PRAGUE UNIVERSITY
OF ECONOMICS
AND BUSINESS

Acta Informatica Pragensia
2022, Volume 11, Issue 1, pp. 15–35
https://doi.org/10.18267/j.aip.167

Article Open Access

Image-based Product Recommendation Method for E-commerce Applications Using Convolutional Neural Networks

Pegah Malekpour Alamdari [1], **Nima Jafari Navimipour** [2,3], **Mehdi Hosseinzadeh** [4], **Ali Asghar Safaei** [5], **Aso Darwesh** [6]

[1] Department of Computer Engineering, Qeshm Branch, Islamic Azad University, Qeshm 7953163135, Iran
[2] Department of Computer Engineering, Tabriz Branch, Islamic Azad University, Tabriz 5157944533, Iran
[3] Future Technology Research Center, National Yunlin University of Science and Technology, Douliou 64002, Taiwan
[4] Pattern Recognition and Machine Learning Lab, Gachon University, 1342 Seongnamdaero, Sujeonggu, Seongnam 13120, Republic of Korea
[5] Department of Medical Informatics, Faculty of Medical Sciences, Tarbiat Modares University, Tehran, Iran
[6] Information Technology Department, University of Human Development, Sulaimaniyah 0778-6, Iraq

Corresponding author: Mehdi Hosseinzadeh (mehdi@gachon.ac.kr)

Abstract

Recommender systems (RS) are designed to eliminate the information overload problem in today's e-commerce platforms and other data-centric online services. They help users explore and exploit the system's information environment utilizing implicit and explicit data from internal e-commerce systems and user interactions. Today's product catalogues include pictures to provide visual detail at a glance. This approach can effectively convert potential buyers into customers. Since most e-commerce stores use product images to promote, arouse users' visual desires and encourage them to buy products, this paper develops an image-based RS using deep learning techniques. To perform the research, we use five convolutional neural network (CNN) models to extract the features of the products' images. Then, the system uses the features to calculate the similarity between images. The selected CNN models are VGG16, VGG19, ResNet50, Inception V3 and Xception. We also analysed four versions of the MovieLens dataset to demonstrate the accuracy improvement of the recommendations, including 100k, 1M, 10M and 20M. Results of the experiment showed a significant increase in accuracy compared with traditional approaches. Also, we express many related open issues including use of multiple images per item, different similarity metrics, other CNN models, and the hybridization of image-based and different RS techniques for future studies. This method also provides more accurate product recommendations on e-commerce platforms than traditional methods.

Keywords

Image-based recommender systems; Recommender systems; E-commerce; Deep learning; Convolutional neural network.

Citation: Alamdari, P. M., Navimipour, N. J., Hosseinzadeh, M., Safaei, A. A., & Darwesh, A. (2022). An Image-Based Product Recommendation for E-commerce Applications Using Convolutional Neural Networks. *Acta Informatica Pragensia*, 11(1), 15–35. https://doi.org/10.18267/j.aip.167

Academic Editor: Zdenek Smutny, Prague University of Economics and Business, Czech Republic

1 Introduction

The concept of e-commerce is defined as the conduct of commercial transactions online. It refers to the sale, delivery and payment of goods and services online. Thus, it includes various technologies, including mobile commerce, electronic funds transfers, supply chain management, Internet marketing and online transaction processing (Wigand 1997, Turban et al. 2002). Internet-based commerce offers many advantages, such as global reach, range of choices for consumers, shorter supply chains, lower costs, ease of use, trusted payment methods (Ratnasingham 1998), personalized portals and the ability to shop at home. On the other hand, its weaknesses include fraud and online insecurity, data privacy concerns, purchasing related to old experiences with products and services, and an over-reliance on electronic technologies. Businesses and consumers participate in e-commerce transactions in different ways. Businesses to businesses (B2B), businesses to consumers (B2C), and consumers to consumers (C2C) are examples. Amazon, eBay, Alibaba, Walmart and Netflix are some of the most tangible examples of e-commerce platforms worldwide. Most of the e-commerce platforms use some technologies such as features of social networks (Sharif et al. 2013), expert clouds (Navimipour et al. 2015, Hazratzadeh and Jafari Navimipour 2016), mobile facilities, big-data mining results, and even machine learning concepts to satisfy their customers and gain stability in the business world. Similarly, deep learning methods are widely applied to e-commerce recommender systems (Da’u and Salim 2019, Shoja and Tabrizi 2019).

Recommender systems (RS) are intelligent agents working in e-commerce platforms to continually adapt and learn users' behaviours to meet their interests. Thus, they consider customers' experiences and opinions and recommend items and products that they will find most relevant from among the possible results (Alamdari et al. 2020). Moreover, RS make better decisions when they know more about the features of consumers and items (Zhou et al. 2004, Huang et al. 2019). They collect more data from available resources to feed their algorithms and generate relevant concerns for users. Also, they implement facilities to enhance application adaption to each user (Resnick and Varian 1997). In addition to solving the information overload problem, RS assist users in finding products related to their preferences by providing personalized services (Adomavicius and Tuzhilin 2005). User-item interactions, such as ratings or buying behaviour, attribute information about users and items such as textual profiles or relevant keywords, are fundamental input data to feed basic models of RS. Collaborative filtering (CF) methods use the user-item interactions (Schafer et al. 2007, Alyari and Jafari Navimipour 2018), and content-based filtering (CBF) approaches employ user and item features (Lops et al. 2011). Specifying user requirements and external knowledge bases and constraints are essential information for KBF methods (Burke 2000). Demographic RS use demographic information about users to construct recommendations based on mapping specific demographics to ratings or buying propensities (Al-Shamri 2016). Hybrid systems can combine the strengths of different kinds of RS to create systems that can operate more robustly in a wide variety of settings (Burke 2002). Also, the significant benefits of RS for e-commerce are the revenue of high conversion rate, customer satisfaction, personalization and assistance in exploring the system for users.

This study aims to show the improvements of the accuracy in recommendations using image-based RS for e-commerce. Providing product recommendations based on visual features is currently an open research topic, especially for the fashion industry (Lu et al. 2021, Vasudevan et al. 2021). Visual recommendations are based on the significant impact of the visual appearance of items on customers’ decisions (He et al. 2019). Convolutional neural network (CNN) models are a useful approach to extracting visual features (Mbelwa 2021). CNN models are categorized under deep learning methods as part of machine learning approaches. In this case, the user visits the e-commerce platform; then, the system recommends products according to features extracted from product images. We employed the CNN models to extract the features of images. Then, we calculated the similarity matrix for the products to use conventional methods such as collaborative filtering to emphasize the accuracy of the recommendations. Thus, the contributions of this paper are as follows:

- finding a suitable CNN model for the problem statement of increasing the accuracy of recommendations;
- describing image-based methods using deep learning models;
- comparing CNN models based on images used in e-commerce platforms and reporting the results by considering the selected CNN models.

The rest of the work is arranged as follows: Section 2 describes existing studies using the machine learning and image-based approach in RS related to the proposed method. Section 3 expresses the proposed method. The experimental method and evaluation results are provided in Section 4. Finally, the conclusion and suggested future studies are expressed in Section 5.

2 Related Work

Several studies have been carried out on using CNN models for RS in e-commerce platforms because they are potent tools in processing unstructured multimedia data such as images utilizing convolution and pool operations. Feature extraction is the fundamental consideration task for most CNN-based RS. CNN models can enable the system to use feature representation from data sources such as images, audios and videos. This section briefly reviews several papers that have dealt with the CNN-based recommendation approach.

Shankar et al. (2017) presented a unified end-to-end approach to building a large-scale visual searching and recommendation system for e-commerce using a unified deep convolutional neural network architecture. They designed a method to learn embedding by taking the notion of visual similarity across several semantic granularities using image retrieval by comparing it against the state of the art on the Exact Street2Shop dataset. The result showed that a visual recommendation engine is a powerful tool in the hands of any e-retailer.

He and McAuley (2016) examined the application of visual features of product images for personalized ranking tasks on implicit feedback datasets. They introduced a visual Bayesian personalized ranking (VBPR) algorithm, which incorporates visual features extracted from the CNN model into matrix factorization (MF). The results using multiple large real-world datasets show the significantly outperforming ranking methods. Also, He and McAuley (2016) extended VBPR by examining the user’s fashion awareness and the evolution of visual factors that a user considers when picking products. Their scalable approach utilized the deep CNN feature extractor to model the visual dimensions of items as well as the associated temporal dynamics.

