International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

A Hybrid Recommender System Exploits Layered Convolutional Neural Networks.

[1]Naveen Kumar Navuri, [2]Dr. CVPR Prasad

Submitted:14/03/2024 **Revised**: 29/04/2024 **Accepted**: 06/05/2024

Abstract: Enhancing user engagement and satisfaction in e-commerce platforms by incorporating customer preferences and interests into product recommendations is of paramount importance. However, accurately capturing these preferences, both explicit and implicit, from the vast array of products available in catalogues poses a significant challenge. In this study, we propose a novel approach that leverages deep learning techniques to extract latent preferences from product images. Our method focuses on extracting relevant data from the features of interest in product images, thereby enabling the identification of underlying customer preferences. We demonstrate the efficacy of our strategy by integrating this data extraction process into a product-based recommendation algorithm. Through experimental validation, we showcase the effectiveness of our approach in generating personalized suggestions tailored to individual customer preferences. Our findings underscore the potential of deep learning-based methodologies in harnessing visual cues to enhance the personalization of e-commerce recommendations, thereby improving user experience and engagement.

Keywords: *E-commerce platforms, Customer preferences, Product recommendations, Deep learning techniques Latent preferences, Product images.*

1. Introduction

In the field of electronic commerce, the pursuit of enhancing user experience and engagement by means of personalized product recommendations is an enduring endeavor. An approach that is increasingly being explored for enhancing recommendation systems is the utilization of visual data present in product images [1]. By incorporating visual recognition methods, this data can be harnessed to gain a deeper understanding of customer preferences, ultimately resulting in an improvement in the quality of recommendations provided.Traditional recommendation systems place considerable importance on user-item interactions and explicit feedback to generate suggestions [2]. However, extracting latent preferences from such data can be a challenging task, especially when explicit feedback is scarce. Furthermore, incorporating implicit preferences adds an additional layer of complexity to the recommendation process. To address the challenges faced in analyzing category patterns in product images, this study presents a novel strategy. This strategy involves the use of deep learning techniques, specifically Convolutional Neural Network (CNN) architectures, to efficiently extract meaningful features from product images [3].

By utilizing pre-trained CNN models, our approach reduces the need for extensive hyper-parameter tuning and training time, distinguishing it from traditional methods.To achieve seamless integration with other data sources and facilitate comprehensive customer profiling, a fully connected layer can be added on top of the CNN network [4]. This approach enables the effective refinement and combination of preference data extracted from images. This study exemplifies the successful implementation of such a method, demonstrating its potential for enhancing the recommendation process.

The development of recommendation systems has resulted in various filtering paradigms; however, the concentration on depositories with diverse categories of homogenous material has been relatively restricted [5]. In contrast, our proposed solution employs a product-based filtering method, which utilizes supplementary information about items to capture client preferences in a prompt and efficient manner. This method not only provides real-time scalability, but also allows for the swift adaptation to changing customer preferences.

The primary focus of this research is the creation of a novel recommendation system that utilizes product images to learn customer preferences. Our proposed solution combines the output of convolutional neural network models and interaction data, resulting in an item-based recommendation system that provides personalized suggestions tailored to individual tastes [6]. In the subsequent sections, we will present

[1]*Research Scholar, ANU, Guntur*
[2]*Dean Academics, Malla Reddy Engineering College for women, Hyderabad and Research Supervisor, ANU, Guntur*
Emails: naveennavuri@gmail.com [2]prasadcvpr@gmail.com

empirical evaluations using data obtained from The Movie Database [7], showcasing the effectiveness and practicality of our proposed method.

2. Preliminary

This research presents a method for generating recommendations using meta-data from product images, which are widely available in most domains. By utilizing visual recognition techniques, we can identify category patterns in the images and enhance the quality of recommendations produced by the recommender model [8][9]. The research suggests the utilization of pre-trained convolutional neural network (CNN) models to extract image features, thereby decreasing the time and computational resources necessary for model training [10]. By adding a fully connected layer on top of the CNN network, the extracted features can be easily integrated with other sources, allowing for more effective utilization of preference information [11].

For generating recommendations, the study uses item-based filtering, which captures customer preferences by exploiting auxiliary information about items [12]. Unlike more traditional methods, item-based filtering is fast and scalable in real-time. The result is a new recommendation system that learns from the images of the products that customers have purchased and recommends the Top-n items that they might like. The suggested item-based recommendation system integrates the outcomes of the CNN model, which supplies details about implicit customer preferences, and incorporates interaction data, specifically the rating matrix.

Convolutional Neural Networks (CNNs) are used to extract features from the product images [13]. CNNs are artificial neural networks that specialize in object and action recognition. They are more efficient and effective than fully connected networks because they use convolutional operations instead of matrix operations [14]. The architecture of CNNs is modeled after the human visual system, processing visual information in a hierarchical manner [15]. The convolution layer is the first layer in a CNN and takes an image as input. It applies a set of filters to produce feature maps that capture local patterns and edges. The pooling layer follows the convolution layer and down samples the feature maps to reduce their size and introduce invariance to image translation, rotation, and scaling. The fully connected layer flattens the feature maps into a vector and feeds them into a standard neural network. It produces a vector of probabilities, with each element corresponding to a class or label that the image belongs to [16].

