JointRec: A Deep-Learning-Based Joint Cloud Video Recommendation Framework for Mobile IoT

Sijing Duan, Deyu Zhang, *Member, IEEE*, Yanbo Wang, Lingxiang Li, *Member, IEEE*, and Yaoxue Zhang, *Senior Member, IEEE*

***Abstract*—In the era of Internet of Things (IoT), watching videos on mobile devices has been a popular application in our daily life. How to recommend videos to users is one of the most concerned problem for Internet video service providers (IVSPs). In order to provide better recommendation service to users, they deploy cloud servers in a geo-distributed manner. Each server is responsible for analyzing a local area of user data. Therefore, these cloud servers form information islands and the characteristics of data present nonindependent and identically distribution (non-i.i.d). In this scenario, it is difficult to provide accurate video recommendation service to the minority of users in each area. To tackle this issue, we propose JointRec, a deep learning-based joint cloud video recommendation framework. JointRec integrates the JointCloud architecture into mobile IoT and achieves federated training among distributed cloud servers. Specifically, we first design a dual-convolutional probabilistic matrix factorization (Dual-CPMF) model to conduct video recommendation. Based on this model, each cloud can recommend videos by exploiting the user's profiles and description of videos that users rate, thereby providing more accurate video recommendation services. Then, we present a federated recommendation algorithm which enables each cloud to share their weights and train a model cooperatively. Furthermore, considering the heavy communication costs in the process of federated training, we combine low-rank matrix factorization and 8-bit quantization method to reduce uplink communication costs and network bandwidth. We validate the proposed approach on the real-world data set, and the experimental results indicate the effectiveness of our proposed approach.**

***Index Terms*—Deep learning, federated training, JointCloud computing, mobile Internet of Things (IoT), nonindependent and identically distribution (non-i.i.d) data setting, video recommendation system.**

Manuscript received July 13, 2019; revised August 25, 2019; accepted September 24, 2019. Date of publication October 1, 2019; date of current version March 12, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61702561 and Grant 61702562, in part by the 111 Project under Grant B18059, in part by NSF under Grant ECCS-1554576, in part by the Innovation-Driven Project of Central South University under Grant 2016CXS013, in part by the International Science and Technology Cooperation Program of China under Grant 2013DFB10070, and in part by the China Hunan Provincial Science and Technology Program under Grant 2012GK4106. *(Corresponding author: Deyu Zhang.)*

The authors are with the School of Computer Science and Engineering, Central South University, Changsha 410083, China (e-mail: dsjyfd012@csu.edu.cn; zdy876@csu.edu.cn; wangyb@csu.edu.cn; lingxiang.li@csu.edu.cn; zyx@csu.edu.cn).

Digital Object Identifier 10.1109/JIOT.2019.2944889

I. Introduction

NOWADAYS, the recommendation system becomes an effective way to deal with information overhead, especially in the ever-growing multimedia applications [1]–[6]. As one kind of online multimedia applications, video services have received great attention from Internet video service providers (IVSPs). They aim at providing the most relevant video to target audiences, e.g., Youtube, Netflix,[1] IMDB,[2] Movielens,[3] and iQIYI [7], [8]. In the era of Internet of Things (IoT), watching videos on mobile devices has been a popular application in our daily life. In order to collect user data, the IVSPs deploy distributed cloud servers in different places [9], [10]. Each server is responsible for storing and analyzing the data generated by users who are located in specific areas [11], [12]; the user data from different places are nonindependent and identically distribution (non-i.i.d). Therefore, it is hard to recommend videos to the minority of users precisely. For example, as shown in Fig. 1, there are four cloud servers distributed in different areas. Bob is a ten-year-old child who lives in Area A, where the proportion of adults accounts for the majority. These users prefer the entertainment videos and soap operas. Peter is an old man who lives in Area C; the major users in this area are teenagers, and their interests tend to educational videos. Due to the difference of user distribution in these two areas, the characteristics of data produced by them are quite different, and forming information islands between them. If each server makes video recommendation decisions merely based on its local data, it will bring information deviation, leading to nonprecise recommendation. Furthermore, given the distributed nature of this scenario, the centralized method has the following inherent weaknesses.

1) Single point of failure. The centralized server may fail due to attacks, which threatens the reliability of the system.
2) All data are required to send to the centralized server and process centrally, leading heavy cost and communication overhead for centralized system [13].
3) Considering the difference of data distribution, the centralized recommendation results cannot be representative of the preference of single local areas.

[1]https://en.wikipedia.org/wiki/Netflix
[2]https://en.wikipedia.org/wiki/IMDb
[3]http://grouplens.org/datasets/movielens/

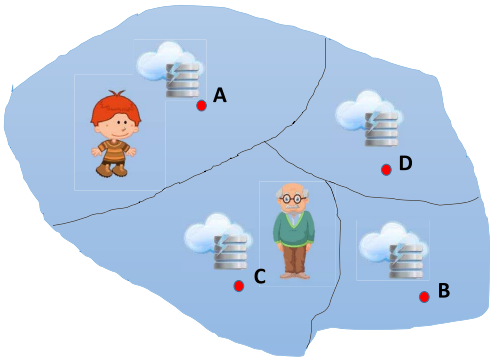


Fig. 1. Distributed video recommendation under non-IID scenario.

Therefore, a distributed strategy is more suitable for such a scenario.

Recently, the emerging form of "shared global economy" has required cloud resources to be collaboratively exploited by the distributed cloud providers in a geo-distributed manner [14]. Wang *et al.* [14] proposed JointCloud, a cross-cloud cooperation architecture for integrated Internet service customization. With this architecture, the distributed cloud servers are able to cooperate with each other to provide video recommendation service for the minority of users in each areas. On the other hand, from the perspective of communication, it is a challenge to provide efficient communication between distributed clouds [15]–[17].

In this article, our motivation is to provide more accurate video recommendation services to the minority of users under the above-mentioned scenario and achieve efficient communication. To reach this goal, several challenges must be addressed.

1) *How to Recommend Videos When the Data Distribution Is Non-i.i.d?* There have been many researches on video recommendation, hardly any studies focus on the impact of non-i.i.d data distribution on video recommendation. Due to the data distribution of watching videos in a specific area depending on the local users, any local data will not be representative for the global distribution. Therefore, the features of user data in different places are non-i.i.d. How to recommend videos under non-i.i.d scenario is a challenge.
2) *How to Design Recommendation Model to Extract Features From Nonlinear User-Video Data Under Non-i.i.d Scenario?* In the non-i.i.d case, user attributes (e.g., age, gender, and occupation) are the key data for the division of different types of users. It requires the recommendation algorithm to be able to adapt to the user attributes. Furthermore, there is much implicit information including in various nonlinear user-video feedback data, e.g., ratings and reviews [18]. It is necessary to design a recommendation model to extract deep level features from these data.
3) *How to Realize Cooperation Among Distributed Clouds?* Each cloud is in charge of collecting data from the areas it covers, leading to information islands among these clouds. In this case, the video recommendation results based on the local data performed by the single cloud will bring information deviations and inaccurate recommendation results for the minority of users. Therefore, how to enable the cooperation of distributed clouds, realizing information fusion, is another challenge remaining to be solved.
4) *How to Achieve Efficient Communication and Reduce Network Cost?* Distributed training requires significant communication bandwidth for frequent information exchange. Hence, how to reduce communication cost is another challenge.

In this article, we adopt the JointCloud architecture in mobile IoT, to recommend videos to the minority of mobile users when the data are non-i.i.d in the distributed scenario. To improve the performance of recommendation, we propose a dual-convolutional probabilistic matrix factorization (Dual-CPMF) model for video recommendation. This model characterizes the latent preferences of users by exploiting the user's profiles and textual information of videos that users have rated. Furthermore, we present a federated recommendation (FR) algorithm to enable information fusion among distributed clouds. Finally, we also propose an efficient communication strategy to reduce communication overheads and maintain the performance of the system. In summary, the main contributions of this article are as follows.

