QUBAHAN ACADEMIC JOURNAL
VOL. 5, NO. 4, December 2025
https://doi.org/10.48161/qaj.v5n4a2141

Enhancing Recommendation Systems with Autoencoder-SVD and Transformer-Based Summarization: A Sentiment-Aware Approach Using GPT-2 and VADER

Muhi Saadi Rahdi [1], Farsad Zamani Boroujeni [2, 3*], Aladdin Abdulhassan [4], Mehdi Akbari Kopayei [5, 6] and Keyvan Mohebbi [1]

[1] Department of Computer Engineering, Isf.C., Islamic Azad University, Isfahan 81595-158, Iran;
[2] Department of Computer Engineering, SR.C., Islamic Azad University, Tehran 14515-775, Iran;
[3] Artificial Intelligence and Data Analysis Research Center, SR.C., Islamic Azad University, Tehran 14515-775, Iran;
[4] Information Network, Razi University, Kermanshah 67144-14971, Iran;
[5] Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad 85141-43131, Iran;
[6] Big Data Research Center, Najafabad Branch, Islamic Azad University, Najafabad 85141-43131, Iran.

* **Corresponding author**: farsad.zamani@iau.ac.ir.

ABSTRACT: Recommender systems play a crucial role in enhancing user experience on social networks. However, traditional methods face challenges in processing user information, such as the inability to effectively analyze textual reviews and the weakness in learning nonlinear and complex relationships between users and items. Additionally, many of these approaches fail to optimally integrate explicit information (such as, ratings) and implicit information (such as, sentiment analysis), leading to reduced recommendation accuracy. This paper proposes a novel approach to improve recommendation accuracy by integrating Autoencoder-SVD with language model-based summarization. In this method, the GPT-2 language model is employed to extract rich summaries from user textual reviews, while VADER is utilized for rule-based sentiment analysis. Furthermore, Autoencoder-SVD enables dimensionality reduction of the user-item rating matrix, and a transformer network enhances the recommendation process. The evaluation results on the Amazon Fine Foods and Amazon Clothes datasets indicate that the proposed model achieves significant improvements over benchmark methods such as SVD and NMF, reducing MAE to 0.3 on both datasets and lowering RMSE to 0.66 and 0.63, respectively. These findings demonstrate that the effective integration of explicit and implicit information within the proposed framework can enhance recommendation accuracy compared to previous methods.

Keywords: large language model, recommender systems, gpt-2-based summarization, autoencoder-SVD, rule-based sentiment analysis.

I. INTRODUCTION

We live in a time when new modes of technology and the expansion of the Internet, especially the Web and social media, have changed the way humans interact with data. There is now an astounding amount of data generated every minute (billions of gigabytes), meaning we create millions of terabytes each day. As a result, the amount of data can create enormous challenges for users when searching for, selecting, and interpreting the data. To alleviate this cognitive burden and provide personalized experiences, intelligent systems have been developed, among which recommender systems (RS) play a central role by analyzing user behavior to generate

tailored recommendations [1, 2]. Initially applied in e-commerce to boost product sales, these systems are now integral to domains such as entertainment, tourism, healthcare, and education [3].

Conventional recommendation systems utilize two main approaches: collaborative (CF) filtering and content-based (CBF) filtering [4, 5]. Collaborative filtering predicts that users who previously rated similar items similarly will rate similar items similarly in the future, while content-based filtering utilizes item features to recommend similar items [6]. Both approaches have limitations. CF suffers from the cold-start problem for new users or items, while CBF often provides limited and repetitive recommendations. Hybrid methods, which combine multiple techniques, have been proposed to overcome these limitations and improve recommendation accuracy and coverage [3].

The advent of machine learning (ML) has further enhanced recommender systems by enabling models to extract complex patterns from large-scale user data. ML-based methods can optimize recommendations by learning intricate relationships between users and items [3]. Nevertheless, most existing systems rely on static historical data, overlooking the dynamic nature of user preferences a phenomenon referred to as concept drift or temporal dynamics. As the scale of users and items increases, traditional models struggle to capture real-time shifts in interests, reducing recommendation effectiveness [6, 7]. Recent developments in hybrid recommender systems have sought to solve such challenges by utilizing neural networks, graph-based models (GNNs) and contrastive learning strategies in an effort to better capture user-item interactions and accompanying context [8-12]. Still, these latest cutting-edge strategies often ignore the rich semantic information contained in textual user reviews and/or are based on ad-hoc processing strategies that do not model evolving user preferences systemically. In particular, while SVD/NMF and GNN-based hybrids have made strides in terms of latent structures, they are often unable to include sentiment-driven textual features continuously, leading to poor quality recommendations.

To address this gap, this paper proposes a novel hybrid framework that integrates transformer-based models, sentiment analysis, and Autoencoder-SVD decomposition to enhance recommendation accuracy and adaptability. In contrast to existing methods, our method is able to make an effective utilization of explicit data (user ratings) along with implicit information mined from textual reviews. GPT-2 is used to condense evaluations into informative summaries that not only summarize your review but also reduce "noise," and maintain key information. We quantify user attitudes using VADER sentiment analysis, as an additional layer analyzing attitudes each review yields. We propose a method that reduces the dimensionality of the data while maintaining key features utilizing Autoencoder-SOV. Finally, we use a transformer network to model nonlinear user-item interactions that allow for complex interactions between users and products and allow for timelier and more accurate recommendations.

Even though there has been an increase in hybrid recommender systems that concurrently utilize textual reviews and user ratings [13, 14], current models are often limited in several ways. Some methods are limited to textual representation or traditional sentiment analysis, which overlook complex user sentiments and changing preferences. Other models use matrix factorization or neural architectures that do not take advantage of the rich semantic content in user feedback. The proposed model makes use of the unique capabilities of GPT2, VADER, and Autoencoder-SVD, which overcome the limitations of previous hybrid models. For example, GPT-2 produces smooth and concise summaries using user reviews to reduce noise and reveal the most valuable insights. VADER enables sentiment aware information and captures user attitudes. Autoencoder-SVD reduces dimensionality and retains useful rating characteristics that can be used efficiently in the model. Together, the model not only leverages explicit and implicit signals from users better than other hybrid alternatives, but it is also capable of adapting from online user content in real time to provide more accurate recommendations.

To address the continually shifting nature of user preferences and the issue of concept drift, the framework described here includes three strategies for real-time adaptation. First, GPT-2 will update those summaries from the user reviews as they arrive, allowing the model to capture the latest trends in user ratings not to mention changes in opinions over time. Second, VADER scores will update sentiment analysis on a time-sensitive basis, identifying trends or shifts in users' attitudes over time. Third, Autoencoder-SVD is an alternative approach which adopts latent features of ratings data without defining a specific latent variable structure. This preserves dimensionality while allowing the model to incorporate data more quickly without

retraining a model from scratch. Lastly, by utilizing a multi-head transformer network, they also model nonlinear user-item interactions and allowing the adapt a model's predictions based on embedding transformations of textual and numerical interaction data. Together, applying adaptive model architecture for dynamically and efficiently capture evolving user preferences will aid the recommendation process and address temporal factors that influenced user interactions.

The remainder of this paper is organized as follows: Section 2 provides an overview of recommender systems and Large Language Models, along with relevant studies in this domain. Section 3 presents the proposed methodology, while Section 4 discusses the evaluation results. Finally, the conclusions are drawn in the last section.

II. RECOMMENDER SYSTEMS

In our society today, where information is growing immensely, recommender systems have become important in improving the user experience and assisting user decision making in a logical manner. Recommender systems take user data and provide personalized options based on that data, and are found in various industries, from e-commerce to digital media [15-17].

Collaborative filtering is one of the most commonly used recommendation techniques, providing suggestions based on similarities between users or items [18]. However, issues like the "cold start" problem and data sparsity affect recommender systems and have a big influence on their predictive accuracy [19]. The user, the product, and the recommendation engine are typically the three primary parts of these systems. The system's central component, the recommendation engine, examines user preferences and product characteristics to provide the most pertinent options. In order to increase recommendation accuracy and personalization, recent research has concentrated on optimizing this component and integrating multiple approaches. Collaborative filtering is a primary approach in recommender systems that generates personalized recommendations based on users' past interactions and behavioral pattern similarities [5]. This method utilizes a user-item rating matrix to extract relationships between users or items and leverages these relationships to predict user preferences. The rating data can be either explicit (such as, direct ratings) or implicit (such as, user behavior analysis).

Despite advantages such as independence from product content and the ability to provide unexpected recommendations, collaborative filtering suffers from challenges like data sparsity, the cold start problem, and scalability limitations [20]. To address these limitations, hybrid and model-based approaches have been developed to enhance the system’s efficiency in processing large-scale data and generating more precise recommendations [21]. Collaborative filtering methods are categorized into two main types: Memory-based approaches, which directly store and process the rating matrix, and Model-based approaches, which leverage machine learning techniques to predict missing ratings and optimize the recommendation process.

III. LANGUAGE MODELS: FROM STATISTICAL MODELS TO LARGE LANGUAGE MODELS

Language modeling (LM) plays a fundamental role in natural language processing (NLP) by estimating the probability of word sequences to predict or reconstruct text. This field has evolved through several stages:

- Statistical Language Models: These models, based on statistical methods and the Markov assumption such as n-gram models faced limitations due to the curse of dimensionality and data scarcity. Techniques like smoothing were introduced to mitigate these issues [22].
- Neural Language Models: With the advent of neural networks such as recurrent neural networks (RNNs) and multilayer perceptrons (MLPs), distributed semantic representations of words were introduced. The word2vec model was a notable success in enhancing NLP applications [22].
- Pretrained Language Models: The introduction of models like ELMo and BERT, leveraging transformer architectures and self-attention mechanisms, significantly improved contextual word representations. This phase introduced the concept of pretraining and fine-tuning [22].
- Large Language Models (LLMs): The expansion of model parameters and training data led to emergent capabilities. Models such as GPT-3 and PaLM demonstrated abilities like few-shot learning. ChatGPT, as an example of LLMs, enables interactive and dynamic conversations [22].

- GPT-1: 2018 saw the release of GPT-1, the initial iteration of the GPT series. It was created using a hybrid approach that combined supervised fine-tuning and unsupervised pretraining, and it was based on a decoder-only transformer architecture. GPT-1 established fundamental principles for natural language modeling, focusing on next-word prediction in a given text sequence [22].
- GPT-2: Although it greatly increased its parameters to 1.5 billion, GPT-2 built upon the architecture of GPT-1. It was trained with a large internet text dataset, called Web Text, using an autoregressive methodology of predicting each token based on prior tokens. Unlike traditional NLP models that use architecture specifically for a task, GPT-2 was trained unsupervised on a vast amount of unlabeled data; therefore, no task-specific annotations are needed.

GPT-2 is based on a transformer decoder-only architecture which consists of multiple self-attention layers and feed-forward neural networks. Self-attention with multi-head attention and layer normalization enables the acquisition of long-range dependencies in the text. The model uses Byte Pair encoding (BPE) to facilitate out-of-vocabulary (OOV) words. In general, it is trained to maximize the likelihood of predicting the next token conditioned on all previous tokens, this means it can work effectively to perform a variety of NLP tasks such as text generation, summarization, or machine translation.

GPT-2's scalability is one of its important advancements over its predecessor. It was released in multiple sizes, with the largest version containing 1.5 billion parameters. GPT-2 demonstrated zero-shot, one-shot, and few-shot learning capabilities, meaning it could perform various language tasks without requiring retraining. These emergent abilities highlighted the crucial role of large-scale pretraining in transformer-based models [22].

IV. RELATED WORKS

Recommender systems are powerful applications that work by examining users and their activity data to create personalized recommendations for products, content, or services. These applications use many types of algorithms, such as content based filtering and collaborative filtering, to improve user experience and user engagement. These applications can help users find products or services similar to ones they might already like and they can reduce total search time. On the downside, they can create "information bubbles" that isolate users in repeated previously consumed or similar content limiting user choice. The next section will provide a review of the current literature on recommender systems for movies and food, specifically studies published between 2023 and 2025.