Competition /Challenge	CNN Architecture	Year Introduced	Data set	Key Features/Contributions
ImageNet Large Scale Visual Recognition Challenge (ILSVRC)	AlexNet	2012	ImageNet	Pioneered the use of deep convolutional neural networks (CNNs) for image classification tasks. Introduced innovations such as ReLU activation, dropout regularization, and data augmentation.
ImageNet Large Scale Visual Recognition Challenge (ILSVRC)	VGG (Visual Geometry Group)	2014	ImageNet	Utilized a deep architecture with small 3x3 convolutional filters, achieving competitive performance on image classification tasks. Known for its simplicity and effectiveness.
ImageNet Large Scale Visual Recognition Challenge (ILSVRC)	ResNet (Residual Network)	2015	ImageNet	Introduced residual learning, enabling the training of very deep neural networks with hundreds of layers. ResNet architectures, such as

				ResNet-50 and ResNet-101, have become standard benchmarks for various visual recognition tasks.
ImageNet Large Scale Visual Recognition Challenge (ILSVRC)	Inception (GoogLeNet)	2014	ImageNet	Proposed the Inception architecture, which introduced inception modules with parallel convolutional layers of different filter sizes. Achieved state-of-the-art performance on image classification tasks with improved computational efficiency.
MNIST	LeNet-5	1998	MNIST	Pioneered the use of convolutional neural networks (CNNs) for handwritten digit recognition. Consisted of convolutional and average pooling layers followed by fully connected layers.
CIFAR	ResNet	2015	CIFAR	Adapted ResNet architectures for image classification on the CIFAR dataset, achieving superior performance with deep residual networks.
COCO	Mask R-CNN	2017	COCO	Extended Faster R-CNN by adding a segmentation branch, enabling instance segmentation on the COCO dataset. Achieved state-of-the-art results in object detection and segmentation tasks.

3. Related Work

Collaborative filtering based on memory is a common recommendation technique that identifies neighbors for each user or item based on similarity measures and predicts ratings based on the ratings of these neighbors [17]. Model-based collaborative filtering has been successfully employed with various techniques such as Bayesian, rule-based, and decision tree methods [18]. Another commonly used method for collaborative filtering is latent factor models, which aim to uncover hidden factors that explain the ratings [19]. These models can be either deterministic or probabilistic and can incorporate additional information to enhance the predictions.

The most of the studies have demonstrated the potential of deep learning in recommendation systems such as multi-layer perceptron networks, deep candidate generation methods, factorization machines, and restricted Boltzmann machines [20]. These methods have been applied to various recommendation tasks, including collaborative filtering, music, and android background services. Recurrent neural networks can

also be used in recommender systems, as demonstrated by a method that combines an autoencoder and a recurrent neural network to predict user purchase behavior [21].

This research aims to integrate CNNs and transfer learning with traditional recommender systems for document, resource, and e-learning recommendation tasks, with the goal of creating a unified framework for e-commerce that combines the strengths of both conventional and CNN-based methods [22].

Recommendation systems can be constructed through the use of collaborative filtering, content-based filtering or a combination of both. [23]. Collaborative filtering based on memory is a well-established approach that relies on similarity measures to locate neighbors for every user or item and anticipates ratings based on their ratings [24]. However, this method faces issues related to data scarcity and necessitates dimensionality reduction. Collaborative filtering can also be approached as a categorization issue, with any classification or regression algorithm capable of addressing it. Model-based collaborative filtering has proven successful across various procedures, including Bayesian, rule-based, and decision tree methods [25].

Deep learning techniques have facilitated the emergence of advanced recommendation systems. Recent investigations have highlighted the utility of deep learning methods, such as Jingyuan Chen et al.'s employment of a multi-layer perceptron network to augment the collaborative filtering model [26] and Paul Covington, Adams, and Emre Sargin's proposal of a deep candidate generation approach for YouTube recommendations [27]. The development of factorization techniques combined with feature engineering using deep learning has led to the emergence of the Deep Factorization Machine [28]. Furthermore, Restricted Boltzmann Machines (RBM) have been applied to music and android background service recommendations [29]. Recurrent neural networks are also utilized in recommender systems, such as the fusion of an encoder for article depiction and a recurrent neural network for forecasting user purchase behavior [30]. Moreover, convolutional neural networks (CNNs) have been extensively adopted in document, resource, and e-learning recommenders [31].

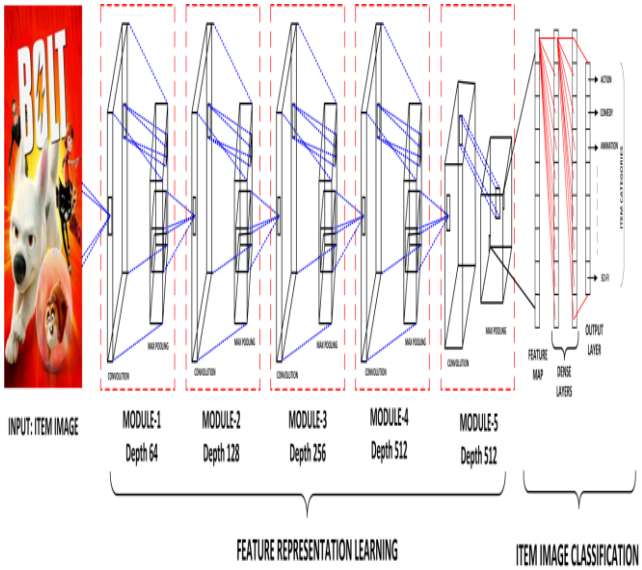
This study seeks to examine the viability of combining deep learning strategies with conventional recommender systems in e-commerce. Specifically, it explores employing convolutional neural networks (CNNs) to bolster the recommendation procedure. One major challenge in constructing a recommender system entail capitalizing on a consumer's prior purchase history to estimate their preferences and requirements. Utilizing a customer's transaction records as input to a recommender model permits the capture of temporal dynamics and contextual aspects influencing their decision-making process. As a result, customized and pertinent recommendations ensue, contributing to enhanced customer gratification and retention [32].

