DeepRec: On-device Deep Learning for Privacy-Preserving Sequential Recommendation in Mobile Commerce

Jialiang Han
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing
Peking University Information Technology Institute (Tianjin Binhai), Tianjin
China
hanjialiang@pku.edu.cn

Yun Ma*
Institute for Artificial Intelligence, Peking University, Beijing
China
mayun@pku.edu.cn

Qiaozhu Mei
School of Information, University of Michigan, Ann Arbor
USA
qmei@umich.edu

Xuanzhe Liu
Key Lab of High-Confidence Software Technology, MoE (Peking University), Beijing
China
xzl@pku.edu.cn

ABSTRACT

Sequential recommendation techniques are considered to be a promising way of providing better user experience in mobile commerce by learning sequential interests within user historical interaction behaviors. However, the recently increasing focus on privacy concerns, such as the General Data Protection Regulation (GDPR), can significantly affect the deployment of state-of-the-art sequential recommendation techniques, because user behavior data are no longer allowed to be arbitrarily used without the user's explicit permission. To address the issue, this paper proposes DeepRec, an on-device deep learning framework of mining interaction behaviors for sequential recommendation without sending any raw data or intermediate results out of the device, preserving user privacy maximally. DeepRec constructs a global model using data collected *before* GDPR and fine-tunes a personal model continuously on individual mobile devices using data collected *after* GDPR. DeepRec employs the model pruning and embedding sparsity techniques to reduce the computation and network overhead, making the model training process practical on computation-constraint mobile devices. Evaluation results show that DeepRec can achieve comparable recommendation accuracy to existing centralized recommendation approaches with small computation overhead and up to 10x reduction in network overhead.

CCS CONCEPTS

• **Information systems** → *E-commerce infrastructure*; *Mobile information processing systems*; • **Computing methodologies** → **Supervised learning**; • **Security and privacy** → **Software and application security**.

*Corresponding author

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3449942

KEYWORDS

Sequential Recommendation, On-device Training, Privacy-Preserving, Mobile Commerce, GDPR

ACM Reference Format:
Jialiang Han, Yun Ma, Qiaozhu Mei, and Xuanzhe Liu. 2021. DeepRec: On-device Deep Learning for Privacy-Preserving Sequential Recommendation in Mobile Commerce. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3449942

1 INTRODUCTION

With the rapid growth and widespread adoption of smartphones and tablet computers, mobile commerce [3] is becoming the dominant form of e-commerce during recent years. It is forecasted that mobile commerce will outpace non-mobile commerce in 2021, accounting for more than half of total e-commerce sales [2]. The special characteristic of user interactions on mobile devices, i.e., accessing the Internet at any time and any location, advances mobile commerce to provide even more personalized experience than traditional e-commerce by leveraging the temporal or spatial information of user interaction behaviors to precisely recommend items. Among these efforts, sequential recommendation [11, 25, 26, 33, 61] is one of the most promising approaches because the user interactions on mobile devices are usually fragmented and momentary [30] and sequential recommendation can effectively work on a small scale of fragmented data. Just like many tasks where deep neural networks significantly improve the performance [9, 23, 32], state-of-the-art sequential recommenders are mostly based on deep learning techniques [14, 57].

However, the enforcement of policies and legislation formulations to regulate the usage of privacy data, such as the General Data Protection Regulation (GDPR) [1] and the California Consumer Privacy Act (CCPA) [2] may make sequential recommendation no longer work. The reason is that current sequential recommenders in mobile commerce usually rely on the data that are not related to the

[1] https://gdpr-info.eu/
[2] https://oag.ca.gov/privacy/ccpa/

core business of mobile commerce, such as clicking on items [25] and dwell time on pages [4]. According to GDPR, *a specific, freely-given, plainly-worded, and unambiguous consent should be given by the data subject, i.e. the user, and the data subject has the right of access, the right to data portability, and the right to be forgotten.* As a result, users may not allow the recommender to use their interaction data for privacy concerns. Although there have been efforts on privacy-preserving deep learning techniques, such as the shared model [50], multi-party computation [21], transformation [18, 70], partial sharing [46], model splitting [13], and encryption [17, 63], the risks of privacy leakage still exist [10, 39, 54, 59, 60] because these approaches require to send intermediate results out of the users' devices, and the intermediate results could be used to infer users' identities and interests. As a result, users may still refuse the recommender to use their data, even encrypted or disturbed.

How can we reassure the users that the recommenders have no or least privacy leakage? A most straightforward answer is to make sure that no data, whether raw data or intermediate results, are sent out of devices during recommendation. Such an idea has been tried on the next-word-prediction task of input method applications. For example, DeepType [66] trains a global model on the cloud using massive public corpora crawled from the Internet (i.e., BBC News and Twitter), and then incrementally customizes the global model with privacy data to get a local model on individual devices. It maintains fixed representations of the full word vocabulary on the device. Compared with the next-word-prediction task, the scenario of sequential recommendation in mobile commerce has no common item vocabulary like the word vocabulary, and the volume of the item vocabulary is evolving, dynamic, and larger than the word vocabulary. As a result, the techniques proposed by DeepType cannot work in the scenario of sequential recommendation in mobile commerce.

To address the issue, in this paper, we propose DeepRec, an on-device deep learning framework for privacy-preserving sequential recommendation, with no raw data or intermediate results being uploaded. First, we train a Recurrent Neural Network (RNN) based global recommendation model with existing interaction behavior data of all users collected *before* GDPR on the cloud, and push the global model to individual devices. Second, we perform fine-tuning over an RNN-based personal recommendation model with interaction behavior data of each user collected *after* GDPR on her own device, and extract a unique user-specific representation from her personal model. Third, the individual device pulls a recommendation item candidate set from the cloud, and then we combine the latent information of the user representation and item representations to complete the recommendation process.

However, due to the computation limitation of mobile devices, deploying deep learning based applications on mobile devices faces special challenges [7, 65]. To make DeepRec practical, we particularly address two technical issues of mobile devices, i.e., network overhead and computational overhead. To reduce network overhead between the cloud and devices, we adopt the Least Absolute Shrinkage and Selection Operator (Lasso) regression, or L_1 regularization, to sparsify the item representations of the item candidate set. To reduce computational overhead on the devices, we adopt an Automated Gradual Pruner (AGP) [72] to compress both recurrent unit layers and fully connected layers of the recommendation model.

To evaluate the performance of DeepRec, we conduct extensive experiments over a widely-adopted publicly released user behavior dataset from Taobao (the most popular e-commercial application in China) [71], which contains about 1 million users and 72 million clicks. We comprehensively measure various key metrics including accuracy, network overhead, and computational overhead. The experimental results show that DeepRec achieves up to 10x reduction in network overhead, reducing the computational overhead of on-device training towards the range of computational resources of middle-class mobile devices, with minimal loss in accuracy. We also make the source code of DeepRec publicly available online [3]. The contributions of this paper are summarized as follows.

- We propose an on-device deep learning framework for privacy-preserving sequential recommendation, with no raw data or intermediate results being uploaded, which alleviates the risks of privacy leakage and violation of privacy-related laws, in benefit of recommendation service providers.
- We reduce both network and computational overhead of mobile devices by an Automated Gradual Pruner (AGP) and Lasso regression (L_1 regularization), to make sure that it is more feasible and less resource-consuming to perform the on-device training process, in benefit of application developers.
- We evaluate the performance of DeepRec on a large-scale public dataset, and the experimental results show that we achieve up to 10x reduction in network overhead and reduce computational overhead towards the range of middle-class mobile devices, with minimal loss in accuracy.

