IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 20 March 2023, accepted 17 April 2023, date of publication 25 April 2023, date of current version 3 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3270260

RESEARCH ARTICLE

To Cluster or Not to Cluster: The Impact of Clustering on the Performance of Aspect-Based Collaborative Filtering

SUMAIA MOHAMMED AL-GHURIBI[1,2], SHAHRUL AZMAN MOHD NOAH[2], (Member, IEEE), MAWAL A. MOHAMMED[3], SULTAN NOMAN QASEM[1,4], AND BELAL ABDULLAH HEZAM MURSHED[5,6]

[1]Department of Computer Science, Faculty of Applied Sciences, Taiz University, Taiz, Yemen
[2]Center for Artificial Intelligent Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi 43600, Malaysia
[3]Software Engineering Department, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
[4]Department of Computer Science, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University, Riyadh 11432, Saudi Arabia
[5]Department of Computer Science, College of Engineering and IT, Amran University, Amran 9677, Yemen
[6]Department of Studies in Computer Science, Mysore University, Mysore, Karnataka 570006, India

Corresponding author: Sumaia Mohammed Al-Ghuribi (somaiya.ghoraibi@gmail.com)

ABSTRACT Collaborative filtering (CF) is one of the most widely utilised approaches in recommendation techniques. It suggests items to users based on the ratings of other users who share their preferences. Thus, one of the aims of CF is to find reliable neighbours. Typically, CF produces a sparse user-item rating matrix, when relying only on the ratings to identify the precise neighbours, resulting in poor performance. User reviews can be essential in overcoming those situations because of the diverse elements available in reviews. The most popular element is aspects, which can provide a fine-grained analysis of users' behaviours, thus improving personalised recommendations. However, increasing the number of aspects also results in sparsity, therefore may deteriorate the recommendation performance. As a result, clustering of aspects may lessen this sparsity, but it is yet unclear how much this would affect the performance of CF systems. This study proposes a CF approach based on aspect clustering that addresses the above issue in terms of rating prediction. The approach aims to reduce the sparseness in the multi-criteria rating matrix by grouping aspects into clusters based on their semantic similarity, which will be less expensive and require less memory to discover the neighbourhood set. Our approach extracts aspects and represents them using Google's pre-trained Word2vec model. Then, aspects are organised into clusters using the K-means clustering algorithm. Multi-dimensional Euclidean distance is used as a similarity measure for finding the appropriate neighbours and predicted ratings of unseen items are then made using the *k*NN algorithm. This study also identifies the number of aspects that significantly impacts CF performance. Experiments are carried out using a real large-scale dataset: the Amazon movie dataset. Evaluation is also performed by comparing CF performance of the proposed approach with three different baseline approaches. Results show that the proposed approach improves CF performance compared to other approaches in terms of three predictive accuracy metrics.

INDEX TERMS Collaborative filtering, user reviews, aspects, Word2vec, K-means clustering, Euclidean distance.

I. INTRODUCTION

Recommender systems (RSs) are content-filtering programs that try to alleviate information overload by offering personalised items to users based on their tastes or item information [1]. There are three main recommendation approaches: collaborative-based, content-based, and hybrid. RS's process has two phases: rating prediction phase, and item recommendation phase. This study focuses on the rating prediction phase of the collaborative filtering (CF)

The associate editor coordinating the review of this manuscript and approving it for publication was Ali Shariq Imran.

approach, which is the most commonly utilised approach in RSs [2]. CF produces a recommendation for a target user based on the similarities between the target user and other users who have previously expressed similar preferences. The primary premise behind the CF method is that users who have previously exhibited similar behaviours are more likely to have similar preferences in the future. Although CF techniques have yielded impressive outcomes, some main challenges are involved, such as data sparseness due to insufficient user ratings, reducing the effectiveness of RSs.

The number of e-commerce and social websites encouraging users to discuss their experiences has recently increased significantly. As a result, online comments (i.e., reviews) on various issues continue to increase exponentially [3], [4]. These reviews can be incorporated into the recommendation process as a new source of information because of the sizeable and rich information related to users or items they include (i.e., the review elements). Reviews can be utilised to solve the data sparsity and hence improve the performance of RSs by providing fine-grained analyses of users' behaviours, which can help improve personalised recommendations. For this purpose, different elements can be derived from reviews, such as aspects, review helpfulness, and contextual information [1], [5]. Aspects, also referred to as opinion targets, are concepts in which the opinions are presented in a specific text/review. Aspects of restaurants, for instance, might be broken down into categories like ambience, meal quality, service, and pricing range.

Most of the studies that exploits aspects in CF demonstrates their benefit in improving CF performance, particularly in the rating prediction task [6], [7], [8], [9]. Extracting aspects from reviews is a challenging process due to the distinct characteristics of the reviews. As a result, most current studies [10], [11], [12] use fixed aspects defined manually or available in some datasets such as Yahoo! Movies. On the other hand, learned aspects extracted from reviews using specific methods are preferred over fixed ones [1], [13]. Learned aspects can help identify user preferences better, because the user will only include the aspects that are important to him. Furthermore, learned aspects can be utilised to describe items more accurately, which facilitate personalised recommendations based on user preferences.

Conventional CF relies on a user rating matrix to find the reliable neighbours of target users. Each cell in the rating matrix represents an overall rating of an item for a user. Based on the rating matrix values, neighbours with the highest similarity values to the target user are selected using any similarity metric method such as the Cosine function or the Pearson correlation function [14]. In contrast to aspect-based CF, each aspect is a criterion that must be considered throughout the neighbour selection process in addition to the overall rating. For this purpose, the similarity computation of conventional CF must be extended to reflect multi-criteria CF. There are two possible ways of multi-criteria CF [15], [16], [17]. The first is to calculate the similarity value concerning each criterion separately and then aggregate these values into a single similarity value. The second is to calculate the distances with respect to the multiple criteria directly in the multi-dimensional space. In this case, neighbours are users with similar ratings for the same criteria (i.e., aspects) as the target user. The cost of finding the nearest neighbours rises as the number of users/items and aspects increases. Clustering of aspects can reduce such a cost as the matrix dimensions will be reduced [10], [18]. However, there have been some concerns that it will reduce the performance of CF systems. Yet, to what extent it will impact the performance of CF systems is subject to further research.

This study, therefore, proposes a CF approach based on aspect clustering involving large-scale datasets. The approach aims to minimise the cost of discovering neighbours for a target user through a clustering strategy for aspects that reduces the sparseness in the multi-criteria rating matrix while still providing a good CF performance [19]. Throughout this paper, we refer to such an approach as aspect-clustering-based CF. The selection of relevant neighbours is a critical factor for improving CF performance. To a certain extent, the predictive accuracy of CF approaches depends on the choice of neighbours [10], [20], [21]. This study uses the semantically enhanced aspect extraction (SEAE) approach proposed in an earlier work [22] to extract aspects. The aspects are then represented using Google's pre-trained Word2vec model and grouped into clusters based on their semantic similarity using the K-means clustering algorithm. In addition, user similarity is calculated based on the cluster value (i.e., each cluster has a different number of aspects) rather than an individual aspect. This study explores the impact of clustering aspects on the performance of CF systems in terms of their prediction accuracy. We also examine whether all aspects extracted from reviews have the same impact on CF performance or whether some have a more significant effect than others. The contributions of this study are summarised as follows:

- We propose an enhanced aspect-based CF approach by clustering the learned aspects to minimise the cost of the CF rating prediction process without compromising its accuracy involving large-scale datasets.
- We demonstrate that clustering the learnt aspects makes the selection of neighbours more relevant, as evidenced by enhanced prediction accuracy, since the prediction accuracy of CF techniques depends on the choice of neighbours.
- We carried out several experiments and identified the aspects that impact the performance of aspect-based CF.

The remainder of this paper is organised as follows. Section II presents a quick overview of the current state of the art in our field. Section III presents our proposed approach. Section IV describes the experimental results, and Section V discusses the evaluation and analyses the proposed and baseline approaches results. Finally, Section VI summarises this study and gives research directions for the future.

II. STATE OF THE ART

Presently, RSs are widely regarded as one of the most significant tools in the digital world. Over the last two decades, RSs have been successfully employed as information filtering methods to address information overload problems [1], [23]. They have been integrated into various platforms, including e-commerce and social networking sites [24]. RSs usually concentrate on two issues: predicting ratings and making recommendations [25], [26]. The present study focuses on predicting ratings for the CF approach, which is currently the most popular approach used in RSs to identify neighbours [27], [28], [29].

A. CONVENTIONAL COLLABORATIVE FILTERING

In the traditional CF recommendation algorithm, a user–rating matrix **R**: **U**×**I**, in which a set of items **I** are rated by a set of users **U,** is created for making recommendations for a target user. In general, the matrix **R** matrix has numerous missing ratings (i.e., unrated items), and the more missing ratings there are, the sparser the matrix becomes, which results in poor CF performance. Conventional CF aims to estimate the values of these unrated items to recommend the relevant ones to the target user. The quality of the predicted values impacts the quality of the recommendations. The rating predictions for these ratings can be calculated using various prediction algorithms, including nearest-neighbour algorithms and model-based approaches. The k-nearest neighbour (kNN) algorithms have emerged as one of the most popular algorithms for CF [30], [31].

kNN's entire success can be attributed to its automation of the process of acquiring and integrating content that reflects human decisions [31]. By doing so, it can compute inherently meaningful recommendations rather than just the compatibility of two items' specifications [31]. The kNN algorithm is used for regression and classification, and it was introduced by Joseph Hodges and Evelyn Fix in 1951 [32]. It is then developed as a non-parametric, lazy learning method by Thomas Cover. The kNN algorithm assumes that similar things converge. It is determined by the similarity of item features (also known as closeness, distance, or proximity) and does not make any assumption regarding the data distribution [33]. Because the points are in the features' space, the algorithm presumes that the data is in feature space; this enables the concept of distance. It also implies that each training dataset consists of a collection of vectors, each of which is connected to a class label. The number of neighbours that affect the classification is determined by a single given value of k.

A predicted rating $p(U_1, i)$ for user $U_1$and item i is obtained by first using kNN to select the k best correlated (or most similar) users for the target user U_1. The Pearson correlation and cosine-based functions are the most commonly utilised measures for determining user–item similarity [34]. Below is a detailed description of the two functions.

1) PEARSON CORRELATION COEFFICIENT

The Pearson correlation coefficient determines the strength of a relationship between two user-item pairings by using the following equation [35]:

$$Sim(U_1, U_2) = \frac{\sum_{i=1}^{n} \left(r_{U_1,i} - \overline{r_{U_1}}\right)\left(r_{U_2,i} - \overline{r_{U_2}}\right)}{\sqrt{\sum_{i=1}^{n} \left(r_{U_1,i} - \overline{r_{U_1}}\right)^2}\sqrt{\sum_{i=1}^{n} \left(r_{U_2,i} - \overline{r_{U_2}}\right)^2}} \tag{1}$$

where $Sim(U_1, U_2)$ represents the similarity of two users U_1 and U_2; $r_{U_1,i}$ is user U_1's rating for item i; $\overline{r_{U_1}}$ is user U_1's average rating, and n is the overall number of user-item pairs.

