Birds of a Feather Help: Context-aware Client Selection for Efficient Federated Learning

Hangrui Cao*, **Qiying Pan***, **Yifei Zhu**[1], **Jiangchuan Liu**[2]

[1] Shanghai Jiao Tong University, Shanghai, China
[2] Simon Fraser University, Burnaby, Canada
hangrui@umich.edu, sim_arity@sjtu.edu.cn, yifei.zhu@sjtu.edu.cn, jcliu@sfu.ca

Abstract

Federated learning emerges as an effective technique to train neural network models collaboratively among individual edge devices without centralizing the raw data. However, the data heterogeneity in such a distributed system often breaks the classical IID assumption on data statistics and greatly harms the accuracy and convergence speed. In this paper, we propose to intelligently select clients through exploiting the data correlations among clients to improve learning performance. We propose a novel Neural Contextual Combinatorial Bandit approach, NCCB, that gracefully handles the non-trivial relationship between the extracted features and rewards, and satisfies the combinatorial constraints imposed by the federated learning. We theoretically prove that NCCB has a bounded regret. Extensive experiments on real-world datasets further demonstrate that our approach can achieve up to 50% higher speed to reach target accuracy than the FedAvg baseline, and 17% higher accuracy than state-of-the-art solutions.

Introduction

With the increasing awareness of privacy protection, laws and regulations, such as the GDPR in European Union (European Commission 2018) and the CCPA in California (Bukaty 2019), are established worldwide to restrict the access of raw edge data. This greatly challenges the traditional gather-and-analyze paradigm, where a centralized server collects data from all clients into one place to conduct model training or data analysis. Federated learning is proposed to conduct neural network training via collaboration among clients and servers without uploading the raw data. Since its introduction, it has been widely studied and applied in various machine learning scenarios, including computer vision (Liu et al. 2020b), natural language processing (Yang et al. 2018), medical care (Huang et al. 2019) and etc.

While the existing machine learning theories heavily rely on the assumption of identical independent distribution (IID) of raw data, clients' local data in the federated setting naturally is non-IID. This is also known as data heterogeneity, one of the unique challenges emerged from federated learning, along with device heterogeneity, network heterogeneity, etc. The existence of data heterogeneity has severe consequences on various aspects of federated learning, including sub-optimal prediction performance, longer convergence time, and unfairness among clients (Kairouz et al. 2019).

Previous studies either ignore this non-IID challenge or assume some known local data statistics. For example, many local weight update algorithms are proposed still following the IID setting (Liu et al. 2020a; Uddin et al. 2020). Wang et al. propose a control algorithm to fully utilize all limited network resources under the assumption that datasets follow four specific distributions (Wang et al. 2019). Yu et al. propose to speed up convergence by assuming that the datasets follow identical distributions but may be correlated with each other (Yu, Yang, and Zhu 2019). More recent works start to leverage the posterior information, like local model weights (Wang et al. 2020; Chen et al. 2020), revealed during the training process to help future learning.

However, these attempts in handling the data heterogeneity challenge either fail to model the real world distributions accurately or incur large overhead in extracting posterior information. In this paper, we argue that datasets at different devices, though no longer follow the IID property, actually may correlate with each other, which can then be leveraged to help the federated learning process. For example, the corpus difference of two sport journalist can be smaller than that of one sport journalist and one literature professor, which may help the natural language processing training. The images captured by two cameras in similar scenarios are more likely to be similar than the images captured by the cameras that are deployed for different purposes, which may help the computer vision model training. In fact, such kind of correlations among different entities have been confirmed or validated in various disciplines. One representative example is the language usage correlation between different clients studied in sociology, where clients living in the same social circle also tend to use similar words or phrases, which is formally known as "language homophily" (Aiello et al. 2012). Homophily is generally believed to form due to the process of selection and social influence, and can be vividly captured by an English proverb, "Birds of a feather flock together", which motivates our title (McPherson, Smith-Lovin, and Cook 2001). *Intuitively, if we could capture this type of correlation among individual devices in federated learning, and consider such correlation when selecting clients, a good*

*These authors contributed equally.

global model can be trained by more representative clients with fewer communication rounds.

However, multiple challenges directly emerge if we want to identify and leverage this type of correlation in federated model training. First, compared with traditional distributed machine learning, federated learning enforces tighter requirements on privacy, prohibiting the sharing of raw data. How to capture the correlation among local dataset without revealing private information remains a problem. Second, even if we have successfully extracted the correlations in a privacy-preserving way, the relationship between different correlations and final performance is still unclear. Consequently, how to leverage this relationship to help the client selection process also needs to be designed. Last but not least, rather than selecting one client, we need to select multiple clients per round in federated learning. Selecting m clients per training round from n total clients can have $O(n^m)$ choices in total. How to identify the best combinations in this exponential space is non-trivial.