Chu and Tsai (2017) utilized visual data effectiveness, including images of food and restaurants, to develop a restaurant recommendation method. They also used the CNN model to extract visual features and text representation of input data for MF, Bayesian personalized ranking matrix factorization (BPRMF), and factorization machine (FM) methods to test the approach performance. Their results show an improvement in the performance to some degree but not significantly.

Vandecasteele et al. (2017) proposed a (semi-)automatic interactive tagging process of objects in video streams using novel and state-of-the-art deep learning concepts and methodologies. They explained how deep learning-based video analysis techniques facilitate video summarization, semantic keyframe clustering and (similar) object retrieval. Also, they provided insights into user tastes by performing evaluation and optimization of application users’ experience. The proposed method enables intelligent interaction between TV audiences and brands, so producers and advertisers can offer potential consumer-tailored promotion, e-shop items and free samples. Thus, the method engages the user by using an e-commerce advertisement channel from video frames.

Yu et al. (2018) introduced a coupled matrix and tensor factorization model for an aesthetic-based clothing recommendation method. They used CNN models to extract image features and aesthetic features. The proposed method can capture users' aesthetic preferences based on experiments on real-world datasets.

Wang et al. (2017) studied the impacts of visual features on point-of-interest (POI) recommendation or location recommendation. They adopted CNN models to extract image features and proposed a visual content-enhanced POI recommender system (VPOI). They used probabilistic matrix factorization (PMF) to explore the interactions between visual content and (i) latent user factor and (ii) latent location factor.

Chen et al. (2017) explained a smart search engine for online shopping using the Amazon dataset and two CNN models, including VGG and AlexNet. They used neural networks to provide the closest product based on the product images. The results show an accuracy improvement of the method. Also, they used Jaccard similarity to calculate the similarity score. However, their study requires scalability, and they stay satisfied with about 0.3% of the dataset images due to limited computational resources.

Tuinhof et al. (2019) introduced an image-based recommendation system that uses a fashion dataset to train a CNN model to solve image classification tasks. They used the trained model as a visually aware feature extractor to feed the ranking system. Also, they used the ball tree search, a special implementation of the K-NN algorithm for simple ranking to overcome memory resource limitations. They concluded that the method can be helpful in other domains such as music.

In summary, many researchers have used CNN for visual searching and feature representation learning in RS. This paper focuses on using extracted features from image sources to propose an image-based recommendation technique. The following section explains the proposed method in more detail.

3 Methodology and System Architecture

Sparsity and accuracy are two known concerns for recommender systems. To overcome the sparsity problem, researchers use content data in addition to the rating matrix to produce recommendations. By design, the system extracts the features of images of items using pre-trained CNN models and makes a similarity matrix to provide more accurate recommendations. We did not change the default parameters of the pre-trained models used in this study. So, first, this section briefly explains the problem statement and then describes the proposed solution by considering such system requirements as image data preparation and applying the machine learning methods. Finally, the applied image-based method is described in more detail.

3.1 Problem statement

In this research, we are going to use product images. Thus, the proposed method is based on the similarity between image features because recommender systems generally consider similarity values between users and items. A similarity matrix can be produced based on implicit or explicit data to prepare and recommend items. Thus, the similarity process is the core problem in RS. For example, in collaborative filtering (CF) as the most widely applied and practical method used in RS, the system tries to predict the user's interest by measuring the similarity of items or users and uses them to update the ranking matrix and recommends the appropriate items using it (Leng et al. 2016). However, the system relies on predicting the association between users or items. For example, in the case of e-commerce RS, the user may buy the recommended product or help the system improve the quality of predictions.

New items or new users have either zero or minimal interactions in e-commerce platforms. So, the CF algorithms that work based on item interactions have problems making recommendations. Thus, the system needs time to have good interactions to make recommendations. This period for every new item or new user is called the cold-start problem. The cold-start problem occurs since the collaborative approach has a fundamental constraint on dealing with a new item or user. Some hybrid systems use other approaches such as content-based filtering (CBF) to resolve the problem. The lack of sufficient interaction data for new users or new items will prevent the system from listing and recommending new items.

Additionally, the system cannot filter personalized recommendations to new users. In some systems, from the registration process, new users' interests are requested using some questionnaire forms to get to know the users' general preferences. Thus, the CF method is appropriate when the model uses enough data to learn, such as a rating matrix and other implicit data. Accordingly, some developers try to include remarkable content-based and implicit data such as features to result in more accurate recommendations than the rating matrix alone.

The rating matrix is a matrix of rating values that the users have given to those products they buy. Since the number of users interested in post-purchase ratings on purchased items is deficient, this matrix is sparse, especially in early stages of the system, which means that many cells in the matrix are empty. Thus, cold start is a significant known problem in the CF method. The sparsity of the rating matrix is calculated using the following formula.

$$Sparsity = \left(1 - \frac{Number\ of\ zero\ elements}{Total\ number\ of\ cells}\right) \times 100 \rightarrow \\ (1 - \frac{Number\ of\ ratings\ in\ matrix}{(Number\ of\ users) \times (Number\ of\ items)}) \times 100 \qquad (1)$$

We used the MovieLens[1] dataset to make the experiments in our study. Table 1 shows the percentage of the zero elements of the rating matrix of four versions of the MovieLens dataset. Based on the table, MovieLens 20M is the sparsest dataset.

Modern e-commerce platforms use images and other media for their product profiles to focus on high-quality traffic and to turn their website visitors into buyers. Additionally, pictures give users a more natural sense of the products and attract them. Also, image-based systems support using visual features, which old systems ignore. Thus, the proposed method uses CNN models to extract features of images. The system calculates visual similarity using distance measurements such as the cosine method. It provides more accurate recommendations than using a rating matrix alone.

Table 1. *MovieLens dataset sparsity percentage.*

Dataset MovieLens	Rating	Movies	Users	% Sparsity
20M	20000263	27278	138493	99.47059
10M	10000054	10681	71567	98.69179
1M	1000209	3900	6040	95.75391
100K	100000	1682	943	93.69533

In this study, the core question is how to use product images to determine visual similarity between items in recommender systems. By running the visual similarity process offline, the system can generate recommendations as fast as possible. That makes it a significant advantage of the proposed method. Therefore, a key aspect is detecting similarity between images in order to enhance the quality of recommendations. For instance, similarity can be used in conjunction with other metrics to generate more

[1] Available from https://grouplens.org/datasets/movielens/

accurate results. This study also used image feature data and a rating matrix to improve the system accuracy.

3.2 Convolutional neural network approach

For improving the quality of recommendations, RS utilize implicit data. Additionally, more data help mitigate cold-start issues, especially for new items and users who lack interaction. In addition, extensive feature extraction enables the system to learn more accurately and behave efficiently. In this study, we extracted features by using convolutional neural networks (CNN). Convolutional models are in a particular category of deep learning methods. "Deep" refers to the presence of several hidden layers beyond traditional simple neural networks. The developers use CNN models to recognize and classify images, and if the system does not consider the last layer of the network, the model appears as a feature extraction mechanism. Accordingly, the proposed method uses CNN models to extract visual features. Once visual features are extracted, the feature vectors are used to make a similarity matrix. Both similarity and ranking data are used to make recommendations. As a result, the method maximizes recommendation accuracy. In this study, we use cosine measurement to compute similarity values (Huang et al. 2013, Elkahky et al. 2015).

Generally, CNN models classify input images. Image classification is a task of labelling the image with relative concepts. Similarity learning is a process of learning how similar two images are. For example, VGG, a CNN model proposed by Simonyan and Zisserman (2014), gives 92.7% top-5 test accuracy in ImageNet. It is a dataset of over 14 million images belonging to 1000 classes. Developers do not consider the last layer to use CNN models as feature extractors, so the output for each image is a feature vector based on the learned weights.

