Beyond Accuracy: A Comparative Analysis of Recommendation Models Incorporating Quantitative and Qualitative Evaluation

MSc Research Project

MSc. in Artificial Intelligence

Visakh Vijayakumar Nair

Student ID: 23198273

School of Computing

National College of Ireland

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Visakh Vijayakumar Nair
Student ID:	23198273………………………………………………………..……
Programme:	MSc in Artificial Intelligence **Year:** 2024…………………..
Module:	Practicum………………………………………………………….………
Supervisor:	Rejwanul Haque ………………………..………
Submission Due Date:	Dec 12, 2024 ………………………..………
Project Title:	Beyond Accuracy: A Comparative Analysis of Recommendation Models Incorporating Quantitative and Qualitative Evaluation
Word Count:	6,572 **Page Count :** 17 ………..

Signature: ………………………………………………………………………………

Date: 12 Dec, 2024 ……………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Beyond Accuracy: A Comparative Analysis of Recommendation Models Incorporating Quantitative and Qualitative Evaluation

Visakh Vijayakumar Nair

23198273

Abstract – Recommendation systems plays a vital role in content delivery, user accusation and user retention in various domains. This research develops and compares two recommendation models – embedding-based ranking model and a behavioural pattern learning model – using data from the Findups Daily news application. This study evaluates these models on quantitative metrics and evaluation criteria analysis such as diversity, scalability and cold-start problem. This research underscores the importance of aligning recommendation systems with application-specific needs to optimize user engagement and satisfaction.

Keywords – Recommendation system, Content-Based Recommendation, Behavioural Analysis, Embedding, Recommendation Ranking

1. Introduction

Recommendation systems have become integral to modern digital platforms, fundamentally transforming user interaction by personalizing content delivery. These recommendations are widely implemented in e-commerce, streaming services, social media, and news applications, which improve user satisfaction and engagement while increasing business revenue (Ricci, et al., 2022). Usually, to evaluate recommendation system, quantitative metrics such as accuracy, precision, recall, and F1-score, which measure the system's ability to predict user preferences. But in real-world scenarios, some additional evaluation criteria, like system-level metrics, user-centric metrics, and qualitative metrics are equally crucial for ensuring the practicality, scalability, and user satisfaction of a recommendation system (Chen, et al., 2020).

Beyond prediction accuracy, other evaluation dimensions have a major role in the real-world success of recommendation systems. Such dimensions include system-level metrics (scalability and computational efficiency), user-centric metrics (diversity and novelty), and qualitative metrics (explainability). Each of these evaluation criteria addresses unique challenge in deploying recommendation systems at scale.

A. System-Level Metrics

To ensure that a recommendation system can handle a growing user base and data volume without losing performance, the system should be scalable and computationally efficient. As we consider the platforms like Netflix and Amazon, which processes millions of transactions daily, requiring systems that can operate efficiently in real-time. The models, that perform well with high accuracy score in a controlled environment or in testing phase, may falter in large-scale applications due to computational constraints or excessive latency, rendering it is inefficient for real-world deployment (Hunt, et al., 2015). Therefore, scalability and efficiency are important to ensure that a recommendation system remain responsive and cost-effective in large-scale operations.

B. User-Centric Metrics

User satisfaction hinges not only on accurate recommendations but also on the diversity and novelty of the content provided. Diversity ensures that users are exposed to wide range of contents and preventing them from being trapped in a "filter bubble" where they repeatedly encounter similar recommendations. For example, the recommendations system of Netflix is not limited to one genre, instead it encourages viewers to explore wide range of content from different genre and languages. On the other hand, novelty which measures the system's ability to introduce fresh content to the users, which is totally new to them. Music and video streaming platforms like Spotify and YouTube recommend new music and video to maintain user engagement.

C. Qualitative Metrics

Explainability is one of the top qualities of a recommendation system, which provide transparent and understandable recommendations. This metrics is especially important in domains where trust and informed decision-making are essential, such as healthcare, finance, or legal systems. In consumer-focused platforms like e-commerce or entertainment, explainability can enhance user confidence by making recommendations feel more tailored and trustworthy (Zhang, et al., 2022). Regardless of the predictive accuracy, systems that lack explainability may risk alienating users.

To address the diverse challenges faced by recommendation systems, a comprehensive evaluation framework is essential. For instance, in an e-learning platform recommending courses, while high precision and recall might suggest relevant courses, but low diversity could lead to repetitive suggestions, limiting the user's ability to explore new topics. Moreover, if a system cannot accommodate new users or product, or fail to explain its recommendations, it may lose credibility and usability. This scenario illustrates the need to balance predictive accuracy with broad evaluation criteria to optimize user experience and system practicality (Jannach, et al., 2010).

By considering these facts, it is understandable that, different domains prioritize evaluation criteria based on their requirements. In personalized healthcare, explainability and trustworthiness are important for ethical decision-making. In fast-paced environments like news applications, scalability and real-time efficiency are essential for delivering timely recommendations. But the social media platforms are focused on diversity and novelty to improve user engagement and retention and avoiding content fatigue (Tang et al., 2015). By including these broader evaluation criteria, recommendation systems can better address the unique challenges of various applications.

In this project, two different recommendation models - The Embedding-Based Ranking Model and Behavioral Pattern Learning Model – will be developed and evaluate those models based on quantitative metrics and other evaluation criteria. To complete this study, the dataset from the Findups Daily (daily news application with more than thousand users) will be utilized. The aim of this project to understand the working of two recommendation model with different approach and how its performance impacts user experience, system efficiency, and overall effectiveness in delivery relevant recommendations.