1) We propose to model user profiles and video properties by using convolutional neural networks (CNNs), motivated by the superiority of CNNs in extracting complex features. By using the textual information and ratings from users and videos, our model can extract the unique latent factors of users and videos. These latent factors are then used to predict ratings. Notably, unlike the existing works that mainly focus on exploiting the features from item's perspective, we shift the focus to user's perspective. Finally, we incorporate the video and user latent factors into probabilistic matrix factorization (PMF) model to improve the recommendation accuracy.
2) We integrate JointCloud architecture into mobile IoT scenario. By training the models in distributed clouds, this framework can provide accurate video recommendations to the minority of mobile users when the data distribution is non-i.i.d. Specifically, this process includes single cloud training steps where each cloud trains recommendation model by using its local data. It also includes global aggregation steps where different clouds upload their training weights to an aggregator. The aggregator then aggregates all the parameters by taking a weighted average. After aggregation, it distributes the updated parameters to each single cloud for the next iteration.
3) We design a weight compression algorithm to decrease communication overhead during federated training. Low-rank matrix factorization (MF) and 8-bit quantization are combined together and achieve a compression ratio of 12.83 without sacrificing accuracy.
4) We comprehensively evaluate the performance of the proposed methods via extensive experiments on a real-world data set. The experimental results demonstrate that our proposed video recommendation model outperforms

existing classic and the state-of-the-art models. By federated training, our approach is more applicable to the non-IID scenario.

The remainder of this article is organized as follows. Section II presents related works. In Section III, we describe the system architecture of JointRec and introduce the Dual-CPMF model for video recommendation in details. The FR strategy and efficient communication strategy will be given in Section IV. The experimental results are shown in Section V and the conclusion is presented in Section VI.

II. Related Work

A. Deep-Learning-Based Video Recommendation

Recently, the deep-learning techniques have been studied widely in the video recommender systems. Convington *et al.* [7] designed a two-stage recommendation approach using deep learning, which allows YouTube to make recommendations from large video corpus. Kim *et al.* [19] proposed a framework called ConvMF that integrates CNN into PMF to capture textual information of documents and enhance the rating prediction accuracy. Different from [19] that focused on exploiting video content from item's perspective, our work considers both user attributes and video properties, and applies CNN on user documents to further capture the user preferences. Wu *et al.* [20] developed a novel recommender system based on recurrent neural networks that can accurately model the user and movie dynamics on Netflix and IMDB data sets. Zhao *et al.* [8] developed a heterogeneous movie recommendation model that exploits the textual descriptions, user ratings, and social relationships. However, these approaches are only suitable to traditional centralized recommendation systems, and they rarely pay attention to extracting user profile features that are essential in distributed JointCloud video recommendation, whose distribution of data is non-IID.

B. Distributed Learning and JointCloud Computing

From the perspective of distributed learning, Povey *et al.* [21] studied distributed training by iteratively averaging local training models. Arjevani and Shamir [22] investigated distributed machine learning based on gradient descent from a theoretical point of view. But these works only consider the data setting and have unrealistic assumptions that the data at different nodes are i.i.d, whereas the more general cases involve non-i.i.d. which is hard to solve. Google in [23] proposes a communication-efficient learning method for training decentralized data called federated learning (FL). Each client has a local training data set and computes an update to the current global model maintained by the server. The experiments demonstrate that it is robust to unbalanced and non-i.i.d data distributions. However, the FL approach cannot be directly applied to our learning task, because the data derive from individual mobile devices and train locally. Our work shifts the focuse on multicloud cooperation with each other; each cloud data center stores user data of several specific regions and trains jointly. Wang *et al.* [24] addressed the problem of learning model parameters from data distributed across multiple edge nodes, but they mainly focused on theoretical analysis of distributed machine learning. The practical applications among distributed nodes and the joint cloud are not considered. Yue *et al.* [25] presented the JointCloud computing data trading architecture (JCDTA), which is optimized to solve the data trading problem for cross-stakeholder in the JointCloud environment. Based on JointCloud Blockchain, Chen *et al.* [26] achieved a privacy-protected and intercloud data fusing platform that satisfies the demand for data mining and analytic activities in IoT. Fu *et al.* [27] proposed JCLedger, a blockchain-based distributed ledger for JointCloud computing.

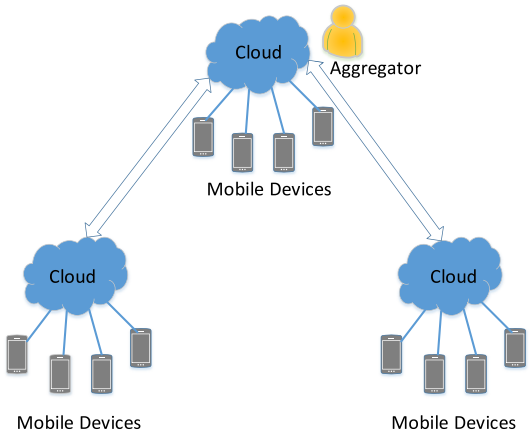


Fig. 2. System architecture of JointRec.

In contrast to the above works, our research in this article emphasizes the JointCloud computing on the non-IID data scenario, where the features of users and data distribution are various among different cloud servers.

III. System Architecture and Dual-CPMF Video Recommendation

We consider a distributed JointCloud video recommender system architecture where cloud servers are distributed in different places, as illustrated in Fig. 2. The user data of watching a video in these regions are collected and stored in a specific cloud server. Each cloud server trains a video recommendation model using the local data. Among these clouds, one of the cloud is designated as the aggregator, which is responsible for aggregating the training parameter files. Once other cloud servers receive the requests, they send the parameter files to the aggregator. Then the aggregator returns the new parameter files to these cloud servers after some processing. Some key notations frequently used in this article are summarized in Table I.

In the following, we describe the Dual-CPMF model for video recommendation in detail. As shown in Fig. 3, the user and video document refer to the user's reviews on the videos and the description and name of the videos, respectively. The user profile includes the user's attributes, i.e., user id, gender, age, and occupation. The video profile refers to the genres of videos. The network structures of user and video are similar; they utilize the CNNs with word embedding to extract the feature of users and videos from the textual information. Then,

TABLE I
KEY NOTATIONS

Notation	Definition
$\vec{e_i}$	The embedding vector of the i-th word
E_i	The user word embedding matrix for user i
l	The length of the document
c_i^j	The j-th shared weight of user i
m_p^i	The pooling feature vector for document of user i
n_f	The number of filters
W_c	The weight matrix of convolutional layer
b_c	The bias of convolutional layer
T_i,S_j	The feature vector of user and video
X_i,Y_j	The raw input documents of user i and video j
U, V	The latent factor of user and video
θ_i, ξ_j	The Guassian noise matrix of user i and video j
I_{ij}	The indicator to imply whether the user rates video
σ_U, σ_V, σ_W	The variance of Gaussian noise matrix U, V, W
r_{ij}	The real rating value of user i for video j
u_i, v_j	The updated value of user i and video j latent vector.
W_1, W_2	The all parameters in User-CNN and Video-CNN
$g_n(t)$	The gradient descent of cloud n at time t
$w_n(t)$	The weight file of cloud n at round t
$H_n^{a\times b}$	The weight update matrix of cloud n
h_{max}, h_{min}	The maximum and minimal value of $H_n^{a\times b}$

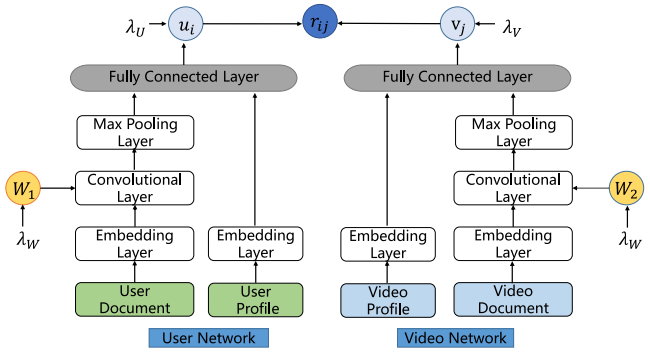


Fig. 3. Dual-CPMF model.

the PMF model is applied to take the convoluted features as the latent factor to predict the rating matrix. With the predicted rating matrix, we can conduct top-N video recommendations to users. λ_U, λ_V, and λ_W are three constant values which will be discussed in the following.