There exist multiple ways to merge a customer's purchasing record into a recommender model, contingent upon the kind and amount of accessible data, the model's intricacy and scalability, and the intended recommendation outcome.

- **Using purchase frequency or recency as features:** This is a simple and intuitive way to use the purchase history as input, by counting how often or how recently a customer has bought a certain product or service. These features can be used to measure the customer's interest or engagement with a product or service, and to rank them accordingly. For example, a recommender model can suggest products or services that have been frequently or recently purchased by similar customers or by the same customer in the past.
- **Using purchase sequences as features:** This is a more sophisticated way to use the purchase history as input, by encoding the order and the timing of the purchases as features. These features can be used to capture the sequential patterns and the temporal dependencies that exist in the purchase history, and to generate recommendations that are consistent with the customer's behavior and goals. For example, a recommender model can suggest products or services that are likely to be bought next by similar customers, or by the same customer in a similar situation.
- **Using purchase embeddings as features:** This is an advanced way to use the purchase history as input, by learning a low-dimensional representation of each purchase as a vector of numbers, called an embedding. These embeddings can be used to encode the semantic and contextual information that is associated with each purchase, such as the product or service category, the price, the rating, the review, etc. By using these embeddings as features, we can measure the similarity and the diversity of different purchases, and to generate recommendations that are tailored to the customer's preferences and needs. For example, a recommender model can suggest products or services that are similar or complementary to those that have been previously purchased by similar customers, or by the same customer.

4. Methodology

The proposed e-commerce framework integrates traditional recommendation methods with convolutional neural networks (CNNs) to generate top-n recommendations based on both item content and collaborative filtering [33]. The framework classifies items into categories using a fully connected neural network and a pre-trained CNN, which are then used to build a hybrid model that generates personalized suggestions [34]. The framework consists of two stages, with the first stage utilizing a deep CNN architecture to extract feature vectors from item images and classify items. The second stage uses the feature vectors to build the hybrid model and generate the top-n personalized recommendations.



The figure above shows an example of the CNN architecture used in our framework

4.1 Product classification method

Assisting online customers in selecting products from a wide range of offerings can be a challenging task for e-commerce companies. With a vast array of products and associated metadata (such as descriptions, features, photos, and customer reviews) to keep up with, many companies struggle to remain current. When presenting products to customers, the category of the item is a critical factor to consider [35]. Products can potentially have numerous labels or classifications. Therefore, this study presents a Convolutional Neural Network (CNN)-based model for classifying product images [36]. Each category's extracted feature map is treated as unique data for generating recommendations.

In this study, the proposed architecture is a stacked network incorporating a pre-trained convolutional network and a fully-connected layer for image classification. This architecture benefits from transfer learning, which enables the utilization of the trained model's weights and decreases the time required for model training [37, 38].

When neural networks are trained on data, the acquired knowledge is stored in the network's weights. However, training large amounts of data on a heavily layered network from scratch can be a time-consuming process that may take several days. To address this issue, one neural network can extract weights and transfer them to another, a process known as transfer learning [39]. This approach was adopted in the current study.

Given our goal of assigning categories to images of products, we propose using pre-existing models for this task. Our system takes into account the VGG19 (Visual Geometry Group) CNN model, which has been pre-trained on millions of annotated images from ImageNet and has 19 layers [40]. The idea behind VGG19 is to add more network layers to better reflect the image's characteristics and semantics [41].

A convolutional neural network (CNN) is a type of artificial neural network that can process images and recognize objects. A CNN consists of several layers that perform different operations on the input image, such as convolution, pooling, and classification. The following is an overview of the network architecture of a CNN:

- The **INPUT** layer receives an image as a matrix of pixels, with each pixel having a value for its color channel (red, green, or blue). The size of the input layer is determined by the width, height, and number of channels of the image. For example, an input layer for a color image of size 224 x 224 pixels would have a dimension of [224 x 224 x 3].
- The **MODULE1** of layers consists of two convolutional layers and a pooling layer. The convolutional layers apply filters to the input image to extract low-level features, such as edges and corners. The filters are learned during the training process and have a size and number that are specified by the network designer. The activation function for the convolutional layers is the rectified linear unit (ReLU), which outputs the positive part of its input. The pooling layer reduces the size of the feature maps by applying a max operation over a sliding window, which preserves the most important information. The number of parameters to be learned in this block is (1792 + 36928).
- The **MODULE2** of layers consists of two convolutional layers and a pooling layer, similar to the first block. However, the filters in this block are larger and more numerous, which allows the network to extract higher-level features, such as shapes and patterns. The number of parameters to

be learned in this block is (73856 + 147584).