2) COSINE SIMILARITY

Cosine similarity is a vector-space model based on linear algebra rather than just a statistical approach, which differs from the Pearson-based measure. It computes the cosine angle between two vectors in a multi-dimensional space. The lower the angle and the greater the similarity between the vectors, the closer the cosine value to 1. The cosine similarity between two users U_1 and U_2 is described as follows:

$$Sim(\overrightarrow{U_1}, \overrightarrow{U_2}) = \frac{\overrightarrow{U_1}.\overrightarrow{U_2}}{\left|\overrightarrow{U_1}\right| * \left|\overrightarrow{U_2}\right|} = \frac{\sum_{i=1}^{n} r_{U_1,i} * r_{U_2,i}}{\sqrt{\sum_{i=1}^{n} r_{U_1,i}^2}\sqrt{\sum_{i=1}^{n} r_{U_2,i}^2}} \tag{2}$$

After the k most similar users to user U_1 (i.e., neighbours) are selected, the rating prediction of user U_1 for item i is made using the following function:

$$p(U_1, i) = \frac{\sum_{U \in N(U_1)} Sim(U_1, U) \cdot r_{U,i}}{\sum_{U \in N(U_1)} Sim(U_1, U)} \tag{3}$$

where $p(U_1, i)$ is the prediction of user U_1's rating of item i, $r_{U,i}$ refers to user Us rating of item i, $Sim(U_1, U)$ represents the similarity score of two users U_1 and U, and $N(U_1)$ is the user U_1's neighbour set.

B. ASPECT-BASED COLLABORATIVE FILTERING

Several studies have taken advantage of textual user reviews to improve CF performance. Several review elements can be extracted and incorporated into CF to enhance its performance [1], [5]. The present study focuses on the aspects element which has the most significant impact on CF performance. An aspect is a concept that depicts a topic and needs to be included in each item, such as the aspects 'story', 'actor', and 'director', which are well-known concepts for movies. Ironically, despite the benefits of the aspects element for improving CF performance, only a few researchers have looked at their potential as a tool for enhancing CF performance and resolving CF issues [36], [37]. Most of these researchers rely on defining a fixed number of aspects due to difficulties in extracting them from the reviews. On the other hand, some researchers employ a text analysis technique called aspect-based sentiment analysis (ABSA). ABSA aims to extract the aspects and identify the sentiment associated with each aspect. Two main tasks are involved in

it: aspect extraction and aspect sentiment analysis, and these tasks can be categorised into three types: semi-supervised, supervised, and unsupervised [38]. As the size of the reviews grows, finding labelled data for reviews, which is required for supervised approaches, becomes more difficult. As a result, most research focuses on unsupervised approaches, which do not require laborious and time-consuming data annotation tasks, nor do they suffer from domain adaption issues [39], [40]. The unsupervised approaches are based on vocabulary, frequency, syntactic relations, and topic models [22].

Rather than relying merely on the overall rating for making recommendations, aspects can enhance the recommendations by providing a fine-grained analysis of the users' interests. The efficiency of this element in improving CF performance has been demonstrated by the research that extracts this element from user reviews and incorporates it into CF [7], [9], [27], [41]. Musto et al. [6] extracted aspects and sub-aspects using Kullback-Leibler divergence, a non-symmetric measure and Nielsen's lexicon [42] based on the AFINN wordlist to assign the sentiment score for each extracted aspect/sub-aspect. The extracted aspects were then integrated into a multi-criteria user-item CF algorithm, in which the multi-dimensional Euclidean distance [15] is used to calculate the similarity between two user-item pairs. The experimental results for different datasets— Amazon, TripAdvisor, and Yelp—proved that the proposed algorithm outperformed all algorithms based on matrix factorisation and the single-criterion recommendation algorithms in term of the mean average error metric. Bauman et al. [43] developed a method for recommending items to improve the user's experience with the recommended items by using their most valuable aspects. The valuable aspects for users are identified using the Sentiment Utility Logistic model, for which an opinion parser: Double Propagation, is used for extracting the aspect-sentiment pairs. The experiments used the Yelp dataset, and the results for this method outperformed the most positive/negative aspect approaches and the most popular aspect approach.

As artificial neural networks have advanced, researchers have looked into employing deep learning approaches to extract aspects and improve RSs [44], [45], [46], [47]. For example, Da'u et al. [45] presented a model that employs a deep learning technique for extracting aspect–sentiment pairs to improve the accuracy of the recommendations. It comprises two parts: extracting the aspect–sentiment pairs and predicting ratings. The experimental findings demonstrated the proposed model's usefulness in enhancing recommendation accuracy.

Some studies, such as the work of Wasid and Ali [10], utilise user-based clustering to group users into clusters that generate neighbourhood sets. The K-means algorithm is used to cluster users, and a Mahalanobis distance metric is used to compute similarities between users. Experiments show that the performance of CF with user clustering outperforms CF without clustering using the Yahoo! Movies dataset. Zhang et al. [48] presented a user-based clustering approach to reduce the impact of data sparsity. In their algorithm, users with similar preferences are grouped into a similar cluster, and only neighbours from the same cluster as the target user are chosen. The efficiency of the proposed algorithm in enhancing CF performance was demonstrated by experimental findings utilising both the MovieLens and HetRec2011-MovieLens datasets. Xiaojun [49] presented an algorithm for clustering users using K-means clustering and developed an improved similarity metric to discover relevant neighbours to the target user. Then a list of recommendations was produced. The experimental findings demonstrated the effectiveness of the suggested algorithm in solving the data sparsity problem, and the recommendations were able to adapt to changes in the user's interests.

Although studies incorporating aspects into the CF recommendation process are in existent, little to no studies have addressed aspect-based CF utilising clustering. Instead, we hypothesised that clustering of aspects in multi-criteria CF would result in the selection of more similar neighbours and thus improve the CF performance in terms of rating prediction.

The objective of the current study is to propose a practical CF approach based on aspect clustering in which learnt aspects are grouped into clusters according to their semantic similarity using the K-means clustering algorithm. We extract aspects using the semantically enhanced aspect extraction (SEAE) approach described in [22]. The extracted aspects mentioned in the user review are assigned scores using a domain-specific lexicon discussed in the previous work [50]. Because the review text is often brief and only contains a limited number of aspects, only a few extracted aspects will obtain scores for a given review. The rest will be left without scores, creating a highly sparse multi-criteria rating matrix. This issue is the reason for developing the proposed approach using clustering in this study. The approach aims to reduce the sparseness in the multi-criteria rating matrix by grouping aspects into clusters based on their semantic similarity. The size of the multi-criteria rating matrix will thus depend on the number of aspect clusters rather than the total number of aspects. As a result, it will be less expensive and require less memory to discover the neighbourhood set. Furthermore, the proposed approach is designed to increase the ability to locate dependable and accurate neighbours for target users, which will improve CF performance. In addition, this study examines whether all the learnt aspects have the same impact on CF performance or whether some have a more significant impact than others. As such, other scenarios which do not have explicit feedback in the form of user reviews are beyond the scope of this work. Finally, CF performance using the proposed approach is compared with different approaches to evaluate its efficiency.

III. METHODOLOGY

This study aims to investigate the effectiveness of clustering aspects for improving the performance of aspect-based CF

in terms of rating prediction. This study's methodology is based on an experimental approach using the Amazon movie dataset[1] [51], a large real-world dataset. A two-phase methodology has been proposed to achieve the aim: dataset preparation with aspects clustering and developing an aspect-clustering-based CF approach. The former phase aims to populate the dataset with aspect clusters. Thus, it includes two steps: extracting and clustering aspects and extending the used dataset by including new attributes that describe the aspect clusters. The latter phase aims to develop an aspect-clustering-based CF approach by following a two-step procedure for selecting the appropriate parameters to gain better CF performance. The steps are determining the optimum number of clusters and identifying the significant aspects. **FIGURE 1** describes the general methodology of this study. The following sections describe each phase in detail.

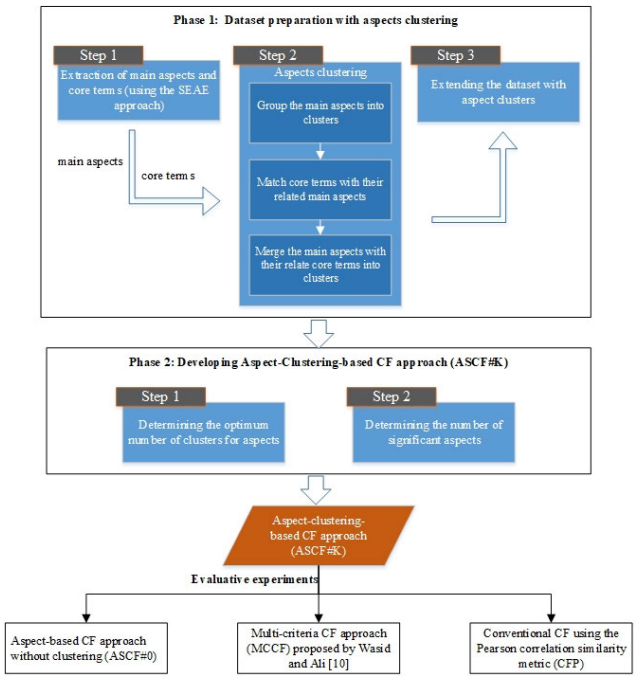

FIGURE 1. The two-phase methodology.

A. PHASE 1: DATASET PREPARATION WITH ASPECTS CLUSTERING

This study uses a real large-scale dataset: the Amazon movie dataset. In order to use this dataset in our study, this phase extracts aspects from the dataset, and groups them into clusters. The aspect clusters are then added to the dataset as new attributes so that they can be used in the following stage of developing the aspect-clustering-based CF approach. This phase consists of two steps: extracting and clustering aspects, and extending the Amazon movie dataset.

[1] http://jmcauley.ucsd.edu/data/amazon/links.html

1) EXTRACTING AND CLUSTERING ASPECTS

This step extracts the aspects from the Amazon movie dataset using the Semantically Enhanced Aspect Extraction (SEAE) approach described in [22] and the blocking technique mentioned in [52]. The SEAE is a hybrid approach consisting of three approaches: syntactic-relation-based and frequency-based, that work in parallel—followed by a semantic similarity-based approach that aims to filter the aspects and extract only the ones relevant to the target domain. The SEAE approach generates a list of main aspects and a list of core terms. The core terms are words closely related to the main aspects but do not appear in the list of the main aspects due to their low frequency compared to the main aspects. In particular, limiting the aspects to specified words and ignoring those with similar meanings can have a negative impact on the aspect extraction process. In the movie domain, for example, the word *picture* appears in the list of main aspects, but the words *image*, *photo, photograph* and *snapshot* do not appear in the list, although having a similar meaning to the word *picture*. Ignoring those words might negatively impact the extraction process; therefore, these words are identified as the core terms for the aspect *picture* using the SEAE approach. To summarise, the extraction process results using the SEAE approach are the main aspects and the core terms relating to the main aspects.