The remainder of this paper is organized as follows. In Section 2, we formally define the problem of the privacy-preserving sequential recommendation with no raw data or intermediate results being uploaded. In Section 3, we present the detailed design of DeepRec and optimizations to reduce network and computational overhead. In Section 4, we evaluate DeepRec on a widely-adopted publicly released dataset to demonstrate its effectiveness and efficiency. In Section 5, we discuss some limitations of DeepRec and show possible solutions. In Section 6, we compare DeepRec with related work. Finally, Section 7 concludes the paper.

2 PROBLEM STATEMENT

In this section, we present the problem statement of privacy-preserving sequential recommendation with no raw data or intermediate results being uploaded *after* GDPR. The problem statement of sequential recommendation is that, given a period of contexts $C = C_{t_1}, C_{t_2}, ..., C_{t_n}$, sequential recommendation is to learn a function f which maps C to the target $p_{i,j} : p_{i,j} \Leftarrow f(C)$. The context C_{t_k} could be the users' metadata, interaction behaviors, or their social relationship at the timestamp t_k. The target $p_{i,j}$ is the likelihood that a user i would prefer to click or purchase an item j at the timestamp t_n. Now let us define some important concepts in our setting.

[3] https://github.com/hanjialiang/DeepRec

Definition 1 (Business Necessary Data): Business necessary data are interaction behavior data necessary for the basic business of recommendation engines, like purchasing behaviors of e-commercial platforms or ordering behaviors of e-music platforms. Even though business necessary data are likely to be privacy sensitive, they are supposed to be gathered and uploaded to accomplish the basic business of the mobile commerce platforms that recommendation engines are supporting. For example, if a user prohibits a mobile commerce platform from gathering her purchase behaviors, she cannot even order or receive anything. We use $BN_{i,t}$ to represent the item ID that a user i purchases at the timestamp t.

Definition 2 (Privacy Sensitive Data): Privacy sensitive data are interaction behavior data unnecessary for the basic business of recommendation engines but considered to be privacy sensitive. For example, user-specific data that do not require login to collect are usually privacy sensitive, like click behaviors of mobile commerce platforms or dwell time on different images of image sharing platforms. We use $PS_{i,t}$ to represent the item ID that a user i clicks at the timestamp t.

Definition 3 (Item Candidate Set): From business necessary data of all users, an item candidate set $Can_{i,t}$ for each user i at the timestamp t, can be built or learned. The recommendation results for the user i at the timestamp t are retrieved and re-ranked from $Can_{i,t}$.

Definition 4 (Privacy-Preserving Sequential Recommendation with No Data Uploaded): We assume that, *after* GDPR, users do not allow to upload their privacy sensitive data to the cloud. Note that *no data uploaded* in this definition refer to privacy sensitive data *after* GDPR in our scenario. There are business necessary data and privacy sensitive data *before* GDPR of all users on the cloud server. The goal of privacy-preserving sequential recommendation with no data uploaded is to achieve a similar or better accuracy of the next click prediction on devices. We define the business necessary data set of all users before the timestamp t_k as $\mathcal{BN}_{t_k} = \{BN_{i,t}|\forall i; t \leq t_k\}$, and the privacy sensitive data set of all users before the timestamp $t_{GDPR} < t_k$ when GPPR takes effect as $\mathcal{PS}_{t_{GDPR}} = \{PS_{i,t}|\forall i; t \leq t_{GDPR}\}$. Then, privacy-preserving sequential recommendation to acquire the likelihood p_{i,j,t_k} that a user i would click an item j at the timestamp t_k can be formulated as

$$p_{i,j,t_k} = f(\mathcal{BN}_{t_k}, \mathcal{PS}_{t_{GDPR}}, \{PS_{i,t}|t \leq t_k\}, Can_{i,t_k}; j), \quad (1)$$

where f can be a personal model trained and inferred on the user i's device. Note that the computational resources of mobile devices are not as sufficient as those of the cloud, and the network bandwidth and the cellular data plan of users are limited. Therefore, the computational overhead and network overhead of the on-device privacy-preserving sequential recommendation should be relatively small to match limited resources of mobile devices. The objective of the privacy-preserving sequential recommendation at the timestamp t_k is to solve the constrained optimization problem as

$$\begin{aligned} &\theta^* = argmin_\theta \sum_i \sum_j \mathcal{L}(p_{i,j,t_k}, label_{i,j,t_k}; \theta), \\ &COM(f) \leq \mathcal{COM}, \\ &NET(f) \leq \mathcal{NET}, \end{aligned} \quad (2)$$

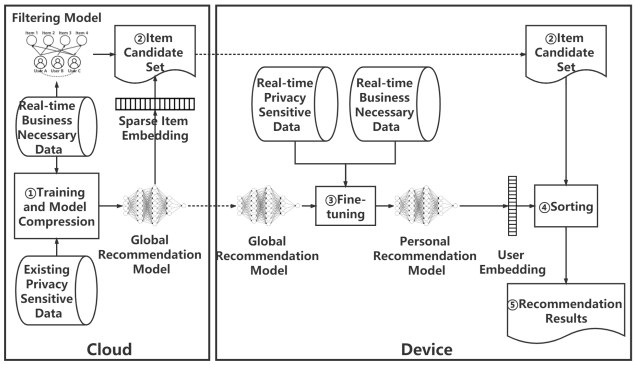


Figure 1: The architecture of DeepRec

where $\mathcal{L}$ is a classification loss function like the cross-entropy loss, or a learning-to-rank loss function [49, 52, 56], $label_{i,j,t_k} \in \{0, 1\}$ is the ground truth whether the user i actually clicks the item j at the timestamp t_k, and θ is the parameter set of the on-device personal model to optimize. $COM(f)$ is a function that maps a model f to its computational overhead, $\mathcal{COM}$ is the computational overhead threshold of mobile devices. $NET(f)$ is a function that maps a model f to its network overhead, $\mathcal{NET}$ is the network overhead threshold of mobile devices.

3 APPROACH

In this section, we first introduce the workflow of DeepRec. Then, we present the design of two fundamental models in this framework, i.e, the filtering model and the recommendation model. Finally, we explain several optimizations to reduce network and computational overhead.

3.1 Workflow Overview

Figure 1 shows the workflow of DeepRec. First, we train a Gated Recurrent Unit (GRU) [8] based recommendation model, i.e. the global model, with existing interaction behavior data of all users collected *before* GDPR, extract a personal recommendation item candidate set for each user through a collaborative filtering based model on the cloud, and push the global model to individual devices. Second, we perform fine-tuning over a GRU-based recommendation model, i.e. the personal model, with interaction behavior data of each user collected *after* GDPR on her own device, and extract a unique user-specific embedding from her personal model. Third, a user's device pulls her recommendation item candidate set from the cloud, and then we calculate the inner products of her user embedding and candidate item embeddings to acquire scores representing probabilities she would like to click an item in the current session. Note that, *fine-tuning* in the personal training phase means freezing the layers before the last GRU layer in the global model, and re-training only the last GRU layer and fully connected layer(s). The intuition behind fine-tuning is that the first several GRU layers can generalize from the global training set to the personal training set. So only the last GRU layer and fully connected layer(s) require to be re-trained, to fit the user-specific next click prediction.

