Survey on Federated Recommendation System

Zeyu Huang, Xiaofeng Zhang, Yuhang Li, Ying Tan, Kunyang Wang, Meng Song and Li Jiang

Abstract—Federated learning is a machine learning setting where many clients collaboratively train a statistical model under command of a central server, during which the training data is conserved locally in each client without uploading. Since this training protocol does not involve data transferring, it become a popular method among Internet Giants to deal with their client's data and to utilize jointly their data across their branches located in different countries. Further, recommendation is one of the most common scenario in Internet Giants' services which demands large amount of personal information to make personalized recommendation. Therefore, it is an interesting question that how to construct recommendation systems under the federated learning modality. In this survey, we first make a thorough review individually for recommendation system and federated learning. Then we make our best to conclude the present work on federated recommendation system. We also provide some future directions in this branch.

Index Terms—Recommendation System, Federated Learning

1 Introduction

Recommendation systems have been attached great importance to since the appearance of the first papers on collaborative filtering in the mid-1990s [1], [2], [3]. The interest in this area still remains high because of the large amount of practical applications that help users to deal with information overload and provide personalized recommendations, content, and services to them. Examples of such applications include recommending books, CDs, and other products at Amazon.com [4], movies by MovieLens [5], and news at VERSIFI Technologies (formerly AdaptiveInfo.com) [6].

In recent years, despite the traditional recommendation methods that work well on practical personalized recommendation scenarios [7] [8] [9], researchers are now having growing interest in neural network even deep neural network based methods [10], [11], [12]. Deep neural networks effectively learn potential explanatory factors and useful representations from input data. It not only reduces the work of manual feature design, but also enables the recommendation model to include heterogeneous content information, such as text, images, audio and even video. Deep neural networks also improve the performance of sequence modeling, such as recurrent neural networks. Besides, deep learning is highly flexible. With the help of a deep learning framework, it is easy to combine different neural structures to build hybrid and complex recommendation models to capture different features and factors at the same time.

With the development of mobile devices, e.g. mobile phones, wearable devices, and autonomous vehicles, and the development of wireless communication infrastructures, large Internet service producers aim to provide instant and personalised contents for mobile device users. For example, language models can improve speech recognition and text entry, and image models can automatically select good photos.

- *Z. Huang, X. Zhang, Y. Li, Y. Tan, K. Wang and M. Song are with Sino-French Engineer School, Beihang University, Beijing 100191, China. E-mail: {ZY2024104, xiaofeng_z, liyuhang, tanying, wangkunyang, songmeng}@buaa.edu.cn.*
- *L. Jiang is with Orange Lab Beijing, Beijing XXXXX, China. E-mail: li.jiang@orange.com.*

Manuscript received xxxx xx, xxxx; revised xxxx xx, xxxx.
(Corresponding author: xxxxxx.)

Because of the increase of public concern about personal privacy on Internet,e.g. the General Data Protection Regulation (GDPR) [1] of EU, how to efficiently and legally utilize user's personal information, e.g., self profiles, cookies, has become a widely discussed topic among researchers in giant Internet companies. Their recommendation methods often generate recommendation results based on the user's browsing history, so this process is likely to infringe the users' privacy.

Google firstly proposed Federated Learning in 2017 [13](original version on arxiv Feb. 2016). Federated Learning has become a standard protocol to deal with privacy preserving problem. Federated learning and federated data analysis are now being applied in consumer digital products. Google makes extensive use of federated learning in the Gboard mobile keyboard [14], [15], as well as in features on Pixel phones [16]. Many advanced methods have been proposed in literature in recent years.For example, Federated matched averaging (FedMA) algorithm designed for federated learning of modern neural network architectures e.g. convolutional neural networks (CNNs) and LSTMs [17].

In literature, surveys for recommendation system [18], [19] and for federated learning [20], [21], [22] independently have already been published. However, to the best of our knowledge, there is no survey dedicated to the area of combined federated recommendation systems. Due to the appearance of some federated based recommendation models and the wide usage of recommendation systems in modern society, we found it essential conduct a survey on federated recommendation systems. For example, Tao Qi et al. proposed a framework named FedNewsRec to recommend news with privacy protection [23].

In this paper, we first conduct a thorough review on recommendation system and on federated learning separately.

1. https://gdpr-info.eu/

Then in Section 4, we first review the existed combined methods proposed in literature, and then try to make a conclusion on how to combine the federated learning and the recommendation system.

2 Related works of Recommendation Systems

2.1 Traditional Recommendation Method

2.1.1 Content-based Filter Recommendations

Content-based recommendation algorithm (CBF) [24] mainly uses user's historical behavior information and user's personal information to recommend. The basic idea of content-based recommendation algorithm is: If users have a lot of contact with and praise a certain item in their historical behavior, it can be considered that users will probably like other items with high similarity to this item. [25] The core idea of this recommendation algorithm determines that it highly depends on the integrity of user information. In the case of users without perfect personal information, user profile cannot be established accurately, and the accuracy of recommendation will be greatly reduced. The figure 1 shows the structure of the recommendation method proposed by Lops.

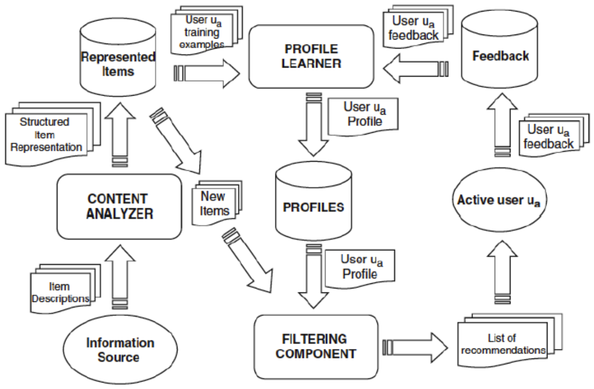


Fig. 1. High level architecture of a Content-based Recommender

Content based recommendation usually has three steps, which are item representation, profile learning and recommendation generation. The system analyzes the user's content, extracts the content according to the preset rules, and represents it as an attribute tensor. Then we generate a model for user u by judging his past preferences. After the above steps, we get the recommendation scores of different items for the target users. After that, the system will sort the item recommendation scores, compare the fit degree between the profile and the item, and then recommend the items that are most in line with the user's interests in the form of a list.

The advantages of CBF can be concluded as follows:

1) User independence: since each user's profile is obtained according to his own preferences for items, it has nothing to do with the behavior of others. A significant benefit of CBF's user independence is that the profile changes of other users will not affect themselves.
2) Good interpretability: CBF recommends items that meet the user's taste according to the user profile, which has good interpretability.
3) A new item can be recommended immediately (new item problem): as long as a new item is added to the item library, it can be recommended immediately, and the chance of being recommended is the same as that of the old item.

The disadvantages of CBF [26] can be concluded as follows:

1) The feature extraction of item is difficult, that is to say, content analysis is limited.
2) The over specialization of users can not be mined: since CBF's recommendation only depends on users' past preferences for certain items, its recommendations will be similar to users' past preferences.
3) Unable to generate recommendation for new users: new users have no preference history, so they can't get their profile, and they can't generate recommendation for them.

2.1.2 User-Based Collaborative Filtering

Collaborative filtering is the most popular [27] type of recommendation algorithm, which has been widely used in industry. It doesn't need much knowledge of specific domain, and can get better recommendation effect through machine learning algorithm based on statistics. It depends on the user's habitual data, but it has no special requirements for recommendation objects, and can deal with unstructured complex objects. The biggest advantage of collaborative filtering is that it is easy to implement in engineering and can be easily applied to products. [28] At present, most of the practical recommendation algorithms are collaborative filtering recommendation algorithms.

Generally, we can divide collaborative filtering into three categories: user based, item-based and model-based CF.

According to all users' preferences for items, this model finds "neighbor" user groups similar to current users' tastes and preferences. In general applications, it uses "K-neighbor" algorithm. Then, based on the historical preference information of the K neighbors, the model makes recommendations for the current users.

User based collaborative filtering algorithm is to find users' preferences for goods or content (such as goods purchase, collection, content review or sharing) through users' historical behavior data, and measure and score these preferences. Then, according to the attitude of different users to the same product or content, the relationship between users is calculated. Finally, we recommend products among users with the same preferences. In short, if user 1 and user 3 both buy products 2 and 3, and give 5-star praise, then user 1 and user 2 are the same kind of users. Products 1 and 4 purchased by user 1 can be recommended to user 3.

User based collaborative filtering algorithm mainly includes two steps:

The key of the first step is to calculate the interest similarity of two users. Here, collaborative filtering algorithm mainly uses the similarity of behavior to calculate the similarity of user interest. Suppose user u and user v, let

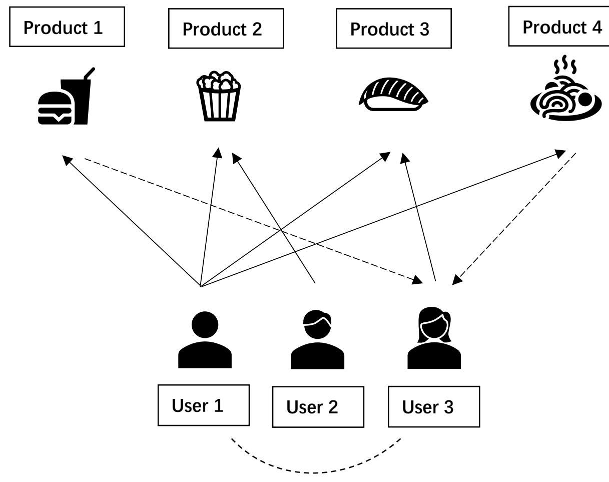


Fig. 2. Schematic diagram of user-based CF

$N(u)$ denote the set of items with positive feedback that u had, and let $N(v)$ denote the set of items with positive feedback that V had. Then, we can simply calculate the interest similarity of u and V through the following Jaccard formula:

$$\mathrm{W_{uv}} = \frac{|N(\boldsymbol{u}) \cap N(\boldsymbol{v})|}{|N(\boldsymbol{u}) \cup N(\boldsymbol{v})|} \tag{1}$$

Or we can calculate cosine similarity:

$$\mathrm{W_{uv}} = \frac{|N(u) \cap N(v)|}{\sqrt{|\mathrm{N(u)}||\mathrm{N(v)}|}} \tag{2}$$

After getting the interest similarity between users, the UserCF algorithm will recommend K items that users like most similar to their interests. The following formula measures user u's interest in item i in the UserCF algorithm:

$$p(u,i) = \sum_{v \in S(\mathrm{u,k}) \cap \mathrm{N(i)}} w_{uv} r_{vi} \tag{3}$$

where $S(u,K)$ contains K users whose interests are closest to those of user u, $N(i)$ is the set of users who have acted on item i, w_{vi} is the interest similarity between user u and user v, and r_{vi} represents user v's interest in item i. Defects of UserCF algorithm

In fact, the UserCF algorithm is rarely used in the industry, and the ItemCF algorithm is used more often. The main defects of UserCF are as follows: firstly, with the increase of the number of users, the time complexity of calculating the interest similarity between two pairs of all users will become larger and larger, and it is proportional to the square of the number of users. Secondly, it is difficult for the UserCF algorithm to make a convincing explanation for the recommended results.