After the aspects are extracted, this step arranges them into clusters based on their semantic similarity and compiles the core terms with their associated main aspects. Specifically, this step partitions aspects into K clusters based on their semantic similarity using Google's pre-trained Word2vec model and the K-means clustering method. K-means is a widely used distance/centroid-based algorithm, where distances are determined in order to allocate a point to a cluster [53]. The K-means algorithm associates each cluster with a centroid and aims to minimise the sum of the distances between the cluster centroid and the points assigned to the cluster. The distance between aspects (i.e., points) is calculated based on cosine similarities that are represented using a word embedding algorithm: Google's pre-trained Word2vec model that can accurately determine the similarity values among words. We chose Google's pre-trained Word2vec model because of its large vocabulary, which is trained using the Google News dataset on about 100 billion words and consists of three million words and phrases [54]. The clustering aspects process comprises three sub-steps: group the main aspects into clusters, match the core terms with their related main aspects and merge the main aspects with their core terms into the identified clusters. The three sub-steps are described in the following sub-sections.

a: GROUP THE MAIN ASPECTS INTO CLUSTERS

In this step, the main aspects are extracted from the Amazon movie dataset using the SEAE approach are divided into clusters based on their semantic similarity. This step is accomplished by a simple function shown in **FIGURE 2**,

```
Function Group_MainAspect is
  Input: Main_Aspects (49 words)
  Output: AspectClusters_list that contains the aspects in each
      determined cluster based on the number of clusters.

  Aspect_array ← Read the main aspects from the input list and append
      them to Aspect_array list
   # run the pre-trained Google Word2vec model because word similarity will be
     calculated based on it.
  wvmodel←gensim.models.KeyedVectors.load_word2vec_format
      ("\GoogleNews-vectors-negative300.bin",binary←True)
  Words ← wvmodel [ Aspect_array ]  # Add the words to wvmodel to
      measure the similarity based on it.
  NUM_CLUSTERS ←8          # Determine the number of clusters; four
                    numbers are tested [8, 10, 13, 15]
   # Run the KMeansClusterer with the specified number of clusters and define the
                    distance measure (cosine similarity)
  kclusterer←KMeansClusterer(NUM_CLUSTERS,
      distance=nltk.cluster.util.cosine_distance, repeats←300)
   # assign cluster to each word ' Clustering process'
  Clusters←kclusterer.cluster (Words, assign_clusters ← True)
  AspectClusters_list ← save the words with their cluster in a list for
     later use
```

FIGURE 2. A function to partition the main aspects into clusters.

TABLE 1. Examples of how the main aspect for several core terms is determined.

Core Term	Similarity of main aspects with the core terms [(main_aspect$_1$, *similarity_value$_1$), ….. , (main_aspect$_n$, *similarity_value$_n$)]
actioner	[('**movie**', 0.59175), ('film', 0.51723)]
baby	[('**child**', 0.65740), ('girl', 0.52895)]
divorcee	[('**woman**', 0.51156), ('girl', 0.50160)]
instrumentals	[('music', 0.50714), ('**song**', 0.58670)]
lady	[('**woman**', 0.62888), ('girl', 0.50032)]
narrative	[('**story**', 0.54204), ('script', 0.51466)]
novella	[('**book**', 0.58202), ('script', 0.50964)]
poem	[('book', 0.52152), ('**song**', 0.59482)]
portrayer	[('**character**', 0.54754), ('actor', 0.51588)]
screenwriter	[('movie', 0.53432), ('film', 0.56543), ('actor', 0.58402), ('**script**', 0.65176)]
victim	[('man', 0.52303), ('**woman**', 0.54501)]

which takes the main aspects as input and generates aspect clusters, which are then stored in *AspectClusters_list*. For clarification, the main aspects are clustered using the K-means clustering algorithm. Cosine similarity is used for measuring the similarity between aspects. This step specifies four different values for the number of clusters (K), with K = 8, 10, 13, and 15. These K numbers were chosen with the help of the elbow method, a heuristic one that determines the best number of clusters. It is based on the idea that a number of clusters should be chosen such that adding another cluster does not result in significantly better data modelling [55].

b: MATCH CORE TERMS WITH THEIR RELATED MAIN ASPECTS

Core words are derived using the SEAE approach; most of these terms are semantically related to the main aspects. Due to the small number of occurrences of these terms relative to the main aspects, they do not appear in the list of the main aspects. However, ignoring such terms would have a detrimental effect on the SEAE performance and subsequently affect CF performance. Several core terms are extracted from the Amazon movie dataset using the SEAE approach. This step seeks to match these terms with the most relevant main aspects based on their semantic similarity.

Because this study focuses on the movie domain, all the extracted main aspects are relevant to the movie domain. As a result, several extracted main aspects are semantically similar, implying that some core terms are likely to be related to multiple main aspects. The decision to link a core term to a specific primary aspect is based on the aspect having the highest similarity value for that term.

Table 1 shows an example of some core terms and a list of the main aspects with the similarity values to the core terms. The aspects with the highest similarities to the core terms are selected. The words are represented using Google's pre-trained Word2vec model, and the semantic similarity between the core term and the main aspect is calculated using the cosine similarity. For example, the core term *baby* is most like the aspect *child*. Thus, the aspect *child* will be selected.

c: MERGE THE MAIN ASPECTS WITH THEIR CORE TERMS INTO CLUSTERS

This step combines the results of the previous two steps to produce a list of aspects that have been grouped into the clusters determined in step a (i.e., the main aspects grouped into clusters) together with the core terms identified in step b. This step is conducted using the function defined in **FIGURE 3**. The function is applied using all four specified values of K (number of clusters). The function produces a list named *AspectCoreTerms_list* for each value of K. **FIGURE 4** displays a scheme showing the results of this step.

2) EXTENDING THE DATASET WITH THE ASPECT CLUSTERS

The Amazon movie dataset comprises nine attributes: (['reviewerID', 'ProductID', 'reviewerName', 'helpful', 'reviewText', 'summary', 'overall', 'unixReviewTime', 'reviewTime']). The dataset is extended by adding two additional attributes: 'Aspects' and 'Brilliant aspect', whose values are assigned using the SEAE approach. These two attributes are defined as follows:

a: ASPECTS

This attribute contains a list of all the aspects mentioned in a user review. Each cell in the list depicts the following description of an aspect: {(AspectName$_1$, AspectWeight$_1$, AspectRating$_1$, NumberOfOccurrences$_1$), [(AspectName$_2$, AspectWeight$_2$, AspectRating$_2$, NumberOfOccurrences$_2$), …[(AspectName$_n$, AspectWeight$_n$, AspectRating$_n$, NumberOfOccurrences$_n$)}. For example, the following description of aspects {('music', 0.3163, 0.083, 2), ('story', 0.6132, 0.0499, 4), (cast, 0.3279. -0.0014, 1)} is taken from one review contained in the dataset. For more details:

- AspectWeight depicts an aspect's weight, which reflects the importance of it. It is determined using

```
Function GatherCoreTerms_MainAspect is
  Input: Main aspects grouped in clusters, main aspects with related core terms.
  Output: AspectCoreTerms_list, which contains the aspects in each determined cluster with their core terms.

  DF_Aspect ← Read the grouped main aspects from the input list into a dataframe
  DF_AspectCoreTerms ← Read the main aspects with related core terms from the input list into a dataframe
  GroupedAspect_CoreTerms ← []
  For each Groupno in range (DF_Aspect .shape[0]) do # loop based on number of clusters
    aspect_Group ← append all the aspects in Groupno into a list
    Aspect_CoreTerms←[]
    For each aspect in range (len (aspect_group) ) do # loop based on number of aspects in each cluster
        For each core_term in range ( DF_AspectCoreTerms.shape[1] ) do
          Core_Aspect ← []
          Core_Aspect.append (aspect_Group[aspect] )    # main aspect
                IF ( aspect_Group[aspect] = DF_AspectCoreTerms .columns [core_term] )
                  For each l in range ( DF_AspectCoreTerms [aspect_Group [aspect] ].shape [0] ) do
                    Core_Aspect. append ( DF_AspectCoreTerms [aspect_Group[aspect]][l] )
                  Break
        Aspect_CoreTerms.append ( Core_Aspect )
        GroupedAspect_CoreTerms.append (( df['Group'][Groupno], Aspect_CoreTerms))
  AspectCoreTerms_list ← append GroupedAspect_CoreTerms list into a list
```

FIGURE 3. A function for merging aspects with their core terms.

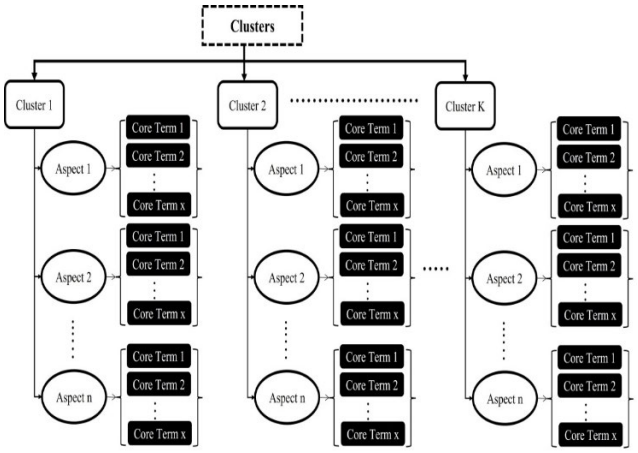

FIGURE 4. The result of merging the main aspects with their core terms into clusters.

the Modified Term Frequency–Inverse Document Frequency (TF-IDF) proposed by Zhu et al. [56].

- AspectRating is a score given to the aspect based on the sentiment words belong to the aspect. The scores of these sentiment words are determined using the domain-specific lexicon proposed in [50].
- NumberOfOccurrences shows the number of times the aspects mentioned in a review.

b: BRILLIANT ASPECT

This attribute identifies an aspect a user focused on more than the other mentioned aspects within the user's written review. The Brilliant aspect has the highest rating among all the aspects mentioned in the review. Referring to the previous example, the aspect 'music' will be selected as the Brilliant aspect since it is the highest ratings among the three aspects.

After the dataset has been extended, this step partitions the aspects in the Aspects attribute into the clusters defined in the previous step. Thus, the new attribute Aspects is defined as follows (4), shown at the bottom of the page, where K is the number of clusters, n is the number of aspects, and m is the number of core terms. $A_{i,j}$ is the name of j main aspect in cluster i, and $W_{A_{i,j}}$, $R_{A_{i,j}}$, and $O_{A_{i,j}}$ are the weight, rating and occurrences of aspect $A_{i,j}$ respectively. $C^s_{A_{i,j}}$ is the core term s related to the main aspect $A_{i,j}$. $W_{C^s_{A_{i,j}}}$, $R_{C^s_{A_{i,j}}}$, and $O_{C^s_{A_{i,j}}}$ are the weight, rating and occurrences of the core term $C^s_{A_{i,j}}$ respectively. The $Feature_i$ refer to the average rating of all aspects and core terms in cluster i.