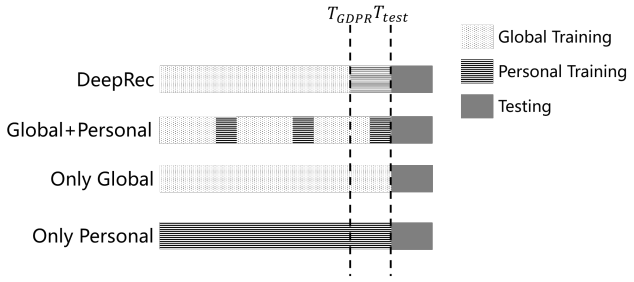


Figure 2: Learning pipelines of DeepRec and other architectures

Figure 2 shows the training and testing pipelines of DeepRec and other classic architectures. We partition the whole procedure into 3 stages: global training, personal training, and testing, by T_{GDPR} and T_{test}. T_{GDPR} is the timestamp after which privacy sensitive data are used for only training a personal model for each user, instead of being uploaded and training the global model, and T_{test} is the timestamp after which labeled privacy sensitive data are used for only testing, instead of training. In Figure 2, *DeepRec* trains a global model with all users' privacy sensitive data before T_{GDPR} and fine-tunes a personal model for each user with her privacy sensitive data before T_{test}, which satisfies the requirements of GDPR. The *global + personal* model trains a global model and fine-tunes a personal model for each user with all users' privacy sensitive data before T_{test}, which does not satisfy the requirements of GDPR because GDPR prohibits uploading privacy sensitive data without the user's consent. The *only global* model only trains a global model for all users with all users' privacy sensitive data before T_{test}, which is the training schedule for almost all state-of-the-art approaches of sequential recommendation, as far as we are concerned. Finally, the *only personal* model only trains a personal model for each user with her privacy sensitive data before T_{test}.

3.2 Filtering Model

To reduce the computational overhead of ranking the inner products of a given user embedding and item embeddings, and to reduce the network overhead of pulling item embeddings from the cloud, we use a k-nearest neighbor item-based collaborative filtering (item-CF) model [11, 51] to extract an item candidate set from all items. The main idea of item-CF is that a user is likely to purchase an item that shares similar features, i.e. user-item interactions, with items she purchases before. The filtering model predicts an estimated probability p_{m,n,t_k}, representing the estimated probability a user m *clicks* an item n at the timestamp t_k, from a business necessary data matrix $(x_{m,n,t_k})^{M*N}$ where $x_{m,n,t_k} \in \{0,1\}$ represents whether the user m *purchases* the item n before the timestamp t_k. Then, from the business necessary data matrix, we calculate the cosine similarities of the item n and other items, and acquire k-most similar items to the item n. Thus, the estimated probability $p_{m,n,t_k} = \frac{\sum_{i_b} sim(i_n,i_b)x_{m,b,t_k}}{\sum_{i_b}|sim(i_n,i_b)|}$, where i_n represents the n^{th} column vector of the matrix $(x_{m,n,t_k})^{M*N}$, i_b represents the column vector of each k-most similar item to the item n, and $sim(i_n,i_b) = \frac{i_n \cdot i_b}{\|i_n\|\|i_b\|}$.

For the privacy-preserving purpose, we use business necessary data, i.e. purchasing behaviors, to estimate the probability of click behaviors that cannot be uploaded due to GDPR. It is not a common practice to use only *purchase* behaviors to directly predict the probability of *click* behaviors, because this significantly harms the prediction accuracy, as shown in Section 4.3. Therefore, we only use the filtering model for guidelines and obtain an item candidate set from the universal set of items. We manually set a threshold probability $p_{candidate}$ and obtain a user m's recommendation item candidate set $Candidate_{m,t_k} = \{n|p_{m,n,t_k} > p_{candidate}\}$ at the timestamp t_k. For certain users without any feedback, i.e. new users to the recommendation engine, we choose the same number of most popular items as their item candidate set, instead of using k-nearest neighbor item-CF.

3.3 Recommendation Model

Inspired by GRU4REC [25], we use a Gated Recurrent Unit (GRU) based neural network to model the sequential information behind the historical interaction behaviors and then extract user representations. Like GRU4REC, we conduct session-parallel mini-batches and sampling on the output to customize GRU for sequential recommendation. The reasons for using GRU-based models are as follows. First, RNN-based and transformer-based [58] models can extract sequential information and dependencies from users' historical interaction behaviors. Second, as the personal model is supposed to be trained on mobile devices, the computational overhead should be as small as possible, which prevents us from introducing state-of-the-art transformer-based models, like BERT [12]. Therefore, RNN-based models could be a reasonable choice. Still, experimental results of related work [25, 42] show that GRU-based models perform better than both classic RNN unit based models and Long Short-Term Memory (LSTM) based models on the task of sequential recommendation.

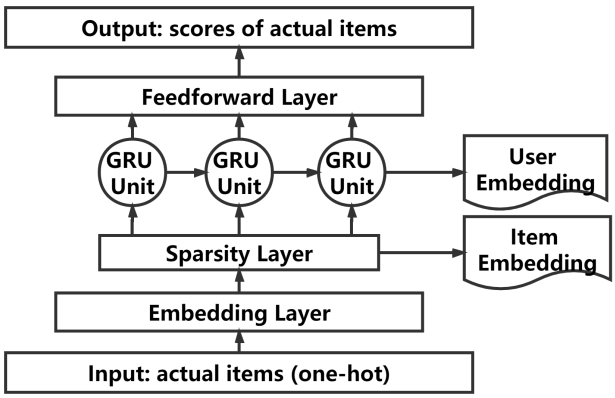


Figure 3: The GRU-based recommendation model

The detailed neural network structure of DeepRec is shown in Figure 3. The input of the recommendation model is a sequence of actual items in one-hot encoding, which are projected into low-dimensional dense embeddings through the embedding layer. To reduce network overhead, the dense embedding is then passed

through a sparsity layer to induce sparsity in the item embedding. After that, the sequence of item embeddings is passed into one or several GRU layer(s) to mine sequential information behind historical interaction behaviors. If multiple GRU layers are used, the input of the next layer is the hidden states of the last layer. The output of GRU layer(s) represents the click probability distribution of items. Besides, the output of GRU layer(s) represents the user's current items of interest, namely the user embedding. Finally, the user embedding is passed into one or several feed-forward layer(s) to output the scores of actual items that the user is likely to click next.

3.4 Sparsity of Embeddings

To reduce network overhead between the cloud and devices, we perform the Lasso regression, or L_1 regularization, to sparsify the item embeddings of the item candidate set. Lasso regression extracts principal information from raw embeddings through sparse encoding, which can be formulated as

$$\begin{aligned} \hat{E}_i &= \begin{cases} E_i, if\ |E_i| > \gamma \\ 0, otherwise \end{cases}, \\ \mathcal{L}_{lasso} &= \sum_i \left|\hat{E}_i\right|, \\ \hat{\mathcal{L}} &= \mathcal{L} + \lambda_{lasso}\mathcal{L}_{lasso}, \end{aligned} \tag{3}$$

where E_i is the i^{th} element of the raw embedding, $\hat{E}_i$ is the i^{th} element of the sparse embedding, $\mathcal{L}$ is the original loss function, λ_{lasso} is a weight to measure the importance of the Lasso penalty, and $\hat{\mathcal{L}}$ is the adjusted loss function with a Lasso penalty. In the sparse layer of DeepRec, the input embedding is truncated with a threshold γ, and only the elements above γ are kept in the output embedding. Then, the lasso penalty term $\lambda_{lasso}\mathcal{L}_{lasso}$ is added to the adjusted loss function, induce sparsity in the output embedding.