Table 2. *Pre-trained model input image size and weight file volume size.*

Model	Input size	Weight volume	Top-1 accuracy	Top-5 accuracy	Parameters	Depth	Feature vector size
VGG16	224x224	>500 MB	0.713	0.901	138,357,544	23	4096
VGG19	224x224	>500 MB	0.713	0.900	143,667,240	26	4096
ResNet50	224x224	~100 MB	0.749	0.921	25,636,712	168	2048
Inception V3	299x299	90-100 MB	0.779	0.937	23,851,784	159	131072
Xception	299x299	90-100 MB	0.790	0.945	22,910,480	126	2048

We selected five pre-trained CNN models to construct the proposed method, including VGG16, VGG19, ResNet50, Inception V3 and Xception. Based on the architecture of these models, the last layer is a classifier that recognizes 1000 different object categories, similar to objects we encounter in our day-to-day lives, with high accuracy. These CNN models are trained on ~1.2 million training images with another 50,000 images for validation and 100,000 for testing. Thus, many applications use them to provide machine learning visual detection, including personal assistant robots, self-driving cars and object detection usages. We used these pre-trained models to extract features from images of products. Next, the system uses feature vectors to determine the similarity values between each pair of images based on the cosine function. In this case, the proposed method uses the deep learning technique in feature extractor mode to make an image-based recommender system. Table 2 shows the input image size and weight volume file of the models. Therefore, in preparation for the input image, resizing the images needs to be considered related to the input size. It also shows the feature vector size for every model.

The classifier block is the last fully connected layer in a CNN model. In the absence of a classifier block, the model will return the feature vector of the input image. In this study, we do not examine the classifier block of all the chosen models.

3.3 Workflow and algorithm of the proposed method

The main goal of the proposed method based on using more data to overcome the cold-start problem is to increase prediction accuracy, especially when the user-item rating matrix has a considerable number of empty cells. In turn, high-quality recommendations can be provided. Additionally, sophisticated systems track user interactions such as clicking, liking, reviewing, ranking and ordering to achieve greater accuracy. Nevertheless, in this paper, we examined only the rating matrix and images of products. Therefore, we have the set of users U and items I to make the rating matrix R with m rows for users and n columns for items. Therefore, each entry (i, j) in this matrix contains the score that the user i sets for the item j. The main idea of collaborative filtering is to determine the rating of the user u on the item m; the system finds other items that are similar to the item m, and the user u's scores based on those similar items the method infer their rating on the item m. Usually, the system selects a limited number of the nearest neighbour items. In addition, the system uses pairwise metrics to compute the similarity between the items such as cosine and other distances. Figure 1 illustrates the experiment workflow of the proposed method. The details are outlined in the following section.

The CF mechanism recommends items based on how similar users liked similar items. However, the CBF method recommends items with similar metadata tags. Thus, to use CBF, the system requires more detailed data. In a hybridization approach to CF and CBF, the system has two requirements:

a) determining similar items based on image features; and
b) providing enough metadata tags (implicit or explicit data).

CNN models provide additional data, such as metadata tags, with similarity data. The system also uses a ranking matrix. It is essential to determine how to take a similarity measurement of items or users in this regard.

The cosine similarity method is a general way of comparing vectors. It produces values in a range from zero to one for positive-valued vectors. In geometry, when an angle ranges from 0° to 90° between two lines, the cosine similarity falls between 1 and 0. When computing cosine similarity for numerical vectors, the order of the values in the vectors is ignored. Therefore, the result of 1 does not necessarily mean that it is identical for two vectors of numbers. Following is the formula for calculating the cosine similarity between two feature vectors.

$$Similarity(u, v) = \frac{r_u \, r_v}{\|r_u\| \|r_v\|} \tag{2}$$

Where r_u and r_v are the latent feature vectors of the item u and the item v, respectively and $s(u, v)$ is just the cosine similarity measure between image features of the products u and v.

For example, we considered the fourth-to-last layer as a feature vector of the input image of a product using VGG16 in Keras[2]. We did the same task for other models. As the posters of movies from the MovieLens dataset are available on the TMDB website, we used the appropriate API codes to download their image files. Next, we extracted the feature vector of each image using the selected CNN models.

Algorithm 1 presents how we crawled the TMDB website and downloaded all movie poster files from it. The process diagram is shown in Figure 2.

[2] See, https://keras.io/

Algorithm 1. *Downloading movie posters using web scraping techniques.*

```
Input: Url_list from the MovieLens dataset
Data: Crawling the TMDB website to download movie posters.
Initialize: Initialize failure list to keep unsuccessful attempts of image
downloading to check later automatically or manually.
For url in Url_List do
     Try:
      Open the url in a programmable browser
      Wait until all elements of page are downloaded
      Find the poster element
      Download and save the image file on disk
      Failure:
          If any error occurred
             Add url in failure_list
```

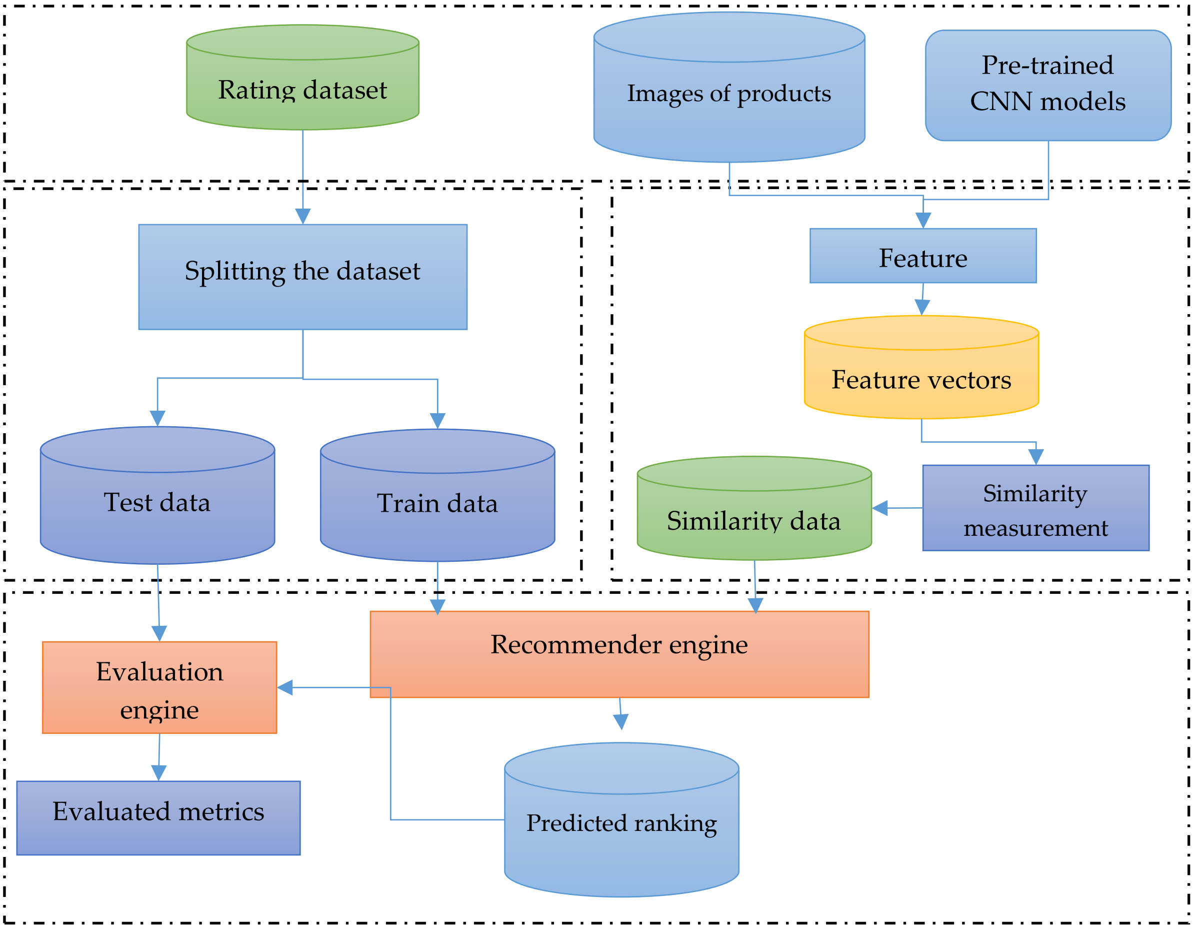

Figure 1. *Experiment workflow of proposed method.*

The similarity score computation algorithm takes two images as input and returns the similarity score. Two forms of results are as follows:

a) a binary label, i.e., 1 for the same images and 0 for a discrepancy; or
b) a real number showing how similar a pair of images are using ranges between 0 and 1.