- The **MODULE3** of layers consists of four convolutional layers and a pooling layer, similar to the previous blocks. However, the filters in this block are even larger and more numerous, which allows the network to extract complex features, such as parts of objects and scenes. The number of parameters to be learned in this block is (295168 + 590080 + 590080 + 590080).
- The **MODULE4** consists of four convolutional layersand a pooling layer, which use RELU as the activation function to perform feature extraction. The pooling layer reduces the dimensionality of the feature maps by applying the max-pooling operation. The number of parameters to be learned in this block is (1180160 + 2359808 + 2359808 + 2359808).
- The **MODULE5** comprises four convolutional layersand a pooling layer, which use RELU as the activation function to perform high-level feature extraction. The pooling layer further reduces the dimensionality of the feature maps by applying the max-pooling operation. The number of parameters to be learned in this block is (2359808 + 2359808 + 2359808 + 2359808).
- The final layer is a fully connected layer that performs the classification task. It takes the output of the last pooling layer and maps it to a vector of scores for each possible class. The activation function for this layer is the softmax function, which normalizes the scores to probabilities that sum up to one. The number of parameters to be learned in this layer depends on the number of classes in the dataset.
- KMODEL is comprised of fully connected layers. Specifically, it features a dense layer with 1024 units that utilizes a non-linear activation function, followed by a dropout layer and a layer with a sigmoid function. The final layer generates the class scores that classify the item category. These class scores represent the probability of belonging to a particular category. Additionally, the activation function RELU is employed to capture the non-linearity present in the data.

The image of an item is fed into a Convolutional Neural Network (CNN) architecture that performs a series of transformations to extract features from low-level to high-level [42]. The CNN architecture has many parameters that need to be learned at each layer during the training process to classify the item and assign it a score correctly. The parameters are adjusted and optimized by comparing the predicted category of the image with the actual category in the training set. This is done using gradient descent, an optimization technique [43]. The learning rate and momentum of the optimizer are important factors for finding the best values for the parameters.

In this study, the proposed CNN architecture uses stochastic gradient descent as the optimizer to minimize the error of the loss function and find the optimal weights [44]. After obtaining the feature maps of each item category in the repository, they are used in the next step. The feature maps help the recommendation algorithm to infer the preferences of a user based on their previous purchases. This paper then continues to discuss how to build a model for making recommendations.

4.2 Recommendation method

We propose a novel approach to generate Top-N recommendations based on item features [45]. Our method leverages the visual information of the items that the user has bought before to infer their preferences and interests. This is particularly useful in scenarios where the item catalog is large and diverse, and where the user may have heterogeneous tastes across different categories. Unlike traditional product-based Top-N recommendation algorithms, which rely on the similarity of the items in the user's purchase history [46], our approach can capture the fine-grained details and nuances of the items that appeal to the user.

The goal of Top-N recommendation models is to produce a ranked list of items for each user, according to their predicted relevance [47]. Our model uses images of the items that the user has bought before as inputs to learn their latent preferences and generate personalized recommendations.

Product-based recommendation models are still prevalent among online retailers, despite the growing popularity of user-based models [48]. The reason is that user-based models require a lot of computation to find and compare products that have been rated similarly by millions of customers in real-time [49]. Product-based models, on the other hand, have high throughput and can handle real-time data processing. However, product-based models may also produce recommendations that are too simple and predictable, without any element of surprise [50]. To address this issue, we propose a filtering algorithm that uses the image and category of items to generate a list of products for a specific customer. This way, we can provide more adventurous recommendations that do not frustrate the customer with obvious items. Our algorithm is similar to the baseline models for top-n recommendations [51], but it also uses item image similarity to infer the latent preferences of customers based on their previous purchases. By comparing the extracted feature maps of the items that

the customer has requested and bought with the feature map of the item's category in the repository, we can identify the items that are most likely to appeal to the customer.

In the proposed scheme, 'L' signifies the rating matrix including customers' evaluations of items. 'C' denotes the collection of customers, while 'S' signifies the set of items in the inventory. Likewise, 'X' is a subset of items in 'S' and 'Xc' is the set of items rated by the target customer 'C'. Additionally, 'Xt' is the target item for which similar items must be found, and 'xy' is an individual item with an id of 'y'. Finally, the '#' symbol in the algorithm represents a comment line that explains the next step, while 'NNx' is the set of nearest neighbors for item 'x'.

4.3 Algorithm: Top-n recommendation using a product-based approach

The primary objective of Top Recommendations (c, L, n) is to identify the top n recommended items (X) for a specific target customer (c) based on an L-Rating matrix containing all customers' preferences (ratings) for all items. This approach allows for the recommendation of n items to the customer, with "n" indicating the desired number of recommendations.

Furthermore, the algorithm evaluates items using the specified threshold rating and employs the following data type to store the items information. It incorporates add () module and sort() module to add and arrange elements in a list.

The primary processes of the algorithm are as follows:

Calculate the nearest neighbors (items) of a target item, and generate the top-n recommendations using the neighbors' data.