2. Related Work

In the research (Javaji & Sarode, 2023), introduces a combined approach of Graph Neural Networks (GNN) and BERT embeddings to create better recommendation systems. To understand the relationships and interactions between users and items, they utilized GNN which is an easier method to identify patterns in data. They used shared preferences to connect user information. At the same time, BERT is implemented to analyze and extract meaningful insights from the text, such as product descriptions and user reviews. By implementing both these technologies, the system can predict more accurate and personalized recommendations, even when dealing with challenges like insufficient data about new users or items. This approach is useful especially when application is required to handle different types of data structures. But this system has some drawbacks also. The model requires huge computational resources to solve highly complex graph structure and running transformer-based models simultaneously. The second problem is that the system heavily depends on high-quality and diverse data. Poorly structured graph data or noisy textual content can reduce the effectiveness of the hybrid model, leading to suboptimal recommendations.

A similar study conducted by (Lin, et al., 2022) explains a new way to improve recommendation framework by fusing two methods – Collaborating Filtering that uses structured data from knowledge graphs and Content-Based Filtering that uses unstructured data like text descriptions. The researchers developed two models – the Knowledge-Graph Enhanced Meta-Preference Network (KMPN) and a variant of NRMS, named it as NRMS-BERT, which implemented pre-trained BERT for encoding item descriptions. The fusion framework allows these models to exchange information effectively, addressing inconsistencies between in two systems. To complete the research, they used the Amazon-Book-Extended and the Movie-KG-Dataset, which are enriched with textual descriptions to facilitate content-based learning. The results demonstrated that the combine approach provided better accuracy and relevance of recommendations, especially at cold-start scenarios. But at the same time, the framework requires high computational requirement for training and depends on clean, high-quality data.

In another study (Wu, et al., 2021)explains a different approach which enhance personalized news recommendation by using pre-trained language models (PLMs) like BERT, RoBERTa, and UniLM. While traditional approaches that relay on text representation, the researchers proposed a framework which utilized PLMs to build a standard recommendation pipeline, enabling the capture of deep semantic contexts from news content. This approach includes finetuning PLMs for news encoding and using an attention mechanism to pool contextual embeddings and candidate news representations. The system is trained using datasets like MIND, containing six weeks of click logs from 1 million users, and a multilingual MSN dataset spanning seven languages. The model shows a significant performance improvement in both offline and online evaluations. By deploying it on Microsoft News, the PLM-based models achieved up to 10.68% more clicks and 6.04% more pageviews globally. Even though, the approaches face some challenges, such as computational overhead due to the size of PLMs and the need for high-quality datasets to ensure effectiveness.

In Jeong, et al. (2019), proposed a model that combined BERT and Graph Convolutional Networks (GCN). The BERT is used for understanding textual context and GCN is for leveraging citation relationships to recommend scholarly papers. The proposed methodology utilizes two-part encoders – to extract semantic embeddings from surrounding text of citation placeholder, they use a context encoder which is powered by BERT and a citation encoder employing GCNs to model relationships in citation graphs. Then these encoded outputs are combined and fed into a feedforward neural network to predict appropriate citations. The proposed method showed an improvement by over 28% Mean Average Precision (MAP) and recall metrics, showcasing the power of combining deep language modeling with citation graph analysis. The main drawback of this model is that it is computationally expensive and heavily dependent on well-processed datasets. To study points out how implementing textual and graph-based insights can significantly enhance citation recommendation systems, which helps researchers to locating relevant references efficiently.

Zhang, et al. (2024), presents a technology called MP4SR to enhance recommendation systems that struggle with limited user data. The authors implemented a technique called multimodal pre-trained to combine information from different sources, like image and text. To process item features and a Transformer-based encoder to analyze user behavior sequences, the model uses pre-trained tools like Sentence-BERT and CLIP. To complete this study, they tested MP4SR on multiple datasets like Amazon's Pantry, Arts, Office, and Goodreads' Poetry and showed better results than other methods. Even though this approach is comparatively better in handling sparse data and provides strong recommendation, but it is a complex method to implement and requires significant computing power. The main strength is it can build a recommendation system using limited data, but it may face challenges in working with very large datasets.

A novel model is proposed in (Shang, et al., 2019), which fused Graph Neural Networks (GNN) and BERT to build a recommendation system in medication by leveraging hierarchical medical knowledge and single-visit electronic health record (HER) data. G-BERT enhances medical code representations using graph attention networks to embed medical ontology and integrates these embeddings into a Transformer based visit encoder. The visit encoder is pre-trained on single-visit data, which is usually decarded by other models, and fine-tuned on multi-visit longitudinal EHR data for prediction tasks. The model was tested on MIMIC-III EHR dataset, which outperformed other baseline models like RETAIN and GAMENET in accuracy metrics such as F1-Score and Jaccard similarity. This model has one main drawback, it is difficult to scale the model to more diverse datasets.

