INTEK Jurnal Penelitian
Vol. 11, No. 2, pp. 1-7, Oktober 2024

Compound Critiquing Approach for Laptop Recommendation in Conversational Recommender System Using Collaborative Filtering

Ummu Husnul Khatimah[1,a] and Z. K. A. Baizal [2,b*]
[1,2] Department School of Computing, Telkom University, Jl. Telekomunikasi No. 1, Terusan Buah Batu, Bandung, Jawa Barat, Indonesia, 40257
[a] ummuhusnul@student.telkomuniversity.ac.id, [b*] baizal@telkomuniversity.ac.id (Corresponding Author)

***Abstract*—This study proposes a method for laptop recommendation in a conversational recommender system (CRS) by integrating collaborative filtering with the Apriori algorithm. The CRS interacts with users to help them find laptops that match their preferences, allowing them to provide feedback or critiques on the recommendations. This research emphasizes the use of compound critiques, which allow users to express preferences on multiple attributes at once, leading to more personalized recommendations. The Apriori algorithm identifies frequent itemsets from these critiques, which are then used to iteratively update recommendations. Evaluation results show that the High Support (HS) strategy, which focuses on commonly preferred features, produces more efficient recommendations, with a shorter average session duration of 38.01 seconds compared to the Low Support (LS) 41.30 seconds and Random (RAND) 50.49 seconds. This approach improves the recommendation process by better aligning with user preferences, which in turn improves interaction efficiency.**

Keywords*— *Apriori algorithm, collaborative filtering, conversational recommender system, compound critiques

I. Introduction

The development of laptops continues to enhance specifications and features to meet the needs of academic, office, and everyday activities [1]. However, these features often make it difficult to select the right laptop, especially for specific needs like programming or graphic design. Therefore, a recommender system is needed to help users find laptops according to their preferences without requiring deep technical knowledge [1]. The goal of a recommender system is to provide efficient suggestions that match user needs.

A recommender system aims to offer recommendations that are efficient and tailored to the needs and preferences of users [2]. A Conversational Recommender System (CRS) is designed to help users find products that fit their needs through conversational interaction [3]. In the recommender system, users will enter their preferences. Based on these preferences, the collaborative filtering model will predict recommendations that align with the user's preferences. However, if users are not satisfied with the recommendations, they cannot provide critique, so CRS is required to interact continuously with the system, and the system will serve users through various questions or request feedback [4]. In terms of feedback, critiquing is used. Critiquing is an approach in recommender systems that allows users to provide feedback or critique the recommendations provided [5].

Previous research that implemented a recommendation system for chatbot-based laptops showed several weaknesses, such as chatbots that only provide recommendations based on limited criteria, so users have limitations when asking for other recommendations [6]. In addition, other research also shows that laptop recommendation systems often do not allow users to provide feedback or interact with the system to get better recommendation results [7]. These weaknesses indicate that current recommendation systems are not yet able to handle complex and dynamic

DOI: https://doi.org/10.31963/intek.v11i2.4893

user preferences. In addition to the research mentioned, there have been several studies that developed and evaluated the compound critiquing method in CRS. One of the advantages of this approach is the innovation in compound critiquing, which allows users to provide more relevant feedback compared to unit critiquing. By using compound critiquing, users can express their preferences on multiple attributes at once, which significantly improves the user experience in recommendation systems [8]. In addition, this research also proposes a dynamic approach in critique selection, where the displayed critiques are customized based on previous interactions and remaining products. This increases the relevance of the critiques provided and helps users make better decisions [9]. The use of the Apriori algorithm in generating comp critiques also demonstrates a powerful data-driven approach, which is not only relevant but also reliable, providing a solid foundation for the development of more effective recommendation systems [9].

Therefore, this research proposes a CRS method that uses compound critiquing as a mechanism to obtain feedback from users on the recommendations [10]. Instead of providing feedback on one attribute at a time [11], users can express their preferences simultaneously on multiple attributes in a single critique. This method has been previously tested using compound critiques and the Apriori algorithm [8], [12]. Apriori algorithm is one of the algorithms for finding association rules in data mining [13]. It is efficient in generating compound critiquing and has a significant impact on system efficiency as the feedback provided by users consists of a set of product attributes [14]. For example, in the context of a laptop recommendation system, a user might say, “I want a laptop with more RAM and memory”. The system then adjusts subsequent recommendations based on this combined criticism, reflecting the user's preferences more comprehensively. The innovation of this research is that users can critique the recommendation results from the system not only for one feature, but for more than one feature.