This step is performed by a function named DataSet_GroupingAspects, presented in the next section, which takes two inputs: the extended Amazon movie dataset and the *AspectCoreTerms_list* generated in the previous step, which contains the aspects in each identified cluster with their core terms. This function returns a new dataset called *Modified Dataset*, which includes the extended dataset and the additional attributes specified for aspects.

The function generates a dataset for each of the four K values. The differences among the generated datasets are the number of the added attributes and the aspects that belong to each cluster which identify by the value of K. In detail, the procedure of this function is the same for each K value. It focuses on partitioning the aspects list in the *Aspects* attribute into the previously determined clusters. The aspects with their core terms corresponding to each cluster are stored in a newly added attribute. The $Feature_i$ attribute is then updated with the average rating for all aspects listed in each cluster.

B. PHASE 2: DEVELOPING AN ASPECT-CLUSTERING-BASED COLLABORATIVE FILTERING APPROACH

This phase aims to develop an aspect-clustering-based CF approach to evaluate how well the clustering improves CF performance. Before determining the appropriate approach for this study, we must check two parameters: the optimal number of aspect clusters and the optimal number of the used aspects. Follows is the description of identifying these two parameters.

1) DETERMINING THE OPTIMUM NUMBER OF CLUSTERS FOR ASPECTS

The previous phase produces four Amazon datasets, each containing different aspect clusters based on the proposed K value. This step uses the kNNwithMeans rating prediction algorithm described in Eq (5) to determine the optimum number for aspect clustering that will improve the accuracy

$$\text{Aspects} = \left\{ \begin{array}{l} \left(A_{i,j}, W_{A_{i,j}}, R_{A_{i,j}}, O_{A_{i,j}}, \left(C^s_{A_{i,j}}, W_{C^s_{A_{i,j}}}, R_{C^s_{A_{i,j}}}, O_{C^s_{A_{i,j}}} \mid s = 1, ..m\right), Feature_i\right) \\ (i = 1..K, j = 1..n) \end{array} \right\} \tag{4}$$

```
Function DataSet_GroupingAspects is
Input: AspectCoreTerms_list, which contains the aspects in each identified cluster with their core terms,
       Extended Amazon movie dataset
Output: Modified Dataset, in which aspects are organised in clusters for each review text.
DF_GroupedAspect ← Read AspectCoreTerms_list into a dataframe
DF_Data ← Read the extended Amazon dataset into a dataframe
No_clusters ← 8 #  [8], [10], [13], [15] each time a number is selected
Review_Details ← []
For each i in range (DF_Data.shape[0]) do
    Reviewaspects ← []
    brilliantaspect_Group ← [] # Define the brilliant aspect group
    Reviewaspects ← DF_Data ['Aspects'] [i] # contains all of the aspects mention in the review
    Detailed_ Reviewaspects ← [] # Array that contains the review's aspects with the defining group and position of each aspect
    For each l in range (len (Reviewaspects)) do
      wrd ← Reviewaspects [l][0] # position 0 contains the aspect name
      wrd2 ← DF_Data [' brilliant '][i][0]
      For each k in range ( DF_GroupedAspect.shape[0] ) do
        For each m in range (len(DF_GroupedAspect['Group'] [k])) do
          For each n in range (len(DF_GroupedAspect['Group'] [k] [m])) do
            clusterword ← DF_GroupedAspect['Group'] [k][m][n]
            IF (wrd = clusterword)
              Detailed_ Reviewaspects.append((wrd,Reviewaspects[l][1], Reviewaspects [l][2], k,m, Reviewaspects [l][3])
            IF (wrd2 = clusterword)
              brilliantaspect_Group.append((wrd,Reviewaspects[l][1], Reviewaspects [l][2], k,m, Reviewaspects [l][3])
            Break
    For each nu in range (No_clusters) do
      Aspect_Group (nu) ← [], tot_Score (nu) ← 0, tot_Occurance (nu) ← 0, tot_no (nu) ← 0
    For each ind in range (len(Detailed_ Reviewaspects )) do
      IF (User_TotalAspect[ind][4]! = -1) # The aspect has not already been checked
        tmp ← []
        tot_Score=0, tot_Occurance=0, tot_no=0
        cluster ← Detailed_ Reviewaspects [ind][3]
        pos ← Detailed_ Reviewaspects [ind][4]
        tmp.append (Detailed_ Reviewaspects [ind])
        tot_Score + ← Detailed_ Reviewaspects [ind][2]
        tot_Occurance + ← Detailed_ Reviewaspects [ind][5]
        tot_no + ←1
        For each ind2 in range (ind+1, len (Detailed_ Reviewaspects )) do
          # check whether there is any other aspect in the review that has a similar cluster and position
          IF ((Detailed_ Reviewaspects [ind2][4] = pos) and (Detailed_ Reviewaspects [ind2][3] = cluster))
            tmp.append (Detailed_ Reviewaspects [ind2])
            tot_Score + ← Detailed_ Reviewaspects [ind2][2]
            tot_Occurance + ← Detailed_ Reviewaspects [ind2][5]
            tot_no + ← 1
             # Change value of the appended cell it will not be checked again
            Detailed_ Reviewaspects [ind2] ← list (Detailed_ Reviewaspects [ind2])
            Detailed_ Reviewaspects [ind2][4] ← −1
            Detailed_ Reviewaspects [ind2] ← tuple (Detailed_ Reviewaspects [ind2])
        Cno ← cluster
        Aspect_Group (Cno). append (tmp)
        tot_Score (Cno) + ← tot_Score, tot_Occurance (Cno) + ← tot_Occurance, tot_no (Cno) + ← tot_no
        IF (len (Aspect_Group (Cno)) > 0)
          Aspect_Group (Cno). append (('TOTALS_O_N',tot_Score (Cno),tot_Occurance (Cno),tot_no(Cno)))
    x1 ← DF_Data.reviewerID[i], x2 ← DF_Data.ProductID[i], x3 ← DF_Data.reviewerName[i]
    x4 ← DF_Data. helpful[i], x5 ← DF_Data.reviewText[i], x6 ← DF_Data.summary[i],
    x7 ← DF_Data. overall[i], x8 ← DF_Data.unixReviewTime[i], x9 ← DF_Data.reviewTime[i],
    x10 ← DF_Data. Aspects[i], x11 ← DF_Data.brilliant_Aspect[i]
    Review_Details.append ((x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, Aspect_Group0, Aspect_Group1, Aspect_Group2, Aspect_Group3,
      ….,Aspect_Group (No_clusters), brilliantaspect_Group, Feature0, Feature1, Feature2, …., Feature (No_clusters )))
Modified dataset ← save Review_Details into the modified list
```

of the CF rating prediction process.

$$p\left(U_1, i\right) = \overline{r_{U_1}} + \frac{\sum_{U \in N(U_1)} Sim\left(U_1, U\right) . \left(r_{U,i} - \overline{r_U}\right)}{\sum_{U \in N(U_1)} Sim\left(U_1, U\right)} \quad (5)$$

where $p\left(U_1, i\right)$ is the prediction function for user U_1's rating of item i, $sim(U_1, U)$ is the similarity value between the two users U_1 and U, $r_{U,i}$ is user U's rating of item i, $\overline{r_{U_1}}$ is user U_1's mean rating, and $N(U_1)$ is the neighbour set of user U_1.

The similarity between users is calculated using the multi-dimensional Euclidean distance metric, one of the most popular metrics. The following equation describes its formulas for determining the distance $d(R(U_1, i), R(U_2, i))$ between U_1 and U_2 users on item i.

$$Euclidean\ distance = \sqrt{\sum_{c=0}^{k} |R_c(U_1, i) - R_c(U_2, i)|} \quad (6)$$

where k denotes the number of criteria determined by the number of clusters, and $R_c(U_1, i)$ is user U_1's rating of item i on criterion c. Simply put, the average distance for all the shared items (I) between two users is the overall distance between the two users, as shown in Eq (7):

$$dist(U_1, U_2) = \frac{1}{|\mathrm{I}(U_1, U_2)|} \sum_{i \in \mathrm{I}(U_1, U_2)} \times d(R(U_1, i), R(U_2, i)) \quad (7)$$

The relation between the distance and similarity is an inverse relationship. The greater the distance between two users, the smaller their similarity value, and vice versa. As a result, using this distance measure, the similarity between two users is calculated as follows:

$$Sim(U_1, U_2) = \frac{1}{1 + dist(U_1, U_2)} \quad (8)$$

The optimum number for aspect clustering is determined by the best results obtained from the four experiments for the different K values representing the best CF performance. The CF performance for rating prediction is evaluated using the predictive accuracy metrics, which determine how closely the predicted ratings match actual ratings provided in the dataset. In particular, Mean Absolute Error (MAE), Mean Square Error (MSE) and Root Mean Square Error (RMSE) are used. A lower value of these metrics indicates a higher CF performance since they typically calculate the error difference between predicted and actual ratings [57]. The following equations define the equations of the three metrics, respectively.

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} (p_i - r_i)}{N} \quad (9)$$

$$\mathrm{MSE} = \frac{\sum_{i=1}^{N} (p_i - r_i)^2}{N} \quad (10)$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (p_i - r_i)^2}{N}} \quad (11)$$

where N: is the size of the test set, p_i: predicted rating calculated by CF approach, and r_i: actual rating given by the user.

2) DETERMINING THE NUMBER OF SIGNIFICANT ASPECTS

The second step of phase 2 determines whether all the main aspects that were extracted from the Amazon movie dataset using the SEAE approach have the same impact on CF performance or whether some have a more significant impact than others. The work of Musto et al. [6] was the inspiration of this step; they used their extraction method to extract 50 aspects and found that the CF algorithm performs better when only 10 of the 50 extracted aspects are used rather than all 50 extracted aspects.

This step evaluates CF performance using three different numbers/forms of aspects, the description of them as follows:

- Top-10 most frequent aspects (**Frtop-10**): the top 10 aspects among all the extracted aspects with the highest number of occurrences (i.e., frequencies).
- Top-10 most relevant aspects to the domain (**Sitop-10**): the top 10 aspects of the extracted aspects with the highest semantic similarity values to the movie domain.
- The brilliant aspect: the aspect with the highest rating among all the aspects mentioned by a user.

The top-10 aspects for the previous two forms are shown in Table 2.

TABLE 2. Top 10 aspects based on frequency and semantic similarity.

Top-10 by Frequency (Frtop-10)	Top-10 by Semantic Similarity (Sitop-10)
movie, film, time, scene, character, story, thing, performance, actor, man	movie, scene, story, performance, man, series, season, episode, show, life

For each number/form of aspects mentioned before, an experiment is conducted to assess CF performance in terms of rating prediction using the kNNwithMeans algorithm shown in Eq (5). The aspects are organised into 10 clusters in the Frtop-10 and Sitop-10; each cluster represents the score of an aspect included in the form. The multi-dimensional Euclidean distance metric is used to determine the similarities between users based on all their shared items. For the brilliant aspect, neighbours for a target user are selected based on their similarities with the target user in the shared items and have a similar brilliant aspect.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

This section presents the experiments carried out to accomplish the previously described phases. All the experiments use the extended Amazon movie dataset. We only consider users that have rated at least 20 movies and movies that have been rated by at least 20 users, yielding a total of 13,214 users, 17,022 movies, and 650,145 reviews. Also, all the experiments employed the kNNWithMeans rating prediction algorithm shown in Eq (5), with $k = 50$. The CF performance is assessed using three error metrics: MAE, MSE, and RMSE. The following subsections provide and discuss the results of each phase of the proposed approach.