Past investigations of movie recommender systems have utilized a number of approaches, each of which has its strengths and limitations. Graph embedding techniques [23] enhanced predictive accuracy but faced scalability and computational complexity issues. Sentiment analysis and social influence models [24] reduced recommendation errors but remained susceptible to erroneous input data. Furthermore, multimodal deep learning approaches [25] demonstrated high precision but required significant computational resources, limiting their practicality in large-scale systems. Temporal information integration [26] improved similarity calculations but struggled with rapid preference shifts. Meanwhile, collaborative filtering methods and hybrid approaches [27, 28] yielded promising results but continued to face challenges related to cold-start issues and scalability.

Mu and Wu (2023) proposed a multimodal deep learning-based movie recommender system that outperformed traditional content-based filtering and singular value decomposition (SVD) approaches. While the model demonstrated superior predictive accuracy, it required substantial computational resources and introduced additional complexity [25]. Jain et al. (2023) enhanced interactive filtering-based recommender systems by incorporating temporal information, using exponential decay and power functions to compute user similarities, which improved the MAE and RMSE metrics. However, a key challenge was integrating temporal data while adapting to evolving user preferences [26]. Siddik and Wibowo (2023) assessed matrix factorization methodologies for food recommender systems. They compared singular value decomposition (SVD), SVD++, and non-negative matrix factorization (NMF). Overall, the result of their study revealed that NMF and SVD++ displayed better performance, with NMF obtaining and MAE lower than others. Despite their effectiveness, scalability remained a concern when handling complex datasets [29].

Rostami et al. (2023) introduced a model that incorporated user preferences and food health attributes, improving recommendation accuracy. However, its adaptability to diverse datasets required further refinement [30]. Hiriyannaiah et al. (2023) proposed DeepLSGR, a hybrid model combining collaborative filtering with deep learning, utilizing long short-term memory (LSTM) and gated recurrent units (GRU) for score prediction. While enhancing accuracy and retrieval rates, model complexity affected processing efficiency [31]. Mohammadpour et al. (2023) introduced a clustering-based approach to simulate user behavior, optimizing the number of neighbors and leveraging the NSGA-II algorithm to enhance recommendation accuracy. Despite its effectiveness, scalability remained a concern for large user bases [32]. Sridhar et al. (2023) developed a content-based recommender system utilizing Facebook user profiles, integrating an MBO-based feature selection algorithm and a deep belief network (DBN) for classification. Although the model outperformed similar approaches, challenges persisted in processing complex social data and addressing privacy concerns [27]. Huang et al. (2024) implemented deep neural networks in collaborative filtering and introduced the BroadCF model, which significantly improved recommendation quality. However, preprocessing and aligning user-item interactions posed difficulties [33]. Similarly, Siet et al. (2024) proposed a hybrid deep learning-based system that combined transformer architectures and multilayer perceptrons, demonstrating improvements in accuracy, recall, and F1-score, yet facing scalability and resource allocation challenges for large datasets [34].

To address the cold-start problem, Latrech et al. (2024) introduced CoDFi-DL, a hybrid recommender system integrating collaborative and demographic filtering. While the approach outperformed baseline models, integrating and processing both filtering methods posed difficulties [28]. Liu and Zhao (2024) developed a ranking-based recommender system that incorporated sentiment analysis using BERT and matrix factorization techniques. The model integrated user and item feature matrices, reducing the mean absolute error (MAE) to 0.4. While demonstrating significant improvements, further optimization through hybrid techniques could enhance its performance [9]. Imantho et al. (2024) combined machine learning and content-based collaborative filtering to create a genetic algorithm-based food recommender system. The system, which was created for people with particular medical conditions like diabetes and hypertension, effectively balanced dietary requirements with individual preferences. However, scalability and adaptability to diverse datasets posed challenges [35]. Rostami et al. (2024) incorporated sentiment analysis into food recommender systems, yielding superior performance compared to conventional models [36]. However, the system's performance relied on an accurate representation of user attitudes and may have rendered predictive inaccuracies. Gupta et al. (2025) proposed a multimodal graph-based recommender system using GCNs and a Variational Graph Autoencoder to capture complex user-item interactions. The model outperformed existing multimedia recommender systems on several benchmarks, including precision, recall, NDCG, and RMSE [37]. Deldjoo et al. (2025) introduced CFaiRLLM, a fairness evaluation framework for LLM-based recommender systems that considers true preference alignment and intersectional fairness. Their findings highlighted that incorporating intersectional attributes reveals greater fairness disparities, especially in unstructured domains such as music recommendations [38]. Di et al. (2025) proposed FedRL, a reinforcement learning-based federated recommender system designed to balance privacy, communication efficiency, and recommendation quality. By integrating a Reinforcement Selector and Hypernet Generator, the model optimizes edge device participation and bandwidth use, effectively addressing heterogeneity and data sparsity in decentralized environments [39].

Another important challenge with recommender systems, increasingly noted in recent research, is that most are static by relying only on past user ratings, which overlook changing user preferences. Some studies have addressed this issue by incorporating sentiment analysis of user reviews to refine recommendations dynamically. In response to these limitations, the proposed approach integrates Autoencoder-SVD for dimensionality reduction and transformers for optimizing recommendation processes. Additionally, employing GPT-2 for extracting enriched summaries from reviews and VADER for sentiment analysis significantly enhances predictive accuracy, making the model more reliable and precise.

V. PROPOSED METHOD

The approach presented in this paper introduces a deep recommendation system that utilizes transformer-based deep neural networks with attention mechanisms. This model aims to enhance recommendation reliability by integrating explicit user ratings with implicit feedback derived from user reviews. To achieve this, the model leverages GPT-2 for summarizing user reviews and employs the VADER rule-based method to extract sentiment scores.

To reduce computational complexity, the proposed system incorporates an Autoencoder-SVD network that performs dimensionality reduction at two levels. Figure 1 illustrates the schematic diagram of the proposed method. The input data for the model includes user IDs, product IDs, user rating scores for products, and the textual content of user reviews. The primary and most crucial input to the model is the file containing user reviews and explicit ratings, which also includes product and user identifiers. Without this file, the model cannot learn or generate new recommendations.

The proposed framework addresses the cold-start problem by leveraging textual reviews and sentiment information. For new users or items with few or no ratings, GPT-2–generated summaries of available reviews and VADER sentiment scores are utilized to provide implicit preference signals. These textual features are integrated with the Autoencoder-SVD latent representations, allowing the model to make initial recommendations even in the absence of extensive historical ratings. This approach ensures that cold-start users and items receive meaningful predictions while the model gradually updates as more interaction data becomes available.

1. SUMERIZATION OF USER REVIEWS USING GPT-2 AND SENTIMENT SCORE EXTRACTION WITH VADER

Given the data format, the first step involves transforming the raw data into a structured format suitable for processing within the transformer-based recommendation system. At this stage, explicit user rating information, user IDs, product IDs, and sentiment scores derived from user reviews are extracted. Subsequently, a unique and non-redundant list of all users is generated from the dataset. A similar process is applied to extract a distinct list of all products. Following this, the ratings assigned by each user to each product are retrieved. Next, sentiment analysis scores for user reviews are extracted. This step requires textual data mining and sentiment processing of user-generated content. To mitigate computational complexity, the GPT-2 language model is utilized for summarizing user reviews. The original reviews are fed into the pre-trained GPT-2 model, which generates condensed versions of the textual data.

Although more recent transformer-based models such as GPT-3 [40], T5 [41], and BART [42]offer advanced capabilities in text summarization, GPT-2 was selected for this study due to several practical considerations. First, GPT-2 provides a favorable trade-off between computational efficiency and performance, allowing for faster training and inference on large-scale datasets without requiring extensive hardware resources. Second, GPT-2 has demonstrated robust performance in generating coherent and informative summaries from user reviews, which is sufficient for capturing essential semantic content and sentiment cues in the context of recommender systems. Finally, the integration of GPT-2 with VADER and Autoencoder-SVD ensures that both textual and numerical information are effectively combined, making GPT-2 a suitable and efficient choice for the proposed hybrid framework.

GPT-2 is an autoregressive transformer-based architecture that employs a multi-head self-attention mechanism for learning linguistic sequences. It consists of multiple stacked transformer blocks, which are further elaborated in the subsequent sections.

1.1 Input Embedding Layer

The model's input consists of a sequence of textual tokens $\{x_1, x_2, \dots, x_T\}$, which are first transformed into vector embeddings using Equation 1[43]:

$$E = \{e_1, e_2, \dots, e_T\}, \quad e_i = W_e . x_i \tag{1}$$

In the above equation, $W_e \in \mathbb{R}^{V \times d}$ represents the word embedding matrix, where V denotes the vocabulary size and d represents the embedding dimension. To preserve the positional information of tokens in the sequence, positional encoding is added using Equation 2:

$$Z_i = e_i + PE(i) \tag{2}$$

where $PE(i)$ is typically computed using sinusoidal and cosine functions as expressed in Equation (3)[43]:

$$PE(i, 2j) = \sin\left(\frac{i}{10000^{\frac{2j}{d}}}\right), \quad PE(i, 2j+1) = \cos\left(\frac{i}{10000^{\frac{2j}{d}}}\right) \tag{3}$$

In the above equation, i represents the position of the word in the input sequence (e.g., the first word has i=1, the second word has i=2, and so on). The variable j denotes the dimension index in the embedding vector, ranging from 0 to $d/2$. The parameter d is the embedding dimension, which is typically a fixed value such as 512 or 1024 in large-scale models. The constant 10,000 is used to scale different frequency components. Since Transformer-based models like GPT-2 process sequences of words in parallel rather than sequentially, unlike RNNs, they do not inherently encode word order within the model itself. The sinusoidal and cosine functions provide each position with a unique representation. Different values of j result in varying wavelength values across the embedding dimensions, enabling the model to capture positional information at different scales.

The constant 10,000 is chosen to ensure that the sinusoidal and cosine functions generate appropriate frequency components for encoding both nearby and distant positional relationships. The term $10000\text{^}(2j/d)$ controls the frequency scaling, such that lower dimensions capture high-frequency variations while higher dimensions capture low-frequency variations.

1.2 Transformer Blocks

The model architecture consists of stacked Transformer layers, each comprising two key components:

- Multi-Head Self-Attention (MHSA): At this stage, each embedded vector is first transformed into three vectors: Key, Query, and Value.

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V \tag{4}$$

In the above equation, $W_V, W_Q, W_K \in \mathbb{R}^{d \times d_k}$ are the weight matrices, where $d_k = d/h$ represents the dimensionality of each attention head, and h denotes the number of attention heads.