By leveraging the bidirectional nature of self-attention, inspired by BERT, (Sun, et al., 2019) introduced a model for sequential recommendations. Traditional sequential models process user interaction data in a left-to-right manner. But BERT4Rec employs the Cloze task to mask random items in a sequence and predict them using both left and right context, which improves representation quality. For the context-aware modeling of user behavior and incorporating positional embeddings to maintain sequence order, they stacked bidirectional Transformer layers. They used datasets like Amazon Beauty, Steam, and MovieLens to train the model. The results show an effective through performance enhancement in metrics such as HR, NDCG, and MRR. The model faces challenges like rigid order assumption of prior unidirectional models and ensures efficient training with scalable Cloze objectives. The main drawback of this approach is the high computational complexity due to the quadratic scaling of self-attention mechanisms with sequence length. But it brings some merits such as improved performance over state-of-the-art methods, as BERT4Rec captures bidirectional dependencies and adapts effectively to varying sequence lengths.

Xu et al., (2023), the authors have done an extensive review on various deep learning techniques applied to recommendation systems – content analysis and categorizing, collaborative filtering, and hybrid approaches. Various architectures are explained in this survey – neural collaborative filtering, autoencoders, recurrent neural networks, convolutional neural networks, and attention mechanisms. The authors explained how each model is handling different challenges like scalability, cold-start problems, and dynamic user preferences. Mainly, the paper is focused on the requirement of embedding techniques for feature representation and explores the integration of contextual information, such as user-item interactions and temporal dynamics. For the study, the authors used MovieLens and Amazon datasets to evaluate and compares the performance of deep learning models against traditional algorithms like matrix factorization. The survey demonstrates the potential of deep learning to capture complex patterns and improve recommendations, but it also highlights drawbacks of models such as integrity issues and sensitivities to hyperparameter tuning.

In another study (Zhang, et al., 2022), the authors introduce a framework which utilizes pre-trained Transformers for conversational recommendation systems (CRS). They combined a Knowledge Graph with Transformers to improve both item recommendation and conversational responses. The approach features a Bag-of-Entities (BOE) loss that encourages the system to generate contextually relevant dialogue by embedding item-related concepts, such as genre or attributes, into the conversation. Furthermore, the model ties an alignment loss to connect word embedding created by the Transformer with embeddings from the knowledge graph. This makes sure that the textual representations from the Transformer and the structural knowledge from the knowledge graph work well together which allow the system to better understand and generate recommendations based on both sources.

To understanding the impact of Large Language Models (LLM) in revolutionizing the recommendation systems, some researchers conducted a study (Vats, et al., 2024). Usually, traditional methods required a predefined data about user preferences. But LLMs use task-specific prompts to understand user needs and make recommendations directly. The authors state that these models show amazing performance in understanding context and adapting to different situations, that makes them very handy for personalized and interactive recommendations. The authors explain systems like LlamaRec and RecMind, which utilizes LLMs to enhance recommendations in areas like shopping, movies, and online conversations. They used datasets from MovieLens and Amazon Kindle and tested different methods on these datasets, where LLMs performed better than traditional systems by understanding the context better and making accurate predictions. Nevertheless, the LLMs are so sensitive to the way questions are asked, and sometimes provides incorrect suggestions.

3. Research Methodology

In this research, we will conduct a comprehensive analysis of two recommendation systems by evaluating their quantitative performance and other evaluation criteria. Two different recommendation model build on different approach is developed – Behavioral Pattern Learning Approach and Embedding-Based Ranking Approach - to compare the performance and working of them. The process begins with exploring and understanding the dataset, followed by applying data processing and feature engineering to extract meaningful features for training. Each model approach demands a unique dataset transformation, and we ensure that the input format aligns with the specific requirements of each model.

3.1. Dataset

This study utilizes a dataset from Findups Club Private Limited, which collected from their news application, Findups Daily. The data was stored in MongoDb database, and it contains three main collections.

News Collection: This collection contains the information about the news articles. Each article contains a unique identifier id (*news_id*), along with its title (*title*) and the news description (*description*).

User Collection: This collection contains details specific to users. It includes a unique identifier for each user (*user_id*) and the date and time when the user started to use the app (*createdOn*).

User Read Log Collection: This collection contains the interactions between users and news articles. It contains a unique log identifier (_id), the ID of the news article viewed (news_id), the ID of the user who viewed it (*user_id*), and the timestamp of the interaction (*time_stamp*). Then if the user read that news, they update the (*is_read*) variable as True, otherwise it will set as False. This data is crucial for tracking user behavior and creating models that predict future user preferences.

The Findups Daily app functions in a way by initially it displays news titles accompanied by cover images to users. As the user scroll through the app, they can see as much news headlines as they need. And if they find any interesting article, they can open up the news and read by clicking on it.

The app records user interactions in a detailed manner. When a user scrolls past a news headline, the system logs the interaction by saving the *user_id*, *news_id*, and the *time_stamp*. At the same time, the *is_read* field is marked as False, indicating that the user has not yet opened the article. If the user decides to read the article, the system updates the *is_read* status to True, reflecting that the article was fully accessed by the user. This structure tracking mechanism provides rich interaction data, which is crucial for building and evaluating recommendation systems tailored to user preferences.

For this research, the dataset is merged into a single comprehensive dataset named *read_logs*. This one single combined dataset contains 84,588 entries, which provides a clear structure of user interactions, enabling the development of accurate recommendation models. Each entry in the *read_logs* dataset includes the following fields:

- *news_id*: The Unique identifier of the news article.
- *user_id*: The Unique identifier of the user.
- *news_content*: The full text or description of the news article.
- *time_stamp*: The timestamp when the user saw the news article.
- *is_read*: A binary value (True/False) indicating whether the user interacted with or read the article.