In this paper, we present a novel Neural Contextual Combinatorial Bandit (NCCB) approach to embed the prior information of clients and select the right set of clients for federated learning. NCCB gracefully handles the privacy-aware context embedding, relationship learning, and combinatorial selection challenges. Specially, our context-aware client selection mechanism consists of two parts, the context feature extraction part and the client selection part. In the first part, a local feature extractor is designed to embed the raw data vectors into a short-length feature vector via a light weight locality sensitive hashing hashing function. In the client selection part, the clients are first grouped into different clusters to reduce the profiling space. We then design a neural network to estimate the non-trivial relationship between feature and reward. A bandit selector acts simultaneously to identify the right set of clients to maximize system utility in each round. In summary, our contributions are as follows:

- We propose a context-aware client selection mechanism that exploits the correlation among clients to improve federated learning.
- A novel neural combinatorial contextual bandit approach is proposed to embed the correlation information as contexts and select a combination of clients for federated learning.
- Different from the classical bandit approaches, our bandit approach not only can learn the non-trivial relationship between features and rewards, but also can non-trivially handle the combinatorial requirement requested specifically in federated learning.
- We prove theoretically that our bandit achieves $\mathcal{O}\left(T\sqrt{\log(1+\frac{Tn}{\lambda})}(1-\frac{\lambda}{Tw}))^J\right)$ bounded regret. Extensive experiments on real-world datasets further validate the superior performances of our approach in accuracy and speed to reach target accuracy.

System Model and Problem Formulation

In this section, we first model the overall federated learning system and users' utility. Then we formulate our client selection problem as a combinatorial bandit problem.

System Model

In a typical federated learning systems, suppose there are n clients in total. For each client $i \in [1, 2, \cdots, n]$, the corresponding dataset is D_i. A federated learning server selects m clients per round t for model training with a total of T rounds. We denote these selected clients as B_t at round t. The server first sends the global weights to initialize the models in local devices[2]. After receiving the weights, each client trains the model based on its own dataset and obtains different weights matrix $w_1, w_2, \cdots, w_n$. These local weights are then aggregated at the server side to update a new global model w_{global}, which are distributed to clients further to start a new iteration. The whole procedure is illustrated in Fig. 1(a). Let w^T denote the final model weight and $\sum_{i=1}^{n} l(w^T, D_i)$ measures the final loss of the model where l is the loss function. The whole federated learning process tries to minimize

$$\min \quad \sum_{i=1}^{n} l(w^T, D_i) \tag{1}$$

We next model the user's utility in the federated settings considering two major factors. Training neural networks with a larger scale of dataset results in a finer model that fits more variations of data. Dataset size of a client thus greatly affects its contribution to the model. In addition, a client with a larger training loss implies that the current global model performs worse locally and thus needs more training to improve the performance with regards to this local dataset. Therefore, from the federated learning's perspective, the utility U of a client i with dataset D_i can be defined as

$$U(i) = |D_i| \sqrt{\frac{1}{|D_i|} \sum_{d \in D_i} Loss(d)^2} \tag{2}$$

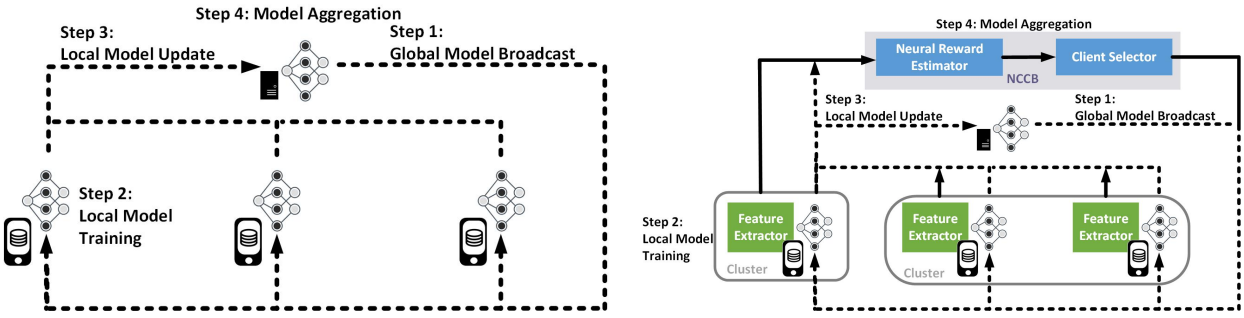
where $Loss(d)$ is the loss function determined by local model. This utility form is also used in (Lai et al. 2021).

With more clients included in training, the global model performs better because of larger size of dataset. However, since the global model is the aggregation of total m local models, as m increases, the contribution of adding one more client to the global model becomes weaker. Consequently, we define the utility of a combination of selected clients to be

$$U(B_t) = \sqrt{\sum_{i \in B_t} U(i)^2} = \sqrt{\sum_{i \in B_t} \left[|D_i| \sum_{d \in D_i} Loss(d)^2 \right]} \tag{3}$$

which is squared root of sum of squared utility functions of all training clients.

[2]Clients and devices are used interchangeably in this paper.



(a) Traditional federated learning system

(b) Our scheme

Figure 1: System overview

Problem Formulation

We study the client selection problem in federated learning so that the best representative clients can be selected to improve the federated learning process. Namely, a fixed number of clients has to be selected to join the federated learning in each round in order to maximize overall utility in Eq. 3.

In the federated learning process, since the utility of each user depends on its training loss, which further depends on the local data, user utilities remain unknown to the server. Exploration is required to train the model on a client and get the corresponding utility. As we explore more clients in the system, their utilities become gradually clear. An efficient system should also exploit previous experience in selecting multiple clients with large utilities. Therefore, the tradeoff of exploration and exploitation in the client selection process makes bandit an appropriate formulation form.