Based on the CNN feature extraction approach, Figure 3 shows the image similarity calculation algorithm. The results are saved in a matrix for providing the recommended item based on traditional methods. Thus, the system uses the similarity value of each pair of images to generate the recommendation. However, real e-commerce platforms allow using multiple images for every product. Thus, the system incorporates other similarity checks into the processing module.

Feature extraction is performed using algorithm 2 written in Python. Also, we measured the run time to express the productivity of selected methods.

Algorithm 2.

```
1. fileResult = Open data file to record results of feature extraction process in
   write binary mode access.
2. Initialize feature vector size, width and height of input image.
3. Initialize model and load it into memory.
4. Read full path of all image files from disk to a list fileList.
5. For i in range(0, len(fileList))
     a. Image = load_image_file (fileList[i])
     b. X = preprocess result of image based on selected model
     c. Features = model.predict(X)
     d. FeaturesVector = convert features to vector based on selected model
     e. Save FilmId(i), FeaturesVector to fileResult
6. Close fileResult
```

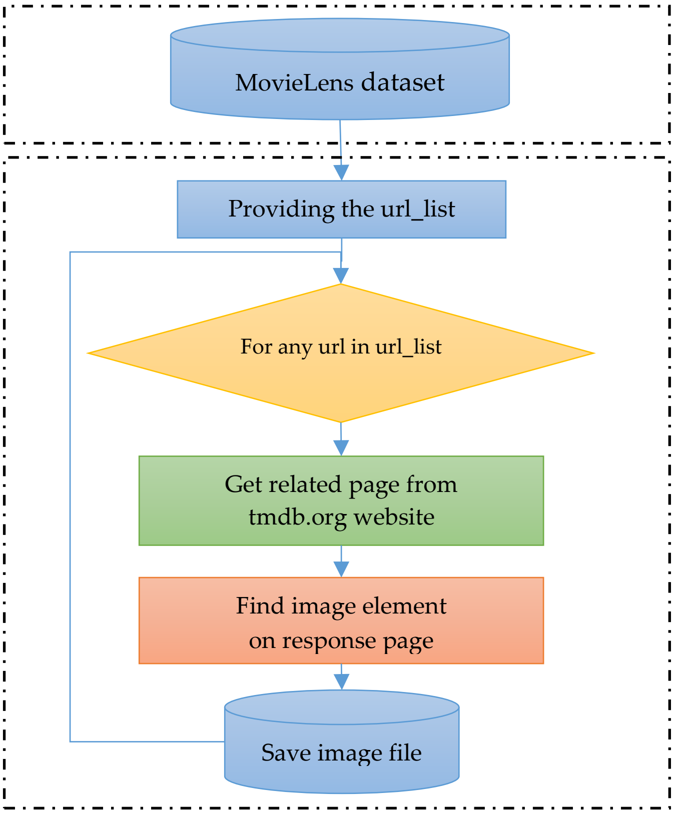

Figure 2. *Downloading process for all movie posters from TMDB.ORG website.*

3.4 Similarity calculation

Similarity is a pairwise metric such as the similarity value between the item i and the item j or $sim(F_i, F_j)$. Note that F_i is the feature vector of the item i. To compute the similarity of feature vectors of each pair of images, we used the cosine function. Thus, we prepared a similarity file based on the following formula:

$$Similarity\ file = \left\{ i, j, sim(F_i, F_j) \middle| i, j \in N, sim(F_i, F_j) = \frac{F_i\ F_j}{\|F_i\| \|F_j\|} \right\} \tag{3}$$

Algorithm 3.

```
1. SimilarityResult = Open data file for recording similarity of feature
   extraction vectors in write text mode access
2. Initialize feature vector size, width and height of input image
3. Initialize model and load it into memory
4. Read all feature vectors for a model to a list featuresList
5. For $i$ in range(0, len(featuresList)-1)
      a. For $j$ in range($i+1$, len(featuresList))
      b. $SimValue\ = Similarity\_Value(featuresList[i], featuresList[j])$
      c. Save $i$, $j$, SimValue to SimilarityResult
6. Close SimilarityResult
```

As a consequence of Algorithm 3, a large amount of memory is required to load all feature vectors in the first step. The program performance may be negatively affected due to the number of resources used by this task, such as memory management. Therefore, we changed the algorithm to do the task in blocks of data. The similarity calculations are performed by configuring the block size to 2000 features. For faster results, parallel computation can be applied. It also covers the dynamic changes of the e-commerce environment, such as inserting new products in catalogue databases.

3.5 Recommendation generator

For all images of products, the system performs the following steps:

- feature extraction based on selected CNN models for every single image; and
- similarity score calculation for all pairs of images.

The computational cost is directly affected by the number of images and size of the feature vectors. Inception V3 takes more time and uses more memory resources than others, for example. Xception and ResNet50 vectors are also processed faster than VGG16 and VGG19 vectors.

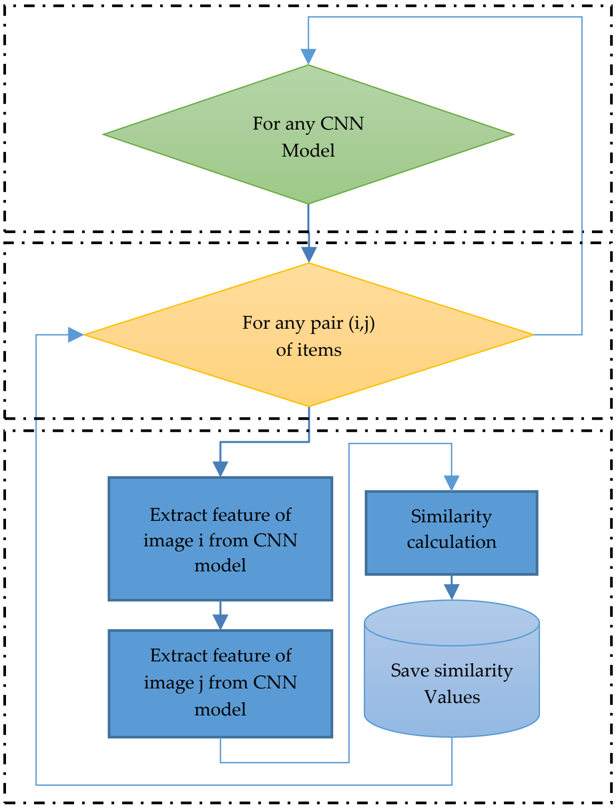
For any CNN Model
For any pair (i,j) of items
Extract feature of image i from CNN model
Extract feature of image j from CNN model
Similarity calculation
Save similarity Values

Figure 3. *Image similarity calculation algorithm based on CNN feature extractor approach.*

We assumed that the similarity between extracted features and the predicted rating values related directly to this study. Consequently, we used different settings for each selected CNN model during the experiments. We therefore generated five similarity matrices of extracted features based on the selected CNN models to check RMSE and MAE metrics for each model. A CF method was used to measure metrics and compare the results to evaluate the proposed method.

4 Experimental Results

In this section, we present the experimental results of the proposed method. Five models and four datasets are used to compute RMSE and MAE metrics. Experimental scenarios and details about the datasets are also given. The experimental design is described briefly in Section 4.1, along with the dataset, tools and other preparation settings. As part of section 4.2, we introduce the MAE and RMSE metrics that we used to evaluate the proposed method. Based on the experiments, it is explained in more detail in Section 4.3 why the method has been improved significantly.

4.1 Experiment design

Users' views of product images on an e-commerce website are critical to preserving their interest and engaging them to order the recommended items (Maros et al. 2019). Thus, we used movie posters to illustrate the importance of using images as part of the recommendation process to improve the quality of recommendations. Using web scraping techniques, we downloaded posters of movies from The Movie Database[3] (TMDB) website using the MovieLens dataset and the links.csv file.

[3] See, https://www.themoviedb.org/

Python is used for coding the required programs. Other libraries used include TensorFlow[4] and Keras. For the experiments, we also used pre-trained CNN models. The program code was run on a personal computer. First, we obtained feature vectors of the images using the selected CNN models. Then, using cosine similarity, we calculated and saved the similarity between every pair of images for all pictures. The number of computational resources consumed and process time varies according to the feature vector length in different CNN models. The most extensive length is for Inception V3. On the contrary, ResNet50 and Xception have the smallest vector length.