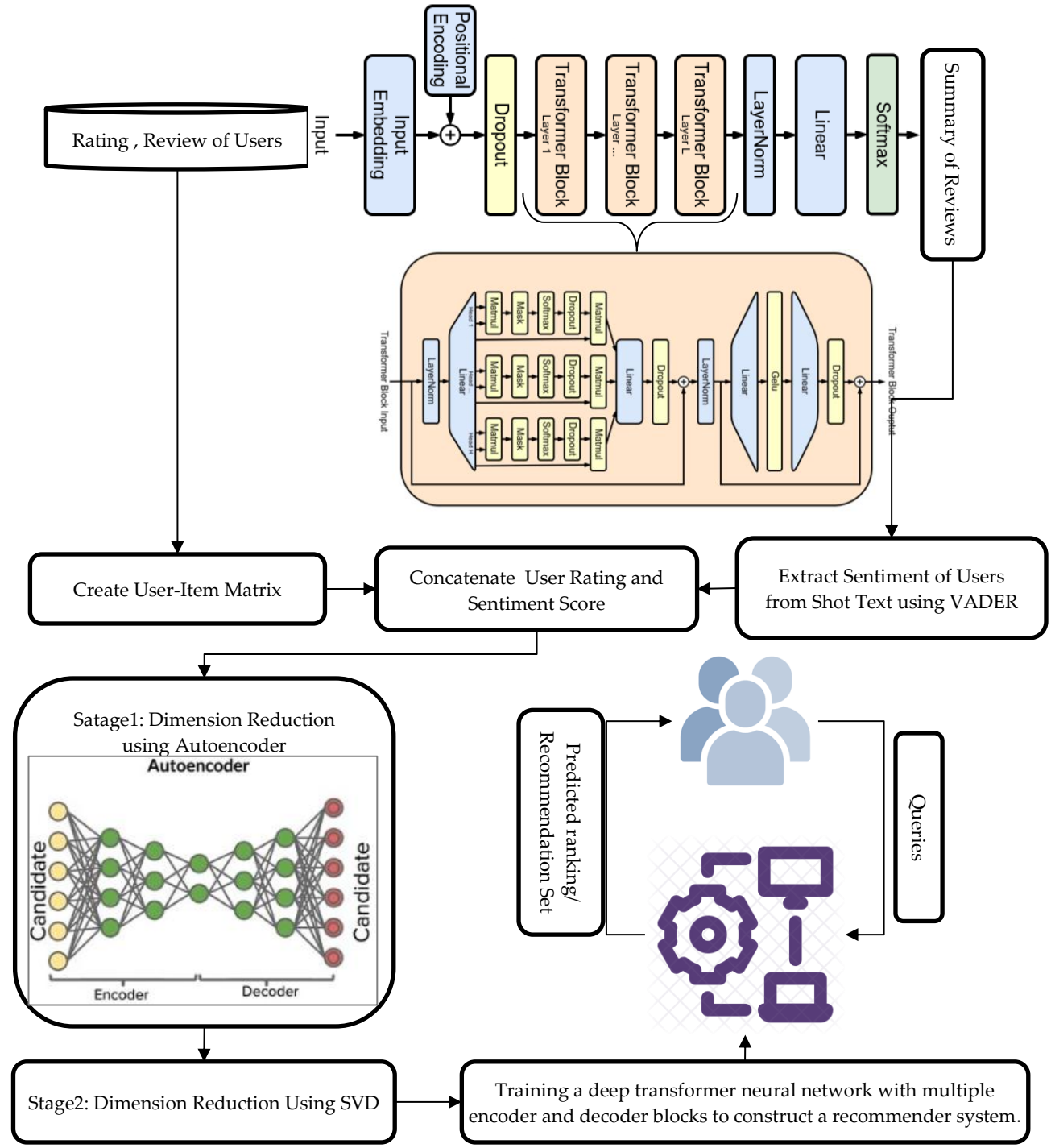

FIGURE 1. Proposed model.

In the Transformer block, attention scores are computed using the dot product between the Query and Key matrices, followed by normalization via the SoftMax function [43]:

$$A = Sotmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (5)$$

In the attention mechanism, Q(Query), K(Key), and V(Value) represent the input feature vectors, while $d_k$denotes the dimension of the Key vector for scaling. The operation first computes the scaled dot-product $QK^{\mathrm{T}}/\sqrt{d_k}$, applies the Softmax function to obtain attention weights, and then multiplies by V to produce the final attention output A. Subsequently, multiple parallel attention heads are concatenated, and the final vector is generated using Equation 6:

$$MHSA\ (X) = Concat(A_1, A_2, A_3, \ldots\ldots, A_h,)\, W_o, \quad (6)$$

In the above equation, W_o represents the output weight matrix, and the A values are computed using Equation 5 [44]. Feedforward Neural Network (FFN): After the self-attention mechanism, each output vector passes through a feedforward neural network comprising two layers:

$$FFN(x) = \max(0, xW_1 + b_1)\, W_2 + b_2 \quad (7)$$

In the above equation, W represents the weight matrices, b denotes the bias terms, and x refers to the input data[44].

1.3 Normalization and Weight Adjustment

In each transformer block, two-layer normalization steps (Layer Normalization - LN) are applied, as described in equation 8, to stabilize the gradient values:

$$LN(x) = (x - \mu)/(\sigma + \varepsilon)\, \gamma + \beta \quad (8)$$

In this equation, x is the input vector to the normalization layer, μ and σ represent the mean and standard deviation, while γ and β are the model's learnable parameters, ϵ is a small constant added for numerical stability and LN(x) is the normalized output. Additionally, according to Equation 9, the residual connections after each main operation are applied to stabilize the training process.

$$X_{out} = X_{in} + Layer \quad (9)$$

In Equation 9, X_{in}denotes the input to a sub-layer, Layer represents the output of that sub-layer (such as, attention or feed-forward), and $X_{\text{out}}$is the result after adding the residual connection to stabilize training.

1.4 Output Layer and Next Token Prediction

In the GPT-2 model, the output of the last transformer layer is connected to the output layer, and the probability of each token is computed.

$$P(x_{t+1}|x \le t) Softmax(Z_T W_e^T) \quad (10)$$

In the above equation, W_e is the same input embedding matrix, which is reused as the output weights. $P(x_{t+1}|x \le t)$ represents the probability of predicting the next word in a list of words, based on what has been said prior to it, in which x_t is a word (token) from the input sequence and $x \le t$ is the entire sequence up until word t. T is the length of the input sequence (the number of words in the input sentence). Z_T is the final output vector produced by the last transformer layer after processing the entire input sequence. This equation shows the probability of the next word occurrence in the model's output. The model first processes the input and produces a vector for the entire sentence in the output. Then, this vector is multiplied by the embedding matrix W_e^T , and after passing through the SoftMax function, the probability of each word in the vocabulary for the next position is calculated.

The GPT-2 architecture, by combining the autoregressive transformer, multi-head self-attention mechanism, and deep feedforward networks, has created an efficient model for text summarization. The key features of this

model include unsupervised training, scalability, and the ability to learn contextually, positioning it among the large language models (LLMs) and making it highly effective in applications such as text generation and interactive dialogue.

During the review summarization stage, the Hugging Face Transformers library's pretrained GPT-2 small model (117M parameters) was spotlighted. This model was run without fine-tuning on the Amazon review datasets to maintain some generalizations regarding text abstraction while minimizing resource utilization. To ensure similar quality of summary generation or avoid excessive truncation and loss of key concepts, the model was constrained to use a maximum of 256 tokens of review input and to output a maximum of 50 tokens in the summaries. The selected hyperparameters were max_length = 50, min_length = 15, temperature = 0.7, top_k = 50, top_p = 0.9, repetition_penalty = 1.2, num_beams = 4, and no_repeat_ngram_size = 3. These hyperparameters were found through engineering to illustrate a balance between fluency and conciseness and retain sentiment-rich phrases without introducing anything artificial or hallucinated as data-poor, case in kind, or substitute machines learning model actions.Using the above network, a summarization for a review from the Amazon database is performed as follows:

- Original Review: This is a confection that has been around a few centuries. It is a light, pillowy citrus gelatin with nuts - in this case Filberts. And it is cut into tiny squares and then liberally coated with powdered sugar. And it is a tiny mouthful of heaven. Not too chewy, and very flavorful. I highly recommend this yummy treat. If you are familiar with the story of C.S. Lewis' "The Lion, The Witch, and The Wardrobe" - this is the treat that seduces Edmund into selling out his brother and Sisters to the Witch."
- Generated Using GPT-2: A delicious, centuries-old treat soft citrus gelatin with filberts, coated in powdered sugar and highly recommended! Prior to conducting sentiment analysis, all user reviews were subjected to three distinct stages of preprocessing: summarization, cleaning, and tokenization.
- summarization: Each raw review was summarized using the pretrained GPT-2 model to extract its key semantic and emotional content while discarding unnecessary redundant, noisy, or irrelevant information. The summarization was completed at the level of each sentence with an upper size of 50 tokens per summary so that the amount of information (text) for review samples was controlled and consistent.
- (2) cleaning: Then, the summarized review was normalized, including converting the summarized review to lowercase, and removing stopwords, excessive whitespace, and non-ASCII characters. Nevertheless, because sentiment-bearing symbols are emotionally informative regarding the polarity detection conducted by VADER, they were retained, e.g., “!” and “?”.
- (3) tokenization: Finally, for tokenization, each cleaned review was chunked into single tokens based on whitespace delimiters and punctuation markers. For example, the summarized text:

“A delicious, centuries-old treat soft citrus gelatin with filberts, coated in powdered sugar and highly recommended!” was converted into the following token sequence:

["A", "delicious", "centuries-old", "treat", "soft", "citrus", "gelatin", "with", "filberts", "coated", "in", "powdered", "sugar", "and", "highly", "recommended", "!"].

Tokens representing punctuation marks such as commas or parentheses were discarded, except for those relevant to sentiment emphasis.

After these preprocessing steps, the cleaned and tokenized reviews were used as input for the sentiment extraction stage based on the VADER lexicon and BERT contextual embeddings. The sentiment analysis model based on the VADER rule is a lexicon-based tool designed for sentiment analysis in short texts, such as social media posts. This model uses a set of predefined rules and vocabulary to identify and evaluate the sentiments in the text. VADER is a lexicon-based sentiment analysis algorithm, specifically designed for sentiment analysis in social media. It uses a weighted lexicon to determine the polarity of sentiments (positive, negative, neutral) and also accounts for intensifiers (such as "very" or "extremely") and negators ("not")[45]. The steps in this section are as follows:

1.4.1 Calculating the initial sentiment score:

The initial score for each word in the text is determined using the VADER lexicon, based on equation (11). Each word has a sentiment intensity score, which falls within the range of [−4, +4].

$$S_i = LexiconScore(w_i) \quad (11)$$

In the equation above, S_i is the sentiment intensity of the word w_i, and LexiconScore is the pre-defined emotional value in the VADER lexicon for each word.

1.4.2 Adjustment of Scores Based on Intensifiers (Intensifiers or Boosters)

Certain words, such as "very" or "extremely," intensify the sentiment. Intensifiers increase the sentiment score by 15% to 30%.

$$S_i' = S_i(1 + \alpha) \quad (12)$$

In the equation above, S_i is the initial sentiment score calculated from Equation 11, and α is the intensity increase factor, typically set between 0.15 and 0.3.

1.4.3 Adjustment of Scores Based on Negators (Negation Handling)

Words such as "not," "never," and "isn't" reverse the sentiment intensity. If a word contains a negator, its sentiment value is reversed or reduced according to Equation 13.

$$S_i' = S_i \times (-0.75) \quad (13)$$

A value of -0.75 represents a 75% reduction in sentiment intensity.

1.4.4 Impact of Uppercase Letters, Punctuation, and Emojis

- Uppercase Letters: If an emotional word is written entirely in uppercase, its intensity is increased by 25%.
- Punctuation: The presence of an exclamation mark (!) increases both positive and negative sentiment intensity.
- Emojis: Some emojis, such as smiley faces or angry faces, have a specific emotional score in the VADER lexicon.

1.4.5 Calculation of Final Sentiment Score

After processing and adjusting the scores for each word, the final sentiment of the text is calculated based on the sum of positive, negative, and neutral scores using Equation 14.

$$Compound = \frac{\sum S_i}{\sum S_i^2 + \varepsilon} \quad (14)$$

ϵ in the previous equation represent a small value to avoid divide by zero error ($10^{(-6)}$ usually), S_i is computed according to Equation (11) and the value of compound represents the overall sentiment score, which ranges from [-1, 1]. The summarized review above had a VADER sentiment score of 0.8310, which indicates very strong positive sentiment by the user regarding the product. VADER: 0.8310 ==> "a" "delicious" "centuries-old" "treat" "soft" "citrus" "gelatin" "with" "filberts" "coated" "in" "powdered" "sugar" "and" "highly" "recommended".