3.5 Sparsity of Neural Networks

To reduce the computational overhead of training personal models on devices, we perform an Automated Gradual Pruner (AGP) [72] to compress the recommendation model. Pruning is a commonly practiced methodology to induce *sparsity* (i.e. a measure of how many elements in a tensor are exact zeros, relative to the tensor size) in weights of the neural network. Pruning assigns weights satisfying a certain criteria to zero and those trimmed weights will be shut down permanently and not participate in back-propagation. A level pruner is a stable pruner where a neural network is pruned into a specific sparsity level, through a threshold magnitude criteria. AGP is a scheduled level pruner, formulated as

$$s_t = s_f + \left(s_i - s_f\right)\left(1 - \frac{t - t_0}{n\Delta t}\right)^3 \quad for\ t \in \{t_0, t_0 + \Delta t, ..., t_0 + n\Delta t\}, \tag{4}$$

where s_t is the sparsity level in the t time step (i.e. epoch), s_i is the initial sparsity level and s_f is the final target sparsity level. The main idea of AGP is to initially prune the weights rapidly when redundant connections are abundant and gradually reduce the number of pruned weights when redundant connections are becoming fewer.

The reasons why we choose the AGP are as follows. First, AGP does not make strong assumptions on the structure of the neural network, hence we can prune redundant weights not only in fully connected layers but also in recurrent unit layers. Second, it is straightforward and not difficult for AGP to tune hyperparameters. Besides, AGP does not require expert knowledge to design a student network, as knowledge distillation [27] requires. Third, AGP reduces computational overhead in both the training phase and the inference phase and allows for fine-tuning or re-training.

4 EVALUATION

In this section, we conduct extensive experiments over a widely-adopted publicly released user behavior dataset, which contains about 1 million users and 72 million clicks. We comprehensively measure various key metrics including accuracy, network overhead, and computational overhead.

4.1 Dataset

To evaluate the performance of DeepRec and baselines, we use a public user behavior dataset from Taobao [71], namely UserBehavior[4], which contains nearly a million users and nearly 100 million behaviors including click, purchasing, favoring, and adding items to the shopping cart from November 25 to December 03, 2017. In specific, a user-item interaction contains domains including user ID, item ID, item's category ID, behavior type, and timestamp. We manually partition the interaction behavior data into sessions by using a 706-second idle threshold. Like GRU4Rec [25], we filter out sessions of length 1, filter out sessions with less than 12 clicks, and filter out clicks from the testing set where the item clicked is not in the training set.

To simulate the scenario of GDPR, we manually set a GDPR deadline at the Unix timestamp T_{GDPR} = 1512057600 (i.e. 00:00 on December 01, 2017), after which click data can only be used to fine-tune a personal model on devices, instead of being uploaded to train the global model on the cloud. To evaluate the performance of DeepRec, we manually set a training deadline at the Unix timestamp T_{test} = 1512230400 (i.e. 00:00 on December 03, 2017), after which click data can only be used to test, instead of training whether the global model or personal models.

To simulate the cold start scenario that some new users register *after* GDPR, we manually partition users into old users and new users by the average of timestamps of their clicks. Users with their click timestamp averages above 90% click timestamps of all users are assigned as new users, while others are assigned as old users. New users appear themselves after T_{GDPR}, and a new user's clicks can only be used to train her personal model instead of the global model. Note that old users' clicks are used to train both the global model and their personal models. After the above filtering and partitions, the scale and statistics of UserBehavior are shown in Table 1.

4.2 Metrics and Implementation

Like GRU4Rec [25], we choose Recall@20 and MRR@20 to evaluate recommendation accuracy. Recall@20 is the proportion of instances with the ground truth next-click items among the top-20 predicted items in all instances, which does not consider the ranks of the ground truth next-click items. MRR@20 is the average reciprocal

[4]https://tianchi.aliyun.com/dataset/dataDetail?dataId=649

Table 1: Description of the dataset

Scale	Users	Items	Clicks
Global training (old users)	876,914	922,390	51,908,146
Personal training and testing (old users)	788,472	886,658	17,200,197
Personal training and testing (new users)	97,199	567,213	2,810,257

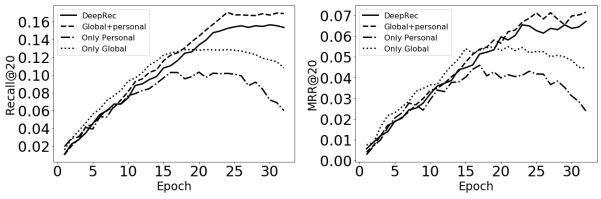
ranks of ground truth next-click items among the top-20 predicted items in all instances, which considers the ranks of the ground truth next-click items. To measure the sparsity of an embedding or a neural network, we use the proportion of exact zero elements among all elements in a tensor or the weight matrix.

We implement the DeepRec prototype based on Pytorch [44] to train a global model on the cloud and TensorFlow.js [53] to train a personal model on the device, and deploy it with the WebView of an experimental application. In the global training phase, we carefully choose hyper-parameters by running experiments at several combinations of random points of the hyper-parameter space. We choose the optimal hyper-parameter combination by selecting the highest Recall@20 and MRR@20 on the validation set, which is randomly divided from the training set. The number of GRU layer(s) is one, with 100 hidden units. The batch size is 50 and the loss function is cross-entropy. The embedding size of the embedding layer is 10000. The optimizer is Adagrad [15]. The learning rate is 0.01. Both momentum and weight decay are 0.

In the personal training phase, we take advantage of WebView [35]. WebView is a control for displaying web content inside Android applications based on the Chrome engine, which provides a built-in JavaScript parser. We leverage WebView to implement the adaptation of TensorFlow.js interpreter inside Android applications, including bidirectional calling of both the Web environment and the Android application, and dynamically loading Android files in the Web environment. Besides, instead of packaging the model file (.pth) together with the application installation package (.apk), we dynamically loads the individual compressed model file as a configuration, to keep the installation package light-weight and flexible. In specific, we store the model file and other configuration files in the external storage directory of Android. When the application starts, our framework reads the configuration file from the storage and registers the corresponding model metadata. When the on-device training phase starts, our framework decompresses and loads the global model according to the file path in model metadata. We use Google Pixel 2 smartphones (Android 8.0, Octa-core (4x2.45 GHz + 4x1.9 GHz) Kryo, Adreno 540) to conduct experiments to measure the cost of on-device training.

4.3 Recommendation Accuracy

We evaluate the effectiveness of the global model and personal models. The learning pipelines of DeepRec and other architectures are demonstrated in Section 3.1. Figure 4 shows the learning curves of Recall@20 and MRR@20 of DeepRec and baselines on the testing set. We notice that *only personal* models and the *only global* model converge earlier than mixed models. However, *only personal* models suffer a poor accuracy, because a single user's training data are so few that her personal model is easy to overfit and converges to a local optimal point. DeepRec performs better than the *only global* model, i.e. GRU4Rec [25], because DeepRec both takes advantage of the common click patterns of all users through the global model and mines a user's unique click pattern through her personal model. Although *global + personal* models actually predict the next-click item slightly better than DeepRec, they are not suitable for the privacy-preserving scenario, because users have the right to keep their privacy sensitive data on devices due to GDPR. The recommendation accuracy of DeepRec and baselines is shown in Table 2 for comparison.