We thus formulate our practical online client selection problem as a combinatorial bandit problem, in which the server and clients can be regarded as the player and arms. Note that unlike the classical bandit problem where only one arm needs to be selected, the combinatorial bandit here aims at capturing the client size requirement in federated learning, where a set of clients, also known as arm set(super arm) in bandit, has to be selected. We use a decision variable $B_t = (x_1^t, x_2^t, \cdots, x_n^t), \forall t \in [1, 2, \cdots, T]$ to denote the selection results in round t. If a client i is selected, its corresponding indication variable $x_i^t = 1$. Otherwise, $x_i^t = 0$. Denote the combination of optimal arms at round t is B_t^*. In the process of exploration and exploitation, the aim of combinatorial bandit is to select a set of arms in each round to minimize the cumulative regret. In summary, our client selection problem is formulated as

$$\min_{x} \quad \mathbb{E}\left[\sum_{t=1}^{T}(U(B_t^*) - U(B_t))\right]$$
$$s.t. \quad \begin{cases} \sum_{i=1}^{n} x_i^t \leq m, \forall t \in [1, 2, \cdots, T] \\ x_i^t \in [0, 1], \forall i \in [1, 2, \cdots, n], t \in [1, 2, \cdots, T] \end{cases} \tag{4}$$

The first constraint limits the number of selected clients for each round. The second indicates the binary selection.

Design and Analysis

In this section, we first provide an overview for our context-aware client selection mechanism. We then introduce the details of our mechanism and conduct regret analysis and complexity analysis after that.

Overview

Our context-aware client selection consists of three parts, a local feature extractor, a reward estimator, and a client selector as illustrated in Fig. 1(b). At the client side, a local feature extractor maps the raw data vector to a feature vector without revealing private information. At the server side, a reward estimator learns the relationship between feature vectors and actual rewards and estimates the next-round rewards. Based on the estimator and other parameters, the client selector returns a list of client to minimize the regret based on a Neural Combinatorial Contextual Bandit.

Privacy-Preserving Context Extractor

The key requirement in the context extractor is to preserve data utility without uploading the raw data. Existing solutions use the weights generated after training on local dataset as a feature for each device. However, this posterior information requires extra model training and incurs extra large computation cost (Wang et al. 2020; Chen et al. 2020). Local sensitive hashing (LSH) hashes similar items into one bucket (Gionis 1999) and is widely-used to generate fingerprints for documents(Zhao et al. 2020) and images (Zhou et al. 2021). For a vector $\boldsymbol{v} = (v_1, v_2, \cdots, v_k)$ with size k, n indices $\{i_1, i_2, \cdots, i_n\}$ from $\{1, 2, \cdots, k\}$ are chosen randomly. Hence, a new vector $\boldsymbol{v'} = (v_{i_1}, v_{i_2}, \cdots, v_{i_n}) = (v'_1, v'_2, \cdots, v'_n)$ of n directions is formed. n different scalar hash functions $h_i : \mathbb{R}^n \to \mathbb{Z}$ defined by Eq. 5 are used to project $\boldsymbol{v'}$.

$$h_i(\boldsymbol{v'}) = \lfloor \frac{v'_i - b_i}{\eta} \rfloor \tag{5}$$

where b_i is uniformly generated in $[0, \eta)$ and η is the quantization step. The generated vector $\boldsymbol{h}(\boldsymbol{v}) = (h_1(\boldsymbol{v'}), h_2(\boldsymbol{v'}), \cdots, h_n(\boldsymbol{v'}))$ extracts the contents of vector $\boldsymbol{v}$ and can be indexed with an integer $g(\boldsymbol{h}(\boldsymbol{v})) : \mathbb{Z}^k \to \mathbb{Z}$. For two near vectors $\boldsymbol{v}, \boldsymbol{u}$ their corresponding hashes $g(\boldsymbol{v}), g(\boldsymbol{u})$ are similar.

Therefore, LSH is a desirable lightweight candidate in distilling the prior information of local data without revealing raw data to server. Among all the LSH algorithms, SimHash(Manku, Jain, and Sarma 2007) is widely used to process large-scale documents(Jafari et al. 2021), and has been used in some federated scenario, like cookie management (Google 2018). Therefore, we also take advantage of SimHash algorithm to extract content features of datasets in the context of NLP federated Learning. In SimHash, we first break the text into n features. Next we use hash functions to convert the features into k-bit binary numbers where k is the hash size. The example features can be hashed into $h_i, 1 \leq i \leq n$. Then, we assign different positive weights $w_i > 0, 1 \leq i \leq n$ to features according to the contents. For each feature i, if the jth bit of its corresponding hash value h_i is 0, we turn w_i into $-w_i$ as the weighted result. Otherwise we keep w_i as the weighted number. As a result, each k-bit hash binary number of feature i is weighted into k integers denoted as $h'_{i1}, h'_{i2}, h'_{i3}, \cdots, h'_{ik}$ where

$$h'_{ij} = \begin{cases} w_i, & h_{ij} == 1 \\ -w_i, & h_{ij} == 0 \end{cases} \tag{6}$$