4.2 Metrics

The primary purpose of the proposed method is to predict the rating values for those items that users do not rate; thus, to evaluate the performance of the proposed method, we used the root mean square error (RMSE) and the mean absolute error (MAE). The ideal method would be to minimize this value to improve the accuracy of the recommendations.

Suppose that $\hat{R}$ is the fully predicted data resulting from the proposed model and R_{test} is testing data. Therefore, the RMSE formula is:

$$RMSE(\hat{R}, R_{test}) = \sqrt{\frac{1}{|R_{test}|} \sum_{(i,j)\in R_{test}} (r_{test,ij} - \hat{r}_{ij})^2} \quad (4)$$

Where $\hat{R}$ is the predicted data, $|R_{test}|$ stands for the number of testing sets, and $\hat{r}_{ij}$ is the predicted rating value of the item i that is rated by the user j.

In addition, the MAE is an average of the absolute errors $|e_i| = |y_i - x_i|$, where y_i is the prediction forming the proposed method and x_i is the true data from the dataset. Thus, the MAE formula is:

$$MAE = \frac{\sum_{i=1}^{n} | y_i - x_i |}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n} \quad (5)$$

Also, we partitioned the rating records into two sections. Thus, we used 80% of the data to train the model and the other 20% to test the predicted results.

4.3 Obtained results

This subsection shows the results of the proposed method. We tested the method using pre-trained CNN models, including VGG16, VGG19, ResNet50, Inception V3 and Xception. We used those five matrices to compute the top-k recommendation item based on visual similarity between images. Therefore, each model takes the product images, and then the system makes the similarity matrix. Next, we compared the results to specify the best model.

All data manipulations and processes were made using Python. Because the total number of images was 26,938 files, the system produced 725,655,844 similarity values for each model. Finally, the system uses the resulting data to get the nearest neighbours, especially for products with no ranking feedback from users.

4.3.1 Splitting the dataset

It is essential to consider the accuracy of the proposed method. We first split the dataset into two parts. The system utilizes 80 percent of the data for the training task and the remaining 20 percent to determine the method accuracy. For each train and test set in the selected dataset, Table 3 shows the number of users and items and the ranking and sparsity. The information indicates that the test data are sparser than the

[4] See, https://www.tensorflow.org/

train data for all the datasets. The experiments attempt to demonstrate that the MAE and RMSE metrics decrease when the proposed method is applied. Though k-fold cross-validation is more valuable than the fixed-rate splitting method and effectively evaluates a model over a small dataset by training multiple models, it is computationally expensive. As such, the method is not employed in the current work.

Table 3. *Splitting of dataset versions.*

Dataset MovieLens	Split data	Users	Items	Ranking	Sparsity
100k	Train	943	1656	80000	94.88%
	Test	940	1400	20000	98.48%
1M	Train	6040	3680	800167	96.40%
	Test	6034	3444	200042	99.04%
10M	Train	69878	10644	8000043	98.92%
	Test	69808	10230	2000011	99.72%
20M	Train	138493	25865	16000210	99.55%
	Test	138313	20304	4000053	99.86%

4.3.2 Traditional method evaluation

We chose the neighbourhood-based collaborative filtering method to compare with the proposed method. The philosophy of nearest item neighbourhood for rating prediction is to determine the rating of the user u on the item m; the system can find other items that are similar to the item m, and based on the user u's ratings on those similar items, the system infers his/her rating on the item m. Therefore, we implemented the Item KNN algorithm (Deshpande and Karypis 2004) on all selected datasets and obtained the RMSE and MAE values. Table 4 shows the metric values in each dataset. According to this table, MoveLens 20M shows more accuracy than the other datasets. However, using the proposed method indicates a significant reduction in RMSE and MAE values.

Table 4. *Results of applying Item KNN to datasets.*

Dataset MovieLens	Users/ items	rating	MAE	RMSE
100k	943 1,682	100,000	0.779812	0.997992
1M	6,040 3,900	1,000,209	0.752643	0.977491
10M	71,567 10,681	10,000,054	0.715044	0.956686
20M	138,493 27,278	20,000,263	0.706612	0.945456

Figure 4 shows the RMSE resulting from using Item KNN on four versions of MovieLens. According to Figure 4, the RMSE value is inversely related to the rating records. Thus, the higher the amount of the data, the lower the amount of the RMSE result.

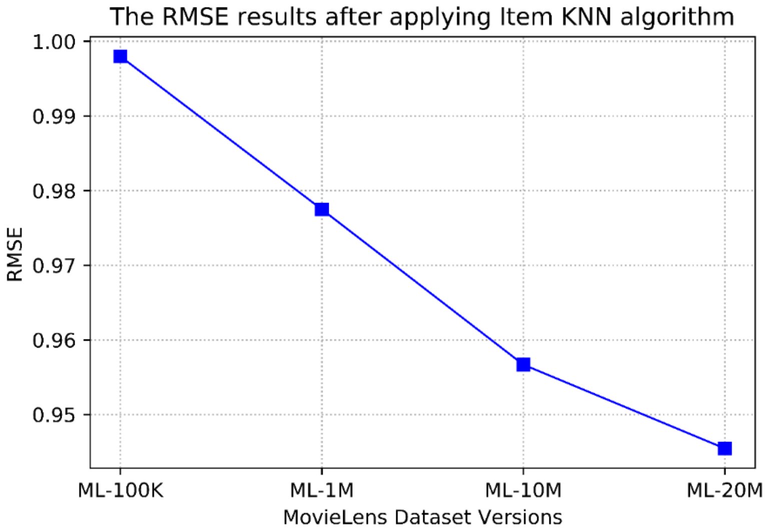

Figure 4. *RMSE results for four versions of MovieLens.*

In addition, Figure 5 shows the MAE values resulting from using Item KNN algorithm on the four datasets. Based on Figure 4 and Figure 5, the minimum values of RMSE and MAE are for the MovieLens 20M dataset. We used the test file to obtain the values.

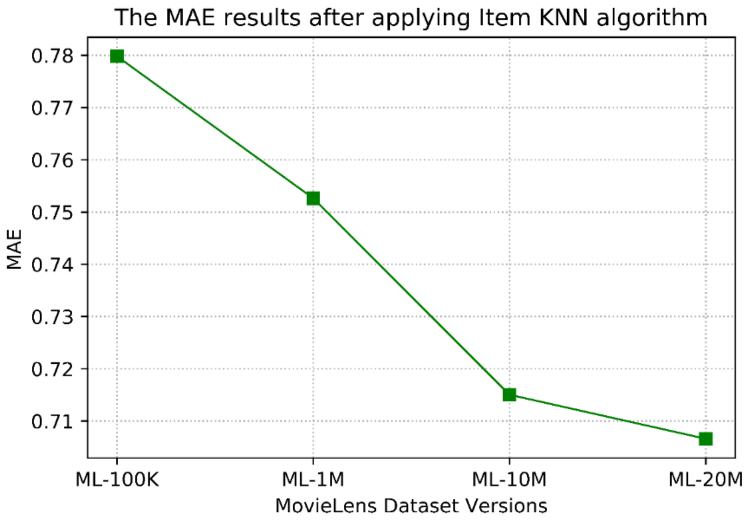

Figure 5. *MAE results for four versions of MovieLens.*

4.3.3 Proposed method evaluation

This subsection expresses the comparison resulting from MAE and RMSE metrics between the Item KNN as a traditional method and the proposed method. We used five pre-trained CNN models, and the results include all values when applied to the selected versions of the MovieLens dataset. Based on the values, the proposed method provides better accuracy than the traditional method in all the experiments. Table 5 shows the comparison metrics for MovieLens dataset versions.

Table 5. *Comparison metrics for MovieLens Dataset versions.*

Dataset version	CNN model	MAE	RMSE
ML-100K	VGG16	0.737082	0.945206
	VGG19	0.738427	0.947529
	ResNet50	0.739471	0.948777
	Inception V3	0.731102	0.937454
	Xception	0.735855	0.942896
ML-1M	VGG16	0.684183	0.871793
	VGG19	0.688413	0.876232
	ResNet50	0.687604	0.875703
	Inception V3	0.687806	0.875828
	Xception	0.687471	0.875213
ML-10M	VGG16	0.61925	0.805667
	VGG19	0.619261	0.805863
	ResNet50	0.618356	0.804792
	Inception V3	0.61878	0.805224
	Xception	0.619648	0.80635
ML-20M	VGG16	0.60644	0.794647
	VGG19	0.607222	0.79564
	ResNet50	0.609503	0.798504
	Inception V3	0.608039	0.796844
	Xception	0.611336	0.800474

A comparison chart of the RMSE values in dataset versions and selected methods are shown in Figure 6. Figure 6 shows that Item KNN has higher RMSE values when using the traditional algorithm than when using the proposed method. MovieLens 20M is also the sparsest dataset of all the versions. Also, its lowest RMSE values show the method effectiveness, when data are sufficient despite the high sparsity rate.

