Applied Soft Computing Journal 166 (2024) 112140

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Elevating recommender systems: Cutting-edge transfer learning and embedding solutions

Aamir Fareed [a], Saima Hassan [a,*], Samir Brahim Belhaouari [b,*], Zahid Halim [c]

[a] Institute of Computing, Kohat University of Science and Technology, Kohat, 26000, Pakistan
[b] Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Doha, P.O. Box 3411, Qatar
[c] Faculty of Computer Science and Engineering, GIK Institute of Science and Technology, Topi, Pakistan

ARTICLE INFO

Keywords:
Recommender systems
Deep transfer learning
Multimodal embedding
Data sparsity
Cold-start problem

ABSTRACT

In today's information age and connected economy, Recommender Systems (RS) plays a vital role in managing information overload and delivering personalized suggestions to users. This paper introduces a multistage model that leverages multimodal data embedding and deep transfer learning to accurately capture user preferences and item characteristics, resulting in highly tailored recommendations. A key innovation in this model is the incorporation of an image dataset in the second phase, which addresses cold-start problems for new items by providing additional visual context. Our approach excels in overcoming challenges related to data sparsity and cold-start issues, thereby providing users with realistic and relevant product recommendations. To validate the effectiveness of the proposed model, we conducted extensive evaluations using three diverse datasets: data from Brazilian e-commerce platforms, the MovieLens 1M dataset, and the Amazon Product Review dataset. These evaluations involved comprehensive comparisons with standard RS methods to assess performance improvements. The results indicate that our proposed model significantly outperforms traditional RS techniques in terms of accuracy and reliability. Our model provides more accurate and meaningful recommendations by effectively addressing issues such as cold-start and data scarcity. Specifically, the model achieved Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) scores of 0.5883 and 0.4012, respectively, which demonstrate its superior performance metrics across all datasets tested.

1. Introduction

As the information on the internet grows at an unprecedented rate, users inevitably become less capable of finding the data relevant to their preferences [1]. To address this issue, researchers have explored the area of recommender systems (RS) that automatically provide recommendations based on personal preferences [2]. RS make shopping more convenient for customers, increases sales, and brings extra traffic to eCommerce websites [3]. RS reduce the time users spend browsing large and unexpected collections of items by proposing relevant items according to their preferences [4]. Researchers have developed many algorithms to provide precise and practical tailored suggestions based on user choices. Various algorithms, such as collaborative filtering (CF) systems, content-based (CB) systems, and hybrid systems, are commonly used in RS [5]. While CF approaches rely on identifying common preferences among users with similar interests, CB methods evaluate the features of items that have piqued a user's interest. Hybrid RS integrates CF and CB algorithms to generate personalized recommendations [6]. However, the computational cost and algorithmic complexity may rise if CB and CF approaches are used to solve cold-start and sparsity concerns [7].

While RS have advanced significantly, they still struggle with serious challenges, including data sparsity and the cold-start problem. When rating data is insufficient or unavailable for specific users or items, the quality of the suggestions may be impacted [8]. Despite many proposals for methods to solve these problems, they are still not always successful, especially when there are many users and items, but few of them have been rated [9]. Profiling users and items is intricate and should consider both explicit and implicit aspects [10]. Still, traditional recommendation techniques often do not utilize this data, leading to unreliable suggestions. Deep learning has significantly influenced information retrieval and RS [11]. Many RS researchers have recently used deep neural networks (DNN) to boost the quality of suggestions. Nevertheless, as DNNs anticipate user and item attributes based on these interactions, they are constrained when dealing with sparse user-item interactions [12]. Another method, called Deep Transfer Learning (DTL), uses deep learning models already trained on smaller datasets to

* Corresponding authors.
E-mail addresses: cs001af@kust.edu.pk (A. Fareed), saimahassan@kust.edu.pk (S. Hassan), sbelhaouari@hbku.edu.qa (S. Brahim Belhaouari).

https://doi.org/10.1016/j.asoc.2024.112140
Received 14 March 2024; Received in revised form 15 June 2024; Accepted 14 August 2024
Available online 24 August 2024

transfer information learned through training these models on broader, relevant datasets [13]. This approach can leverage information from previously trained deep models, improving prediction accuracy and speed [14]. Though DTL has shown promising results in NLP, computer vision, and other ML domains, it is yet to be fully adopted in RS [15].

Deep learning has created many powerful and accurate RS. For example, user preferences have been captured more comprehensively using Convolutional Neural Networks (CNNs) [16]. Additionally, Recurrent Neural Networks (RNNs) have been applied to item-based recommendations to capture time dynamics in user behaviour [17]. Natural language processing (NLP) models are also researched for personalized recommendations on user-item interactions [18]. Although the increased costs and complexities of the deep learning strategies might not be well-received by RS, these strategies were employed in NLP contexts [19].

This paper introduces a new hybrid model for recommendation systems, named Deep Transfer Learning and Multimodal Embedding (DTLME). DTLME is primarily concerned with effectively solving the cold-start issues of users and items. DTLME also uses a cooperative filtering method for cold-start problems concerning new users. It covers learning for user/item embedding according to all ratings users submit. This is because the model can guarantee to represent user preferences and to suggest highly relevant items. The DTLME model is highly scalable as it does not require a large amount of data and allows easy feature extraction from many different sources. By implementing both deep transfer learning and collaborative filtering, the model can generate recommendations superior to many traditional approaches. To mitigate the issue of sparse training data, multimodal embedding (ME) is applied to extract dense feature vectors. Given these extensive and complex user-item profiles, a user-based collaborative filtering (UBCF) approach determines the k closest users for each user. The following are the main contributions of this study:

- Developing an item-item similarity measure that utilizes visual similarities between items, deep transfer learning may be used to solve the issue of new item-cold start
- Using multimodal embedding to create a dense user as well as item matrices, which will fix the rating matrix's sparsity problem
- Creating a hybrid recommendation framework that generates top-n suggestions tailored to each user using K-nearest neighbours
- Creating a list of the top items by merging the top items from either the item-item similarity matrix as well as the similarity clusters made throughout the recommendation process.

The structure of the article is as follows: Section 2 presents a literature review of the relevant studies. Section 3 outlines the proposed DTLME model and methodology. Section 4 presents the experimental setup, while Section 5 reports the results of this research. Finally, the article is concluded in Section 6.

2. Literature review

This section reviews existing Recommender Systems (RS) and addresses the cold-start problem within RS. RS commonly utilize two primary filtering approaches: Content-Based (CB) Filtering and Collaborative Filtering (CF), which employ algorithms to provide suggestions based on consumer data. While CB systems recommend items based on content similarity, CF systems rely on user collaboration to identify common preferences. Hybrid filtering systems combine the advantages of CB and CF to enhance recommendation accuracy. However, when new items are introduced or new users interact with the system, RS encounter the cold-start problem, which manifests as challenges in making accurate recommendations due to a lack of prior data. The cold-start problem can be categorized into two distinct issues: new user and new item cold-starts. The new user cold-start is particularly challenging as the system lacks sufficient knowledge about the individual to provide accurate suggestions. Researchers [20] have explored various explicit and implicit data collection approaches to address this issue, including meta-learning, active learning models, doc2vec, and demographic information. However, these methods often suffer from high computational complexity and domain relevancy constraints. To mitigate data sparsity, several user-based collaborative filtering modifications have been proposed, such as singular vector decomposition, similarity tests, and recursive prediction algorithms [21]. Item-based similarity metrics have also been used for cold-start problems, but this is relatively ineffective in solving this problem. Clustering algorithms are proven to provide high success in predicting future values because they group those items that are similar in nature [22]. For example, hierarchical clustering algorithms will group users based on social information and then give movie recommendations.

Cluster-based methods are better in terms of both the identification of users with high affinities to target items and addressing the challenges which arise due to higher dimensionality and sparseness. For instance, Hierarchical clustering algorithms have been used to cluster individuals based on their social traits, whereas rankings are based on collaborative filtering strategies. Some researchers have simply modified the "k" parameter in clustering algorithms in a way to handle individual preference, and others have used social network analysis to determine the quality of a recommendation [23]. The integration of rating matrices with auxiliary side information has proved effective in boosting RS performance and quality [24]. Matrix factorization (MF) methods have integrated rating matrices with side information embeddings, and social data to improve overall performance by merging user–and item-item similarity matrices [25]. Neural Social Recommendation models, combining MF principles with social data and user embeddings, have been developed to reveal latent attributes for accurate predictions [26]. Moreover, deep learning-based recommendation systems have been further employed with social data sparseness and social inconsistencies in the collaborative filtering framework [26].

Graph Neural Networks (GNNs) have been utilized to capture user-item interactions based on user-item graphs and latent similarities, improving recommendation accuracy [27]. Knowledge Graph Embedding (KGE) models enhance input matrices by learning embeddings from entities and relationships in knowledge graphs, addressing sparsity problems [28]. Multi-relational auto-encoders (MRAE) employ neural networks to apply multi-relational data, revealing associations between users and items [29]. Deep collaborative filtering (DCF) addresses sparsity and cold-start issues by using simple vectors, side information, and stacked denoising auto-encoders [30]. Despite their effectiveness, these approaches often rely on supplementary data, which existing systems may inadequately utilize, limiting recommendation quality.

