Personalized Video Recommendation Using Rich Contents from Videos

Xingzhong Du, Hongzhi Yin, Ling Chen, Yang Wang, Yi Yang, and Xiaofang Zhou, *Fellow, IEEE*

Abstract—Video recommendation has become an essential way of helping people explore the massive videos and discover the ones that may be of interest to them. In the existing video recommender systems, the models make the recommendations based on the user-video interactions and single specific content features. When the specific content features are unavailable, the performance of the existing models will seriously deteriorate. Inspired by the fact that rich contents (e.g., text, audio, motion, and so on) exist in videos, in this paper, we explore how to use these rich contents to overcome the limitations caused by the unavailability of the specific ones. Specifically, we propose a novel general framework that incorporates arbitrary single content feature with user-video interactions, named as collaborative embedding regression (CER) model, to make effective video recommendation in both in-matrix and out-of-matrix scenarios. Our extensive experiments on two real-world large-scale datasets show that CER beats the existing recommender models with any single content feature and is more time efficient. In addition, we propose a priority-based late fusion (PRI) method to gain the benefit brought by the integrating the multiple content features. The corresponding experiment shows that PRI brings real performance improvement to the baseline and outperforms the existing fusion methods.

Index Terms—Recommender model, personalization, video content analysis, rich content features, late fusion

1 Introduction

Watching online videos has become one of the indispensable entertainment activities in daily life. Many famous websites, such as YouTube, Netflix and Hulu, host a tremendous number of videos to meet such demand. The massive video repositories have placed an enormous burden on users when trying to find videos of interest [1], [2]. To address this problem, most video websites have adopted recommender systems as a promising way to help users explore the world of videos [3], [4]. Existing recommender methods can be categorized into three classes [1]: content-based, collaborative filtering (CF)-based, and hybrid. Content-based methods [5] recommend items to users based on the content similarities between the user profile and item contents. CF-based methods [6], [7] accomplish the same task by the behavior similarities between the users or items. Hybrid methods [8], [9] seek the best of both worlds by combining both content and CF-based methods, and have gained increasing popularity in recent years [10], [11], [12].

Most existing hybrid recommender models [10], [11], [12] use only textual contents to facilitate recommendations. When these models are used for video recommendation, they are fragile because the textual contents are often missing, scarce, or poor-quality for user-generated videos. For example, plenty of videos on Youtube only have titles, and many of them are not suitable for the content-based recommendation task as they are deliberately titled to attract users. If the hybrid recommender models are forced to generate recommendations with these texts, their performance will be much worse than the expectation. The limitation caused by the unavailability of the textual contents has been noticed in a few works [13], [14], [15] for video, music or product recommendation. They propose to exploit non-textual content features (e.g., color histograms) to overcome the limitation. However, similar to models using textual contents, all these models [13], [14], [15] only explore single specific non-textual content features. Thus, these models still face the performance drop when their dependent content features are unavailable.

The recent study in [16] shows that multiple types of content features can influence users' choices on videos. Enlightened by this new finding, we propose to use rich contents from videos to enhance the recommendation task. Our rich content features consist of both textual content features and non-textual content features. The textual content features are made up of video descriptions and video meta information such as actors and directors, and the non-textual content features are made up of audios [17], scenes [18] and motions [19] from videos themselves. With rich content features, our goals are two folds: (1) we want to discover a general and effective hybrid model, which is able to integrate any single content feature into collaborative filtering, to marginalize the performance drop caused by the unavailability of one specific content; (2) we want to explore a multiple feature fusion method, which is inspired by the success of the multiple feature fusion in other relevant areas [17], [20], to further improve the recommendation accuracy.

- *X. Du, H. Yin, and X. Zhou are with the School of Information Technology & Electric Engineering, The University of Queensland, St Lucia, QLD 4072, Australia. E-mail: {x.du, h.yin1}@uq.edu.au, zxf@itee.uq.edu.au.*
- *L. Chen and Y. Yang are with the Centre for Artificial Intelligence, University of Technology Sydney, Ultimo, NSW 2007, Australia. E-mail: {ling.chen, yi.yang}@uts.edu.au.*
- *Y. Wang is with the School of Computer Science & Engineering, The University of New South Wales, Sydney, NSW 2052, Australia. E-mail: wangy@cse.unsw.edu.au.*

Manuscript received 27 Dec. 2017; revised 1 Oct. 2018; accepted 3 Dec. 2018. Date of publication 7 Dec. 2018; date of current version 4 Feb. 2020. (Corresponding author: Hongzhi Yin). Recommended for acceptance by X. Li. Digital Object Identifier no. 10.1109/TKDE.2018.2885520

With the two-fold goals, we start our study by analyzing the performance of existing hybrid models [10], [11], [14], [21] with single content features, because each of them is tightly coupled with one specific content feature. The performance of these existing models are tested in two different but important scenarios [10]: in-matrix recommendation and out-of-matrix recommendation. These two scenarios are also known as warm-start and cold-start recommendations. Specifically, for a test case, if the video in it has appeared in training data (i.e., the user-video interaction matrix), then it is a in-matrix recommendation; otherwise, it is a out-of-matrix recommendation. We find that none of these existing models could achieve very sound performance in both in-matrix and out-matrix scenarios even with their original content features. In particular, we observe that weighted matrix factorization (WMF)-based models [10], [11], [14], [22] achieve higher accuracy than Bayesian personalized ranking (BPR)-based models [21], [23] in the in-matrix scenario, but lower accuracy in the out-of-matrix scenario. WMF-based models' poorer performance in out-of-matrix scenario is attributed to that their model are designed for in-matrix recommendation especially. For example, collaborative deep learning (CDL) model is a WMF-based model proposed in [11]. CDL uses stack denoising auto-encoder (SDAE) to make its content side serve in-matrix recommendation when ratings are sparse. Accordingly, CDL neglects its content side is also important for out-of-matrix recommendation. To address that, we propose a collaborative embedding regression (CER) model to effectively incorporate collaborative filtering with single content features. Our extensive evaluations show that, CER significantly outperforms both WMF and BPR-based models in the out-of-matrix scenario with any content feature, while keeps the excellent performance of WMF-based models in the in-matrix scenario. Moreover, CER's model training is more efficient on large datasets than the other models'.

Most existing hybrid recommender models including our proposed CER model are designed to work with single content features, which neglects the performance improvement from the multiple feature fusion. Thus, how to design an effective fusion method to further improve the recommendation accuracy is another challenge in our study. In other areas, there are two widely used yet independent strategies to fuse multiple content features [24]: early fusion and late fusion. The early fusion combines the multiple content features before model training [17]. Even though sometimes early fusion methods obtain better performance than the individual content features, they have a number of limitations. First, the input content feature space is fixed, so early fusion models need to retrain from a scratch when different content features come. This property makes early fusion unadaptive to the streaming data from the real-world recommender systems. Second, Most works on early fusion concatenate multiple content features into single ones as the input for model training [12], [25], [26]. The concatenation results into very high dimension that leads to extremely unprecedentedly computational costs in training. Third, another line of the early fusion methods make all the content features additive in a homogeneous latent space and use the sum of them as the input for training [12]. However, the textual, audio, visual and motion information contained in videos are quite diverse and heterogeneous. It is unreasonable to add them in a homogeneous latent space. The other line of the fusion research focuses on the late fusion of multiple features. A typical late fusion method obtains the fused scores by the weighted sums of the scores from different content features [27]. Learning-to-rank techniques (e.g., ranking SVM) [17], [27] are state-of-the-art late fusion methods. Compared to early fusion, late fusion treats the content features in heterogeneous space and is more flexible to adapt to the streaming data and more efficient in training [28]. A number of recent successful multimedia event detection systems [17], [20] have adopted late fusion in combining multiple features.

In this paper, we explore the late fusion strategy to further improve CER using rich content features. Specifically, we propose a priority-based late fusion method (PRI) whose innovation is using exponential weights to model the overwhelming influences from the stronger content features. Given rich content features, PRI first prioritizes the content features by evaluating their recommendation accuracy in validation. After that, PRI applies grid search to obtain the optimal base for the exponential weights. Finally, PRI calculates each content feature's weight using the optimal base and the exponents derived from the priority positions. The effectiveness of PRI has been evaluated by comparing it with five fusion methods including both early and late fusion on two real-world large-scale datasets.

To summarize, the contributions of this paper include:

- We have proposed a novel and versatile framework for effective personalised video recommendation in both out-of-matrix and in-matrix scenarios by exploiting rich content features from videos.
- We have deeply analyzed and studied the performance of existing hybrid models in both the in-matrix and out-of-matrix scenarios. Based on the sound study, we propose a collaborative embedding regression model, which effectively combines collaborative filtering with arbitrary single textual or non-textual content feature, to generate more accurate video recommendation in both in-matrix and out-of-matrix scenarios.
- We further study how to generate more accurate recommendations for new videos by fusing rich content features. We find existing fusion methods, either early or late, cannot consistently make performance improvement due to large accuracy divergences between different content features. We therefore propose a priority-based late fusion (PRI) method that prioritizes the content features and assigns the exponential weights according to the priorities.

2 Background and Related Work

Given m users and n items, $r_{ij} \in \{?, +\}$ denotes the ith user's implicit feedback on the jth item, where '+' means ith user likes jth item; '?' means ith user dislikes or is not aware of jth item. As a convention [23], we transform r_{ij} to $\{0, 1\}$ as a implicit rating. Putting all the ratings together forms the implicit rating matrix denoted as $R = \{0, 1\}^{m \times n}$ for recommender models, as shown in Fig. 1.