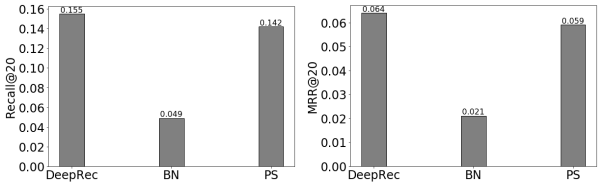
(a) Recall@20 (b) MRR@20

Figure 4: Learning curves of DeepRec and baselines

Table 2: Accuracy of DeepRec and baselines

Approach	Recall@20	MRR@20
Global + personal	0.168	0.069
DeepRec	0.155	0.064
Only global (GRU4Rec)	0.128	0.052
Only personal	0.101	0.043

Summary: DeepRec *can effectively predict the next-click item and achieve a higher accuracy than the only global model and only personal models.*



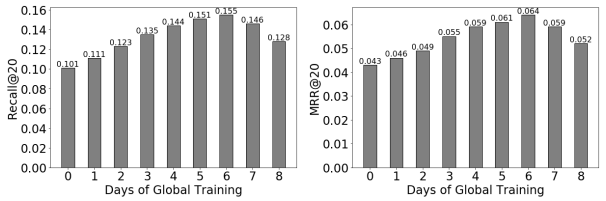
(a) Recall@20 (b) MRR@20

Figure 5: The necessity of privacy sensitive data

To answer the question that whether only business necessary data, i.e. purchasing behaviors, are enough to predict the next-click item, we train our model with only business necessary data, only privacy sensitive data, and both of them. As shown in Figure 5, the accuracy of the model trained with only privacy sensitive data is close to that of DeepRec, however, the accuracy of the model trained with only business necessary data is much lower than that of DeepRec. The explanation is rather straightforward, the task

of sequential recommendation is to predict the next-click item based on the history of user behaviors, therefore, it makes more sense to use privacy sensitive data including click behaviors to predict the probability of the next-click item, instead of business necessary data. Therefore, we cannot use only business necessary data, i.e. purchasing behaviors, to achieve an accurate sequential recommendation.

***Summary**: Both business necessary data and privacy sensitive data are necessary for the next-click prediction task. Only using business necessary data can hardly predict the next-click item effectively.*

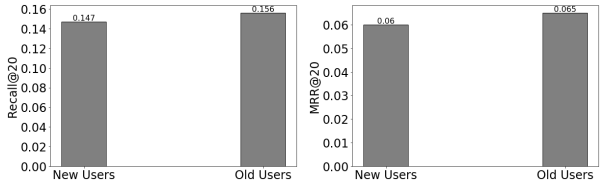


(a) Recall@20 (b) MRR@20

Figure 6: The influence of the size of the global training set

To measure how GDPR deadline T_{GDPR}, i.e. the size of the global training set, affects the recommendation accuracy, we scale both the period of the global training set and the personal training set by ranging T_{GDPR} from 0:00 on the first day to 23:59 on the last day of the training set. The detailed recommendation accuracy is shown in Figure 6, where 0-day global training actually represents *only personal* models, and 8-day global training actually represents the *only global* model. The results show that from 0-day to 6-day global training, the recommendation accuracy increases, while from 6-day to 8-day global training, the recommendation accuracy drops. The reason why the accuracy increases from 0-day to 6-day global training is that more behavior data used for global training can better learn the common patterns of all users and avoid overfitting. The reason why the accuracy drops from 6-day to 8-day global training is that personal training is not long enough to fully mine a user's unique click pattern.

***Summary**: As the size of the global training set increases, the recommendation accuracy gradually increases and then gradually drops.*

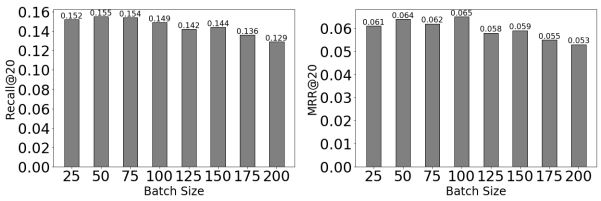


(a) Recall@20 (b) MRR@20

Figure 7: The generalization of DeepRec on new users

To measure the generalization of DeepRec on new users, we measure the recommendation accuracy of old users and new users, as shown in Figure 7. Intuitively, the recommendation accuracy of new users is supposed to be significantly lower than that of old users, because the global model cannot learn directly from the historical interaction behaviors of new users. Surprisingly, the experimental results indicate that although new users' interaction behavior data are not uploaded or used for global training, the recommendation accuracy of new users is close to that of old users. Therefore, DeepRec can effectively learn the click pattern of old users during global training and generalize this pattern to that of new users during personal training.

***Summary**: The recommendation accuracy of new users is close to that of old users, which means that* DeepRec *generalizes well on new users, although their interaction behaviors are not used for global training.*



(a) Recall@20 (b) MRR@20

Figure 8: The influence of update intervals of fine-tuning personal models

Personal training is not supposed to be performed once new user-item interactions appear on devices, because mobile devices are not always idle and available for training which consumes computational resources and the battery. On the other hand, intuitively, if the personal model is not updated for a long time, the recommendation accuracy is likely to drop. To measure how update intervals influence the recommendation accuracy, we scale the size of mini-batch during personal training from 25 to 200. Note that the personal model is not updated and is used for only inference until a mini-batch ends. As shown in Figure 8, within a relatively wide range, the update intervals of fine-tuning personal models hardly influence the recommendation accuracy. Therefore, even if a user keeps her personal model unchanged for a relatively long period, the recommendation accuracy would not be seriously affected.

***Summary**: The personal model can be kept not updated for a relatively long period without much accuracy loss if the state of the user device is not allowed for personal training.*

4.4 Network Overhead

There are mainly two parts of network overhead between the cloud and the device. On the one hand, the device needs to download the item candidate set from the cloud. On the other hand, the device needs to download the global recommendation model from the cloud. We utilize Lasso regression (L_1 regularization) to sparsify item embeddings of the item candidate set, and AGP to sparsify the global recommendation model.

To measure how the sparsity of item embeddings affects the recommendation accuracy, we scale the target sparsity of embeddings and compare Recall@20 and MRR@20. As shown in Figure 9, as

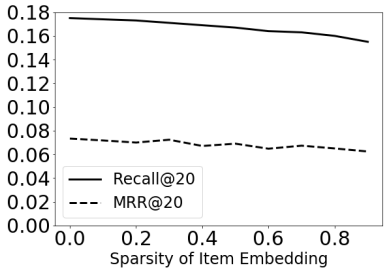


Figure 9: The influence of Lasso regularization

the sparsity of item embeddings increases, the recommendation accuracy drops slowly, which encourages us to choose an aggressive embedding sparsity of 90%, to reduce the network overhead between the cloud and the device.

Summary: DeepRec *can achieve a high sparsity of item embeddings with an acceptable recommendation accuracy loss.*

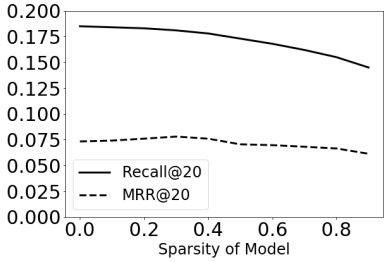


Figure 10: The influence of AGP model compression

To measure how the sparsity of the global recommendation model affects the recommendation accuracy, we scale the target sparsity of the global model and compare Recall@20 and MRR@20. As shown in Figure 10, as the sparsity of the model increases, the recommendation accuracy drops slowly, which encourages us to choose an aggressive model sparsity of 80%, to reduce the network overhead between the cloud and the device.

Summary: DeepRec *can achieve a high model sparsity with an acceptable recommendation accuracy loss.*

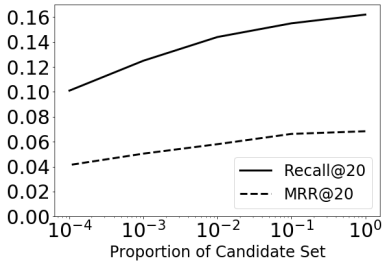


Figure 11: The influence of the size of the item candidate set

Note that we also use an item-CF based filtering model to produce an item candidate set. To measure how the size of the item candidate set affects the recommendation accuracy, we scale the proportion of the candidate set over all items and compare Recall@20 and MRR@20. As shown in Figure 11, it turns out that the recommendation accuracy would drop rapidly as the proportion of the candidate set becomes smaller. Hence, we carefully choose a conservative proportion of 10%, to balance network overhead and the recommendation accuracy.