Based on this background, this study proposes a CRS method that uses compound critiquing as a mechanism to obtain feedback from users on the recommendations provided [8]. With this approach, users can express their preferences simultaneously on multiple attributes in a single critique, providing more accurate recommendation results that suit their needs. Based on this background, this research has the following main contributions: To recommend laptops based on compound critiquing that fit the Collaborative Filtering model and evaluate the efficiency of Apriori algorithm in compound critiquing.

II. Research Methodology

In this research, the design of a critiquing recommender system aims to improve recommendation personalization by utilizing user feedback. The system combines the Collaborative Filtering algorithm with the Apriori method to identify and manage multiple critiques from users.

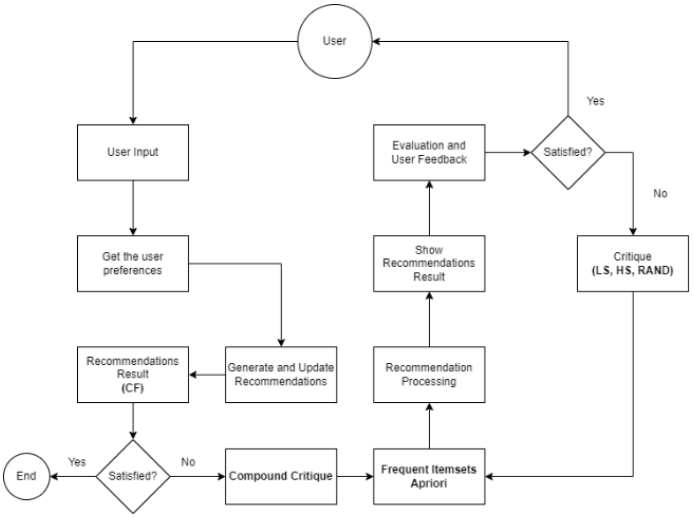

Fig 1. System Design

Fig 1 illustrates the workflow of the conversational recommender system (CRS). First, users provide input in the form of their laptop preferences. The system then identifies these preferences and generates initial recommendations using Collaborative Filtering (CF) techniques [1], [15], which are based on previous user data and similar items. The results of these recommendations are displayed to the user for evaluation. The user then provides feedback on whether they are satisfied with the recommendation or not. If the user is satisfied, the process ends. However, if the user is not satisfied, the user will provide specific critiques based on

DOI: https://doi.org/10.31963/intek.v11i2.4893

laptop attributes such as LS (Low Support) referring to how often a particular critique is applied or selected in the context of existing data, HS (Highly Support) referring to the critique most frequently applied or selected by other users, or RAND (Random) referring to a random critique from the pool of available critiques [8], [9].

These critiques are then processed using the Apriori algorithm to find patterns of frequently occurring itemsets [16], helping the system understand more complex combinations of preferences. Based on this data, the system updates and refines the recommendations. This process is repeated by displaying the updated recommendations to the user. Users can also provide combined critiques, covering multiple attributes at once, such as "I want a lighter laptop and more RAM". These combined critiques allow the system to customize the recommendations more accurately. The system continues to update recommendations based on the combined critiques and itemset patterns found by the Apriori algorithm until the user is satisfied with the recommendations provided.

This flow shows how the conversational recommender system works iteratively to generate recommendations that match the user's preferences.

A. Data Collection

The dataset used in this study was obtained from Kaggle. The obtained laptop data consists of 893 cases of laptop products that have 17 features including brand, name, price, spec_rating, processor, CPU, RAM, type_ram, ROM, type_rom, GPU, size_display, resolution_width, resolution_height, warranty, os.

B. Data Preprocessing

Data preprocessing is performed to form a dataset that is ready to be used in the recommender system. In this study, we remove null values, remove duplicate data, remove unused attributes, separate and remove data units of measure, group data, and change rating data from a 0-100% scale to a 1-5 scale. Then, data splitting was done with a proportion of 20% for test data and 80% for training data. The training data is used by the model to learn the interaction patterns of users with laptop items, while the test data is used to test the model's ability to recommend the right items to users.

