the process of dictionary generation and topic labeling. Despite the program's advancements, certain challenges have been encountered, two of which are worth highlighting:

- Bias towards Tangible Topics: Users tend to provide comments more frequently on tangible aspects such as rule complexity or luck dependency, as they perceive these issues to be more significant. In contrast, they may not express concerns about the lack of player interaction as frequently, even though it may be equally relevant.

- Difficulty in Detecting Abstract Topics: Some of the more abstract topics are exceedingly challenging to systematically identify, as they are contingent upon the individual player's perspective and are often communicated within the context of broader, overarching issues. For instance, users may express dissatisfaction with game balance or poorly designed end-game scenarios, rather than explicitly mentioning an unfair advantage held by a specific player in later stages of the game.

These challenges highlight the inherent complexities associated with accurately detecting and labeling certain topics, warranting the involvement of a human operator to ensure a comprehensive and nuanced analysis.

2 Background

During the program design phase, extensive research was conducted on various areas related to the algorithms involved. The following topics were explored:

- Synset Generation using WordNet: The focus was on understanding how to generate synsets (sets of synonymous words) using WordNet and directing the synonym generation process towards specific topics. [1]
- Topic Modeling in Python: The focus here was on understanding the concept and implementation of topic modeling using Python. [2]
- Selection of Optimal Number of Topics: The exploration involved finding an optimal number of topics for the program before adopting an alternative approach for determining the number of topics. [3] [4]
- Improving Gensim Program Performance: The aim was to enhance the performance of the program built with the Gensim library.

By researching these areas, a comprehensive understanding of the underlying algorithms and techniques was gained, facilitating the development of an effective program design.

3 System Design

During the developmental phase, the corpus was extracted from the comments on board games and expansions found in the "Hot Topics" section of the BoardGameGeek (BGG) website. This approach ensured the acquisition of a substantial and unbiased dataset. Once the code development was completed, another retrieval was performed from the same "Hot Topics" section.

The collected comments, along with relevant information, were stored in a database. Subsequently, preprocessing was applied to each comment, and the processed results were saved in the same database. The following steps were undertaken to enhance the quality of the data:

1. Reviews without text were removed from the dataset.
2. Non-English comments were efficiently deleted, albeit sacrificing some readability in favor of time efficiency.
3. The number of words and characters in each comment were recorded in the database.
4. Messages shorter than five words were excluded from further analysis.
5. Punctuation marks were removed, and the text was converted to lowercase.
6. Tokenization, stopword removal, lemmatization, and stemming were performed using the Gensim library. The processed text was saved in the database.
7. An additional step was implemented to filter out verbs.
8. Resulting term lists with fewer than five elements were discarded.

After the text underwent various preprocessing steps, the dictionary generation process commenced. To accomplish this, a user-driven system was developed. The user was prompted to select the desired topic and provide related words and terms in five separate lists. These input lists were utilized to generate a variety of synonyms and related terms. Additionally, a function was implemented to display the longest comments, allowing the user access to common vocabulary found in the reviews.

The input word lists were categorized based on their respective subjects to mitigate biases introduced by using a single comprehensive list. The recommended categories were as follows: actions or verbs, components or nouns related to the topic, positive adjectives, negative adjectives, and any words not fitting into the previous categories.

The dictionary was completed by incorporating the most frequently occurring terms from the corpus. This was done to ensure coverage in cases where the selected topics were not adequately represented or the generated dictionary lacked sufficient size to yield accurate results. A threshold was set to include words appearing in more than five percent of comments. Furthermore, a curated list of frequent terms that influenced dictionary creation was removed prior to saving.

During this process, two reference JSON files were generated. One file facilitated the transformation of stemmed terms into their full word forms, while the other file tracked the topic from which each word was generated. Following the dictionary creation, the LDA model was trained using the corpus and the generated dictionary. The resulting model was saved, and the outcomes were presented in both readable word format and parent category format when generating synonyms. This allowed for manual labeling of the topics by the user.

The automatic labeling procedure involved calculating the origin count of each term for each topic. Original categories that appeared more than five times were considered, and the most frequent category was assigned as the label. In cases where a tie occurred, multiple labels were assigned. Notably, the original categories "complex" and "complicated" were merged due to significant overlap

in their concepts. All the necessary information for automatic labeling was derived from the JSON files created during the dictionary creation process.

Initially, the number of topics might seem counterintuitive. In the early stages of development, a test was conducted, indicating that around 10 topics yielded the most consistent results for clustering comments under general topics. However, after several iterations of the code, it was discovered that generating more topics than needed and selectively focusing on specific topics aligned with desired criteria proved to be a more effective approach for identifying specific topics.