3.2. Data Preprocessing

Data preprocessing is a crucial step because a major portion of the dataset contains textual content, which cannot be fed directly to a machine learning model. The preprocessing pipeline for this project is designed to process the text-heavy nature of news data and the sequential structure of user interactions.

3.2.1 Data Cleaning

From the raw dataset which consisted of 3 MongoDb collections – News Collection, User Collections, and User Read Log, it is merged into a single dataset called *read_logs*. After carefully analyses the data, using pandas, identified there are some missing values and duplicate entries in the dataset. So, the following cleaning operations were applied on it.

- Handling Missing Values: Records with missing critical fields such as *news_id*, *user_id*, or *time_stamp* were removed to ensure data integrity.
- Deduplication: Duplicate entries in the *read_logs* were identified and removed. Duplicate records could arise from repeated user interactions logged erroneously.
- Timestamp Parsing: The *time_stamp* field was converted into a standardized datetime format to enable temporal sorting of user interactions.

These cleaning steps resulted in a consistent dataset with 84,588 valid entries.

3.2.2. Embedding Generation

Most of the data is in text format, so it needs to be converted into numerical form before using it in a neural network model. To accomplish this, a pre-trained BERT model can be used to create embeddings, which are dense vectors that represent the meaning of text in numbers. These embeddings help the model to understand the content and relationships in the text.

Tokenization: The initial step in an embedding process is tokenizing the text data, which means breaking down the text into smaller parts, and called it as token. These tokens include words, sub words, or individual characters, depending on the tokenizers' rule. In this project, we use BERT tokenizer.

Special tokens such as [CLS] (start of a sequence) and [SEP] (end of a segment) were appended to each sequence. These special tokens enhance BERT's ability to contextualize the content, particularly for tasks requiring a holistic understanding of the text. By padding shorter sequence and trimming longer ones in the tokenization process, it ensures uniforms sequence lengths.

Embedding Extraction: Once tokenized, the sequences were fed into a pre-trained BERT model to generate embeddings. As an output, the BERT model provides a 768-dimensional vector-was extracted for each sequence. This vector captured the main meaning and context of the news content. The key benefits in using a pre-trained BERT model – BERT is trained on a huge amount of text, so it understands word meanings and their relationships very well.

Embedding Integration: After calculating the embeddings of each news content, it is added as a news column in the dataset, making them part of each news entry. This allowed the embeddings to be used directly in building recommendation models without needing further processing.

There are multiple reasons behind in choosing BERT embeddings.

- BERT understands the meaning of words based on their context in a sentence.
- Representing text as fixed-size vectors makes it easier to process large datasets in machine learning.
- BERT can handle semantic structures which ensures robustness, even in cases where the news content contains non-standard expressions.

3.2.3 Feature Engineering

In this project, we will build a recommendation system which can predict the next news a user will read based on news they read before. For this project, feature engineering is focused on creating a sequential dataset that captures user behavior patterns and interactions with new articles.

Sequential Dataset Construction

To model user preferences effectively, it is required to record the sequential nature of their interactions with news content. A new dataset, user_sequence_entries, is created to capture these temporal patterns. The following steps are applied to build the dataset:

1. Chronological Sorting: The user interactions logs are sorted based on the *time_stamp* field to make sure that the reflected the natural progression.

2. Sequence Generation: For each user, group of four news articles were formed. The first three articles represented the user's recent interactions, while the fourth was the target article.
3. Feature Representation:
   a. The embeddings of the four previously read articles were concatenated to form a 3072-dimensional vector (768 dimensions per article).
   b. The embedding of the target article was appended to this vector, resulting in a total feature size of 3840 dimensions.
4. Labeling: A new column, *is_read*, is added which holds a binary label, indicating whether the user read the target article (True) or ignored it (False). This label was used as the target variable for classification during model training.

Now, the resulting dataset, user_sequence_entries, contains 3840-dimensional feature vector representing user behavior and target article characteristics. And followed by a binary target variable indicating user engagement with the target article.

4. Behavioral Pattern Learning Model

The aim of the Behavioral Pattern Learning Model is to identify user behavioral patterns from a structured data, and recommend content based on that. A neural network is developed to predict user engagement by learning from the embeddings of previous interacted content. The model leverages dense layers to capture non-linear relationships and employs activation functions like ReLU to introduce non-linearity.

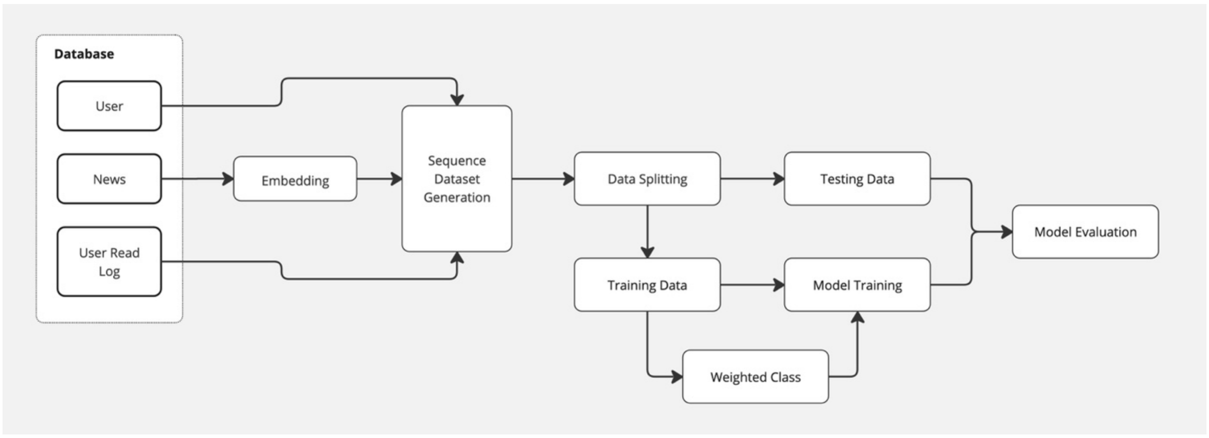

figure 4. 1: The working mechanism of Behavioural Pattern Leaning Model.