A. Word Embedding Layer

As shown in Fig. 4, the embedding layer transforms a raw document into a textual matrix. Each word is represented as a feature vector. In this article, we adopt the pretrained word embedding model Glove [28]. Then, we have

$$E_i = \left[\vec{e}_1||\vec{e}_2||\cdots||\vec{e}_l\right] \tag{1}$$

where vector $\vec{e}_i$ represents the embedding vector of the ith word, the E_i denotes the user word embedding matrix for user i, and l is the length of the document.

B. CNN Architecture

The CNNs can extract the textual features; it includes three layers: 1) the convolutional layer; 2) the pooling layer; and 3) the fully connected layer.

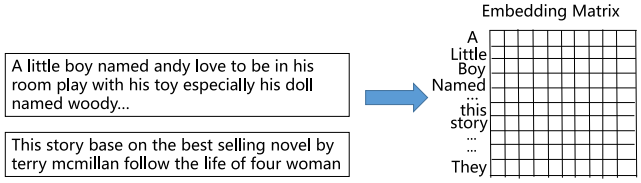


Fig. 4. Word embedding representation.

Convolutional Layer: The convolutional layer adopts the filters to extract features of the documents [29]. Here, we set the width of these filters as the width of the input matrix, and then slide the filters over full rows of the input embedding matrix to form the feature map [30]. The new features after convolution operation can be expressed as follows:

$$c_i^j = f\left(W_c^j * E_i + b_c^j\right) \tag{2}$$

where $W_c^j \in \mathbb{R}^{w*s}$ is the jth shared weight (w is the filter size and s is the dimension of word embedding), b_c^j is the bias, $*$ denotes the convolution operator, and f is a nonlinear activation function. There are several nonlinear activation functions, such as sigmoid, tanh, and ReLU; we select ReLU to avoid the problem of vanishing gradient.

Pooling Layer: After the convolutional layer, each filter generates a feature map. We then apply the max-pooling operation over the corresponding feature map. The pooling layer extracts the most prominent convoluted feature of each feature map, and constructs a fixed size vector. The feature vector at this layer for user i can be formulated as follows:

$$m_p^i = \left[\max\left(c_i^1\right), \max\left(c_i^2\right), \ldots, \max\left(c_i^{n_f}\right)\right] \tag{3}$$

where m_p^i is the pooling feature vector for user document of user i, and n_f is the number of filters.

Fully Connected Layer: In the fully connected layer, high-level features extracted by convolutional and pooling layers are projected by using nonlinear activation function

$$T_i = \tanh\left(W_f * m_p^i + b_f\right) \tag{4}$$

where W_f is the weight matrix of the fully connected layer and b_f is the bias.

Finally, we take the output of the fully connected layer as our latent factor for PMF. Therefore, through the above processes, the CNN architecture can be viewed as a function that takes a textual document as input, and returns feature vectors of users and videos as follows:

$$T_i = cnn_u(W_1, X_i) \tag{5}$$

$$S_j = cnn_v\left(W_2, Y_j\right) \tag{6}$$

where T_i and S_j are the user and video feature vectors, respectively. W_1 and W_2 denote all the weight and bias variables. X_i and Y_j are the raw input documents of user i and video j.

C. Probabilistic Matrix Factorization

Then, we adopt the PMF model to generate the ratings. Assuming we have N users and M videos, the ratings are represented by the $R \in \mathbb{R}^{N\times M}$ matrix. With the user and video feature matrix T_i and S_j from CNN architecture, we further

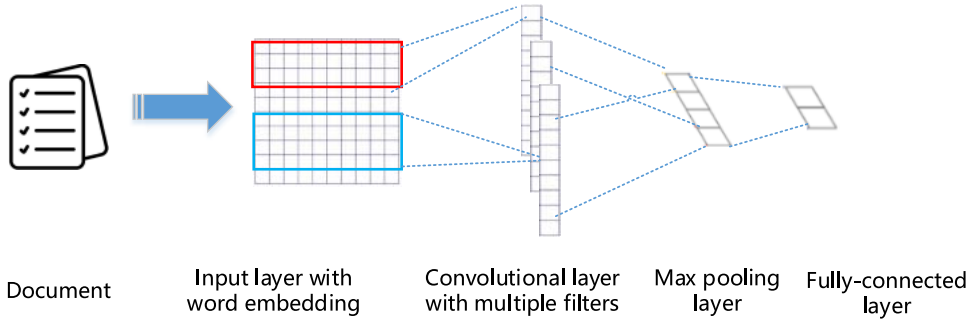


Fig. 5. Convolutional layer.

add two zero-mean spherical Gaussian noise variables with variance σ^2. Hence, the user latent factor U_i for user i and the video latent factor V for video j are given by

$$U = T_i + \theta_i \tag{7}$$

$$V = S_j + \xi_j \tag{8}$$

where θ_i and ξ_j are Guassian noise matrix, θ_i follows zero-mean Gaussian distribution, and the variance is σ_U^2. ξ_j follows zero-mean Gaussian distribution whose variance is σ_V^2. Namely, $\theta_i \sim \mathcal{N}(0, \sigma_U^2 I_{ij})$ and $\xi_j \sim \mathcal{N}(0, \sigma_V^2 I_{ij})$

$$p\left(\theta|\sigma_U^2\right) = \prod_i^{\mathcal{N}} \mathcal{N}\left(\theta_i|0, \sigma_U^2 I_{ij}\right) \tag{9}$$

$$p\left(\xi|\sigma_V^2\right) = \prod_j^{\mathcal{M}} \mathcal{N}\left(\sigma_j|0, \sigma_V^2 I_{ij}\right). \tag{10}$$

Furthermore, we denote $W_1, W_2 \sim \mathcal{N}(0, \sigma_W^2 I_{ij})$ for the training weights. As mentioned above, the conditional distributions over user and video latent factor are given by

$$p(U|W_1, X_i, \sigma_U^2) = \prod_i^N (u_i|T_i, \sigma_U^2 I_{ij}) \tag{11}$$

$$p(V|W_2, Y_j, \sigma_V^2) = \prod_j^M (v_j|S_j, \sigma_V^2 I_{ij}) \tag{12}$$

where r_{ij} is the real-rating value, $\mathcal{N}(x|0, \sigma^2)$ is the probability density function, and I_{ij} is an indicator that $I_{ij} = 1$ implies that the user rates video and 0 otherwise.

In order to optimize the variables in user and video latent models, weight and bias variables of CNNs, we can maximize a posterior estimation (MAP) to conduct PMF

$$\begin{aligned} \max_{U,V,W_1,W_2} & \; p\left(U, V, W_1, W_2|R, X_i, Y_j, \sigma^2, \sigma_U^2, \sigma_V^2, \sigma_W^2\right) \\ = & \max_{U,V,W_1,W_2} \Big[p\left(R|U, V, \sigma^2\right) p\left(U|W_1, X_i, \sigma_U^2\right) \\ & \times \; p\left(V|W_2, Y_j, \sigma_V^2\right) p\left(W_1|\sigma_W^2\right) \\ & \times p\left(W_2|\sigma_W^2\right)\Big]. \end{aligned} \tag{13}$$

D. Parameters Optimization

Given a training data set, there are many parameters to be optimized in Dual-CPMF. We want to find the MAP estimate of U, V, W_1, W_2, predicting the missing values in ratings R and using the predictions to do recommendation. Actually, the maximization of the posterior probability is equivalent to minimizing the joint log-likehood. By taking negative logarithm on (13), it can be reformulated as follows:

$$\begin{aligned} \mathcal{L}(U, V, W_1, W_2) = & \sum_i^N \sum_j^M \frac{I_{ij}}{2} \left\|\left(r_{ij} - u_i^T v_j\right)\right\|_2 + \frac{\lambda_U}{2} \\ & \times \sum_i^N \|u_i\|_2 + \frac{\lambda_V}{2} \sum_j^M \left\|v_j\right\|_2 + \frac{\lambda_w}{2} \\ & \times \sum_k \left\|\left(W_1^k + W_2^k\right)\right\|_2 \end{aligned} \tag{14}$$

where λ_U is σ^2/σ_U^2, λ_V is σ^2/σ_V^2, λ_w is σ^2/σ_W^2, and u_i and v_j indicate the updated value of user and video latent vector.