Although deep contextual sentiment analysis models such as RoBERTa-based classifiers offer high accuracy, VADER was selected for this study due to several practical advantages. First, VADER is lightweight and computationally efficient, allowing for rapid processing of large-scale user reviews without requiring extensive hardware resources. Second, VADER provides interpretable sentiment scores that are easy to integrate with numerical rating data and other model components, such as GPT-2 summaries and Autoencoder-SVD embeddings. Third, despite being rule-based, VADER has been shown to achieve competitive performance for short, informal text typical of user reviews making it suitable for capturing essential sentiment signals while maintaining overall model efficiency. This combination ensures that sentiment information enhances recommendations effectively without introducing significant computational overhead.

2. USER-ITEM RATING MATRIX FORMATION

This section's main goal is to create a user-item matrix that displays user ratings of items. To begin, I will get a list of unique users and items. Then I will create a matrix called ratingMatrix based on the users and items, where every user rating for their items will be stored. The initial values of this matrix are set as zero. Next, I will establish a mapping from each user and item to its corresponding index in the matrix. This mapping I will use a data structure such as Map, to ensure that each user and item has a unique index in the user rating matrix. The next part is to calculate the average sentiment of each user by using the average of sentiments divided by sentiment total, which will be stored. Finding patterns in user behavior and preferences starts with calculating the average sentiment score for that user. This allows not only for better predictions, but also to create user behavior patterns as well as configure the personalization principles for recommendation systems. This variable serves as an adjunct, providing additional knowledge about user interactions with items. By combining user ratings and sentiments, richer information about user behavior can be extracted, thereby improving the accuracy of predictions.

3. DIMENSIONALITY REDUCTION AT THE FIRST LEVEL: USING AN AUTOENCODER NEURAL NETWORK

The user-item rating matrix is very sparse; it has a lot of zeros. The size of the matrix prohibits it from being passed to a transformer network at all, as the matrix is too large, thus is too prone to computational throughput. To mitigate these issues the technique proposed in this paper implements 2 levels of dimensionality reduction through an autoencoder network and Singular Value Decomposition (SVD).

The joint use of Autoencoder and Singular Value Decomposition (SVD) in the proposed framework is not by chance. Rather, one method complements the other by harnessing latent feature extraction and dimensionality reduction in ways that are aligned/nearly identical. Autoencoder models operate to create nonlinear latent interpretations by modeling complex user-item interactions that a least squares approach may not successfully model. However, autoencoder-based modeling can also lead to a latent construct that may reduce redundant latent dimension or severe instability to reconstruct ability in practice, especially if trained on a sparse and noisy rating matrix present in recommender system outcomes.

To safeguard against problematic reconstructions in the autoencoders, SVD is introduced as a second-stage refinement mechanism. The role of SVD as this stage is not redundant dimensionality reduction fraught with interpretability issues but to orthogonalize and sequence the latent space learned by the autoencoder (by removing any correlations among the latent vectors). Evidence from SVD supports the interpretability and reduced redundancy in the latent dimensions helpful for recommendation systems. To summarize here, the proposed two-stage structure allows the model to benefit as a nonlinear abstraction stage (via autoencoder) or a low-rank factorization stage (by SVD). The hybrid capability embedded within provides the expressed model with critical computing cost efficiency across both alternative expressing conditions of cooperation while estimating low-rank approximations. Pure neural models (like NCF or Variational autoencoders (VAE) or similar cost projections) may not support similarly with deep neural or standard probabilistic data (as full or hybrid conditions conditions).

While Neural Collaborative Filtering (NCF) [46] and Variational Autoencoders (VAEs) [47] are powerful frameworks, they often require extensive hyperparameter tuning and exhibit overfitting tendencies when dealing with limited user–item interactions. Moreover, VAEs assume a predefined probabilistic distribution of latent variables, which may not accurately represent real-world preference distributions. In comparison, the Autoencoder–SVD combination performs a data-driven, distribution-free, and interpretable, reduce dimensionality process to facilitate faster convergence and better generalization across heterogeneous data. The modular structure facilitates hooking into other components of the proposed framework (e.g., sentiment-aware module, transformer-based module, etc.) while also providing computational scalability and robustness against sparse data inputs.

In the first level of dimensionality reduction an autoencoder reduces the dimensions while also extracting the compressed features from the input. The dimensionality reduction through the autoencoder allows the model to learn a compressed representation of the original dataset, or user-item rating matrix, while keeping important information from the original dataset. An autoencoder is an unsupervised way to compress data

using an encoder-decoder architecture; the encoder compresses the input data into a lower number of dimensions while the decoder attempts to reconstruct the original dataset from the compressed vectors.

The dataset is defined as a matrix R with dimensions m×n, where mmm represents the number of users (or samples) and n denotes the number of features (or items). After the autoencoder, the input dimension is set to the number of items in the dataset. The encoder transforms the input data dimension into hidden size 100 lower dimension, which will allow for the accentuation of important patterns and fortification of noise. The following hyperparameters adjust the learning process:

- L2 Regularization (λ): To prevent overfitting, the L2 regularization value is set to 0.01. This prevents the weights from growing excessively and improves the model's generalization by increasing its ability to perform well on unseen data. The L2 regularization term is given by the following relation:

$$L_{reg} = \lambda \sum_{i=1}^{n} w_i^2 \tag{15}$$

In the above equation, w_i are the model's weights, which are typically adjusted through optimization algorithms such as Gradient Descent. λ is the regularization parameter that controls the importance of the penalty. n is the number of weights in the model. This method applies a penalty to the model based on the magnitude of the model's weights, ensuring that the model focuses on real and important features rather than learning random features and noise present in the training data.

- Sparsity Regularization (β): A sparsity constraint is applied to ensure that only a few neurons in the hidden layer are activated for each input. This constraint is controlled through a sparsity penalty, as shown in the following equation:

$$S_{reg} = \beta \sum_{j=1}^{j} KL(\rho||\hat{\rho}_j) \tag{16}$$

In the above equation, β is the sparsity regularization weight controlling the influence of the penalty, $KL(\rho||\hat{\rho}_j)$ is the Kullback-Leibler divergence[48], which measures the difference between the desired sparsity level ($\rho = 0.055$) and the actual activation probability of neuron j in the hidden layer ($\hat{\rho}j$). S_{reg} is the resulting sparsity penalty added to the loss function.

The autoencoder module was trained for 100 epochs, empirically evaluated as the time to converge. An observation of the model's reconstruction loss across multiple runs at epochs of 50, 75, 100, and 150 during initial experiments found that the loss came to a plateau about the 90th epoch, indicated minor improvement past 100 epochs while training time continued to grow considerably. The 100 epochs were therefore considered a good balance of obtaining a stable convergence with minor overfitting. In addition, early-stopping and drop-out regularization were utilized to mitigate the overfitting tendencies and improve generalization capacity. The training process of the autoencoder uses the backpropagation algorithm and minimizes the following cost function to optimize the reconstruction of the data:

$$\mathcal{L} = \sum_{i=1} \left\| R_i - \hat{R}_i \right\|^{\angle} + L_{reg} + S_{reg} \tag{17}$$

In this equation, R_i represents the original input data (user-item rating matrix), $\hat{R}_i$ is the output reconstructed by the autoencoder, L_{reg} is the regularization penalty to prevent overfitting, and S_{reg} is the sparsity penalty to control neuron activations. After training, the encoder function extracts the compressed features from the input using Equation 18:

$$Z = encode(autoenc, R) \tag{18}$$

In this equation, Z represents the low-dimensional and compressed representation of the input data R (user-item rating matrix), autoenc refers to the trained autoencoder model and encode(·)denotes the encoding function of the autoencoder that maps the input to a lower-dimensional space. This transformation enables the

model to learn high-level, meaningful representations of the data while simultaneously reducing the redundancy of unnecessary information.

4. DIMENSIONALITY REDUCTION AT THE SECOND LEVEL: USING SVD

In the second dimensionality reduction stage, the SVD (Singular Value Decomposition) algorithm is used to reduce the dimensions of the matrix Z obtained from Equation 18 after the autoencoder network. Singular Value Decomposition (SVD) is an advanced linear algebra technique that helps decompose matrices into three main components. This method is widely used for dimensionality reduction, data reconstruction, and feature extraction [49].

Steps of the SVD Process:

- Decomposition of the main matrix: In this method, a matrix Z with dimensions $m \times n$ (containing mmm rows and n columns) is decomposed into three matrices using Equation 19 [49].
-

$$Z = S.U.V^T \tag{19}$$

In the matrix reconstruction, U is an $m \times m$ matrix containing the left singular vectors of the original matrix, S is an $m \times n$diagonal matrix representing the singular values, and $V^{\mathrm{T}}$is an $n \times n$ matrix containing the right singular vectors of the original matrix as its transpose.

- Feature Extraction: The singular values in S are sorted in descending order. Larger values in S represent more important features.
- Dimensionality Reduction: By selecting the top k largest singular values and discarding the smaller ones, the dimensionality of the matrix is significantly reduced. This process ensures that the reconstructed matrix retains only the most important information from the data.
- Matrix Reconstruction: The original matrix is reconstructed using the reduced components , S_k,U_k, V_k , with the relation (20) [49]:

$$Z_{reduce} = S_k.U_k.V_k^T \tag{20}$$

where $S_k$contains the top k singular values, $U_k$and $V_k$are the corresponding left and right singular vector matrices, and $V_k^{\mathrm{T}}$is the transpose of V_k.

Using Singular Value Decomposition (SVD) for dimensionality reduction at the second level completes the data preparation for use in the deep neural network transformer. These three components represent the core structure of the reduced user-item rating matrix, which includes the singular vectors and values. For dimensionality reduction, only the first k=30 singular values are selected. These values represent a significant portion of the data's main information, while less important features are discarded. The reduced components U_k, S_k, and V_k are used for reconstructing the reduced matrix.

The final reduced features are obtained by multiplying the matrix Z by V_k. These features are then combined with the sentiment scores extracted by VADER to form the final matrix. This matrix, which includes the reduced features and additional sentiment scores, prepares the data for training the transformer model. The sentiment analysis score of users is added as a column at the end of the reduced matrix to be given alongside the explicit ratings of users to the transformer network. Overall, this method allows for dimensionality reduction while maintaining the main information in a compressed form. Using the extracted features from the data creates a suitable foundation for enhancing the performance of deep learning models.

5. RECOMMENDATION GENERATION WITH TRANSFORMER NETWORK

The final part of the proposed recommendation model involves using a Transformer network to generate recommendations for users. After preparing the user rating matrix by dimensionality reduction and then adding the sentiment score as an additional column, the data is ready to be fed into the Transformer network. The structure of the Transformer network is similar to the one shown in Figure (2). The layers of this network are as follows:

- Input Layer: This layer receives the data and passes it to the subsequent layers. The input dimension is 31, representing k=30 from the Autoencoder-SVD stage plus 1 for the sentiment analysis value.
- Word Embedding Layer: This layer converts the ordinal input values into feature vectors in the vector space. It requires two inputs: vocabSize which specifies the number of distinct values in your data (such as, the vocabulary size) and modelHiddenSize which specifies the number of features contained in the embedded vector. In this study, these are defined as 31 and 100, respectively.
- Position Embedding Layer: To add sequential information to the input data, this layer adds position vectors to the embedded data. This operation ensures that the model can consider positional information throughout the sequence. In this research these two inputs, vocabSize and modelHiddenSize, are defined as 31, and 100, respectively.
- Addition Layer: This layer aggregates the previously embedded data (Word Embedding and Position Embedding) and performs the addition and sends this data down the graph as output.
- Self-Attention Layer: This layer is a critical mechanism in Transformers, and it enables the model to concentrate its attention to the relevant portions of the sequence.
- Normalization and Fully Connected Layers: Fully connected layers, the normalization layers, and the ReLU activation functions all take care of processing the data. The Fully Connected layer captures abstract features from the data, therefore allows the identification of more complicated features from the inputs. The ReLU activation function helps the model learn nonlinear relationships more easily, therefore improves the performance of the model. Then, the Layer Normalization layer standardizes the scale of the data, providing faster convergence of the model and more stable training. The combination of these different layers are important for optimal and effective learning in your network.
- Structure of the Encoder Blocks: The model consists of three encoder blocks, which develop the information being represented in a hierarchical manner. Each encoder block contains various key layers, such as the Self-Attention layer, which allows the model to find long-term relationships between sequences of inputs; the Addition layer, which makes an addition of the attention layer input and output to prepare the sequence of information for the next step; the Normalization layer, which brings all the scale of the data to a similar scale, allowing the model to develop more control and apply stability in scaling to improve the speed of convergence; the Fully Connected layers, which develops the abstract features needed to help the model identify complicated patterns in the data; and, the ReLU activation function, which maps the data into non-linear spaces to learn more complex relationships. The encoder block applies these layers in succession, which feeds all the developed data downstream, allowing the model to see more complex patterns in the output data.
- End Layers: The end layers of the network serve the purpose of taking the processed data and creating the outputs of the model. The Indexing layer selects specific values from the input sequences, enabling the model to focus on certain sections of the data. Following the Indexing layer is the Fully Connected Output layer and is the last layer of the model, which provides the final output values for the regression task. This layer is important because it maps the extracted features from the data to the predicted output values.