Then we aggregate all the weighted result $\sum_i h'_{ij}$ for all bits $j \in [1, k]$. Denote the aggregated numbers as $(v_1, v_2, \cdots, v_k)$. Finally we can obtain the fingerprint for the text $(sign(v_1), sign(v_2), \cdots, sign(v_k))$ where $sign(x) : \mathbb{Z} \to [0, 1]$ is defined by

$$sign(x) = \begin{cases} 1 & x > 0 \\ 0 & x < 0 \end{cases} \tag{7}$$

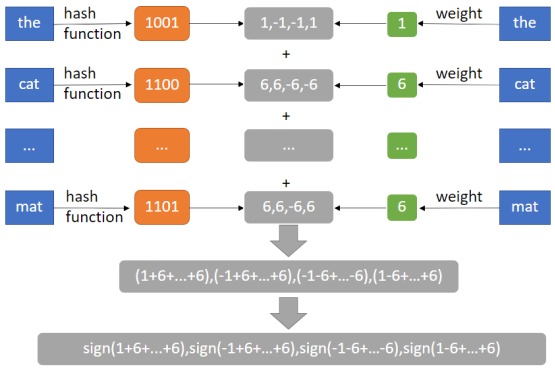


Figure 2: A toy example for SimHash

To facilitate understanding, Fig. 2 demonstrates the SimHash process of a phrase "the cat sat on the mat"[3]. This phrase is first broken into 'the', 'cat', 'sat', 'on', 'the' and 'mat'. After computing the corresponding weight values and hash values from weight and hash function, we negate some of the weight values since the corresponding hash bits are zero rather one. For example, for phrase "the", the signs of second and third weight values are negated and become (1, -1, -1, 1) since the hash value of this word is 1001. After aggregating the corresponding value in each phrase and passing through a sign function, we get the final feature vector.

[3]In practice, we will preprocess text and remove trivial words.

The Neural Combinatorial Contextual Bandit

Based on the represented features, we propose a neural combinatorial contextual bandit (NCCB) approach to embed the feature as context and select clients intelligently. The detailed algorithm design is shown in Alg. 1. The NCCB approach is made up of 3 steps.

Algorithm 1: NCCB

Input: Number of rounds: T; number of total arms: n; number of clusters: c; number of selected arms per round: m; a control constant k

Observe contextual vectors $\boldsymbol{x_i}, i \in \{1, 2, \cdots, n\}$
Cluster all the arms into c clusters $C_1, C_2, \cdots, C_c$
Initialize all the counters $Ct(C_i)$ to 0
Randomly initialize $\boldsymbol{\theta_0}$
for $t \leftarrow 1, ..., T$ **do**
    Initialize the selected arm set $B_t = \phi$
    $C_{un} = \{C_i | Ct(C_i) \leq k\}$.
    $C_{sel} =$ Randomly select $\min\{m, |C_{un}|\}$ clusters from C_{un}
    for cluster $C \in C_{sel}$ **do**
        Randomly select an arm b_i from C
        $B_t = B_t \cup b_i$
    end for
    if $|C_{un}| < m$ **then**
        for $i \leftarrow 1, \cdots, n$ **do**
            Compute upper confidence bound U_i using $f(\boldsymbol{x_{i,t}}; \boldsymbol{\theta_{t-1}})$
        end for
        for $i \leftarrow 1, \cdots, m - |C_{un}|$ **do**
            $b_i \leftarrow \text{argmax}_{j \in \{1, \cdots, n\} \wedge b_j \notin B_t} U_i$
            $B_t = B_t \cup b_i$
        end for
    end if
    Play b_i and observe $r_{b_i,t}$ for all $b_i \in B_t$
    $\boldsymbol{\theta_t} \leftarrow$ train Neural Network using $\{\boldsymbol{x_{b_j}}\}_{b_j \in B_i, i \in \{1, \cdots, t\}}, \{r_{b_i,i}\}_{b_j \in B_i, i \in \{1, \cdots, t\}}$, and $\boldsymbol{\theta_{t-1}}$
    Update counter $Ct(C_i)$ for all $b_j \in B_t$ and $b_j \in C_i$
end for

Preparation Before any trial, the feature vector for user i is denoted as $\boldsymbol{x_i}$, $i \in \{1, 2, \cdots, n\}$. We cluster all the users into c clusters denoted as $C_1, C_2, \cdots, C_c$ using the feature vectors. A counter is set for each cluster to indicate the number of times users in this cluster are selected. An exploring threshold $k \in \mathbb{N}$ controls the number of selection for each cluster. This prevents the abuse of some specific arms and increases explorations especially at early stage of training as we will see in the following. The system initializes the counter of each client to zero.

Client selector A set $C_{un} = \{C_i\}$ includes all the unexplored clusters whose counters are less than or equal to k is

formed. One arm will be selected per chosen cluster. If the number of clusters of these kinds exceeds the required number of arms m, they will be selected uniformly at random. It may happen that the number of clusters in C_{un} is smaller than m. Then, the system computes the upper confidence bound U_i for each arm and determines the top $m - |C_{un}|$ arms with the largest $m - |C_{un}|$ upper confidence bounds from a neural reward estimator. The rewards of all the clients are then predicted and the remaining $m - |C_{un}|$ arms are selected as shown as following:

$$\begin{aligned} &b_{|C_{un}|+1}, b_{|C_{un}|+2}, \cdots, b_m \\ &= (\text{argsort}_{i \in \{1,\cdots,k\} \wedge i \notin [b_1,\cdots,b_{|C_{un}|}]} U_i)[l:r] \end{aligned} \tag{8}$$

where $j \in 1, 2..., n$, argsort is a function that takes the selected U_i array, sort them from smallest to biggest based on the upper confidence bound values and return n indexes, $l = n - (m - |C_{un}|) + 1$ and $r = n$. The arms selected from unexplored clusters C_{un} should be excluded since repeated selections of one arm are prohibited. This procedure is displayed from line 18-21 in Alg. 1. After selection, the system increases the counter of each cluster according to the selected clients.