In order to obtain the optimal value of U and V, we adopt the coordinate descent algorithm, which iteratively optimizes a latent variable by fixing the remaining variables during training process. We first fix U(or V), W_1, W_2 and take derivative of $\mathcal{L}$ with respect to U(or V) and set it to zero. Solving the corresponding equations will lead to the updating rule as follows:

$$u_i \leftarrow \left(VI_iV^T + \lambda_U I_K\right)^{-1} (VR_i + \lambda_U T_i) \tag{15}$$

$$v_j \leftarrow \left(UI_jV^T + \lambda_V I_K\right)^{-1} \left(UR_j + \lambda_V S_j\right) \tag{16}$$

where I_i is a diagonal matrix whose element is I_{ij}, $(i = 1, \ldots, N)$, and R_i is a vector with $(r_{ij})_{j=1}^N$ for user i. For video j, I_j and R_j are defined similarly. Given U and V, we can further optimize parameters W_1 and W_2. However, the weight parameters cannot be optimized analytically as we do for U and V, because they are closely related to the features in CNN architecture. We note that the loss function $\mathcal{L}$ can be interpreted as a error function with regularized terms. By fixing U and V, we have

$$\begin{aligned} \mathcal{E}(W_1, W_2) = & \frac{\lambda_U}{2} \sum_i^N \|(u_i - T_i)\|_2 + \frac{\lambda_V}{2} \sum_j \left\|v_j - S_j\right\|_2 \\ & + \frac{\lambda_W}{2} \sum_k^{w_k} \left\|\left(w_{1,k} + w_{2,k}\right)\right\|_2 + \text{constant}. \end{aligned} \tag{17}$$

According to (17), we utilize the backpropagation algorithm to, respectively, optimize W_1 and W_2.

After the repeated iterations, the optimization of parameters is updated until convergence. With the optimized U, V, W_1, W_2, we can calculate the unknown values of ratings R and predict the latent preferences

$$\hat{r}_{ij} \approx u_i^T v_j = (T_i + \theta_i)^T \left(S_j + \xi_j\right). \tag{18}$$

Recalling T_i and S_j are the feature vectors that are extracted from CNN architecture. θ_i and ξ_j stand for the Gaussian noise matrix for user i and video j. With respect to the cold-start problem, we can use the predicted user–video matrix to conduct video recommendations.

Now we obtain the training model parameters for video recommendation in each cloud. In the next section, we will study the FD strategy among distributed clouds.

IV. Federated Recommendation and Efficient Communication Strategy

A. Federated Recommendation Strategy

We present a distributed gradient-descent algorithm for FR strategy. We suppose that there are N clouds distributed in different regions, each with local data set of size $s_1, \ldots, s_n, \ldots, s_N$. Each cloud n performs a single batch gradient calculation per communication round t. Specifically, at $t = 0$, all clouds download the same initial parameters from aggregator, then they compute the gradient-descent $g_n(t) = \nabla L_n(w_n(t))$, $(t = 0, 1, 2, \ldots)$ on its local data set. $w_n(t)$ is the weight file of cloud n at round t. This step refers to local update. After one or more local training, the aggregator gathers these gradients and updates the parameters for next iteration by applying a weighted average of all resulting models. We define this process as global update. For each cloud n, the local update rule is defined as

$$w_n(t+1) = w_n(t) - \eta g_n(t) \tag{19}$$

where L_n is the loss function of each cloud n, and $\eta \geq 0$ is the learning rate. Furthermore, the global update on the aggregator is defined as follows:

$$w(t+1) = \sum_{n=1}^{N} \frac{d_n}{d} w_n(t+1) \tag{20}$$

where $d = \sum_{n=1}^{N} d_n$.

The FR strategy is presented in Algorithm 1. the compress and decompress functions will be explained in Section IV-B.

Algorithm 1: Federated Recommendation Strategy

Input: The N clouds are indexed by n; the weight file w_n of each cloud n; the amount of mobile devices d_n in each cloud, the local minibatch size B.

{**Aggregator**};
Initialize w_0;
for *each round t=1,2,...* **do**
  Compress(w_0);
  for *each cloud n in parallel* **do**
    stransfer(ip, remote, local, usr, psd);
    request(url);
    for *each* $w_n(t)$ **do**
      Decompress($w_n(t)$);
    $w_n(t+1), d_n, n \leftarrow$ Cloud(n,$w_n(t)$);
    response();
    d += d_n;
  $w(t+1) \leftarrow \sum_{n=1}^{N} \frac{d_n}{d} w_n(t+1)$;
{**Single Cloud**};
Cloud(n, $w_n(t)$):
  listen(port);
  Decompress($w_n(t)$);
  user_loadweight($w_n(t)$);
  for *each local epoch n from 1 to B* **do**
    CNN_Model():
      $w_n(t+1) = w_n(t) - \eta \nabla L_n(w_n(t))$;
    ctransfer();
    Compress($w_n(t+1)$);
    return $w_n(t+1)$ to aggregator.

B. Efficient Communication Strategy

As mentioned above, each single cloud independently computes a weight update to the current model based on its local data, and communicates the update to an aggregator cloud, where the single cloud-side updates are aggregated to compute a new global model. Therefore, the communication efficiency is of the utmost importance. The goal of increasing communication efficiency of FR is to reduce the cost of sending weight file to the aggregator, while learning from data stored on single cloud with limited Internet connections. In this article, we propose a weight compression strategy to reduce the uplink communication cost. We compress the weight update files by using a combination of low-rank MF [31] and 8-bit probabilistic quantization [32] before sending it to the aggregator, and then we decompress the files before global training.

Low-Rank Matrix Factorization: Supposing the weight update matrix of cloud n is $H_n^{a\times b}$, $a \leq b$, we express $H_n^{a\times b}$ as the product of two matrices: $H_n^{a\times b} = P_n^{a\times k}\, Q_n^{k\times b}$, $k = b/N$, where N is a positive integer which affects the compression performance. After MF, the number of elements can be compressed by ϱ of the original. ϱ is a compression ratio, where $\varrho = (k \times (a+b))/(a \times b)$, $\varrho \in [(1/N), (2/N)]$.

Eight-Bit Quantization: After the low-rank MF, we further compress the updates by 8-bit quantizing. Let $h_{\max}$ and $h_{\min}$ as the maximum and minimum value of weight matrix $H_n^{a\times b}$. First, we equally divide $[h_{\min}, h_{\max}]$ into 2^M intervals. The length of each interval is l_{in}. For any value h in matrix $H_n^{a\times b}$, we calculate the position pos of h in the interval, $\text{pos} = \lfloor (h/l_{\text{in}}) \rfloor$. Finally, we transform the pos into 8-bit value. The weight file can be compressed by $(32/M)$x.

The aggregator will decompress the weight file when receiving the compressed update from each single cloud. Decompression is the reverse process of compression. The weight compression algorithm is presented in Algorithm 2.