C. Recommendation Method

Collaborative Filtering (CF) is a methodology used in recommender systems, where the prediction of preference or rating of an item is based on historical data from users or items that share similar characteristics [15], [17]. In this research, a CF approach combined with singular value decomposition (SVD) algorithm is used to generate initial recommendations. SVD was chosen due to its ability to manage large and small datasets, as well as to efficiently reduce dimensionality, which ultimately results in more precise predictions. Compared to alternative methodologies, such as the standard Matrix Factorization approach, the SVD method showed superior performance in handling sparsity and providing a more precise estimation of user preferences. Other alternatives such as Neural Collaborative Filtering (NCF) offer the advantage of being able to capture complex non-linear interactions between users and items; however, these methods require greater computational resources and are more difficult to implement. The selection of SVD in this study was done by considering its computational efficiency as well as its ability to provide adequate results in the context of the dataset used. To generate rating prediction calculations with the SVD algorithm, the SVD model must be trained using equation (1).

$$R = U \,.\, S \,.\, V^T \tag{1}$$

Description, R: the rating matrix containing the user's rating value of the product, U: the matrix describing the user's relationship with the latent factor, S: the diagonal matrix containing the singular value of the latent factor, V^T: the matrix describing the product's relationship with the latent factor. To calculate the rating prediction of each product after the SVD model is trained, we can use equation (2).

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \tag{2}$$

DOI: https://doi.org/10.31963/intek.v11i2.4893

Description, μ: the global average of all ratings, b_u: user bias u, b_i: item i bias, q_i: item i factor, dan p_u: user factor u.

D. Compound Critiquing

Identify user feedback on recommendations to generate critique patterns that reflect preferences for item attributes. This process compares the basic product features in the current recommendation with alternative products from the case base (CB) [8]. For example, in Table 1, Lenovo laptops with their specifications are compared with Asus laptops, generating critique patterns based on differences in features such as brand, processor, RAM, monitor, and price.

Table 1. Critique Pattern

	Current Case	**Case c from CB**	**Critique Pattern**
Brand	Lenovo	Asus	!=
Processor	Intel Core i5	Intel Core i3	!=
RAM	16	8	<
Monitor	14.0	15.6	>
Memory	512	512	=
Price	5972900	4499000	<

Each product feature has two possible critiques: "<" or ">" for numerical features, and "=" or "!=" for categorical features [8], [9], [14], so there are 2^n possible criticisms for n features. Algorithms such as Apriori are used to extract common patterns from user critiques and find the best patterns. For example, a user may want a cheaper laptop with more RAM, resulting in the critique patterns [RAM >] and [Price <]. This process allows recommender systems to more effectively understand and respond to complex user preferences.

E. Apriori Algorithm

The Apriori algorithm is used to generate critique patterns based on user preferences in conversational recommender system (CRS). One of the important metrics in this algorithm is the support [16], which measures the frequency of occurrence of an itemset in the dataset, we can use equation (3).

$$Support(A) = \frac{Total\ transactions\ containing\ A}{Support(A)} \quad (3)$$

Using equation (3), the system can identify itemsets that co-occur frequently, as well as how significant the relationship between the itemsets is. Another alternative often used in association analysis is the FP-Growth algorithm, which can be more efficient as it does not require exploration of combinations of itemsets that do not meet the minimum support threshold, as done by Apriori. FP-Growth is also faster as it works by building a frequency tree (FP-tree) which allows for faster exploration of frequently occurring itemsets. However, Apriori was chosen in this research because it is easier to understand and implement and has the flexibility to adapt to various datasets. The selection of Apriori was done with the consideration of ease of interpretation and implementation, especially in the context of the data used in this study. This information is then used to customize recommendations based on a combination of critique provided by users, resulting in more relevant and personalized recommendations [18].

III. Results and Discussion

A. Calculation of Rating Prediction

Table 2 is the result of the prediction calculation using the SVD model on laptop rating. This calculation of rating prediction is used to display the initial recommendation results in CRS. Where, when users input their preferences, the system will display initial recommendations based on their preferences and in accordance with the calculated rating predictions.

Table 2. Calculation of Rating Prediction

laptopId	**brand**	**rating**	**predicted_rating**
1	HP	3	2.779112
2	HP	1	2.805067
3	Acer	2	2.964638
4	Lenovo	2	2.808101
5	Apple	2	2.758237
...	...	...	...
889	Asus	2	2.988833
890	Asus	3	3.041729
891	Asus	5	2.927251
892	Asus	3	3.049844
893	Asus	4	2.998949

DOI: https://doi.org/10.31963/intek.v11i2.4893

B. Generating Frequent Itemsets

Table 3 shows the results of generating frequent itemsets using the Apriori algorithm with a min_support value of 0.01. This min_support value ensures that only itemsets that appear in at least 1% of the total transactions are considered as frequent itemsets. By using a low min_support, the algorithm can capture more itemsets, including those that appear infrequently, but are still relevant in a particular context.