Neural reward estimator In the mean time, we also need a reward estimator f that takes an arm and its context and estimate the corresponding reward. Unlike the classical linear contextual bandit where the estimation function is assumed to be linear, in our approach, a neural network is used to handle the complex relationship between features and rewards in our problem, which could be parameterized as

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = \sqrt{m} \boldsymbol{W}_L \sigma(\boldsymbol{W}_{L-1} \sigma(\cdots \sigma(\boldsymbol{W_1 x}))) \tag{9}$$

where L is the number of layer, $\boldsymbol{W}_l, l \in [1, \cdots, L]$ are matrices whose elements are coefficients, σ is an activation function and $\boldsymbol{\theta} = [\text{vec}(\boldsymbol{W}_1), \text{vec}(\boldsymbol{W}_2), \cdots, \text{vec}(\boldsymbol{W}_L)]$.

The loss function is defined as

$$L(\boldsymbol{\theta}) = \sum_{t=1}^{t} \sum_{b \in B_t} \frac{(f(\boldsymbol{x}_b; \boldsymbol{\theta}) - r_b(t))^2}{2} + w\lambda \frac{\left\| \boldsymbol{\theta} - \boldsymbol{\theta^0} \right\|_2^2}{2} \tag{10}$$

where w is the network width, λ is the regularization parameter and $\boldsymbol{\theta^0}$ is the initial parameter of the neural network model. For $l \in [1, \cdots, L-1]$,

$$\boldsymbol{W}_l^0 = \begin{pmatrix} \boldsymbol{W} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W} \end{pmatrix} \tag{11}$$

where elements in $\boldsymbol{W}$ of size $\frac{w}{2} \times \frac{w}{2}$ are drawn from $N(0, \frac{4}{w})$. We also initialize

$$\boldsymbol{W}_L^0 = (\boldsymbol{w}^T, -\boldsymbol{w}^T) \tag{12}$$

where entries in $\boldsymbol{w}$ of size $\frac{w}{2}$ are drawn from $N(0, \frac{2}{w})$.

Following the neural contextual bandit in (Zhou, Li, and Gu 2020), the upper bound is calculated as

$$U = f(\boldsymbol{x}; \boldsymbol{\theta}) + \gamma \sqrt{\frac{\nabla f(\boldsymbol{x}; \boldsymbol{\theta})^T \boldsymbol{Z}^{-1} \nabla f(\boldsymbol{x}; \boldsymbol{\theta})}{w}} \tag{13}$$

where $\boldsymbol{Z}$ is iteratively updated by

$$\boldsymbol{Z} = \boldsymbol{Z} + \frac{\nabla f(\boldsymbol{x}; \boldsymbol{\theta})^T \nabla f(\boldsymbol{x}; \boldsymbol{\theta})}{w} \tag{14}$$

after each training and γ is the scaling factor.

After selecting m arms, the corresponding actual rewards are observed sequentially. The neural network will be trained whenever a new reward is observed, with its parameter $\boldsymbol{\theta}$ updated. The reward of each arm $r_{b_t,t}$ is calculated from the training loss using Eq 2.

Complexity Analysis

For each training epoch, executing line 6-12 in Alg. 1 takes $\mathcal{O}(c + m)$. The worst case of each training epoch is that the number of unexplored clusters $|C_{un}|$ is smaller than the required number of selected arms m. In this case, the system computes $f(\boldsymbol{x_{i,t}}; \boldsymbol{\theta_{t-1}})$ for n times. Each time computing takes $\mathcal{O}(w^3 L)$. Thus, executing line 13-20 in Alg. 1 takes $\mathcal{O}(w^3 L + m \log n)$. The training of deep neural network takes $\mathcal{O}(w^3 LET^2)$ where E is the epoch number set for training neural network in line 23 in Alg. 1. The factor T^2 is related to the number of dataset. Hence, the time complexity is $\mathcal{O}(w^3 mT^3 LE + w^3 nLT)$.

Regret Analysis

In this subsection, we analyze the regret for NCCB. NCCB is combined of two parts, exploration and exploitation. Thus we can divide regret regret $R(T)$ into two terms:

$$\mathbb{E}[R(T)] = \mathbb{E}[R_r(T)] + \mathbb{E}[R_i(T)] \tag{15}$$

where $\mathbb{E}[R_r(T)]$ is the regret in exploration phases and $\mathbb{E}[R_i(T)]$ is the regret in exploitation phases. We will prove that both two terms are upper bounded. We first discuss the submodularity of utility function which helps to present the analysis of regret in exploration phases.

Lemma 1 *The utility function defined in Eq.3 is submodular.*

All proofs can be found in the appendix. In this way, we can prove Theorem 1 which states that $\mathbb{E}[R_r(T)]$ is bounded by combining Lemma 1 and theorem 4.2 in (Nemhauser and Wolsey 1978).

