Privacy-Aware Federated Learning for Page Recommendation

Shuai Zhao * Roshani Bharati * Cristian Borcea † Yi Chen *
* New Jersey Institute of Technology, Martin Tuchman School of Management
Leir Research Institute for Business, Technology, and Society, Newark, NJ, USA
† New Jersey Institute of Technology, Department of Computer Science, Newark, NJ, USA
Email: sz255@njit.edu, rb485@njit.edu, borcea@njit.edu, yi.chen@njit.edu

***Abstract*—Traditional page recommendation models are endangered by stricter privacy regulations, such as the General Data Protection Regulation (GDPR). The performance of these models suffer when only a part of the users share their personal data, such as cookies, with web servers, while the rest of the users choose to opt-out from sharing these data. Furthermore, these models are not designed to provide recommendations for users who do not share their data. This paper addresses the question of how to provide good page recommendations to all users, independent of their privacy attitudes. We propose Fed4Rec, a privacy-preserving framework for page recommendation based on federated learning (FL) and model-agnostic meta-learning (MAML), which allows machine learning models to train on data collected from both *public users*, who share data with the server, and *private users*, who do not share data with the server. Fed4Rec enables recommendations for both public users, computed at the server, and private users, computed at their local devices. Private users' data are stored only on user devices and never shared with the server. FL is used to train on local data, and Fed4Rec shares with the server only partial model parameters, computed on local devices. MAML is used to jointly train on the public data and the model parameters from the private users. We compare Fed4Rec against several baseline frameworks, using a publicly available dataset from a large news portal. The results show that Fed4Rec outperforms the baselines in terms of recommendation accuracy. We also conduct one ablation study to examine the impact of varying the ratio between the number of public and private users. Fed4Rec performs better than the baselines for all ratios, but it is especially beneficial when the percentage of public users is low.**

***Index Terms*—page recommendation, federated learning, meta-learning, privacy regulation, deep learning**

I. Introduction

Effective recommendation of relevant online articles is widely used in industry and extensively studied by the research community. Traditional page recommendation models rely on centralized data repositories that store user visits and browsing behaviors [1], [2]. A page recommendation model is trained on these data to identify the users' interests and make relevant page recommendations. However, with the implementation of privacy regulations around the world, such as the the General Data Protection Regulation (GDPR) by the European Union (EU) in 2018 [3] and the California Consumer Privacy Act (CCPA) in 2020 [4], the performance of traditional page recommendation models will start to suffer due to a lower amount of data. Furthermore, these models are not designed to work for users who do not share data with the web servers. To be compliant with the new privacy regulations, a website asks users whether they are willing to accept cookies and share data with the publisher, as shown in Figure 1. Typically, some users accept, while others reject sharing personal data with the web servers.

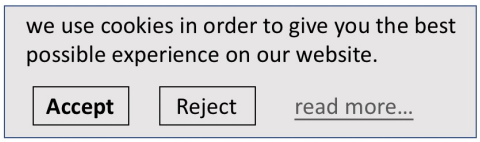

Fig. 1. Example of cookie notice required by GDPR.

In this research, we consider both types of users: users who share personal data with the server, referred to as *public users*, and users who refuse to share data with the server, referred to as *private users*. A publisher stores and has access to the data of public users. The data of the private users, on the other hand, are stored on their local devices and are not available to the publisher. Existing page recommendation studies do not consider this type of user privacy preference; they just use the data collected by the server (i.e., data from public users in our case). A simple solution to provide page recommendations for both types of users is to use multiple instances of a traditional page recommendation model. One instance is trained on the dataset of the public users at the publishers, and then used to provide recommendations for these users. The other instance can be trained at each device of a private user, and then used to provide recommendations for each such user. Unfortunately, training models separately for all public users and each individual private user cannot perform well because each model will under-learn.

The main challenge of our research is how to leverage data from both types of users in order to provide good page recommendations for all users, while respecting the privacy preferences of the private users. To solve this challenge, we propose Fed4Rec, a privacy-preserving framework for page recommendation based on Federated Learning (FL) [5] and model-agnostic meta-learning (MAML) [6], which allows machine learning models to train on data collected from both

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

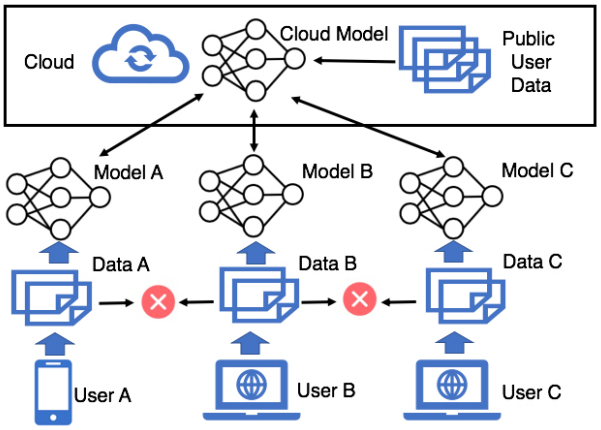

Fig. 2. Main idea of the Fed4Rec framework.

public users and private users. FL helps Fed4Rec because it can learn an algorithm across multiple decentralized edge devices without exchanging their data samples. FL was proven effective for applications such as keyboard predictions [7] and self-driving cars [8]. MAML helps Fed4Rec because it can adapt quickly to new users in recommender systems [9], [10]. Furthermore, MAML is designed to improve classification or regression performance by learning with only a small amount of training data. This feature allows Fed4Rec to predict user's preferences based on only a limited number of visited pages.

The main idea of Fed4Rec is illustrated in Figure 2. FL trains local models on the devices of private users and uploads the model parameters to the cloud to create a global cloud model. MAML is used to build this model by jointly learning from the FL model parameters and the public dataset. The intuition behind MAML is that some internal representations are more transferable than others, and the neural network may learn internal features that are broadly applicable to all users. Specifically, MAML in Fed4Rec allows private users to benefit from the public users' data. Fed4Rec is designed to work with gradient-optimized models that have two layers: a base, user-agnostic layer, and an adaptation, user-specific layer. The base layer feeds into the adaptation layer. In this paper, we design a model in which the base layer is a bi-directional Gated Recurrent Unit (GRU) [11] layer to model the page visit sequences, and the adaptation layer is an attention layer [12] to model the contextual information for fast user adaptation.