Table 3. Frequent Itemsets

support	itemsets
1.000000	(price <)
0.925843	(processor !=)
0.412360	(Ram =)
0.558427	(Ram_type =)
0.710112	(ROM =)
...	...
0.020225	(Ram >, price >, Ram =, processor !=, price <, display_size <, ROM =, Ram_type =)
0.050562	(display_size =, Ram >, price >, Ram =, processor !=, price <, ROM =, Ram_type =)
0.016854	(Ram >, Ram_type !=, price >, Ram =, processor !=, price <, display_size <, ROM =)
0.024719	(display_size =, Ram >, Ram_type !=, price >, Ram =, processor !=, price <, ROM =)
0.015730	(display_size =, Ram >, price >, Ram =, price <, ROM =, Ram_type =, processor =)

In Table 3, some itemsets have varying support values. For example, the itemset (price <) has the highest support of 1.000000, indicating that every transaction in the dataset includes a price condition below a certain value. Meanwhile, itemsets with more complex attribute combinations, such as (display_size =, Ram >, price >, Ram =, price <, ROM =, Ram_type =, processor =), show a lower support value of 0.015730, indicating that they are rarely found together in the data.

Itemsets with high support values (HS) indicate general user preferences, while itemsets with low support values (LS) reveal more specific or unique preferences [8], [9]. By analyzing these frequent itemsets, the system can identify patterns of user preferences and use the information to generate recommendations that are more relevant and in line with user needs.

C. Evaluation

In this research, three critique strategy schemes were tested, namely High Support (HS), Low Support (LS), and Random (RAND). This is done to test the efficiency of each rule using the Apriori algorithm and identify which strategy is the most efficient. At first, the user will enter preferences into the system, which then the system will generate initial recommendations. Next, each scheme will repeatedly display recommendations based on the HS, LS, and RAND critique strategies. Each scheme has a different number of cycles, where a cycle is defined as the process when the recommender system takes a critique and generates a new recommendation. The scheme will continue until the system no longer issues recommendations to the user. In this case, the system will stop displaying recommendations when the filtered user preferences dataframe becomes empty, as there is no more data that matches the user preferences and the tested critique rules.

Based on the test results using the Apriori algorithm, the recommender system generates the average execution time per cycle for each strategy. This research chose to calculate the average time rather than the total because each scheme has a different number of cycles and the focus of this research is to measure which rules are the most efficient in one cycle. The HS strategy had times of 0.0651 seconds, 0.0623 seconds, and 0.0440 seconds, while the LS strategy showed times of 0.2267 seconds, 0.0801 seconds, and 0.0651 seconds. On the other hand, the RAND strategy recorded times of 0.2281 seconds, 0.1168 seconds, 0.0739 seconds, and 0.0832 seconds. These execution times illustrate the efficiency of each strategy in identifying frequent itemsets and generating recommendations.

Frequent itemsets found through the Apriori algorithm are used to generate association rules that guide the HS, LS, and RAND critique strategies selected by the user. Figures 2 and 3 show the results that the HS strategy requires 2 rounds to generate recommendations that match the user's preferences, with an average session duration of 38.01 seconds. The LS strategy, although slightly slower in execution time, also requires 2 rounds with an average session duration of 41.30 seconds. Meanwhile, the RAND strategy requires 4 rounds with a longer average session duration of 50.49 seconds.