Finally, the results could be examined in various formats. To evaluate the program's fidelity, a subset of comments was manually reviewed, and related categories were determined. This allowed for a side-by-side comparison between the model's results and the reviewer's findings.

4 Evaluation

Three distinct sets of tests were devised to evaluate the performance of the system:

1. Pseudo-supervised Testing: In this test, forty random comments were manually labeled before the final dataset of reviews underwent dictionary creation. After training the model, the results of these labeled comments were extracted and compared. This comparison allowed for an assessment of false negatives and false positives.

2. Random Comment Evaluation: Code was implemented to enable users to display and review the results of random comments from the entire database. This feature facilitated a more comprehensive evaluation of individual performance.

3. Top Comment Display: The system includes code that displays the top five comments from the database based on different factors. It takes into account that longer messages have a disadvantage as the program produces results that add up to 1. Consequently, longer messages tend to have lower percentages for each topic. On the other hand, shorter messages that focus on a single topic receive a higher estimation for that particular topic. To address this issue, a solution inspired by inverse document frequency was implemented. Two sets of values are provided: the percentage estimation and the relative value relative to the number of matches against the dictionary vocabulary. The latter is used to determine if there is sufficient data for a precise estimation, with a threshold set at five matches. Users have access to these two sets of values to sort and evaluate the performance of the program across the entire database, providing a broader perspective.

These testing methodologies allow for a comprehensive assessment of the program's performance, covering both individual comments and the overall database.

5 Conclusions

After successfully implementing a satisfactory dictionary generation system, I proceeded with the labeling phase using different parameters within the same Latent Dirichlet Allocation (LDA) model.

Based on the outcomes of various experiments, it can be inferred that the automatic labeling system, as evidenced by results from a small sample, is equally if not more efficient than manual labeling. However, it is worth noting that the program exhibited superior performance on certain topics. Notably, the topic of 'luck' consistently emerged throughout the entire development phase. Regardless of the number of topics the model was configured to identify (ranging from six to thirty), it consistently generated a single topic that encompassed nearly all terms associated with 'luck'.

Measurements	Manual Labels	Bound 5 Labels	Bound 4 Labels
Accuracy	62.92%	68.75%	59.58%
Positives	69.12%	46.32%	80.88%
Negatives	60.47%	83.45%	51.16%

Table 1: Accuracy results

Conversely, topics such as 'downtime' and 'interaction' displayed a considerably higher number of false negatives compared to true positives. Furthermore, the model exhibited a tendency to misclassify comments unrelated to these topics as pertaining to 'bash the leader' and 'bookeeping'.

Measurements	Complexity	Luck	Interaction	Bash the leader	Downtime	Bookeeping
True positives	23	8	4	2	1	6
False negatives	3	0	10	2	5	4
False positives	11	2	8	8	10	12

Table 2: Individual labels results

Hypothesizing on the factors influencing these results, three key aspects come to light. Firstly, the significance attributed to different issues by the board game community plays a role. Specifically, players place great importance on game engagement, balance, and ease of learning. Consequently, topics like 'complexity' are frequently mentioned, providing the model with a higher chance of detecting related terms and achieving accurate classification.

Secondly, when writing reviews, users tend to avoid subjective phrases such as 'I felt like...' or 'For me...'. Instead, they prefer to express more assertive statements. This tendency can lead to issues of morphing, wherein a comment that might have originally conveyed a lack of player involvement in combat, for example, becomes 'The combat system drags for too long'. As a result, more abstract and subjective topics like player 'interaction' or unproductive 'downtime' may suffer, leading to the misclassification of comments addressing these aspects.

Lastly, certain topics prove to be excessively specific, hindering precise classification. This is evident from the higher proportion of false positives observed for topics such as 'bookkeeping' and 'bash the leader'. The LDA model tends to assign these labels to more general topics that encompass specific ones. For instance, comments discussing 'bash the leader' may also touch upon the game balance, end-game scenarios, and winning strategies, all of which are inherently linked to the concept of 'bash the leader'.

References

[1] Hongyan Jing. "Usage of WordNet in Natural Language Generation". In: *Columbia University* 1998 (1998), pp. 1–7.

[2] Micah D. Saxton. "A Gentle Introduction to Topic Modeling Using Python". In: *Theological Librarianship* 2018 (2018), pp. 1–10.

[3] Jingxian Gan and Yong Qig. "Selection of the Optimal Number of Topics for LDA Topic Model—Taking Patent Policy Analysis as an Example". In: *Entropy* 2021 (2021), pp. 1–45.

[4] Marten Wegkamp Xin Bing Florentina Bunea. "Optimal Estimation of Sparse Topic Models". In: *Journal of Machine Learning Research* 2020 (2020), pp. 1–45.