We evaluate Fed4Rec using the Globo dataset [13], provided by the most popular news portal in Brazil [13]. In the evaluation, we compare against three baseline frameworks: (i) Arden [14], which trains the model on a public dataset and applies it on private users, (ii) Transfer Learning (TL), which trains the model on a public dataset and then re-trains it on the private user data, and (iii) Federated Average (FA) [15], which creates an FL global model based on local training at private users and model parameter aggregation in the cloud.

The experimental results show that Fed4Rec outperforms the baselines in terms of recommendation accuracy. We also conduct an ablation study to examine the impact of varying the ratio between the number of public and private users. Fed4Rec performs better than the baselines for all ratios, but it is especially beneficial when the percentage of public users is low. The main contributions of this paper are summarized as follows:

- We identify a new problem that will affect the effectiveness of page recommendation models, caused by the recent introduction of online privacy regulations around the world. The problem consists in splitting the user population into public users and private users. To the best of our knowledge, we are the first to study page recommendation models in this new setting.
- We propose a novel solution for this problem, Fed4Rec, which uses FL and MAML to jointly learn a page recommendation model from private and public user data. We also propose a gradient-optimized, two-layer deep learning model that works in conjunction with Fed4Rec. This model learns the user-agnostic page visit sequence and adapts quickly to user preferences. Fed4Rec is able to protect user privacy, while integrating a large variety of user behaviors.
- Our experiments demonstrate the effectiveness of Fed4Rec on improving model accuracy. With more users refusing to share their personal data with web servers, it is important to notice that Fed4Rec performs best, compared to the baselines, when the percentage of public users is relatively low.
- The proposed framework can be leveraged for other applications that may have both public and private user data, such as video recommendation, e-commerce product recommendation, or mobile user location prediction.

The remainder of the paper is organized as follows. Section II discusses the related work. Fed4Rec and its associated deep learning model are described in Section III. The experimental evaluation is presented in Section IV. Section V concludes the paper and discusses future work.

II. Related Work

A. Page Recommendation

Collaborative filtering, content-based recommendation, and matrix factorization are early techniques to achieve wide success in the field of web page recommendation systems [16]. Recently, these methods have been replaced by deep learning based models for higher performance, such as Wide&Deep [17], DeepFM [18], and GRU4REC [11]. It is intuitive to model page reading as a time series problem, which is still the dominant modeling approach in session-based recommendation [19]. For example, GRU4REC applies the Gated Recurrent Units (GRU) method, a type of Recurrent Neural Networks (RNN), to model sequences and obtains promising results.

All of the above-mentioned page recommendation methods rely on centrally-stored user behavior data for model training. User behaviors on websites are privacy-sensitive, and not all

users are comfortable sharing their data with publishers. With new privacy regulations being enacted around the world (e.g., GDPR [3], CCPA [4]), a large number of users are expected to refuse sharing data with web servers. Fed4Rec gives users the choice to keep their data private or to share it with the server in exchange for a higher service quality. Notwithstanding their privacy choices, both users types can benefit from high accuracy page recommendations in Fed4Rec.

B. Federated Learning and Data Privacy

Our general approach belongs to the family of federated learning (FL) techniques. FL is a decentralized learning approach in which a model is trained on every user device and, then, its updated parameters are shared with the server to build an aggregated, global model [5], [20]. FL has several key properties or challenges, which do not exist in other types of distributed learning: the training data on a given client is not representative of the population distribution (Non-IID); some users make much heavier use of the service or app than others, leading to varying amounts of local training data (Unbalanced); the number of clients are large (Massively Distributed); the communication between clients and the cloud server is expensive (Limited Communication). FL has been successfully used in many types of applications, such as smart phone keyboard prediction [21] and healthcare [22].

Most of the previous learning models train using either only public users (i.e., traditional centralized models) or only private users (i.e., FL models). In contrast, Fed4Rec trains jointly on public and private users. There is only one work that considers the same setting, Arden [14], but this work does not target page recommendation nor does it utilize FL. In Arden, the server uses the public data to train models, and every client (private device) sends their encrypted input data to the server to make predictions. Arden has two limitations. First, it does not utilize the private data in the training phase, which is a waste. Second, there could be large communication costs between a client and the server if the client's input is large. Comparatively, in our framework, private users re-train the model using their data before making predictions; also, private users only share model parameters instead of raw data with the server, which reduces the communication cost substantially. In Section IV, we experimentally compare Fed4Rec and Arden.

C. Meta-Learning

One of the main challenges of FL is the statistical heterogeneity of the data. Data residing in different devices are not identically distributed [23]. Additionally, in the case of online page recommendation, users' interests in articles change continuously over time, and models need to quickly adapt to the changes based upon few available new data points [24].

Meta-learning, also called learning-to-learn, aims to train a model that can rapidly adapt to a new task with a few examples [6]. It can be classified into three types: metric-based, memory-based, and optimization-based meta-learning. The metric-based approach is usually based on the computation of similarity between different tasks [25]. However, this only works in a centralized learning setting because it needs to compare the similarity between users via their personal information and recorded behavior. In our case, this is not possible because the data of private users are not shared with each other (or with the server). Memory-based learning requires a specific order of tasks and utilizes reinforcement learning to optimize the final goal. In our case, however, there is no order among tasks or users. Recently the adaption of meta-learning for recommendation systems mainly utilizes the optimization-based meta-learning, and it regards each unique user as a different task [9], [10].