The proposed Deep Transfer Learning Multimodal Embedding (DTLME) model distinguishes itself from existing recommendation systems (RS) through several key innovations. First, the model integrates multimodal data in DTLME, where visual product features are represented from images, session logs, and social network embeddings [31]. This fusion of diverse data types helps a model create more holistic user and item profiles, therefore really boosting the accuracy of recommendations. Traditional RS models rely only on user-item interactions, which inherently limit their capacity to capture user preferences and item characteristics [32]. Moreover, the DTLME model intensely exploits transfer learning and leverages the latent features from pre-trained models like VGG-16 [33] for fine-tuning in the e-commerce domain. The method allows the model not to need much training because the pre-existing knowledge in large datasets reduces the need for extensive training; thus, it addresses the cold-start problem considerably. The use of deep transfer learning makes the DTLME model adapt to new information and, therefore, versatile and robust in different e-commerce contexts [34].

Table 1
Comparison of DTLME with existing models.

Feature	Traditional models	DTLME model
Data modalities	Single (textual or visual)	Multimodal (textual and visual)
Cold-Start handling	Limited (user or item-based)	Hybrid (collaborative + DTL)
Transfer learning utilization	Minimal	Extensive (fine-tuning pre-trained models)
Scalability	Moderate	High (efficient with sparse data)
Computational efficiency	High resource demand	Optimized with pre-trained models

Moreover, the model features a hybrid framework of collaborative filtering and content-based techniques. The multimodal embeddings induce several kinds of similarity matrices, which are used in both offline feature learning and online recommendation stages [35]. Such a combination ensures that this DTLME model can provide accurate and scalable recommendations with balanced strengths both in collaborative and content-based approaches. The DTLME model performance has been evaluated with the Brazilian E-Commerce dataset. Experimental results demonstrate that DTLME outperforms traditional RS models, including Collaborative Singular Value Decomposition (CSSVD), Tensor Factorization (TF), and Bayesian Personalized Ranking (BPR), across multiple performance metrics such as precision, recall, F1-score, Mean Absolute Error (MAE), and Root Mean Square Error (RMSE). The model's superior handling of data sparsity and cold-start issues marks a significant advancement over existing technologies, showcasing its potential to enhance recommendation systems in the e-commerce industry. To provide a clearer comparison, Table 1 summarizes the key differences and advancements of the DTLME model over existing approaches.

3. Proposed methodology

Fig. 1 illustrates the dual-phase methodology adopted in our research. The first stage involves a two-step process with offline feature learning. In our method, we suggest a two-stage procedure that improves the quality of the product recommendations. First, the latent feature vector is formed for every item from an image dataset according to transfer learning using OxfordNet architecture of VGG-16 convolutional neural networks [36]. This vector, which has the visual information for each item is then utilized as an Item Similarity Matrix based on the cosine similarity. During the second step, a multimodal data embedding approach is employed to provide latent matrices for both users and user-user and also user item similarities on top of e-commerce Brazilian datasets [36]. This approach involves projecting the users and all products into a common feature space that embeds their natural affinities. For this purpose, we employ a hybrid of matrix factorization and also neural network-based techniques. The obtained similarity matrices serve as the input data for a hybrid recommendation model that fuses both content-based and collaborative filtering techniques. These matrices are merged to provide the input for the recommender model in the next online phase that will generate relevant recommendations based on the active users of the system.

This section represents stage 1. In this section, a very comprehensive description of the proposed DTLME model is provided. First, we outline the general structure of DTLME and then we provide detailed explanations for its elements.

3.1. The design of the suggested model

The model is very interactive, quick and also easy to access. With the combination of user feedback and also item suggestions, this model can recognize as well as respond to the preferences. In addition, it can also adapt to the many modifications of user preferences in terms of time. All these factors contribute to a very personalized and also user-specific experience. This model enables the consumers to search and choose their favourite products with minimal effort of time consumption.

3.1.1. Deep transfer learning

To expedite the development process, we utilize transfer learning [13], a technique that leverages pre-trained models to accelerate and enhance model training for specific tasks. Transfer learning allows developers to use a pre-trained Convolutional Neural Network (CNN) model, such as VGG-16, and fine-tune it with domain-specific data, significantly reducing the time and computational resources required to develop a high-performing model. The VGG-16 model is selected due to its high accuracy and widespread use in image classification tasks [37]. It is pre-trained on the ImageNet dataset, which contains millions of labelled images across thousands of categories. This extensive pre-training allows the model to learn rich feature representations that can be transferred to new tasks. The original output classification layer of the VGG-16 model, which is specific to the ImageNet categories, is removed. All other layers of the VGG-16 model are frozen, meaning their weights are not updated during the fine-tuning process. This preserves the knowledge acquired during the pre-training phase. The VGG-16 model consists of 16 layers, including 13 convolutional layers (CL) and 3 fully connected layers (FC), interspersed with pooling layers (PL) for down-sampling. The architecture is depicted in Fig. 2.

To adapt the VGG-16 model to our specific dataset, new fully connected layers are added on top of the pre-trained layers. These new layers are initialized with random weights and made trainable. This setup allows the model to learn task-specific features during the fine-tuning process. The model is fine-tuned using our domain-specific dataset, which consists of product images from the e-commerce platform. The training process after adding the new layers to the network is termed fine-tuning. This allows the newly added layers to be updated while keeping all the already pre-trained layers fixed. This method enables a quick adaptation to the new task but somehow suppresses the learned robust ImageNet feature representation. After these fine-tuned training processes, VGG-16 is then used to extract latent feature vectors of each of the product images; these vectors describe visual characteristics such as texture, colour, and shape. The feature vectors would be flattened into a one-dimensional array and then normalized to an equal scale. These vectors are referred to as Items Feature Vectors. Similarity matrices were produced using item-item similarity and then saved in the recommendation model for future use [38]. The VGG16 model, which includes frozen and trainable pre-trained layers, was used for this purpose, and its framework is illustrated in Fig. 3. This approach enables us to capture the underlying relationships between items based on visual features, providing a more accurate and comprehensive approach to RS. By utilizing a pre-trained model, we can efficiently extract item-based latent features and calculate similarity matrices, reducing computational costs and improving performance.

A CNN model is utilized to classify input images to enhance the accuracy of the recommendations. This helps to reduce the algorithm's processing time, as only items belonging to the predicted class are considered in similarity calculations. Visual features compute the item-item similarity using cosine similarity measures, resulting in a similarity matrix.

3.2. Multimodal embedding

The extraction of relevant features from user and item data is a critical task under the ME module. This module comprises various smaller modules such as User Features Extraction, Item Feature Extraction, feature reduction technique, and the generation of user-user and user-item co-relation matrices. The purpose of this feature extraction process is to

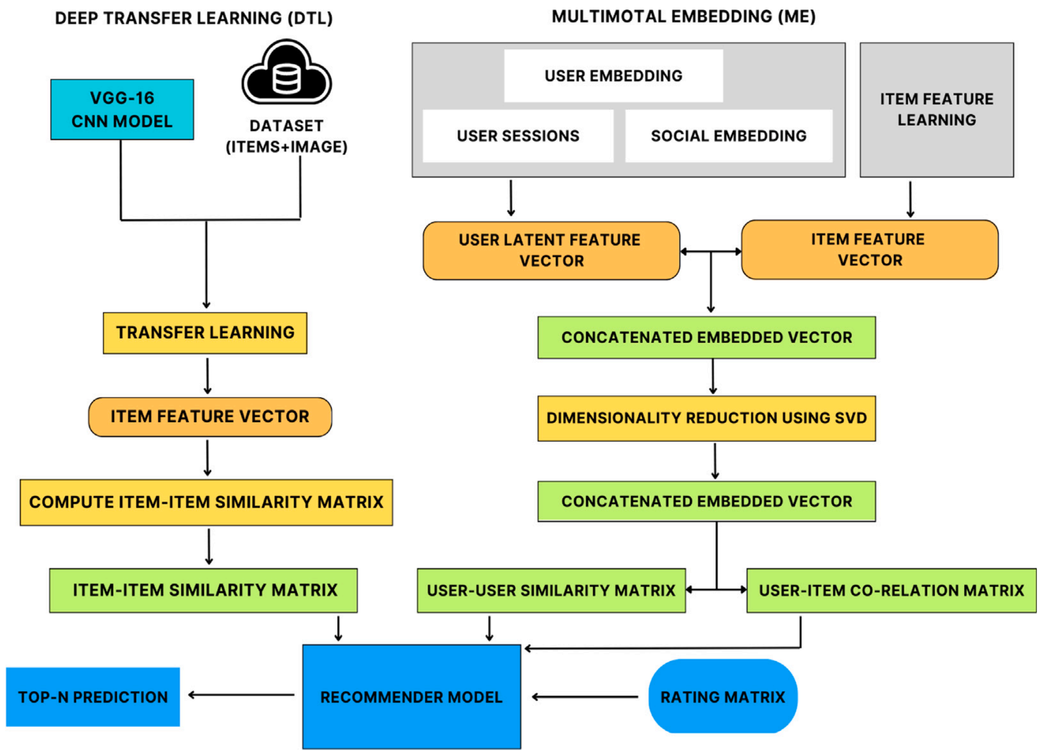

Fig. 1. Structure of the DTLME model that has been proposed for feature engineering.