2.1.3 Item-based Collaborative Filtering

The Item-based Collaborative filtering algorithm [27] is similar to the User-based Collaborative filtering algorithm, which only interchanges items and users. Based on the relationship between items, the model recommends similar items to users. After calculating the similarity between items, we can get the degree of association between different items, so as to recommend the items with high degree of association with the items that the target users have browsed. [26] In short, if multiple users purchase product 1 and product 3 at the same time, then the correlation between product 1 and product 3 is high. When user 3 also purchases product 3, it can be inferred that he also needs to purchase product 1.

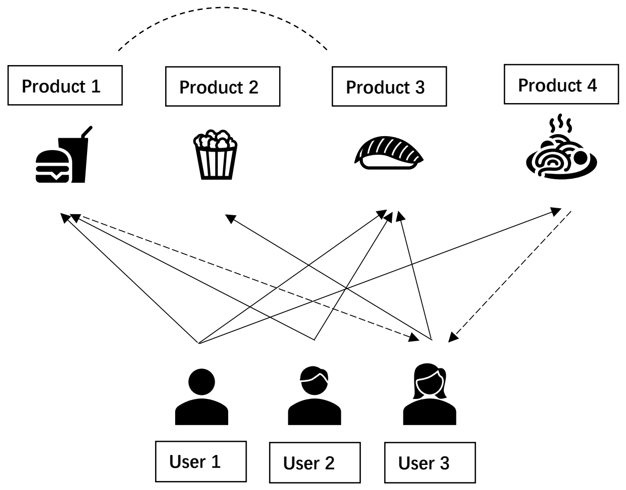


Fig. 3. Schematic diagram of item-based CF

The article based collaborative filtering algorithm mainly includes two steps:

1) Calculate the similarity between items.
2) According to the similarity of items and the user's historical behavior, generate a recommendation list of target users.

We use the following formula to define the similarity of items:

$$\mathrm{W_{ij}} = \frac{|N(i) \cap N(j)|}{|N(i)|} \tag{4}$$

where $|N(i)|$ is the number of users who like item i, $|N(j)|$ is the number of users who like item j, and $|N(i) \cap N(j)|$ is the number of users who like both item i and item j. It can be seen from the above definition that in collaborative filtering, two items are similar because they are liked by many users at the same time. The higher the similarity of two items is, the more people like the two items at the same time.

To avoid recommending popular items, we can use the following formula:

$$\mathrm{W_{ij}} = \frac{|N(i) \cap N(j)|}{\sqrt{|\mathrm{N(i)}||\mathrm{N(j)}|}} \tag{5}$$

This formula penalizes the weight of item j, thus reducing the possibility that popular items will be similar to many items.

After getting the similarity between items, ItemCF calculates user u's interest in an item j through the following formula:

$$p(u,j) = \sum_{i \in N(u) \cap S(j,k)} w_{ij} r_{ui} \tag{6}$$

where $N(u)$ is the set of items that users like, $S(i,K)$ is the set of K items most similar to item i, w_{ji} is the similarity of items j and i, and r_{ui} is the interest of user u in item i. The meaning of the formula is that the more similar the current item is to the item that the user was interested in, the more likely it is to get a higher ranking in the user's recommendation list.

In ItemCF, two items can produce similarity because they appear in the list of interest items of multiple users at the same time, so users will contribute to the similarity of two items in their interest list. However, the contribution of different users is not the same. Active users, compared with inactive users, contribute less to the similarity between items. John S. Breese proposed the concept of IUF (inverse user frequency)[ ] in his paper. The calculation formula is as follows:

$$w_{ij} = \frac{\left|\sum_{u\in N(i)\cap N(j)} \frac{1}{\log(1+N(u))}\right|}{\sqrt{|N(i)||N(j)|}} \tag{7}$$

Now we can compare UserCF with ItemCF. In terms of performance, UserCF is suitable for scenarios with a small number of users, while ItemCF is suitable for scenarios where the number of items is significantly less than the number of users. In terms of real-time, the new behavior of users in UserCF does not necessarily lead to the immediate change of recommendation results, but the new behavior of users in ItemCF will certainly lead to the real-time change of recommendation results. From the perspective of cold start, UserCF is mainly faced with the cold start problem of new users, but it can immediately recommend new online items; ItemCF is on the contrary, new users can recommend an item to them as long as they have an action on it. However, for new online items, ItemCF can't recommend them to users immediately.

2.1.4 Model-based Collaborative Filtering

Model based collaborative filtering [29] is the most popular type of collaborative filtering. It is not a specific algorithm model, but a general term of many algorithms. We will not explain each model in detail here, but mainly classify and summarize its ideas. Model based collaborative filtering is a model that uses scoring information to learn and predict. The main idea is to use attributes to build the relationship between users and items. The attributes represent the potential characteristics of users and items in the system, such as the categories that users like and items belong to.

For this problem, we usually use the idea of machine learning to model and solve, the mainstream methods can be divided into: association algorithm, clustering algorithm, classification algorithm, regression algorithm, matrix decomposition and hidden semantic model. Then, we will introduce them separately.

1. Association algorithm

Generally, we can find the frequent itemsets or sequences in all the items that users purchase, and do frequent set mining to find the frequent n itemsets or sequences of related items that meet the support threshold. Commonly used association recommendation algorithms include apriori, FP Tree, PrefixSpan and so on.

2. Clustering algorithm

Using clustering algorithm to do collaborative filtering is similar to the previous collaborative filtering based on users or items. We can cluster according to users or items based on a certain distance measure. Common clustering recommendation algorithms include K-means, birch, DBSCAN and spectral clustering, etc.

3. classification algorithm

If we divide the score into several segments according to the user's score, the problem becomes a classification problem. For example, the most direct way is to set a score threshold, which can be a binary or multi classification problem. Common classification recommendation algorithms include logistic regression and naive Bayes, both of which are highly explanatory.

4. regression algorithm

It seems more natural to use regression algorithm for collaborative filtering than classification algorithm. Our score can be a continuous value rather than a discrete value. In the process of building the model, we can use the gradient descent algorithm to update and optimize the weight of each attribute. Common regression recommendation algorithms include ridge regression, regression tree and support vector regression.

5. matrix decomposition

Matrix factorization is widely used in collaborative filtering. The traditional SVD requires that the matrix must be dense without missing data. However, our user item rating matrix is a typical sparse matrix, so it is more complex to use the traditional SVD directly for collaborative filtering. At present, the mainstream matrix factorization recommendation algorithms are mainly some variants of SVD, such as FunkSVD, BiasSVD and SVD++. The biggest difference between these algorithms and traditional SVD is that the user item matrix M is decomposed into two low dimensional matrices instead of three.

6. hidden semantic model

The general idea of the implicit semantic model is: it is assumed that each user has his own favorite tags, and the granularity of these tags can be different. If the user's interest in different tags is known, and the proportion of each tag of each item is known, the user potential factor matrix and the item potential factor matrix can be obtained. Using these two matrices, we can calculate the target user's preference for specific items, and then give the recommendation results.

2.1.5 Hybrid Recommendations

Hybrid recommendation technology is committed to the mixed use of one or more recommendation algorithms in order to achieve better recommendation effect. As a research hotspot in recent years, the most common way of hybrid recommendation is to integrate clustering, association rule mining and other technologies in collaborative filtering technology to solve the problem of cold start. The common structures are as follows [5]

1) Weighted structure: the results of a variety of recommendation techniques are weighted and mixed to complete the recommendation. The simplest weighting method is linear mixing. In practical application, the accuracy of different recommendation systems can be compared, so as to adjust the weight of each system.
2) Transformation structure: this structure needs to be adjusted according to the actual situation. First, one

recommendation technology is used. If this technology fails to meet the standard, another recommendation technology can be considered.

3) Merge structure: a variety of recommendation techniques are used at the same time to provide users with a variety of references. This structure is more user-friendly, and it is also very popular in practical applications.
4) Waterfall structure: this structure consists of two stages. First, a recommendation technique is used to screen out the general candidate results, and then these candidate results are used as the input of the second recommendation technique to further accurately recommend the results. This structure has a high fault tolerance rate for the first recommendation technology, and the accuracy of the recommendation results is greatly improved.

2.2 Deep Learning based Recommendation Method

2.2.1 Multi-Layer Perceptron

In the deep learning recommendation system, the most basic structure is multilayer perceptron (MLP). [30] MLP is like a black box to deeply combine and cross the input features, and then output the prediction of interest index. Other deep recommendation models are generated on the basis of multi-layer perceptron. So we can say that MLP is the most basic and core structure in the whole deep recommendation system.

MLP is also called artificial neural network (ANN). In addition to the input and output layer, it can have multiple hidden layers in the middle. [31] The simplest MLP only contains one hidden layer, which is a three-layer structure. Its structure is shown in Fig. 4 .

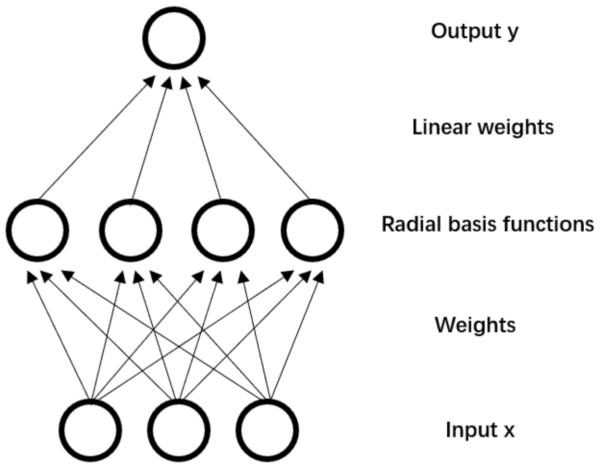


Fig. 4. MLP with one hidden layer

Fig. 4 shows that the MLP layers are fully connected. The bottom layer of multilayer perceptron is the input layer, the middle layer is the hidden layer, and the last layer is the output layer. Assuming that the input layer is represented by vector x, the output of the hidden layer is $F(W_1x + B_1)$, where W_1 is the weight, B_1 is the bias, and F is the activation function. Using activation function, the nonlinear factors can be added, so that neural network can approximate any nonlinear function. In MLP model, there are two kinds of activation functions: sigmoid and tanh.

1.sigmod:

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{8}$$

The derivative is:

$$\sigma'(x) = \sigma(x)(1 - \sigma(x)) \tag{9}$$

Sigmoid function is also called logistic function, its output range is (0, 1), that is to say, any variable can be mapped between 0 and 1.

2.tanh:

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{10}$$

The derivative is:

$$(\tanh x)' = \operatorname{sech}^2 x = 1 - \tanh^2 x \tag{11}$$

The value range of tanh is $[-1, 1]$. When the feature difference is obvious, its effect will be very good, and it will continue to expand the feature effect in the cycle process.

Finally, the output layer. The hidden layer to the output layer can be regarded as a multi category logistic regression, that is, softmax regression, so the output of the output layer is $\mathrm{softmax}(W_2X_1 + B_2)$, where X_1 represents the output $F(W_1x + B_1)$ of the hidden layer. In the recommendation problem, the output of the output layer is the score of recommendation degree. The closer the score is to 1, the higher the recommendation degree is. On the contrary, the lower the recommendation degree is.