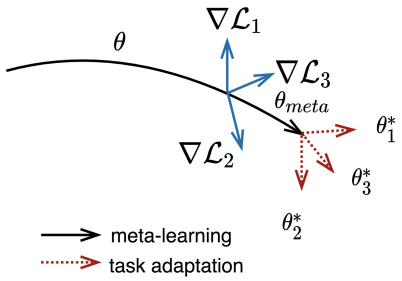

Fig. 3. Illustration of the optimization-based meta-learning algorithm.

Among the optimization-based meta-learning algorithms, MAML [6] is the most popular. MAML learns the best model initialization parameters θ through the gradient descent method, as shown in Figure 3. For new tasks T_1, T_2, T_3, the model can quickly adapt to the best model parameters, θ_1^*, θ_2^*, and θ_3^*, respectively. In order to achieve this goal, first, the model is optimized separately for each task using gradient descent. Then, in the meta-optimization phase, it aggregates the different gradient directions of the learned models for each task ($\nabla\mathcal{L}_1, \nabla\mathcal{L}_2, \nabla\mathcal{L}_3$). After that, it minimizes the losses with the learned models (learning-to-learn process). In the inference period, the model only needs the limited data to quickly adapt to a new task.

In Fed4Rec, we regard each task as estimating a user's article preferences and reading habits (such as short-term or long-term influence on current reading interest) in the page recommendation. Based on this idea, we propose a MAML-based technique to learn a good model initialization based on the public users and then share it with the private users. The private users can then utilize the shared model to adapt quickly based on their private data in order to increase personalization.

III. System Design

Given the historical page visit behaviors of private users U_v and public users U_p, in which a publisher's server can only access the data from public users, our goal is to recommend relevant pages to all users by jointly learning the behaviors of the two types of users, private and public.

Fed4Rec learns from both types of users by combining FL and MAML. At the private user side, it uses FL to train a local model based on the user's data and uploads the trained model parameters to the server. At the server side, it uses

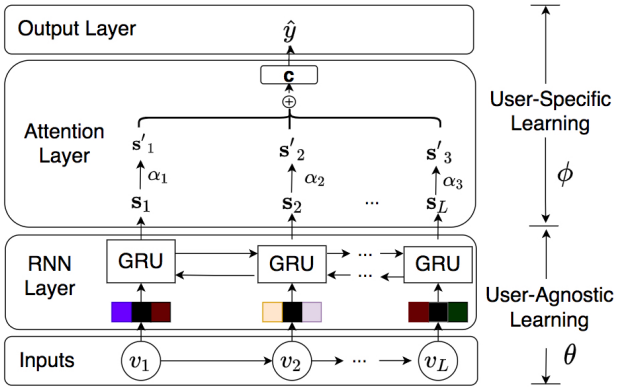

Fig. 4. Page recommendation model.

MAML to jointly learn a global model from the local model parameters uploaded by private users and the public users' raw data. This global model can then be downloaded by private users to perform page recommendations on their devices or can be used at the server for page recommendations for public users.

We continue this section with a description of our page recommendation model that works in conjunction with our framework. Then, we describe the Fed4Rec framework and how it uses this model.

A. Page Recommendation Model

Figure 4 shows our gradient-optimized, two-layer model. The first layer is the user-agnostic RNN layer, and the second one is the user-specific attention layer. The model assumes that each user has a page visit sequence, stored at the server for public users and at the user's device for private users. Following previous work [11], [26], we use a Recurrent Neural Network (RNN)-based model for page recommendation, which has been shown to perform well in sequential data modeling. Furthermore, we add the attention mechanism [12] to model the contextual information that optimizes the attention parameters of each "task" for fast adaptation. This alleviates the need for different metric spaces across different users.

For the RNN layer, we choose a bi-directional Gated Recurrent Unit (GRU) layer to model the page visit sequence. GRU and LSTM [27] have comparable performance for many applications [28], but we choose GRU because its network is much simpler and therefore more efficient to train. Without loss of generality, one could replace GRU with LSTM in our proposed model. The key to GRU is the multiplicative gates, which allow GRU memory cells to store and access information over long periods of time, thereby avoiding the vanishing and exploding gradient problems. GRU takes fixed-length L page visit sequences as the input. We utilize bi-directional GRU because it captures hidden dependencies between the two directions and existing work demonstrates that it performs better than uni-directional versions [29]. The details of bi-directional GRU are defined as follows:

$$
\begin{aligned}
z_t^f &= \sigma(\theta_z^f \cdot [h_{t-1}^f, v_t]) \\
r_t^f &= \sigma(\theta_r^f \cdot [h_{t-1}^f, v_t]) \\
\tilde{h}_t^f &= tahn(\theta^f \cdot [r_t^f * h_{t-1}^f, v_t]) \\
h_t^f &= (1 - z_t^f) * h_{t-1}^f + z_t^f * \tilde{h}_t^f \\
z_t^b &= \sigma(\theta_z^b \cdot [h_{t+1}^b, v_t]) \\
r_t^b &= \sigma(\theta_r^b \cdot [h_{t+1}^b, v_t]) \\
\tilde{h}_t^b &= tahn(\theta^b \cdot [r_t^b * h_{t+1}^b, v_t]) \\
h_t^b &= (1 - z_t^b) * h_{t+1}^b + z_t^b * \tilde{h}_t^b \\
\mathbf{s}_t &= [h_t^f, h_t^b]
\end{aligned}
\tag{1}
$$

In which h_t^f and h_t^b are the forward and backward hidden states at timestamp t. The state at timestamp t is the concatenated states of the two directions: $\mathbf{s}_t$. h_{t-1}^f and h_{t+1}^b are the forward hidden outputs at time stamp $t-1$ and backward hidden outputs at time stamp $t+1$ respectively. σ is the sigmoid function. v_t is the visited page at timestamp t. $z_t^f, r_t^f, \tilde{h}_t^f, z_t^b, r_t^b, \tilde{h}_t^b$ are the intermediate calculations. $\theta_z^f, \theta_r^f, \theta^f, \theta_z^b, \theta_r^b, \theta^b$ are the bi-directional GRU model layer parameters. To simplify, we use θ to refer to all the layer parameters mentioned above. Therefore, bi-directional GRU takes the L-sequence of page embedding $\mathbf{v} = [v_1, v_2, ..., v_L]$ as the input, and it outputs the corresponding sequences of states.

$$
\mathbf{s}_t = f(v_t; \theta) \tag{2}
$$

The representation learner $f(\cdot, \theta)$ in equation 2 encodes the input sequence $\mathbf{v}$ to a corresponding sequence of states $[\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_L]$, where f in our case is a recurrent bi-directional GRU with parameter θ. The goal of learning θ is to obtain meta-learned task-agnostic parameters that can provide meaningful encodings of the input page visit sequences.