Given a target user, a recommender system is required to find personalized top-k items that the user is potentially interested in. The task can be further divided into two scenarios, namely in-matrix and out-of-matrix. In the in-matrix scenario, systems recommend top-k items which have not been rated by the target user but have been rated by other users [3], [29]. Based on the collaborations between similar users or items, state-of-the-art models [10], [11], [14], [22] apply collaborative

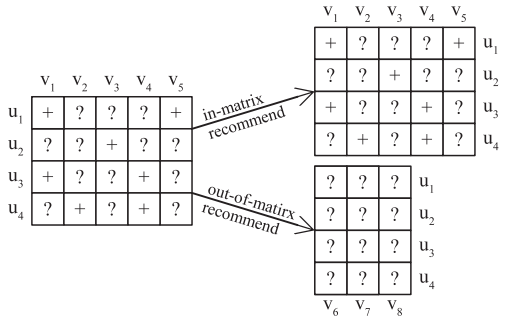


Fig. 1. Implicit rating matrix, in-matrix recommendation, and out-of-matrix recommendation.

filtering to generate recommendations. In the out-of-matrix scenario, systems recommend top-k new items that have not been rated by any user [10] (i.e., cold-start recommendation [10], [30]). Since the collaborations between users or items are not existing, CF-based models become ineffective, whereas content-based models perform better.

Weighted matrix factorization [31] and Bayesian personalized ranking [23] represent the state-of-the-art recommender models in in-matrix scenario. Both of them construct objective functions based on matrix factorization and learn collaborations between users or items during the training. After training, a latent vector is learned for each user or item. Users' ratings on items are then predicted by the inner products between latent vectors. Finally, items with the highest predicted ratings will be selected as the top-k recommendation for users. The major difference between WMF and BPR is the learning objective. In particular, WMF [31] aims at minimizing rating prediction errors, while BPR [23] aims at preserving pair-wise personalized orders.

Recently, both WMF and BPR were extended to incorporate content features, so their variants can be applied to generate recommendations in both in-matrix and out-of-matrix scenarios. The representative WMF-based models include collaborative topic regression (CTR) [10], deep content-based music recommender model (DPM) [14] and collaborative deep learning [11]. CTR and CDL are specific to the textual features of items, while DPM is specific to audio features of items. The representative BPR-based models are visual Bayesian personalized ranking (VBPR) [21] and Visual-CLiMF [32]. VBPR and Visual-CLiMF are specific to visual features from pretrained convolutional neural networks (CNN). Visual-CLiMF enhances VBPR by learning the approximate reciprocal rank instead of pair-wise rank in the optimization. Different from above models, VideoTopic proposed in [33] use textual and visual features simultaneously to perform out-of-matrix video recommendations, which is actually a kind of early fusion. Because poor visual feature and few user collaborations are used in model training, VideoTopic's accuracy is much lower than those achieved by CF-based models [10], [11], [21].

3 Video Content Features

The whole work flow in our study that covers from rich content feature extraction to personalized video recommendation generation is depicted in Fig. 2. In this section, we first describe how the rich content features, including both textual and non-textual, are extracted for video recommendation. These features are crucial for the training of the hybrid models, and they directly decide the out-of-matrix recommendation performance.

3.1 Textual Content Features

Many existing hybrid video recommender systems [3], [4], [11] make aware of the video contents by texts, which include but not limit to titles, descriptions, reviews as well as meta information for the videos. Based on these texts, two kinds of textual content features are often extracted: word vectors and meta vectors. In our work, we construct one word vector and one meta vector for each video. Word vectors are extracted from the titles, descriptions and reviews. The process concatenates texts into single documents, removes stop words, and collects tokens by stemming [11]. After that, a number of top tokens of the highest TF-IDF values are selected to form the vocabulary. With the vocabulary, a word vector is created for each video with token frequencies. Meta vectors are extracted from official information about videos such as producers, countries, languages, release dates, actors, genres and so on. The process selects a number of top meta tokens according to the highest global frequencies to form the vocabulary. After that, a binary meta vector is created for each video according to token existences because a meta token for a video appears at most once.

3.2 Non-Textual Content Features

In addition to textual contents, videos themselves also contain non-textual contents. Yang et al. [13] extract the normalized color histogram and aural tempos as non-textual content features for videos. However, the experimental results reported in [13] show that these features are not effective for video recommendation. One possible reason is that these features fail to distinguish between videos that share similar colors but have irrelevant contents. For example, given a video about the sky and another video about the sea, the normalized color histogram will result in a high similarity between the two videos due to the common color blue. In this case, hybrid models would recommend sky-related videos to users who like seas in a high probability.

The limitations of the non-textual content features used by previous works [13], [15] are gradually broken through by recent findings in computer vision area [19], [20], [34]. Enlightened by this, we propose to extract diverse non-textual content features from videos in terms of audio, scenes, and motions to model the fact that users are attracted by videos in different ways [16]. We assume the informative non-textual features useful for personalised recommendation include audio (e.g., scary videos usually have similar sound effect), scene (e.g., the interstellar movies usually have similar image background) which is encoded in images, and motion (e.g., many romantic movies have the motion of "kiss") which is represented as series of sequential images. In particular, the non-textual content features MFCC [17], SIFT [35], [36], IDT [19], and CNN [18] that have achieved noticeable successes in video analysis tasks [17], [19], [20] recently are extracted to work with the hybrid models in our work. Their individual details are described as follows.

1. MFCC (mel-frequency cepstral coefficients). MFCC measure the audio changes in the sound track. We use MFCC to capture the audio contents within the videos. Our MFCC features are extracted as follows: 1) down-sampling the audio track of a video to 16 kHz with 16 bit resolution; 2) using a window size of 25 ms and a step size of 10 ms to

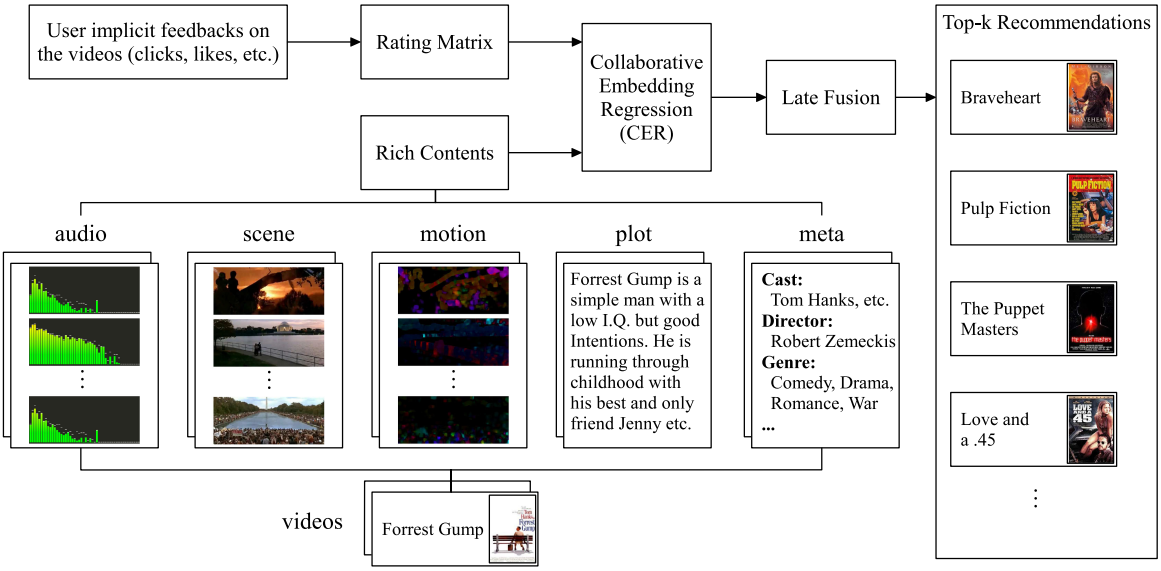


Fig. 2. A running example to illustrate how the rich content features are extracted from the videos and used with the user implicit ratings to generate the personalized video recommendation.

set the signal extractor with 13 channels; and 3) concatenating the first, second derivatives, and the energy of each signal to form a 40-dimension vector.

2. SIFT (scale invariant feature transform). SIFT [37] quantizes the texture information inside the images. In our study, we use two variants of SIFT to capture the scene and motion information in the videos respectively. They are OSIFT (opponent SIFT) [38] and MoSIFT (motion SIFT) [36]. OSIFT applies the light color change and shift on the original RGB color space to capture more robust scene contents in the video frames. An OSIFT feature has 384 dimensions. MoSIFT uses the optical flow between frames to capture the motion contents from in the videos. A MoSIFT feature has 256 dimensions.

3. IDT (improved dense trajectory). IDT [19] uses dense sampling and camera motions removing techniques to capture the motion contents in the videos. In previous studies [17], [20], IDT performs better than MoSIFT. An IDT feature has 426 dimensions.

4. CNN (convolutional neutral network). CNN uses the deep convolutional neural network and the large-scale labeled datasets to learn how the human classifies an image. One recent research [21] shows that using a pre-trained CNN on ImageNet [35] to extract features make the product recommendation have the visually-aware ability. Inspired by this, we use the pre-trained CNN model from the VGG group [18] to extract tensors from the pool_5 layer with spatial pooling [39]. Accordingly, each sampled frame has 49 CNN features with 512 dimensions.