DOI: https://doi.org/10.31963/intek.v11i2.4893

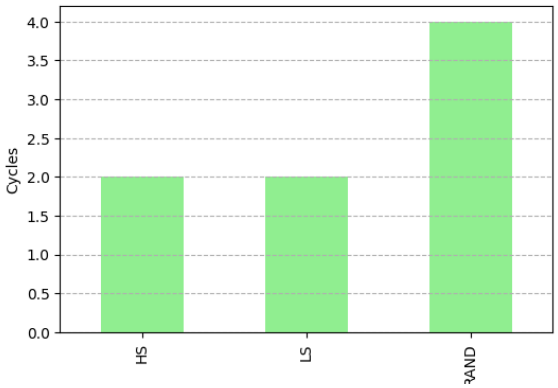

Fig 2. Total Session Cycles

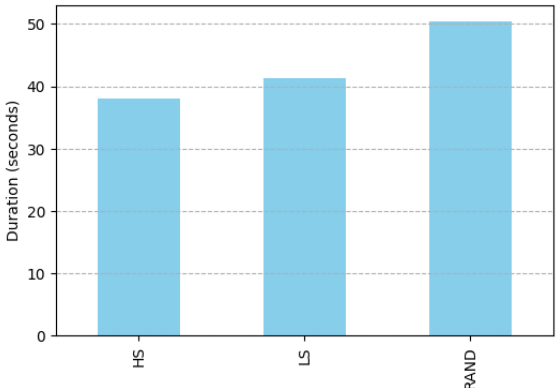

Fig 3. Average Session Duration

Overall, these results indicate that the HS strategy is more efficient in execution than LS, while the RAND strategy shows a relatively slow execution time and requires more rounds, making it less efficient than HS and LS.

D. Discussion

In the development of a Conversational Recommender System (CRS), this study shows that the compound critiquing method with the Apriori algorithm can be used in the recommender system. Previous studies from Smyth et al. show that dynamic critique methods using Low Support (LS) can improve efficiency by reducing the length of recommendation sessions by 40% [8], [14], but this study found that the High Support (HS) strategy is more efficient in terms of average session duration than either the LS or RAND (Random) strategy. Although the compound critiquing method proved to be effective, it is important to note that this study used inputs from real users to obtain their preferences, rather than data from artificial users. The use of data from real users better reflects the complexity and variation of real user behavior, so that the results obtained can be more relevant and accurate.

The HS strategy in the Apriori algorithm has rules with high support values and frequent occurrence of itemsets in most cases. Apriori retains and uses itemsets with high support to find association rules that are stronger and occur more frequently in the dataset. For example, the rule [Ram >, price <], which indicates a preference for larger RAM and lower price, has a support value of 1.0, indicating that this rule is highly relevant in the user context. On the other hand, the LS strategy in Apriori tends to generate rules with the lowest support values, where infrequent itemsets appear, such as the rule [price >, display_size <, processor !=, ROM <], which has a support value of only 0.01011. The RAND strategy selects rules randomly without considering the support value, so it does not show a consistent pattern and is less reliable than the HS and LS strategies.

The analysis of this study shows that the HS strategy is more efficient. It can be seen from the fact that the HS strategy generates rules with high support, which means these rules are more frequent and relevant to users. This supports the argument that the HS strategy allows the system to be more responsive to user preferences, thus speeding up the recommendation process. However, the bias towards using datasets that may not reflect the complexity of the real world is worth noting. Thus, although the HS strategy showed better performance in the tested conditions, it is important to consider further testing with real user data to validate these findings.

IV. Conclusion

A CRS that combines collaborative filtering, compound critiquing, and the Apriori algorithm is shown to improve efficiency and accuracy in understanding users' complex preferences. Collaborative filtering with the Singular Value Decomposition (SVD) algorithm generates rating predictions for initial recommendations, while compound critiquing is used by users to provide simultaneous feedback on multiple attributes. Apriori algorithm then identifies frequent itemset patterns to

DOI: https://doi.org/10.31963/intek.v11i2.4893

update recommendations. In this study, the evaluation results show that the HS strategy is more efficient with an average session of 38.01 seconds, compared to LS 41.30 seconds and RAND 50.49 seconds, although RAND requires more cycles. The integration of compound critiquing and Apriori will provide more personalized and effective recommendations. Future research can explore the implementation of this system on various platforms such as websites and chatbots, for more efficient and accurate interaction.

Acknowledgment

The author would like to thank all those who have supported this research. Especially to the supervisor who has provided guidance and direction as well as valuable advice in the preparation of this research. Without the contribution and support of these various people, this research would not have been realized.

References

[1] D. Amelia Chandra *et al.*, "Penerapan Metode Item Based Collaborative Filtering Berbasis Web Pada Recommender System Laptop," *Engineering And Technology International Journal Juli*, vol. 3, no. 2, pp. 2714–755, 2021, doi: 10.55642/eatij.v3i02.

[2] E. Eli Lavindi and A. Rohmani, "Aplikasi Hybrid Filtering Dan Naïve Bayes Untuk Sistem Rekomendasi Pembelian Laptop Hybrid Filtering and Naïve Bayes Application for Laptop Purchase Recommendation Systems," *Journal of Information System*, vol. 4, no. 1, pp. 54–64, 2019.