A. PHASE 1: DATASET PREPARATION WITH ASPECTS CLUSTERING

This section provides the results of the two steps performed for phase 1.

1) EXTRACTING AND CLUSTERING ASPECTS

In this step, the aspects are extracted from the Amazon movie dataset using the SEAE approach, resulting in extracting

TABLE 3. Distributions of the main aspects based on different numbers of clusters.

Cluster	Aspects (K = 8)	Cluster	Aspects (K = 10)
#1	['life', 'man', 'woman', 'kid', 'child', 'girl', 'hour']	#1	['point', 'match', 'hour']
#2	['performance', 'video', 'match', 'production']	#2	['man', 'woman', 'kid', 'child', 'girl']
#3	['film', 'show', 'cast', 'set', 'comedy', 'audience']	#3	['show', 'performance', 'fun', 'audience', 'sense']
#4	['action', 'role', 'director', 'music', 'acting']	#4	['action', 'role', 'cast', 'set', 'acting', 'production']
#5	['something', 'nothing', 'point', 'fun', 'moment', 'picture', 'sense', 'idea']	#5	['movie', 'film', 'episode', 'music', 'comedy', 'song']
#6	['movie', 'story', 'character', 'plot', 'script']	#6	['scene', 'something', 'nothing', 'moment']
#7	['scene', 'season', 'actor', 'star', 'fan', 'song']	#7	['story', 'character', 'life', 'plot', 'idea', 'script']
#8	['dvd', 'series', 'episode', 'version', 'book', 'feature', 'release', 'edition']	#8	['season', 'actor', 'star', 'fan', 'director']
		#9	['dvd', 'video', 'picture']
		#10	['series', 'version', 'book', 'feature', 'release', 'edition']
Cluster	**Aspects (K = 13)**	**Cluster**	**Aspects (K = 15)**
#1	['hour']	#1	['music', 'audience']
#2	['music', 'fun', 'audience', 'song']	#2	['song']
#3	['man', 'woman', 'kid', 'child', 'girl']	#3	['kid', 'child', 'girl']
#4	['show', 'series', 'season', 'action', 'set', 'feature', 'match']	#4	['set', 'match']
#5	['character', 'life', 'actor', 'episode', 'plot', 'role', 'cast', 'acting']	#5	['scene', 'season', 'point', 'moment', 'hour']
#6	['scene', 'moment']	#6	['story', 'character', 'life', 'episode', 'plot', 'role', 'cast']
#7	['point', 'picture', 'sense', 'idea']	#7	['man', 'star', 'fan', 'woman']
#8	['performance', 'production']	#8	['action', 'release']
#9	['something', 'nothing']	#9	['performance', 'production']
#10	['movie', 'film', 'story', 'director', 'book', 'comedy', 'script']	#10	['movie', 'film', 'actor', 'director', 'book', 'picture', 'acting', 'script']
#11	['star', 'fan']	#11	['something', 'nothing']
#12	['dvd', 'video']	#12	['show', 'comedy']
#13	['version', 'release', 'edition']	#13	['fun', 'sense', 'idea']
		#14	['dvd', 'video']
		#15	['series', 'version', 'feature', 'edition']

49 main aspects and 481 core terms. Then, the aspects are clustered by applying the three sub-steps of the clustering aspects. The results of these steps are described below.

a: GROUP THE MAIN ASPECTS INTO CLUSTERS

In this step, the main aspects are divided into K clusters based on their semantic similarity using the K-means clustering algorithm. The output of this step generates four different lists because K is set to four different values: 8, 10, 13, and 15. The distributions of the main aspects in the lists that were generated based on each K value are shown in Table 3. This table demonstrates how effectively the K-means algorithm performs because all the aspects that are contained in the same cluster are semantically similar. The efficacy of the K-means algorithm is further demonstrated by the fact that the size of each cluster for each proposed K value is generally consistent with the other clusters.

b: MERGE CORE TERMS WITH THEIR RELATED MAIN ASPECTS

This step generates a list of core terms for each main aspect based on the semantic relation between the core term and the main aspect. Table 4 displays the results of this step and lists each main aspect along with its associated core terms. The main aspects are listed in order of the highest number of core terms (i.e., song) to the lowest. In addition, the lists of core terms are alphabetically sorted. Two points stand out in Table 4. First, it shows how closely related the meanings of most core terms are to the relevant aspects, such as the core terms 'image' and 'photo' for the aspect **picture** and 'essay' and 'novel' for the aspect **book**. The second point is that some of the core terms, such as those listed for the **performance**, **edition**, and **character** aspects, are spelt incorrectly. This type of error is common in many.user written reviews and considering these words will positively impact the aspect-based CF approach.

c: MERGE THE MAIN ASPECTS WITH THEIR CORE TERMS INTO CLUSTERS

This step creates a list comprising the aspects organised into clusters along with their core terms for each proposed K value (i.e., 8, 10, 13, 15). Table 5 illustrates the result of this step for K = 8 as an example, where each cluster contains the relevant aspects based on their semantic similarity using the

TABLE 4. All the main aspects with their lists of core terms.

song	film	movie	comedy	character	girl	dvd	something	book	director
album	auteur	actioner	absurdist	antihero	aunt	amazon	actually	anthology	administrator
anthem	biopic	actioners	campy	archetype	babe	bluray	always	author	adviser
ballad	biopics	adapation	comedic	baddie	blonde	boxset	anyway	autobiography	assistant
catchy	celluloid	boxoffice	comedienne	believability	brunette	cd	definitely	bestseller	associate
dancey	cinema	cineplex	comic	believable	boy	disc	do	biography	chairman
dirge	cinemascope	crapfest	comedian	believably	boyfriend	disney	else	book	chief
ditty	cinematic	filmgoer	drama	charachter	daughter	disneyland	everybody	bookshop	consultant
duet	cinematically	filmgoers	dramedy	charactor	girlfriend	dvd	everyone	bookstore	coordinator
falsetto	cinematographer	gorefest	hilarious	charater	grandmother	flix	kind	essay	dean
hummable	cinematography	gorefests	hilarity	charcter	granddaughter	fullscreen	maybe	fiction	deputy
hymn	doco	gorehound	humor	charcters	kidnapper	hbo	obviously	manuscript	directer
intro	docu	grindhouse	improv	charecter	kitten	hdtv	really	memoir	executive
instrumentals	docudrama	grossed	melodrama	likable	lass	ipod	so	novel	head
jazzy	documentarian	moive	mockumentaries	likeable	mother	laserdisc	somebody	novelist	manager
jingle	documentary	moviegoer	mockumentary	mythos	niece	laserdiscs	someone	novella	president
lullaby	featurette	oater	romcom	persona	old	netflix	someplace	paperback	secretary
lyric	filmaker	onscreen	satire	personality	playmate	nintendo	something	preface	spokesman
lyrical	filmed	pic	satiric	personna	princess	pixar	sort	tome	spokesperson
lyricist	filming	porno	satirical	portrayer	schoolmate	playstation	stuff	treatise	supervisor
melodic	filmmaker	prequel	shtick	protaganist	schoolgirl	sony	thing		
melody	filmmaking	prequels	slapstick	protagonist	stepdaughter	tv	think		
melodious	filmography	quadrilogy	spoof	storywise	stepfather	vcr	whatever		
ode	fim	remake	standup	superhero	stepsister	vhs			
playlist	fims	scarefest	starring	unlikable	tomboy	youtube			
poem	flic	sequal	sitcom	villain	teen				
remix	flick	sequel	tragicomedy	villainous	teenager				
remixed	flim	splatterfest	unfunny	backstory					
remixes	moviemaker	suspenser	vaudeville	storyline					
rendition	moviemakers	thriller	zany						
riff	moviemaking								
sang	reshot								
setlist	screenwriting								
sing	silents								
singing	talkie								
sings	telefilm								
songwriter	theatrical								
soulful	trilogy								
soundtrack	triology								
sung	verite								
songwriting									
verse									

kid	music	fun	women	script	child	nothing	video	actor	star	man
ballplayer	musician	funny	person	scriptwriter	baby	anybody	audio	actress	heartthrob	gentleman
brat	band	awesome	victim	screenplay	babysitter	certainly	camera	artiste	hunk	men
dad	classical	crazy	businesswoman	screenwriter	caregiver	everything	clip	costar	legend	motorcyclist
daddy	dance	enjoy	divorcee	potboiler	childrens	hardly	footage	costars	phenom	policeman
dude	entertainment	enjoyable	hitchhiker	libretto	daycare	little	slideshow	entertainer	standout	robber
fella	folk	entertaining	housewife	narration	grandchild	no	streaming	playwright	stardom	suspect
grandma	gospel	exciting	lady	scripted	grandparent	none	tape	singer	starlet	thief
grandpa	guitar	fabulous	maid	scripting	infant	nowhere	vid	stuntman	superstar	man
guy	jazz	fantastic	motorist	scriptwriting	son	never	videotape	thespian		gentleman
knucklehead	musical	fun	nun	storyboard	toddler	nobody	videotaped			men
lad, tyke	musically	goofy	policewoman	storyboards						motorcyclist
mom	musicianship	laugh	prostitute	teleplay						policeman
momma	orchestral	nice	waitress							robber
mommy	piano	thrill								
schooler	polka	wonderful								
youngster	saxophone									

episode	season	fan	version	story	picture	match	idea	performance	plot	release	audience	production
cliffhanger	finale	devotee	incarnation	article	image	clash	concept	peformance	conspiracy	relased	audiance	output
eps	league	diehard	iteration	fable	photo	matchup	notion	perfomance	plotline	released	crowd	producer
epsiode	opener	fanatic	variant	narrative	photograph	rematch	suggestion	performace	plotted	releasing	viewer	producing
epsiodes	seaons	fanbase	verion	retelling	portrait	tournament	theory	perfromance	plotting			
parter	spring	fandom	verison	tale	snapshot	game	thought	preformance				
premiering	summer	supporter	verson									
vignette	weekend											

moment	sense	hour	show	edition	feature	cast	acting	set	scene	point	life	series	role	action
juncture	feel	day	showed	editon	featured	casted	act	setting						
time	feeling	minute	showing	edtion	featuring	casting	acted							

TABLE 5. All main aspects with their core terms are organised into eight clusters.