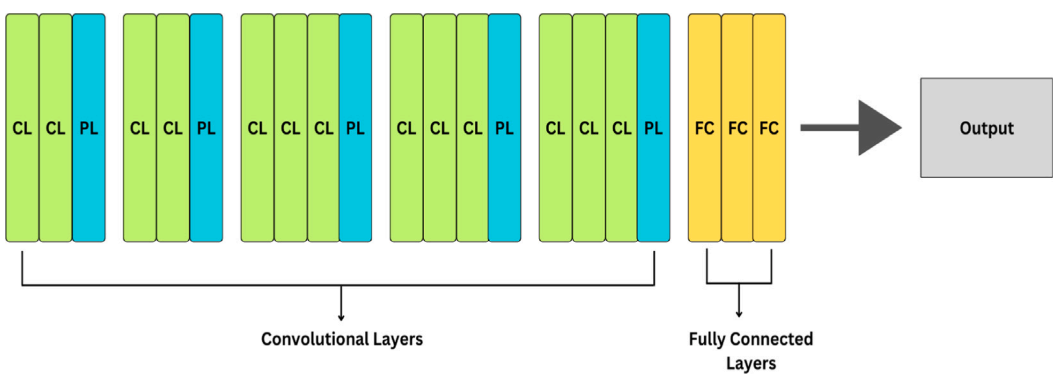

Fig. 2. VGG16 generic model.

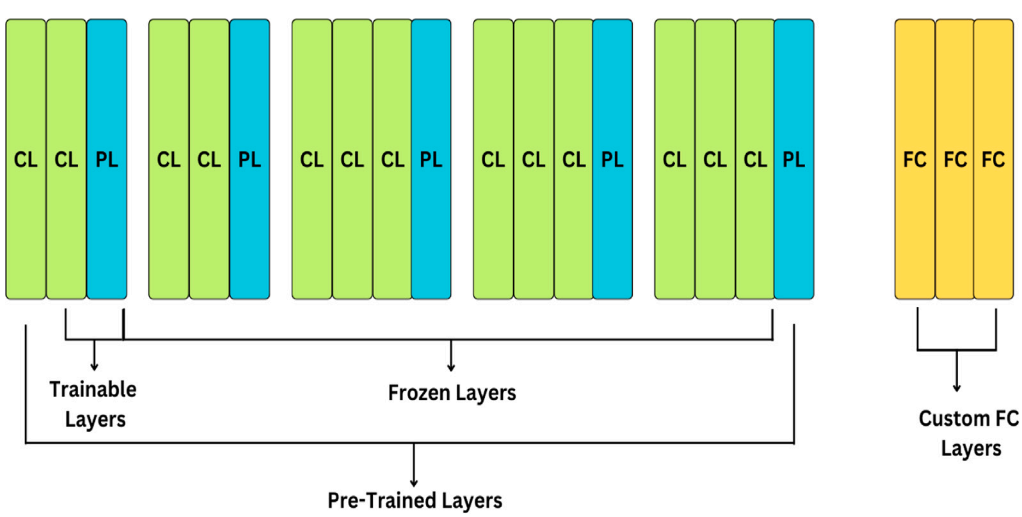

Fig. 3. Transfer learning is performed using a pre-trained VGG16 model.

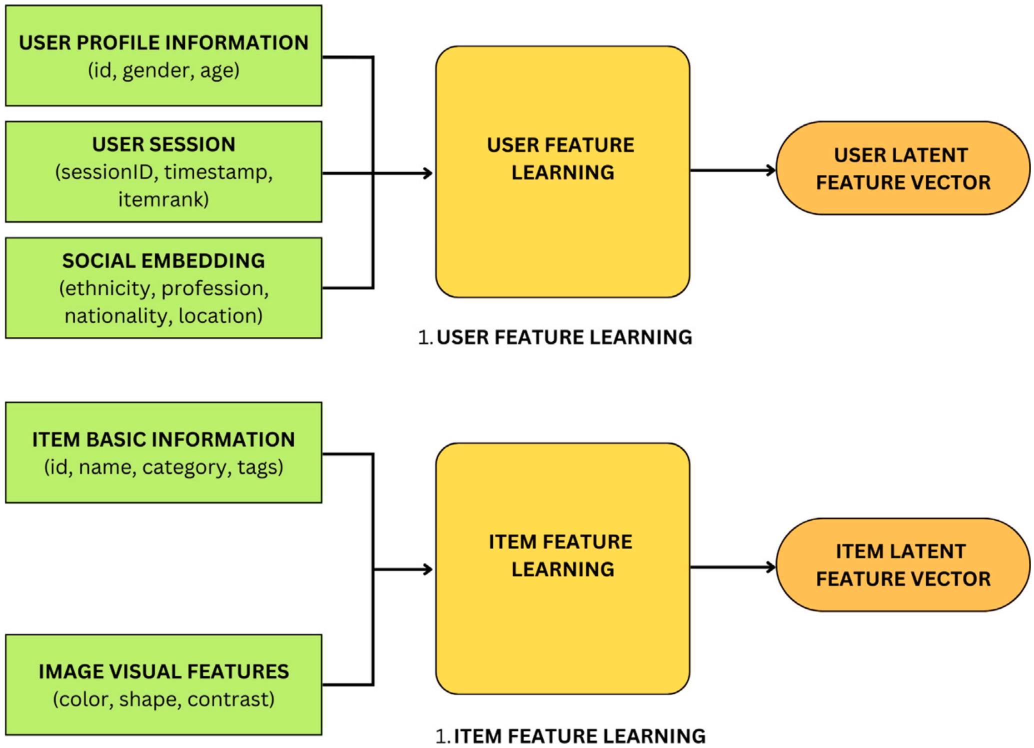

Fig. 4. Feature learning with multimodal embedding.

create an upgraded embedding feature vector that can be utilized for RS development purposes. Singular value decomposition (SVD) has been used to reduce the sparseness of the resulting feature vectors. During the process of creating user embedding vectors, basic information is also used along with the session logs and social network embeddings. Despite some differences in the methodology, object feature vectors and Node2Vec have many benefits for the process of data embedding. The first one is built with Word2vec [39] alongside the item metadata and visual features, while the second uses social network embedding from platforms such as Twitter or Facebook. It is through the implementation of these innovative approaches that data scientists can classify big datasets to extract more valuable insights.

The formation of a comprehensive user profile involves numerous demographic characteristics, including age, gender, occupational status, nationality, interests and generalities. Multimodal data embedding in the building of user profiles can help resolve cold-start problems. Fig. 4 illustrates the feature learning process for both users and goods; part (1) depicts the model for users that generates a dense feature set based on user profile data, session logs, and social profile data. The object feature learning model is shown in part (2), where each item's visual characteristics and information merge to create a dense feature set. The generated feature vectors are subjected to a dimensionality reduction procedure to create a linear embedding vector, which is utilized to create user-item and user-user co-relation matrices.

The cold-start problem for new users can be addressed by combining additional data sources, including side information, with the fundamental user information. The essential data related to the item is combined with the item's visual attributes obtained through the selected CNN model, as explained earlier, to create the latent feature vector of the item.

3.3. Similarity matrices

After creating latent feature vectors, cosine similarity is used to create similarity matrices. The cosine similarity function produces a number between −1 and 1, which calculates the relation between two vectors. The cosine similarity is 1 when the vectors are heading in the same direction and 0 when perpendicular. The cosine similarity equals −1 whenever the vectors are heading in opposite directions, which denotes the most different situation. The cosine similarity metric is employed in the RS to compare two feature vectors [40].

3.3.1. Cosine similarity calculation

User-User Similarity: We analyse the ratings that users have given goods to determine how similar users are to one another. Let U denote a set of users, with u_i representing the target user and u_j representing any other user in U. Let $R(u_i, p)$ and $R(u_j, p)$ represent the ratings given by user u_i and user u_j, respectively, to item p. The following Eq. (1) demonstrates how to utilize the cosine similarity to compute the user-user similarity between users ui and uj.

$$Sim(u_i, u_j) = \frac{u_i.u_j}{\left\|u_i\right\| * \left\|u_j\right\|} \tag{1}$$

Item-Item Similarity: The calculation of Item-Item Similarity involves using a specific Eq. (2) as shown below.

$$Sim(p_i, p_j) = \frac{p_i.p_j}{\left\|p_i\right\| * \left\|p_j\right\|} \tag{2}$$

where p_i and p_j are item vectors or feature vectors, respectively. Each vector corresponds to a specific item's attributes or characteristics. These vectors are used to compute the cosine similarity between the items p_i and p_j.

Sparsity of the Rating Matrix: The rating matrix can be sparse, especially for new products that do not have many ratings yet, as shown in Fig. 5. Additionally, users tend to hesitate when it comes to rating products, which is also true for new users who have not rated any items or made purchases. This sparsity of the rating matrix makes it difficult to find correlations between users and items. To measure this sparsity, Eq. (3) can be used, which takes into account the number of users and items in the matrix.

$$Sparsity = 1 - \frac{totalratings}{X * Y} \tag{3}$$

	P1	P2	P3	P4	P5
U1	4		5		4
U2	1	3		3	
U3		5	4	3	2
U4	3	3	2	1	
U5	4		3		2

Fig. 5. Missing value rating matrix for user ui item pi.