2.2.2 Convolutional Neural Networks

Convolutional neural networks (CNN) have been successfully developed in the field of computer vision. Subsequently, many studies have used CNN for news recommendation and proved to achieve good results. There are two main functions of CNN in news recommendation.

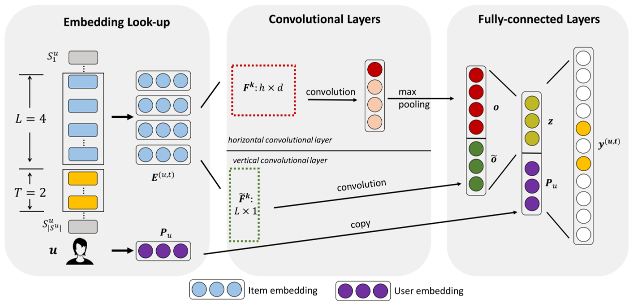


Fig. 5. Caser model: capture changes in user interests by using horizontal filters and vertical filters

1. Capture changes in user interests over time. Traditional content-based recommendation methods usually score the similarity between user representation and item representation in the same feature space, ignoring changes of users' interests. Through a 3-D tensor representation, CNN can capture the user's interest spatially and temporally and have a good performance in the domain of session-based next item recommendation. Kumar, V. et al. [32] propose a recommendation model which uses semantic similarity between words of the title and text of the news

in user's history and news considered for recommendation as input to a 3-D CNN, in order to extract the temporal news reading pattern of the users. Park, K. et al. [33] use CNN model to capture user preferences and to personalize recommendation results. Tang, J. and K. Wang propose the Caser model [34] which regards item embedding matrix generated by items in the user's history as "image" and processes them by using horizontal filters and vertical filters, as shown in Fig. 5. In the training, it needs to slide to select a sequence of length L in the user history to generate a set of samples. In this model, horizontal filters can be trained to capture union-level patterns and vertical filters can be trained to capture point-level sequential patterns. But it also has some limitations because it cannot distinguish the number of occurrences of an important feature and the location where it occurs. In addition, Caser network is too shallow to fail when modeling complex relationships and long-term dependencies. Therefore, Yuan, F. et al. [35] propose a simple and effective generative model formed of a stack of holed convolutional layers that is capable of learning high-level representation from both short- and long-range item dependencies.

2. Modeling news representation. Inspired by the application of CNN in the field of computer vision and natural language processing, CNN can be also applied to extract text features and obtain news representation. Wang, H. et al. [10] propose Deep Knowledge-Aware Network (DKN) model. As shown in Fig. 6, its key component KCNN is a multi-channel CNN that combines the word embedding, entity embedding and context embedding of the news title. Wu, C. et al. [36] use CNN to extract the features of the word embedding representations of the news title and body, and then combine with the category embedding to get the news representation. There are some similar ways to learn news representation using CNN [37] [38].

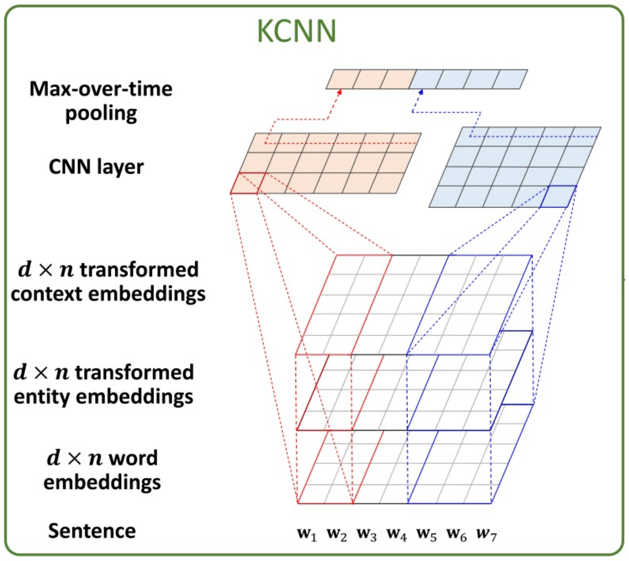

KCNN
Max-over-time pooling
CNN layer
$d \times n$ transformed context embeddings
$d \times n$ transformed entity embeddings
$d \times n$ word embeddings
Sentence
w_1 w_2 w_3 w_4 w_5 w_6 w_7

Fig. 6. Knowledge-aware CNN

2.2.3 Recurrent Neural Network

In 2015, Recurrent neural network (RNN) was applied to session-based recommendation system for the first time [39] and its structure is shown in Fig. 7. Hidasi, B., et al. modified the basic GRU in order to fit the recommendation task better by introducing session-parallel mini-batches, mini-batch based output sampling and ranking loss function. In news recommendation, RNN is mainly used to extract reading interests from the sequence of user's behaviors and generates user representation. Park, K., et al. improve the session-based RNN model for news recommendation [33]. Okura, S., et al. [40] use a variant of a denoising autoencoder to represent the articles and generate user representations by using RNN with browsing histories as input sequences. Khattar, D., et al. [11] use also an attention-based recurrent layer to learn user reading history. An, M., et al. [38] propose the LSTUR model which uses GRU network to learn user short-term representation from the representation of news their recently browsed. In the DAN model, ARNN as sequential information extractor is an attention-based LSTM, whose aim is also to capture the user's history sequential features from the user's history readings. RNN has two main structures: LSTM [41] and GRU [42]. It has been proved that GRU performs better than LSTM in almost all tasks [43]. In news recommendation, some researches have obtained the same result. In [40] and [44], GRU outperformed LSTM on the task of learning user representation.

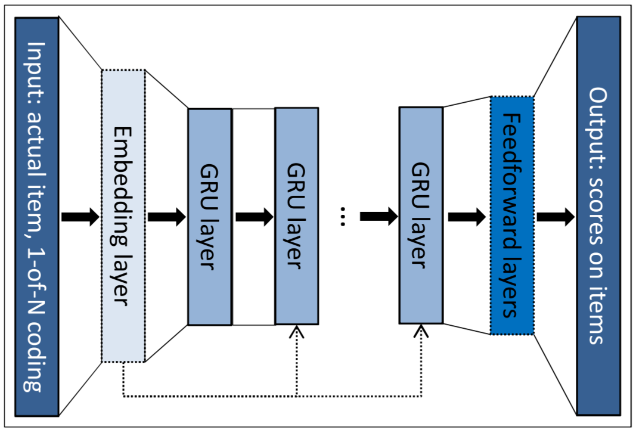


Fig. 7. General architecture of GRU4Rec

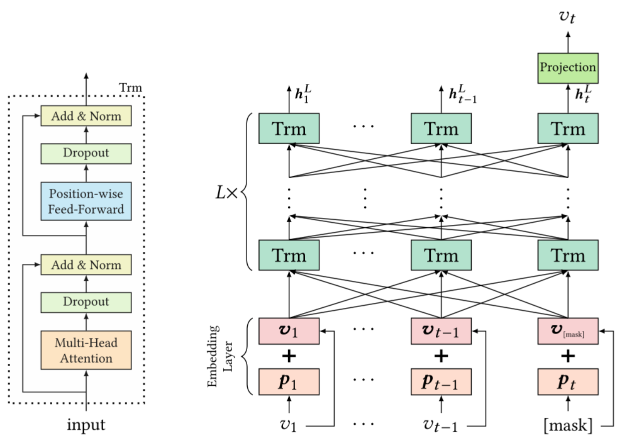
Both RNN and CNN can process user browsing records and capture user interest over time, but CNN often requires storage of users' complete browsing records on online news platforms, which may bring huge challenges to the storage and may cause heavy latency. Besides, RNN-based method performs better when data sets contains considerable sequential patterns.

2.2.4 Attention Mechanism

The attention mechanism is also very important and commonly used in the news recommendation model. It can be used as a component combined with CNN or RNN to extract key information, or it can be used for recommendation independently based on the Transformer model [45].

In the news profile aspect, the attention layer used to be combined with CNN, in order to extract the critical words, entities and events from news articles. In the [36], the authors firstly use CNN to process word embeddings of news title and topic, and then add a word-level attention network to select important words within the context of each news title. In the user profile aspect, the attention mechanism is used in combination with RNN or used alone, to extract user interests and obtain user representation. In the DKN model mentioned above [10], the attention net was used for user interest extraction from the news representation by KCNN. The same method is used in [36] and [37]. In the HRAM [11], the user history component is implemented by an attention-based recurrent network. The DAN model [38] contains two attention-based modules, the attention-based RNN model to find the sequential features of users' behaviors, and the attention-based neural network model to discover features of user's interest.

There are also some studies based on the Transformer model only, without CNN and RNN. The Fine-Grained Deep Knowledge-Aware Network [12] proposed by Jie Gao, X. at al. (2018) has four-level self-attention modules: word-level self-attention, item-level self-attention, user-level self-attention and multi-head attention in which word- and item-level self- attention are used for news representation, user-level self-attention is used for user representation and multi-head attention is used to concatenate history and candidate news. Experiments on real-word data have shown that performance is improved by 1.8% on the DKN model. As shown in Fig. 8, to address the limitations of the uni-directional sequence model, Sun, F., et al. (2019) [46] of Alibaba Group propose BERT4Rec model which employs the deep bidirectional self-attention to model user behavior sequences. Besides, Wu, C., et al. (2019) [47] multi-head self-attention to learn contextual representations of words by capturing their interactions in news encoder and to learn the representations of users from their browsed news in user encoder.



(a) Transformer Layer. (b) BERT4Rec model architecture.

Fig. 8. BERT4Rec

2.2.5 Graph Neural Network

News articles are usually highly concentrated and full of knowledge facts and the existing recommendation methods cannot find the knowledge-level connections between news, so the recommendation results are limited to simple patterns. Therefore, DKN [10] integrates the knowledge graph into the deep learning model to learn News presentation. In the DKN model, they use the existing entity linking technology [48] [49] to link the mentions in the text with the predefined entities in the knowledge graph; then use these identified entities to construct subgraphs, and extract their relationships from the original knowledge graph; since the knowledge subgraph obtained at this time is sparse and lacks diversity, the subgraph is expanded to all entities within one hop of the identified entity; after obtaining the knowledge graph, the entity embedding representation will be calculated by using the existing scoring functions [50], [51], [52], [53]. Liu, D., et al. [54] enhance the existing knowledge graph from three aspects: (1) adding a new group of entities for recording topic context information; (2) adding collaborative edges between entities based on users' click behaviors and co-occurrence in news articles; and (3) removing news-irrelevant relations. Joseph, K. and H. Jiang (2019) [55] propose an algorithm to compute the shortest distance between named entities, across news articles. Ren, J., Long, J., & Xu, Z (2019) [56] generate graph embeddings of the nodes using node2vec, and then calculate the user-news relatedness. In most current methods, the entity embedding representation is calculated based on the generated knowledge graph and then integrated into the news representation.

2.3 Dataset

2.3.1 MovieLens

MovieLens [2] contains rating data of multiple users on multiple movies, including movie information and user information..As a classic dataset of recommendation, which has been used and tested by many famous papers, it has several versions,, corresponding to different data volume, which can meet the needs of different models.

2.3.2 Jester

Jester joke [3] is a website that recommends and shares jokes online. The dataset was released by Ken Goldberg from the University of California, Berkeley. It has 73496 users rating 100 jokes for 4.1 million times. The scoring range is a continuous real number from - 10 to 10.