V. Experimentation Results

In this section, we first evaluate the performance of our approach on a real-world data set from four perspectives:

Algorithm 2: Weight Compression Algorithm

Input: the weight file w_n, N, M=8
Output: the compressed $U_n^{a\times k}$, $V_n^{k\times b}$
for *each* $H_n^{a\times b}$ *from every layer of* w_n **do**
  h_{max} = Max($H_n^{a\times b}$);
  h_{min} = Min($H_n^{a\times b}$);
  l_{in} = (h_{max} - h_{min}) / 2^M;
  if $a > 1$ **then**
    //*Low Rank Matrix Factorization*;
    $k = \frac{a}{N}$;
    $U_n^{a\times k}$ = Random initialization matrix;
    $V_n^{k\times b} = U_n^{k\times a}\ H_n^{a\times b}$;
    //*M-bit Quantization*;
    for *each* u *in* $U_n^{a\times k}$ **do**
      $j = \lfloor u/l_{in} \rfloor$;
      u = BYTES(j);
    for *each* v *in* $V_n^{b\times k}$ **do**
      $j = \lfloor v/l_{in} \rfloor$;
      v = BYTES(j);
  else
    $U_n^{a\times k} \leftarrow H_n^{a\times b}$ **for** *each* h *in* $U_n^{a\times k}$ **do**
      $j = \lfloor h/l_{in} \rfloor$;
      v = BYTES(j);
return $U_n^{a\times k}$, $V_n^{k\times b}$;

TABLE II
STATISTICS OF MOVIELENS-1M DATA SET

#users	#videos	#ratings	density
6040	3544	993482	4.641%

1) the accuracy on rating predictions compared with competitors; 2) the parameter analysis; 3) the performance of JointRec system; and 4) the effectiveness of efficient communication strategy.

A. Data Set Description

We evaluate the performance on MovieLens. It is an extensively used data set for movie recommender system. It is composed of users' explicit ratings on the movies from 1 to 5 score. Furthermore, we only select Movielens-1M data set because it provides some additional information besides ratings. It consists of attribute information of each user, i.e., age, occupation, gender, and the movies' genres as well. These features are significant to the studies of distributed JointRec. We can construct the non-IID data scenario for different regions according to the users' age. (Note that we are unable to use the Movielens-10M and Movielens-20M data sets because they do not contain these information, and thus, we could not utilize the user profiles difference among each areas.) Furthermore, we enrich this data set with plot summary for each movie, provided by IMDB.[4] Table II shows the statistics of the data set.

[4]http://www.imdb.com/

TABLE III
DATA DISTRIBUTION ON EACH CLOUD

Cloud	Users	Proportion	Videos	Majority	Minority
Distribution 1					
Cloud 1	898	0.149	3286	865	33
Cloud 2	1337	0.221	3325	1308	29
Cloud 3	3805	0.630	3470	3791	14
Distribution 2					
Cloud 1	726	0.120	3354	626	100
Cloud 2	1325	0.220	3433	1125	200
Cloud 3	3989	0.660	3649	3689	300

B. Setup

We conduct our experiments on the networked prototype system with three cloud servers. They are distributed in three different places and interconnected via http protocol. These three distributed clouds have a local IoT data set. In addition, we assign one server to be the aggregator. These cloud servers are performed on three Linux Work-Station with Intel Xeon CPU E5-2683 v3@2.00 GHz, Nvidia GTX TITAN X, and 32-GB memory.

To investigate the JointRec system, we first partition the data set into three age groups, i.e., G1 group (sum = 876 users): the elders over than 50 years old, G2 group (sum = 1325 users): the teenagers under 24 years old, G3 group (sum = 3839 users): the users between age of 20 to 50. The G1 group is mainly distributed in cloud 1. The G2 group is mainly distributed in cloud 2. The G3 group is mainly placed in cloud 3. Then, we select some users from each group and add in other clouds. In this case, there are two types of data on each cloud server: majority user group data and minority user group data, forming the non-i.i.d on each cloud. We make comparisons under three different cases. In case 1, each cloud server trains a model on its local data and then conducts federate training according to our approach. In case 2, each cloud server performs training locally, but they do not interact with each other. In case 3, all the data are trained centrally in a cloud server. In the experiment, we test two data splits. Table III shows the data distribution on each cloud.

C. Preprocessing and Parameters Setting

We preprocess the data set as follows: 1) converting the users' age, gender, occupation, and movie genres into equal length arrays; 2) constructing user-plot data according to users' ratings; 3) calculating the tf-idf score for each word in user-plot and movie-plot data; 4) selecting top 8000 distinct words as a vocabulary; and 5) spliting the data set into the training set (80%), the validation set (10%), and the test set (10%). Additionally, we remove the items which do not have plot documents in the whole data set and the users that have less 3 ratings [19].

We set the initial size of latent dimension of U and V to 50. The value of λ_U is 100 and λ_V is 10. In the CNN model: 1) we use Adam as the optimizer and each mini-batch consists of 128 training samples; 2) we use three different filters to extract features; and 3) we set the dropout rate to 0.1 to reduce overfitting.

TABLE IV
OVERALL COMPARISON RESULTS OF RMSE

Model	PMF	CTR	CDL	ConvMF	Dual-CPMF
Value	0.8971	0.8969	0.8879	0.8595	0.8472
Improve	5.56%	5.54%	4.58%	1.43%	

D. Competitors

We compared our proposed model Dual-CPMF with four competitors: PMF, CTR, CDL, and ConvMF, which are the representative works on extending MF with content information in recommender systems. PMF [33] is a basic rating prediction model that only uses ratings. Collaborative topic regression [34] (CTR) combines PMF and LDA to use ratings and documents. Collaborative deep learning [35] (CDL) uses both SDAE and PMF to improve accuracy. ConvMF [19] (convolutional PMF) is a context-aware recommendation model that integrates CNN into PMF.

E. Evaluation Metrics

Root mean squared error (RMSE) is a extensively used measurement for the performance evaluation of rating predictions. A smaller value indicates a better performance of predictions [36]

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i,j}^{N,M}\left(r_{ij} - \hat{r}_{ij}\right)^2} \tag{21}$$

where $\hat{r}_{ij}$ is the predicted value, and n is the number of ratings. As for the top-K recommendations, Recall@K ($R@K$) and Precision@K (P@K) are selected to evaluate the recommendation accuracy

$$R@K = \frac{|R(i)| \cap |T(i)|}{|T(i)|} \tag{22}$$

$$P@K = \frac{|R(i)| \cap |T(i)|}{|R(i)|} \tag{23}$$

where $R(i)$ is the top-K videos in the recommended list for user i, $T(i)$ is the item set that user i likes, and $|\cdot|$ indicates the number of elements in the set. $R@K$ refers to the fraction of items returned in the top-K list. $P@K$ refers to how many recommended items are accurate in the top-K list.

F. Experimental Result

The experimental results consist of two aspects: 1) for the centralized training, we mainly evaluate the performance of our proposed video recommendation model and 2) for the distributed training, we compare the performance under different cases. They include federated training among distributed clouds, single cloud training without information fusion based on local data set independently and the centralized baseline.

1) Prediction Accuracy Evaluation: We first compare the results of rating prediction and evaluate the prediction accuracy of our model. Table IV shows the RMSE results of our approach and other models. We can observe that Dual-CPMF outperforms other methods, proving the effectiveness of the user feature extraction for model fitting. Note that "Improve" indicates the degree of improvement of "Dual-CPMF" over these methods. The results indicate that the users' profiles play an important role in recommendation model, and extracting features from user perspective can model users' preferences better. Therefore, our proposed model is more suitable for video recommendation.

TABLE V
RMSE OVER VARIOUS SPARSENESS OF TRAINING DATA ON THE DATA SET

Model	Ratio of training set to the entire dataset			
	20% (0.93%)	40% (1.86%)	60% (2.78%)	80% (3.71%)
PMF	1.1137	0.9964	0.9251	0.9097
ConvMF	0.9908	0.9361	0.8822	0.8595
Dual-CPMF	0.9700	0.9143	0.8596	0.8472
Improvement	2.09%	2.32%	2.56%	1.43%

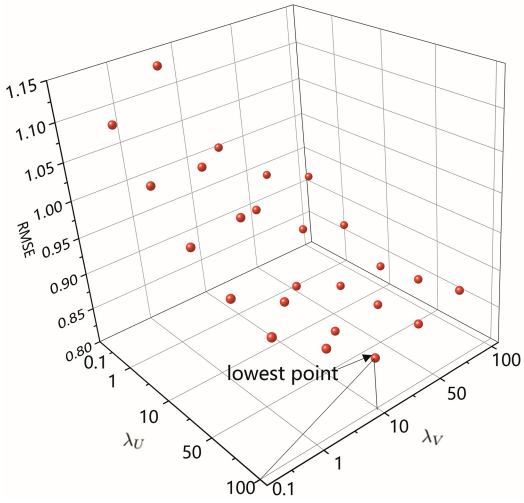


Fig. 6. Parameter analysis of λ_U and λ_V.