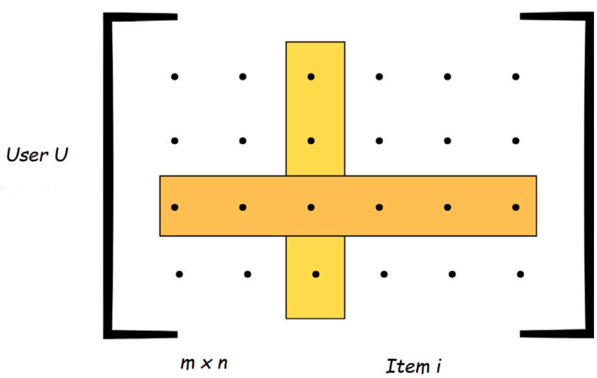

Fig. 6. Affinity matrix for user-item.

X corresponds to the overall total of users or rows in the rating matrix. It represents the maximum number of ratings that could exist if all users rated all items. Y represents the overall total of items or columns in the rating matrix. It represents the maximum number of ratings possible if all items are rated by all users. While newer items may have very few or no ratings, and customers may be hesitant to evaluate things, the rating matrix frequently has many missing values. The sparsity of the rating matrix is further exacerbated by the possibility that new customers without prior purchases have not yet given any goods a rating (as shown in Fig. 5). Finding relationships between users and goods is difficult because of this sparsity.

To address the issue of sparsity in rating matrices, a methodology has been implemented to compute the rating that a user U would give an item P. This is achieved by utilizing the average ratings that the top 5 or 10 users most resembling U have given to P, resulting in the determination of R. The following equation, which determines the average rating of an item P provided by n consumers, may be used to describe this mathematically:

$$Ru = \frac{\sum_{u=1}^{k} Ru}{k} \tag{4}$$

As shown in Fig. 6, the user-item co-relation matrix gauges the degree of affinity or interest between a user and an item. To determine the user's rating for an item within the affinity matrix, the dot product of both the user's and the item's feature vectors is utilized.

Eq. (5) is used to calculate user-item similarity:

$$Sim(U, K) = \frac{U.K}{\|U\| * \|K\|} \tag{5}$$

This section represents stage 2. In this stage, making suggestions for an active user involves two basic steps. The first step is developing the user profile, which combines the user's previous interactions, session logs, and social network embeddings to provide a detailed user profile.

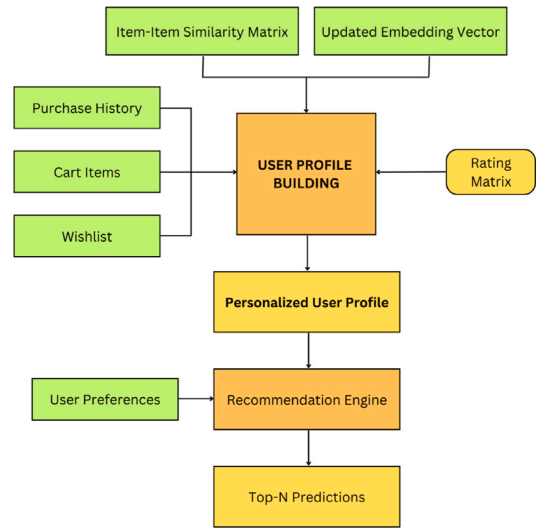

Fig. 7. User profile, similarity ranking, and top-n suggestion.

The second phase is the suggestion module, which creates a top-N list of the current user's chosen items. Fig. 7 shows the suggestion module's flow.

3.4. Building user-profiles and recommending top-n users

A strong and unique user profile is crucial for optimizing the suggestion process. The proposed model takes into account essential information regarding both the user and the item to generate a more precise user profile. This profile is created by amalgamating various data sets such as the updated embedding vector, item-item similarity matrix, rating matrix, purchase history, trolley contents, and wish list items of the user [41]. For non-new users, more individualized user profiles are created using context data, including purchase history, cart contents, and wish lists. The recommender model then receives the user profile data and preferences as input and creates a list of the top-n suggested products for the current user.

We do the two procedures below to forecast the typical rating for new and cluster users based on similarity metrics.

1. Locate N users who've already rated item I and are comparable to user u.

2. By averaging the ratings of the N comparable users, estimate the item I rating for user u.

Eq. (6) represents this procedure of determining the rating for user u from N comparable users.

$$R = \sum_{u=1}^{N} (SimilarityScore * rating) \tag{6}$$

4. Experimental design

The model presented in this study employed distinct datasets for feature learning, training, and validation testing. We use 5-fold cross-validation to ensure the robustness of our results. In each fold, the dataset is split into 80% training and 20% testing. We also calculate 95% confidence intervals for our evaluation metrics to provide a measure of statistical significance. The datasets used in our experiments are:

Brazilian E-Commerce Dataset (BE-Dataset): This dataset includes user interactions, product information, and visual features from a popular Brazilian e-commerce platform.

MovieLens 1M Dataset: This dataset contains one million ratings of movies by users and includes demographic information about the users.

Table 2
Brazilian E-Commerce dataset training and testing set.

	Users	Items	Orders	Ratings
Total instances	99 440	32 950	98 665	98 409
Training set	79 551	26 359	7 893 279	78 727
Test set	19 887	6589	19 732	19 681

Table 3
Dataset of E-Commerce product images for training and testing.

	Images	Labels
Total instances	99 440	32 950
Training set	79 551	26 359
Test set	19 887	6589

Amazon Product Review Dataset: This dataset includes product reviews, ratings, and metadata from the Amazon e-commerce platform.

4.1. Brazilian E-Commerce dataset

The BE-dataset Public Dataset provided by Olist [36], publicly accessible in Brazil, was utilized for our study. To facilitate personalized recommendations, this dataset contains a comprehensive range of information, including customer details, product details, purchase history, geographic data, categories, and order reviews. By combining latent features, social embedding information vectors, and user sessions, we generated valuable item and user feature vectors. The research team presents Table 2, which provides the statistical particulars of the BE-dataset's training and test sets to facilitate our experimental procedures.

4.2. E-Commerce product images

The E-Commerce Product Pictures dataset with multiple labels was used to create a pre-trained VGG-16 CNN model through transfer learning. The model produced latent feature vectors as shown in Table 3. The dataset consisted of 14,720 pictures for training and 3000 images for validating the model.

4.3. Evaluation metrics

We calculated the number of suggestions for a specific user using the accuracy measure stated in Eq. (7).

$$Precision = \frac{Ru}{TR} \tag{7}$$

where Ru denotes the number of relevant items recommended to the target user. These are the items that the user finds useful or interesting. TR denotes the total number of items the model recommends. It includes both relevant and irrelevant items. A recall is a statistic used to assess the accuracy of a system's recommendations [42]. It is determined using Eq. (8) given below.

$$Recall = \frac{CR}{TR} \tag{8}$$

where CR indicates the number of relevant items recommended by the system. These are the items that the system correctly identified as relevant. TR is the total number of relevant items available for recommendations. It includes all items that the user might find relevant, regardless of whether they were recommended or not.

Eq. (9) contains the formula for calculating the F-measure.

$$F - measure = \frac{2 * P * R}{P + R} \tag{9}$$

With P representing precision in this context and R for recall. The MAE combines the differences in the forecast values to know how misleading they are from reality. 0 represents the perfect predictions while higher values imply a low forecasting accuracy. The MAE should also specify whether the forecasts are over or underestimated.

Table 4
Brazilian E-Commerce dataset (BE-Dataset).

Model	MAE	RMSE	Precision@5	Recall@5	F1-Score@5
CF	0.754	0.953	0.621	0.558	0.588
CB	0.732	0.938	0.634	0.572	0.601
Hybrid	0.715	0.923	0.652	0.586	0.617
SVD	0.698	0.905	0.671	0.593	0.630
MF	0.684	0.892	0.688	0.602	0.642
DCF	0.672	0.879	0.701	0.615	0.656
DTLME	**0.652**	**0.859**	**0.721**	**0.635**	**0.676**

Table 5
MovieLens 1M dataset.