2.3.3 Book-Crossings

Book-Crossing [4] is the ratings of books by users of the on-line Book-Crossing community, which Cai-Nicolas Ziegler collected from Book Crossing in 2004 using a crawler program. The age and other attributes of these users are stored anonymously for analysis.

2.3.4 Last.fm

Last.fm [5] is a recommended dataset for music. For each user, the dataset records the user's basic information (age, gender, nationality, etc.), as well as the sequence of songs the user listens to. It is the representative of implicit feedback data set with context information.

2. https://grouplens.org/datasets/movielens/
3. http://eigentaste.berkeley.edu/dataset/
4. http://www2.informatik.uni-freiburg.de/ cziegler/BX/
5. https://grouplens.org/datasets/hetrec-2011/

2.3.5 *Wikipedia*

Wikipedia[6] is an encyclopedia co-authored by Wikipedia users. Now it has been widely used in social network analysis and Wikipedia user behavior research. For example, a user's editing-page behavior can be regarded as an implicit review of following this page with interest, and it is very helpful to analyze users' interest models for reading news by using Wikipedia thesaurus.

2.3.6 *Netflix*

Netflix [7] is the official data set used in the Netflix Prize competition. The data consists of about 100 million movie ratings, and the goal is to predict missing entries in the movie-user rating matrix.

2.3.7 *MIND*

Constructed from the user click logs of Microsoft News, MIND [57] contains 1 million users and more than 160k English news articles, each of which has rich textual content such as title, abstract and body.

3 Related works of Federated Learning System

Federated learning is proposed by Google in 2016 [58], [59], [60]. Their main idea is to build machine learning models based on data sets that are distributed across multiple devices while preventing data leakage.

3.1 Definition and formulation

Define N data owners $\{\mathcal{F}_1, \ldots, \mathcal{F}_N\}$, all of whom wish to train a machine learning model by consolidating their respective data $\{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$. A conventional method is to put all data together and use $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_N$ to train a model $\mathcal{M}_{SUM}$. A federated learning system is a learning process in which the data owners collaboratively train a model $\mathcal{M}_{FED}$, in which process any data owner $\mathcal{F}_i$ does not expose its data $\mathcal{D}_i$ to others. In addition, the accuracy of $\mathcal{M}_{FED}$, denotes as $\mathcal{V}_{FED}$ should be very close to the performance of $\mathcal{M}_{SUM}$, $\mathcal{V}_{SUM}$. Formally, let δ be a non-negative real number, if

$$|\mathcal{V}_{FED} - \mathcal{V}_{SUM}| < \delta \tag{12}$$

we say the federated learning algorithm has δ-accuracy loss.

The federated learning process aims to learn this model under the constraint that device-generated data is stored and processed locally, with only intermediate updates being communicated periodically with a central server. In particular, the goal is typically to minimize the following objective function:

$$\min_w F(w), \text{ where } F(w) := \sum_{k=1}^{N} p_k F_k(w). \tag{13}$$

Here, N is the total number of devices, $p_k \geq 0$ and $\sum_k p_k = 1$, and F_k is kth local objective function for the kth device. The local objective function is often defined as the empirical risk over local data, i.e., $F_k(w) = \frac{1}{n_k} \sum_{j_k=1}^{n_k} f_{j_k}(w; x_{j_k}, y_{j_k})$, where n_k is the number of samples available locally. The user-defined term p_k specifies the relative impact of each deviece, with two natural settings being $p_k = \frac{1}{n}$ or $p_k = \frac{n_k}{n}$, where $n = \sum_k n_k$ is the total number of samples. We will reference problem throughout the article, but, as discussed below, we note that other objectives or modeling approaches may be appropriate depending on the application of interest.

3.2 Categorization and architectures

In this section, we focus on the categorization of federated learning. In general, a federated learning system can be classified following a few aspects: distributing data, the model, privacy frameworks, communication architectures, etc. With the difference of data amount, federated learning system can be categorized into two different types: *private* and *public*.

- **Private federated learning systems** have few amounts of entities, each with huge amounts of data and computation power. The challenge that private federated learning systems have to overcome is how to transfer computations to data centers under certain constraints.
- **Public federated learning systems** have a large number of entities but they have small quantities of data and computation power.

In [20], another categorization method is proposed. A federated learning can be classified into three different types: *Horizontal federated learning*, *Vertical federated learning* and *Federated transfer learning*. In the following, we will discuss in detail these three different types of federated learning systems.

3.2.1 *Horizontal Federated Learning*

Horizontal federated learning, or sample-based federated learning, is introduced in the scenarios that data sets share the same feature space but different in samples. For example, two regional banks may have very different user groups from their respective regions, and the intersection set of their users is very small. However, their business is very similar, so the feature spaces are the same. Shokri and Shmatikov proposed a collaboratively deep learning scheme where participants train independently and share only subsets of updates of parameters [61]. In 2017, Google proposed a horizontal federated learning solution for Android phone model updates [60]. In that framework, a single user using an Android phone updates the model parameters locally and uploads the parameters to the Android cloud, thus jointly training the centralized model together with other data owners. A secure aggregation scheme to protect the privacy of aggregated user updates under their federated learning framework is also introduced [62]. Phong et al. uses additively homomorphic encryption for model paramter aggregation to provide security against the central server [63].

3.2.2 *Vertical Federated Learning*

Vertical federated learning or feature-based federated learning is applicable to the cases that two data sets share the

6. https://www. wikipedia.org/
7. http://archive.ics.uci.edu/ml/datasets/Netflix+Prize

TABLE 1
1 column table

Dataset	Field	Users	Items	Ratings	Density	Rating Scale
MovieLens 1M	Movie	6040	3883	1000209	4.26%	[1-5]
MovieLens 10M	Movie	69,878	10,681	10,000,054	1.33%	[0.5-5]
MovieLens 20M	Movie	138,493	27,278	20,000,263	0.52%	[0.5-5]
Jester	Joke	124,113	150	5,865,235	31.50%	[-10,10]
Book-Crossings	Book	92,107	271,379	1,031,175	0.0041%	[1,10], and implicit
Last.fm	Music	1892	17632	92,834	0.28%	Play Counts
Wikipedia	Webpage	5,583,724	4,936,761	417,996,366	0.0015%	Iteractions
Netflix	Movie	2,649,429	17,770	100,480,507	2.637%	[1,5]

same sample ID space but differ in feature space. For example, consider two different companies in the same city, one is a bank, and the other is an e-commerce company. Their user sets are likely to contain most of the residents of the area, so the intersection of their user space is large. However, since the bank records the user's revenue and expenditure behavior and credit rating, and the e-commerce retains the user's browsing and purchasing history, their feature spaces are very different. Suppose that we want both parties to have a prediction model for product purchase based on user and product information.

3.2.3 Federated Transfer Learning

Federated Transfer Learning applies to the scenarios that the two data sets differ not only in samples but also in feature space. Consider two institutions, one is a bank located in China, and the other is an e-commerce company located in the United States. Due to geographical restrictions, the user groups of the two institutions have a small intersection. On the other hand, due to the different businesses, only a small portion of the feature space from both parties overlaps. In this case, transfer learning techniques can be applied to provide solutions for the entire sample and feature space under a federation (Figure2c). Specially, a common representation between the two feature space is learned using the limited common sample sets and later applied to obtain predictions for samples with only one-side features.

3.3 Typical Process

A basic and widely used framework is Federated Averaging (FedAvg) [146] proposed in 2016, as shown in Figure 2a. In each iteration, the server first sends the current global model to the selected parties. Then, the selected parties update the global model with their local data. Next, the updated models are sent back to the server. Last, the server averages all the received local models to get a new global model. FedAvg repeats the above process until reaches the specified number of iterations. The global model of the server is the final output.

While FedAvg is a centralized FL framework, SimFL, proposed by Li et al. [124], represents a decentralized FL framework. In SimFL, no trusted server is needed. In each iteration, the parties first update the gradients of their local data. Then, the gradients are sent to a selected party. Next, the selected party use its local data and the gradients to update the model. Last, the model is sent to all the other parties. To ensure fairness and utilize the data from different parties, every party is selected for updating the model for about the same number of rounds. SimFL repeats a specified number of iterations and outputs the final model.

3.4 Technical challenge

3.4.1 Challenge 1: Expensive Communication

Communication is a critical bottleneck in federated networks, which, coupled with privacy concerns over sending raw data, necessitates that data generated on each device remain local. Indeed, federated networks are potentially comprised of a massive number of devices, e.g., millions of smart phones, and communication in the network can be slower than local computation by many orders of magnitude [64], [65]. In order to fit a model to data generated by the devices in the federated network, it is therefore necessary to develop communication-efficient methods that iteratively send small messages or model updates as part of the training process, as opposed to sending the entire dataset over the network. To further reduce communication in such a setting, two key aspects to consider are: (i) reducing the total number of communication rounds, or (ii) reducing the size of transmitted messages at each round.

3.4.2 Challenge 2: Systems Heterogeneity

The storage, computational, and communication capabilities of each device in federated networks may differ due to variability in hardware (CPU, memory), network connectivity (3G, 4G, 5G, wifi), and power (battery level). Additionally, the network size and systems-related constraints on each device typically result in only a small fraction of the devices being active at once, e.g., hundreds of active devices in a million-device network [66]. Each device may also be unreliable, and it is not uncommon for an active device to drop out at a given iteration due to connectivity or energy constraints. These system-level characteristics dramatically exacerbate challenges such as straggler mitigation and fault tolerance. Federated learning methods that are developed and analyzed must therefore: (i) anticipate a low amount of participation, (ii) tolerate heterogeneous hardware, and (iii) be robust to dropped devices in the network.

3.4.3 Challenge 3: Statistical Heterogeneity

Devices frequently generate and collect data in a non-identically distributed manner across the network, e.g., mobile phone users have varied use of language in the context of a next word prediction task. Moreover, the number of data points across devices may vary significantly, and there may be an underlying structure present that captures the relationship amongst devices and their associated distributions. This data generation paradigm violates frequently-used independent and identically distributed (I.I.D.) assumptions in distributed optimization, increases the likelihood of stragglers, and may add complexity in terms of modeling, analysis, and evaluation. Indeed, although the canonical federated learning problem aims to learn a single global model, there exist other alternatives such as simultaneously learning distinct local models via multi-task learning frameworks [67]. There is also a close connection in this regard between leading approaches for federated learning and meta-learning [68]. Both the multi-task and meta-learning perspectives enable personalized or device-specific modeling, which is often a more natural approach to handle the statistical heterogeneity of the data.

3.4.4 Challenge 4: Privacy Concerns

Finally, privacy is often a major concern in federated learning applications. Federated learning makes a step towards protecting data generated on each device by sharing model updates, e.g., gradient information, instead of the raw data [69], [70], [71]. However, communicating model updates throughout the training process can nonetheless reveal sensitive information, either to a third-party, or to the central server [72]. While recent methods aim to enhance the privacy of federated learning using tools such as secure multiparty computation or differential privacy, these approaches often provide privacy at the cost of reduced model performance or system efficiency. Understanding and balancing these trade-offs, both theoretically and empirically, is a considerable challenge in realizing private federated learning systems.