Unlike MFCC, MoSIFT and IDT which take the whole audio or video file as input, OSIFT and CNN are only applicable for the images. Following [17], [20], we fetch 5 frames per second from the video. After all the raw features are extracted, we apply SSR (signed squared root) [17] to normalize these features for further processing.

Each video converts into a feature tensor after the extraction no matter which non-textual content is used. Nevertheless, the feature tensor is not the ideal format for the content-side learning in hybrid models [10], [11], [14], [21], so these tensors are encoded to vectors for model training. In particular, we apply two state-of-the-art encoding methods, fisher vector (FV)[34] and VLAD [40], to transform feature tensors to vectors based on the reported settings [17], [19], [20]. These vectors are of real values and suitable for linear models [34], [40]. The encoding methods and the resulting dimensions for individual non-textual content features are recorded in Table 1. We notice that encoded non-textual content features are of high dimensions, which makes them hard work with hybrid recommender models. Thus, principle component analysis (PCA) is applied to reduce the dimensions of encoded non-textual content features to 4000.

4 Video Recommendation

In this section, existing recommender models' performance in both in-matrix and out-of-matrix scenarios is first presented. Based on the performance comparison, we analyze why existing recommender models cannot deliver effective video recommendations with arbitrary content features. To address the problem, an improved recommender model, collaborative embedding regression, that can work with any single content feature is proposed. Based on CER, a priority-based late fusion method (PRI) is further developed. PRI prioritizes content features by validation accuracy and assigns enough large weights to the content features of high priorities, which can further improve recommendation accuracy in our later evaluations.

4.1 Existing Models on Arbitrary Content Features

Given rich content features, an interesting question naturally arises: how do state-of-the-art recommender models perform with arbitrary content features in top-k recommendations. To explore the results, we evaluate the in-matrix and out-of-matrix accuracy of representative WMF and BPR-based recommender models using the MovieLens 10M dataset [41]. In this preliminary evaluation, models are not only training with their specific content features but also with other ones if possible. The details about recommender models and content features are listed in Table 2.

The recommender models listed in Table 2 were tested in both in-matrix and out-of-matrix settings with their optimal settings according to the reports in [10], [11], [14], [21], [22], [23]. We use in-matrix and out-of-matrix accuracy as the

TABLE 1
The Dimensions of the Encoded Non-Textual Content Vectors

Feature	Encoder	Dimension
MFCC	FV	10240
OSIFT		98304
MoSIFT		68608
IDT		128304
CNN	VLAD	131072
		65536

TABLE 2
Recommender Models and Single Content Features in Reproduction

Model	Content Feature
WMF, BPR	N/A
CDL, VBPR, CTR	WORD, META
CDL, VBPR, DPM	MFCC
CDL, VBPR	CNNFV

coordinates to draw combinations of models and content features in Fig. 3. Specifically, we use the evaluation metric Accuracy@30 to plot the points. Additionally, in the legend of Fig. 3, the subscripts of models denote the in-use content features. Fig. 3 provides the following observations:

1) *WMF-based recommender models achieved the best performance in in-matrix scenario*. In Fig. 3, all the WMF-based models (i.e., WMF, CTR, DPM and CDL) are located to the right of the BPR-based models (i.e., BPR and VBPR). In addition, the in-matrix performance of WMF-based models (e.g., CDL) do not vary obviously with respect to different content features. All these facts indicate that WMF-based models are better than BPR-based models for video recommendation in in-matrix scenario.
2) *BPR-based recommender models achieved the best performance in out-of-matrix scenario*. Also in Fig. 3, given a particular content feature, the position of VBPR is always higher than that of all the WMF-based models. This shows that VBPR is currently the most effective model in out-of-matrix scenario, yet indicates the content-side learning inside existing WMF-based models are not effective for out-of-matrix video recommendation.

Our reproduction experiment shows that none of the existing recommender models achieve superior accuracy in both in-matrix and out-of-matrix scenarios even with their nominated content features. In the next section, we will propose a general recommender model, collaborative embedding regression, to address the issue.

4.2 Collaborative Embedding Regression

Existing WMF-based models [10], [11], [14] are designed for document or music recommendation tasks where single content features are often strong enough to support content-side learning. In particular, CTR [10] incorporates the word vectors with WMF and performs content-side learning by latent Dirichlet allocation (LDA). Since the optimization of LDA is based on word count, CTR naturally fails to support non-textual content features that are of real values. Compared to CTR, DPM [14] and CDL [11] perform content-side

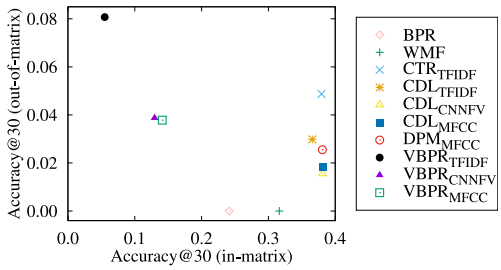


Fig. 3. Accuracy of existing hybrid recommender models in both in-matrix and out-of-matrix scenarios, where positions near the top right corner indicate better overall performance. To clearly display positions of the models that only support in-matrix recommendation, we shift the origin of the vertical axis to a higher position.

learning with MFCC vectors and word vectors respectively by multiple layer perception (MLP) and stacked de-nosing auto-encoder. However, their content-side learning aims at improving the in-matrix recommendation accuracy when ratings are sparse and implicitly requires the input feature vectors to be of non-negative values. As a result, DPM and CDL's out-of-matrix performance is poor according to Fig. 3 even with their specific content features. Compared to documents or music, videos are associated with multiple content features that none of them can solely support effective recommendation. In this situation, a general rather than specific content-side learning is necessary for hybrid models. VBPR's performance in out-of-matrix shows that linearly embedding content features in the content-side learning is general and powerful. Based on above analysis, we propose a general WMF-based recommender model, collaborative embedding regression, to achieve state-of-the-art in-matrix and out-of-matrix recommendation accuracy with arbitrary single content features.

Let d denote the dimension of the content vector and k denote the dimension of the latent vector. The whole generation process of CER with an individual content feature is described below.

1) For each user i, draw a user latent vector $w_i \in \mathcal{R}^{k\times 1}$:

$$w_i \sim \mathcal{N}(0, \lambda_u^{-1} I_k). \tag{1}$$

2) Generate an embedding matrix $E \sim \mathcal{N}(0, \lambda_e^{-1} I_k)$.
3) For each video j:
   a) Generate a content latent vector $h'_j \in \mathcal{R}^{k\times 1}$:

$$h'_j = E^T f_j. \tag{2}$$

   b) Draw a latent video offset vector $\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I)$, and then set the video latent vector as:

$$h_j = h'_j + \epsilon_j. \tag{3}$$

4) For each user-video pair (i, j), draw the rating:

$$r_{ij} \sim \mathcal{N}(w_i^T h_j, c_{ij}^{-1}). \tag{4}$$

where I_k is an identity matrix, $f_j \in \mathcal{R}^{d\times 1}$ is a feature vector, $E \in \mathcal{R}^{d\times k}$ is an embedding matrix, and c_{ij} is the confidence parameter for the user-item pair (i, j). Following [10], [11], the value of c_{ij} is defined below:

$$c_{ij} = \begin{cases} 1, & \text{if } r_{ij} = 1 \\ 0.01, & \text{if } r_{ij} = 0. \end{cases} \tag{5}$$

Note that, in step $3(a)$, we use linear embedding instead of non-linear learning adopted by CTR, DPM and CDL. This is more general for content-side learning with arbitrary content features from videos. In step $3(b)$, h'_j that embeds content features to latent vectors serves as the bridge between content-side and collaborative-side learning.

Learning the Parameters. The lowest mean absolute error (MAE) of rating prediction is achieved when the jointly posterior probability of W, H and E is maximized. However, directly computing the full posterior of parameters is intractable. As a result, CER is trained by minimizing the negative log-likelihood in this paper as follows:

$$\begin{aligned}\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{c_{ij}}{2}(w_i^T h_j - r_{ij})^2 + \frac{\lambda_u}{2}\sum_{i=1}^{m} w_i^T w_i \\ + \frac{\lambda_v}{2}\sum_{j=1}^{n}(h_j - E^T f_j)^T(h_j - E^T f_j) + \frac{\lambda_e}{2}||E||_F^2,\end{aligned} \tag{6}$$

where λ_u, λ_v and λ_e are the hyper parameters and $||\cdot||_F$ denotes the Frobenius norm. When hyper parameters are given, the optimal latent vectors w_i and h_j as well as the embedding matrix E are learned by performing the alternating least squares (ALS). Specifically, given the current estimation of E, we calculate the derivatives with respect to w_i and h_j, set them to zeros, and apply following updates for w_i and h_j in each iteration:

$$\begin{aligned} w_i &\leftarrow (HC_iH^T + \lambda_u I_k)^{-1} HC_iR_i \\ h_j &\leftarrow (WC_jW^T + \lambda_v I_k)^{-1}(WC_jR_j + \lambda_v E^T f_j),\end{aligned} \tag{7}$$

where $W = (w_i)_{i=1}^m \in \mathcal{R}^{k\times m}$ is the matrix concatenated by user latent vectors, $H = (h_j)_{j=1}^n \in \mathcal{R}^{k\times n}$ is the matrix concatenated by video latent vectors, and $F = (f_j)_{j=1}^n \in \mathcal{R}^{d\times n}$ is the content feature matrix. For user i, $C_i \in \mathcal{R}^{n\times n}$ is a diagonal matrix with c_{ij}, $j = 1\cdots, n$ as the diagonal elements, $R_i \in \{0,1\}^{n\times 1}$ is a vector with r_{ij}, $j = 1\cdots, n$ as its elements. For video j, C_j and R_j are similarly defined.