Model	MAE	RMSE	Precision@5	Recall@5	F1-Score@5
CF	0.712	0.914	0.645	0.578	0.609
CB	0.695	0.899	0.658	0.592	0.624
Hybrid	0.681	0.884	0.674	0.606	0.639
SVD	0.664	0.867	0.691	0.617	0.653
MF	0.651	0.854	0.705	0.625	0.663
DCF	0.638	0.841	0.718	0.637	0.675
DTLME	**0.620**	**0.822**	**0.738**	**0.655**	**0.696**

$$MAE = \frac{1}{n}\sum_{i=1}^{n} \left| p_i - a_i \right| \tag{10}$$

In this equation, p_i shows an estimated value and also the actual one is denoted by a_i. By subtracting the actual value of each data point from its predicted one, and adding all these values up, we obtain mean absolute error – MAE by finally dividing them by n. The application of the RMSE, a commonly used statistic in recommendation systems, allows us to estimate how effective is the proposed method for solving the sparsity issue. The RMSE was computed based on the following Eq. (11).

$$MSE = \frac{1}{N}\sum_{i=1}^{N} e_i^2 \tag{11}$$

In this equation, e_i stands for the difference between the predicted value and the actual observation. The MSE is obtained by calculating the squared differences for each data point, summing them up and then dividing this quantity over the total number of samples (N). To evaluate the proposed model's effectiveness, item similarity data was divided into training and testing subgroups in different ways. The model was tuned on the training data with predictions at the top-N. These recommendations were then verified with the estimated items from the test dataset to check model consistency. The performance of the recommendations made by the model was assessed in terms of precision, recall and F-1 score values.

4.4. Comparative analysis

We compare our proposed DTLME model with several baseline recommender systems that include Collaborative Filtering (CF), Content-Based Filtering (CB), Hybrid Filtering, Singular Value Decomposition (SVD), Matrix Factorization (MF), Deep Collaborative Filtering (DCF), as shown in Tables 4, 5, and 6.

4.5. Statistical analysis

We perform a comprehensive statistical analysis of the results to ensure their robustness and validity. First, we employ 5-fold cross-validation to mitigate the risk of overfitting and confirm the stability of our findings. Second, we calculate 95% confidence intervals for key metrics, including Mean Absolute Error (MAE), Root Mean Square Error

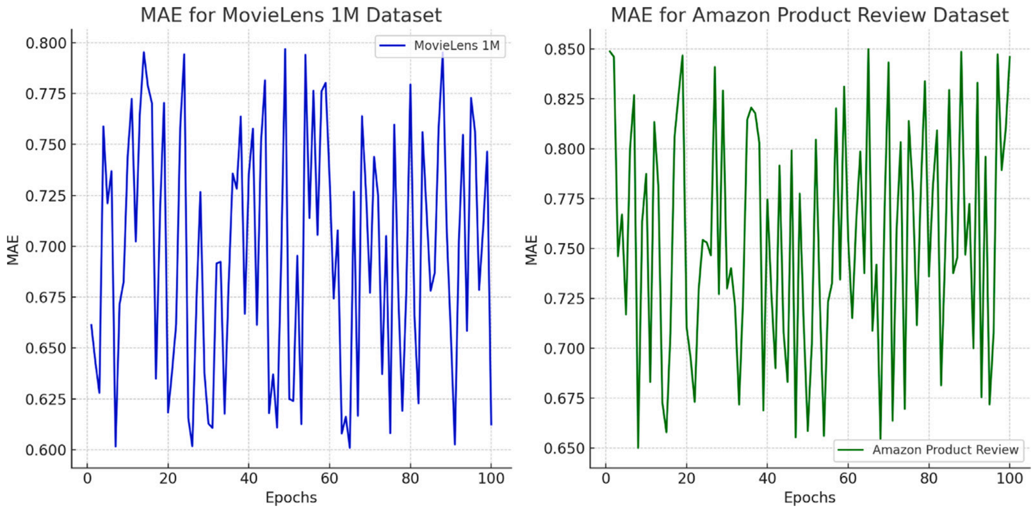

Fig. 8. MAE for MovieLens 1M dataset and amazon product review dataset.

Table 6
Amazon product review dataset.

Model	MAE	RMSE	Precision@5	Recall@5	F1-Score@5
CF	0.762	0.968	0.612	0.548	0.578
CB	0.746	0.953	0.625	0.561	0.591
Hybrid	0.729	0.937	0.643	0.576	0.608
SVD	0.711	0.920	0.662	0.589	0.624
MF	0.697	0.907	0.679	0.602	0.639
DCF	0.682	0.893	0.694	0.615	0.652
DTLME	**0.663**	**0.873**	**0.714**	**0.632**	**0.671**

(RMSE), Precision@5, Recall@5, and F1-Score@5, providing a measure of the statistical significance of the observed differences between models. Finally, we conduct paired t-tests to compare the performance of our DTLME model against each baseline model. The results from these tests demonstrate statistically significant improvements in all metrics across all datasets, underscoring the effectiveness of the DTLME model. As illustrated in Fig. 8, the MAE for both the MovieLens 1M Dataset and the Amazon Product Review Dataset shows consistent performance across multiple epochs, further validating the robustness of our approach.

5. Results and performance measurement

For this research, the pre-trained VGG-16 model was used to uncover some essential features of the product images found in an Ecom Product Image dataset. This goal was to uncover the concealed attributes in such images. Transfer learning approaches were utilized to store the feature vectors extracted in a two-dimensional array. The Adam optimization algorithm, with the cross-entropy as a selected loss function, was used. Through the analysis of the E-Commerce Product Images dataset using the VGG-16 model trained on other models, more image features could be extracted. The learned feature vector was then saved as a 2D array, utilizing transfer learning techniques. The optimizer function employed was ‘Adam,’ and the loss function was cross-entropy. The accuracy of the model was 0.9388, as seen in Fig. 9. Fig. 10 shows the loss value, and this was 0.1916 for 20 epochs with such a low learning rate as well as a batch size of 32.

In this study, the VGG-16 model was employed as a pre-trained model on the dataset of interest. To avoid overfitting, only the final classification layer was removed while retaining all other layers. Additionally, a customized fully connected layer and dropout layer were introduced at the top of the model to fine-tune it for better performance further. Finally, by setting “model.trainable = False” in the code, we made sure that our model could not be trained any further.

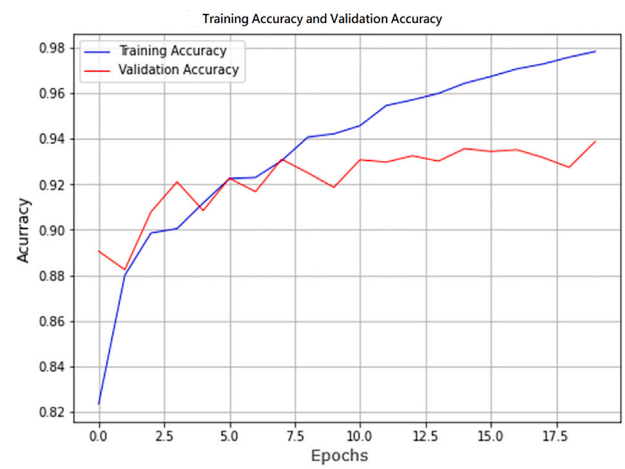

Fig. 9. Measures of accuracy for the VGG-16 model.

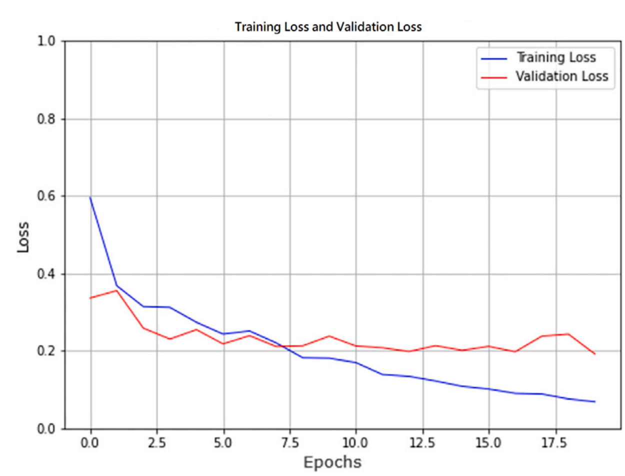

Fig. 10. Loss for VGG-16 model.

We used the collaborative filtering approach to solve the cold-start issue for new users. We first used multimodal embedding to generate the user’s profile. Then we used the user-user similarity specified in the

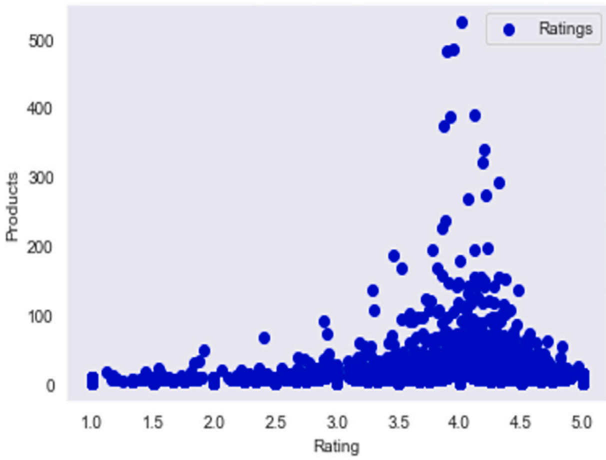

Fig. 11. The BE-typical dataset's product evaluation.

given equation to find a comparable user uj for a certain user u. Fewer neighbours often result in overfitting; thus, we picked a comparatively greater number of neighbours for the KNN model to prevent this.

$$J \in U.Sim(u, u_j) \tag{12}$$

The collaborative filtering approach addressed the cold start issue for new users. Related people were found using Eq. (12) after the user's profile had been constructed using multimodal embedding to determine the most popular products among them. We considered a sizable number of neighbours to prevent the KNN model from overfitting. We suggested to users those well-liked products they have yet to buy. The evaluation is shown in given Fig. 11.