3.5 Current work

In this section, we summarize and compare the existing studies on FL. We summarize existing popular and state-of-the-art research work, as shown in Table 1. From Table 1, we have the following four key findings.

First, most of the existing studies consider a horizontal data partitioning. We conjecture a part of the reason is that the experimental studies and benchmarks in horizontal data partitioning are relatively ready than vertical data partitioning. However, vertical FL is also common in real world, especially between different organizations. Vertical FL can enable more collaboration between diverse parties. Thus, more efforts should be paid to vertical FL in order to fill the gap.

Second, most studies consider exchanging the raw model parameters without any privacy guarantees. This may not be right if more powerful attacks on machine learning models are discovered in the future. Currently, the mainstream methods to provide privacy guarantees are differential privacy and cryptographic methods (e.g., secure multi-party computation and homomorphic encryption). Differential privacy may influence the final model quality a lot. Moreover, the cryptographic methods bring much computation and communication overhead and may be the bottleneck of FLSs. We look forward to a cheap way with reasonable privacy guarantees to satisfy the regulations.

Third, the centralized design is the mainstream of current implementations. A trusted server is needed in their settings. However, it may be hard to find a trusted server especially in the cross-silo setting. One naive approach to remove the central server is that the parties share the model parameters to all the other parties and each party also maintains the same global model locally. This method bring more communication and computation cost compared with the centralized setting. More studies should be done for practical FL with the decentralized architecture.

Last, the main research directions (also the challenging) of FL are to improve the effectiveness, efficiency, and privacy, which are also three important metrics to evaluate an FLS. Meanwhile, there are many other research topics on FL such as fairness and incentive mechanisms. Since FL is related to many research areas, we believe that FL will attract more researchers and we can see more interesting studies in the near future.

Table 1: Comparison among existing published studies. LM denotes Linear Models. DM denotes Decision Trees. NN denotes Neural Networks. CM denotes Cryptographic Methods. DP denotes Differential Privacy.

FL Studies	main area	data partitioning	model implementation	privacy mechanism	communication architecture	remark
FedAvg [146]			NN			
FedSVRG [110]			LM			
FedProx [123]			LM, NN			
SCAFFOLD [104]			LM, NN			
FedNova [209]			NN			
Per-FedAvg [62]		horizontal	NN			
pFedMe [54]			LM, NN			SGD-based
IAPGD, AL2SGD+ [84]			LM		centralized	
IFCA [74]			LM, NN	＼		
Agnostic FL [152]			LM, NN			
FedRobust [172]			NN			
FedDF [130]			NN			
FedBCD [137]	Effective Algorithms	vertical				
PNFM [233]		horizontal	NN			
FedMA [208]						NN-specialized
SplitNN [208]		vertical				
Tree-based FL [237]				DP	decentralized	
SimFL [120]		horizontal		hashing		
FedXGB [139]			DT			DT-specialized
FedForest [138]						
SecureBoost [46]		vertical				
Ridge Regression FL [159]		horizontal		CM		
PPRR [43]						LM-specialized
Linear Regression FL [180]		vertical	LM			
Logistic Regression FL [87]						
Federated MTL [188]					centralized	multi-task learning
Federated Meta-Learning [39]						meta-learning
Personalized FedAvg [96]			NN			
LFRL [132]						reinforcement learning
FBO [52]			LM	＼		Bayesian optimization
Structure Updates [111]						
Multi-Objective FL [245]						efficiency improvement
On-Device ML [94]						
Sparse Ternary Compression [183]			NN			
DPASGD [145]					decentralized	
Client-Level DP FL [71]						
FL-LSTM [147]				DP		
Local DP FL [22]			LM, NN			privacy guarantees
Secure Aggregation FL [25]			NN	CM		
Hybrid FL [200]	Practicality Enhancement	horizontal	LM, DT, NN	CM, DP		
Backdoor FL [17, 193, 207]						
Adversarial Lens [21]			NN			
Distributed Backdoor [221]						robustness and attacks
Image Reconstruction [69]						
RSA [116]			LM			
Model Poison [63]			LM, NN			
q-FedAvg [125]			LM, NN	＼	centralized	fairness
BlockFL [109]			LM			incentives
Reputation FL [101]						
FedCS [161]			NN			
DRL-MEC [212]						edge computing
Resource-Constrained MEC [211]			LM, NN			
FedGKT [88]			NN			
FedCF [15]	Applications		LM			collaborative filter
FedMF [35]						matrix factorization
FedRecSys [196]			LM, NN	CM		recommender system
FL Keyboard [86]			NN			natural language processing
Fraud detection [241]			NN			credit card transaction
FedML [89]		horizontal &vertical	LM, NN	＼	centralized &decentralized	
FedEval [36]			NN		centralized	general purpose benchmarks
OARF [91]			NN	CM,DP	centralized	
Edge AIBench [85]			＼		＼	
PerfEval [160]	Benchmarks					
FedReID [246]		horizontal	NN		centralized	
semi-supervised benchmark [236]				＼		targeted benchmarks
non-IID benchmark [134]					＼	
LEAF [30]			＼		centralized	datasets
Street Dataset [141]			＼		＼	

3.6 Platforms

Several platforms exist for FL as well. Thanks to the growth of federated learning, there are a lot of industries and research teams that research federated learning for product and research development [73]. There are plenty of platforms and architectures for FL, with more examples including [74], [75], [76], [77]. These platforms and architectures help refine federated learning better. One other architecture involves using blockchain on federated learning, where the authors of [78], [79], [80], and [81] propose an architecture incorporating both Blockchain and federated learning. By proposing this architecture, there will be no need for centralized training data, and devices could potentially get trained much faster. The architectures discussed is not only applicable for federated learning in general, but it can apply for other industries too.

- **PySyft**[8] - PySyft is mainly geared towards privacy. It handles private data from the models' training using federated learning within PyTorch, which is another library in the Python library.
- **Tensor Flow Federated (TFF)**[9] - TFF is another platform for federated learning. TFF, it provides users with a more flexible and open framework for their needs.
- **FATE (Federated AI Technology Enabler)**[10] - FATE is another open-source project geared towards FL. The platform was initiated by the Webank AI division. The goal of FATE is to provide a secured computing architecture, where a secure computing protocol is implemented based on encryption. FATE can support various federated learning architectures and machine learning algorithms, including logistic regression, transfer learning, etc. Through multiple upgrades, the platform was able to have a new tool that allowed for a more visual approach to FL: FATEBoard. The new upgrades also let the platform have FL modeling pipeline scheduling and life cycle management tools called FATEFlow.
- **Tensor/IO**[11] - This particular platform is a platform that brings the power of TensorFlow to mobile devices such as iOS, Android, and React native applications. While this platform does not implement any machine learning methods, the platform cooperates with TensorFlow to ease the process of implementing and deploying models on mobile phones. It can run on iOS and Android phones. The library also has choices of back-end programming languages the consumer can choose from when using this platform. The languages it supports are objective-c, Swift, Java, Kotlin, or JavaScript. A benefit of using this platform is that prediction can be done in as little as five lines of code.
- **Functional Federated Learning in Erlang (FFL-ERL)** - Erlang is a structured, dynamictyped programming language that has built-in parallel computing support, which is suitable for establishing real-time systems. This particular platform was proposed by the authors Gregor Ulm, Emil Gustavsson, and Mats Jirstran back in 2018. In the authors' work, they go through the mathematical implementation of their proposed platform. For testing, the authors actually generated an artificial data-set. This particular platform, unfortunately, does have a performance penalty [82].
- **CrypTen**[12] - CrypTen is actually a framework geared towards privacy preservation built on PyTorch. There are a few benefits to using this platform, especially for machine learning. One benefit is that the platform was made with real-world challenges in mind. So it has the potential to be applicable to a lot of real-world medical care challenges. Another benefit is that CrypTen is library-based. So it is easier for users to debug, experiment on, and explore different models. Currently, the CrypTen platform runs only on the Linux and Mac operating systems.
- **LEAF**[13] - LEAF is a framework for FL, multi-tasking, meta-learning, etc. This platform has several datasets available for experimentation. Caldas et al. proposed the framework LEAF back in 2019 [83]. Core components of LEAF consists of three components: the datasets, implementation references, and metrics.
- **FedLearner**[14] - Fedlearner is collaborative machine learning framework that enables joint modeling of data distributed between institutions.

4 The Overview of Recommendation Systems under Federated Learning Framework

With the explosive development of online services, products, and information resources, the phenomenon of "information overload" is becoming more and more serious, which brings a significant information burden to people. Recommendation system is able to cope with the information overload issue and realize the customized information service for each person. Currently, Recommendation system has played a significant role in many domains, such as e-commerce (Alibaba, Amazon), video service (Netflix, YouTube, and TikTok), online-music services (Pandora, Yahoo), Advertising (Google, Facebook), both the user and service providers benefit a lot from the personalized recommendation. However, the current collected data are privacy-sensitive, and any kind of privacy leakage can leads to very severe problems for both users and the service providers.

The ultimate goal of Recommendation System is to understand user preference better than users themselves [84]. Recommendation System needs to collect users' personal information and behavioral data for training process to achieve a comprehensive and profound user preference perception to predict users' next step behavior. The prediction quality varies depending on the scale, diversity, and timeliness of the data collected. However, these data quality factors are also positively related to privacy risks. This phenomenon leads to an unavoidable "privacy-personalization trade-off problem" [85]. As illustrated in Fig. 9, internal privacy threats come from a curious or malicious service provider. Driven by commercial interests, service providers may violate privacy contracts and collect user data without authorization or share data with third parties. Internal staff may also use access permissions to spy on users' privacy. External privacy threats mainly come from hacker attacks. Due to the lack of adequate data security protection, data

8. https://github.com/OpenMined/PySyft
9. https://github.com/tensorflow/federated
10. https://github.com/FederatedAI/FATE
11. https://doc-ai.github.io/tensorio/
12. https://crypten.ai/
13. https://leaf.cmu.edu/
14. https://github.com/bytedance/fedlearner

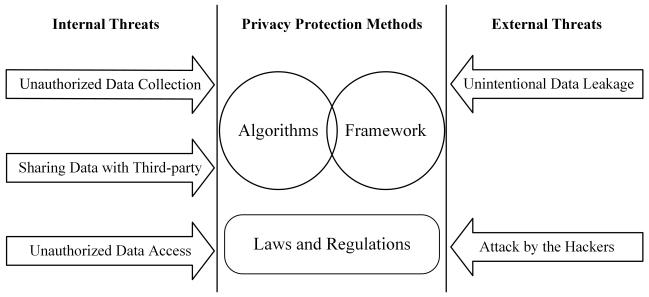


Fig. 9. Internal and External Privacy Threats to the RS and Available Countermeasures

is maliciously stolen. Sometimes there will be unconscious data leakages, such as data surveillance by law enforcement agencies or publicly anonymous data being de-anonymized by a third party.