After W and H are updated, the derivatives with respect to E are computed and set to zero. Then, E is applied the following update:

$$E \leftarrow (\lambda_v FF^T + \lambda_e I_d)^{-1}(\lambda_v FH^T). \tag{8}$$

Similar to CTR and CDL, CER supports both in-matrix and out-of-matrix rating prediction. For in-matrix predictions, given a user-video pair (i, j), the rating $\hat{r}_{ij}$ is estimated as $w_i^T(E^T f_j + \epsilon_j)$. For out-of-matrix prediction, the rating $\hat{r}_{ij}$ is predicted as $w_i^T E^T f_j$ since no offset is observed. In summary, CER's rating predictor is defined as:

$$\hat{r}_{ij} = \begin{cases} w_i^T h_j, & \text{in-matrix setting} \\ w_i^T E^T f_j, & \text{out-of-matrix setting.}\end{cases} \tag{9}$$

4.3 Multiple Feature Fusion

CER model is designed to work with single content features as existing hybrid models [10], [11], [14], [21]. Based on CER, we further explore how to use multiple content features to improve the video recommendation accuracy. Specially, we discuss three feature fusion methods that possibly facilitate the video recommendation with multiple content features.

The first method concatenates multiple content feature vectors associated with same videos into single big vectors and then feeds the big vectors into CER for model training. Assuming there are L content features in total, the concatenation is denoted as follow:

$$f_j \leftarrow [f_j^1, f_j^2, \ldots, f_j^L]. \tag{10}$$

This fusion method is expected to learn the shared latent factors among the concatenated features. It does not introduce any modification on the objective function of CER, but it will significantly increase the training cost because the time complexity of CER's training is proportional to the dimension of the feature vector f_j.

The second method adds multiple content latent vectors $h_j^{\prime l}$ together, as done in CKE [12]. The content latent vectors in the generation process of CER are thus redefined as:

$$h'_j = \sum_{l=1}^{L} h_j^{\prime l} = \sum_{l=1}^{L} E^l f_j^l. \tag{11}$$

Compared to the first method, the second method compresses the dimension so that the training is faster, but it needs to modify the objective function of the CER by adding regularization terms of all the embedding matrices. In addition, updating formulas of model parameters need to change accordingly.

The first two methods are early fusion. They aim at mapping multiple content feature spaces to a shared homogeneous one. However, the textual, audio, visual and motion information contained in videos are quite diverse and heterogeneous. As shown in Fig. 3, the out-of-matrix accuracy of the best model varies greatly with respect to the different content features. This indicates that the construction of a shared latent space without losing some important and meaningful content patterns is usually unreasonable. Moreover, early fusion methods require re-training models when the deployed content features are changed (e.g., adding new ones). Based on above discussions, we think early fusion methods tend to be poor for video recommendation with multiple content features.

In a recent finding from other areas [17], [20], the performance divergence caused by heterogeneous content features can be addressed by late fusion, which fuses prediction scores from multiple content features to form more relevant ones. Inspired by this finding, we think late fusion has the potential to improve the video recommendation with multiple content features. We thus propose the third method to calculate fusion scores for videos. The score calculation is as follows:

$$\bar{r}_{ij} = \sum_{l=1}^{L} \pi_l \hat{r}_{ij}^l, \tag{12}$$

where L is the number of content features; π_l is the weight of the lth content feature; $\hat{r}_{ij}^l$ is the predicted rating based on the lth content feature. How to compute the weights is the major challenge of late fusion. A naive solution to compute weights, namely average fusion, is to treat each content feature equally. Recall that Fig. 3 shows significant performance divergences between different content features. Average fusion tends to neglects the divergences so as to lead to inferior fusion performance. Another solution is to learn the weights using a learning-to-rank method [28].

TABLE 3
Denotations of the Variables Used in Collaborative Embedding Regression (CER)

Denotation	Explanation
u_i	ith user's latent vector
h_j	jth video's latent vector
f_j	jth video's content vector
h'_j	jth video's content latent vector
ϵ_j	the offset vector between jth video's latent vector and content latent vector
r_{ij}	ith user's rating on jth video
c_{ij}	ith user's confidence on jth video
I_k	identify matrix with k dimensions
I_k^{-1}	the inverse of matrix I_k
E	matrix that embeds f_j to h'_j
λ	the hyper parameter for regularization
π_l	the weight of lth content feature in the late fusion
$\|\|\cdot\|\|_F$	Frobenius norm

However, as shown in our later experiments, learning-to-rank models cannot correctly quantize the influence of each content feature in the out-of-matrix recommendation.

In this paper, we propose a priority-based late fusion method (PRI) which aims at making the weights reflect higher influences from more effective content features. In details, PRI prioritizes content features based on the validation out-of-matrix performances and generates a feature ranking list from high to low, then iteratively assigns the weight of value $\pi_l = p(1-p)^{l-1}$ to lth content feature in the ranking list where $p \in [0.5, 1)$ is a hyper parameter. Note that, for any ranking position t (i.e., $\forall t > 0$), the inequality $\sum_{l=t+1}^{L} \pi_l \leqslant \pi_t$ holds in the related weights, which is not guaranteed in existing late fusion methods. This inequality ensures that the lth content feature always has higher weight than the total weight of the remaining less powerful content features. In other words, PRI allows more effective content features have more impacts in the fusion, which will be examined in our multiple feature fusion evaluation.

To clearly illustrate the computation of PRI's weights, we present an example in Table 4 where four content features are given and ranked. Note that, hybrid models (including our CER) with different content features achieve the similar recommendation performances in in-matrix scenario due to the absolute dominance of CF [10], so we only apply PRI in out-of-matrix scenario with multiple content features.

5 Experiments

In this section, we first introduce the datasets, the comparison models and methods, and the experimental settings in our evaluations. Then, we report the experimental results in terms of single content recommending and multiple content fusion. In addition, we present some important findings from our experiment that are emphasized by the bold font.

5.1 Dataset Description

We used the MovieLens 10M [41] and the Netflix 100M[1] as the base datasets for our empirical studies. We do not use the YouTube 8M[2] and the YouTube faces DB[3] because they do not provide user-video interactions. Both of MovieLens and Netflix do not contain videos or links for downloading, so we attempted to collect the videos from YouTube by ourselves. After several attempts, we finally downloaded the trailers instead of the full-length videos, because most of the full-length videos are not available for downloading due to copyright restrictions. We also manually checked whether the trailers were really for the original full-length videos. For those trailers that are mismatched, we used the available video clips on YouTube from the original videos instead. By these means, we collected 10380 videos of the 10682 movies for the MovieLens 10M dataset, and 4831 videos of the 17770 movies for the Netflix 100M. The fps of videos is 24. The minimum, maximum and mean lengths of videos in for different datasets are listed in Table 5. Videos' widths are resized to 240 pixels for accelerating the non-textual content feature extraction. Their heights were resized proportionally according to the aspect ratio.

TABLE 4
An Example of the Weights Used by PRI when p is Set to 0.5

Feature	META	CNNFV	IDT	MFCC
l	1	2	3	4
π_l	0.5	0.25	0.125	0.0625

For the MovieLens dataset, we use the movie IDs from IMDB[4] that the MovieLens dataset provides to collect the textual content of the videos. Specifically, we crawled the movie plots, actors, directors, companies, languages and genres for the collected videos. After data collecting and natural language processing, we first created a corpus where each document is made up of the corresponding movie title and plot. After that, the top 20000 words in the corpus were selected as the vocabulary according to the global TF-IDF values, following [10] and [11]. Given the vocabulary, a word vector for each movie was generated by counting the word frequencies. The remaining textual data include actors, directors, languages, companies, genres and other meta items. They form a meta vector that is binary for each movie. We wanted to compare the effectiveness of the textual features fairly, so we again selected top 20000 meta items to generate the meta vectors. The Netflix dataset has only the titles for the movies. We used these titles to crawled the same types of textual data from IMDB as done for the MovieLens dataset, and apply the same text feature extraction process on the collected data.