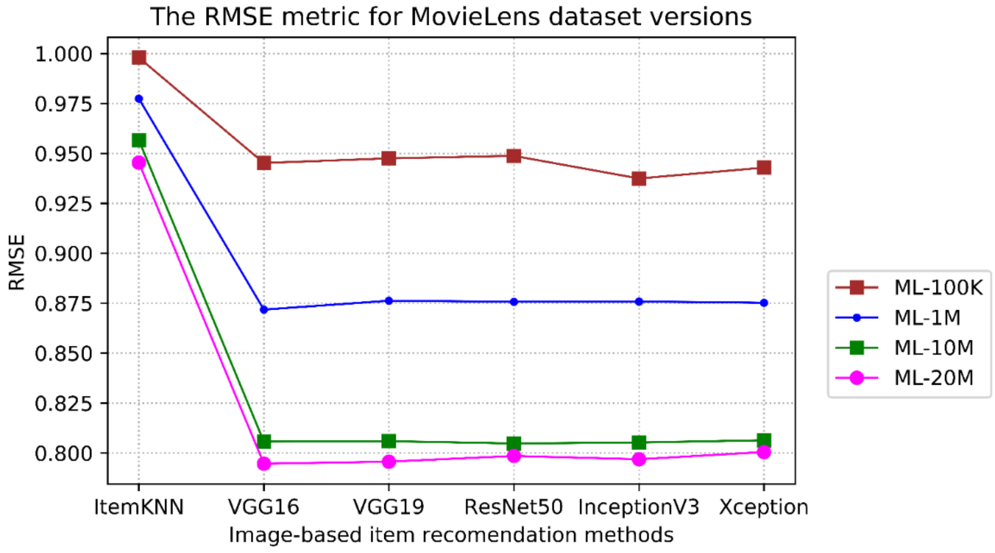

Figure 6. *Comparison of RMSE values for traditional method and proposed method.*

Figure 7 shows the MAE values. Based on the results obtained, VGG19 achieves better accuracy than others.

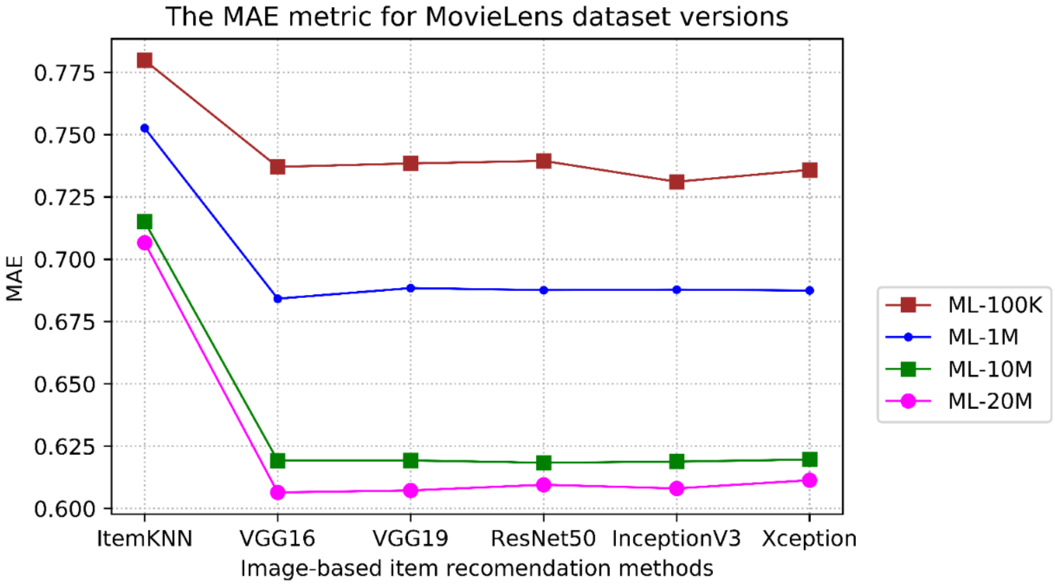

Figure 7. *Comparison of MAE values for traditional method and proposed method.*

According to the results, the proposed method is more accurate than traditional methods. Additionally, VGG16 predicts more accurate results for higher data volumes than other models. In general, when using the selected models, the system shows little difference regarding RMSE. In this way, the proposed method is more accurate than previous methods regardless of the type of pre-trained CNN model.

Thus, the RMSE and MAE values from all datasets are considered in the proposed and traditional methods. The RMSE was lowest in VGG16, while it was highest in the traditional method. In general, these results indicate that using image-based models for recommender systems will increase the quality of recommendations.

We set our experimental configuration based on K=20 for nearest neighbours. Also, to show different numbers of neighbours and the impacts of them, we made the experiment for MovieLens 100k, which has higher RMSE and MAE values based on Figures 6 and 7. Thus, we considered a set of neighbours for $K = \{n \mid n \in (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)\}$.

Figure 8 shows the MAE, and Figure 9 shows the RMSE values with several numbers of item neighbours for MovieLens 100k. Based on the results, the proposed method has better rating prediction accuracy related to traditional methods.

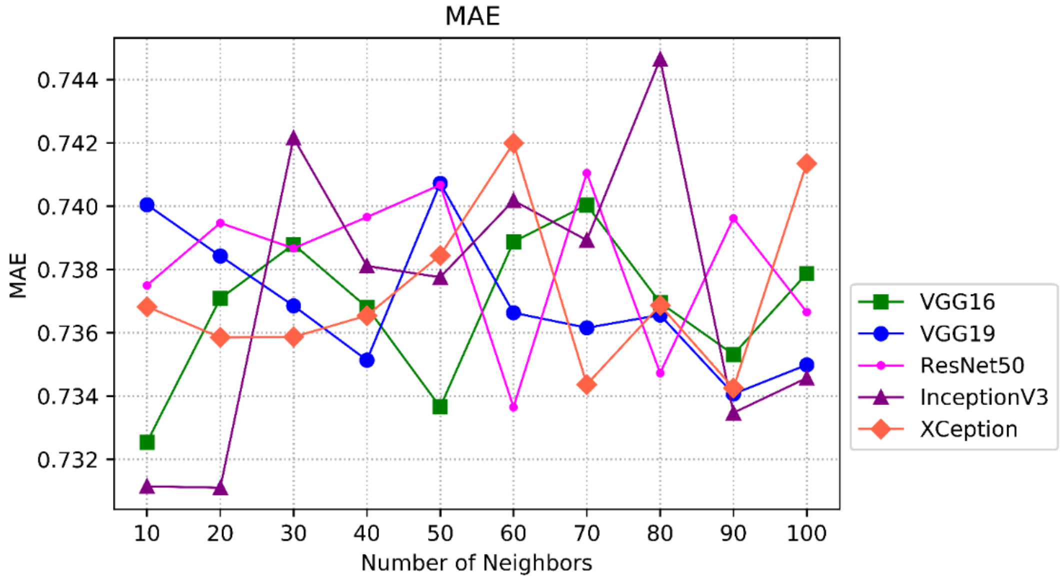

Figure 8. *Comparison of rating prediction accuracy based on MAE metric with different user neighbour numbers for selected model.*