4.1. Model Design

To implement a Behavioral Pattern Learning Model, an efficient, but simple, architecture is designed to process structured data and learn user interaction patterns. The architecture is finalized by continuously testing various parameters and sequential data structure.

The model's architecture is refined by experimenting with experimenting the number of articles considered and adjusting hidden layers.

By iterating on the number of articles consider, it is identified that as the number of articles considered is increased, the recall value has increased. Specifically, using embeddings of 2 news articles and 1 target article resulted in a recall of 21.3%, which increased to 32.3% with 3 news articles and 1 target, and further improved 36.7% with 4 news articles and 1 target. This highlights that incorporating more historical interactions allows the model to better capture user behavior patterns.

Furthermore, by adjusting the number of hidden layers and neurons showcased that increasing the complexity of the network improved accuracy. The final architecture was selected to strike a balance between computational efficiency and performance, ensuring robust learning without overfitting.

By considering all these values, the final architecture is confirmed. The architecture includes with 1 input layer with 3840 input neurons, 3 hidden layers with ReLU as activation function and output layer with one neuron with sigmoid as activation function. The neural network is implemented using TensorFlow library in Python.

Number of Neural Network Layers	*Number of News Articles*	*Number of Target Articles*	*Total Entries*	*True Values*	*False Values*	*Vector Dimension*	*Accuracy*	*Recall*
4	2	1	85547	3842	81705	2304	78.6%	25.1%
4	3	1	84588	3777	80811	3072	79.9%	28.2%
4	4	1	83674	3720	79954	3840	72.2%	29.3%
4	5	1	82800	3660	79140	4608	82.7%	15.3%
4	6	1	81960	3620	78340	5378	4.3%	100%
4	7	1	81155	3578	77577	6145	4.7%	100%
5	2	1	85547	3842	81705	2304	23.5%	47.1%
5	3	1	84588	3777	80811	3072	34.4%	62.85%
5	**4**	**1**	**83674**	**3720**	**79954**	**3840**	**58.3%**	**73%**
5	5	1	82800	3660	79140	4608	65.1%	39.2%
5	6	1	81960	3620	78340	5378	86.5%	17.7%
5	7	1	81155	3578	77577	6145	83.1%	13.4%

Table 4. 1: Variation of quantitative performance over sequence dataset and neural network layers.

4.2. Model Compilation

The model is compiled using Adaptive Moment Estimation (ADAM) Optimizer, which suits well in handling large datasets and dynamic learning rates. The binary cross-entropy loss function was chosen as it is ideal for binary classification tasks, enabling the model to calculate the difference between predicted probabilities and actual labels effectively. Additionally, the evaluation metrics included accuracy to measure the overall correctness of predictions and recall emphasizing the model's ability to identify true positive instances, ensuring a focus on user engagement.

Why ADAM Optimizer? ADAM was chosen for its ability to handle sparse gradients and adapt learning rates during training. It combines the strengths of momentum optimization and RMSProp, making it well-suited for high-dimensional data like embeddings.

Why Binary Cross-Entropy Loss? Binary cross-entropy was used because the target variable is binary. This loss function minimizes the difference between predicted probabilities and true labels, making it ideal for classification tasks.

4.3. Initial Evaluation

At the initial evaluation, the model showcased an accuracy of 98.9%. But the recall is only 3.6% which indicates the poor performance of model in identifying positive cases (True labels).

Why Recall is Important? For recommendation systems, recall is a crucial metrics because it measures the system's ability to correctly identify relevant articles that users would engage with. High recall ensures that the system suggests a significant portion of the articles users are interested in, which is vital for user satisfaction and engagement.

It is more important to identify the right articles than to avoid recommending irrelevant ones because for this project we consider the users will be professionals who wants to improve their career by reading articles. If they miss relevant content or important news that will affect their professional life and eventually users lose interest. A system with high recall ensures users have access to variety of contents which find appealing to them, even if it occasionally includes less relevant content. This approach aligns with the primary objective of recommendation systems: maximizing user engagement by prioritizing relevance.

4.4. Implementing Class Weights.

By checking the data, it is identified that the dataset is biased. The target labels of 80,811 out of 84,588 entries in the dataset is marked as False, which causes the poor recall. This imbalance caused the model to favor predicting the majority class.

To address the bias, class weights are introduced. Class weights assign higher importance to under-represented labels (in this case, False) during training. By applying this, the model learns to consider these cases as equally important as the majority class. The class weights were computed dynamically based on the label distribution, ensuring proportional adjustments.

4.5. Final Evaluation

After applying class weights, the model showed a significant improvement in performance.