Explaining the algorithmic in detail:

Algorithm: Nearest Neighbors-based Item Recommendation
Input:
Target item feature vector x
Repository of items |Item|
Distance threshold distance_th
Output:
List of nearest neighbor items neighbors

1. **Begin**
2. Initialize an empty list for nearest neighbors: neighbors = []
3. Iterate over each item y in the repository:
   i. Calculate the distance between target item x and current item y: dist = distance(x, y)
   ii. If the distance is less than the distance threshold:
   iii. Add the current item y to the list of nearest neighbors: neighbors.append(y)
4. Return the list of nearest neighbor items: neighbors
5. **End**

Algorithm: Top-N Item Recommendations
Input:

- Customer ID c
- Ratings of items by customers L
- Number of recommendations n
- Threshold rating threshold_rating
- Distance threshold distance_threshold

Output:
Top-N recommended items rec_list

1. **Begin**
2. Extract the list of items rated by customer c: rated_items = L[c]
3. Initialize an empty list for storing items and their scores: item_list = []
4. Iterate over each item rated by the target customer:
   a. If the rating of the item is greater than the threshold rating:
   i. Find nearest neighbors for the current item: nearest_neighbors = nearest_neighbors(xk)
   ii. For each neighbor item in nearest_neighbors:
   iii. Calculate the similarity score between the rated item xk and the neighbor item xy: score = similarity(xk, xy) * lk
   iv. Add the neighbor item and its score to the item list: item_list.append((xy, score))
5. Sort the item list based on scores in descending order: item_list.sort(reverse=True)
6. Extract the top-n recommendations from the sorted item list: rec_list = item_list[:n]
7. Return the list of top-N recommended items: rec_list
8. **Ends**

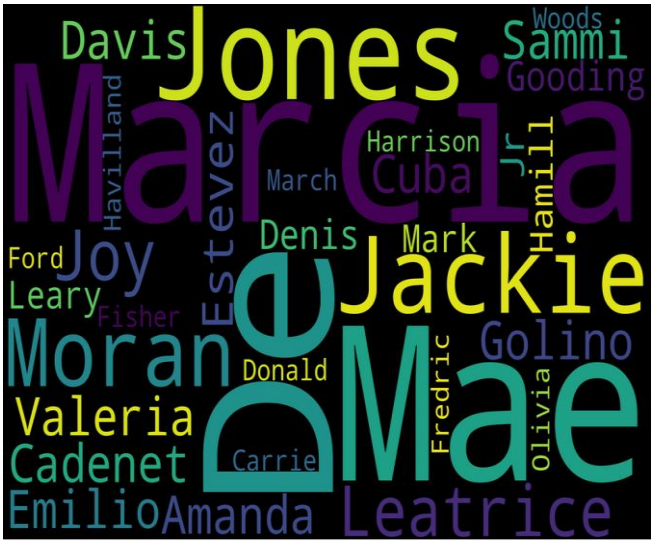
The current method uses a product-based filtering approach based on the nearest neighbor model, which establishes the neighborhood based on item features [52]. This method reduces the computational burden compared to a user-based approach using a memory-based algorithm [53]. Before computing the neighbors, a feature map is constructed for each item by extracting the last layers of the CNN, as described earlier [54]. The similarity measures obtained can be stored and reused in subsequent calculations, making Euclidean distance, L2 regularization, and cosine distance suitable alternatives [55]. Additionally, this method's implementation

complexity is lower than that of conventional and hybrid recommender methods [56]. These methods have a reasonable chance of producing original, serendipitous suggestions [57], and the cold start problem can be resolved using content analysis in item-based suggestions [58].

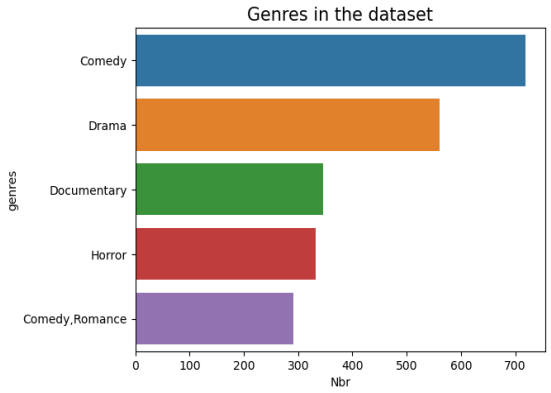
5. Experiments and Results

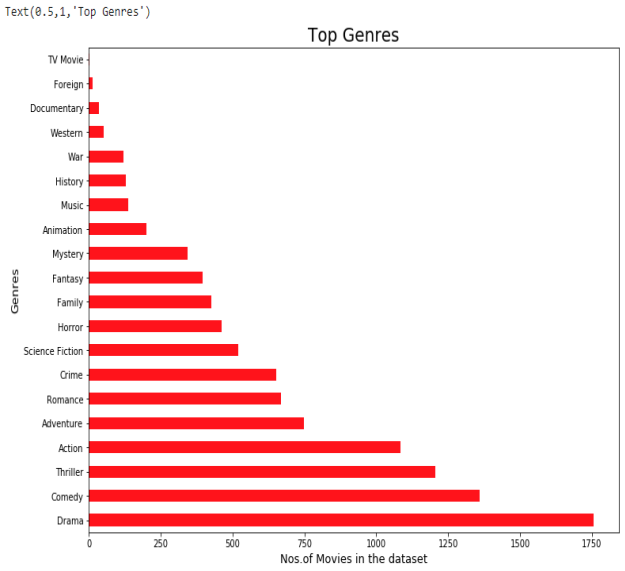
The objective of this study is to evaluate the performance of a recommendation method that generates valuable suggestions. Our main focus is on assessing the accuracy of the feature maps generated by the classification model, as opposed to the recommendation model. This is because our hybrid approach, which uses a pre-trained convolutional neural network (VGG19) to classify items based on their images, represents a novel application of the model.