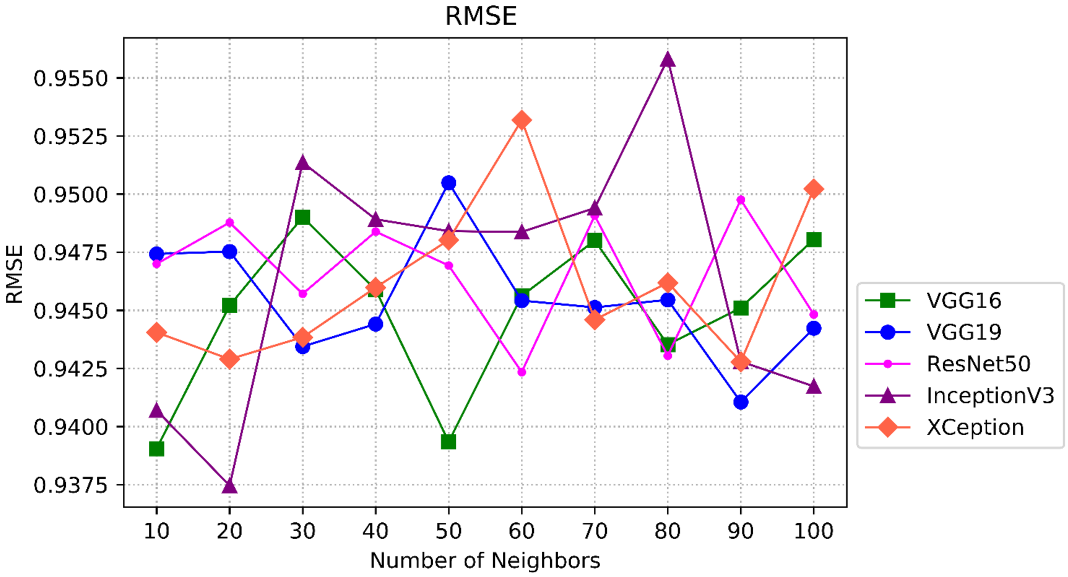

Figure 9. *Comparison of rating prediction accuracy based on RMSE metric with different user neighbour numbers for selected model.*

Figures 10 and 11 show the difference between the traditional method and the proposed method on both MAE and RMSE resulting from the proposed method and Item KNN for different numbers of neighbours. Based on Figure 10, all CNN models produce lower MAE values than the traditional method, especially when we increase the number of neighbours.

Figures 8 and 9 show the RMSE and MAE values when using different neighbours to run the Item KNN algorithm. The difference is max-min = 0.750-0.730 = 0.02 for the MAE metric and max-min = 0.09575-0.9350 = 0.0225 for the RMSE in all models. However, Figures 10 and 11 show a valuable difference in comparison with the traditional method. The difference in values in the whole number is minimal, and the figures show that even if we increase the number of neighbours in this algorithm using neural models, it will not change much; however, as the number of neighbours grows in the traditional method, the RMSE and MAE increase.

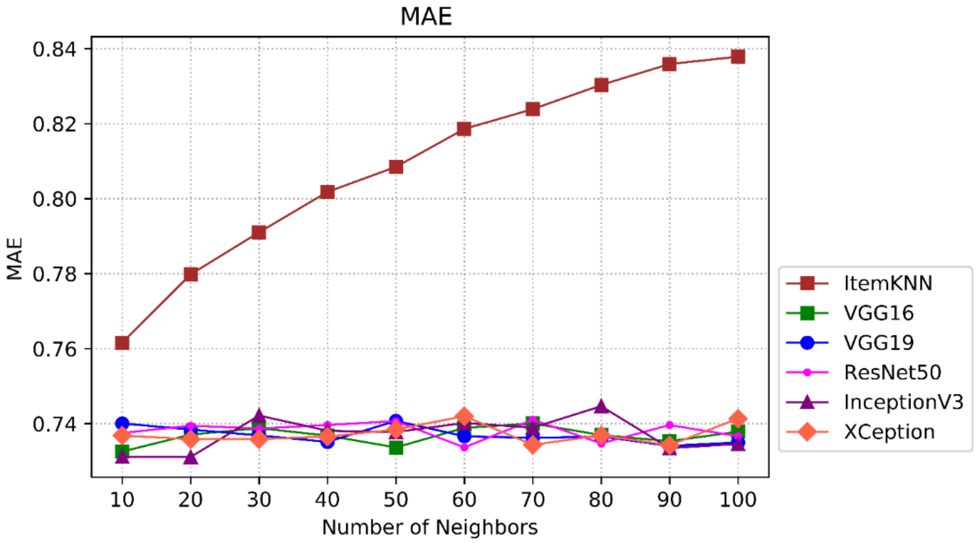

***Figure 10.** Comparison of rating prediction accuracy based on MAE of proposed method and traditional method.*

Figure 11 emphasizes the better accuracy of the proposed method than traditional ones when we apply the algorithms to MovieLens 100K. Therefore, the image-based method can increase the accuracy of recommendations.

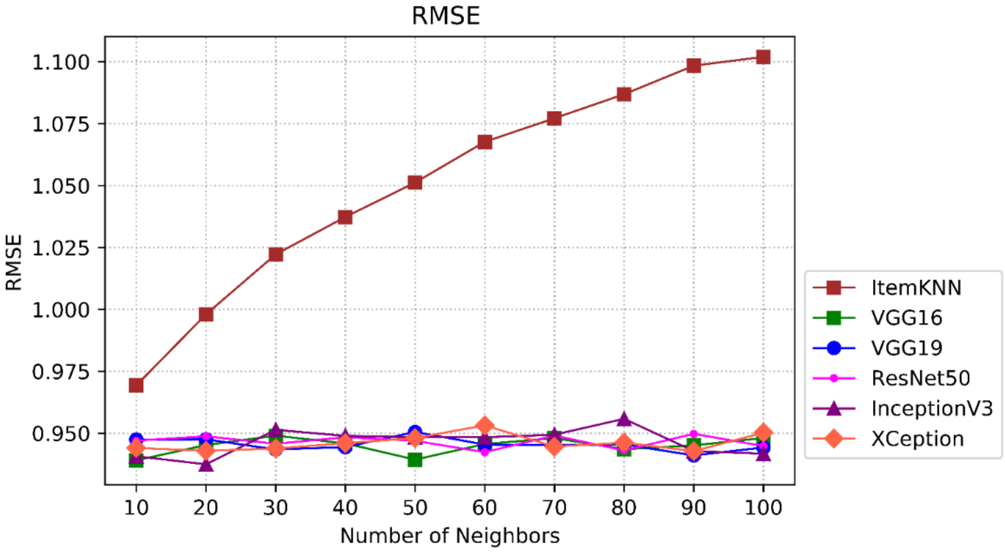

***Figure 11.** Comparison of rating prediction accuracy based on RMSE of proposed method and traditional method.*

5 Conclusion and Future Work

Enhancing the accuracy of recommendations in any data retrieval system, especially recommender systems, requires analysing both implicit and explicit data. The paper proposes an image-based recommendation method based on convolutional neural networks using five pre-trained models. The method uses features extracted from product images to formulate better recommendations. By using cosine similarity, each pair of latent feature vectors is scored on their similarity. To validate the importance of using image-based RS, we use four versions of the MovieLens dataset. The CNN models are VGG16, VGG19, ResNet50, Inception V3 and Xception. The recommendation algorithm in this research is the Item KNN algorithm. Compared to traditional methods, the proposed method provided lower RMSE and MAE values. This study demonstrates the improved performance of the proposed method by means of comparison tables and charts.

As a result, we make the following suggestions for further investigation, aiming to improve the accuracy and relevance of recommendations based on images of products:

- using multiple images and some video frames for each item;
- using other similarity or distance methods;
- using other pre-trained CNN models;
- using a hybridization approach for multiple CNN models or multiple similarity scores as well as using multiple RS techniques; and
- using new CNN models.

For future studies, we also recommend examining the computational complexity and resource limitations when using more data.

Additional Information and Declarations

Acknowledgements: We would like to express our gratitude to all the authors and contributors whose works we used to complete our study.

Conflict of Interests: The authors declare no conflict of interest.

Author Contributions: P.M.A.: Data curation, formal analysis, investigation, programming, methodology, writing – original draft; N.J.N.: methodology, project administration, supervision, writing – review & editing; M.H.: Supervision. A.A.S.: Validation, writing – review & editing; A.D.: Writing – Reviewing and Editing.