2) Impact of Various Sparseness of the Data Set: As shown in Table V, we compare the RMSE on different sparsenesses of the data set. Because the ConvMF outperforms CTR and CDL, we only select PMF and ConvMF for comparison. Our approach performs better than both of them over all ranges of sparseness. Specifically, we note the improvement of Dual-CPMF over ConvMF from 2.09% to 1.43% when data density increases from 0.93% to 3.71%. It indicates that Dual-CPMF can obtain more accurate predictions by providing more ratings.

3) Parameter Analysis: To investigate the impact of the parameters on the performance of Dual-CPMF, we conduct parameter analysis on λ_U and λ_V. We consider four values of λ_U and λ_V, i.e., 0.1, 1, 10, 50, 100 in this experiment. Fig. 6 shows the RMSE changes of Dual-CPMF under different values of λ_U and λ_V on the data set. λ_U implies how T_i affects user latent factor U and λ_V indicates how S_j affects video latent factor V. Note that the prediction error is high when both parameters approach 0, while the value decreases when λ_U and λ_V increase. It indicates that increasing the textual information of user and video is a benefit for increasing the accuracy of rating prediction. We can also see that the best

TABLE VI
IMPACT OF LATENT DIMENSION D

Model	Latent Dimension D			
	10	20	50	100
PMF	0.8685	0.8665	0.8683	0.8646
ConvMF	0.8863	0.8770	0.8566	0.8554
Dual-CPMF	0.8568	0.8526	0.8489	0.8474

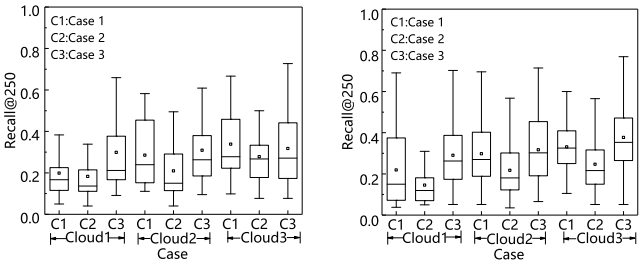


Fig. 7. Recall@250 comparison for the minority user groups under two data distribution: (a) distribution 1 and (b) distribution 2.

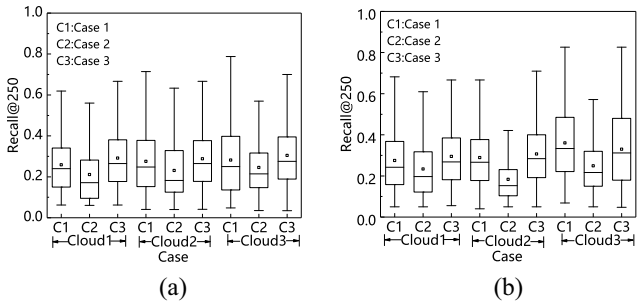


Fig. 8. Recall@250 comparison for the majority user groups under two data distribution: (a) distribution 1 and (b) distribution 2.

performance of recommendation is achieved when (λ_U, λ_V) takes value of (100, 10), the "lowest point" (RMSE = 0.8472) in the figure. It implies that the influence of users' perspective on the prediction accuracy is greater than that of videos' perspective. However, when λ_U exceeds a certain threshold, the accuracy begins to decrease, and it is likely that it would have side effect when we put much emphasize on the user information. Furthermore, we note that the ideal combination of two parameters on ConvMF and Dual-CPMF is the same, but we achieve the smaller RMSE than the ConvMF model. It indicates that the implements of Dual-CNN structure and the feature extraction of user profiles are more suitable for matrix MF, and they are helpful for building a more effective video recommender system.

In addition, we compare the influence of dimension D of the latent factor on the performance of Dual-CPMF and other competitors. These latent factors contain characteristics of users and movies. The higher value of D implies the richer information. As shown in Table VI, Dual-CPMF outperforms all other models in any cases. Furthermore, we can observe that the performances of all models increase when factor dimensionality D grows, which indicates that more textual information brings better prediction accuracy.

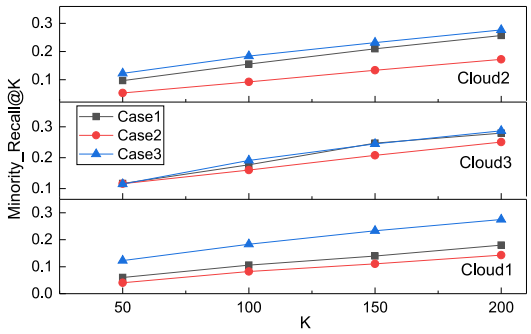


Fig. 9. Recall@K comparison for the minority user groups under different cases.

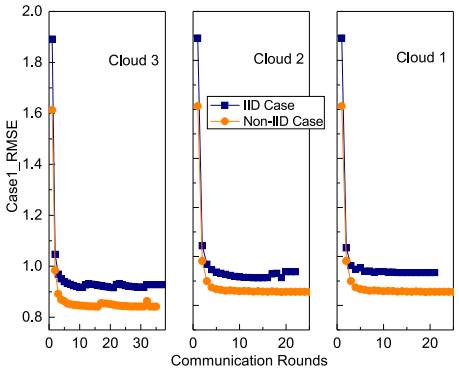


Fig. 10. RMSE for IID case and non-IID case.

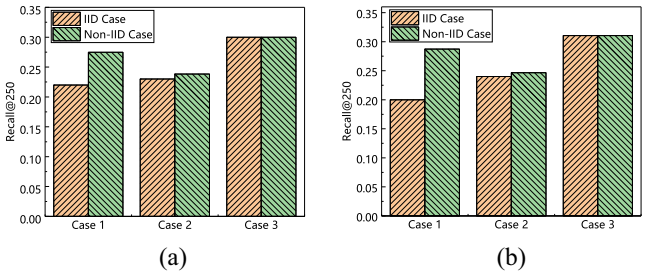


Fig. 11. Recall@250 comparison for IID and non-IID case under two data distribution: (a) distribution 1 and (b) distribution 2.

4) Performance Evaluation of Distributed Video Recommendation System: In Figs. 7 and 8, we concentrate on performance evaluation in different cases. In the figure, the bar in a box is the average recall of the model. The upper and bottom borders of a box represent 75 percentile and 25 percentile. The tips of the upper and bottom whiskers represent the max and min values. Fig. 7 shows the Recall@250 comparison for the minority user groups under two data distribution. The recall of our method (case 1) is higher than case 2 in all clouds on average, validating our intuition that federated training and parameter fusion can improve the performance in the distributed video recommendation system. Although the average recall of case 1 is slightly lower than in case 3, with the increasing number of users, the recall of our method approaches case 3, and sometimes even better than case 3. It is probably due to the distributed fusion training making use of the computation resources of multiple clouds.