The TMDb dataset was collected by TMDb, an online community, since 2008. The dataset provides information on over 360,000 films in 39 languages and has been used by researchers and practitioners globally. The dataset covers a five-year time span from 2012-2017 and contains over 5,000 documents of film metadata. The dataset has been preprocessed to remove duplicates, missing information, and unlabeled records. The dataset includes a multi-label classification model suitable for images of items, which are movie posters. The dataset includes a set of 20 possible classifications for films, referred to as "movie genre." The below is the wordcloud for the cast column in the dataset, the following figure depicts the variety and frequency with which film genres can be found in the dataset.

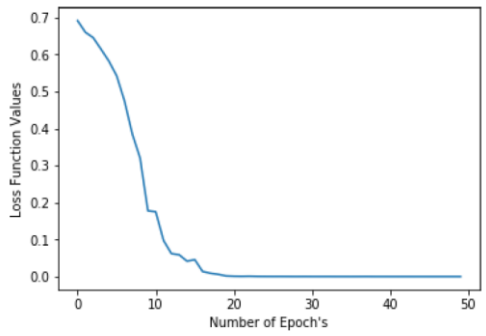


	genres	Nbr
0	Comedy	719
1	Drama	561
2	Documentary	346
3	Horror	333
4	Comedy,Romance	291

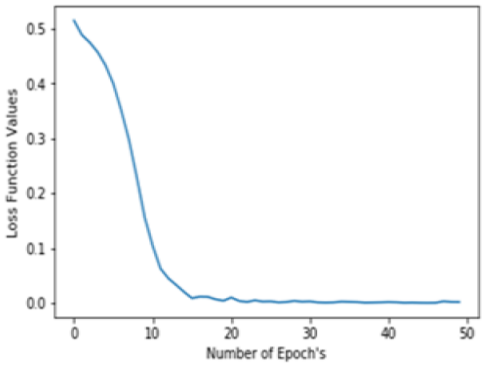




The dataset at hand comprises over a thousand records of **drama, comedy, and thriller genres**, as evidenced by the preliminary examination (refer to Figure). The Table illustrates the model's categorization of items that meet these criteria. The images in the dataset are posters with dimensions of 256 pixels on the longest side. The proposed input module for the CNN scans the image from left to right, two pixels at a time, with a 5x5 window. The tensor generated by the convolution module consists of 64 components and has dimensions of 5x5x64. The convolution layer is equipped with 1600 settings. All objects in these classes are fed into the CNN and trained to decrease the error of the loss function (see Figure). Despite undergoing training for 50 iterations, the model demonstrates minimal improvement after only 30 epochs.



For the movie posters of movie genre Comedy model is trained and summary of the training is depicted in Figure, an after 30 epoch less reductions is observed in the loss function.



Performance metrics considered for assessment of classification model accuracy, f1- score and precision. Accuracy determines effectiveness of the classification model in classifying the images into their categories/genres.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

Precision focus on number of correctly classified images divided by the number of images labelled by the model as positive.

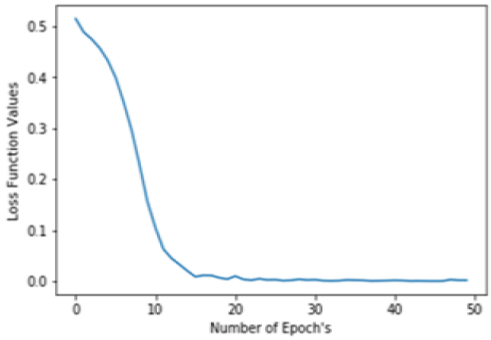
$$Precision = \frac{TP}{TP + FN}$$

TP is number of true-positive predictions. FP is number of false-positive predictions. Similarly, recall focus on number of correctly classified positive images divided by the number of positive images in the dataset considered.

$$Recall = \frac{TP}{TP + FN}$$

TP is number of true-positive predictions. FN is number of false-negative predictions. F1 score also known as F measure, focus in the model's ability in attaining the balance between precision and recall.

$$F = 2 * \frac{Precision * Recall}{Precision - Recall}$$



Genre Type	Classification Performance Metrics		
	Accuracy	F1 Score	Precision
Thriller	0.9184	0.8454	0.9169
Comedy	0.834	0.6551	0.7451
Drama	0.7265	0.5551	0.6531

The results displayed in the Table were obtained following the fine-tuning of the model's hyperparameters. The proposed architecture features an extensive array of hyperparameters, owing to its 19+ layers. Initially, the collected data is divided into a 80:20 ratio for the training and testing datasets. The model is then trained on the training set to acquire the parameters. Subsequent experimental analysis on the model using the testing datasets yielded convincing outcomes. To enhance performance, the model was re-trained with a reduced learning rate compared to the initial model training. However, only the rightmost layers, which bear more semantic representation, were re-trained, leaving the parameters of the earlier layers unchanged. Moreover, new dense layers and dropout layers were added. This led to a substantial improvement in the model's performance. Undertaking cross-validation during training allowed the model to attain superior knowledge representation without the risk of overfitting.