Theorem 1 *There exists a real number ϕ such that the regret $\mathbb{E}[R_r(T)]$ in the exploration phase is bounded by*

$$\mathbb{E}[R_r(T)] \leq (1 - \frac{1}{e}) mcT\phi. \tag{16}$$

The next task is to analyze the regret in exploitation phases. Since the exploitation steps are modifications to NeuralUCB (Zhou, Li, and Gu 2020), we utilize lemma 6.3 and lemma 6.4 derived in (Zhou, Li, and Gu 2020) to validate Theorem 2 which provides the bound for $\mathbb{E}[R_i(T)]$.

Theorem 2 *Taking $\boldsymbol{r} = [r_{i,t}]_{i \in \{1,\cdots,t\}, b_j \in B_i} \in \mathbb{R}^{Tn}$, $S \geq \sqrt{2\boldsymbol{r}^T \boldsymbol{H}^{-1} \boldsymbol{r}}$ and $\tilde{d} = \frac{\log \det(\boldsymbol{I} + \frac{\boldsymbol{H}}{\lambda})}{\log(1 + \frac{Tn}{\lambda})}$ where $\boldsymbol{H}$ is defined in Definition 5.1 in (Zhou, Li, and Gu 2020), the regret in exploita-*

tion phases is bounded by

$$\mathbb{E}[R_i(T)] \leq 3m\sqrt{T}\sqrt{\tilde{d}\log(1+\frac{Tn}{\lambda})+2} \left[\upsilon\sqrt{\tilde{d}\log(1+\frac{Tn}{\lambda})+2+2\log T} +2\sqrt{\lambda}S+\beta(1-\frac{\lambda}{Tw}))^J\sqrt{\frac{T}{\lambda}}\right]+2m \quad (17)$$

where υ and β are constants.

Therefore, the regret of NCCB is upper bounded based on Theorem 1 and Theorem 2.

Evaluation

Experiment Setup

Dataset description We evaluate our scheme using Superuser dataset(Internet Archive 2021) and Yelp dataset(Yelp 2004). Superuser is an online forum where users can post and answer questions and comment others' questions and answers. The raw dataset includes information of posts, users, votes as well as comments. All the posts and comments data contain their corresponding user ids. Yelp is one of the biggest review websites where users post their reviews after visiting restaurants and other places. The dataset includes the information of users and reviews. All the reviews are stored along with their corresponding user ids. The raw superuser dataset consists of 194085 users' data and the yelp dataset consists of 2189457 users. The original dataset is highly skewed with most of users having a limited number of comments. Therefore, we remove the users with less than 150 comments or posts, and identify 761 valid users in Superuser dataset and 700 users in Yelp dataset to join federated learning. For both of the two dataset, we follow the method in (Hard et al. 2018) to remove stop words.

Federated learning setting We use the post bodies of different users to model the local dataset at different client devices naturally. These posts are used to perform the next-word-prediction task, a classical NLP task used in other federated literature. We reprocess the train dataset size for each client to be the same to exclude the influence of data size. We use the stacked LSTM model to perform word prediction as described in (Sundermeyer, Schlüter, and Ney 2012) with embedding size 200 and LSTM size 200. For Superuser dataset, we set number of LSTM layer to 2 while for Yelp dataset the number of layer is increased to 5, since the latter dataset is bigger. Learning rate and learning rate per local are set as $\gamma = 0.002$ and $\gamma_d = 0.9993$. Local epoch is set to $E = 5$ and the number of clients participated in training each round is set to $N = 5$. The training is done with a 24G-memory NVIDIA RTX3090.

Comparison approaches We compare our approach with three other approaches: random, Oort (Lai et al. 2021), k-LinUCB (Li et al. 2010). Random selection, is the default selection approach proposed in FedAvg, where clients are randomly selected to join the federated learning. Oort is a state-of-the-art stochastic bandit solution for client selection in federated learning(Lai et al. 2021). k-LinUCB is another representative contextual combinatorial bandit solution. It assumes a linear relationship between reward and feature vector, and selects k clients with top k expected rewards. The observed rewards are then used to renew parameters in the linear function through regression. In K-LinUCB and Oort setting, the reward is defined the same as in Eq. 2.

Evaluation metrics We select Top-1 accuracy and speed to reach target accuracy as evaluation metrics. The Top-1 accuracy refers to the conventional accuracy where the predicted word must be the expected answer. We also evaluate the convergence speed for the training process, which is represented by the number of rounds to reach the target accuracy. The target accuracy is the maximum accuracy that the model obtains under the random selection method.