Accuracy: 58.3%

Recall: 73%

From this result with higher recall and accuracy, it indicates that the model can effectively identify a higher proportion of relevant articles. Although accuracy slightly decreased, this trade-off was acceptable because recall is more critical in this context.

The implementation process successfully addressed the challenges in class imbalance and low recall. By applying class weight, the model demonstrated a significant improvement in recall, which making it a more effective recommendation system for capturing articles users are likely to engage with. With the use of sequential data, class weights, and a carefully designed neural network, this project establish a robust foundation for recommending personalized content.

5. Embedding-Based Ranking Model

To compare the advanced recommendation model using Transformer, BERT, and Neural Network, we need to build a baseline content-based recommendation model. The Embedding-Based Ranking Model leverages the semantic similarity between news articles and user interests to provide personalized recommendations. We will discuss about the concept; steps taken to design and implement this approach.

5.1. Concept and Working Mechanism

The Embedding-Based Ranking Model focuses on understanding the embedding value of the news content and align them with the user's preferences. Using the embedding, the model can transfer the textual data into numerical representations, the model identifies user interest and ranks articles. To understand user interest, the system creates a profile of user based on their interaction history and comparing it to the semantic profiles of other articles. By calculating and ranking the similarity between the user's preferences and the available articles, the system recommends content most likely to engage the user.

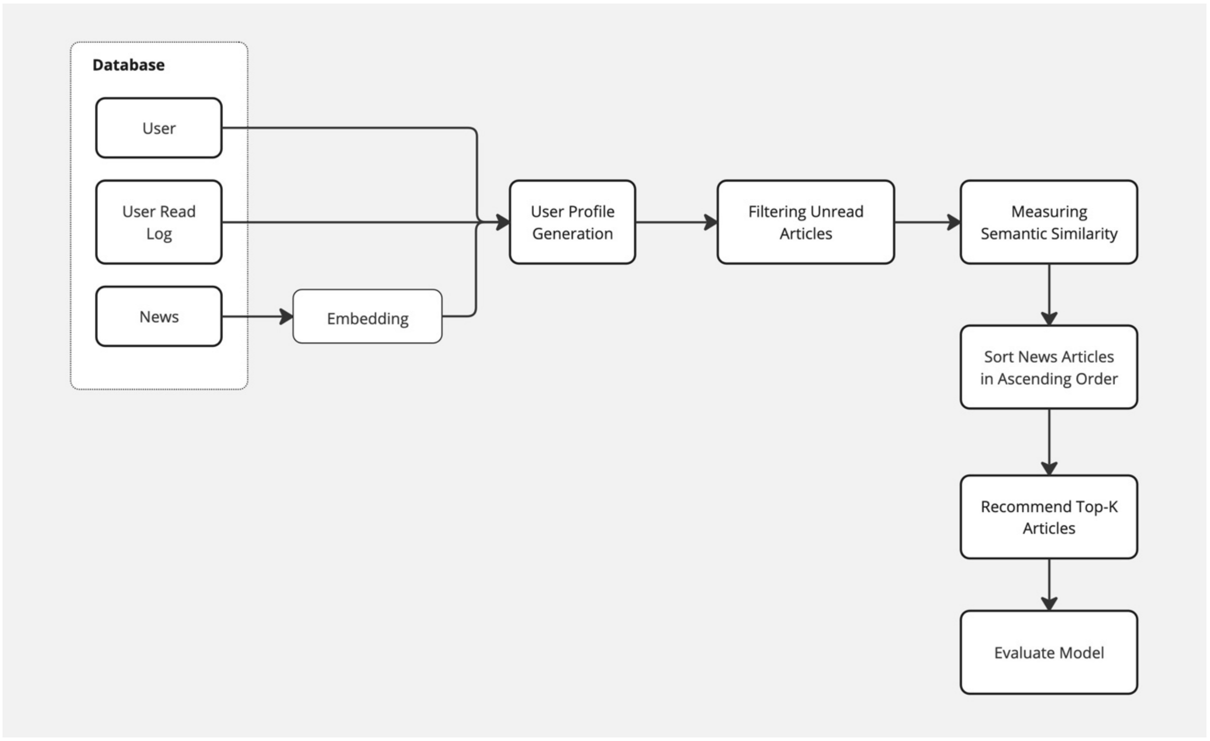

Figure 5. 1: Working Mechanism of Embedding-Based Ranking Model.

5.2. Key Steps in Embedding-Based Ranking Model

Step 1: Generating News Embedding

The initial step of this process is creating embeddings for each news article in the dataset. An embedding is a high-dimensional numerical vector that encodes that encodes the semantic meaning of a text. Even we have options like using bag-of-words or TF-IDF to embed the articles, but we use pre-trained BERT model for the same. Because the models often represent text data using bag-of-words or TF-IDF, which are inefficient to capture the semantic nuances and relationship between words. At the same time, embeddings generated by pre-trained models like BERT or Transformer-based models provide a meaningful representation of text, and it preserves contextual relationships between words.

To generate embedding for each news article, the description was tokenized using BERT's tokenizer, which break the text into sub-word units suitable for processing by the embedding model. To extract embedding, the tokenized data will be passed through the pre-trained BERT model to extract embeddings.

The embedding will contain 768-dimesional vector, were stored in a new column within the news dataset. This enables efficient retrieval and will reduce the need to recompute embeddings for the same articles during recommendations.

Step 2: Identifying User Interest Embedding

The next step is to determine a user's preferences by calculating their interest embedding. This embedding acts as a summary of the use's engagement history, capturing their overall tastes and thematic inclinations.