Data Availability: The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Adomavicius, G., & Tuzhilin, A. (2005). 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), 734–749. https://doi.org/10.1109/tkde.2005.99

Al-Shamri, M. Y. H. (2016). User profiling approaches for demographic recommender systems. *Knowledge-Based Systems*, *100*, 175–187. https://doi.org/10.1016/j.knosys.2016.03.006

Alamdari, P. M., Navimipour, N. J., Hosseinzadeh, M., Safaei, A. A., & Darwesh, A. (2020). A Systematic Study on the Recommender Systems in the E-Commerce. *IEEE Access*, *8*, 115694–115716. https://doi.org/10.1109/access.2020.3002803

Alyari, F., & Jafari Navimipour, N. (2018). Recommender systems. *Kybernetes*, *47*(5), 985–1017. https://doi.org/10.1108/k-06-2017-0196

Burke, R. (2000). *Knowledge-based recommender systems*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.21.6029&rep=rep1&type=pdf

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370. https://doi.org/10.1023/a:1021240730564

Chen, L., Yang, F., & Yang, H. (2017). *Image-based product recommendation system with convolutional neural networks*. Stanford University. http://cs231n.stanford.edu/reports/2017/pdfs/105.pdf

Chu, W.-T., & Tsai, Y.-L. (2017). A hybrid recommendation system considering visual information for predicting favorite restaurants. *World Wide Web*, *20*(6), 1313–1331. https://doi.org/10.1007/s11280-017-0437-1

Da'u, A., & Salim, N. (2019). Sentiment-Aware Deep Recommender System With Neural Attention Networks. *IEEE Access*, *7*, 45472–45484. https://doi.org/10.1109/access.2019.2907729

Deshpande, M., & Karypis, G. (2004). Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems*, *22*(1), 143–177. https://doi.org/10.1145/963770.963776

Elkahky, A. M., Song, Y., & He, X. (2015). A Multi-View Deep Learning Approach for Cross Domain User Modeling in Recommendation Systems. In *WWW '15: Proceedings of the 24th International Conference on World Wide Web* (pp. 278–288). ACM. https://doi.org/10.1145/2736277.2741667

Hazratzadeh, S., & Jafari Navimipour, N. (2016). Colleague recommender system in the Expert Cloud using features matrix. *Kybernetes*, *45*(9), 1342–1357. https://doi.org/10.1108/k-08-2015-0221

He, M., Zhang, S., & Meng, Q. (2019). Learning to Style-Aware Bayesian Personalized Ranking for Visual Recommendation. *IEEE Access*, *7*, 14198–14205. https://doi.org/10.1109/access.2019.2892984

He, R., & Mcauley, J. (2016). Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW '16: Proceedings of the 25th International Conference on World Wide Web* (pp. 507–517). ACM. https://doi.org/10.1145/2872427.2883037

He, R., & McAuley, J. (2016). VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*. AAAI.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management*. ACM. https://doi.org/10.1145/2505515.2505665

Huang, Y., Wang, N., Zhang, H., & Wang, J. (2019). A novel product recommendation model consolidating price, trust and online reviews. *Kybernetes*, *48*(6), 1355–1372. https://doi.org/10.1108/k-03-2018-0143

Zhou, H.-h., Ltu, Y.-j., Zhang, W.-q., & Xie, J.-y. (2004). A Survey of Recommender System Applied in E-commerce. *Application Research of Computers*, 1, no. 3. http://en.cnki.com.cn/Article_en/CJFDTotal-JSYJ200401003.htm

Leng, Y., Lu, Q., & Liang, C. (2016). A collaborative filtering similarity measure based on potential field. *Kybernetes*, *45*(3), 434–445. https://doi.org/10.1108/k-10-2014-0212

Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based Recommender Systems: State of the Art and Trends. In *Recommender Systems Handbook* (pp. 73–105). Springer. https://doi.org/10.1007/978-0-387-85820-3_3

Lu, Y., Gao, M., & Saga, R. (2021). *Apparel Recommender System based on Bilateral image shape features*. arXiv preprint arXiv:2105.01541. https://arxiv.org/abs/2105.01541

Maros, A., Belém, F., Silva, R., Canuto, S., Almeida, J. M., & Gonçalves, M. A. (2019). Image Aesthetics and its Effects on Product Clicks in E-Commerce Search. In *Proceedings of the SIGIR 2019 eCom workshop. CEUR-WS*. http://ceur-ws.org/Vol-2410/paper25.pdf

Mbelwa, H. (2021). *Image-based poultry disease detection using deep convolutional neural network*. Nelson Mandela African Institution of Science and Technology. https://dspace.nm-aist.ac.tz/handle/20.500.12479/1344

Jafari Navimipour, N., Rahmani, A. M., Habibizad Navin, A., & Hosseinzadeh, M. (2015). Expert Cloud: A Cloud-based framework to share the knowledge and skills of human resources. *Computers in Human Behavior*, *46*, 57–74. https://doi.org/10.1016/j.chb.2015.01.001

Ratnasingham, P. (1998). The importance of trust in electronic commerce. *Internet Research*, *8*(4), 313–321. https://doi.org/10.1108/10662249810231050

Resnick, P., & Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, *40*(3), 56–58. https://doi.org/10.1145/245108.245121

Schafer J.B., Frankowski D., Herlocker J., & Sen S. (2007). Collaborative Filtering Recommender Systems. In Brusilovsky P., Kobsa A., Nejdl W. (eds.) *The Adaptive Web* (pp. 291-324). Springer. https://doi.org/10.1007/978-3-540-72079-9_9

Shankar, D., Narumanchi, S., Ananya, H., Kompalli, P., & Chaudhury, K. (2017). *Deep learning based large scale visual recommendation and search for E-Commerce*. arXiv preprint arXiv:1703.02344. https://arxiv.org/abs/1703.02344

Sharif, S. H., Mahmazi, S., Jafari Navimipour, N., & Farid Aghdam, B. (2013). A Review on Search and Discovery Mechanisms in Social Networks. *International Journal of Information Engineering and Electronic Business*, *5*(6), 64–73. https://doi.org/10.5815/ijieeb.2013.06.08

Shoja, B. M., & Tabrizi, N. (2019). Customer Reviews Analysis With Deep Neural Networks for E-Commerce Recommender Systems. *IEEE Access*, *7*, 119121–119130. https://doi.org/10.1109/access.2019.2937518

Simonyan, K., & Zisserman, A. (2014). *Very deep convolutional networks for large-scale image recognition*. arXiv preprint arXiv:1409.1556. https://arxiv.org/abs/1409.1556

Tuinhof, H., Pirker, C., & Haltmeier, M. (2019). Image-Based Fashion Product Recommendation with Deep Learning. In *LOD 2018: Machine Learning, Optimization, and Data Science* (pp. 472-481). Springer. https://doi.org/10.1007/978-3-030-13709-0_40

Turban, E., King, D., Lee J., & Viehland, D. (2002). Electronic commerce: A managerial perspective 2002. Prentice Hall.

Vandecasteele, F., Vandenbroucke, K., Schuurman, D., & Verstockt, S. (2017). Spott: On-the-Spot e-Commerce for Television Using Deep Learning-Based Video Analysis Techniques. *ACM Transactions on Multimedia Computing, Communications, and Applications*, *13*(3s), 1–16. https://doi.org/10.1145/3092834

Vasudevan, S., Chauhan, N., Sarobin, V., & Geetha, S. (2020). Image-Based Recommendation Engine Using VGG Model. In *Advances in Communication and Computational Technology, Select Proceedings of ICACCT 2019* (pp. 257–265). Springer. https://doi.org/10.1007/978-981-15-5341-7_21

Wang, S., Wang, Y., Tang, J., Shu, K., Ranganath, S., & Liu, H. (2017). What Your Images Reveal: Exploiting Visual Contents for Point-of-Interest Recommendation. In *WWW '17: Proceedings of the 26th International Conference on World Wide Web* (pp. 391–400). ACM. https://doi.org/10.1145/3038912.3052638

Wigand, R. T. (1997). Electronic Commerce: Definition, Theory, and Context. *The Information Society*, *13*(1), 1–16. https://doi.org/10.1080/019722497129241

Yu, W., Zhang, H., He, X., Chen, X., Xiong, L., & Qin, Z. (2018). Aesthetic-based Clothing Recommendation. In *WWW '18: Proceedings of the 2018 World Wide Web Conference* (pp. 649–658). ACM. https://doi.org/10.1145/3178876.3186146

Editorial record: The article has been peer-reviewed. First submission received on 5 October 2021. Revisions received on 17 November 2021, and 2 December 2021. Accepted for publication on 2 December 2021. The editor in charge of coordinating the peer-review of this manuscript and approving it for publication was Zdenek Smutny.

Acta Informatica Pragensia is published by Prague University of Economics and Business, Czech Republic.

ISSN: 1805-4951