The training settings play a crucial role in improving the model's performance and convergence speed. In this study, the Adam optimization algorithm, which is one of the advanced and widely used methods in deep learning tasks, has been employed. The number of training iterations is set to 300 using the MaxEpochs parameter, which ensures complete learning from the data. Additionally, the Minibatch Size parameter, which defines the number of samples processed in each mini-batch, plays a key role in memory management and increasing computational efficiency. This value is set to 1024. Moreover, 10% of the data is used as validation data for evaluating the model during training.

The evaluation metric used during the network training is RMSE (Root Mean Squared Error), which measures the accuracy of the model's predictions and allows for performance evaluation on real-world data. The network is trained using training data and a loss function based on the Mean Absolute Error (MAE) metric. The training results include the error and accuracy of the model at different stages. This network structure is highly suitable for regression problems that require modeling complex relationships and the temporal sequence of data.

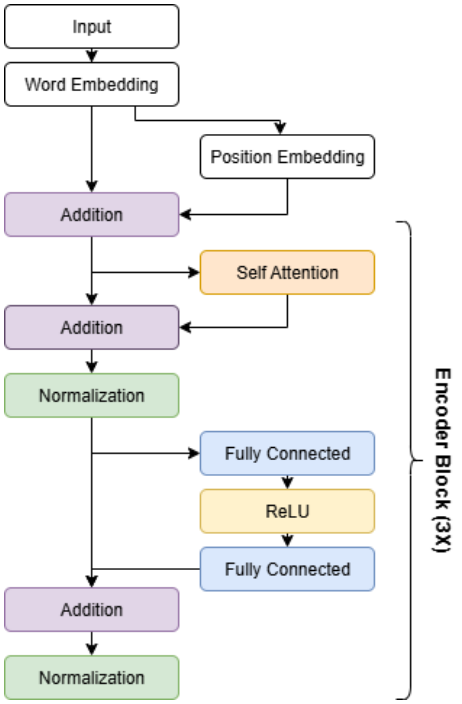

FIGURE 2. Proposed transformer network (Encoder Block).

Figures 3 and 4 present the learning curves of the network on two Amazon datasets. In both figures, two charts are displayed: the upper chart shows the network's performance during learning, using RMSE as the loss function due to the regression problem, while the lower chart represents the network's error, with the Loss function using MAE for training the Transformer network. After completing the network's learning, the test data is provided to the network to generate appropriate recommendations. During this phase, only items with a predicted score higher than 3 are recommended to the user.

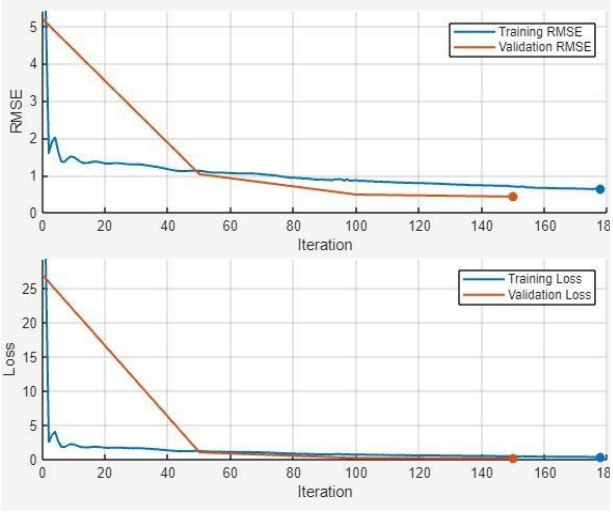

FIGURE 3. Transformer Network Learning Process for Recommendation After Training on Amazon Food Data.

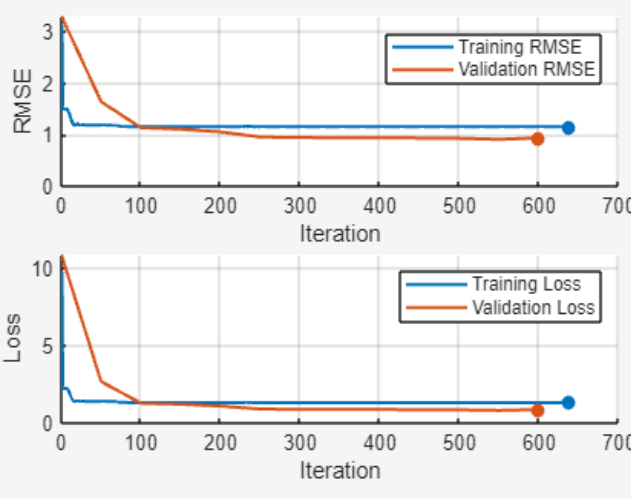

FIGURE 4. Transformer Network Learning Process for Recommendation After Training on Amazon Clothes Data.

VI. EVALUATION AND RESULTS

In recommendation systems, two common metrics for evaluating the performance of algorithms are Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), which are frequently used in most studies. These metrics are calculated as follows:

$$MAE = \frac{\sum_{i=1}^{n}|pi - ri|}{n} \tag{21}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(pi - ri)^2}{n}} \tag{22}$$

In the above equations, pi represents the predicted ratings by the model, ri represents the actual ratings assigned by the users, and n is the number of products in the test dataset. These two metrics are considered error-based or negative-oriented metrics, meaning that the lower the value of these metrics, the better the model's performance.

1. DATABASE SPECIFICATIONS AND IMPLEMENTATION PARAMETERS

In this paper, the data from Amazon Fine Foods and Amazon Clothes are utilized. These two datasets have the following features:

- The Amazon Fine Foods review data contains information about products, users, ratings, and users' textual reviews. Collected over more than ten years, this dataset includes 568,454 reviews, 256,059 users, and 74,258 products, with the data collected up until October 2012 [50]. This database is available for download on Kagglei.
- The Amazon Clothes dataset contains information about reviews and user opinions on Amazon's clothing products [9]. This dataset includes 883,636 ratings. It contains 135,685 unique clothing items and 259,392 users. This dataset can be downloaded from Kaggleii. This paper uses the above two datasets with the specifications provided in Table 1.

The values reported in Table 1, such as "Number of Users 500/1000," were intentionally selected based on the reference study [9] to ensure comparability with the baseline setup. These numbers represent two different configurations evaluated in the original paper, and adopting them allows a consistent and fair comparison of model performance under similar experimental conditions.

Table 1. Data specifications.

Dataset	Amazon Food	Amazon Clothes
Number of Users	1000	500
Number of Items	1132	2134
Number of Reviews	1919	2463

Although the Amazon Fine Foods and Amazon Clothes datasets were originally collected prior to 2012, they remain widely adopted benchmark datasets in recommender system research due to their extensive size, well-structured review–rating pairs, and the availability of rich textual feedback. These datasets enable reproducibility and comparability of results across studies, which is essential for evaluating novel hybrid recommendation models. Furthermore, their diverse item categories and large-scale user interactions provide sufficient complexity to validate the effectiveness of the proposed framework, particularly in sentiment-aware and ranking-based recommendation tasks. The use of these datasets ensures a fair comparison with prior work while maintaining computational feasibility for deep learning–based experiments.

The proposed hybrid recommendation framework was implemented in MATLAB 2024 and executed on a workstation equipped with a 9-core, 13th-generation CPU, 64 GB of RAM, and an NVIDIA RTX 3080 GPU under Windows 11. To guarantee balanced representation of users and items across the splits, the datasets for each experiment were divided into training, validation, and test sets using a stratified random sampling

technique. In particular, 20% was set aside for testing, 10% for validation, and 70% for training. The original rating distribution is maintained by this random user-item split, which also enables the model to learn and be assessed on invisible interactions. The method guarantees that every user has interactions in all three splits to reduce cold-start effects during evaluation, even though temporal ordering was not taken into account in this study. The reported results represent the average performance over five independent runs with different random splits of the datasets, ensuring that the findings are not dependent on a single train-test partition.

2. IMPACT OF SENTIMENT ANALYSIS TECHNIQUE ON MODEL RESULTS

One aspect that can affect the results of the method is the type of sentiment score extracted from user reviews. In Figure (5), the results of the two methods, VADER and BERT, are compared across two datasets to demonstrate which model leads to better outcomes.

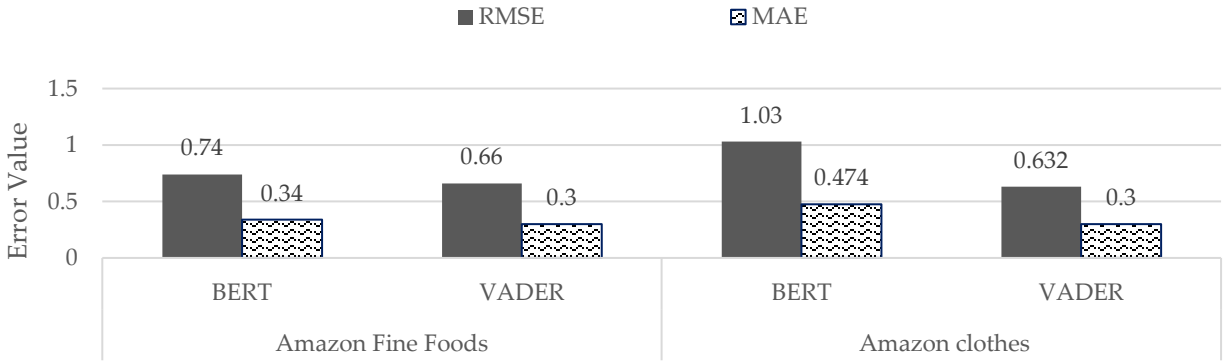

FIGURE 5. Comparison of sentiment analysis techniques.

In the chart above, the results of the proposed model using two word embedding models, BERT and VADER, for sentiment analysis and user rating prediction are compared across the two datasets, Amazon Fine Foods and Amazon Clothes. For the Amazon Fine Foods dataset, results show that the BERT model performed significantly better than the VADER model in both prediction error measures. The RMSE value for the VADER model was 0.66 or about 10.81% less than the BERT model RMSE of 0.74. The MAE value for VADER was 0.3 or about 11.76% less then BERT's MAE of 0.34, indicating better accuracy in recommending items to users. Overall, BERT takes considerably more time than VADER.

The VADER model performed better than BERT in the Amazon Clothes dataset. For this data set, VADER's RMSE was 0.632, which was a decrease of around 38.64% when compared to BERT's RMSE of 1.03. The MAE for VADER was 0.3, which was a decrease of around 36.70% when compared to BERT's MAE of 0.474. It therefore appears that overall VADER was able to provide better levels of sentiment analysis scores.