TABLE VII
COMPARISON OF COMPRESSION RATIO

Compression Method	RMSE			Model Size	Compression Ratio
	Cloud 1	*Cloud* 2	*Cloud* 3		
Baseline	0.8885	0.9285	0.8547	85.58MB	1x
Matrix Factorization	0.8854	0.9283	0.8543	43.7MB	1.95x
8-bit Quantification	0.8873	0.9368	0.8541	21.46MB	3.98x
MF + 8-bit Quantification	0.8804	0.9281	0.8545	6.67MB	12.83x

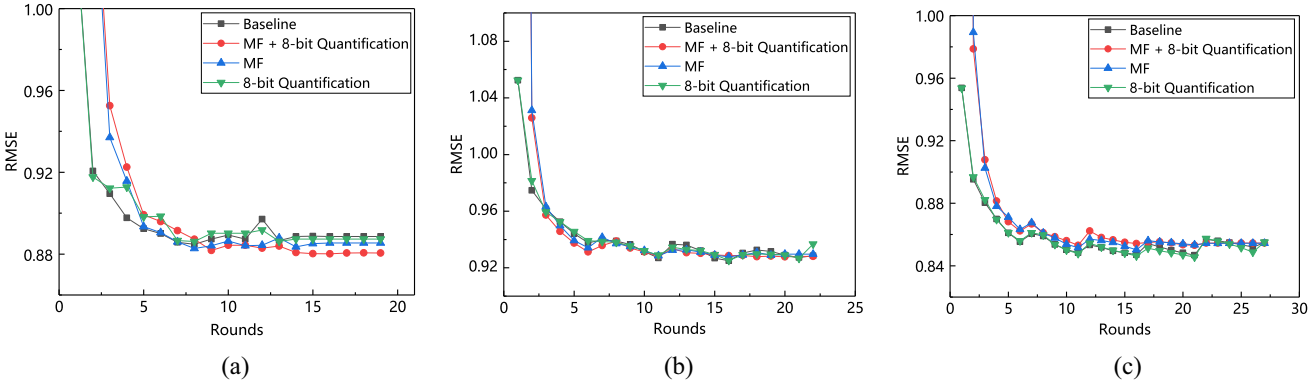


Fig. 12. Comparison of RMSE with MF, 8-bit quantization and combination. (a) RMSE of cloud 1, (b) RMSE of cloud 2, and (c) RMSE of cloud 3.

Furthermore, we can see that the recall of case 1 shows an increasing trend from clouds 1 to 3. This is because the number of users on these three clouds is increasing, and the more users will bring better recommendation performance. Fig. 8 shows the Recall@250 comparison for the majority user groups under two data distribution. As shown in the figure, the results of majority user group are similar to the minority user groups in Fig. 7.

Fig. 9 shows the comparison of different Recall@K for the minority of user groups in different cases. As shown in the figure, with the value of K increasing from 50 to 200, the recall of all cases in clouds 1 to 3 increases, and our method (case 1) outperforms the one in case 2, although the performance of case 3 is better than case 1 in cloud 1. As the number of users increases in clouds 2 and 3, the value of recall in case 1 approaches case 3.

5) Comparison for IID Case and Non-IID Case: To evaluate the performance of our approach in the i.i.d and non-i.i.d scenarios, we compared RMSE and Recall@250 in Figs. 10 and 11, respectively. In the IID case, we select the same percentage of users from each age group according to the proportion in Table III and assign to each cloud. In this case, the age distribution among each cloud is i.i.d. and the number of users under IID and non-IID cases is equal in each cloud.

As shown in Fig. 10, we can observe that the RMSE under the non-IID case is lower than the IID case, indicating that our approach in the non-IID case achieves better prediction accuracy than in the IID case. In addition, we compare the recall under two data distributions in Fig. 11. We can see that the recall of IID case under case 1 is higher than the non-IID case. However, the results in cases 2 and 3 are similar. This is because these clouds conduct distributed fusion and federated training in case 1. Whereas in cases 2 and 3, they do not adopt federate training and weight fusion, so their recalls are similar under the IID case and the non-IID case, validating our method is applicable to the non-IID scenario.

6) Comparison of Compression Ratio: We evaluate the reduction in the network bandwidth using weight compression ratio as follows:

$$\text{Compression Ratio} = \text{size}\left[H_n^{a\times b}\right] \Big/ \text{size}\left[\text{compress}\left(H_n^{a\times b}\right)\right]. \tag{24}$$

Table VII shows the results of RMSE and compression ratio on clouds 1–3. We compare the performance of several compression methods. The MF, 8-bit quantization, and the combination compression method give 1.95×, 3.98×, and 12.83× larger compression than baseline with no increase of RMSE.

7) Comparison of RMSE Under Compression: In Fig. 12, we compare three types of compression method introduced in Section IV-B. Fig. 12(a)–(c) shows the RMSE convergence curves on clouds 1–3. As we can see in Fig. 12(a), the combination compression method is in the bottom row and baseline is in the top row. It indicates that the combination compression method performs better than others on *cloud* 1, whereas on clouds 2 and 3, shown in Fig. 12(b) and (c), the curves of three compression methods are much closer to the baseline. This is caused by the different distribution of users on three clouds. Therefore, we can conclude that weight compression not only reduces the communication overhead but also has no impact on the recommendation performance.

8) Comparison of Recall/Precision/F-Measure Under the Compression: In addition to the comparison of RMSE, we also discuss the Recall@250, Precision@250, and F-measure ($\alpha = 1$) on these three clouds. The recall and precision

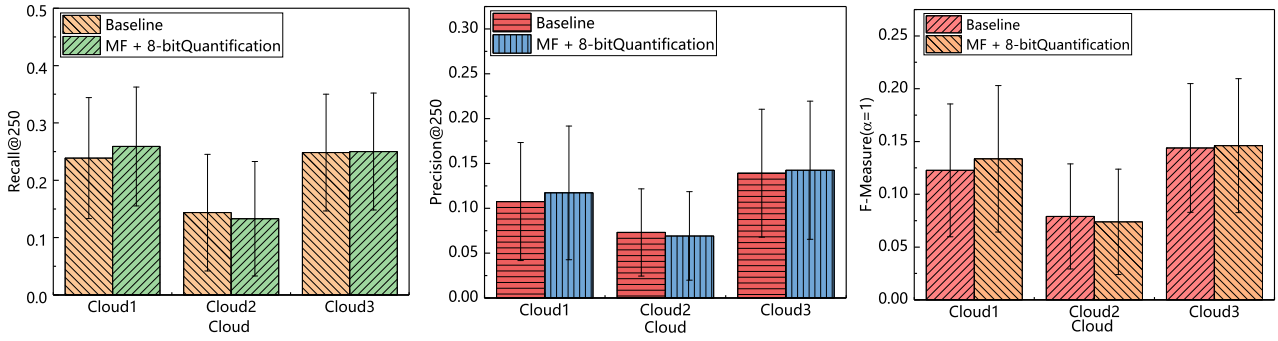


Fig. 13. Comparison of Recall@250, Precision@250, and F-measure($\alpha = 1$). (a) Comparison of recall, (b) comparison of precision, and (c) comparison of F-measure($\alpha = 1$).

are defined above. The F-measure considers the recall and precision comprehensively

$$\text{F-Measure} = \frac{(\alpha^2 + 1)PR}{\alpha^2 P + R} \tag{25}$$

where P is the precision and R is the recall, and α is a variable. In Fig. 13, we can observe that the values of the three metrics under compression are close to the baseline. The results show that weight compression strategy has less impact on system performance in terms of recall, precision, and F-measure.

VI. Conclusion

In this article, we have proposed JointRec, a deep-learning-based video recommendation framework for minority mobile users. To this end, we have designed a Dual-CPMF video recommendation model; this model can extract the unique latent features of users and videos from the users' profiles and the description of videos, thereby achieving more accurate recommendation performance. Then, we have proposed the FR strategy where each distributed cloud trains the model based on local area data and updates training weights cooperatively. Considering the heavy communication cost during weight update, we have developed a weight compression algorithm to reduce network bandwidth and communication overhead. We have evaluated the performance of JointRec on real-world movie data set. The experimental results demonstrate that JointRec can recommend accurate videos to both the minority and majority users, while the efficient communication strategy can achieve 12.83× larger compression ratio than the baseline with no loss of performance.

In the future, we plan to investigate the edge-end-cloud orchestrated distributed video recommendation, where the recommender system is deployed in the edge server in proximity to mobile users. In such a case, the data generated by mobile users can be directly processed locally on edge servers rather than being sent to the remote clouds. How to deal with the cooperative computing among them is a new challenge.

References

[1] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surveys*, vol. 52, no. 1, p. 5, 2019.