[3] A. Iovine, F. Narducci, and G. Semeraro, "Conversational Recommender Systems and natural language:: A study through the ConveRSE framework," *Decis Support Syst*, vol. 131, Apr. 2020, doi: 10.1016/j.dss.2020.113250.

[4] Z. K. Abdurahman Baizal, Y. R. Murti, and Adiwijaya, "Evaluating functional requirements-based compound critiquing on conversational recommender system," 2017 5th Int. Conf. Inf. Commun. Technol. ICoIC7 2017, vol. 0, no. c, 2017, doi: 10.1109/ICoICT.2017.8074656.

[5] Y. Jin, W. Cai, L. Chen, N. N. Htun, and K. Verbert, "MusicBot: Evaluating critiquing-based music recommenders with conversational interaction," in *International Conference on Information and Knowledge Management, Proceedings*, Association for Computing Machinery, Nov. 2019, pp. 951–960. doi: 10.1145/3357384.3357923.

[6] S. Tjayadi and V. C. Mawardi, "Laptop Recommendation Intelligent Virtual Assistant using Recurrent Neural Network with RPA for Data Scraping," 2022 IEEE 7th International Conference on Information Technology and Digital Applications (ICITDA), Yogyakarta, Indonesia, 2022, pp. 1-6, doi: 10.1109/ICITDA55840.2022.9971263.

[7] A. E. Wijaya and D. Alfian, "Sistem Rekomendasi Laptop Menggunakan Collaborative Filtering Dan Content-Based Filtering," *Jurnal Computech & Bisnis*, vol. 12, no. 1, pp. 11–27, 2018.

[8] B. Smyth, L. McGinty, J. Reilly, and K. McCarthy, "Compound critiques for conversational recommender systems," Proc. - IEEE/WIC/ACM Int. Conf. Web Intell. WI 2004, pp. 145–151, 2004, doi: 10.1109/WI.2004.10098.

[9] J. Reilly, K. McCarthy, L. McGinty, and B. Smyth, "Explaining compound critiques," Oct. 2005. doi: 10.1007/s10462-005-4614-8.

[10] A. C. Fatiyah, Z. K. A. Baizal and A. T. Wibowo, "Compound Critiquing for Improving Query Refinement on Conversational Recommender System," 2022 10th International Conference on Information and Communication Technology (ICoICT), Bandung, Indonesia, 2022, pp. 340-345, doi: 10.1109/ICoICT55009.2022.9914858.

[11] C. S. Fatoni, E. Utami, and F. W. Wibowo, "Online Store Product Recommendation System Uses Apriori Method," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Dec. 2018. doi: 10.1088/1742-6596/1140/1/012034.

[12] M. M. Hasan and S. Zaman Mishu, "An Adaptive Method for Mining Frequent Itemsets Based on Apriori and FP Growth Algorithm," 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), Rajshahi, Bangladesh, 2018, pp. 1-4, doi: 10.1109/IC4ME2.2018.8465499.

[13] Y. Djenouri, Z. Habbas, D. Djenouri, and M. Comuzzi, "Diversification heuristics in bees swarm optimization for association rules mining," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Verlag, 2017, pp. 68–78. doi: 10.1007/978-3-319-67274-8_7.

[14] J. Reilly, K. McCarthy, L. McGinty, and B. Smyth. Dynamic Critiquing. In P.A. Gonzalez Calero and P. Funk, editors, Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04). Springer, 2004. Madrid, Spain.

[15] J. L. Herlocker, J. A. Konstan, L. G. Terveen, dan J. T. Riedl, "Evaluating Collaborative Filtering Recommender Systems," ACM Transactions on Information Systems, vol. 22, no. 1, pp. 5-53, Jan. 2004.

[16] M. Al-Maolegi and B. Arkok, "An Improved Apriori Algorithm For Association Rules," *International Journal on Natural Language Computing*, vol. 3, no. 1, pp. 21–29, Feb. 2014, doi: 10.5121/ijnlc.2014.3103.

[17] A. A. Fakhri, Z. K. A. Baizal, and E. B. Setiawan, "Restaurant Recommender System Using User-Based Collaborative Filtering Approach: A Case Study at Bandung Raya Region," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, May 2019. doi: 10.1088/1742-6596/1192/1/012023.

[18] J. Reilly, J. Zhang, L. McGinty, P. Pu, and B. Smyth, "Evaluating compound critiquing recommenders: A real-user study," in *Proceedings of the ACM Conference on Electronic Commerce* (EC'07), San Diego, CA, 2007, pp. 114–123.