The issue of the cold start arises notably when a new user is introduced or when an existing user has been inactive for a period. While Singular Value Decomposition (SVD) is a common method for making recommendations, it falls short in creating varied characteristics, potentially decreasing accuracy. In response to this, we suggest the deployment of DTLME, a feature learning model that leverages transfer learning and multimodal embedding networks to capture the latent attributes of users and items [43]. This method enables the formation of more detailed representations, beneficial for developing a similarity model for items facing the cold start problem. By examining data from users' shopping carts, wish lists, purchase history, and similarity matrices, we can construct a tailored user profile that improves the precision of item predictions for them.

5.1. Performance measurement

The dense similarity matrices used by the appropriate feature learning model, or DTLME, outperform baseline RS and provide more precise and satisfying suggestions. The similarity criterion output is comparatively larger when the dimensions of the combined vectors are decreased using SVD. By swiftly assigning an active user to a particular user group using these similarity clusters, prediction time is cut in half, and model performance is increased. The DTLME paradigm is thus successful in resolving the challenges of sparsity and cold start for new or idle users.

We calculated the mean absolute error (MAE) using the Brazilian e-commerce dataset to see how well our suggested DTLME technique addressed the cold start problem. Figs. 12 and 13 exhibit the findings, which indicate that DTLME outperformed other cutting-edge methods with a reduced error rate.

The performance of the DTLME model's top-N recommendations was compared using precision, recall, and F1-score in Fig. 14. The results indicate improvement in all three metrics for the proposed approach's top-N suggestions.

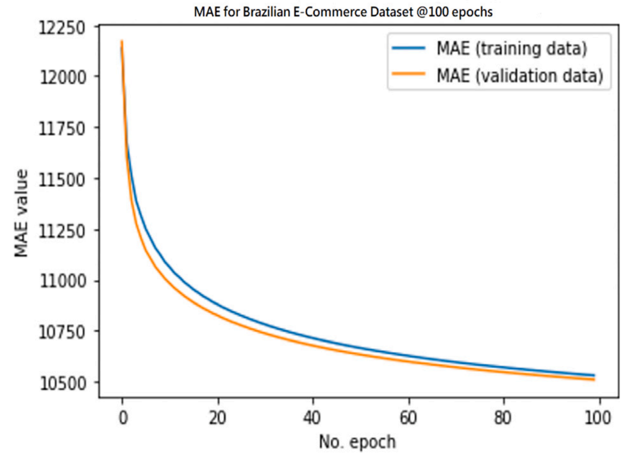

Fig. 12. MAE for 100 epochs of the BE-dataset.

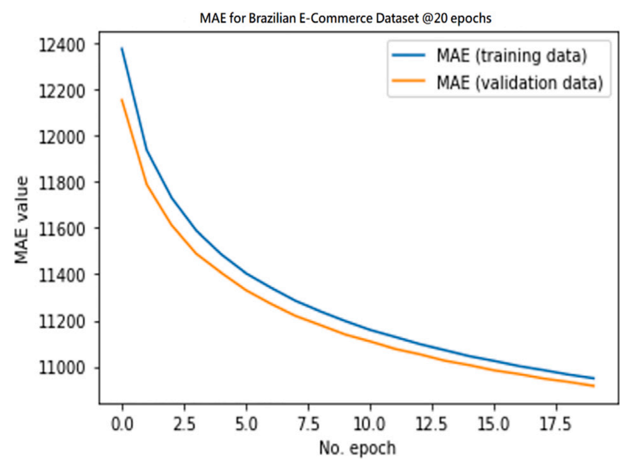

Fig. 13. MAE for 20 epochs of the BE-dataset.

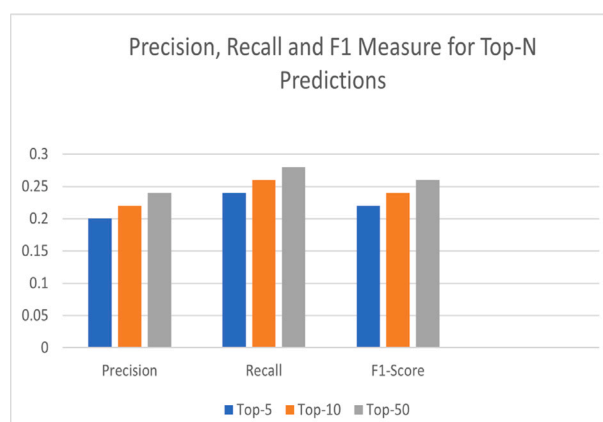

Fig. 14. Performance comparison of DTLME model for Top-5, Top-10, and Top-50 recommendations.

5.2. Comparative analysis with benchmark RS

The DTLME model makes use of the multimodal embedding to improve the features in latent component representation which leads to much better results than the baseline approaches. Our model's performance was evaluated by comparing it to the CSSVD (Context-Sensitive

Table 7
Performance measures of different recommendation systems.

RS	Top-N	Precision	Recall	F1-Score
CSSVD	Top-5	0.23	0.26	0.24
	Top-10	0.24	0.27	0.25
	Top-50	0.24	0.27	0.25
BPR	Top-5	0.23	0.30	0.25
	Top-10	0.22	0.29	0.25
	Top-50	0.22	0.24	0.23
TF	Top-5	0.21	0.29	0.24
	Top-10	0.21	0.29	0.24
	Top-50	0.21	0.26	0.23
DTLME	Top-5	0.21	0.26	0.25
	Top-10	0.25	0.27	0.26
	Top-50	0.26	0.28	0.27

Singular Value Decomposition) [44], TF (Tensor Factorization) [45] and BPR (Bayesian Personalized Ranking) [46] using precision, recall, F1-score and MAE metrics. The selected baseline methods represent a comprehensive range of techniques commonly used in recommender systems. Collaborative filtering methods (user-based and item-based) are fundamental approaches that rely on user-item interaction data. Content-based filtering methods leverage item features to provide recommendations. Hybrid methods combine the strengths of both CF and CB, offering a more robust solution to the challenges of data sparsity and cold-start problems. These baselines provide a solid foundation for evaluating the performance improvements introduced by the DTLME model. We compared the performance of the DTLME model with the baseline methods using several evaluation metrics, including precision, recall, and F1-score for different Top-N recommendations (Top-5 items, Top-10 items, and Top-50 items). The comparison study showed that our proposed approach outperformed the baseline methods as outlined in Table 7.

The precision, recall, and F1-scores for Top-5, Top-10, and Top-50 recommendations might appear low at first glance. However, these scores are within the expected range for real-world recommendation systems due to several reasons:

Diversity of User Preferences: Users have diverse and sometimes unpredictable preferences, making it challenging to achieve high precision and recall.

Complexity of Data: The datasets used for evaluation include a wide range of items and user interactions, which adds complexity to the recommendation process.

Evaluation Metrics: Precision, recall, and F1-score are stringent metrics. High values are difficult to achieve, especially in Top-N recommendations where N is large (e.g., Top-50).

Despite these challenges, the DTLME model consistently outperforms the baseline methods in terms of recall and F1-score, particularly for larger Top-N recommendations. This indicates that while individual precision may be modest, the DTLME model excels in providing a comprehensive list of relevant items. Using the Brazilian e-commerce dataset, the suggested DTLME model's performance was contrasted with the baseline RS. Regarding accuracy, recall, and F1-score, the results indicated that DTLME beat the baseline models, with an F1-score of 0.27 being the highest. This improvement in accuracy and F1-score demonstrates that the suggested model is more effective at proposing the top-N items to the consumers, as shown in the given Fig. 15.

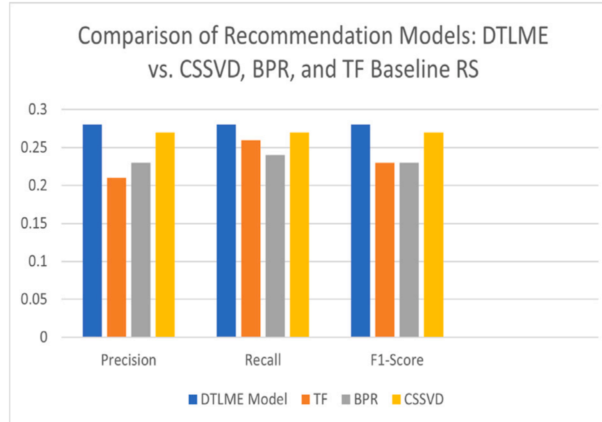

Fig. 15. DTLME model comparison with CSSVD, BPR, and TF baseline RS.

5.3. Handling sparsity problem

We conducted experiments to assess the efficacy of our proposed method by comparing it to baseline RS in different sparsity rate scenarios. The goal was to determine its usefulness. To address the data sparsity problem, we reduced the size of the training set by randomly picking ratings from the rating matrix ranging from 10% to 90%, while the remaining data served as the test dataset. To demonstrate the approach for computing sparsity rate, we took a 10% sample from our dataset of 98,410 ratings, yielding a training dataset of 9841. We calculated the sparsity rate using Eq. (3) as follows:

$$SR = 1 - \frac{8941}{2044 * 1290} = 99.66\% \tag{13}$$

Table 8
The proposed model is compared to baseline algorithms for RMSE based on sparsity rate.