Given the collected video list, the ratings associated with the missing videos and the users that have no ratings for the collected videos were removed. We transformed the remaining ratings in both datasets into $\{0, 1\}$ to model the implicit feedback as [10], [11], [42]. Specifically, we changed the ratings with the highest value 5 to 1 and all the other ratings to 0. After that, we got a MovieLens implicit dataset that has 1,543,593 positive feedbacks and a Netflix implicit feedback dataset that has 13,016,825 positive feedbacks. In these two implicit feedback datasets, the positive feedbacks fill 0.21 and 0.56 percent of the implicit rating matrix

1. https://en.wikipedia.org/wiki/Netflix_Prize

2. https://research.google.com/youtube8m/index.html
3. https://www.cs.tau.ac.il/~wolf/ytfaces
4. http://www.imdb.com

TABLE 5
The Statistics of Video Lengths on Both Datasets

	MovieLens	Netflix
length_{min}	14.58s	14.90s
length_{max}	578.97s	370.73s
length_{mean}	127.59s	127.16s

respectively. To clearly show the changes between original and processed data, we summarize the important changes between them in Table 6. The comparison between the rating density and the positive density shows that the implicit feedback transformation makes the training positives even less, which has been widely observed in the past literatures [10], [11], [21]. We made the experimental data and code publicly available for reproducible purpose.[5]

5.2 Experimental Setup

5.2.1 Evaluation Metric

We adopt the evaluation metric Accuracy@k used in [29], [43], [44], [45] to evaluate the top-k video recommendation accuracy in our experiment. The Accuracy@k metric reveals the ratio between the number of the overall positives that are correctly predicted by the personalized recommendations and the number of the total ground truth positives, where a higher value means better performance. In previous works [10] and [11], the value of k was selected from $\{50, 100, 150, 200, 250, 300\}$, which was too large for a user to receive at once in a real-world video recommender system [4]. Therefore, k is selected from $\{5, 10, 15, 20, 25, 30\}$ in our evaluation. Given the value of k, the steps to calculate Accuracy@k value are as follows:

1. Inferring the users' ratings on their unrated videos by the latent vectors and the content vectors;
2. Generating a ranking list of the unrated videos for each user according to the descending order of the corresponding inferred ratings;
3. Selecting the top-k videos from the ranking list to form the recommendation for each user;
4. Counting the number of hits on the test set D_{test}: for each user-video pair (i, j) in the test set D_{test}, if video j appears in user i's top-k recommendation list, we have a hit (i.e., the ground truth video is recommended to the user), otherwise we have a miss;
5. Calculating the overall Accuracy@k by Eq. (13).

$$Accuracy@k = \frac{\#Hit@k}{|D_{test}|}, \tag{13}$$

where $\#Hit@k$ denotes the total number of hits in the test set, and $|D_{test}|$ is the number of all test cases.

We adopt 5-fold cross validation method. Thus, we report the mean Accuracy@k of each recommender model or method.

5.2.2 Data Preparation

Each dataset in our evaluation is divided into the training set, the in-matrix test set, and the out-of-matrix test set. We do the dataset division five times to measure the mean accuracy of each recommender model or method in our evaluation. We apply the following steps to make these sets occupying 60, 20, 20 percent of the total positive ratings respectively:

(a) Splitting the unique video ids in the dataset into five folds randomly and uniformly, then splitting the ratings into five folds according to the split video ids;
(b) Selecting the rating fold from (a) one by one as the out-of-matrix test set;
(c) Merging the rest four rating folds from (b), then randomly re-splitting them into four folds uniformly;
(d) Selecting one rating fold from (c) as the in-matrix test set;
(e) Merging the rest three folds from (d) as the training set for model training.

In order to tune the hyper parameters of each model, the training set was further divided: 90 percent of the training ratings were used to train the model and 10 percent of the training ratings were used for validation. The in-matrix test set shares the same video ids with the training set. It was used to evaluate the accuracy of collaborative filtering side in each model. The out-of-matrix test set has totally different video ids from the training set's. It was used to evaluate the effectiveness of each content feature and the accuracy of content side in each hybrid recommender model.

Based on the video titles we have, we construct the sparse text vectors (denoted as **ST**) for the videos to simulate the situation where limited texts are available such as the users' generated or uploaded videos.. The accuracy measured on ST will be used as the baseline to show how the non-textual contents improve the recommendation accuracy when the recommender system lacks enough textual contents.

We also design two rich content feature sets to work with the fusion methods. These rich content feature sets aim to explore the impact on accuracy when the textual contents are absent. The settings of these rich content feature sets are as follows:

- *Rich content feature set* contains all the content features we extract in this study. This set is used to examine whether a fusion method can fuse textual contents and non-textual contents properly.
- *Rich non-textual content feature set* contains all the non-textual content features we extract in this study. This set is used to examine whether a fusion method can use the non-textual contents to achieve the same performance as the textual contents.

5.2.3 Comparison Model and Method

We compare our proposed CER model with the six state-of-the-art recommender modelds. For fair comparison, the

5. https://github.com/domainxz/top-k-rec

TABLE 6
The Statistics of the MovieLens and the Netflix Datasets in Evaluation

Index	MovieLens		Netflix	
	original	processed	original	processed
#user	69,878	69,878	480,189	478,624
#video	10,682	10,380	17,770	4,831
#rating	10,000,054	9,988,676	100,480,507	62,714,775
rating density	1.34%	1.38%	1.18%	2.71%
#implicit positive	1,544,812	1,543,593	23,168,232	13,016,825
positive density	0.21%	0.21%	0.27%	0.56%

dimension of the latent vectors in all the models is set to 50. Below, we provide the details of the evaluated models as well as their hyper parameter settings in our experiment for the reproducible purpose.

Weighted Matrix Factorization [31] is a classical collaborative filtering model that works in in-matrix setting. In our experiment, WMF achieved its highest accuracy with $\lambda_u = 0.01, \lambda_v = 0.01$.

Collaborative Topic Regression [10] combines CF with textual content. CTR learns the content latent vectors from word vectors using LDA. In our experiment, CTR was trained with both word and meta vectors, and it achieved the highest accuracy with $\lambda_u = 0.1$, $\lambda_v = 10$.

DeepMusic (DPM) [14] originally combines CF with audio content. DPM uses MLP to learn the content latent vectors from the MFCC vectors. In our experiment, DPM was extended to work with all the content vectors in the rich set, and it achieved the highest accuracy with $\lambda_u = 0.1$ and $\lambda_v = 10$.

Collaborative Deep Learning [11] originally combines CF with textual content. CDL uses stack denoising auto-encoder to learn the content latent vectors from the word vectors. In fact, SDAE can process the non-textual contents by replacing the binary visible layer with Gaussian visible layer. We therefore modified CDL to work with both textual and non-textual content vectors in our experiment. Referring to [11], we chose the three-layer architecture for content side of CDL. The chosen architecture achieved its highest accuracy with $\lambda_u = 0.1$, $\lambda_v = 10$ and $\lambda_n = 1000$.

Bayesian Personalized Ranking (BPR) [23] uses pair-wise optimization to perform CF. Similar to WMF, BPR is only applicable in the in-matrix scenario. In our experiment, BPR achieved its highest accuracy with $\lambda_u = 0.0025$, $\lambda_i = 0.0025$, $\lambda_j = 0.00025$ and $\lambda_b = 0.0$.

Visual Bayesian Personalized Ranking [21] is an extension of BPR to combine CF with scene content. VBPR can work with any individual content feature. VBPR achieved its highest accuracy with $\lambda_u = 0.0025$, $\lambda_p = 0.0025$, $\lambda_i = 0.0025$, $\lambda_j = 0.00025$, $\lambda_b = 0.0$ and $\lambda_e = 0.0$.

Collaborative embedding regression is proposed in this paper to combine CF with any individual content feature. CER uses linear embedding to learn the content latent vectors from the content vectors. CER achieved its highest accuracy with $\lambda_u = 0.1$, $\lambda_v = 10$ and $\lambda_e = 1000$.

We also compare the proposed priority-based late fusion method with six widely used fusion methods. The descriptions of these fusion methods are as follows:

Accuracy fusion (ACC) is a late fusion method that uses the validation accuracy of the individual content features as the weights to calculate the fusion rating for the videos. We use ACC as a heuristic baseline to validate whether the exponential weights in our priority-based fusion method are effective.

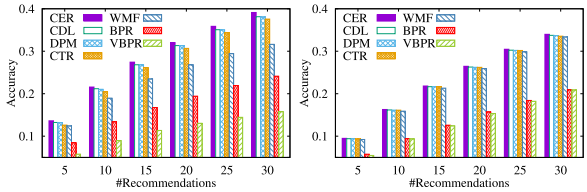
Average fusion (AVG) is a late fusion method that averages the predicted ratings from different content features.

Ranking SVM (SVM) [27] is a late fusion method that applies the point-wise learning-to-rank process.

Ranking BPR (BPR)[23] is a late fusion method that applies the pair-wise learning-to-rank process.

Early fusion by content vector concatenating (ECT) that has been applied in [21] is an early fusion method presented in Section 4.3. It concatenates all the feature vectors to learn the unified latent space by linear embedding.

Early fusion by latent content vector stacking (ESK) that has been applied in [12] is an early fusion method presented in Section 4.3. It adds all the content latent vectors in a element-wise way to learn the unified latent space.



(a) MovieLens (b) Netflix

Fig. 4. In-matrix accuracy comparison between different recommender models.

Priority-based fusion (PRI) is the late fusion method proposed in this paper. Like ACC, PRI obtains the accuracy of each content feature on the validation set first. Then, it prioritizes the content features and assign them the exponential weights like Table 4. We applied grid search on p to select its best value from $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. On the validation set, we found PRI achieved its best performance on both rich content feature sets with $p = 0.5$.

5.3 Evaluation on Individual Features

We evaluate the performances of recommender models on single content features in this study. Specifically, we compare CER with state-of-the-art recommender models in both in-matrix and out-of-matrix scenarios. The comparison validates that CER is a general and effective hybrid model to work with different content features.