For the attention layer, similar to sentiment analysis on sentence text input [30]–[32], a content-based global attention mechanism is added in order to enable the model to focus on different aspects of pages [33], such as page topics. The idea is to model the attention influence from different previous states in different aspects. The specific attention mechanism is defined as follows:

$$
\begin{aligned}
\alpha_t &= \boldsymbol{\phi}_{ATT}^T \mathbf{s}_t \\
\mathbf{s'}_t &= \alpha_t \mathbf{s}_t \\
\mathbf{c} &= \frac{1}{L} \sum_{t=1}^{L} \mathbf{s'}_t
\end{aligned}
\tag{3}
$$

where $\boldsymbol{\phi}_{ATT}$ represents the attention parameter vector. For each memory state $\mathbf{s}_t$, we calculate its inner product with the attention parameter, resulting in a scalar α_t. The scalar α_t rescales each state $\mathbf{s}_t$ into $\mathbf{s'}_t$, which are averaged to obtain the final representation $\mathbf{c}$ of a user. The attention retrieves relevant information from a sequence and learns the importance of each past page visit through attention weight α_t that contributes to the final prediction.

Once an input page visit sequence is encoded into the vectorized representation $\mathbf{c}$, we apply a softmax classifier parameterized by ϕ_W to obtain the prediction $\hat{y}$.

$$\hat{y} = softmax(\mathbf{c}; \phi_W) \tag{4}$$

The parameters are learned using a categorical cross-entropy loss function, which is commonly used in multi-class classification problems. Formally the loss function is defined as follows:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y \cdot log(\hat{y}) \tag{5}$$

in which y is the ground-truth label and N is the number of samples. For simplicity, we refer to $\phi = \{\phi_{ATT}, \phi_W\}$ as the user-specific learned parameters.

B. Federated Page Recommendation Framework

The Fed4Rec framework works with gradient-optimized models that have one user-agnostic and one user-specific layers, such as the model we described above. Our next problem is how to learn the model in a setting that includes both public and private users. Figure 5 illustrates the learning framework of Fed4Rec. Our framework has two components: one server and multiple clients (e.g., laptops, smart phones). Each client represents one private user and stores the data for this user. The public users' data are stored at the server. Therefore, our problem can be further split into two sub-problems:

- At the **client** side: Given the downloaded global model from the server, how to update the local model using the client private data?
- At the **server** side: Given the uploaded parameters of the trained local models and the raw data of public users, how to aggregate and update the global model?

The communication between the clients and the server cannot be in the form of raw data. First, the private users, by definition, do not want the server to have their raw data. Second, while it is possible to transfer the raw data of the public users to each client, it would take too much bandwidth due to the large amount of public data, especially because the data of public users are being generated in a stream. Third, the public data may not even fit in the client storage, especially on mobile devices. Given these reasons and inspired by federated learning [15], the communication between the clients and the server is in the form of model parameters. In this way, Fed4Rec can protect user privacy as well as decrease the bandwidth/storage requirements. In order to further protect user privacy, Fed4Rec could add local differential privacy (LDP) onto the model parameters in the communication phase [34]. In order to achieve this goal, a common method is to add Laplace noise to the data. In this paper, we focus on the overall framework for model training and leave LDP for future work.

The server starts with training the recommendation model using the data shared by public users. The server uses model

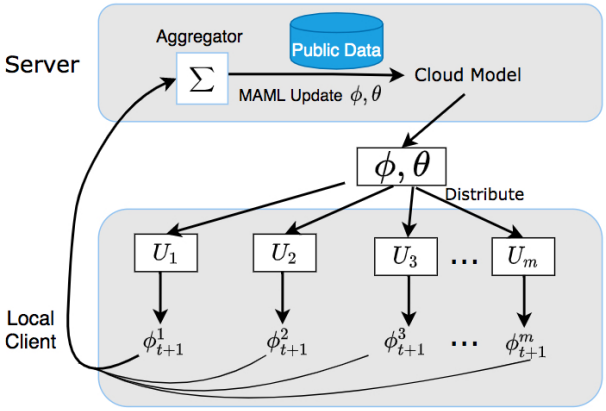

Fig. 5. Fed4Rec learning framework for page recommendation.

agnostic meta-learning (MAML) to perform this step. We use a popular version of MAML that omits the second derivatives for the meta-optimization, resulting in a simplified and faster implementation, known as First-Order MAML (FOMAML) [6], [35]. The updated gradient is calculated as follows:

$$\begin{aligned}\phi, \theta &= \phi, \theta - \beta\nabla_{\phi,\theta}\mathcal{L}(\phi, \theta - \alpha\nabla_{\phi,\theta}\mathcal{L}(\phi, \theta)) \\ &= \phi, \theta - \beta\nabla_{\phi,\theta}\mathcal{L}(\phi', \theta') \\ &= \phi, \theta - \beta(\nabla_{\phi',\theta'}\mathcal{L}(\phi', \theta') \cdot \nabla_{\phi,\theta}(\phi', \theta')) \\ &\approx \phi, \theta - \beta\nabla_{\phi',\theta'}\mathcal{L}(\phi', \theta')\end{aligned} \tag{6}$$

In which, ϕ' and θ' are the updated model parameters based on the learning process. ϕ is learned from the attention layer in the model, and θ is learned from the RNN layer in the model. The final "learning-to-learn" process, shown in Equation 6, is equivalent to simply remembering the last gradient and applying it to the initial parameters. Therefore, the data of the public users can be utilized to update the whole model.