[2] T. Yang, H. Liang, N. Cheng, R. Deng, and X. Shen, "Efficient scheduling for video transmissions in maritime wireless communication networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 9, pp. 4215–4229, Sep. 2015.

[3] X. Zhang *et al.*, "Improving cloud gaming experience through mobile edge computing," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 178–183, Aug. 2019.

[4] X. Zhang, H. Yin, D. O. Wu, G. Min, H. Huang, and Y. Zhang, "SSL: A surrogate-based method for large-scale statistical latency measurement," *IEEE Trans. Services Comput.*, to be published.

[5] R. Xing, Z. Su, N. Zhang, J. Luo, H. Pu, and Y. Peng, "Trust based intrusion detection and learning aided incentive mechanism for autonomous driving," *IEEE Netw.*, to be published.

[6] Y. Wang, Z. Su, Q. Xu, T. Yang, and N. Zhang, "A novel charging scheme for electric vehicles with smart communities in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8487–8501, Sep. 2019.

[7] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for YouTube recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 191–198.

[8] Z. Zhao *et al.*, "Social-aware movie recommendation via multimodal network learning," *IEEE Trans. Multimedia*, vol. 20, no. 2, pp. 430–440, Feb. 2018.

[9] H. Yin *et al.*, "Edge provisioning with flexible server placement," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 4, pp. 1031–1045, Apr. 2016.

[10] Z. Su, Y. Hui, and T. H. Luan, "Distributed task allocation to enable collaborative autonomous driving with network softwarization," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2175–2189, Oct. 2018.

[11] X. Zhang, H. Huang, H. Yin, D. O. Wu, G. Min, and Z. Ma, "Resource provisioning in the edge for IoT applications with multi-level services," *IEEE Internet Things J.*, to be published.

[12] Y. Wang, Z. Su, and N. Zhang, "BSIS: Blockchain-based secure incentive scheme for energy delivery in vehicular energy network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 3620–3631, Jun. 2019.

[13] D. Zhang, Y. Qiao, L. She, R. Shen, J. Ren, and Y. Zhang, "Two time-scale resource management for green Internet of Things networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 545–556, Feb. 2019.

[14] H. Wang, P. Shi, and Y. Zhang, "JointCloud: A cross-cloud cooperation architecture for integrated Internet service customization," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2017, pp. 1846–1855.

[15] T. Yang, Z. Zheng, H. Liang, R. Deng, N. Cheng, and X. Shen, "Green energy and content-aware data transmissions in maritime wireless communication networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 751–762, Apr. 2015.

[16] D. Zhang *et al.*, "Near-optimal and truthful online auction for computation offloading in green edge-computing systems," *IEEE Trans. Mobile Comput.*, to be published.

[17] D. Zhang, Z. Chen, M. K. Awad, N. Zhang, H. Zhou, and X. S. Shen, "Utility-optimal resource management and allocation algorithm for energy harvesting cognitive radio sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3552–3565, Dec. 2016.

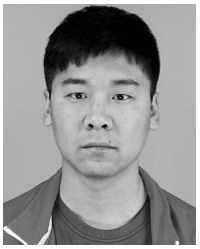
[18] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang, "Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS)," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2014, pp. 193–202.
[19] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 233–240.
[20] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *Proc. 10th ACM Int. Conf. Web Search Data Min.*, 2017, pp. 495–503.
[21] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *arXiv*, Oct. 2014. [Online]. Available: https://arxiv.org/pdf/1410.7455v1.pdf
[22] Y. Arjevani and O. Shamir, "Communication complexity of distributed convex learning and optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1756–1764.
[23] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," *arXiv*, Oct. 2016. [Online]. Available: https://arxiv.org/pdf/1602.05629v2.pdf
[24] S. Wang, T. Tiffany, S. Theodoros, L. K. K. M. Christian, T. He, and C. Kevin, "When edge meets learning: Adaptive control for resource constrained distributed machine learning," in *Proc. 37th IEEE Conf. Comput. Commun.*, 2018, pp. 63–71.
[25] X. Yue, H. Wang, W. Liu, W. Li, P. Shi, and X. Ouyang, "JCDTA: The data trading architecture design in joint cloud computing," in *Proc. IEEE 24th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, 2018, pp. 1–6.
[26] W. Chen, M. Ma, Y. Ye, Z. Zheng, and Y. Zhou, "IoT service based on jointcloud blockchain: The case study of smart traveling," in *Proc. IEEE Symp. Service Orient. Syst. Eng. (SOSE)*, 2018, pp. 216–221.
[27] X. Fu, H. Wang, P. Shi, Y. Fu, and Y. Wang, "JCLedger: A blockchain based distributed ledger for jointcloud computing," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. Workshops (ICDCSW)*, 2017, pp. 289–293.
[28] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[29] D. Tang, B. Qin, Y. Yang, and Y. Yang, "User modeling with neural network for review rating prediction," in *Proc. Int. Conf. Artif. Intell.*, 2015, pp. 1340–1346.
[30] Z. Wang, Y. Zhang, H. Chen, Z. Li, and F. Xia, "Deep user modeling for content-based event recommendation in event-based social networks," in *Proc. INFOCOM*, 2018, pp. 1304–1312.
[31] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *arXiv*, Dec. 2014. [Online]. Available: https://arxiv.gg363.site/pdf/1412.6115.pdf
[32] Y. Liu, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018, pp. 1–13.
[33] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.
[34] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2011, pp. 448–456.
[35] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. KDD*, 2014, pp. 1235–1244.
[36] M. Elahi, Y. Deldjoo, F. B. Moghaddam, L. Cella, S. Cereda, and P. Cremonesi, "Exploring the semantic gap for movie recommendations," in *Proc. 11th ACM Conf. Recommender Syst.*, 2017, pp. 326–330.

Sijing Duan is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Central South University, Changsha, China.

Her current research interests include edge computing and mobile deep learning.

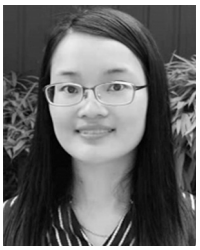
Deyu Zhang (S'15–M'17) received the B.Sc. degree in communication engineering from PLA Information Engineering University, Zhengzhou, China, in 2005, the M.Sc. degree in communication engineering from Central South University, Changsha, China, in 2012, and the Ph.D. degree in computer science from Central South University.

From 2014 to 2016, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Waterloo, ON, Canada. He is currently an Assistant Professor with the School of Computer Science and Engineering, and a Postdoctoral Fellow with the School of Information Science and Engineering, Central South University. His current research interests include stochastic resource allocation in wireless sensor networks and cloud radio access networks, edge computing, and transparent computing.



Yanbo Wang is currently pursuing the graduation degree with the School of Computer Science and Engineering, Central South University, Changsha, China.

His current research interests include edge computing and motion recognition.



Lingxiang Li (S'13–M'17) received the B.S. degree in electrical engineering from Central South University, Changsha, China, in 2010, and the Ph.D. degree in electrical engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2017.

She was a Visiting Ph.D. student under the supervisor of Prof. A. P. Petropulu with Rutgers, The State University of New Jersey, Camden, NJ, USA, from 2015 to 2016. She is currently an Associate Professor with Central South University. Her current research interests include wireless communications, networking, and signal processing, currently focusing on wireless security, mobile edge computing networks, and wireless network economics.

Yaoxue Zhang (M'17–SM'18) received the B.Sc. degree from the Northwest Institute of Telecommunication Engineering, Xi'an, China, in 1982, and the Ph.D. degree in computer networking from Tohoku University, Sendai, Japan, in 1989.

He is currently a Professor with the Department of Computer Science, Central South University, Changsha, China, and also a Professor with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He has published over 200 technical papers in international journals and conferences, as well as nine monographs and textbooks. His current research interests include computer networking, operating systems, ubiquitous/pervasive computing, transparent computing, and big data.

Prof. Zhang is currently serving as the Editor-in-Chief of the *Chinese Journal of Electronics*. He is a Fellow of the Chinese Academy of Engineering.