The system identifies all the articles that user read, specifically those marked as is_read = True. This filtered list represented the set of articles that interested the user. By focusing on explicitly engaged articles, the model ensured the interest embedding was derived from meaningful interactions rather than implicit signals.

To calculate the interested embedding, the embeddings of the filtered articles were aggregated. The mean average embedding of user interest is calculated by summing the vectors of all read articles and dividing by the number of articles. This average embedding acted as a semantic representation of the user's preferences, capturing recurring themes and patterns in the user's reading habits.

For instance, if a user interacts with news related SpaceX, Tesla, Nvidia, their interested embedding will reflect topics and keywords related to technology, innovation, trends, and tech-related events.

Step 3: Filtering Unread Articles.

To ensure the recommendation system provide novel suggestions, the system needs to filter out articles that the user had already engaged with. This step reduced redundancy in the recommendation process and emphasize discovery. For this task, we can utilize the *read_logs* dataset and it is queried to identify all articles the user had already viewed. These articles are removed from the pool of potential recommendation. The remaining articles formed the candidate set for recommendations, ensuring the user only receive suggestions for new, relevant content.

Step 4: Measuring Semantic Similarity

At this stage, the system needs to identify the relevant content for the user. The system evaluates their relevance to the user by comparing their embeddings to the user's interest embedding.

To measure the distance between the article embedding and user interested embedding, the cosine similarity is implemented. Cosine similarity measures the angle between two vectors, providing a normalized measure of their similarity. High cosine similarity indicates strong alignment between the user's interests and the article's content.

To rank each article based on relevance, at first the semantic distance between the user's interest embedding and each candidate article embedding is computed. Then the articles will be sorted by descending order of distance. This ranking mechanism ensured that article most closely aligned with the user's interests appeared at the top of the recommendation list.

Step 5: Recommending Top-K Articles

At this final stage, the system can simply identify the most relevant top-k news articles by selecting the first k articles from the sorted list.

The top-ranked articles were presented as recommendations, with their high semantic similarity to the user's interest embedding ensuring a strong likelihood of engagement. The value of k can be adjusted based on user preferences or application requirements.

Step 6: Model Evaluation

To validate the performance of the model in quantitative, here we are using Top-K Recall method. Top-K Recall measures the system's ability to retrieve relevant items from a ranked list of recommendations. In this model, it assesses whether the news articles that a user would likely engage with are included within the top-k recommended items.

There are numerous reasons which make Top-K Recall metrics more suitable for this project. It emphasizes the inclusion of relevant articles rather than penalizing irrelevant ones, which aligns with the goal of ensuring user satisfaction. Users usually interact with a limited number of recommendations displayed at a time. Measuring recall for varying values of K reflects the model's performance under different realistic scenarios (e.g., a user scrolling through 5, 10, or 20 articles). At the same time, our model suggest content based on a ranked list of recommendation based on embedding similarity, Top-K Recall directly evaluated how well the ranking matches user preferences.

5.3. Performance Analysis

From the evaluation of the model using Top-K Recall revealed one major insight that as we increase the value of K – the number of recommendations considered – the recall values improved. This indicates the model's ability to include more relevant articles within a larger set of recommendations. However, it also demonstrates the diminishing returns of increasing K beyond a certain point.

K-Value	Recall
5	16%
10	27%
20	32%
30	43%
40	39%

6. Evaluation

6.1. Quantitative Performance Analysis

When comparing the two models based on the recall evaluation metrics, the Behavioral Pattern Learning Model outperforms the content-based recommendation system in terms of recall, achieving a recall of 73% compared to the content-based model's Top-K Recall of 43% at K=30. This shows that the Behavioral Pattern Learning Model is more effective at identifying relevant articles within a user's feed.

6.2. Qualitative Analysis

Recall is only one aspect of a recommendation system's effectiveness. But the factors like diversity, cold start handling, and the ability to cater to multiple topics play a crucial role. These factors ensures users are exposed to variety of content and keeping their engagement high. One of the major challenge for a recommendation system is cold start handling, which evaluate how a model provide content to a new users. Additionally, it is essential for platforms like news app to handle multiple topics of contents, where users often have interests spanning various domains. Balancing these factors will help to create a recommendation system that is not only accurate but also adaptable and user-friendly.

a. Personalization

Personalization refers to how well the system recommend content which is relevant to individual users. The Embedding-Based Ranking model directly utilize the user's history to generate personalized recommendations, while the Behavioral Patter Learning model relies on learned patterns to predict user behavior. Here, the embedding-based ranking model excels at personalization by focusing exclusively on user-specific preferences.

b. Cold Start Problem

When the system has limited data about a user or item, there occurs the cold start problem, which the system need to find out to recommend without much data. Systems with cold start problem fail to recommend meaningful content for new users which will impact user satisfaction. Both models have failed to solve this issue. But comparatively, embedding-based ranking model requires only very less data (even one entry of user interacted news article), but the behavioral pattern learning model requires at least 4 entries (because the model predict weather user read the 5th news by considering the 4 other news that the user read before).

c. Diversity of Recommendations

Diversity of Recommendation measures the variety of content provided to the user. Users like to explore variety of content, but it should be interesting to the user. Limited diversity can lead to user fatigue and reduced engagement, as the user may feel the system is too repetitive. The embedding-based ranking model recommends similar articles repeatedly because of its focus on semantic similarity. But behavioral pattern learning model can identify diverse patterns-based on user interactions.