Summary: *A strict filtering model with a small proportion of the item candidate set would reduce network overhead in sacrifice of the recommendation accuracy.*

Table 3: Cost of On-device Training

Metric	DeepRec	DeepRec w/o AGP	DeepRec w/o item embedding
Training Time	1163 seconds	3725 seconds	–
CPU%	66.3%	65.8%	–
Memory	842.9 MB	844.1 MB	∞
Downloaded Item Candidate Set	391.1 MB	391.1 MB	0.4 MB
Downloaded Model	1.71 MB	16.2 MB	156.8 MB

4.5 Computational Overhead

We use several techniques to make on-device training more feasible and less resource-consuming, including AGP, item embedding, and embedding sparsity. To evaluate how each technique affects the feasibility and computational overhead of on-device training, we conduct several ablation experiments on DeepRec. Note that *DeepRec w/o item embedding* means removing the embedding layer and representing items in one-hot encoding. As shown in Table 3, DeepRec without item embedding cannot be deployed on mobile devices because the runtime memory it needs to train a personal model is dramatically beyond that of middle-class mobile devices. Besides, AGP helps to both save training time and reduce the size of the downloaded global model.

Summary: *With the help of several compression methods,* DeepRec *achieves up to 10x reduction in network overhead and reduce computational overhead towards the range of middle-class mobile devices.*

5 DISCUSSION

In this section, we discuss some threats to our approach and evaluation, including the dataset limitation, whether to upload business necessary data, the sustainability of personal models, and the legality of using existing privacy sensitive data, and show some solutions to tackling those limitations.

5.1 Threats to the Dataset

The experimental results are limited by the user behavior dataset, whose period is only 9 days. Because this dataset contains around 1 million users and 72 million clicks, it takes one Tesla M40 GPU several weeks to train the global model and select for the best combinations of hyperparameters. Due to the time limit, we only evaluate the effectiveness and efficiency of DeepRec on one dataset.

The experimental results might change among different datasets. However, we believe that the framework is still effective, and hyperparameters are supposed to be carefully chosen to adapt to other datasets in the scenario of sequential recommendation in mobile commerce. In future work, we will evaluate DeepRec on more datasets of sequential recommendation in mobile commerce.

5.2 Uploading of Business Necessary Data

Some privacy concerns lie in the uploading and usage of business necessary data, like purchasing behaviors of e-commercial platforms or ordering behaviors of e-music platforms. However, they are still supposed to be gathered and uploaded to accomplish the basic business of the mobile commerce platforms that recommendation engines are supporting. Actually, as business necessary data are used to produce an item candidate set, there are some alternatives to avoid uploading business necessary data. For example, a heuristic method is to produce one item candidate set for all users by selecting items of top-k popularity, or produce a personal item candidate set for each user by selecting items of her top-k click frequencies in existing privacy sensitive data collected *before* GDPR. However, those heuristic methods could harm the recommendation accuracy. In future work, we will make efforts to handle the trade-off between accuracy and privacy concerns of uploading business necessary data.

5.3 Sustainability of Personal Models

As shown in Figure 6, before the 6^{th} day, the recommendation accuracy increases gradually as the size of the global training set increases. This indicates that DeepRec still requires an appropriate amount of interaction behavior data to train the global model and avoid overfitting. In an extreme situation that personal models are trained for a much longer period than the global model, DeepRec might degenerate to *only personal* models, which suffer an unstable recommendation accuracy. This problem could be relieved by encouraging some users to volunteer to upload and use their interaction behavior data, online training the global model, and updating all users' personal models incrementally. To encourage those volunteers, they are supposed to gain some benefits, such as achieving a higher recommendation accuracy than those who refuse to upload or use their interaction behavior data. In future work, we will consider privacy-preserving online learning or federated learning [5, 31, 67] to adapt that scenario.

5.4 Legality of Using Existing Privacy Sensitive Data

Unlike online public corpora in DeepType [66], due to the heterogeneous of item metadata, there are almost no public cross-platform datasets for sequential recommendation to conduct global training, as far as we are concerned. Therefore, we have to rely on existing interaction behavior data collected *before* GDPR. This raises the question of whether it complies with GDPR to process existing data collected *before* GDPR. After carefully going through GDPR, we surprisingly find out that there is an exemption called Legitimate Interests[5] in GDPR. Legitimate Interests *allows processing existing user data for legitimate interests without users' consent under careful consideration.* The application scene of this mechanism includes fraud detection, crime prediction, network security protection, and marketing guidance. Therefore, it complies with GDPR to process existing interaction behavior data collected *before* GDPR under Legitimate Interests.

[5]https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/lawful-basis-for-processing/legitimate-interests/

6 RELATED WORK

In this section, we introduce related work about sequential recommendation and privacy-preserving deep learning and compare those methods with DeepRec.

6.1 Sequential Recommendation

Due to the information explosion and the exposure demand of long-tail items, recommender systems have become more and more important in e-commercial platforms [1]. On mobile devices, user interactions are usually fragmented and momentary [30], and contain rich temporal and spatial information [22, 69]. However, Collaborative Filtering (CF) based recommendation [20, 51] and Matrix Factorization (MF) based recommendation [24, 34] can hardly extract sequential information behind user behavior history, because those frameworks treat each user-item interaction independently and equally, and do not consider the timestamps of user-item interactions. To handle this problem, sequential recommendation [62, 64] and session-based recommendation [55, 68] techniques, represented by GRU4REC [25], are proposed to leverage recurrent neural networks (RNN) to extract sequential information behind users' preferences over items. GRU4REC inputs the click sequence in a period, trains an adapted GRU-based model with a point-wise or pair-wise loss [49], and finally outputs the possibility of each item to be clicked next. Many researchers focus on improving the performance of GRU4REC. For example, p-RNN [26] takes unstructured data such as images and texts as features. Bogina et al. [4] consider users' dwell time on items. Hierarchical RNN [48] considers users' identities and constructs a hierarchical RNN with user-level GRU and session-level GRU. Jannach et al. [29] ensemble the results of an RNN-based model and a k-nearest neighbor (KNN) based model. As far as we are concerned, most of the preceding sequential recommendation approaches require to collect and upload user behavior data from devices to the cloud where a global model is trained and fine-tuned. Due to the terms of possession, usage, and portability of user-generated data in GDPR, if a user refuses to make her behavior data uploaded for training a global model, most of the preceding approaches would not be practical anymore, and the user experience could be compromised.

6.2 Privacy-preserving Deep Learning

In privacy-preserving recommendation, most of the preceding approaches focus on applying transformation [16, 43], encryption [17, 63], multi-party computation [21], and other techniques on CF-based recommendation and MF-based recommendation, as far as we are concerned. SerRec [47] introduces two-stage Locality-Sensitive Hashing (LSH) [19] into CF-based recommendation. Nikolaenko et al. [40] perform MF-based recommendation through a cryptographic technique, i.e. garbled circuits. PrivSR [37] allocates

different noise magnitudes to ratings in MF-based social recommendation. However, when it comes to large-scale sequential recommendation, RNN-based models achieve a better performance than CF-based or MF-based models, because RNN can effectively extract sequential information behind users' historical behaviors [14, 57]. Traditional privacy-preserving recommendation approaches face a trade-off between privacy and recommendation accuracy, and cannot be easily transplanted to deep learning based sequential recommendation.