At the client device, similar to federated average [15] in which not all private users join the training in each global learning round, Fed4Rec uses a parameter r to control the ratio of randomly selected private users in order to save the communication cost. For each client participating in a learning round, as shown in Figure 5, Fed4Rec replicates the model parameters ϕ, θ from the server. Then, it adapts the model on the data of each client participating in that round to learn their personalized models by adapting the high-level attention layer. Once the newly trained model is updated at a client device, its parameters are uploaded to the server to be used in the next learning round.

There are several advantages of our Fed4Rec solution. First, since the number of data samples at each private user is limited and the majority of users only have a few new data samples, this approach can avoid overfitting the models at clients. As mentioned earlier, each client only retrains the user-specific parameters. Another advantage of partial retraining is to save computation resources because clients, such as phone or tablets, have limited battery and computing power. Third, this approach saves network bandwidth from the clients to the

Algorithm 1: Fed4Rec

1 **Require:** η_1, η_2 : step size hyperparameters
2 **Require:** U_p, U_v : public users and private users
3 **Require:** randomly initialized model parameters θ_0, ϕ_0
4 **Server executes:**
5 **for** *each global round* $t = 1, 2, \ldots, G$ **do**
6 **global update (First-Order MAML):**
7 $\phi_t \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} \phi_t^k$ when $t \neq 1$, otherwise ϕ_0
8 $\phi_t', \theta_t' \leftarrow \phi_t, \theta_t - \eta_1 \nabla_{\phi_t, \theta_t} \mathcal{L}(\phi_t, \theta_t)$
9 $\phi_{t+1}, \theta_{t+1} \leftarrow \phi_t, \theta_t - \eta_2 \nabla_{\phi_t', \theta_t'} \mathcal{L}(\phi_t', \theta_t')$
10 **local update:**
11 $U_v^t \leftarrow$ (randomly selected $r\%$ of private users in U_v)
12 **for** *each client* $k \in U_v^t$ ***in parallel*** **do**
13 $\phi_{t+1}^k \leftarrow$ ClientUpdate$(k, \theta_{t+1}, \phi_{t+1})$
14 **end**
15 **end**
16 **ClientUpdate**$(k, \theta_{t+1}, \phi_{t+1})$ *// run on client* k
17 $\mathcal{B} \leftarrow$ split private data of client k into batches
18 **for** *each local epoch* i *from 1 to* E **do**
19 **for** *batch* $b \in \mathcal{B}$ **do**
20 $\phi_{t+1}^k \leftarrow \phi_{t+1} - \eta_1 \nabla_{\phi_{t+1}} \mathcal{L}(\phi_{t+1}, \theta_{t+1})$
21 **end**
22 **end**
23 return ϕ_{t+1}^k to the server

server in that only the user-specific parameters, instead of the whole model parameters, are transferred to the server. Fourth, this approach makes it harder to infer user private data from model parameters from the high-level neural network layer compared to model parameters from the low-level embedding layer [36].

We present the pseudo-code of Fed4Rec in Algorithm 1. The initial learning starts from the public users' data because of the requirement to update the whole model. The model parameters ϕ_0 and θ_0 are randomly initialized. If it is not the first round, the server aggregates the received model weights, i.e., taking the weighted average of the received model weights from the clients (line 7), in which n_k is the number of training samples at client k and n is the total number of training samples from selected clients. Then, the global model is updated through FOMAML (lines 8-9). Afterward, the server sends the updated global model to the clients (lines 12-13). On the client side, after learning the models through several local epochs E, a client sends the updated model parameters ϕ to the server (lines 16-22). The learning process is alternatively conducted at the server and the clients, and there are multiple rounds of communications between the server and the clients. The training phase ends when the model converges or reaches the maximum number of global rounds G.

TABLE I
STATISTICS OF THE SAMPLED GLOBO DATASET

Period	16 days	# Users	167,863
#Articles	10,407	# Page views	1,691,996

IV. EVALUATION

This section presents the evaluation of Fed4Rec. The evaluation has three objectives: (1) assess Fed4Rec's recommendation accuracy compared to three baseline frameworks and a centralized model that considers all data public; (2) measure the impact of the ratio between public and private users on recommendation accuracy. While it is expected to achieve higher accuracy for larger ratios of public users, it is important to know if the ratio has a different influence on different frameworks; (3) compare the communication cost between the clients and the server for training of Fed4Rec and the three baselines; This is important especially when the clients are mobile devices, which may have bandwidth limitations.

A. Dataset and Experimental Settings

We perform the evaluation on the Globo dataset, provided by the most popular news portal in Brazil [13]. The statistics of the datasets are shown in Table I. Since the page views are time-ordered, usual cross-validation procedures with randomized allocation of events across data splits cannot be applied [37]. We therefore split the whole dataset into six non-overlapping and continuous subsets, and the final performance is based on the average of all six subsets. In order to fully train the embedding for each page, we remove users and articles with less than six page views iteratively, by following the same setting in [11], [26]. The last 10% of data are hold for testing, and the ratio between training, validation, and testing is 8:1:1 in temporal order. Since the page visit information of private users can never be recorded in reality, here we randomly assign all users into private users and public users to conduct our experiments. $\beta \in [0, 1]$ is the parameter to control the ratio of public users among all users. The model performance is measured on both public and private users.

As we discussed in III, Fed4Rec can work with different models. In addition to the model described in Section 4, we implemented one more model by replacing the bidirectional GRU layer with a convolutional neural network (CNN) layer. The models are implemented using Tensorflow. The dimension of page embedding is set to 250. The page embedding was initialized by the embeddings learned from the article metadata attributes, in which a separate model is trained to classify article categories (editorial subsection in the news portal) based on its textual content and metadata. The details of learning the initial page embeddings through page content are described in [38]. The rest of the model parameters are initialized with the popular Xavier uniform method [39]. Since RNN takes fixed-length input sequences, we pad and truncate the length of an input sequence (L) to be 5. The batch size and the maximum number of epochs are set to 32 and 30, respectively. The maximum number of global rounds is set

to 40 ($G = 40$). The experiments are run on a Ubuntu Linux cluster with 4 NVIDIA P100-SXM2 GPUs. The training goal is to minimize our defined loss. We utilize the Adam optimizer [40] to optimize the parameters because of its fast convergence and set the learning rates η_1, η_2 to 0.001 by default. We use early stopping on the validation set with $patience = 2$ to avoid overfitting.