5.3.1 In-Matrix Evaluation

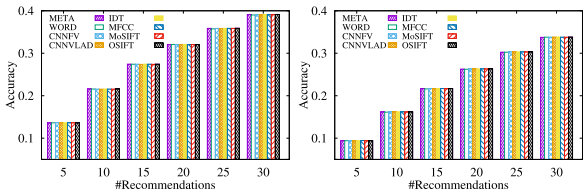
In the in-matrix evaluation, we study two problems as follows:

- How CER performs against hybrid models with single content features in in-matrix scenario?
- Whether CER can generate accurate recommendations with arbitrary single content features in in-matrix scenario?

We first compare CER with the other recommender models that can generate in-matrix recommendations. The accuracy results on MovieLens and Netflix are plotted in Fig. 4a and 4b respectively.

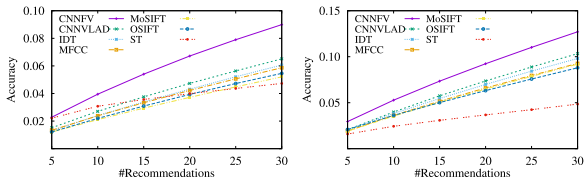
In Fig. 4a and 4b, we only present each hybrid recommender model's highest accuracy with single content features. We observe that CER has consistently achieved higher accuracy than the other models on both datasets, which shows that *CER is capable of generating accurate in-matrix recommendation as other hybrid models*. We also observe that the accuracy gaps between the BPR-based models and WMF-based models are significant, which again validates that the WMF-based models are more effective than the BPR-based models for video recommendation in the in-matrix scenario. Additionally, we notice the accuracy gaps between pure WMF and its variants (CTR, DPM, CDL, CER) are considerable. This observation indicates that the content-side is beneficial to the collaborative-side inside a hybrid model. The comparison also shows that different content-side learning methods inside the WMF-based models have marginal impacts on in-matrix accuracy.

We second explore where CER is able to generate accurate in-matrix recommendations with arbitrary single content features. The performances of CER with different content features are depicted in Fig. 5a and 5b.



(a) MovieLens (b) Netflix

Fig. 5. In-matrix accuracy comparison among different features.



(a) MovieLens (b) Netflix

Fig. 6. Out-of-matrix accuracy comparison in the sparse-text scenario.

From Fig. 5a and 5b, we observe that different content features have similar individual impacts to in-matrix accuracy except WORD vectors. The results show that CER can generate accurate in-matrix recommendations with arbitrary single content features. Considering WORD vectors have been widely used by existing works [10], [11], the results further show that *there is almost no performance drop of changing content features inside CER in in-matrix scenario*.

5.3.2 Out-of-Matrix Evaluation

In the out-of-matrix evaluation, we study two problems as follows:

- How CER performs against other hybrid models with single content features in out-of-matrix scenario?
- Whether CER can generate accurate recommendations with arbitrary single content features in out-of-matrix scenario?

Specifically, we measure the out-of-matrix accuracy of CER, CDL, VBPR, DPM, and CTR on MovieLens (Fig. 7) and Netflix (Fig. 8) with all the single content features. To compare these models clearly, each sub-figure in Figs. 7 and 8 draws the accuracy of different models with one specific content feature, and the sub-figures from the same dataset are sorted by the Accuracy@30 of CER in a descending order.

From Figs. 7 and 8, we observe that CER consistently outperforms the other hybrid models on both datasets with any content feature, which validates that *CER is better than other hybrid models with any single content feature in out-of-matrix recommendation*. The comparison between CER and the other WMF-based models shows that linearly embedding the content features to the content latent vectors seems to be more general and effective than the existing content-side learning in existing hybrid models. We also observe that VBPR that also employs linear embedding achieves worst performances on Netflix with textual content features. The major difference between CER and VBPR is that CER is based on WMF while VBPR is based on BPR in the collaborative-side learning, we thus think the effectiveness of linear embedding holds only if it combines with WMF.

Another observation from Figs. 7 and 8 is about the different impacts of single content features in CER's out-of-matrix recommendation. In details, the descending order of content features ranked by the Accuracy@30 on MovieLens dataset is $META > WORD > CNNFV > CNNVLAD > IDT > MFCC > OSIFT > MoSIFT$, while the order on Netflix dataset is $META > CNNFV > WORD > CNNVLAD > IDT > MFCC > MoSIFT > OSIFT$. According to these comparison chains above, META vectors are the most effective content feature for video recommendation. This is because META vectors contain a lot of precise information that do not often appear in the video clips directly such as casts, producers and published years, which are more relevant with users' preferences on the videos. However, the META vectors are unavailable for most of the videos on a general video website like Youtube, therefore the accuracy of META vectors is only the ideal accuracy that CER could achieve. In practice, WORD vectors are more easily obtainable as they are made up of the video titles and descriptions. We observe that the recommendation accuracy with WORD vectors is a little higher than the accuracy with CNNFV vectors on MovieLens, yet a little lower on Netflix. Considering the video titles and descriptions we collected exactly reflect the video contents, the comparison indicates that, with the generality of CER, the non-textual content features could be the effective alternatives to textual features in common cases.

To further examine the effectiveness of the non-textual part in our rich content features, we design the sparse-text WORD vectors (denoted as ST) that use only the video titles to perform the out-of-matrix recommendation again. The design bases on the fact that many user uploaded videos on Youtube with the titles only. We evaluate the accuracy of ST using CER and compare its accuracy with the non-textual content features in Fig. 6. The comparison shows that the CNNFV vectors significantly outperform the ST vectors on both datasets, while the other non-textual contents significantly outperform ST only on Netflix dataset. Putting the observations from Figs. 6, 7, and 8 together validates that, *CER is more general and effective than existing hybrid models in out-of-matrix scenario, although its accuracy varies significantly with different content features*.

5.3.3 Efficiency of Model Training

We also examine the time cost of training each recommender model. This experiment is conducted on two Intel E5-2690 v2 CPUs, one Nvidia Geforce 1070, and 256GB memory. All the WMF-based models are trained with CPU configuration, while all the BPR-based models are tested with GPU configuration (due to their slow training speeds in CPU configuration). We collect the time cost per iteration in the training stage of each model. Specifically, in WMF-based models, one iteration equals to update all the user latent vectors, all the video latent vectors, and the content side (if has) once. In the BPR-based models, one iteration needs to process all the training positives once. In addition, considering the textual content vectors are usually sparse and the non-textual content vectors are usually dense, we evaluate the time cost per iteration with both WORD vectors and CNNFV vectors.

Table 7 records the time cost of different models on both datasets. From Table 7, without the content-side learning, the BPR model costs less time per iteration than the WMF model on Movielens, while more time per iteration than WMF on Netflix. The reason behind this is that BPR's complexity is proportional to the number of ratings and WMF's complexity is proportional to the number of the users plus the number of the videos. When the training

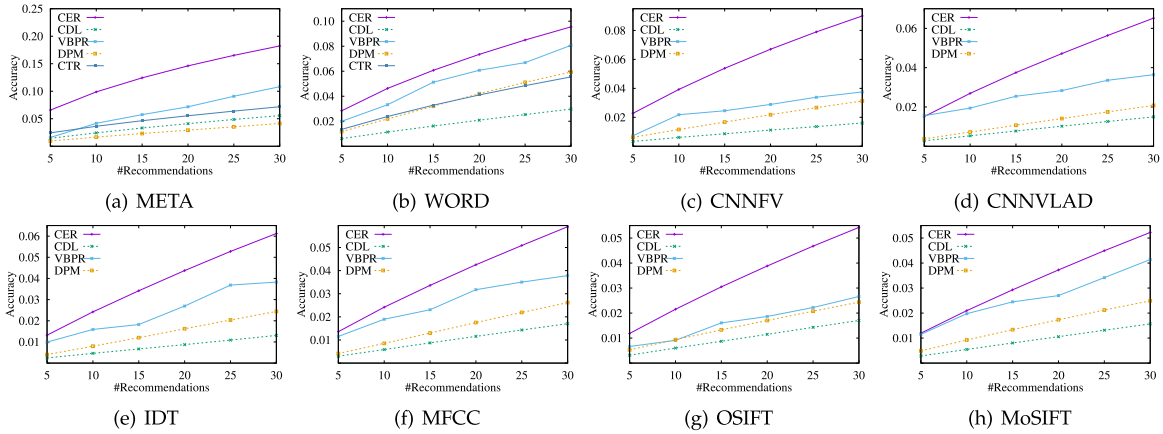


Fig. 7. Accuracy@k of different recommender models and features in out-of-matrix setting on MovieLens implicit dataset.