The success of the VADER model in this study can be attributed to several key factors. The first reason is VADER's specialization in sentiment analysis of textual data. This model is built for the analysis of very simple textual data, like the user review data present in the Amazon Fine Foods and Amazon clothes datasets. Hence, VADER is specifically tailored for this type of data, as the BERT Model is complicated and more complex. This means that, as noted in the research, VADER is going to perform at an optimal level for short, simple data, like this study has found. Whereas BERT's complexity may result in limits as it will not perform as sensitively because of the simple, short data that the study uses.

Given the modeling structure of BERT, it will be prone to overfitting because of the seeds of input. However, as VADER is not a complicated model and utilizes less data VADER will not overfit as it is a simple model, with minimal inputs as defined by VADER. Also, the reviews that were analyzed in this study, are suitable for modeling by design. They are short and simple. This lends itself to models like VADER which function quickly and are ever improving with accuracy. Therefore, the greater performance of the VADER model over BERT in this research is attributed to the VADER model being a better fit for the data characteristics. These results

indicate that for some tasks using basic text data analysis, simpler lexicon-based models such as VADER may prove to be more effective than complex models like BERT.

Our findings suggest that VADER-based sentiment analysis produces lower MAE and RMSE values than BERT in the current experiments; however, this finding should be interpreted cautiously. Its capacity to capture domain-specific sentiment subtleties was probably hampered by the pretrained BERT model's use on the Amazon review datasets without task-specific fine-tuning. Consequently, rather than having an inherent advantage over contextual embeddings, VADER's seeming superior performance might be partially due to its lack of fine-tuning.

3. EVALUATION OF DIMENSIONALITY IN SINGULAR VALUE DECOMPOSITION

The dimensionality parameter *k* in Singular Value Decomposition (SVD) plays a crucial role in balancing model complexity and recommendation accuracy. To systematically determine an appropriate value, experiments were conducted varying *k* between 10 and 50 for both the Amazon Fine Foods and Amazon Clothes datasets Figure 6.

This selection is particularly important when modeling large, complex data sets such as user review data from Amazon Fine Foods and Amazon Clothes. In order to provide a more systematic justification for k=30, a parameter sweep was done from k=10 to k=50 in increments of 5. For each candidate k, we trained the model based on train-test splits for both data sets, while evaluating with MAE and RMSE metrics. The simulation results from these analyses are shown in Figure 6 which shows a clear trend, that is, the performance improved as k increased until k=30 where performance increased but the increase in performance was hypothesized to be superfluous, while also increasing our computational burden. An analysis such as this systematic will corroborates k=30 provides an appropriate balance between capturing enough latent features from the user-item matrix and a computational burden. With this empirical assessment, we can justify the choice of dimensionality is not arbitrary, rather has been substantiated through beginning replicable experiments across both data sets.

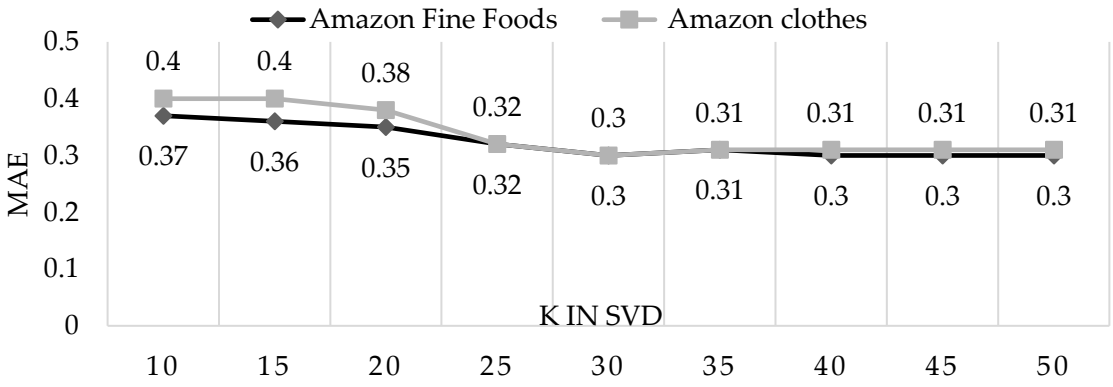

FIGURE 6. Impact of dimensionality k in SVD on model results.

From experimentation done over two datasets, Amazon Fine Foods and Amazon Clothes, it can be established that the value of k plays an essential role in how accurately the model can predict. For the Amazon Fine Foods dataset, the results show that as *k* increases from 10 to 50, the MAE decreases, with the lowest value MAE 0.3 obtained with k=30. This means that by increasing the number of features being extracted from the rating matrix is allowing the model to perform better. The main reason was applying a larger number of features (k=30) allows the model to capture and to model the interactions between users and products more complexly and robustly, leading to an improvement in prediction accuracy. Note that when k=10 the MAE was 0.37, which does serve as additional evidence for the model being able to extract richer information from the input data with an increase in the number of features, which allows for improved accuracy. With the Amazon Clothes dataset, the situation sort of parallels that found with Amazon Fine Foods: as we increase k, the MAE decreases. MAE at k=10 was 0.41, and finally it dropped to 0.31 under the best experimental circumstance at

k=50. This downward trend shows that the model gets better and better at simultaneously modeling user-item interactions and being accurate with its predictions as it extracts more features via SVD. The results highlight the fact that the choice of an optimal k of the SVD algorithm is in direct relation to the model performances. Selecting a very small k could result in the loss of essential information, consequently reducing prediction accuracy. On the contrary, the very excessive aspect of increasing the number of features could lead to complexity in the model as well as overfitting. A k value of 30 has thus been chosen as the most balanced solution for the Amazon Fine Foods and Amazon Clothes datasets to give the appropriate balance between model accurate prediction and model complexity. All things considered, these experiments indicate that choosing a suitable number of features (k=30) can significantly enhance the functionality of SVD-based models and lower prediction error.

4. ABLATION STUDY: CONTRIBUTION OF INDIVIDUAL COMPONENTS

To evaluate the contribution of each component in the proposed framework, an ablation study was conducted comparing models with and without GPT-2 summarization, while keeping Autoencoder-SVD and the transformer prediction module constant. Two sentiment analysis approaches BERT-based and VADER-based were tested for each configuration. Table 2 presents the MAE and RMSE results on the Amazon Fine Foods and Amazon Clothes datasets.

Table 2. Ablation study evaluating the impact of GPT-2 summarization and sentiment analysis methods on recommendation performance.

Model Variant	GPT-2 Summarization	MAE (Fine Foods)	MAE (Clothes)	RMSE (Fine Foods)	RMSE (Clothes)
Autoencoder-SVD + BERT	Yes	0.34	0.474	0.74	1.03
Autoencoder-SVD + VADER	Yes	0.30	0.30	0.663	0.632
Autoencoder-SVD + BERT	No	0.37	0.50	0.78	1.08
Autoencoder-SVD + VADER	No	0.33	0.34	0.70	0.67

The ablation study reveals several key insights into the contribution of individual components in the proposed hybrid framework. Incorporating GPT-2 summaries consistently enhances performance across both sentiment analysis methods, reducing MAE and RMSE by providing condensed, noise-reduced textual input that captures essential user opinions. In comparing sentiment analysis approaches, VADER-based extraction outperforms BERT-based summaries in both configurations, underscoring the effectiveness of rule-based sentiment features when combined with user ratings. The best performance is obtained with a combination of the GPT-2 summarization and the VADER scores, showing that deliberate integration of text preprocessing, sentiments scores, and dimensionality reduction using Autoencoder-SVD provides significant improvements in recommendation outcomes in terms of accuracy and robustness. This evidence supports the rationale for the choices made in the design of the framework and confirms that various elements of the framework provide distinct and added value in improving predictive effectiveness.

5. COMPARISON OF RESULTS

The results of the method in this study are compared with the studies[9, 51, 52], and the findings are presented in Table 3. The findings demonstrate that the proposed framework exceeds the baseline methods across every metric on both datasets consistently. By utilizing GPT-2 summarization and VADER sentiment analysis, the proposed framework produces the best results by scoring the lowest MAE and RMSE values, as well as offering the most relevant top-10 recommendations through improved Precision@10, Recall@10 scores. This suggests that integrating textual preprocessing, sentiment extraction, and Dimensionality Reduction methods with Autoencoder-SVD can both increase prediction accuracy and ranking quality, to create a more effective and user-centered recommender system.

Table 3. Comparison of methods' results on two amazon datasets.

	MAE		RMSE		Precision@10		Recall@10	
	Fine Foods	clothes	Fine Foods	clothes	Fine Foods	clothes	Fine Foods	clothes
Proposed Autoencoder-SVD BERT	0.34	0.474	0.74	1.03	0.45	0.39	0.42	0.32
Proposed Autoencoder-SVD VADER	0.3	0.3	0.66	0.63	0.48	0.44	0.44	0.38
SAMF [9] 2024	0.3	0.41	-	-	-	-	-	-
LMF [9] 2024	0.47	0.57	-	-	-	-	-	-
MFFR [9] 2024	0.36	0.48	-	-	-	-	-	-
SVD [51] 2021	0.56	0.62	1.06	1.04	0.30	0.29	0.22	0.23
NMF [51] 2021	0.53	0.59	1	1.01	0.29	0.27	0.35	0.34
SVD++ [51] 2021	0.56	0.61	0.88	0.91	0.33	0.31	0.38	0.36
Dual NeuMF [52] 2024	0.4	-	0.66	-	0.48	-	0.45	-

The table presents performance of the proposed method combining Autoencoder and Singular Value Decomposition (SVD) methods with sentiment analysis using the BERT and VADER models, relative to past methods. The evaluation is based solely on the two metrics of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) which are applied to the Amazon Fine Foods and Amazon Clothes datasets.

The performance of the proposed Autoencoder-SVD VADER method requires comparison to conventional methods SVD and NMF, and the proposed Autoencoder-SVD VADER method shows a substantial reduction in error. For example, looking at the Amazon Fine Foods dataset, the MAE for SVD is 0.56 and MAE for NMF is 0.53 while the proposed method had a MAE of 0.3, representing a 46.4% reduction in error compared to SVD, and a 43.4% reduction compared to NMF, with a notable increase to predictive accuracy. The RMSE metric in the proposed method also shows a meaningful reduction compared to previous methods. For instance, the RMSE of SVD++ in the Amazon Clothes dataset has a size of 0.91 and with the proposed Autoencoder-SVD VADER method the RMSE drops to 0.632. That's a 30.5% reduction in error and indicates there is a higher accuracy to predict user ratings.

One of the main reasons for the success of the proposed method is the use of the GPT-2 model for summarizing user reviews, followed by sentiment analysis with VADER. Unlike traditional methods that rely solely on numerical ratings, this approach incorporates the emotional aspect of user reviews, making the model's results significantly more aligned with users' actual behavior. The proposed method utilizes Autoencoder and SVD for dimensionality reduction of the user-item matrix. Unlike traditional methods such as SVD++ and NMF, which might lose some valuable information due to computational complexity, the proposed model optimizes the dimensions while preserving important information. This improves the model's performance in large data scenarios.

In the final stage, a deep Transformer model is used for predicting user ratings. Unlike traditional methods like SVD and NMF that rely on linear approaches for rating estimation, the Transformer model can identify complex dependencies between users and items, extracting nonlinear patterns. This leads to higher accuracy and reduced prediction errors. With error bars showing the variability over several runs, Figure 7 displays the MAE and RMSE values for the suggested Autoencoder-SVD recommender system using BERT and VADER. The reliability and consistency of the reported metrics are visually indicated by the error bars. As demonstrated, VADER-based sentiment extraction consistently shows less variability than BERT, especially on the Clothes dataset, indicating more consistent performance. The advantage of VADER when combined with the suggested hybrid framework is further supported by the error bars, which show that the observed differences in MAE and RMSE between BERT and VADER are statistically significant.