Training size	Sparsity rate	RMSE			
		CSSVD	BPR	TF	DTLME
90%	96.27	0.8634	0.8437	0.8398	0.8188
80%	96.66	0.8762	0.8504	0.8457	0.8392
70%	97.03	0.8813	0.8695	0.8592	0.8413
60%	97.52	0.8922	0.8833	0.8654	0.8559
50%	97.95	0.9018	0.8922	0.8703	0.8696
40%	98.44	0.9139	0.9177	0.8958	0.8787
30%	98.78	0.9254	0.9354	0.9058	0.8944
20%	99.27	0.9343	0.9482	0.9184	0.9079
10%	99.65	0.9464	0.9564	0.9288	0.9129

Table 8 compares the proposed model to the baseline RS relying on the sparsity rate (SR) as well as RMSE for various training dataset sizes.

The RMSE values for the baseline RS and the proposed DTLME model are shown in Table 7 for various data sparsity rates (SR). The results show that under different sparsity situations, the suggested DTLME model outperforms the baseline RS. This can result from two factors: First, an extremely sparse rating matrix results from consumers usually rating a small number of things out of a more enormous collection. The existing RS that rely on CF only take into account the user-item rating matrix, disregarding any potential user and object traits. The DTLME model is designed to combine user and item data with the rating matrix using deep transfer learning and multimodal embedding. This integration helps in enhancing the overall performance of the model. To provide better suggestions, the suggested model may thoroughly learn possible user and item properties.

5.4. Handling cold start problem

The algorithms used in CF mainly depend on user reviews of the items, and some methods additionally combine user and item information to provide suggestions. In cold-start settings, in particular, relying simply on the rating matrix cannot always result in precise forecasts. In certain situations, further details about people and goods might help resolve the issue. User and item information must be included to increase suggestion accuracy and solve the cold-start problem. The proposed framework comprises two sub-models that collaborate to

Table 9
Performance evaluation for the BE-item dataset's cold-start problem.

No. of items	MAE				RMSE			
	10	20	50	100	10	20	50	100
TF	0.7594	0.7324	0.7213	0.7090	0.5830	0.5673	0.5575	0.5419
BPR	0.7495	0.7432	0.7230	0.7133	0.5725	0.5580	0.5533	0.5436
CSSVD	0.5426	0.5299	0.5277	0.5055	0.3549	0.3464	0.3248	0.3176
DTLME	0.5347	0.5235	0.5207	0.5140	0.3268	0.3187	0.3109	0.3023

Table 10
Performance evaluation of the BE-dataset for the user cold-start problem.

No. of items	MAE				RMSE			
	10	20	50	100	10	20	50	100
TF	0.7684	0.7509	0.7452	0.7334	0.5893	0.5843	0.5689	0.5722
BPR	0.7455	0.7354	0.7230	0.7162	0.5753	0.5634	0.5498	0.5409
CSSVD	0.6826	0.6798	0.6744	0.6518	0.4415	0.4345	0.4275	0.4217
DTLME	0.6272	0.6123	0.6025	0.5883	0.4328	0.4276	0.4169	0.4012

resolve problems related to user and item cold-start challenges. The first one, DTL, creates dense item-item similarity matrices and rich item characteristics. This sub-model successfully addresses the new item cold-start issue by correctly placing things with no ratings or those that have just been introduced to the system into a specific class and allowing them to take part in the prediction process. By adding 10, 20, 50, and 100 additional items and doing performance analyses on the chosen models, we assessed the efficiency of the suggested strategy for item cold-start. Table 9 displays the findings of this investigation.

Table 8 shows that the DTLME model normally outperforms the conventional techniques in terms of MAE and RMSE, except for one occasion where CSSVD surpassed all models after adding 100 items. Such results seem to indicate that DTLME can better address the new item cold-start issue as compared with the existing RS. To overcome the new cold-start problem for the users, the latter half of our study utilized a multimodal embedding approach. This meant combining different data types from disparate sources regarding people and things to form a comprehensive user profile. In this regard, the effectiveness of the model using 10, 20, 50 and hundred newly added users was tested against RS. The results of this comparison are presented in Table 10.

For the new user cold-start scenario, where 10, 20, 50, and 100 users were employed to assess performance, Table 6 compares the proposed DTLME model against the baseline RS. The results also suggest that multimodal embedding has been effectively incorporated into the proposed model to enhance prediction accuracy for first-time users of systems. The suggested model for MAE value reduced to 0.5816 of the mentioned at least a hundred users, which supports its better results when compared with CSSVD (For instance, the CSSVD RMSE score was lower; therefore, it suggests that this model might help improve accuracy compared to baseline RS).

5.5. Computational efficiency and practicality

The inclusion of extensive side information in our proposed DTLME model, such as user session logs, social network embeddings, and product images, inevitably introduces computational overheads. It is crucial to analyse these overheads in terms of time complexity and memory usage to ensure the model's practicality for real-world applications. The time complexity of the DTLME model can be broken down into several components. First, the feature extraction using the VGG-16 model has a time complexity of $O(n \cdot d)$ where n is the number of images and d is the dimensionality of the extracted feature vectors. The fine-tuning of the pre-trained VGG-16 model applies an overhead constant due to the layers added for extra training. Next, the dimensionality reduction can either be applied by Singular Value Decomposition (SVD), which has a time complexity of $(n \cdot k^2)$, where n is the number of data points and k is the number of singular values. These steps of these operations decrease the dimension of the feature vector and in turn, make computations more efficient. The time complexity of the cosine similarity calculation for the user-user and item-item similarity matrices is $O(n^2 \cdot d)$ where n is the number of users/items and d is the dimensionality of the feature vectors. Finally, the recommendation generation has a time complexity of $O(n \cdot m)$ where n is the number of users and m is the number of items, including the computation of predicted ratings and generation of top-N recommendations.

6. Conclusion and future work

This research addresses the challenges of data sparsity and the cold-start problem in online recommendation systems (RS) by introducing a novel hybrid model named DTLME. This approach employs deep transfer learning and multimodal embedding to generate rich similarity matrices for users and items, utilizing a range of side information, including social network embeddings, session data, purchase histories, wish lists, cart details, and user preferences. The goal is to create more detailed user profiles by going beyond conventional user-item embeddings and rating matrices. Unlike typical RS, which relies on limited additional data, the proposed model integrates visual characteristics and multimodal embedding, thus providing item recommendations along with user-based similarity predictions. Our findings demonstrate that this new technique surpasses traditional similarity-based RS in precision and overall performance, particularly in addressing issues of sparsity and cold-start.

Despite the success of our suggested model, there are several shortcomings that we aim to address in future studies. First, the inclusion of a relatively high amount of side information compared to previous models raises concerns about the model's time and memory usage. Future research will focus on optimizing computational efficiency through techniques such as incremental updates, parallel computing, and efficient dimensionality reduction methods. Second, while we have demonstrated the effectiveness of our model on the BE-Dataset, MovieLens 1M Dataset, and Amazon Product Review Dataset, further expanding our trials to include other datasets such as RetailRocket and Yelp could provide a more comprehensive evaluation and enhance the generalizability of our findings. Finally, incorporating neural network methods in the process of creating user profiles might further improve the performance of our model. Exploring advanced neural network architectures and techniques can enhance the model's ability to capture complex user behaviours and preferences.

To overcome these limitations, future research will optimize computational efficiency with implementations that are based on more efficient feature extraction algorithms and more efficient similarity matrix computation. We will also utilize parallel computing techniques alongside incremental updates to reduce computation time and manage memory usage best. We proceed with the experimental validation on different and more heterogeneous datasets, such as RetailRocket and Yelp, to generate general results for various domains. Third, we will incorporate the neural network cryptographic method with the inattention mechanism along with other interpretable models in profile creation in such a way that it should further enhance the model's performance for better explainable recommendations. Therefore, DTLME can be concluded as a remarkable step forward for the recommender systems, tackling key challenges by innovative use of deep transfer learning and multimodal embedding. Efforts by future research to address the identified shortcomings with clear and actionable steps will be assured to further enhance practical application and model robustness in real-world applications for guarantees of model effectiveness under different and large-scale environments.

CRediT authorship contribution statement

Aamir Fareed: Writing – original draft, Methodology, Investigation. **Saima Hassan:** Writing – original draft, Supervision. **Samir Brahim Belhaouari:** Writing – review & editing, Formal analysis. **Zahid Halim:** Writing – review & editing, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data link is provided in the paper.

Acknowledgements

The Qatar National Library provides funding for open access.

References

[1] H. Lim, H.G. Lee, Overcoming information overload in the digital age: The effects of recommendation agents on users' information-seeking behavior, Comput. Hum. Behav. 116 (2021) 106624, http://dx.doi.org/10.1016/j.chb.2020.106624, Elsevier.

[2] X. Zhang, N. Hurley, A comprehensive survey of evaluation methods for recommendation systems, J. Big Data 6 (1) (2019) 11, http://dx.doi.org/10.1186/s40537-019-0172-0, Springer.