B. Metrics

To quantify the accuracy of page recommendations [11], [26], [41], [42], we selected the Mean Reciprocal Rank and the Hit Rate to measure the top-20 recommended pages.

Mean Reciprocal Rank (MRR@20) is the average of reciprocal ranks of the correctly-recommended items. The MRR metric considers the order of recommendation ranking, where a large MRR value indicates that correct recommendations are in the top of the ranking list. It is formally defined as follows:

$$MRR = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{rank_i} \tag{7}$$

in which N is the number of predictions, and $rank_i$ refers to the rank position of the ground-truth page for the i_{th} prediction. The reciprocal rank is set to 0 when the rank of the ground-truth page exceeds 20. The larger the MRR@20 is, the better the performance is.

Hit Rate (HR@20) is widely used as a measure of predictive accuracy, which checks whether or not the true next item appears in the top ranked items. HR@20 represents the proportion of correctly recommended items among the top-20 items. The larger the HR@20 is, the better the performance is. Compared to MRR, HR does not consider the specific rank information as long as the true next item appears in the top ranked items.

To evaluate the communication cost of the training phase, we split the communication into two components that are examined separately: the communication cost per client for each client-server interaction (measured in **Number of Model Parameters** exchanged between the client and the server per communication round), and the convergence speed (measured in **Number of Communication Rounds** for training).

C. Comparison Systems

The aim of our evaluation is not to evaluate the page recommendation model, but to evaluate the entire Fed4Rec framework that holistically manages public and private users in a mutually beneficial way. We compare Fed4Rec with the following baseline frameworks. To have a fair comparison, all frameworks use the same page recommendation model.

Arden [14] trains the model on public data and applies it directly on private users' data. The private users' input data are transferred to the server to make recommendations, in which random Laplace noise perturbation is added in the communication pipeline to protect user privacy. Arden sacrifices performance to achieve privacy by adding noise. To make the comparison with the other frameworks fair, we do not add the noise perturbation. Arden is originally developed for image recognition, but we adapt it to use our page recommendation model.

TABLE II
RECOMMENDATION ACCURACY OF DIFFERENT FRAMEWORKS FOR ALL USERS

Method	BiGRU + Attention		CNN + Attention	
	HR@20(%)	MRR@20(%)	HR@20(%)	MRR@20(%)
Arden	26.46	8.20	24.94	8.30
TransLearn	26.44	8.19	24.92	8.27
FedAvg	18.45	6.79	16.57	5.78
Fed4Rec	**35.35**	**11.05**	**30.27**	**9.47**
CenModel	39.47	11.86	38.03	11.12

Transfer Learning (TL) utilizes the public data at the server to learn a pre-trained model, similar to Arden. Then, the server sends the pre-trained model to clients. Each client re-trains (fine-tunes) the pre-trained model with their own data and uses this re-trained model to make recommendations afterward.

Federated Average (FA) [15] is a generalization of FedSGD, which allows clients to perform more than one batch update on local data and to exchange the updated weights with the server, rather than the gradients. In FA, all local client models must start from the same initialization, which produces a significant reduction in training [15]. The rationale behind this generalization is that with shared model initialization, averaging the gradients is strictly equivalent to averaging the weights themselves. Since FA only works in the case of all-private users, we assume all users (including public users) are private users. Thus, the public user ratio β is irrelevant to this method.

CenModel is a centralized model that assumes all users are public and applies our model on their data. CenModel acts as an upper bound for the performance that can be achieved by Fed4Rec.

D. Recommendation Accuracy

Table II shows the performance accuracy comparison between Fed4Rec and the baselines, with the ratio of public users $\beta = 0.5$. The results are averaged over all users, public and private. We observe that Fed4Rec outperforms the three baseline frameworks for both models (BiGRU+Attention and CNN+Attention). Furthermore, Fed4Rec's performance is close to the upper bound performance provided by the CenModel.

We notice that FA is the worst method. The main reason is that it considers all users private and there is a large variability in the user behavior. A deep learning model has a powerful ability of representation, but each client could only learn the biased partial gradients. Simply averaging the model parameters cannot effectively learn the aggregated data.

Also, TL does not work well compared to Arden. This is because each local client has limited data to effectively re-train the server model locally, and it is prone to overfitting.

Across all frameworks, the BiGRU+Attention model works better than the CNN+Attention model. This demonstrates the

advantage of BiGRU on modeling sequential data. We also notice that Fed4Rec with CNN+Attention is still better than the baselines using BiGRU+Attention. This proves the significant benefits of Fed4Rec, compared with the baselines.

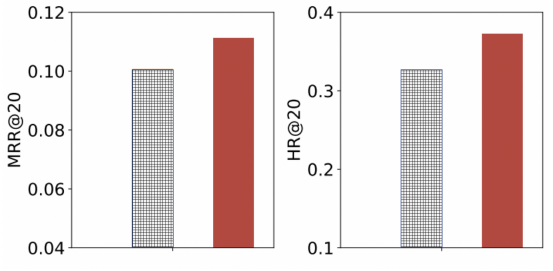

Fig. 6. Recommendation accuracy of different frameworks for public users only: Arden/TL (left bars), and Fed4Rec (right bars).