Therefore, some researchers are considering combining the Recommendation System and the Federated Learning. Because the latter one has capability of connecting isolated data under the premise of ensuring data security and user privacy, thereby providing the possibility to train accurate and robust machine learning models. McMahan et al. [13] first introduced the term FL in 2016 and applied it to update smart-phones' voice recognition and text entry models. The Google team utilize the user's mobile phone to train the local user interaction data, which can continuously optimize the user's local language prediction model, and through the "Federated Averaging" algorithm (FedAvg), by averaging all stochastic gradient descent (SGD) result on user's terminal with a central server, a globally optimized model is obtained. Currently, the study of FL has been carried out in many application fields [22], [86], [87] related to privacy protection and data security. However, this domain still needs to be further explored, and in this subsection, we will present several current works by classifying them depending on the specific method utilized for recommendation stage. There are two main categories of them: Machine Learning method and Deep Learning method.

4.1 Machine Learning Based Federated Recommendation System

The mainstream Federated Recommendation framework employ the Machine Learning Based method such as the collaborative filters and matrix factorization to realize the recommendation. In the following part, we will present the current relevant work in chronological order.

In 2018, Fei Chen et al. [88] proposed a method of federated meta-learning framework for recommendation to share use information at thr model training algorithm level while preserving user privacy. They incorporated meta-learning into the decentralized training process as in federated learning. In this framework, meta-training proceeds naturally in a distributed manner, where each user has a specific model that is trained using local data. The model level training is performed on user devices, and the server only maintains and updates at the algorithm level. The server first sends parameters to a set of sampled users. Then each sampled user trains a user-specific model using current parameters received and evaluates the model on the corresponding device. Finally the server collects these users the test loss gradients to update the parameters. It is not necessary to upload any user data to the server in the process, instead only the parameters and test loss gradients are transferred in each episode. Their experimental results have shown that the two-level meta-learning approach outperformed both stand-alone models without pre-training and unified models with pre-training. Their proposed framework can keep the algorithm and model at a small scale while maintaining high capacities, compared to unified models that are possibly trained in the federated learning approach.

Then in 2019, Muhammad Ammad-ud-din et al. [89] introduced the first Federated Collaborative Filter (FCF) method. They have shown that federation of the collaborative filter is technically challenging. Their method aggregated user-specific gradient updates of the model weights from the clients to update the master model. In particularly, they derived a federated version of the widely used Collaborative Filter method for implicit feedback datasets [90]. However, the proposed method was able to be generalized and can be extended to recommendation scenarios where explicit rating information is available and can also be applied to a more generic class of matrix factorization models. Their FCF model is shown in Algorithm 1. More details can be found in the original paper. Their experimental results analysis demonstrated that the federated model achieves robust and stable solutions by incorporating an adaptive learning rate. The empirical evidence proved that the federated collaborative filter achieved statistically similar recommendation performance compared to the standard method. As a practical outcome, the results established that the federated model can provide similar quality of recommendations as the widely used standard collaborative filter while fully preserving the user's privacy

Algorithm 1 Federated Collaborative Filter

```
Input: M clients
  FL Server
  Initialize the parameters: $\mathbf{Y}$
  for each client $m \in M$ in parallel do
    $\nabla \mathbf{Y}^{(m)}$ = Clientupdate($\mathbf{Y}$)
    $Y = Y - \gamma_m \nabla \mathbf{Y}^{(m)}$ The gradient updating process
  end for
  FL Client
  function: Clientupdate($\mathbf{Y}$)
  update user factor
  compute item factor $\mathbf{Y}^(m)$ gradient: $\nabla \mathbf{Y}^(m)$
  return $\nabla \mathbf{Y}^(m)$ to surver
```

Time comes to 2020, more and more attention has been drawn to the domain of Federated Recommendation. Adrian Flanagan et al. [91] introduce the federated multi-view matrix factorization method that extended the federated learning framework to matrix factorization with multiple data sources. The proposed federated multi-view model is tested on three different datasets and they showed that including the side-information from both users and items increases recommendation performance compared to a standard federated Collaborative Filter. The multi-view approach provides a solution to the cold-start problem common to standard Collaborative Filter recommenders. Their experimental results established that the federated multi-view model can provide better quality of recommendations without comprising the user's privacy. Their recommendation process is shoen in Algorithm 2 and more detailed equations can be found in original paper.

Algorithm 2 Federated Multi-View Matrix Factorization

```
Input: $N_v$ items, $D_u$ user features, $K$ factors
Initialization: master model factor matrix $\mathbf{Q}, \mathbf{U}$, threshold $\theta$

  FL Server
  while True do
    Transmit $\mathbf{Q}, \mathbf{U}$ to users
    Transmit $\mathbf{Q} \longrightarrow$ Item Server
    Receive factor $\mathbf{Q}, \mathbf{U}$ gradients from users and item server
    if NumberGradientUpdate $>= \theta$ then
      Update
    end if
  end while

  FL User
  while True do
    Receive master model
    Generate recommendations
    Compute gradients and transmit to FL Server
  end while

  Item Server
  while True do
    Receive master model
    Compute gradients and transmit to FL Server
  end while
```

Similarly, Senci Ying proposed SharedMF [92], a privacy protection recommendation system model, which can not only complete the distributed recommendation task, but avoid privacy disclosure. Moreover, their model was based on secret sharing technology. Compared with the schemes based on homomorphic encryption, the running speed of their model is greatly improved. Furthermore, Vito Walter Anelli at al. [93] think that the users are entitled to choose the amount of sensitive information for sharing with the server. Due to this idea, they proposed FPL (short for Federated Pair-wise Learning) system, a federated factorization model for collaborative recommendation.Users participating in the federation process can decide if and to which extent they are willing to disclose their sensitive private data. FPL mainly leverages not-sensitive information to reach a competitive accuracy and, at the same time, respect a satisfactory balance between accuracy and privacy. Except for the approaches aforementioned, many researchers also have devoted themselves to this federated recommendation domain [94], [95].

The latest federated recommendation system is called FedRank, proposed by Vito Walter Anelli et al [96].Witnessing the growing concern about privacy, users might want to exploit their sensitive data and share only small fraction of it. In such situation, classic Collaborative filtering approaches are no more feasible. To overcome these problems, FedeRank is proposed, it is a novel recommendation framework that respects the FL paradigm. With FedeRank, private user feedback remains on user devices unless they decide to share it. Nevertheless, FedeRank ensures high-quality recommendations despite the constrained setting. In original paper [96], they have extensively studied the performance of FedeRank by comparing it with other state-of-the-art methods. On the one hand, the results' analysis suggests that centralized recommender systems are not performing at their best. Feeding recommendation systems with all the available feedback, without any filtering, may lead to a worse performance. On the other hand, the competitive results of FedeRank suggest that the FL-based algorithms show a recommendation quality that makes them suitable to be adopted on a massive scale.

4.2 Deep Learning Based Federated Recommendation System

As is known to all that the Graph neural network is widely used for recommendation to model high-order interactions between users and items. However, current Graph-based recommendation methods rely on centralized storage of user-item graphs and centralized model learning. Due to the privacy concern or risks, FedGNN was proposed [97]. Extensive experiments and analysis on six benchmark datasets show that FedGNN can achieve competitive results with existing centralized GNN-based recommendation methods and meanwhile protect user privacy. The framework of FedGNN is shown in 10. It mainly consists of a central server and a large number of user clients. The user client keeps a local subgraph that consists of the user interaction histories with items and the neighbors of this user. Each client learns the user/item embeddings and the GNN models from its local subgraph, and uploads the gradients to a central server. The central server is responsible for coordinating these user clients in the model learning process by aggregating the gradients received from a number of user clients and delivering the aggregated gradients to them.

4.3 Challenges and future directions

4.3.1 Algorithm-Level Challenges

Federated Deep Model for Recommendation. Deep recommendation models could cause severe problems when utilizing non-linear activation functions. Complex functions, e.g., tanh and relu activation functions, are not well supported. This limitation seriously affects the deep models' application in FedRec. For solving this problem, [98] utilized low degree polynomials as the approximation of activation functions. There exists a trade-off between the model performance and the degree of polynomial approximation. This work provides the polynomial approximations with the lowest degrees as possible for three common activation functions, i.e., ReLU, Sigmoid, and Tanh.

Federated Graph Model for Recommendation. Protecting the privacy of structure information in the graph is the main difficulty of federalizing the graph-based models. The Graph-based models for recommendations utilize the relation information between users and items to enrich their representations. The relation information is more complicated than the feature information. Different secure methods

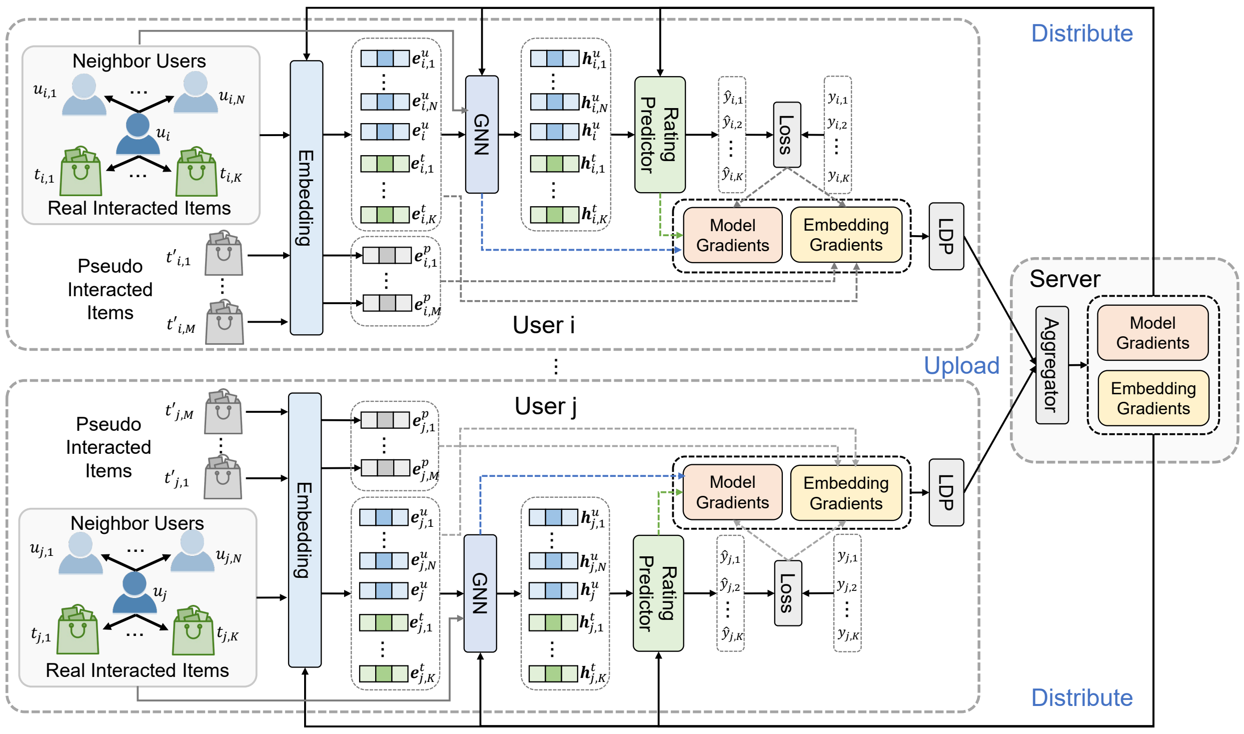


Fig. 10. The framework of the FedGNN approach

are adopted to protect the privacy of the graph in the present works. For instance, [99] utilized a graph sampling method to improve both the efficiency and privacy of the privacy-preserving association rules mining approaches. Users decide locally and privately, whether to become part of the sample. They are in control of their data and maintain sensitive item sets. Users with common interests are represented by the user groups. Neither the recommender nor other users know about the specific item sets of one particular user.