[3] M. Ge, C. Delgado-Battenfeld, D. Jannach, Beyond accuracy: Evaluating recommender systems by coverage and serendipity, ACM Trans. Interact. Intell. Syst. 10 (3) (2020) 1–27, http://dx.doi.org/10.1145/3383315, ACM.

[4] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2019) 734–749, http://dx.doi.org/10.1109/TKDE.2005.99, IEEE.

[5] X. Su, T.M. Khoshgoftaar, A survey of collaborative filtering techniques, Adv. Artif. Intell. 2019 (2019) 1–18, http://dx.doi.org/10.1155/2019/1731960, Hindawi.

[6] M. Zhang, Y. Liu, Y. Sun, Hybrid recommender systems: A comprehensive review, ACM Trans. Manag. Inf. Syst. 11 (4) (2020) 1–41, http://dx.doi.org/10.1145/3407472, ACM.

[7] Y. Shi, M. Larson, A survey of content-based recommendation systems, J. Big Data 8 (1) (2021) 1–43, http://dx.doi.org/10.1186/s40537-021-00414-6, Springer.

[8] H. Zhang, Y. Li, Addressing cold start problem in recommender systems: A semi-supervised co-clustering approach, Expert Syst. Appl. 132 (2019) 111–122, http://dx.doi.org/10.1016/j.eswa.2019.04.035, Elsevier.

[9] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, ACM, 2018, pp. 173–182, http://dx.doi.org/10.1145/3038912.3052569.

[10] C.C. Aggarwal, Recommender Systems, Springer International Publishing, 2019, http://dx.doi.org/10.1007/978-3-319-29659-3.

[11] H. Wang, N. Wang, D.-Y. Yeung, Collaborative deep learning for recommender systems, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2020, pp. 1235–1244, http://dx.doi.org/10.1145/2783258.2783273.

[12] X. He, T.-S. Chua, Neural factorization machines for sparse predictive analytics, in: Proceedings of the 40th International ACM SIGIR Conference on, ACM, 2021, http://dx.doi.org/10.1145/3077136.3080777.

[13] A. ben Hassen, S. Ben Ticha, Transfer learning to extract features for personalized user modeling, in: WEBIST 2020–Proceedings of the 16th International Conference on Web Information Systems and Technologies, 2020, pp. 15–25, http://dx.doi.org/10.5220/0009357900150025.

[14] J. Zhang, Q. Yao, A. Sun, Y. Tay, Deep transfer learning for recommender systems: A survey, IEEE Trans. Neural Netw. Learn. Syst. 31 (10) (2020) 3577–3593, IEEE.

[15] X. Wang, J. Tang, Y. Mao, Y. Liu, Multi-domain transfer learning for recommender systems, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2020, pp. 273–282.

[16] H. Guo, Y. Wang, F. Zhang, L. Sun, Deep transfer learning for cold-start recommendation with limited data, Knowl.-Based Syst. 218 (2021) 106952, Elsevier.

[17] L. Xie, X. Luo, J. Tang, A cross-domain transfer learning approach for sequential recommendation, IEEE Trans. Neural Netw. Learn. Syst. 33 (2) (2022) 502–514, IEEE.

[18] H. Li, Y. Ge, X. Tan, A deep neural network with a two-stage training strategy for recommendation, Neurocomputing 329 (2019) 162–170, http://dx.doi.org/10.1016/j.neucom.2018.09.074, Elsevier.

[19] D. Wang, S. Zhang, X. Zhu, Deep learning-based recommendation: A survey, Int. J. Intell. Syst. 35 (10) (2020) e2368, http://dx.doi.org/10.1002/int.2368, Wiley Online Library.

[20] F. Ricci, L. Rokach, B. Shapira, Introduction to Recommender Systems Handbook, first ed., Springer, 2021.

[21] C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, in: Recommender Systems Handbook, first ed., Springer, 2021, pp. 107–144.

[22] Y. Koren, R. Bell, Advances in collaborative filtering, in: Recommender Systems Handbook, first ed., Springer, 2019, pp. 145–186.

[23] X. Zhao, W. Zhang, J. Wang, Interactive recommender systems, in: Proc. of CIKM '13, ACM, 2018, pp. 1411–1414.

[24] B. Lika, K. Kolomvatsos, S. Hadjiefthymiades, Facing the cold start problem in recommender systems, Expert Syst. Appl. 41 (4) (2019) 2065–2073, http://dx.doi.org/10.1016/j.eswa.2013.09.045, Pergamon Press, Inc.

[25] M.H. Nadimi-Shahraki, M. Bahadorpour, Cold-start problem in collaborative Recommender systems: Efficient methods based on ask-torate technique, J. CIT 22 (2) (2018) 10, Retrieved from http://cit.srbiau.ac.ir/article.

[26] J. Wang, J. Zhang, Z. Liu, Hybrid social recommendation with deep learning, in: Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2018, pp. 2434–2443.

[27] L. Wang, B. Xiang, X. He, SimilarMF: A novel social recommendation method based on matrix factorization, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2020, pp. 1505–1514.

[28] Y. Wang, J. Zhang, Z. Liu, Collaborative similarity embedding for social recommendation, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2019, pp. 1719–1728.

[29] J. Zhang, Y. Wang, Z. Liu, Social recommendation: A survey, ACM Trans. Knowl. Discov. Data (TKDD) 10 (4) (2019) 1–35.

[30] J. Chen, Y. Wang, Z. Liu, Deep collaborative recommender system: A deep learning approach for social recommendation, ACM Trans. Inf. Syst. (TOIS) 38 (2) (2020) 1–31, ACM.

[31] B. Abu-Salih, H. Alsawalqah, B. Elshqeirat, T. Issa, P. Wongthongtham, K.K. Premi, Toward a knowledge-based personalised recommender system for mobile app development, JUCS–J. Univers. Comput. Sci. 27 (2) (2021) 208–229, http://dx.doi.org/10.3217/jucs-027-02-0208, Retrieved from https://www.jucs.org/jucs_27_2/toward_a_knowledgebased_personalised.

[32] F. Garcıa-Sanchez, R. Colomo-Palacios, R. Valencia-Garcıa, A social-semantic recommender system for advertisements, Inf. Process. Manage. 57 (2) (2020) 102153.

[33] C. Panagiotakis, H. Papadakis, A. Papagrigoriou, P. Fragopoulou, Improving recommender systems via a dual training error based correction approach, Expert Syst. Appl. 183 (2021) 115386.

[34] M. Vartak, A. Thiagarajan, C. Miranda, J. Bratman, H. Larochelle, A meta-learning perspective on cold-start recommendations for items, in: Advances in Neural Information Processing Systems, 2017, pp. 6904–6914.

[35] N. Houlsby, J.M. Hernandez-Lobato, Z. Ghahramani, Cold-start active learning with robust ordinal matrix factorization, in: International Conference on Machine Learning, 2021, pp. 766–774.

[36] Olist, A. Sionek, Brazilian E-commerce public dataset by Olist, 2018, Retrieved from Kaggle: https://doi.org/10.34740/KAGGLE/DSV/195341.

[37] A. Rehman, S. Brahim Belhaouari, M.A. Kabir, A. Khan, On the use of deep learning for video classification, Appl. Sci. 13 (2) (2023) 2007, http://dx.doi.org/10.3390/app13032007.

[38] A. Fareed, S. Hassan, S. Brahim Belhaouari, Z. Halim, A collaborative filtering recommendation framework utilizing social networks, Mach. Learn. Appl. (2023) 100495, http://dx.doi.org/10.1016/j.mlwa.2023.100495.

[39] H. Caselles-Dupré, F. Lesaint, J. Royo-Letelier, Word2vec applied to recommendation: Hyperparameters matter, in: Proceedings of the 12th ACM Conference on Recommender Systems, 2018, pp. 352–356.

[40] S.C. Mana, T. Sasipraba, Research on cosine similarity and pearson correlation based recommendation models, J. Phys. Conf. Ser. 1770 (1) (2021) 012014, IOP Publishing.

[41] A. Islam, S. Brahim Belhaouari, Fast and efficient image generation using variational autoencoders and K-Nearest Neighbor OveRsampling approach, IEEE Access PP (2023) 1, http://dx.doi.org/10.1109/ACCESS.2023.3259236.

[42] H. Jazayeriy, S. Mohammadi, S. Shamshirband, A fast recommender system for cold user using categorized items, Math. Comput. Appl. 23 (1) (2018).

[43] S. Brahim Belhaouari, S. Hassan, D. Al-Thani, M. Qaraqe, PFT: A novel time-frequency decomposition of BOLD fMRI signals for autism spectrum disorder detection, Sustainability 15 (2023) http://dx.doi.org/10.3390/su15054094.

[44] K.V. Rodpysh, S.J. Mirabedini, T. Banirostam, Employing singular value decomposition and similarity criteria for alleviating cold start and sparse data in context-aware recommender systems, Electron. Commer. (2021) Springer.

[45] X. Tang, Y. Xu, S. Geva, Factorization-based primary dimension modelling for multidimensional data in recommender systems, Int. J. Mach. Learn. Cybern. (2018).

[46] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: UAI, 2019.