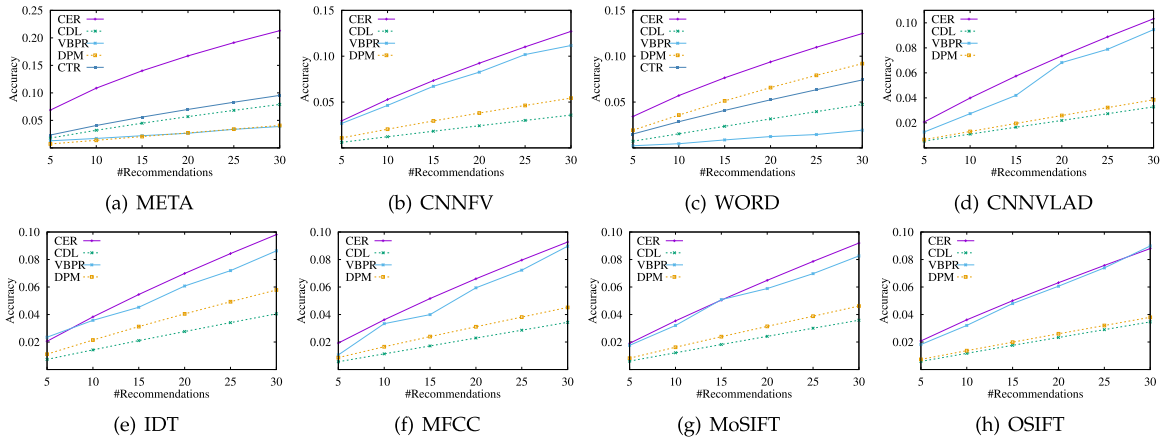


Fig. 8. Accuracy@k of different recommender methods and features in out-of-matrix setting on Netflix implicit dataset.

shifts from Movielens to Netflix, the number of training ratings increases faster than the total number of the users and the videos, which causes BPR to use more time on Netflix accordingly. Taking the content-side learning into consideration, CER achieves the lowest time cost per iteration in most cases. CER is more efficient than the other WMF-based models due to its lightweight content-side learning. In addition, CER is more efficient than VBPR in most cases although VBPR has GPU acceleration. This is because VBPR applies pair-wise learning that updates the video vectors double times compared to only once in the WMF-based models in each iteration.

5.4 Evaluation on Multiple Feature Fusion

In this experiment, we explore whether the proposed priority-based late fusion method (PRI) can make the recommendation more accurate with multiple content features. Our fusion evaluation is based on CER, because it is the most accurate method in our single content evaluation. We prepare the rich content features and the rich non-textual content features for CER, and apply different fusion methods with CER's model or outputs. The accuracy values are measured with out-of-matrix scenario where multiple feature fusion should be effective according to the existing works [17], [25]. Tables 8 and 9 record the accuracy results of different fusion methods on MovieLens and Netflix respectively. In these tables, we use the accuracy results from the best single content feature in the corresponding rich content features as the baseline to help judge whether a fusion method improves the accuracy actually. In addition, we highlight the results of the proposed late fusion method (PRI) with the bold font.

From Tables 8 and 9, we first check the out-of-matrix accuracy results of the early fusion methods. We observe that ECT fusion's accuracy significantly drops when it is applied on the rich content features. This observation shows that concatenating the heterogeneous content vectors together to learn the shared space harms the recommendation accuracy in the video recommendation scenario. Compared to ECT, ESK that adds the content latent vectors in the shared space achieves consistent recommendation accuracy in out-of-matrix scenario, and beat ACG, BPR and

TABLE 7
The Time Cost per Iteration in Terms of Seconds from Each Recommender Model with Two Content Features on Two Datasets

Content	CNNFV		WORD	
Dataset	MovieLens	Netflix	MovieLens	Netflix
CTR	N/A		15.46s	76.43s
DPM	28.02s	69.21s	73.73s	94.32s
CDL	45.37s	82.43s	175.82s	141.10s
VBPR	10.50s	136.28s	18.46s	204.82s
CER	11.00s	67.10s	14.62s	70.02s
WMF	MovieLens	8.52s	Netflix	68.75s
BPR	MovieLens	8.10s	Netflix	116.08s

TABLE 8
Fusion Accuracy of Different Methods on MovieLens Implicit Dataset with the Rich Content Feature Set and the Rich Non-Textual Content Feature Set

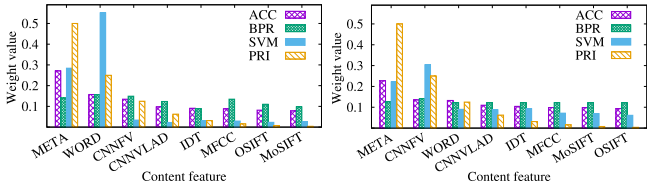
Fusion on the rich non-textual content feature set						
Method	Accuracy@5	Accuracy@10	Accuracy@15	Accuracy@20	Accuracy@25	Accuracy@30
ACC	0.022121	0.038951	0.053396	0.066657	0.078836	0.090326
AVG	0.021243	0.037600	0.051699	0.064683	0.076871	0.088040
BPR	0.021241	0.037602	0.051707	0.064690	0.076864	0.088044
SVM	0.021498	0.037748	0.051894	0.064918	0.077049	0.088096
ECT	0.021830	0.038222	0.051606	0.065720	0.077664	0.089460
ESK	0.021439	0.037826	0.051994	0.064998	0.077119	0.088350
PRI (p=0.5)	**0.023090**	**0.040390**	**0.055746**	**0.069401**	**0.081788**	**0.093239**
CNNFV	0.022765	0.039240	0.053838	0.067042	0.078984	0.089915
Fusion on the rich content feature set						
ACC	0.064621	0.098163	0.124392	0.146985	0.166617	0.184319
AVG	0.063132	0.093990	0.118933	0.140554	0.159595	0.176996
BPR	0.061949	0.092322	0.116883	0.138128	0.156966	0.174059
SVM	0.067023	0.100991	0.127185	0.149112	0.168508	0.186059
ECT	0.041733	0.063908	0.081737	0.097422	0.111366	0.124557
ESK	0.068546	0.101244	0.127280	0.149542	0.169221	0.187168
PRI (p=0.5)	**0.070307**	**0.105696**	**0.132879**	**0.156019**	**0.176204**	**0.194314**
META	0.065780	0.098771	0.124305	0.146079	0.165213	0.182318

TABLE 9
Fusion Accuracy of Different Methods on Netflix Implicit Dataset with the Rich Content Feature Set and the Rich Non-Textual Content Feature Set

Fusion on the rich non-textual content feature set						
Method	Accuracy@5	Accuracy@10	Accuracy@15	Accuracy@20	Accuracy@25	Accuracy@30
ACC	0.029256	0.052103	0.072838	0.092046	0.110091	0.127141
AVG	0.028596	0.051106	0.071609	0.090616	0.108466	0.125376
BPR	0.028590	0.051106	0.071585	0.090590	0.108422	0.125329
SVM	0.028556	0.050811	0.070763	0.089278	0.106692	0.123241
ECT	0.029195	0.052478	0.073142	0.091845	0.100103	0.127278
ESK	0.028517	0.050983	0.071445	0.090408	0.108284	0.125155
PRI (p=0.5)	**0.030394**	**0.054099**	**0.075226**	**0.094740**	**0.112876**	**0.130110**
CNNFV	0.029482	0.052753	0.073430	0.092353	0.110160	0.127079
Fusion on the rich content feature set						
ACC	0.072106	0.114397	0.148046	0.176939	0.202550	0.225676
AVG	0.065187	0.104734	0.136801	0.164413	0.189063	0.211580
BPR	0.063911	0.102926	0.134620	0.162095	0.186601	0.208990
SVM	0.052362	0.084992	0.112074	0.135954	0.157501	0.177489
ECT	0.039491	0.065230	0.087030	0.106488	0.124300	0.140920
ESK	0.066386	0.106135	0.138135	0.165710	0.190251	0.212697
PRI (p=0.5)	**0.073158**	**0.115894**	**0.149731**	**0.178567**	**0.204068**	**0.227094**
META	0.068739	0.108569	0.140139	0.167226	0.191242	0.213063

SVM most of the time. However, both of the early fusion methods fail to beat the baseline all the time in our evaluation. These comparisons validate that the heterogeneous content vectors are hard to form more meaningful shared space in the video recommendation scenario.

The early fusion methods' robustness and effectiveness problems also exist in the late fusion methods. In details, AVG, BPR and SVM only beat the baseline once when they are applied with rich content features on MovieLens. The possible failure of AVG is due to the huge performance gaps among different content features. In Figs. 7 and 8, the highest out-of-matrix accuracy of CER is achieved with META vectors, while the lowest accuracy of CER is achieved with MoSIFT vectors. The highest is three times to the lowest. As a result, average fusion (AVG) weakens the predictability of more powerful content vectors. Both BPR and SVM are learning-to-rank methods. They learn content weights by regarding the predicted ratings from contents as the feature vectors. Nevertheless, these existing learning-to-rank methods cannot learn the content priorities properly in our evaluation. The evidences are shown in Fig. 9 where we draw weight distributions over all the content features of each late fusion method after normalising the weight sum to 1. In Fig. 9, if we regard the weights from ACC as the correct ones, those from BPR and SVM are wrong: BPR treats different contents almost equally, while SVM cannot correctly lift the weight of META over that of WORD according to their significant highest out-of-matrix accuracy differences. Eventually, they fail to generate more accurate recommendations. Despite PRI, our heuristic late fusion method—ACC achieves the largest improvement in Tables 8 and 9. It always beats the baseline, which initially shows the content priorities in late fusion is necessary.



(a) MovieLens (b) Netflix

Fig. 9. Weight distributions from different late fusion methods.

The proposed PRI achieves the highest accuracy in our evaluation, and it beats the baseline on both datasets all the time. Compared to all the other fusion methods, we think the success of PRI has two reasons. First, PRI prioritizes content features as ACC does. This makes PRI beats the other comparison methods as ACC does. Second, PRI does not simply trusts the content features by their accuracy values as ACC does. Accordingly, as Fig. 9 shows, PRI assigns larger weights to the more powerful content features to protect their priorities in the fusion. This makes PRI further beats ACC. It is worth noting that PRI's improvement only happens on CER. For other models, because the absolute performance divergences are small, PRI could not improve the accuracy further even though rich content set presents (due to page limitation, we do not put PRI's results with other models in this paper). The possible reason is that CER's large absolute performance divergences between different features indicate correct feature priorities in out-matrix scenario. In summary, our evaluation validates that *PRI can improve the video recommendation accuracy further when large absolute performance divergences between individual features exist*.