Cluster	Aspects_CoreTerms
#1	[['**life**'], ['**man**', 'gentleman', 'men', 'motorcyclist', 'policeman', 'robber', 'suspect', 'thief'], ['**woman**', 'person', 'victim', 'businesswoman', 'divorcee', 'hitchhiker', 'housewife', 'lady', 'maid', 'motorist', 'nun', 'policewoman', 'prostitute', 'waitress', 'women'], ['**kid**', 'ballplayer', 'brat', 'dad', 'daddy', 'dude', 'fella', 'grandma', 'grandpa', 'guy', 'knucklehead', 'lad', 'mom', 'momma', 'mommy', 'schooler', 'tyke', 'youngster'], ['**child**', 'baby', 'babysitter', 'caregiver', 'childrens', 'daycare', 'grandchild', 'grandparent', 'infant', 'son', 'toddler'], ['**girl**', 'aunt', 'babe', 'blonde', 'brunette', 'boy', 'boyfriend', 'daughter', 'girlfriend', 'grandmother', 'granddaughter', 'kidnapper', 'kitten', 'lass', 'mother', 'niece', 'old', 'playmate', 'princess', 'schoolmate', 'schoolgirl', 'stepdaughter', 'stepfather', 'stepsister', 'tomboy', 'teen', 'teenager'], ['**hour**', 'day', 'minute']]
#2	[['**performance**', 'peformance', 'performace', 'perfromance', 'preformance'], ['**video**', 'audio', 'camera', 'clip', 'footage', 'slideshow', 'streaming', 'tape', 'vid', 'videotape', 'videotaped'], ['**match**', 'clash', 'matchup', 'rematch', 'tournament', 'game'], ['**production**', 'output', 'producer', 'producing']]
#3	[['**film**', 'auteur', 'biopic', 'biopics', 'celluloid', 'cinema', 'cinemascope', 'cinematic', 'cinematically', 'cinematographer', 'cinematography', 'doco', 'docu', 'docudrama', 'documentarian', 'documentary', 'featurette', 'filmaker', 'filmed', 'filming', 'filmmaker', 'filmmaking', 'filmography', 'fim', 'fims', 'flic', 'flick', 'flim', 'moviemaker', 'moviemakers', 'moviemaking', 'reshot', 'screenwriting', 'silents', 'talkie', 'telefilm', 'theatrical', 'trilogy', 'triology', 'verite'], ['**show**', 'showed', 'showing'], ['**cast**', 'casted', 'casting'], ['**set**', 'setting'], ['**comedy**', 'absurdist', 'campy', 'comedic', 'comedienne', 'comic', 'comedian', 'drama', 'dramedy', 'hilarious', 'hilarity', 'humor', 'improv', 'melodrama', 'mockumentaries', 'mockumentary', 'romcom', 'satire', 'satiric', 'satirical', 'shtick', 'slapstick', 'spoof', 'standup', 'starring', 'sitcom', 'tragicomedy', 'unfunny', 'vaudeville', 'zany'], ['**audience**', 'audiance', 'crowd', 'viewer']]
#4	[['**action**'], ['**role**'], ['**director**', 'administrator', 'adviser', 'assistant', 'associate', 'chairman', 'chief', 'consultant', 'coordinator', 'dean', 'deputy', 'directer', 'executive', 'head', 'manager', 'president', 'secretary', 'spokesman', 'spokesperson', 'supervisor'], ['**music**', 'musician', 'band', 'classical', 'dance', 'entertainment', 'folk', 'gospel', 'guitar', 'jazz', 'musical', 'musically', 'musicianship', 'orchestral', 'piano', 'polka', 'saxophone'], ['**acting**', 'act', 'acted']]
#5	[['**something**', 'actually', 'always', 'anyway', 'definitely', 'do', 'else', 'everybody', 'everyone', 'kind', 'maybe', 'obviously', 'really', 'so', 'somebody', 'someone', 'someplace', 'sort', 'stuff', 'thing', 'think', 'whatever'], ['**nothing**', 'anybody', 'certainly', 'everything', 'hardly', 'little', 'no', 'none', 'nowhere', 'never', 'nobody'], ['**point**'], ['**fun**', 'funny', 'awesome', 'crazy', 'enjoy', 'enjoyable', 'entertaining', 'exciting', 'fabulous', 'fantastic', 'goofy', 'laugh', 'nice', 'thrill', 'wonderful'], ['**moment**', 'juncture', 'time'], ['**picture**', 'image', 'photo', 'photograph', 'portrait', 'snapshot'], ['**sense**', 'feel', 'feeling'], ['**idea**', 'concept', 'notion', 'suggestion', 'theory', 'thought']]
#6	[['**movie**', 'actioner', 'actioners', 'adapation', 'boxoffice', 'cineplex', 'crapfest', 'filmgoer', 'filmgoers', 'gorefest', 'gorefests', 'gorehound', 'grindhouse', 'grossed', 'moive', 'moviegoer', 'oater', 'onscreen', 'pic', 'porno', 'prequel', 'prequels', 'quadrilogy', 'remake', 'scarefest', 'sequal', 'sequel', 'splatterfest', 'suspenser', 'thriller'], ['**story**', 'article', 'fable', 'narrative', 'retelling', 'tale'], ['**character**', 'antihero', 'archetype', 'baddie', 'believability', 'believable', 'believably', 'charachter', 'charactor', 'charater', 'charcter', 'charcters', 'charecter', 'likable', 'likeable', 'mythos', 'persona', 'personality', 'personna', 'portrayer', 'protaganist', 'protagonist', 'storywise', 'superhero', 'unlikable', 'villain', 'villainous', 'backstory', 'storyline'], ['**plot**', 'conspiracy', 'plotline', 'plotted', 'plotting'], ['**script**', 'scriptwriter', 'screenplay', 'screenwriter', 'potboiler', 'libretto', 'narration', 'scripted', 'scripting', 'scriptwriting', 'storyboard', 'storyboards', 'teleplay']]
#7	[['**scene**'], ['**season**', 'finale', 'league', 'opener', 'seaons', 'spring', 'summer', 'weekend'], ['**actor**', 'actress', 'artiste', 'costar', 'costars', 'entertainer', 'playwright', 'singer', 'stuntman', 'thespian'], ['**star**', 'heartthrob', 'hunk', 'legend', 'phenom', 'standout', 'stardom', 'starlet', 'superstar'], ['**fan**', 'devotee', 'diehard', 'fanatic', 'fanbase', 'fandom', 'supporter'], ['**song**', 'poem', 'album', 'anthem', 'ballad', 'catchy', 'dancey', 'dirge', 'ditty', 'duet', 'falsetto', 'hummable', 'hymn', 'intro', 'instrumentals', 'jazzy', 'jingle', 'lullaby', 'lyric', 'lyrical', 'lyricist', 'melodic', 'melody', 'melodious', 'ode', 'playlist', 'remix', 'remixed', 'remixes', 'rendition', 'riff', 'sang', 'setlist', 'sing', 'singing', 'sings', 'songwriter', 'soulful', 'soundtrack', 'sung', 'songwriting', 'verse']]
#8	[['**dvd**', 'amazon', 'bluray', 'boxset', 'cd', 'disc', 'disney', 'disneyland', 'flix', 'fullscreen', 'hbo', 'hdtv', 'ipod', 'laserdisc', 'laserdiscs', 'netflix', 'nintendo', 'pixar', 'playstation', 'sony', 'tv', 'vcr', 'vhs', 'youtube'], ['**series**'], ['**episode**', 'cliffhanger', 'eps', 'epsiode', 'epsiodes', 'parter', 'premiering', 'vignette'], ['**version**', 'incarnation', 'iteration', 'variant', 'verion', 'verison', 'verson'], ['**book**', 'anthology', 'author', 'autobiography', 'bestseller', 'biography', 'book', 'bookshop', 'bookstore', 'essay', 'fiction', 'manuscript', 'memoir', 'novel', 'novelist', 'novella', 'paperback', 'preface', 'tome', 'treatise'], ['**feature**', 'featured', 'featuring'], ['**release**', 'relased', 'released', 'releasing'], ['**edition**', 'editon', 'edtion']]

K-means clustering algorithm. Each aspect (shown in bold) is followed by its core terms (if available) to benefit from the diversity of vocabularies for the same aspect, which will improve aspect-based CF performance. These clusters will be used to find neighbours for a target user based on the similarities of each cluster's overall rating (where the clusters

contain multiple aspects) rather than the single rating of each aspect, as in the conventional aspect-based CF.

B. PHASE 2: DEVELOPING AN ASPECT-CLUSTERING-BASED COLLABORATIVE FILTERING APPROACH

This phase aims to develop an aspect-clustering-based CF approach. In order to develop this approach, several experiments are carried out to determine the values of two parameters: the optimal number of aspect clusters and the optimal number of used aspects. The results for identifying the optimal value for each parameter are described below.

1) DETERMINING THE OPTIMUM NUMBER OF CLUSTERS FOR ASPECTS

The results of phase 1 are four new datasets, named *Modified Dataset*, which includes the extended dataset along with the additional attributes specified for aspects. Each dataset is associated with a particular number of aspect clusters (K=8, 10, 13, and 15). This step uses these datasets for performing the CF rating prediction process individually, using the *k*NNwithMeans algorithm to determine the optimal number of aspect clusters. The CF performance for each dataset is reported in terms of the three metrics. The one with the best CF performance determines the optimum number for the aspect clusters (K). In particular, four experiments are conducted to identify the optimum number of aspect clusters, and each experiment is performed using five-fold cross-validation. In each fold, the dataset is split into two parts: training and testing (80% and 20%, respectively). The results of CF performance in terms of MAE, MSE, and RMSE are shown in Table 6.

TABLE 6. Results of the aspect clustering experiments.

Number of Clusters	MAE	MSE	RMSE
8	**0.7477**	**0.9930**	**0.9965**
10	0.7483	0.9940	0.9970
13	0.7489	0.9961	0.9981
15	0.7485	0.9942	0.9971

The best result in Table 6 is presented in boldface, which reflects the best CF performance for the rating prediction process using the *k*NNwithMeans algorithm. As a result, the optimum number for aspect clustering (K) is eight.

One interesting finding from Table 6 is that, when using the elbow method to determine the optimal value for K, the best value was 10. Subsequently, in an effort to demonstrate this, we conducted experiments using three other numbers close to the elbow's (10) and then reported the findings. After the experiments, K=8 yields better results than K=10, indicating that we should conduct further experiments instead of relying just on the elbow method.

2) DETERMINING THE NUMBER OF SIGNIFICANT ASPECTS

In this step, three experiments are performed, each of which offers diverse numbers/forms of aspects, as previously mentioned. It aims to verify whether all 49 aspects extracted from the Amazon movie dataset have the same impact on CF performance or whether some have a more significant impact than others. For each experiment, CF performance is evaluated in terms of the three metrics and then compared with CF performance using all extracted aspects organised into eight clusters (as previously indicated, the optimum number for aspect clustering is eight).

Similarly, five-fold cross-validation is carried out for each experiment, and the dataset is divided into training (80%) and testing (20%). The results of CF performance using the different forms of aspects are reported in Table 7.

TABLE 7. Results for the experiments of the different forms of aspects.