Table 1: Summary of terminal accuracy and relative improvement

Dataset	Method	Accuracy (%)	Improvement (%)
Superuser	Rand	10.035	0.0
	NCCB	11.784	17.429
	Oort	11.372	13.323
	k-LinUCB	11.122	10.832
	FedAVG(IID)		
Yelp	Rand	11.176	0.0
	NCCB	12.547	12.267
	Oort	12.116	8.410
	k-LinUCB	11.810	5.673
	FedAVG(IID)	12.800	14.531

Table 2: Summary of number of rounds to target accuracy and relative improvement

Dataset	Method	Rounds to Target Accuracy	Improvement (%)
Superuser	Rand	283	0.0
	NCCB	134	52.7
	Oort	181	36.0
	k-LinUCB	213	24.7
Dataset	Method	Rounds to Target Accuracy	Improvement (%)
Yelp	Rand	196	0.0
	NCCB	101	48.5
	Oort	109	44.4
	k-LinUCB	108	44.9
	FedAVG(IID)	98	50.0

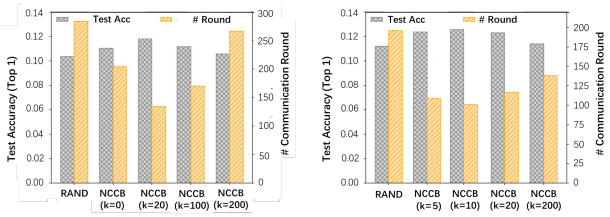
Overall Performance

We summarize the performance of the terminal accuracy and the rounds to target accuracy in Table 1 and Table 2, respectively. Compared with the baseline performance, our NCCB approach improves the accuracy by 17.429% compared with the baseline for Superuser dataset and by 12.267% for Yelp dataset. In addition, we find that K-LinUCB does not perform even as good as the classical stochastic bandit approach, revealing the non-trivial relationship between the context features and the rewards, and the consequences of wrong assumptions.

Sensitivity Analysis

Impact of exploring threshold The exploring threshold k is one important parameter in our method. It affects how

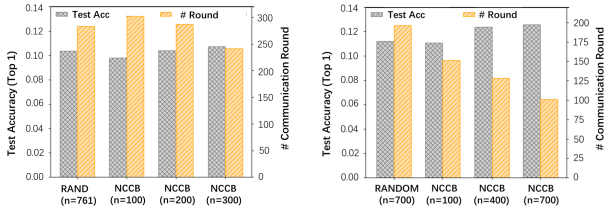
we measure one cluster group as unexplored or not. We set k to be 0, 20, 100, 200 for the Superuser dataset, and 10, 15, 20, 200 for the Yelp dataset. Though NCCB achieves higher performances in all k settings than the widely applied random approach, different k leads to different performances for NCCB. For example, for the Yelp dataset, when we change k from 10 to 5, 20 and 200, the final accuracy decrease by 2.12%, 2.27% and 10.83% respectively. Essentially, k is a measurement of balance condition between exploration and exploitation. If k is very small (e.g. $k = 0$ in Fig. 3(a), $k = 5$ in Fig. 3(b)), it fails to explore enough clients so that the neural reward estimator performs badly. On the other hand, if k is quite large (e.g. $k = 200$ in Fig. 3), the system neural reward estimator is idle during the whole process and fails to exploit previous experience.



(a) Superuser (b) Yelp

Figure 3: Impact of different exploring thresholds

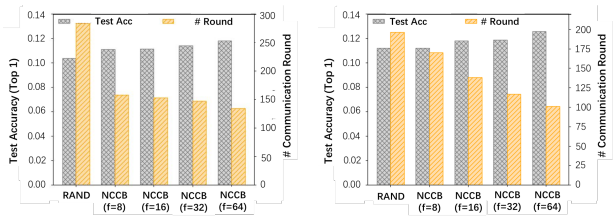
Impact of scale of total participants We explore the impact of total participated clients in training and further compare them with the random approach in the original setting (761/700 training clients). As we can see, with the increase of number of participants, the training performance increases, indicating that NCCB method can capture clients with larger utility and improve the training performance. Further more, compared to the baseline with larger scale, our method can identify desirable clients from a smaller pool of clients and achieve even better performances.



(a) Superuser (b) Yelp

Figure 4: Impact of different system scales

Impact of feature size The feature size of the SimHash vector determines the expressiveness on clients' data. We evaluate the performance of NCCB under different feature sizes generated by SimHash. Figure 5 shows that the decrease of feature size degrades the performance of federated learning. For the superuser dataset, decreasing size of the feature vectors from 64 to 32, 16, 8 would decrease the final accuracy by 3.42 %, 5.49% and 12.16% respectively. For the yelp dataset, when we decrease the size of feature vectors in a similar manner. The longer the bit length used to encode the local data, the more specific this feature captures the characteristics of this client, which helps the client selection process.



(a) Superuser (b) Yelp

Figure 5: Impact of different feature sizes

Related Works

Client selection in federated learning Client selection is an important controlling knob in federated learning to handle various types of heterogeneity. From the solutions' perspective, existing client selection studies generally fall into three categories: bandit-based (Xia et al. 2020; Lai et al. 2021), reinforcement learning based(Wang et al. 2020), heuristic based(Chen et al. 2020; Ouyang et al. 2021). Among bandit-based approaches, Oort in (Lai et al. 2021) formulates the client selection problem as a traditional multi-armed bandit problem without relying on contextual information. Consequently, it relies on training of all clients to get the quality estimation of local dataset, leading to prolonged convergence time. CS-UCB proposed in (Xia et al. 2020) relies on corresponding training time as the contextual feature to conduct UCB, which also requires redundant training rounds. In addition to bandit-based approaches, FAVOR in (Wang et al. 2020) applies a reinforcement learning approach to select device. However, deep reinforcement learning takes much longer time to train without any theoretical performance guarantees. Researchers also heuristically cluster clients based on the model weights after first-round training of each client and randomly select clients without exploiting previous experience (Chen et al. 2020; Ouyang et al. 2021).