Researchers also focus on privacy-preserving deep learning techniques, which can be transplanted to sequential recommendation with some adaptions. Boulemtafes et al. [6] propose a novel multi-level taxonomy of state-of-the-art privacy-preserving deep learning techniques, and conclude that shared model [50], multi-party computation [21], transformation [18, 70], partial sharing [46], model splitting [13], and encryption [17, 63] are currently used for privacy-preserving deep learning, and privacy-preserving deep learning is classified into the server-based setting and the server-assisted collaborative setting. In both settings, and even in the recently emerging federated setting [28, 41], communication between the cloud and devices is still required. Researchers [10, 36, 39, 54, 59, 60] have pointed out that intermediate results transmit between the cloud and devices, even encrypted or disturbed, can still help to infer users' identities and interests, and lead to privacy leakage. However, DeepRec allows on-device training of personal deep learning models, and thus accomplish sequential recommendation with no raw data or intermediate results being uploaded.

The privacy leakage in privacy-preserving collaborative learning or federated learning raises the demand for training and inferring a neural network on devices to ensure no raw data or intermediate results to be uploaded. For example, DeepType [66] trains a global model on the cloud using massive public corpora crawled from the Internet, incrementally customizes the global model with privacy data on devices, makes several optimizations to reduce computational overhead in the *softmax* layer of neural networks, and finally provides a next-word prediction. However, compared with that task, the scenario of sequential recommendation in mobile commerce raises different requirements and challenges. For example, the universal set of items (i.e. vocabulary in DeepType) cannot be downloaded directly to devices due to memory limit. Besides, item embeddings of mobile commerce corpora are usually longer than pre-trained word2vec [38] or GloVe [45] word embeddings. Most importantly, personal item candidate sets are dynamically evolving and updating with fashion trends and user interests. Those requirements and challenges of sequential recommendation in mobile commerce urge DeepRec to pay more attention to network overhead of updating item candidate set, and on-device computational overhead in not only the softmax layer but also recurrent unit layers of RNNs.

7 CONCLUSION

This paper proposes DeepRec, an on-device deep learning framework for sequential recommendation to mitigate privacy issues in mobile commerce. The key principle of DeepRec is that all recommendation procedures related to users' interaction behavior data take place on their mobile devices so that no raw data or intermediate results are sent out of mobile devices. We adopt an Automated Gradual Pruner and Lasso regression to induce sparsity in neural networks and item embeddings to significantly reduce computational overhead and network overhead. We evaluate the performance of DeepRec on a widely-adopted publicly released user behavior dataset, and the experimental results show that we achieve up to 10x reduction in network overhead and reduce computational overhead towards the range of middle-class mobile devices, with minimal loss in accuracy.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China under the grant number 2018YFB1004403, the National Natural Science Foundation of China under the grant number 61725201, the Beijing Outstanding Young Scientist Program under the grant number BJJWZYJH01201910001004, and the Key Laboratory of Intelligent Application Technology for Civil Aviation Passenger Services, CAAC.

REFERENCES

[1] 2015. *Recommender Systems Handbook.* Springer.
[2] 2017. Mobile Commerce Roundup. https://www.emarketer.com/public_media/docs/eMarketer_Mobile_Commerce_Roundup_2017.pdf.
[3] 2020. Mobile Commerce. https://en.wikipedia.org/wiki/Mobile_commerce.
[4] Veronika Bogina and Tsvi Kuflik. 2017. Incorporating Dwell Time in Session-Based Recommendations with Recurrent Neural Networks. In *Proceedings of the 1st Workshop on Temporal Reasoning in Recommender Systems co-located with 11th International Conference on Recommender Systems, RecSys '17*, Vol. 1922. 57–59.
[5] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konecný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards Federated Learning at Scale: System Design. In *Proceedings of Machine Learning and Systems 2019, MLSys '19.*
[6] Amine Boulemtafes, Abdelouahid Derhab, and Yacine Challal. 2020. A Review of Privacy-preserving Techniques for Deep Learning. *Neurocomputing* 384 (2020), 21–45.
[7] Zhenpeng Chen, Yanbin Cao, Yuanqiang Liu, Haoyu Wang, Tao Xie, and Xuanzhe Liu. 2020. A Comprehensive Study on Challenges in Deploying Deep Learning Based Software. In *Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE '20.* 750–762.
[8] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of the Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, SSST@EMNLP '14.* 103–111.
[9] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12 (2011), 2493–2537.
[10] Mauro Conti, Luigi Vincenzo Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. 2016. Analyzing Android Encrypted Network Traffic to Identify User Actions. *IEEE Trans. Inf. Forensics Secur.* 11, 1 (2016), 114–125.
[11] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, and Dasarathi Sampath. 2010. The YouTube Video Recommendation System. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys '10.* 293–296.
[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018).
[13] Hao Dong, Chao Wu, Zhen Wei, and Yike Guo. 2018. Dropping Activation Outputs With Localized First-Layer Deep Network for Enhancing User Privacy and Data Security. *IEEE Trans. Inf. Forensics Secur.* 13, 3 (2018), 662–670.
[14] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential User-based Recurrent Neural Network Recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17.* 152–160.
[15] John C. Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.* 12 (2011), 2121–2159.
[16] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 211–407.

[17] Caroline Fontaine and Fabien Galand. 2007. A Survey of Homomorphic Encryption for Nonspecialists. *EURASIP J. Inf. Secur.* 2007 (2007).

[18] Yingwei Fu, Huaimin Wang, Kele Xu, Haibo Mi, and Yijie Wang. 2019. Mixup Based Privacy Preserving Mixed Collaboration Learning. In *Proceedings of the 13th IEEE International Conference on Service-Oriented System Engineering, SOSE '19.*

[19] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. 1999. Similarity Search in High Dimensions via Hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases, VLDB '99.* 518–529.

[20] David Goldberg, David A. Nichols, Brian M. Oki, and Douglas B. Terry. 1992. Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[21] Oded Goldreich. 1998. Secure Multi-party Computation. *Manuscript. Preliminary version* 78 (1998).

[22] Eduardo Graells-Garrido, Diego Caro, Omar Miranda, Rossano Schifanella, and Oscar F. Peredo. 2018. The WWW (and an H) of Mobile Application Usage in the City: The What, Where, When, and How. In *Companion of the Web Conference 2018 on The Web Conference 2018, WWW '18.* 1221–1229.

[23] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5-6 (2005), 602–610.

[24] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI '17.* 1725–1731.

[25] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *Proceedings of the 4th International Conference on Learning Representations, ICLR '16, Conference Track Proceedings.*

[26] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16.* 241–248.

[27] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *CoRR* abs/1503.02531 (2015).

[28] Mingkai Huang, Hao Li, Bing Bai, Chang Wang, Kun Bai, and Fei Wang. 2020. A Federated Multi-View Deep Learning Framework for Privacy-Preserving Recommendations. *CoRR* abs/2008.10808 (2020).

[29] Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks Meet the Neighborhood for Session-Based Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17.* 306–310.

[30] Amy K. Karlson, Shamsi T. Iqbal, Brian Meyers, Gonzalo A. Ramos, Kathy Lee, and John C. Tang. 2010. Mobile Taskflow in Context: a Screenshot Study of Smartphone Usage. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI '10.* 2009–2018.

[31] Jakub Konecný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated Learning: Strategies for Improving Communication Efficiency. *CoRR* abs/1610.05492 (2016).

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems, NeurIPS '12.* 1106–1114.

[33] Qiwen Liu, Tianjian Chen, Jing Cai, and Dianhai Yu. 2012. Enlister: Baidu's Recommender System for the Biggest Chinese Q&A Website. In *Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12.* 285–288.

[34] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender Systems with Social Regularization. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM '11.* 287–296.