Method	MAE	MSE	RMSE
Frtop-10	0.7478	0.9932	0.9966
Sitop-10	0.7479	0.9935	0.9967
Brilliant_Aspect	0.7486	0.9950	0.9975
All_aspects	**0.7477**	**0.9930**	**0.9965**

Table 7 shows that using **All_aspects** method produces the best results (shown in bold) compared to the other methods. However, the **Frtop-10** and **Sitop-10** methods, on the other hand, also perform well because their values of the error metrics are just slightly different from those of the best method. This suggests that while all aspects are important, the top 10 aspects have the most significant impact on CF performance. Finally, while the **Brilliant_aspect** method has the lowest performance compared to the other methods, it still produces good results for the three metrics since there is no big difference between its results and those of the best method. This highlights that the brilliant aspect can influence the selection of neighbours' process which will affect CF performance.

V. EVALUATION AND ANALYSIS

The aspect-clustering-based CF approach (ASCF#8), in which all the aspects are organized into eight clusters, produces the best results based on prior experiments. The effectiveness of the ASCF#8 approach will be evaluated by comparing its performance with other available approaches. Specifically, the performance of the ASCF#8 approach is compared with three different approaches. The first is the aspect-based CF approach without clustering the aspects (ASCF#0). This approach aims to assess the clustering process's effectiveness in enhancing CF performance using the *k*NN algorithm (*k* value is set to 50 as ASCF#8 approach) with the Euclidean similarity metric. The second compared approach is the multi-criteria CF approach (MCCF) proposed by Wasid and Ali [10], which clustered users based on their shared criteria (i.e., aspects). Our study employs a large-scale

TABLE 8. The results of MAE, MSE and RMSE for all the compared approaches.

Method	MAE	MSE	RMSE	% of improvement		
				MAE	MSE	RMSE
ASCF#8	**0.7477**	**0.9930**	**0.9965**	-	-	-
ASCF#0	0.8522	1.3468	1.1605	12.26%	26.27%	14.13%
MCCF (50)	0.9096	1.4552	1.2063	17.80 %	31.76%	17.39%
MCCF (30)	0.8951	1.4077	1.1865	16.47%	29.46%	16.01%
MCCF (10)	0.8294	1.2141	1.1018	9.85%	18.21%	9.56%
CFP	0.8986	1.3474	1.1608	16.79%	26.30%	14.15%

dataset, whereas Wasid and Ali's study employed a small-scale dataset. The MCCF approach used the Yahoo! Movies dataset, which has 62,156 ratings for 976 movies from 6,078 users. The dataset is further reduced to only 19,050 ratings provided by 484 users in 945 movies by extracting only users who gave ratings for at least 20 movies. This is not the case in our study, which concentrates on large-scale datasets. Besides, the kNN algorithm, which is used for the MCCF approach's rating prediction, has a k value set to 30 in their study [10]. Due to the various dataset sizes utilised in our study and their study, three k values are tested in this evaluation section for MCCF approach. Specifically, we examined three neighbourhood size values: 10, 30, and 50 because we cannot just rely on the neighbourhood size that was determined in their study (i.e., 30). The last approach is the single-criterion CF that used the Pearson correlation similarity metric (CFP). CFP relies only on the overall ratings for rating prediction and does not use aspects.

All the experiments use the Amazon movie dataset. five-fold cross-validation are used for each approach, with the dataset being split into 80% training and 20% testing for each fold. The experiment findings are shown in Table **8** in terms of MAE, MSE, and RMSE as well as the percentage of improved performance of the proposed approach over the baselines.

The results show that the proposed ASCF#8 approach considerably outperformed the baseline approaches in the three metrics. It can be noted from the results that clustering aspects improve CF performance, as hypothesised in this study. This is evidenced by the fact that, when compared to ASCF#0, the values of MAE, MSE and RMSE all indicate improved performance of 12.26%, 26.27%, and 14.13%, respectively. Additionally, the ASCF#8 and ASCF#0 approaches show better performance when compared to the CFP approach, proving that multiple-criteria (i.e., aspects) CF performs better than single-criteria CF. It offers more information about user preferences, which helps identify the most suitable neighbours for the target user and enhances CF performance.

Moreover, according to the results in Table **8**, the MCCF approach performs best when $k = 10$, and these results exceed the CFP approach, which is consistent with Wasid and Ali's findings [10]. On the other hand, the ASCF#8 approach significantly outperforms the MCCF (10) approach with improvements of 9.85%, 18.21%, and 9.56%, respectively in terms of MAE, MSE and RMSE. One of the reasons of this finding is that we rely on learned rather than fixed aspects, unlike Wasid and Ali's work. The learned aspects are aspects extracted from the user reviews, not general ones as the fixed aspects. The fixed aspects are few and technical ones that have an impact on calculating the appropriate neighbors and do not adequately reflect the user preferences [1]. Also, the results of both ASCF#8 and MCCF (10) provide an interesting finding for this study, which is applying approaches designed for small-scale datasets to large-scale datasets does not work efficiently. Lastly, the MCCF (10) approach surpasses the ASCF#0 approach in terms of the error metrics, demonstrating the significance of the clustering process in improving CF performance.

VI. CONCLUSION AND FUTURE WORK

In this study, we proposed an aspect-clustering-based CF approach to improve CF performance for the rating prediction process. The aim of using aspect clustering is to enhance the selection of the neighbourhood set by finding users with similar preferences to the target one, which impacts CF performance. Specifically, the approach aims to reduce the sparseness in the multi-criteria rating matrix by grouping aspects into clusters based on semantic similarity, which will be less expensive and require less memory to discover the neighbourhood set. Aspect clusters are multi-criteria that are integrated into the aspect-based CF approach, and the similarities between users are calculated using the multi-dimensional Euclidean distance to identify the appropriate neighbours for the target user who share similar preferences on the available aspect clusters. The clustering process is done using the K-means algorithm, which proves its efficacy in aspect clustering, as shown in the results. In addition, different forms of aspects are proposed and assessed using the CF rating prediction algorithm to identify the number of aspects that significantly impacts CF performance. Experiments are carried out using the Amazon movie dataset to show the efficiency of the proposed approach in improving CF performance. Results show that grouping aspects into eight clusters and calculating user similarity based on these clusters significantly affects CF performance. Moreover, among the 49 extracted aspects, the top 10 aspects significantly impact CF performance. On the other hand,

utilising all the aspects in the proposed approach is superior to utilising only the top 10 aspects. Finally, the proposed aspect-clustering-based CF approach outperformed the CF approach without clustering and the other baselines in prediction accuracy by the MAE, MSE, and RMSE metrics.

The evaluation was conducted mainly on objective prediction accuracy, i.e., algorithm performance. However, prediction accuracy metrics do not replicate the real user experience. According to McLaughlun and Herlocker [58], precision, recall, and Normalised Discounted Cumulative Gain (NDCG) metrics reflect the user's real experience because, in most cases, users actually received ranked lists from a recommender. Thus, the near future is to evaluate the proposed method based on these metrics.

Furthermore, we also plan to extract and explore more review elements and improve CF recommendation systems using deep learning techniques.

REFERENCES

[1] S. M. Al-Ghuribi and S. A. M. Noah, "Multi-criteria review-based recommender system—The state of the art," *IEEE Access*, vol. 7, pp. 169446–169468, 2019.

[2] A. A. Amer, H. I. Abdalla, and L. Nguyen, "Enhancing recommendation systems performance using highly-effective similarity measures," *Knowl.-Based Syst.*, vol. 217, Apr. 2021, Art. no. 106842.

[3] M. Garcia-Cumbreras, A. Montejo-Raez, and M. C. Diaz-Galiano, "Pessimists and optimists: Improving collaborative filtering through sentiment analysis," *Exp. Syst. Appl.*, vol. 40, no. 17, pp. 6758–6765, Dec. 2013.

[4] B. A. H. Murshed, J. Abawajy, S. Mallappa, M. A. N. Saif, S. M. Al-Ghuribi, and F. A. Ghanem, "Enhancing big social media data quality for use in short-text topic modeling," *IEEE Access*, vol. 10, pp. 105328–105351, 2022.

[5] L. Chen, G. Chen, and F. Wang, "Recommender systems based on user reviews: The state of the art," *User Model. User-Adapted Interact.*, vol. 25, pp. 99–154, Jun. 2015.

[6] C. Musto, M. de Gemmis, G. Semeraro, and P. Lops, "A multi-criteria recommender system exploiting aspect-based sentiment analysis of users' reviews," in *Proc. 11th ACM Conf. Recommender Syst.*, Aug. 2017, pp. 321–325.

[7] A. Da'u, N. Salim, I. Rabiu, and A. Osman, "Weighted aspect-based opinion mining using deep learning for recommender system," *Exp. Syst. Appl.*, vol. 140, Feb. 2020, Art. no. 112871.

[8] P. Liu, L. Zhang, and J. A. Gulla, "Multilingual review-aware deep recommender system via aspect-based sentiment analysis," *ACM Trans. Inf. Syst.*, vol. 39, no. 2, pp. 1–33, Apr. 2021.

[9] B. Ray, A. Garain, and R. Sarkar, "An ensemble-based hotel recommender system using sentiment analysis and aspect categorization of hotel reviews," *Appl. Soft Comput.*, vol. 98, Jan. 2021, Art. no. 106935.

[10] M. Wasid and R. Ali, "An improved recommender system based on multi-criteria clustering approach," *Proc. Comput. Sci.*, vol. 131, pp. 93–101, Jan. 2018.

[11] D. Tallapally, R. S. Sreepada, B. K. Patra, and K. S. Babu, "User preference learning in multi-criteria recommendations using stacked auto encoders," in *Proc. 12th ACM Conf. Recommender Syst.*, Sep. 2018, pp. 475–479.

[12] Y. Zheng, "Utility-based multi-criteria recommender systems," in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2019, pp. 2529–2531.

[13] R. M. D'Addio, M. A. Domingues, and M. G. Manzato, "Exploiting feature extraction techniques on users' reviews for movies recommendation," *J. Brazilian Comput. Soc.*, vol. 23, no. 1, pp. 1–16, Dec. 2017.

[14] V. La Gatta, V. Moscato, M. Pennone, M. Postiglione, and G. Sperli, "Music recommendation via hypergraph embedding," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 10, 2022, doi: 10.1109/TNNLS.2022.3146968.

[15] G. Adomavicius and Y. Kwon, "New recommendation techniques for multicriteria rating systems," *IEEE Intell. Syst.*, vol. 22, no. 3, pp. 48–55, May 2007.

[16] N. Manouselis and C. Costopoulou, "Experimental analysis of design choices in multiattribute utility collaborative filtering," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 21, no. 2, pp. 311–331, 2007.

[17] G. Adomavicius, N. Manouselis, and Y. Kwon, "Multi-criteria recommender systems," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011, pp. 769–803.

[18] A. J. Gallego, J. R. Rico-Juan, and J. J. Valero-Mas, "Efficient k-Nearest neighbor search based on clustering and adaptive k values," *Pattern Recognit.*, vol. 122, Feb. 2022, Art. no. 108356.