The evaluation of the proposed product-based filtering method utilized accuracy metrics such as mean average precision (MAP), recall, and precision. These metrics were employed to conduct a comparative analysis of the performance of non-personalized, user-based, and matrix factorization techniques. The results of this analysis are depicted in Table. Recall at 'k' is a metric that assesses the completeness of recommendations by determining the percentage of top relevant products that are included in the top 'k' recommendations.

$$Recall@k = \frac{\text{Relevant products recommended in top} - \text{k}}{\text{Relevant products}}$$

Precision@k focus on measuring the degree of soundness of the recommender in generating recommending products that are relevant, it determined

as percentage of top ‘k’ recommended products that are relevant.

$$Precision@k = \frac{\text{Relevant products recommended in top} - \text{k}}{\text{k Products recommended}}$$

Mean average precision (MAP) focus on recommenders’ ability in generating list of recommendation with relevant products ahead of irrelevant products.

Methods	Metrics				
	MAP	Recall@10	Recall@20	Precision@10	Precision@20
POP	0.357	0.142	0.208	0.084	0.070
User based CF	0.453	0.238	0.354	0.253	0.219
SVD Factorization	0.474	0.261	0.380	0.295	0.249
Proposed Product based Filtering	0.546	0.365	0.402	0.345	0.309

The proposed filtering method has exhibited efficient results when compared to user-based and non-personalized techniques in terms of the considered metrics. Both user-based and product-based filtering methods are influenced by the neighborhood size, with performance improving as the neighborhood size increases from 10 to 30. The proposed approach outperforms user-based techniques for all neighborhood sizes considered. The proposed filtering approach utilizes static similarity measures and precomputation of product neighborhood. The sensitivity of the proposed model was examined by varying the number of products used for similarity computation from 10 to 100 in increments of 10, and it was observed that the MAP values increased for a number of products set at 40, after which the metric decreased. The precomputation of product neighborhood resulted in better response time compared to user-based filtering and matrix factorization.

6. Conclusion

This work explores the challenge of inferring a customer's hidden preferences using product images they have purchased. To address this, a product-based recommendation algorithm and a novel method for extracting latent preferences using a convolutional neural network (CNN) are introduced. The study proposes a strategy for generating top-n product suggestions using stacked neural networks. The recommendation process involves classifying the target item into the most relevant category and filtering out similar items within that category. The classification model is a CNN stacking network with fully connected layers at the top, trained on item metadata. Items are filtered based on elements of their metadata, specifically their images with similar content. This study recommends using pre-trained CNN architectures (VGG19) to identify image characteristics. This method provides a useful way to locate and capture the picture feature map of objects for image similarity. The experimental results, which were first published in this paper, can be improved through model tuning. The technique is practical and provides top-n suggestions with high-quality values for relevant quality metrics, using the item's image as additional information. The algorithm also has a faster response time compared to user-based and matrix factorization recommendation methods. However, the experiment highlights the limitations of neural networks, as the classification model becomes more accurate with more training records. This research opens up possibilities for further development of deep learning integration to enhance recommendation models with more relevant data. Based on the results of the experiments, it can be concluded that using an ensemble of pre-trained models for item labeling with the category can improve performance, which will be useful for future extensions of this study. Moreover, dynamic customer behavior can be effectively modeled, unlike when using static models.

References

[1] Kang et al., "Visual Product Recommendation with Deep Learning", CVPR 2017.

[2] Zhang et al., "Personalizing User Experience through Visual Features in E-commerce Recommender Systems", Expert Systems with Applications, vol. 145, pp. 396-408, 2020.

[3] Wang et al., "Product Image Based Collaborative Filtering for Personalized Recommendation", CIKM 2017.

[4] Li et al., "A Survey on Visual Feature Representation Methods for Product Recommendation", IEEE Access, vol. 8, pp. 177541-177558, 2020.

[5] Guo et al., "Deep Learning for Personalized Product Recommendation: A Review", IEEE Transactions on Multimedia, vol. 22, pp. 2165-2178, 2020.

[6] He et al., "Robust Fashion Recommendation via

Jointly Modeling Attribute Preference and Sequential Order," IJCAI 2016.

[7] TMDb API, available at <https://www.themoviedb.org/documentation/api>.

[8] Yin, X., & Akata, Z. (2017). Sparse representation of users' fine-grained fashion preferences for personalized recommendation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 1(1), 140–149.

[9] Chen, L., Wu, Q., Ma, W., Sun, G., & Yang, M. H. (2012). Visually similar clothing recommendation. Proceedings of the 1st ACM conference on Recommender systems - RecSys '12, 91–98.

[10] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[11] Lin, T.-Y., Chen, H.-T., Chuang, H.-H., Tsai, P.-S., & Lee, D.-S. (2015). Learning rich representations using deep neural networks for web search ranking at Baidu. Proceedings of the 24th international conference on World wide web, 625–636.

[12] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th international conference on World wide web, 285–295.

[13] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.

[14] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

[15] Fukushima, K. (1988). Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics, 56(4), 193-202.

[16] Krizhevsky, A., Sutskever, I., & Hinton, G

[17] Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Transactions on Internet Technology (TOIT) 2 (2004) 104--123.

[18] Breese, J.S., Heckerman, D., Kadie, C.: Empirically determining tau for use in a bayesian interpolation exttt{k}-nearest neighbor collaborative filtering algorithm. User Modelling and User-Adapted Interac tion 8 (1998) 239--254.

[19] Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. Advances in neural information processing tems. Cambridge University Press, New York, NY, USA (2008) 1257--1265.