Federated Reinforcement Learning Model for Recommendation. The challenge of federalizing reinforcement learning models is to delicately design the state, action, and reward to catch the instant user interest and decide what to share among parties. Although reinforcement learning has an vital role in RecSys, its application in FedRec is still underexplored. Yet, there have been several works about federated reinforcement learning applied in other areas. [100] provided the lifelong federated reinforcement learning architecture for robots to perform lifelong learning of navigation in cloud robotic systems. A knowledge fusion algorithm and transfer learning approach are designed to fuse the robots' prior knowledge and make robots quickly adapt to the new environments.

4.3.2 System-Level Challenges

Federated learning is an active and ongoing area of research. Although recent work has begun to address the challenges discussed in Section 3.4, there are a number of critical open directions yet to be explored. In this section, we briefly outline a few promising research directions surrounding the previously discussed challenges (expensive communication, systems heterogeneity, statistical heterogeneity, and privacy concerns), and introduce additional challenges regarding issues such as productionizing and benchmarking in federated settings.

Extreme communication schemes. It remains to be seen how much communication is necessary in federated learning. Indeed, it is well-known that optimization methods for machine learning can tolerate a lack of precision; this error can in fact help with generalization [101]. While oneshot or divide-and-conquer communication schemes have been explored in traditional data center settings [102], [103], the behavior of these methods is not well-understood in massive or statistical heterogeneous networks. Similarly, one-shot/few-shot heuristics [104], [105] have recently been proposed for the federated setting, but have yet to be theoretically analyzed or evaluated at scale.

Communication reduction and the Pareto frontier. We discussed several ways to reduce communication in federated training, such as local updating and model compression. In order to create a realistic system for federated learning, it is important to understand how these techniques compose with one another, and to systematically analyze the trade-off between accuracy and communication for each approach. In particular, the most useful techniques will demonstrate improvements at the Pareto frontier—achieving an accuracy greater than any other approach under the same communication budget, and ideally, across a wide range of communication/accuracy profiles. Similar comprehensive analyses have been performed for efficient neural network inference [106], and are necessary in order to compare communication-reduction techniques for federated learning in a meaningful way.

Novel models of asynchrony. Two communication schemes most commonly studied in distributed optimization are bulk synchronous approaches and asynchronous approaches (where it is assumed that the delay is bounded). These schemes are more realistic in data center settings—where worker nodes are typically dedicated to the workload, i.e., they are ready to 'pull' their next job from the central node immediately after they 'push' the results of their previous job. In contrast, in federated networks, each device is often undedicated to the task at hand and most devices are not active on any given iteration. Therefore, it is worth studying the effects of this more realistic device-centric communication scheme—in which each device can decide when to 'wake up' and interact with the central server in an event-triggered manner.

Heterogeneity diagnostics. Recent works have aimed to quantify statistical heterogeneity through metrics such as local dissimilarity (as defined in the context of federated learning in [107] and used for other purposes in works such as [108], [109]) and earth mover's distance [110]. However, these metrics cannot be easily calculated over the federated network before training occurs. The importance of these metrics motivates the following open questions: (i) Do simple diagnostics exist to quickly determine the level of heterogeneity in federated networks a priori? (ii) Can analogous diagnostics be developed to quantify the amount of systems-related heterogeneity? (iii) Can current or new definitions of heterogeneity be exploited to further improve the convergence of federated optimization methods?

Granular privacy constraints. The definitions of privacy cover privacy at a local or global level with respect to all devices in the network. However, in practice, it may be necessary to define privacy on a more granular level, as privacy constraints may differ across devices or even across data points on a single device. For instance, Li et al. [68] recently proposed sample-specific (as opposed to user-specific) privacy guarantees, thus providing a weaker form of privacy in exchange for more accurate models. Developing methods to handle mixed (device-specific or sample-specific) privacy restrictions is an interesting and ongoing direction of future work.

Beyond supervised learning. It is important to note that the methods discussed thus far have been developed with the task of supervised learning in mind, i.e., they assume that labels exist for all of the data in the federated network. In practice, much of the data generated in realistic federated networks may be unlabeled or weakly labeled. Furthermore, the problem at hand may not be to fit a model to data as presented in (1), but instead to perform some exploratory data analysis, determine aggregate statistics, or run a more complex task such as reinforcement learning. Tackling problems beyond supervised learning in federated networks will likely require addressing similar challenges of scalability, heterogeneity, and privacy.

Productionizing federated learning. Beyond the major challenges discussed in this article, there are a number of practical concerns that arise when running federated learning in production. In particular, issues such as concept drift (when the underlying data-generation model changes over time); diurnal variations (when the devices exhibit different behavior at different times of the day or week) [111]; and cold start problems (when new devices enter the network) must be handled with care. We defer the readers to [66], which discusses some of the practical systems-related issues that exist in production federated learning systems.

5 Conclusion

Since the majority of the service providers intend to provide personalized recommendation such as news, music, video even products for their users, the recommendation systems has draw more attention than ever before. However, the data used for developing the recommendation system often violate the privacy of the users which is protected by the regulations like GDPR, it is very crucial to develop a method that take account of both the profit of the service providers and the users privacy. On the other hand, Federated learning proposed by Google in 2016 has became a standard protocol to cope with the privacy preserving concern. Thus, many researchers has devoted themselves to combining the recommendation method with the Federated Learning framework.

Acknowledgements

This work was partially supported by Sino-French Engineering School, Beihang University and Orange Beijing Lab.

References

[1] W. C. Hill, L. Stead, M. Rosenstein, and G. W. Furnas, "Recommending and evaluating choices in a virtual community of use," in *Proceedings of Human Factors in Computing Systems*, 1995, pp. 194–201.

[2] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the Conference on Computer Supported Cooperative Work*, 1994, pp. 175–186.

[3] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "word of mouth"," in *Proceedings of Human Factors in Computing Systems*, 1995, pp. 210–217.

[4] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[5] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, and J. Riedl, "Movielens unplugged: experiences with an occasionally connected recommender system," in *Proceedings of the 8th International Conference on Intelligent User Interfaces*, 2003, pp. 263–266.

[6] D. Billsus, C. Brunk, C. Evans, B. Gladish, and M. J. Pazzani, "Adaptive interfaces for ubiquitous web access," *Communications of the ACM*, vol. 45, no. 5, pp. 34–38, 2002.

[7] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," *ACM*, 2001.

[8] M. Deshpande and G. Karypis, "Item-based top- n recommendation algorithms," *Acm Trans.inf.syst*, vol. 22, no. 1, pp. 143–177, 2004.

[9] Z. F. Jiang, J. Yang, E. Hai-Hong, PCNamp, and C. Center, "A content-based recommendation algorithm with social tagging," *Computer engineering Software*, 2015.

[10] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: deep knowledge-aware network for news recommendation," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, 2018, pp. 1835–1844.

[11] D. Khattar, V. Kumar, V. Varma, and M. Gupta, "HRAM: A hybrid recurrent attention machine for news recommendation," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1619–1622.

[12] J. Gao, X. Xin, J. Liu, R. Wang, J. Lu, B. Li, X. Fan, and P. Guo, "Fine-grained deep knowledge-aware network for news recommendation with self-attention," in *2018 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2018, Santiago, Chile, December 3-6, 2018*, 2018, pp. 81–88.

[13] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 54, 2017, pp. 1273–1282.

[14] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, “Federated learning for mobile keyboard prediction,” *CoRR*, vol. abs/1811.03604, 2018.

[15] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, “Applied federated learning: Improving google keyboard query suggestions,” *CoRR*, vol. abs/1812.02903, 2018.

[16] Y. Knaan, “Under the hood of the pixel 2: how ai is supercharging hardware,” *Google AI*, 2017.

[17] H. Wang, M. Yurochkin, Y. Sun, D. S. Papailiopoulos, and Y. Khazaeni, “Federated learning with matched averaging,” in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[18] M. F. Dacrema, P. Cremonesi, and D. Jannach, “Are we really making much progress? A worrying analysis of recent neural recommendation approaches,” in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, pp. 101–109.

[19] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” *ACM Computing Surveys*, vol. 52, no. 1, pp. 5:1–5:38, 2019.

[20] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated machine learning: Concept and applications,” *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 12:1–12:19, 2019.

[21] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed, “Federated learning: A survey on enabling technologies, protocols, and applications,” *IEEE Access*, vol. 8, pp. 140 699–140 725, 2020.

[22] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *Institute of Electrical and Electronics Engineers Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.

[23] T. Qi, F. Wu, C. Wu, Y. Huang, and X. Xie, “Privacy-preserving news recommendation model learning,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, 2020, pp. 1423–1432.

[24] A. V. D. Oord, S. Dieleman, and B. Schrauwen, “Deep content-based music recommendation,” *Advances in neural information processing systems*, 2013.

[25] P. Lops, M. Gemmis, and G. Semeraro, *Content-based Recommender Systems: State of the Art and Trends*. Recommender Systems Handbook, 2011.

[26] M. Papagelis and D. Plexousakis, “Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents,” *Engineering Applications of Artificial Intelligence*, vol. 18, no. 7, pp. 781–789, 2005.

[27] X. He, L. Liao, H. Zhang, L. Nie, and T. S. Chua, “Neural collaborative filtering,” *International World Wide Web Conferences Steering Committee*, 2017.

[28] Z. D. Zhao and M. S. Shang, “User-based collaborative-filtering recommendation algorithms on hadoop,” in *Third International Conference on Knowledge Discovery and Data Mining, WKDD 2010, Phuket, Thailand, 9-10 January 2010*, 2010.

[29] C. C. Aggarwal, “Model-based collaborative filtering,” *Recommender Systems*, 2016.

[30] Ian, D., Longstaff, , , John, F., and Cross, “A pattern recognition approach to understanding the multi-layer perception,” *Pattern Recognition Letters*, vol. 5, no. 5, pp. 315–319, 1987.

[31] F. Huang, “The application of multi-layer perception networks in the parameters optimization of stamping forming process,” in *International Conference on Computer Science Software Engineering*, 2008.

[32] V. Kumar, D. Khattar, S. Gupta, and V. Varma, “Word semantics based 3-d convolutional neural networks for news recommendation,” in *2017 IEEE International Conference on Data Mining Workshops, ICDM Workshops 2017, New Orleans, LA, USA, November 18-21, 2017*, 2017, pp. 761–764.

[33] K. Park, J. Lee, and J. Choi, “Deep neural networks for news recommendations,” in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 2255–2258. [Online]. Available: https://doi.org/10.1145/3132847.3133154

[34] J. Tang and K. Wang, “Personalized top-n sequential recommendation via convolutional sequence embedding,” in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, 2018, pp. 565–573.

[35] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He, “A simple convolutional generative network for next item recommendation,” in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 582–590.

[36] C. Wu, F. Wu, M. An, J. Huang, Y. Huang, and X. Xie, “Neural news recommendation with attentive multi-view learning,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019, pp. 3863–3869.

[37] C. Wu, F. Wu, M. An, Y. Huang, and X. Xie, “Neural news recommendation with topic-aware news representation,” in *Proceedings of the 57th Conference of the Association for Computational Linguistics*, 2019, pp. 1154–1159.