[35] Yun Ma, Dongwei Xiang, Shuyu Zheng, Deyu Tian, and Xuanzhe Liu. 2019. Moving Deep Learning into Web Browser: How Far Can We Go?. In *Proceedings of the World Wide Web Conference, WWW '19.* 1234–1244.

[36] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting Unintended Feature Leakage in Collaborative Learning. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy, SP '19.* 691–706.

[37] Xuying Meng, Suhang Wang, Kai Shu, Jundong Li, Bo Chen, Huan Liu, and Yujun Zhang. 2018. Personalized Privacy-Preserving Social Recommendation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI '18.* 3796–3803.

[38] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the 1st International Conference on Learning Representations, ICLR '13, Workshop Track Proceedings.*

[39] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy, SP '19.* 739–753.

[40] Valeria Nikolaenko, Stratis Ioannidis, Udi Weinsberg, Marc Joye, Nina Taft, and Dan Boneh. 2013. Privacy-preserving Matrix Factorization. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS '13.* 801–812.

[41] Chaoyue Niu, Fan Wu, Shaojie Tang, Lifeng Hua, Rongfei Jia, Chengfei Lv, Zhihua Wu, and Guihai Chen. 2020. Billion-scale federated learning on mobile clients: a submodel design with tunable privacy. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking, MobiCom '20.* 31:1–31:14.

[42] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. 2017. Embedding-based News Recommendation for Millions of Users. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17.* 1933–1942.

[43] Luca Oneto, Sandro Ridella, and Davide Anguita. 2017. Differential privacy and generalization: Sharper bounds with applications. *Pattern Recognit. Lett.* 89 (2017), 31–38.

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Proceedings of Annual Conference on Neural Information Processing Systems 2019, NeurIPS '19.* 8024–8035.

[45] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP '14.* 1532–1543.

[46] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. 2017. Privacy-Preserving Deep Learning: Revisited and Enhanced. In *Proceedings of Applications and Techniques in Information Security - 8th International Conference, ATIS '17 (Communications in Computer and Information Science, Vol. 719).* 100–110.

[47] Lianyong Qi, Xuyun Zhang, Wanchun Dou, Chunhua Hu, Chi Yang, and Jinjun Chen. 2018. A Two-stage Locality-sensitive Hashing Based Approach for Privacy-preserving Mobile Service Recommendation in Cross-platform Edge Environment. *Future Gener. Comput. Syst.* 88 (2018), 636–643.

[48] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17.* 130–137.

[49] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. *CoRR* abs/1205.2618 (2012).

[50] Sandra Servia Rodríguez, Liang Wang, Jianxin R. Zhao, Richard Mortier, and Hamed Haddadi. 2018. Privacy-Preserving Personal Model Training. In *Proceedings of the 2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation, IoTDI '18.* 153–164.

[51] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW '01.* 285–295.

[52] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha A. Larson, Nuria Oliver, and Alan Hanjalic. 2012. CLiMF: Learning to Maximize Reciprocal Rank with Collaborative Less-is-more Filtering. In *Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12.* 139–146.

[53] Daniel Smilkov, Nikhil Thorat, Yannick Assogba, Ann Yuan, Nick Kreeger, Ping Yu, Kangyi Zhang, Shanqing Cai, Eric Nielsen, David Soergel, Stan Bileschi, Michael Terry, Charles Nicholson, Sandeep N. Gupta, Sarah Sirajuddin, D. Sculley, Rajat Monga, Greg Corrado, Fernanda B. Viégas, and Martin Wattenberg. 2019. TensorFlow.js: Machine Learning For The Web and Beyond. In *Proceedings of Machine Learning and Systems 2019, MLSys '19.*

[54] Mengkai Song, Zhibo Wang, Zhifei Zhang, Yang Song, Qian Wang, Ju Ren, and Hairong Qi. 2020. Analyzing User-Level Privacy Attack Against Federated Learning. *IEEE J. Sel. Areas Commun.* 38, 10 (2020), 2430–2444.

[55] Gabriele Sottocornola, Panagiotis Symeonidis, and Markus Zanker. 2018. Session-based News Recommendations. In *Companion of the Web Conference 2018, WWW '18.* 1395–1399.

[56] Harald Steck. 2015. Gaussian Ranking by Matrix Factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15.* 115–122.

[57] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM '19.* 1441–1450.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of Annual Conference on Neural Information Processing Systems 2017, NeurIPS '17.* 5998–6008.

[59] Qinglong Wang, Amir Yahyavi, Bettina Kemme, and Wenbo He. 2015. I Know What You Did on Your Smartphone: Inferring App Usage over Encrypted Data Traffic. In *Proceedings of the 2015 IEEE Conference on Communications and Network Security, CNS '15.* 433–441.

[60] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning. In *Proceedings of the 2019 IEEE Conference on Computer Communications, INFOCOM '19*. 2512–2520.

[61] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM '17*. 495–503.

[62] Teng Xiao, Shangsong Liang, and Zaiqiao Meng. 2019. Hierarchical Neural Variational Model for Personalized Sequential Recommendation. In *Proceedings of the World Wide Web Conference, WWW '19*. 3377–3383.

[63] Pengtao Xie, Misha Bilenko, Tom Finley, Ran Gilad-Bachrach, Kristin E. Lauter, and Michael Naehrig. 2014. Crypto-Nets: Neural Networks over Encrypted Data. *CoRR* abs/1412.6181 (2014).

[64] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Jiajie Xu, Victor S. Sheng, Zhiming Cui, Xiaofang Zhou, and Hui Xiong. 2019. Recurrent Convolutional Neural Network for Sequential Recommendation. In *Proceedings of the World Wide Web Conference, WWW '19*. 3398–3404.

[65] Mengwei Xu, Jiawei Liu, Yuanqiang Liu, Felix Xiaozhu Lin, Yunxin Liu, and Xuanzhe Liu. 2019. A First Look at Deep Learning Apps on Smartphones. In *Proceedings of the World Wide Web Conference, WWW '19*. 2125–2136.

[66] Mengwei Xu, Feng Qian, Qiaozhu Mei, Kang Huang, and Xuanzhe Liu. 2018. DeepType: On-Device Deep Learning for Input Personalization Service with Minimal Privacy Concern. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4 (2018), 197:1–197:26.

[67] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated Machine Learning: Concept and Applications. *ACM Trans. Intell. Syst. Technol.* 10, 2 (2019), 12:1–12:19.

[68] Fajie Yuan, Xiangnan He, Haochuan Jiang, Guibing Guo, Jian Xiong, Zhezhao Xu, and Yilin Xiong. 2020. Future Data Helps Training: Modeling Future Contexts for Session-based Recommendation. In *Proceedings of the Web Conference 2020, WWW '20*. 303–313.

[69] Ming Zeng, Tzu-Heng Lin, Min Chen, Huan Yan, Jiaxin Huang, Jing Wu, and Yong Li. 2018. Temporal-Spatial Mobile Application Usage Understanding and Popularity Prediction for Edge Caching. *IEEE Wirel. Commun.* 25, 3 (2018), 36–42.

[70] Lingchen Zhao, Qian Wang, Qin Zou, Yan Zhang, and Yanjiao Chen. 2020. Privacy-Preserving Collaborative Deep Learning With Unreliable Participants. *IEEE Trans. Inf. Forensics Secur.* 15 (2020), 1486–1500.

[71] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18*. 1079–1088.

[72] Michael Zhu and Suyog Gupta. 2018. To Prune, or Not to Prune: Exploring the Efficacy of Pruning for Model Compression. In *Proceedings of the 6th International Conference on Learning Representations, ICLR '18, Workshop Track Proceedings*.