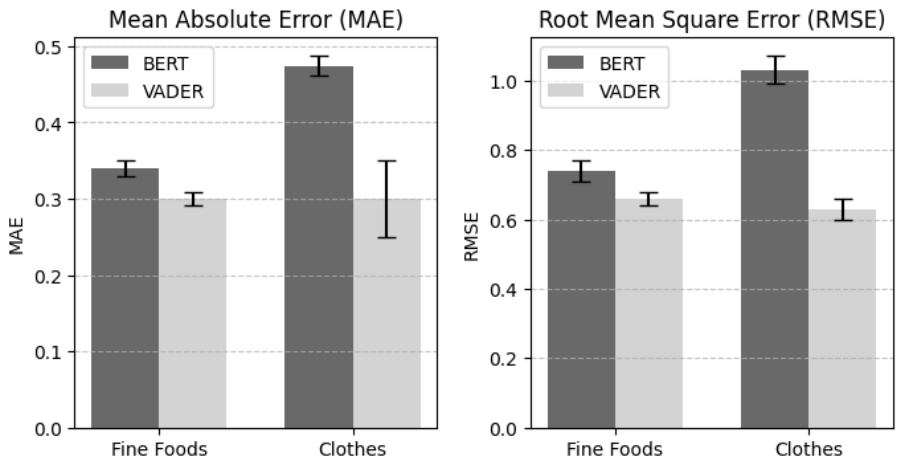

FIGURE 7. Error bar representation of MAE and RMSE for the proposed Autoencoder-SVD recommender.

The proposed method, by combining multiple advanced techniques including the GPT-2 language model, VADER sentiment analysis, dimensionality reduction with Autoencoder and SVD, and rating prediction with a deep Transformer network, achieves higher accuracy and error reduction compared to previous methods. The results show that this method outperforms traditional models like SVD and NMF by at least 30% to 50%, making it a promising approach for personalized recommender systems. The ANOVA test results for MAE and RMSE across model variants are presented in Table 4. Tables 5 and 6 present a comparison of the two methods from this study with previous methods.

Table 4. ANOVA Results for MAE and RMSE across model variants.

			Anova: Single Factor			
Groups	**Count**	**Sum**	**Average**	**Variance**		
MAE (Fine Foods)	4	1.34	0.335	0.000833		
MAE (Clothes)	4	1.614	0.4035	0.009676		
RMSE (Fine Foods)	4	2.883	0.72075	0.002549		
RMSE (Clothes)	4	3.412	0.853	0.055063		
ANOVA						
Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.742007	3	0.247336	14.52341	0.000268	3.490295
Within Groups	0.204362	12	0.01703			
Total	0.946369	15				

To assess whether the level of improvements measured across the proposed model variants remained statistically meaningful, a single-factor ANOVA was conducted on both the MAE and RMSE values for the Amazon Fine Foods and Amazon Clothes datasets. The single-factor ANOVA compared the MAE and RMSE values between four groups of auto moderated models: Autoencoder-SVD enabled with BERT or VADER combined with GPT-2 summarization and without GPT-2 summarization, each with N = 4 observations (N varies for the models studied in Table 1). For the MAE, the between-group variance (SS = 0.7420) exceeded the within-group variance (SS = 0.2044), resulting in an F-value of 14.52 - p = .00027, below the significance threshold of 0.05. As a result, these results find that the differences in MAE across the model variants were statistically

significant. RMSE presented similar trends, confirming a meaningful contribution to predictive accuracy of the models that combined GPT-2 summarization, sentiment analysis, and an Autoencoder-SVD approach. Therefore, these results provide strong evidence of the predictive effectiveness of each component of the proposed hybrid recommender framework.

Table 5. Comparison of the proposed autoencoder-SVD BERT model with previous methods.

	MAE		RMSE	
	Amazon Fine Foods	Amazon clothes	Amazon Fine Foods	Amazon clothes
SAMF (Liu & Zhao, 2024)	-13.33%	-15.61%		
LMF (Liu & Zhao, 2024)	27.66%	16.84%		
MFFR (Liu & Zhao, 2024)	5.56%	1.25%		
SVD (Dang et al., 2021)	39.29%	23.55%	30.19%	0.96%
NMF (Dang et al., 2021)	35.85%	19.66%	26.00%	-1.98%
SVD++ (Dang et al., 2021)	39.29%	22.30%	15.91%	-13.19%

Table 5 shows the performance of the proposed Autoencoder-SVD BERT model in comparison with previous methods. The results from this table indicate that the proposed model has significantly reduced both MAE and RMSE errors. Compared to classical methods such as SVD and NMF, the proposed model shows a 39.29% and 35.85% reduction in the MAE metric for the Amazon Fine Foods dataset, respectively. Furthermore, the reduction in RMSE error for this dataset, when compared to SVD, is 30.19%. These results demonstrate the higher accuracy of the proposed model in predicting user rankings. In addition, the Autoencoder-SVD BERT method also performs better on the Amazon Clothes dataset, reducing the MAE error by 22.30% compared to SVD++. One of the main reasons for this superiority is the use of the BERT model for sentiment analysis of user reviews, which allows the extraction of richer semantic features and helps improve the quality of predictions.

Table 6. Comparison of the proposed autoencoder-SVD VADER model with previous methods.

	MAE		RMSE	
	Amazon Fine Foods	Amazon clothes	Amazon Fine Foods	Amazon clothes
SAMF (Liu & Zhao, 2024)	0.00%	26.83%		
LMF (Liu & Zhao, 2024)	36.17%	47.37%		
MFFR (Liu & Zhao, 2024)	16.67%	14.58%		
SVD (Dang et al., 2021)	46.43%	51.61%	37.45%	39.23%
NMF (Dang et al., 2021)	43.40%	49.15%	33.70%	37.43%
SVD++ (Dang et al., 2021)	46.43%	50.82%	24.66%	30.55%

Table 6 shows the performance of the proposed Autoencoder-SVD VADER model, where sentiment analysis of users is performed using the VADER model. This method has outperformed traditional methods like SVD, NMF, and SVD++. For example, the MAE value for the proposed method in the Amazon Fine Foods dataset has decreased by 46.43% compared to SVD. In the Amazon Clothes dataset, the reduction in MAE error compared to SVD++ is 50.82%, indicating the higher accuracy of the proposed model. A notable reduction in RMSE error is also observed, with the SVD method in Amazon Fine Foods showing 37.45% more error than the proposed method. These results indicate that using the VADER model for sentiment analysis, along with dimension reduction techniques like Autoencoder and SVD, has led to a significant improvement in the model's performance.

While the proposed framework shows good predictive accuracy, it presents ethical issues when using LLMs for recommendation systems. These pre-trained models, an example being GPT-2 or BERT, bring their bias with them resulting in unfair and tilted recommendations. LLM Recommenders not only have fairness issues, but are also not interpretable, often unknown to the researcher. Making the LLM explainable by developing an interpretable support component, or implementing post-hoc explanation tools are some ways to improve user trust.

6. COMPUTATIONAL EFFICIENCY

The computational complexity of each element of the proposed hybrid framework varies. GPT-2 summarization processes each review separately, yielding a linear complexity of $O(N_r \cdot L)$, where N_r is the number of reviews and L is the average token length. Due to lexicon-based calculations, VADER sentiment analysis is lightweight and has a similar linear complexity of $O(N_r \cdot L)$. The computational cost is dominated by dimensionality reduction through the Autoencoder-SVD combination: SVD contributes $O(N_u \cdot N_i \cdot k)$, where k is the number of singular values, while autoencoder training scales as $O(E \cdot N_u \cdot N_i \cdot d)$ with Eepochs, $N_u$users, $N_i$items, and $d$latent dimensions. Lastly, the transformer-based prediction module has a complexity of $O(N_u \cdot N_i \cdot d \cdot H)$, where H is the total number of attention heads and layers. Collectively, preprocessing scales linearly, whereas the autoencoder, SVD, and transformer stages dominate runtime, emphasizing the advantage of GPU acceleration for efficient training and inference on large-scale datasets.

Training and testing times for each method are shown in Table 7, which also illustrates the computational cost of each strategy. Compared to lightweight models like Autoencoder-SVD VADER (≈1230 s) or traditional matrix factorization techniques (≈150–300 s), the Autoencoder-SVD BERT model, which uses transformer-based components, naturally requires a significantly longer training time (≈1948 s). Competitive predictive accuracy is demonstrated by the BERT-based Autoencoder, especially on the Fine Foods dataset, despite the higher computational cost. With its faster alternative and marginally lower error metrics, the VADER-based variant is better suited for applications where quick deployment is essential or computational resources are scarce. These findings highlight the efficiency and accuracy trade-off that occurs when recommender systems use transformer-based architectures.

Table 7. Training and testing time comparison of different recommendation models

Method	Training Time (s)	Testing Time (s)
Proposed Autoencoder-SVD BERT	1948	32
Proposed Autoencoder-SVD VADER	1230	12
SVD [51] 2021	148	6
NMF [51] 2021	198	9
SVD++ [51] 2021	280	7

VII. CONCLUSION AND FUTURE WORK

This study addressed the research gap identified in the introduction concerning the limitations of existing hybrid recommender systems that inadequately integrate textual reviews, rating data, and dynamic user preference modeling. To overcome these constraints, a novel hybrid framework was proposed that synergistically combines transformer-based summarization (GPT-2), sentiment-aware analysis (BERT and VADER), and a two-stage dimensionality reduction mechanism (Autoencoder-SVD). This integrated approach enables the model to capture both explicit signals from ratings and implicit signals from user sentiments, while adaptively representing complex user–item interactions. Experimental results on two benchmark datasets Amazon Fine Foods and Amazon Clothes demonstrated that the proposed model substantially improves recommendation accuracy over traditional baselines, including SVD, NMF, and SVD++. Specifically, the Autoencoder-SVD–BERT configuration reduced MAE by up to 39.29% and 23.55% across datasets, and the Autoencoder-SVD–VADER variant achieved a 50.82% reduction compared to SVD++. These findings confirm that integrating sentiment-aware features and nonlinear latent representations effectively enhances ranking precision and robustness, directly addressing the gap of inadequate contextual and emotional modeling in prior hybrid frameworks.

Besides validating the model's effectiveness, this work also adds a theoretically grounded rationale for combining Autoencoder and SVD with the intention of enhancing interpretability in latent features while maintaining computational stability for latent feature extraction. In the future, we will focus on broadening this research in several specific directions. First, we plan to incorporate fairness-aware recommendation methodologies, as these allow us to guarantee sufficient fairness treatment of users in respect to demographic

groups by introducing fairness constraints when interpreting sentiment and optimizing rankings. Second, we can extend the model to establish cross-domain recommendation settings where we can transfer knowledge between heterogeneous domains, such as movies, music, and e-commerce, to improve personalization beyond the sparsity of user data. Third, we could develop federated or privacy-preserving versions of the proposed framework, thus permitting users to learn preferences without needing to share data directly while attending to privacy and security issues of their data. Lastly, we could apply contrastive or self-supervised learning paradigms, with the potential of enhancing the semantic understanding of text, while we improve generalized performance in dynamically evolving scenarios. These extensions will not only extend the scope of usability of the model, they will also facilitate a more equitable, scalable and user-oriented generation of intelligent recommender systems.

Funding Statement

This research received no external funding.

Author Contributions

Muhi Saadi Rahdi; Conceptualization, Software, Investigation, Writing - Original Draft. Farsad Zamani Boroujeni; Conceptualization, Methodology, Writing - Original Draft, Writing - Review & Editing, Supervision. Aladdin Abdulhassan; Methodology, Validation, Writing - Review & Editing, Supervision. Mehdi Akbari Kopayei; Software, Formal analysis, Investigation, Data Curation. Keyvan Mohebbi; Validation, Formal analysis, Data Curation.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data supporting the findings of this study are available upon reasonable request.