[19] T. Zhu, Y. Ren, W. Zhou, J. Rong, and P. Xiong, "An effective privacy preserving algorithm for neighborhood-based collaborative filtering," *Future Gener. Comput. Syst.*, vol. 36, pp. 142–155, Jul. 2014.

[20] B. Alhijawi, G. Al-Naymat, N. Obeid, and A. Awajan, "Novel predictive model to improve the accuracy of collaborative filtering recommender systems," *Inf. Syst.*, vol. 96, Feb. 2021, Art. no. 101670.

[21] C. Ma, P. Kang, B. Wu, Q. Wang, and X. Liu, "Gated attentive-autoencoder for content-aware recommendation," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Jan. 2019, pp. 519–527.

[22] S. M. Al-Ghuribi, S. A. M. Noah, and S. Tiun, "Unsupervised semantic approach of aspect-based sentiment analysis for large-scale user reviews," *IEEE Access*, vol. 8, pp. 218592–218613, 2020.

[23] A. Ebadi and A. Krzyzak, "A hybrid multi-criteria hotel recommender system using explicit and implicit feedbacks," in *Proc. 18th Int. Conf. Appl. Sci. Inf. Syst. Technol. (ICASIST)*, Amsterdam, The Amsterdam, 2016, pp. 1450–1458.

[24] Y. Lu, R. Dong, and B. Smyth, "Coevolutionary recommendation model: Mutual learning between ratings and reviews," in *Proc. World Wide Web Conf.*, 2018, pp. 773–782.

[25] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: Introduction and challenges," in *Recommender Systems Handbook*. Berlin, Germany: Springer, 2015, pp. 1–34.

[26] Z. Fayyaz, M. Ebrahimian, D. Nawara, A. Ibrahim, and R. Kashef, "Recommendation systems: Algorithms, challenges, metrics, and business opportunities," *Appl. Sci.*, vol. 10, no. 21, p. 7748, Nov. 2020.

[27] C. Yang, X. Yu, Y. Liu, Y. Nie, and Y. Wang, "Collaborative filtering with weighted opinion aspects," *Neurocomputing*, vol. 210, pp. 185–196, Oct. 2016.

[28] N. J. Jamil, S. A. M. Noah, and M. Mohd, "Collaborative item recommendations based on friendship strength in social network," *Int. J. Mach. Learn. Comput.*, vol. 10, no. 3, pp. 437–443, May 2020.

[29] H.-K. Bae, H.-O. Kim, W.-Y. Shin, and S.-W. Kim, "'How to get consensus with neighbors?': Rating standardization for accurate collaborative filtering," *Knowl.-Based Syst.*, vol. 234, Dec. 2021, Art. no. 107549.

[30] R. Ahuja, A. Solanki, and A. Nayyar, "Movie recommender system using K-means clustering and K-Nearest neighbor," in *Proc. 9th Int. Conf. Cloud Comput., Data Sci. Eng.*, Jan. 2019, pp. 263–268.

[31] N. Lathia, S. Hailes, and L. Capra, "KNN CF: A temporal social network," in *Proc. ACM Conf. Recommender Syst.*, Oct. 2008, pp. 227–234.

[32] S. M. Al-Ghuribi and S. Alshomrani, "A simple study of Webpage text classification algorithms for Arabic and English languages," in *Proc. Int. Conf. IT Converg. Secur. (ICITCS)*, Dec. 2013, pp. 1–5.

[33] A. Noaman and S. Al-Ghuribi, "A new approach for Arabic text classification using light stemmer and probabilities," *Int. J. Academic Res.*, vol. 4, no. 3, pp. 114–122, 2012.

[34] L. Liu, N. Mehandjiev, and D.-L. Xu, "Multi-criteria service recommendation based on user criteria preferences," in *Proc. 5th ACM Conf. Recommender Syst.*, Oct. 2011, pp. 77–84.

[35] P. Melville, R. J. Mooney, and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," in *Proc. IAAI*, vol. 23, 2002, pp. 187–192.

[36] W. Zhang, G. Ding, L. Chen, C. Li, and C. Zhang, "Generating virtual ratings from Chinese reviews to augment online recommendations," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 1, pp. 1–17, Jan. 2013.

[37] N. Pappas and A. Popescu-Belis, "Adaptive sentiment-aware one-class collaborative filtering," *Exp. Syst. Appl.*, vol. 43, pp. 23–41, Jan. 2016.

[38] A. S. Fadel, M. E. Saleh, and O. A. Abulnaja, "Arabic aspect extraction based on stacked contextualized embedding with deep learning," *IEEE Access*, vol. 10, pp. 30526–30535, 2022.

[39] M. Shams, N. Khoshavi, and A. Baraani-Dastjerdi, "LISA: Language-independent method for aspect-based sentiment analysis," *IEEE Access*, vol. 8, pp. 31034–31044, 2020.

[40] A. García-Pablos, M. Cuadros, and G. Rigau, "W2VLDA: Almost unsupervised system for aspect based sentiment analysis," *Exp. Syst. Appl.*, vol. 91, pp. 127–137, Jan. 2018.

[41] C.-C. Musat, Y. Liang, and B. Faltings, "Recommendation using textual opinions," in *Proc. Int. Joint Conf. Artif. Intell.*, 2013, pp. 2684–2690.
[42] F. A. Nielsen, "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs," 2011, *arXiv:1103.2903*.
[43] K. Bauman, B. Liu, and A. Tuzhilin, "Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 717–725.
[44] R. Catherine and W. Cohen, "TransNets: Learning to transform for recommendation," in *Proc. 11th ACM Conf. Recommender Syst.*, Aug. 2017, pp. 288–296.
[45] A. Da'u, N. Salim, I. Rabiu, and A. Osman, "Recommendation system exploiting aspect-based opinion mining with deep learning method," *Inf. Sci.*, vol. 512, pp. 1279–1292, Feb. 2020.
[46] I. Cantador, A. Carvallo, and F. Diez, "Rating and aspect-based opinion graph embeddings for explainable recommendations," 2021, *arXiv:2107.03385*.
[47] R. Wang, Y. Jiang, and J. Lou, "ADCF: Attentive representation learning and deep collaborative filtering model," *Knowl.-Based Syst.*, vol. 227, Sep. 2021, Art. no. 107194.
[48] J. Zhang, Y. Lin, M. Lin, and J. Liu, "An effective collaborative filtering algorithm based on user preference clustering," *Appl. Intell.*, vol. 45, no. 2, pp. 230–240, Sep. 2016.
[49] X. Liu, "An improved clustering-based collaborative filtering recommendation algorithm," *Cluster Comput.*, vol. 20, no. 2, pp. 1281–1288, 2017.
[50] S. M. Al-Ghuribi, S. A. Noah, and S. Tiun, "Various pre-processing strategies for domain-based sentiment analysis of unbalanced large-scale reviews," in *Proc. Int. Conf. Advanced Intell. Syst. Inform.* Cham, Switzerland: Springer, 2020, pp. 204–214.
[51] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: Understanding rating dimensions with review text," in *Proc. 7th ACM Conf. Recommender Syst.*, Oct. 2013, pp. 165–172.
[52] K. Thabit and S. M. Al-Ghuribi, "A new search algorithm for documents using blocks and words prefixes," *Sci. Res. Essays*, vol. 8, no. 16, pp. 640–648, Apr. 2013.
[53] A. Penta and A. Pal, "What is this cluster about? Explaining textual clusters by extracting relevant keywords," *Knowl.-Based Syst.*, vol. 229, Oct. 2021, Art. no. 107342.
[54] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*.
[55] P. Bholowalia and A. Kumar, "EBK-means: A clustering technique based on elbow method and K-means in WSN," *Int. J. Comput. Appl.*, vol. 105, no. 9, pp. 1–8, 2014.
[56] L. Zhu, G. Wang, and X. Zou, "A study of Chinese document representation and classification with Word2vec," in *Proc. 9th Int. Symp. Comput. Intell. Design (ISCID)*, Dec. 2016, pp. 298–302.
[57] S. M. Al-Ghuribi and S. A. M. Noah, "A comprehensive overview of recommender system and sentiment analysis," 2021, *arXiv:2109.08794*.
[58] M. R. McLaughlin and J. L. Herlocker, "A collaborative filtering algorithm and evaluation metric that accurately model the user experience," in *Proc. 27th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2004, pp. 329–336.

SHAHRUL AZMAN MOHD NOAH (Member, IEEE) received the B.Sc. degree (Hons.) in mathematics from Universiti Kebangsaan Malaysia (UKM), Malaysia, in 1992, and the M.Sc. and Ph.D. degrees in information studies from The University of Sheffield, U.K., in 1994 and 1998, respectively. He is currently a Professor with the Center for Artificial Intelligence Technology, UKM. He also heads the Knowledge Technology Research Group. His research interests include information retrieval and ontology, with a special emphasis on semantic search and recommender systems. He has published more than 200 research articles in these areas. He is a member of the International Association for Ontology and its Applications (IAOA) and the IEEE Computer Society. He is the Chair of Persatuan Capaian Maklumat dan Pengurusan Pengetahuan (PECAMP).

MAWAL A. MOHAMMED received the B.Sc. degree (Hons.) in software engineering from Taiz University, Yemen, in 2008, and the M.Sc. and Ph.D. degrees from the King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, in 2014 and 2020, respectively. He is currently an Assistant Professor with Prince Sattam Bin Abdulaziz University, Al-Kharj, Riyadh, Saudi Arabia. His research interests include software engineering, big data, and recommender systems.

SULTAN NOMAN QASEM received the B.Sc. degree (Hons.) from the Computer Science Department, Faculty of Science, Al-Mustansiriya University, Baghdad, Iraq, in 2002, and the M.Sc. and Ph.D. degrees from the Faculty of Computer Science and Information Systems, University Technology Malaysia, Johor, Malaysia, in 2008 and 2011, respectively. He is currently an Associate Professor with the Department of Computer Science, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University. He has authored/coauthored more than 55 research publications in peer-reviewed reputed journals, book chapters, and conference proceedings. His research interests include, applied artificial intelligence, data science, multi-objective machine learning, object recognition, and health informatics. He has served as the program committee member for various international conferences and a reviewer for various international journals.

BELAL ABDULLAH HEZAM MURSHED received the B.Sc. degree (Hons.) in computer science and information systems from Taiz University, Yemen, in 2008, and the M.Sc. degree (Hons.) in computer science and the Ph.D. degree from the University of Mysore (UOM), India, in 2016 and 2023, respectively. He has more than eight years of experience in both industry and teaching. He is currently a Teaching Assistant with the Faculty of Engineering and Information Technology, Amran University, Yemen. His research interests include data mining, machine learning, NLP, artificial intelligence, topic modeling, optimization algorithms, big data, and the Internet of Things.

SUMAIA MOHAMMED AL-GHURIBI received the B.Sc. degree (Hons.) in computer science from Taiz University, Yemen, in 2008, the M.S. degree in computer science from King Abdulaziz University, Jeddah, Saudi Arabia, in 2014, and the Ph.D. degree from Universiti Kebangsaan Malaysia (UKM), in 2021. Her research interests include natural language processing, web mining, sentiment analysis, and recommender systems.