d. Computational Cost

Computational cost means the time and resources required for training and generating recommendations. Higher computational cost during training will limit scalability, while high inference costs can slow-down real-time recommendations. The behavioral pattern learning model requires more resource-intensive training process but is efficient during inference. Meanwhile, as comparing with behavioral pattern learning model, embedding-based ranking model have lightweight design during inference, which makes it more suitable for real-time systems with limited computational resources.

e. Explainability

Explainability refers to how easily the process of recommendation can be understood. The systems with explainable mechanism or working methods build user trust and provide actionable insights for improving recommendations. The embedding-based ranking model, which recommend content by providing news articles which have similar embeddings of user interests, establishes a clear rationale for recommendations. In contrast, the behavioral pattern learning model's complex structure makes its decision-making process less transparent, which fails to clearly explain why a user is provided those contents.

f. Scalability

Scalability means how well a recommendation model adapts the change when a new user or article enters the system. Poor scalability can affect the system's performance negatively as the dataset grows, which affects user experience. The embedding-based ranking model needs only embedding of new article or recalculate user interest embedding, which makes the model more scalable. But the behavioral pattern learning model may need retraining with additional data.

With this above analysis, it is clear that both models have strengths, their suitability depends on the specific requirements of the system. The embedding-based ranking model excels in personalization, explainability, scalability, and low computational cost, making it ideal for systems prioritizing transparency and efficiency. On the other hand, behavioral pattern learning model outperforms in recall, and delivering diverse recommendations, making it better suited for dynamic and complex applications.

7. Future Work

The findings of this research highlight the potential for further advancements in recommendation systems. In future work, involving applying these models to larger and more diverse datasets, which might lead to improvements in both accuracy and recall by capturing nuanced user behavior and content features.

Each model has its own strength and unique applications in real-world scenarios. The embedding-based ranking model, with its explainability and computational efficiency, is ideal for domains which requires transparent decision-making and lightweight operations, such as news platform, educational content delivery, or job portals. Conversely, behavioral pattern learning model have the ability in capturing user behavior over time, which suits well in platforms like personalized social media, video streaming platforms, and e-commerce websites, where user behavior changes dynamically over time.

In future research, exploring the hybridization of these models could unlock new possibilities, combining the personalization and explainability of content-based methods with behavioral adaptability of sequential models.

8. Conclusion

This research explores the development and comparison of two recommendation systems models – an embedding-based ranking approach and a behavioral pattern leaning approach. To complete this study, a dataset from the Findups Daily news application is utilized. The embedding-based ranking model focus on semantic similarities by leveraging precomputed embeddings, which provides highly personalized and low computation required recommendation mechanism. In contrast, the behavioral pattern leaning model uses patterns from user behavior to predict future behavior, which provides better performance in recall.

By analyzing both models and comparing their performance, it is identified that the embedding-based ranking approach have better ability in explainability, scalability, and it requires low computational resources, making it ideal for transparent and resource-constrained systems. On the other hand, the behavioral pattern learning approach model showcased its strength in recall, diversity of recommendations, and adaptability, which indicates that its suits to be a better choice for dynamic environments with frequent changes in user behavior.

Bibliography

Hunt, N., G.-U. & A, C., 2015. *The Netflix recommender system: Algorithms, business value, and innovation,* s.l.: ACM Transactions on Management Information Systems.

Ricci, F., Rokach, L. & Shapira, B., 2022. *Recommender systems: Challenges and opportunities. In Recommender systems handbook.* 3rd Edition ed. s.l.:s.n.

Chen, J. et al., 2020. *Bias and debias in recommender system: A survey and future directions.* s.l., 29th ACM International Conference on Information & Knowledge Management.

Zhang, T. et al., 2022. *Toward Knowledge-Enriched Conversational Recommendation Systems,* s.l.: s.n.

Jannach, D., Zanker, M. & Felfernig, A., 2010. *Recommender systems: An introduction,* s.l.: Cambridge University Press.

Javaji, S. R. & Sarode, K., 2023. *Hybrid Recommendation System using Graph Neural Network and BERT Embeddings.* s.l.:s.n.

Lin, W. et al., 2022. *Transformer-Empowered Content-Aware Collaborative Filtering,* s.l.: s.n.

Wu, C., Wu, F., Qi, T. & Huang, Y., 2021. *Empowering News Recommendation with Pre-trained Language Models,* s.l.: s.n.

Jeong, C. et al., 2019. *A Context-Aware Citation Recommendation Model with BERT and Graph Convolutional Networks,* s.l.: s.n.

Zhang, L., Zhou, X., Zeng, Z. & Shen, Z., 2024. *Multimodal Pre-training Framework for Sequential Recommendation via Contrastive Learnin,* s.l.: s.n.

Shang, J., Ma, T., Xiao, C. & Sun, J., 2019. *Pre-training of Graph Augmented Transformers for Medication Recommendation,* s.l.: s.n.

Sun, F. et al., 2019. *BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer,* s.l.: s.n.

Xu, Y.-H.et al., 2023. *A recommendation algorithm based on a self-supervised learning pretrain transformer,* s.l.: s.n.

Vats, A., Jain, V., Raja, R. & Chadha, A., 2024. *Exploring the Impact of Large Language Models on Recommender Systems: An Extensive Review,* s.l.: s.n.