6 Conclusion

In this paper, we explored how to exploit rich content features from videos to improve personalized recommendation accuracy in both in-matrix and out-of-matrix scenarios. We assumed that the rich content features can improve the hybrid recommender methods in two situations: (1) when single specific content features are unavailable; (2) when multiple content features are available. To validate our assumptions, we first extracted multiple textual and non-textual content features from videos. Our initial evaluation showed that existing hybrid models were only effective in either in-matrix or out-of-matrix scenario with rich content features. To overcome the limitation, we proposed a general and effective hybird model, namely collaborative embedding regression, to marginalize the performance drop caused by the unavailability of one specific content. In addition, we studied how to fuse multiple heterogeneous content features to improve the recommendation accuracy further. We proposed a priority-based late fusion method (PRI) to effectively fuse both non-textual and textual content features. We conducted extensive evaluations on single content features to validate the effectiveness of CER, and evaluations on multiple content features to validate the effectiveness of PRI. The experimental results on MovieLens and Netflix datasets showed that the proposed CER and PRI are superior to existing hybrid recommender models and multiple feature fusion methods respectively. In particular, CER has more significant impact on the recommendation accuracy and can work well with existing late fusion methods. PRI must work with the models which has quite diverse performance on different content features. In summary, all the experiments validated our initial assumptions and revealed the benefits of the rich content features for personalized video recommendation.

Acknowledgments

This work was supported by the ARC Discovery Early Career Researcher Award (DE160100308) and ARC Discovery Project (DP170103954; DP190101985).

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] F. Ricci, L. Rokach, and B. Shapira, Eds., *Recommender Systems Handbook*. New York, NY, USA: Springer, 2015. [Online]. Available: http://dx.doi.org/10.1007/978-1-4899-7637-6

[3] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. V. Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, and D. Sampath, "The youtube video recommendation system," in *Proc. 4th ACM Conf. Recommender Syst.*, 2010, pp. 293–296.

[4] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manage. Inf. Syst.*, vol. 6, no. 4, 2015, Art. no. 13.

[5] X. Qian, H. Feng, G. Zhao, and T. Mei, "Personalized recommendation combining user interest and social circle," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 7, pp. 1763–1777, Jul. 2014.

[6] J. Bu, X. Shen, B. Xu, C. Chen, X. He, and D. Cai, "Improving collaborative recommendation via user-item subgroups," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2363–2375, Sep. 2016.

[7] W. Wang, H. Yin, L. Chen, Y. Sun, S. W. Sadiq, and X. Zhou, "ST-SAGE: A spatial-temporal sparse additive generative model for spatial item recommendation," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 3, pp. 48:1–48:25, 2017.

[8] D. Agarwal and B.-C. Chen, "Regression-based latent factor models," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 19–28.

[9] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu, "Personalized recommendation via cross-domain triadic factorization," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 595–606.

[10] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 448–456.

[11] H. Wang, N. Wang, and D. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1235–1244.

[12] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 353–362.

[13] B. Yang, T. Mei, X. Hua, L. Yang, S. Yang, and M. Li, "Online video recommendation based on multimodal fusion and relevance feedback," in *Proc. 6th ACM Int. Conf. Image Video Retrieval*, 2007, pp. 73–80.

[14] A. V. D. Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2643–2651.

[15] Y. Deldjoo, M. Elahi, P. Cremonesi, F. Garzotto, P. Piazzolla, and M. Quadrana, "Content-based video recommendation system based on stylistic visual features," *J. Data Semantics*, vol. 5, no. 2, pp. 99–113, 2016.

[16] W. Hoiles, A. Aprem, and V. Krishnamurthy, "Engagement and popularity dynamics of youtube videos and sensitivity to meta-data," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 7, pp. 1426–1437, Jul. 2017.

[17] R. Aly, R. Arandjelovic, K. Chatfield, M. Douze, B. Fernando, Z. Harchaoui, K. Mcguiness, N. O'Connor, D. Oneata, et al., "The axes submissions at trecvid 2013," presented at the *TRECVID Workshop*, Gaithersburg, MD, USA, 2013.

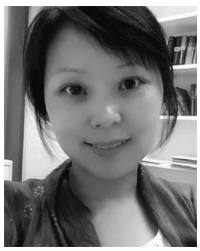
[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," CoRR, vol. abs/1409.1556, 2014.

[19] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3551–3558.

[20] Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative CNN video representation for event detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1798–1807.
[21] R. He and J. McAuley, "VBPR: Visual bayesian personalized ranking from implicit feedback," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 144–150.
[22] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE Int. Conf. Data Mining*, 2008, pp. 263–272.
[23] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.
[24] B. Cui, A. K. Tung, C. Zhang, and Z. Zhao, "Multiple feature fusion for social media applications," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2010, pp. 435–446.
[25] N. Srivastava and R. Salakhutdinov, "Learning representations for multimodal data with deep belief nets," in *Proc. Int. Conf. Mach. Learn. Workshop*, vol. 79, 2012.
[26] W. Wang, B. C. Ooi, X. Yang, D. Zhang, and Y. Zhuang, "Effective multi-modal retrieval based on stacked auto-encoders," *Proc. VLDB Endowment*, vol. 7, no. 8, pp. 649–660, Apr. 2014.
[27] T. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.
[28] Z. Cao, T. Qin, T. Liu, M. Tsai, and H. Li, "Learning to rank: From pairwise approach to listwise approach," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 129–136.
[29] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proc. 4th ACM Conf. Recommender Syst.*, 2010, pp. 39–46.
[30] W. X. Zhao, S. Li, Y. He, E. Y. Chang, J. Wen, and X. Li, "Connecting social media to e-commerce: Cold-start product recommendation using microblogging information," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1147–1159, May 2016.
[31] Y. Koren, R. M. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[32] S. Roy and S. C. Guntuku, "Latent factor representations for cold-start video recommendation," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 99–106.
[33] Q. Zhu, M. Shyu, and H. Wang, "Videotopic: Content-based video recommendation using a topic model," in *Proc. IEEE Int. Symp. Multimedia*, 2013, pp. 219–222.
[34] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 143–156.
[35] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[36] M.-Y. Chen and A. Hauptmann, "Mosift: Recognizing human actions in surveillance videos," 2009.
[37] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[38] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1582–1596, Sep. 2010.
[39] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.
[40] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1704–1716, Sep. 2012.
[41] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interactive Intell. Syst.*, vol. 5, no. 4, 2016, Art. no. 19.
[42] J. Chen, C. Wang, J. Wang, and P. S. Yu, "Recommendation for repeat consumption from user implicit feedback," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 11, pp. 3083–3097, Nov. 2016.
[43] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, "Challenging the long tail recommendation," *Proc. VLDB Endowment*, vol. 5, no. 9, pp. 896–907, 2012.
[44] W. Wang, H. Yin, S. W. Sadiq, L. Chen, M. Xie, and X. Zhou, "SPORE: A sequential personalized spatial item recommender system," in *Proc. IEEE 32nd Int. Conf. Data Eng.*, 2016, pp. 954–965.
[45] W. Wang, H. Yin, Z. Huang, Q. Wang, X. Du, and Q. V. H. Nguyen, "Streaming ranking based recommender systems," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 525–534.

Xingzhong Du received the PhD degree from the University of Queensland. His research interests include data mining and recommender system.

Hongzhi Yin received the PhD degree in computer science from Peking University, in 2014. He is a senior lecturer with the University of Queensland. He received the Australia Research Council Discovery Early-Career Researcher Award, in 2015. His research interests include recommendation system, user profiling, topic models, deep learning, social media mining, and location-based services.



Ling Chen received the PhD degree in computer engineering from Nanyang Technological University Singapore, in 2008 . She is a senior lecturer with the UTS Priority Research Centre for Artificial Intelligence (CAI) and the Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney (UTS). She is currently the director of the Data Science & Knowledge Discovery Laboratory (The DSKD Lab).

Yang Wang received the PhD degree from the School of Computer Science and Engineering, the University of New South Wales (UNSW), Kensington, Australia, in 2015. He is currently a postdoctoral research fellow with UNSW. During the PhD journey, he has published more than 20 research papers, most of which are published at competitive venues, including ACM Multimedia, ACM SIGIR, ACM CIKM, IEEE-ICDM, the *IEEE Transactions on Neural Networks and Learning Systems*, the *IEEE Transactions on Image Processing*, and the *Knowledge and Information Systems*.

Yi Yang received the PhD degree in computer science from Zhejiang University, in 2010. He was a post-doctoral researcher with the School of Computer Science, Carnegie Mellon University. He is currently a professor with the University of Technology Sydney. His current research interests include machine learning and its applications to multimedia content analysis and computer vision, such as multimedia indexing and retrieval, surveillance video analysis, and video content understanding.

Xiaofang Zhou is a professor of computer science with the University of Queensland. He is the head of the Data and Knowledge Engineering Research Division, School of Information Technology and Electrical Engineering. He is the director of the ARC Research Network in Enterprise Information Infrastructure (EII), and a chief investigator of the ARC Centre of Excellence in Bioinformatics. He has been a fellow of the IEEE since 2017.