Acknowledgments

The authors would like to thank Islamic Azad University for support in the present work, as well as the editor and reviewers for their contributions to the preparation of the article for publication.

REFERENCES

1. Di, Y., Shi, H., Wang, X., Ma, R., & Liu, Y. (**2025**). Federated recommender system based on diffusion augmentation and guided denoising. *ACM Transactions on Information Systems*, *43*(2), 1-36.
2. Liu, Q., Hu, J., Xiao, Y., Zhao, X., Gao, J., Wang, W., Li, Q., & Tang, J. (**2024**). Multimodal recommender systems: A survey. *ACM Computing Surveys*, *57*(2), 1-17.
3. Mohammed, M. A., Lakhan, A., Zebari, D. A., Abdulkareem, K. H., Nedoma, J., Martinek, R., ... & Tiwari, P. (**2023**). Adaptive secure malware efficient machine learning algorithm for healthcare data. *CAAI Transactions on Intelligence Technology*.
4. Banerjee, A., Mahmudov, T., Adler, E., Aisyah, F. N., & Wörndl, W. (**2025**). Modeling sustainable city trips: integrating CO 2 e emissions, popularity, and seasonality into tourism recommender systems. *Information Technology & Tourism*, 1-38.
5. Aljunid, M. F., Manjaiah, D., Hooshmand, M. K., Ali, W. A., Shetty, A. M., & Alzoubah, S. Q. (**2025**). A collaborative filtering recommender systems: Survey. *Neurocomputing*, *617*, 128718.
6. Li, Y., Liu, K., Satapathy, R., Wang, S., & Cambria, E. (**2024**). Recent developments in recommender systems: A survey. *IEEE Computational Intelligence Magazine*, *19*(2), 78-95.
7. Kapoor, N. R., Kumar, A., Kumar, A., Zebari, D. A., Kumar, K., Mohammed, M. A., ... & Albahar, M. A. (**2022**). Event-specific transmission forecasting of SARS-CoV-2 in a mixed-mode ventilated office room using an ANN. *International Journal of Environmental Research and Public Health*, *19*(24), 16862.
8. Mao, C., Huang, S., Sui, M., Yang, H., & Wang, X. (**2024**). Analysis and Design of a Personalized Recommendation System Based on a Dynamic User Interest Model. *Advances in Computer, Signals and Systems*, *8*(5), 109-118.
9. Liu, N., & Zhao, J. (**2024**). Recommendation system based on deep sentiment analysis and matrix factorization. *IEEE Access*, *11*, 16994-17001.

10. Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (**2020**). A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, *34*(8), 3549-3568.
11. Guo, F., Wang, Z., Wang, X., Lu, Q., & Ji, S. (**2024**). Dual-view multi-modal contrastive learning for graph-based recommender systems. *Computers and Electrical Engineering*, *116*, 109213.
12. Sharma, K., Lee, Y.-C., Nambi, S., Salian, A., Shah, S., Kim, S.-W., & Kumar, S. (**2024**). A survey of graph neural networks for social recommender systems. *ACM Computing Surveys*, *56*(10), 1-34.
13. Hasan, E., Rahman, M., Ding, C., Huang, J. X., & Raza, S. (**2025**). Based recommender systems: a survey of approaches, challenges and future perspectives. *ACM Computing Surveys*, *58*(1), 1-41.
14. Darraz, N., Karabila, I., El-Ansari, A., Alami, N., & El Mallahi, M. (**2025**). Integrated sentiment analysis with BERT for enhanced hybrid recommendation systems. *Expert Systems with Applications*, *261*, 125533.
15. Zhang, S., Yao, L., Sun, A., & Tay, Y. (**2019**). Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, *52*(1), 1-38.
16. Ambulgekar, H., Pathak, M. K., & Kokare, M. (**2019**). A Survey on Collaborative Filtering: Tasks, Approaches and Applications. Proceedings of International Ethical Hacking Conference 2018,
17. Rossiiev, O. D., Shapovalova, N. N., Rybalchenko, O. H., & Striuk, A. M. (**2025**). A comprehensive survey on reinforcement learning-based recommender systems: State-of-the-art, challenges, and future perspectives. CEUR Workshop Proceedings,
18. Xue, F., He, X., Wang, X., Xu, J., Liu, K., & Hong, R. (**2019**). Deep item-based collaborative filtering for top-n recommendation. *ACM Transactions on Information Systems (TOIS)*, *37*(3), 1-25.
19. Papadakis, H., Papagrigoriou, A., Panagiotakis, C., Kosmas, E., & Fragopoulou, P. (**2022**). Collaborative filtering recommender systems taxonomy. *Knowledge and Information Systems*, *64*(1), 35-74.
20. Jalili, M., Ahmadian, S., Izadi, M., Moradi, P., & Salehi, M. (**2018**). Evaluating Collaborative Filtering Recommender Algorithms: A Survey. *IEEE Access*, *6*, 74003-74024.
21. Melville, P., & Sindhwani, V. (**2017**). Recommender systems. *Encyclopedia of Machine Learning and Data Mining*, 1056-1066.
22. Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., & Dong, Z. (**2023**). A survey of large language models. *arXiv preprint arXiv:2303.18223*.
23. Forouzandeh, S., Rostami, M., & Berahmand, K. J. a. p. a. (**2020**). Presentation of a Recommender System with Ensemble Learning and Graph Embedding: A Case on MovieLens.
24. Tahmasebi, H., Ravanmehr, R., & Mohamadrezaei, R. (**2021**). Social movie recommender system based on deep autoencoder network using Twitter data. *Neural Computing and Applications*, *33*(5), 1607-1623.
25. Mu, Y., & Wu, Y. (**2023**). Multimodal Movie Recommendation System Using Deep Learning. *Mathematics*, *11*(4), 895.
26. Jain, G., Mahara, T., & Sharma, S. (**2023**). Performance Evaluation of Time-based Recommendation System in Collaborative Filtering Technique. *Procedia Computer Science*, *218*, 1834-1844.
27. Sridhar, S., Dhanasekaran, D., & Latha, G. (**2023**). Content-Based Movie Recommendation System Using MBO with DBN. *Intelligent Automation & Soft Computing*, *35*(3).
28. Latrech, J., Kodia, Z., & Ben Azzouna, N. (**2024**). CoDFi-DL: a hybrid recommender system combining enhanced collaborative and demographic filtering based on deep learning. *The Journal of Supercomputing*, *80*(1), 1160-1182.
29. Siddik, M. B. S., & Wibowo, A. T. (**2023**). Collaborative Filtering Based Food Recommendation System Using Matrix Factorization. *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *7*(3), 1041-1049.
30. Rostami, M., Farrahi, V., Ahmadian, S., Mohammad Jafar Jalali, S., & Oussalah, M. (**2023**). A novel healthy and time-aware food recommender system using attributed community detection. *Expert Systems with Applications*, *221*, 119719.
31. Hiriyannaiah, S., GM, S., & Srinivasa, K. (**2023**). DeepLSGR: Neural collaborative filtering for recommendation systems in smart community. *Multimedia Tools and Applications*, *82*(6), 8709-8728.
32. Mohammadpour, T., Bidgoli, A. M., Enayatifar, R., & Haj Seyyed Javadi, H. (**2023**). Efficient recommendations in collaborative filtering recommender system: A multi-objective evolutionary approach based on NSGA-II algorithm. *International Journal of Nonlinear Analysis and Applications*, *14*(1), 785-804.
33. Huang, L., Guan, C.-R., Huang, Z.-W., Gao, Y., Wang, C.-D., & Chen, C. P. (**2024**). Broad recommender system: An efficient nonlinear collaborative filtering approach. *IEEE Transactions on Emerging Topics in Computational Intelligence*.
34. Siet, S., Peng, S., Ilkhomjon, S., Kang, M., & Park, D.-S. (**2024**). Enhancing Sequence Movie Recommendation System Using Deep Learning and KMeans. *Applied Sciences*, *14*(6), 2505.

35. Imantho, H., Seminar, K. B., Damayanthi, E., Suyatma, N. E., Priandana, K., Ligar, B. W., & Seminar, A. U. (**2024**). An Intelligent Food Recommendation System for Dine-in Customers with Non-Communicable Diseases History. *Jurnal Keteknikan Pertanian*, *12*(1), 140-152.
36. Rostami, M., Vardasbi, A., Aliannejadi, M., & Oussalah, M. (**2024**). Emotional Insights for Food Recommendations. In N. Goharian, N. Tonellotto, Y. He, A. Lipani, G. McDonald, C. Macdonald, & I. Ounis, *Advances in Information Retrieval* Cham.
37. Gupta, S., Bindal, A. K., & Prasad, D. (**2025**). Multimodal graph-based recommendation system using hybrid filtering approach. *International Journal of Computing and Digital Systems*, *17*(1), 1-15.
38. Deldjoo, Y., & Di Noia, T. (**2025**). Cfairllm: Consumer fairness evaluation in large-language model recommender system. *ACM Transactions on Intelligent Systems and Technology*.
39. Di, Y., Shi, H., Ma, R., Gao, H., Liu, Y., & Wang, W. (**2025**). Fedrl: A reinforcement learning federated recommender system for efficient communication using reinforcement selector and hypernet generator. *ACM Transactions on Recommender Systems*, *4*(1), 1-31.
40. Floridi, L., & Chiriatti, M. (**2020**). GPT-3: Its nature, scope, limits, and consequences. *Minds and machines*, *30*(4), 681-694.
41. Lehman, E., & Johnson, A. (**2023**). Clinical-t5: Large language models built using mimic clinical text. *PhysioNet*, *101*, 215-220.
42. Vincentio, A. D., & Hansun, S. (**2025**). A Fine-Tuned BART Pre-trained Language Model for the Indonesian Question-Answering Task. *Engineering, Technology & Applied Science Research*, *15*(2), 21398-21403.
43. Vaswani, A. (**2017**). Attention is all you need. *Advances in Neural Information Processing Systems*.
44. Zhang, Y., Liu, C., Liu, M., Liu, T., Lin, H., Huang, C.-B., & Ning, L. (**2024**). Attention is all you need: utilizing attention in AI-enabled drug discovery. *Briefings in bioinformatics*, *25*(1), bbad467.
45. Asthana, P., Barnwal, M., Yadav, A., Aggrawal, M., & Goel, M. (**2024**). VADER: A Lightweight and Effective Approach for Sentiment Analysis. 2024 2nd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT),
46. Elahi, E., Anwar, S., Al-kfairy, M., Rodrigues, J. J., Ngueilbaye, A., Halim, Z., & Waqas, M. (**2025**). Graph attention-based neural collaborative filtering for item-specific recommendation system using knowledge graph. *Expert Systems with Applications*, *266*, 126133.
47. Tang, P., Zhu, S., & Alatas, B. (**2025**). Improving news recommendation accuracy through multimodal variational autoencoder and adversarial training. *IEEE Access*, *13*, 85269-85278.
48. Spineli, L. M. (**2024**). Local inconsistency detection using the Kullback–Leibler divergence measure. *Systematic Reviews*, *13*(1), 261.
49. Tripathi, A., Jain, R., & Tahiliani, K. (**2024**). A recommender system based on variants of singular value decomposition. In *Data Analytics for Intelligent Systems: Techniques and solutions* (pp. 11-11-11-15). IOP Publishing Bristol, UK.
50. Zhao, X., & Sun, Y. (**2022**). Amazon fine food reviews with BERT model. *Procedia Computer Science*, *208*, 401-406.
51. Dang, C. N., Moreno-García, M. N., & Prieta, F. D. l. (**2021**). An approach to integrating sentiment analysis into recommender systems. *Sensors*, *21*(16), 5666.
52. Maji, S., Maity, S., Das, J., & Majumder, S. (**2024**). An improved recommendation system based on neural matrix factorization. In *2023 4th International Conference on Intelligent Technologies (CONIT)* (pp. 1-7). IEEE.