[38] Q. Zhu, X. Zhou, Z. Song, J. Tan, and L. Guo, “DAN: deep attention neural network for news recommendation,” in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, the Thirty-First Innovative Applications of Artificial Intelligence Conference, the Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, 2019, pp. 5973–5980.

[39] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, “Session-based recommendations with recurrent neural networks,” in *Proceedings of the 4th International Conference on Learning Representations*, 2016.

[40] S. Okura, Y. Tagami, S. Ono, and A. Tajima, “Embedding-based news recommendation for millions of users,” in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1933–1942.

[41] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1724–1734.

[43] R. Józefowicz, W. Zaremba, and I. Sutskever, “An empirical exploration of recurrent network architectures,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. JMLR Workshop and Conference Proceedings, vol. 37, 2015, pp. 2342–2350.

[44] M. An, F. Wu, C. Wu, K. Zhang, Z. Liu, and X. Xie, “Neural news recommendation with long- and short-term user representations,” in *Proceedings of the 57th Conference of the Association for Computational Linguistics*, 2019, pp. 336–345.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Proceedings of Annual Conference on Neural Information Processing Systems 2017*, 2017, pp. 5998–6008.

[46] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, “Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1441–1450.

[47] C. Wu, F. Wu, S. Ge, T. Qi, Y. Huang, and X. Xie, “Neural news recommendation with multi-head self-attention,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019, pp. 6388–6393.

[48] D. N. Milne and I. H. Witten, “Learning to link with wikipedia,” in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, 2008, pp. 509–518.

[49] A. Sil and A. Yates, “Re-ranking for joint named-entity recognition and linking,” in *Proceedings of 22nd ACM International Conference on Information and Knowledge Management*, 2013, pp. 2369–2374.

[50] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Proceedings of 27th Annual Conference on Neural Information Processing Systems*, 2013, pp. 2787–2795.

[51] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, “Knowledge graph embedding via dynamic mapping matrix,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, 2015, pp. 687–696.

[52] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embeddings for knowledge graph completion,” in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015, pp. 2181–2187.

[53] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014, pp. 1112–1119.

[54] D. Liu, T. Bai, J. Lian, X. Zhao, G. Sun, J. Wen, and X. Xie, "News graph: An enhanced knowledge graph for news recommendation," in *Proceedings of the Second Workshop on Knowledge-aware and Conversational Recommender Systems and of the 28th ACM International Conference on Information and Knowledge Management*, ser. CEUR Workshop Proceedings, vol. 2601, 2019, pp. 1–7.

[55] K. Joseph and H. Jiang, "Content based news recommendation via shortest entity distance over knowledge graphs," in *Proceedings of Companion of The 2019 World Wide Web Conference*, 2019, pp. 690–699.

[56] J. Ren, J. Long, and Z. Xu, "Financial news recommendation based on graph embeddings," *Decision Support Systems*, vol. 125, 2019.

[57] F. Wu, Y. Qiao, J. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, and M. Zhou, "MIND: A large-scale dataset for news recommendation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3597–3606.

[58] J. Konecný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *CoRR*, vol. abs/1610.02527, 2016.

[59] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *CoRR*, vol. abs/1610.05492, 2016.

[60] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016.

[61] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1310–1321.

[62] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[63] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2018.

[64] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck, "An in-depth study of LTE: effect of network protocol and application behavior on performance," in *Proceedings of ACM SIGCOMM 2013 Conference*, 2013, pp. 363–374.

[65] C. H. van Berkel, "Multi-core for mobile phones," in *Proceedings of Design, Automation and Test in Europe 2009*, 2009, pp. 1260–1265.

[66] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecný, S. Mazzocchi, B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," in *Proceedings of Machine Learning and Systems 2019*, 2019.

[67] V. Smith, C. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proceedings of Annual Conference on Neural Information Processing Systems 2017*, 2017, pp. 4424–4434.

[68] J. Li, M. Khodak, S. Caldas, and A. Talwalkar, "Differentially private meta-learning," in *Proceedings of 8th International Conference on Learning Representations*, 2020.

[69] N. Carlini, C. Liu, J. Kos, Ú. Erlingsson, and D. Song, "The secret sharer: Measuring unintended neural network memorization & extracting secrets," *CoRR*, vol. abs/1802.08232, 2018.

[70] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Privacy aware learning," in *Proceedings of 26th Annual Conference on Neural Information Processing Systems*, 2012, pp. 1439–1447.

[71] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[72] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *Proceedings of 6th International Conference on Learning Representations*, 2018.

[73] J. Xu, B. S. Glicksberg, C. Su, P. B. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021.

[74] Z. Chai, A. Ali, S. Zawad, S. Truex, A. Anwar, N. Baracaldo, Y. Zhou, H. Ludwig, F. Yan, and Y. Cheng, "Tifl: A tier-based federated learning system," in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020, pp. 125–136.

[75] Z. Zhou, S. Yang, L. Pu, and S. Yu, "CEFL: online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9341–9356, 2020.

[76] R. Hu, Y. Gong, and Y. Guo, "Cpfed: Communication-efficient and privacy-preserving federated learning," *CoRR*, vol. abs/2003.13761, 2020.

[77] Y. Liu, S. Sun, Z. Ai, S. Zhang, Z. Liu, and H. Yu, "Fedcoin: A peer-to-peer payment system for federated learning," *CoRR*, vol. abs/2002.11711, 2020.

[78] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *CoRR*, vol. abs/2004.00773, 2020.

[79] H. Kim, J. Park, M. Bennis, and S. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[80] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4298–4311, 2020.

[81] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *Proceedings of 2019 IEEE International Conference on Big Data*, 2019.

[82] G. Ulm, E. Gustavsson, and M. Jirstrand, "Functional federated learning in erlang (ffl-erl)," in *Proceedings of Functional and Constraint Logic Programming - 26th International Workshop*, ser. Lecture Notes in Computer Science, vol. 11285, 2018, pp. 162–178.

[83] S. Caldas, P. Wu, T. Li, J. Konecný, H. B. McMahan, V. Smith, and A. Talwalkar, "LEAF: A benchmark for federated settings," *CoRR*, vol. abs/1812.01097, 2018.

[84] Z. Zhao, H. Lu, D. Cai, X. He, and Y. Zhuang, "User preference learning for online social recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 9, pp. 2522–2534, 2016.

[85] B. Zhang and S. S. Sundar, "Proactive vs. reactive personalization: Can customization of privacy enhance user experience?" *International Journal of Human-Computer Studies*, vol. 128, pp. 86–99, 2019.

[86] N. H. Tran, W. Bao, A. Y. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *2019 Institute of Electrical and Electronics Engineers Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, 2019, pp. 1387–1395.

[87] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *International Journal of Medical Informatics*, vol. 112, pp. 59–67, 2018.

[88] F. Chen, Z. Dong, Z. Li, and X. He, "Federated meta-learning for recommendation," *CoRR*, vol. abs/1802.07876, 2018.

[89] M. Ammad-ud-din, E. Ivannikova, S. A. Khan, W. Oyomno, Q. Fu, K. E. Tan, and A. Flanagan, "Federated collaborative filtering for privacy-preserving personalized recommendation system," *CoRR*, vol. abs/1901.09888, 2019.

[90] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proceedings of the 8th IEEE International Conference on Data Mining*, 2008, pp. 263–272.

[91] A. Flanagan, W. Oyomno, A. Grigorievskiy, K. E. Tan, S. A. Khan, and M. Ammad-ud-din, "Federated multi-view matrix factorization for personalized recommendations," in *Proceedings of Machine Learning and Knowledge Discovery in Databases - European Conference*, ser. Lecture Notes in Computer Science, vol. 12458, 2020, pp. 324–347.

[92] S. Ying, "Shared MF: A privacy-preserving recommendation system," *CoRR*, vol. abs/2008.07759, 2020.

[93] V. W. Anelli, Y. Deldjoo, T. Di Noia, A. Ferrara, and F. Narducci, "How to put users in control of their data in federated top-n recommendation with learning to rank," in *Proceedings of the 36th ACM/SIGAPP Symposium On Applied Computing*, 2021.

[94] T. Li, L. Song, and C. Fragouli, "Federated recommendation system via differential privacy," in *Proceedings of IEEE International Symposium on Information Theory*, 2020, pp. 2592–2597.

[95] M. Ribero, J. Henderson, S. Williamson, and H. Vikalo, "Federating recommendations using differentially private prototypes," *CoRR*, vol. abs/2003.00602, 2020.

[96] V. W. Anelli, Y. Deldjoo, T. D. Noia, A. Ferrara, and F. Narducci, "Federank: User controlled feedback with federated recommender systems," *CoRR*, vol. abs/2012.11328, 2020.

[97] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "Fedgnn: Federated graph neural network for privacy-preserving recommendation," *CoRR*, vol. abs/2102.04925, 2021.

[98] E. Hesamifard, H. Takabi, and M. Ghasemi, "Cryptodl: Deep neural networks over encrypted data," *CoRR*, vol. abs/1711.05189, 2017.

[99] A. Wainakh, T. Grube, J. Daubert, and M. Mühlhäuser, "Efficient privacy-preserving recommendations based on social graphs," in *Proceedings of the 13th ACM Conference on Recommender Systems*. ACM, 2019, pp. 78–86.

[100] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4555–4562, 2019.

[101] Y. Yao, L. Rosasco, and A. Caponnetto, "On early stopping in gradient descent learning," *Constructive Approximation*, vol. 26, no. 2, pp. 289–315, 2007.

[102] L. W. Mackey, A. Talwalkar, and M. I. Jordan, "Divide-and-conquer matrix factorization," in *Proceedings of 25th Annual Conference on Neural Information Processing Systems*, 2011, pp. 1134–1142.

[103] Y. Zhang, J. C. Duchi, and M. J. Wainwright, "Divide and conquer kernel ridge regression: a distributed algorithm with minimax optimal rates," *Journal of Machine Learning Research*, vol. 16, pp. 3299–3340, 2015.

[104] N. Guha, A. Talwalkar, and V. Smith, "One-shot federated learning," *CoRR*, vol. abs/1902.11175, 2019.

[105] M. Yurochkin, M. Agarwal, S. Ghosh, K. H. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97, 2019, pp. 7252–7261.

[106] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, "Adaptive neural networks for efficient inference," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70, 2017, pp. 527–536.

[107] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems 2020*, 2020.

[108] S. Vaswani, F. R. Bach, and M. Schmidt, "Fast and faster convergence of SGD for over-parameterized models and an accelerated perceptron," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 89, 2019, pp. 1195–1204.

[109] D. Yin, A. Pananjady, M. Lam, D. S. Papailiopoulos, K. Ramchandran, and P. L. Bartlett, "Gradient diversity: a key ingredient for scalable distributed learning," in *Proceedings of International Conference on Artificial Intelligence and Statistics 2018*, ser. Proceedings of Machine Learning Research, vol. 84, 2018, pp. 1998–2007.

[110] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *CoRR*, vol. abs/1806.00582, 2018.

[111] H. Eichner, T. Koren, B. McMahan, N. Srebro, and K. Talwar, "Semi-cyclic stochastic gradient descent," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97, 2019, pp. 1764–1773.