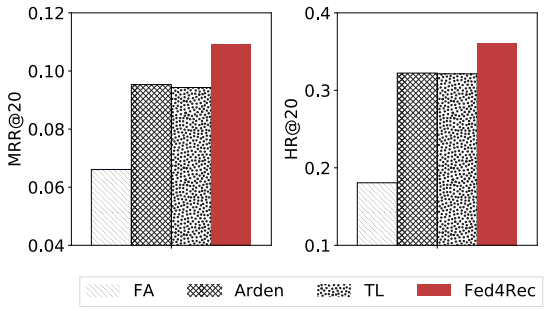

Fig. 7. Recommendation accuracy of different frameworks for private users only.

Next, we zoom into the recommendation accuracy comparison for public users only (Figure 6) and private users only (Figure 7), respectively. Let us notice that FA does not work for public users only; in fact, it considers all users private. Also, Arden and TL obtain the same performance when applied to public users only.

The results show that Fed4Rec achieves the best performance in both cases, and the performance improvement is similar. These results further validate the effectiveness of our joint alternative-learning framework.

E. Impact of the Public User Ratio (β)

Figures 8 and 9 show how the recommendation accuracy of the four frameworks vary with β, the ratio of public users among all users. We observe that the larger $beta$ is, the better the performance a framework achieves. The exception is FA, which assumes all public users work as private users, and therefore it is not impacted by β.

Fed4Rec performs the best for all ratios. This result shows its effectiveness across many potential real-life settings. Let us also emphasize that Fed4Rec's performance is substantially better than the performance of the baselines for lower values of β. This shows that clients are able to learn effectively on their own data and transfer the learned knowledge to the global model at the server. This result validates the advantage

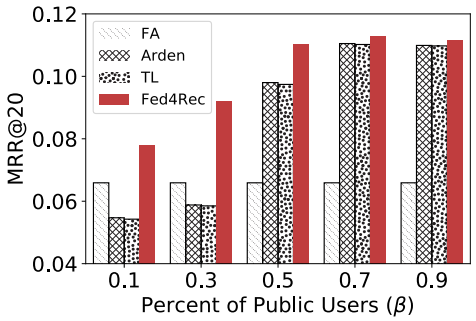

Fig. 8. MRR@20 performance with varying β.

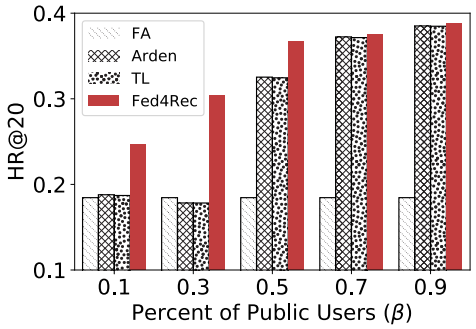

Fig. 9. HR@20 performance with varying β.

of Fed4Rec in learning from private users' model parameters and the effectiveness of updating the user-specific parameters. The results also show that the performance of Arden and TL remains similar with each other across all ratios. Our conjecture is that this is due to the fact that each user in the dataset has limited history and their consecutive behavior vary with time. Thus, TL is not able to effectively learn private user's behavior, even though it fine tunes the model using private user data. Comparatively, Fed4Rec can effectively learn the behavior of private users from their limited and heterogeneous data. It does so because of MAML, which considers each user as a task and works well for few shot learning.

F. Client-Level Communication Cost Analysis

The communication between clients and the server can occur in two phases: training phase and recommendation phase. Arden is different from the other three frameworks in the sense that it does not require communication between the clients and the server in the training phase. It needs communication only in the recommendation phase, where clients send the transformation of the raw data to the server, which then makes the predictions. It is difficult to measure the communication for each client in Arden because the communication depends on the amount of recommendations needed by each client. Unlike Arden, the other frameworks use communication in training, but perform recommendations locally. Therefore, there is no communication cost for these methods in the recommendation phase.

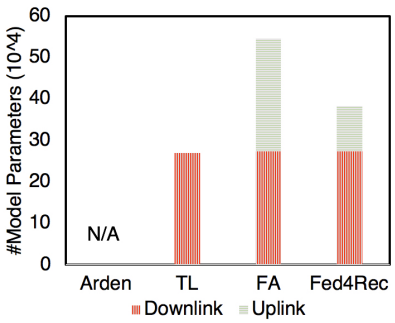

Fig. 10. Communication cost per training round.

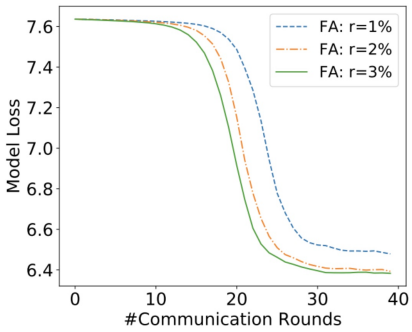

Fig. 11. Training convergence of Federated Average.

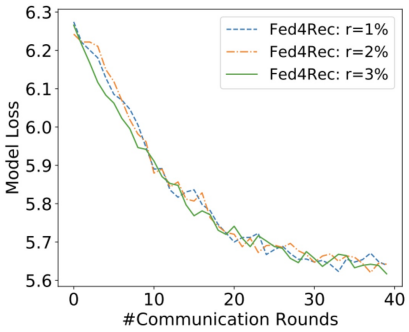

Fig. 12. Training convergence of Fed4Rec.

In the following experiment, we compare the communication cost in the training phase for Fed4Rec, TA, and FA. We also split each client-server interaction into downlink and uplink communication.

Figure 10 shows the communication cost per training round. We observe that Fed4Rec performs better that FA, but worse than TL. Compared to FA, our model saves 60% of the uplink communication from the client to the sever. This is because Fed4Rec requires to train and update only a part of the model structures and uploads only the local model parameters to the server. TL performs best because it does not need to upload anything to the server. A client downloads the global model from the server and then retrains with the local data. We conclude that the extra-communication cost of Fed4Rec vs. TL is the price paid by Fed4Rec for significantly higher recommendation accuracy.