[20] Zhang, Y., Wang, M., Yeung, D.Y.: Deep learning based recommender systems: A survey and new perspectives. ACM Transactions on Intelligent Systems and Technology 10 (2019) 1--29.

[21] Hidasi, Balázs, Olga Kamenetsky, Ferenc Huszár, Domonkos Tikk, Barnabás Póczós, and Tamás Horváth. "Session-Based Recommendations with Recurrent Neural Networks." ArXiv abs/1511.06939 (2015): n. pag.

[22] Ouyang, Wei, et al. "Transferring knowledge between heterogeneous regions for region-level sentiment tclassification." Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. ACM, 2014.

[23] Ricci, Francesco, Lior Rokach, and Bracha Shimshoni. Introduction to recommender systems handbook. Springer Science & Business Media, 2011.

[24] Su, Bin, and Hongning Wang. "Collaborative filtering for personalized news video recommendation." Proceedings of the 2009 ACM SIGMOD international conference on Management of data. ACM, 2009.

[25] Adomavicius, George, and Alexander Tuzhilin. "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions." IEEE transactions on knowledge and data engineering 17.6 (2005): 734-749.

[26] Chen, Jingyuan, et al. "Neural collaborative filtering." Proceedings of the 26th international conference on world wide web. ACM, 2017.

[27] Covington, Paul, Jay Adams, and Emre Sargin. "Deep neural networks for youtube recommendations." Proceedings of the 10th ACM conference on recommender systems. ACM, 2016.

[28] Guo, Junjie, et al. "Deepfm: A factored gradient boosting machine for ctr prediction." Proceedings of the 24th acm sigkdd international conference on knowledge discovery and data mining. ACM, 2018.

[29] Salakhutdinov, Ruslan, Andriy Mnih, and Geoffrey Hinton. "Restricted boltzmann machines for collaborative filtering." Advances in neural information processing systems. 2007.

[30] Okura, Shumpei, Koji Eguchi, and Masatoshi Yoshikawa. "Drmm: Deep reinforcement learning for session-based recommendation." Proceedings of the 2017 ACM conf on info and knowledge management. ACM, 2017.

[31] Kim, Younggyo Seo, et al. "Convolutional neural network for sentence classification." EMNLP 2014,

Baltimore, Maryland, October 25-29, 2014.

[32] Aharon, Michal, and Ronny Lempel. “Image retrieval based on subimage matching.” IEEE transactions on image processing 8.8 (1999): 1162-1168.

[33] He, Xiaotong, et al. "Efficient food recommendation based on visual features." Proceedings of the 2016 ACM on multimedia conference. 2016.

[34] McAuley, Julian, and Jiwei Li. "From folksonomies to taste spaces: Exploring the structure of user-generated tags." Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in informa- tion retrieval. 2009.

[35] Albadawy, Mahmoud, and Mohamed Abdalla. "An intelligent recommendation system for e-commerce: Systematic review and future directions." Journal of Retailing and Consumer Services 47 (2019): 101876.

[36] Cao, Yu, et al. "Attentive collaborative filtering for accurate and diversified recommendation." Proceedings of the 25th ACM international conference on information and knowledge management. 2016.

[37] Rawat, Divya, and Sanjeev Kumar. "Deep convolutional neural networks for image recognition: A comprehensive review." Digital Signal Processing 88 (2019): 102-120.

[38] Pan, Yao, and Qiang Yang. "A survey on transfer learning." IEEE Transactions on knowledge and data engineering 22.10 (2010): 1345-1359.

[39] Tan, Min-Li, and Kristin Bennett. "Ensemble learning for big data analytics: Current status and future directions." Big Data Analytics 1.2 (2016): 99-108.

[40] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[41] Chatfield, Kyle, et al. "Return of the devile: Large scale dataset meets large scale model." International journal of computer vision 113.2 (2015): 157-170.

[42] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." Nature 521.7553 (2015): 436-444.

[43] Ruder, Sebastian. "An overview of gradient descent optimization algorithms." arXiv preprint arXiv:1609.04747 (2016).

[44] Khosla, Abhinav, et al. "Supervised learning of good features." International conference on machine learning. Vol. 10. 2000.

[45] Hu, Fenghui, et al. "Collaborative filtering for large-scale recommendation problems." Proceedings of the 10th international conference on World Wide Web. 2001.

[46] Linden, Greg, Brent Smith, and Jeremy York. "Amazon.com recommendations: item-to-item collaborative filtering." International conference on management of data. 2003.

[47] Steffen, Dominique, and Andreas Geyer-Schulz. "Top-N recommendation revisited: Comparison of neighborhood-based and rank-based methods." Proceedings of the 2016 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 2016.

[48] Creutzburg, Matthias, Markus Nüttgens, and Tobias Schreck. "Analysis of rating patterns in user-generated contents." Proceedings of the 20th ACM international conference on Information and knowledge management. 2011.

[49] Melville, Nathan, David Mooney, and George Karypis. "A hybrid approach to collaborative filtering for very large databases." Proceedings of the seventh ACM conference on digital libraries. 2002.

[50] Ekstrand, Michael D., Joe Konstan, and John Riedl. "User modeling and the netflix prize." UMUAI 21.4 (2011): 333-362.

[51] Deshpande, Madhu, and George Karypis. "Item-based collaborative filtering recommendation algorithms." Proceedings of the 1999 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, 1999.