G. Convergence Analysis

Next we explore the convergence of Fed4Rec compared to FA. Since TL and Arden do not have multiple communication rounds between the server and clients, we exclude them in the convergence analysis. Figure 11 shows the convergence of FA, and Figure 12 shows the convergence of Fed4Rec. The parameter r stands for the selection ratio of private users in each communication round. Overall, we observe that Fed4Rec converges significantly faster than FA. In FA, there is a relatively long "flat" period, and then the model converges quickly. The reason for the "flat" period is because of the variability in user behavior, which makes the gradient directions from individual clients to counteract/neutralize each other. This phenomenon is especially obvious when there is a small fraction of sampled users per training round. In Fed4Rec, on the other hand, there is no such "flat" period, and the model starts to converge immediately. This is because Fed4Rec uses the public user data in MAML, and therefore the learned model can adapt quickly to learn at the clients.

We observe that higher r values result in faster convergence for FA. However, the convergence speed of Fed4Rec does not depend on r. This is because of the usage of both public and private users in its training process, and it demonstrates that Fed4Rec can minimize the communication cost by selecting fewer clients per round and obtaining the same level of convergence.

V. Conclusions and Future Work

This paper explores a new problem with real-life implications: privacy-preserving online page recommendation with joint-learning from public users' data and private users' data. To the best of our knowledge, we are the first to study and propose a solution for this problem. Our solution, Fed4Rec, combines federated learning with model-agnostic meta-learning to outperform several baseline frameworks. Fed4Rec performs especially well for lower ratios of public users, which is expected to be seen in real-life due to privacy regulations. We also demonstrate that Fed4Rec is flexible, as it works well with two page recommendation models, BiGRU+Attention and CNN+Attention.

There are two directions that we plan to investigate as future work. First, due to the limitation of the dataset used in our experiments, we assumed there is no significant behavioral pattern gaps between public users and private users. We will look for other datasets to test this hypothesis and relax it if otherwise. Second, since Fed4Rec is a general framework, not restricted to the application domain of page recommendation, future work could extend it to other domains such as video recommendation or e-commerce product recommendation.

Acknowledgement

This work is partially supported by NSF under grant No. DGE 1565478, by the Leir Foundation, and a Faculty Seed

Grant from the Henry J. and Erna D. Leir Research Institute for Business, Technology, and Society at NJIT. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] L. Li, D. Wang, T. Li, D. Knox, and B. Padmanabhan, "Scene: a scalable two-stage personalized news recommendation system," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 2011, pp. 125–134.

[2] F. Garcin and B. Faltings, "Pen recsys: A personalized news recommender systems framework," in *Proceedings of the 2013 International News Recommender Systems Workshop and Challenge*, 2013, pp. 3–9.

[3] (2018) Gdpr website. [Online]. Available: https://ec.europa.eu/info/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules/eu-data-protection-rules_en

[4] (2019) Data protection laws of the world: Full handbook. [Online]. Available: https://www.dlapiperdataprotection.com/2,143,147,151,152,153

[5] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[6] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1126–1135.

[7] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecny, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[8] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities and challenges," *arXiv preprint arXiv:1908.06847*, 2019.

[9] H. Lee, J. Im, S. Jang, H. Cho, and S. Chung, "Melu: Meta-learned user preference estimator for cold-start recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1073–1082.

[10] H. Bharadhwaj, "Meta-learning for user cold-start recommendation," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[11] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *4th International Conference on Learning Representations*, 2016.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[13] (2018) Globo dataset. [Online]. Available: https://www.kaggle.com/gspmoreira/news-portal-user-interactions-by-globocom

[14] J. Wang, J. Zhang, W. Bao, X. Zhu, B. Cao, and P. S. Yu, "Not just privacy: Improving performance of private deep learning in mobile cloud," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2407–2416.

[15] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[16] P. V. Kumar and V. R. Reddy, "A survey on recommender systems (rss) and its applications," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 2, no. 8, pp. 5254–5260, 2014.

[17] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir *et al.*, "Wide & deep learning for recommender systems," in *Proceedings of the 1st workshop on deep learning for recommender systems*, 2016, pp. 7–10.

[18] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 1725–1731.

[19] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 346–353.

[20] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He, "Federated meta-learning with fast convergence and efficient communication," *arXiv preprint arXiv:1802.07876*, 2018.

[21] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[22] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, 2020.

[23] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[24] C. Feng, M. Khan, A. U. Rahman, and A. Ahmad, "News recommendation systems-accomplishments, challenges & future directions," *IEEE Access*, vol. 8, pp. 16 702–16 725, 2020.

[25] M. Vartak, A. Thiagarajan, C. Miranda, J. Bratman, and H. Larochelle, "A meta-learning perspective on cold-start recommendations for items," in *Advances in neural information processing systems*, 2017, pp. 6904–6914.

[26] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 843–852.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[29] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[30] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations*, 2015.

[31] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Advances in neural information processing systems*, 2015, pp. 1693–1701.

[32] S. Sukhbaatar, J. Weston, R. Fergus *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.

[33] X. Jiang, M. Havaei, G. Chartrand, H. Chouaib, T. Vincent, A. Jesson, N. Chapados, and S. Matwin, "Attentive task-agnostic meta-learning for few-shot text classification," in *2nd Workshop on Meta-Learning at Advances in neural information processing systems*, 2018.

[34] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*, 2020, pp. 2938–2948.

[35] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[36] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[37] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach, "Empirical analysis of session-based recommendation algorithms," *User Modeling and User-Adapted Interaction*, pp. 1–33, 2020.

[38] G. de Souza Pereira Moreira, F. Ferreira, and A. M. da Cunha, "News session-based recommendations using deep neural networks," in *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, 2018, pp. 15–23.

[39] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR*, 2015.

[41] P. M. Gabriel De Souza, D. Jannach, and A. M. Da Cunha, "Contextual hybrid session-based news recommendation with recurrent neural networks," *IEEE Access*, vol. 7, pp. 169 185–169 203, 2019.

[42] T. Qi, F. Wu, C. Wu, Y. Huang, and X. Xie, "Fedrec: Privacy-preserving news recommendation with federated learning," *arXiv*, pp. arXiv–2003, 2020.