Bandit algorithms Our work also contributes to the bandit field by jointly considering the non-trivial contextual relationship and the combinatorial requirement. Classical contextual bandit algorithms such as Linear Upper-Confidence-Bound (UCB) algorithm (Abe, Biermann, and Long 2003) assumes a trivial relationship between the explored rewards and feature vectors, which is hard to meet in real life (Zhou 2015). Neural contextual bandit is introduced to overcome this shortage by relying on a neural network to learn the underlying context-to-reward relationship (Zhou, Li, and Gu 2020). However, it fails to satisfy the combinatorial requirement specific in our problem and thus cannot be applied directly. Contextual combinatorial MAB (Chen, Xu, and Lu 2018) combines both contextual and combinatorial bandit. Nevertheless, it provides no method to estimate reward from

feature vectors. In this paper, we propose to learn the non-trivial relationship between rewards and contexts via a neural network and satisfies the combinatorial requirements required in federated learning. The performance of NCCB is proved both theoretically and empirically.

Conclusion

Federated learning emerges as a promising technique for neural network training without sharing the raw data. Severe data heterogeneity in decentralized clients greatly harms the performance of federated learning. In this paper, we proposed a context-aware client selection approach for communication-efficient federated learning by leveraging the correlation between clients. To be specific, we propose a novel neural contextual combinatorial bandit approach to intelligently select clients to minimize the selection regret. Our approach successfully embeds the context into the selection process, handles the non-trivial relationship between the feature and reward, and works well in the intrinsic combinatorial searching space. Extensive experiments on two real-world datasets demonstrate its effectiveness.

Proof of Lemma 1

Proof 1 *Suppose there is a set of arms B. We have*

$$U(B \cup \{a\}) = \sqrt{\sum_{i \in B \cup \{a\}} U(i)^2} \tag{18}$$

where a is an arm such that $a \notin B$. Since

$$\begin{aligned} &U(B)^2 \leq U(B \cup \{a\})^2 \\ =&\sum_{i \in B \cup \{a\}} U(i)^2 \leq \sum_{i \in B} U(i)^2 + U^2(a) \\ \leq&(U(B) + U(a))^2, \end{aligned} \tag{19}$$

we can obtain

$$U(B) \leq U(B \cup \{a\}) \leq U(B) + U(a). \tag{20}$$

Hence, this is a submodular function.

Proof of Theorem 1

Proof 2 *From Alg. 1, at each exploration phase, at most m arms are selected. Based on theorem 4.2 in (Nemhauser and Wolsey 1978), the regret of selecting a random combination of arms at round t is bounded as shown in*

$$\mathbb{E}[R_r(t)] \leq (1 - \frac{1}{e}) mc\phi \tag{21}$$

where ϕ is maximum reward of arms due to submodularity of the reward function for combination of arms. The exploration phases at most last for T rounds and then we get

$$\mathbb{E}[R_r(T)] \leq (1 - \frac{1}{e}) mcT\phi. \tag{22}$$

Proof of Theorem 2

Proof 3 *We consider the case when $k = 0$. The bandit will undergo exploitation phases for T rounds and each round select m arms.*

Let $r(\boldsymbol{x}^*, t)$ be the reward of the optimal arm and $r(\boldsymbol{x}, t)$ be the estimated reward of selected arm. Lemma 6.3 in (Zhou, Li, and Gu 2020) still holds in NCCB since we select the maximum m upper confidence bound values evaluated with $\boldsymbol{\theta_t}$ which is larger than or equal to the values of arms in optimal combinations. In this way, there exists positive constants α and β such that for any $\delta \in (0, 1)$ if $\dim \boldsymbol{x} \geq \beta \max\{T^7 \lambda^{-7} w^{21} (\log(\dim \boldsymbol{x})^3),\ \lambda^{-\frac{1}{2}} w^{-\frac{3}{2}} (\log(\frac{Tnw^2}{\delta}))^{\frac{3}{2}}\}$, with probability at least $1 - \delta$ based on the initialization of $\boldsymbol{\theta_0}$, the difference between ideal and estimated reward is

$$\begin{aligned} r(\boldsymbol{x}^*, t) - r(\boldsymbol{x}, t) \leq& 2\gamma_{t-1} \min\{\| \frac{\nabla f(\boldsymbol{x}; \boldsymbol{\theta_{t-1}})}{\sqrt{\dim \boldsymbol{x}}} \|, 1\} \\ &+ \alpha (S \dim(\boldsymbol{x}^{-\frac{1}{6}}) \sqrt{\log \dim \boldsymbol{x}} T^{\frac{7}{6}} \lambda^{-\frac{1}{6}} w^{\frac{7}{2}} \\ &+ (\dim \boldsymbol{x})^{-\frac{1}{6}} \sqrt{\log \dim \boldsymbol{x}} T^{\frac{5}{3}} \lambda^{-\frac{2}{3}} w^3). \end{aligned} \tag{23}$$

Hence,

$$\begin{aligned} R_i(T) \leq& \sum_{i=1}^{T} m[r(\boldsymbol{x}^*, t) - r(\boldsymbol{x}, t)] \\ \leq& 2m \sum_{i=1}^{T} \gamma_{t-1} \min\{\| \frac{\nabla f(\boldsymbol{x}; \boldsymbol{\theta_{t-1}})}{\sqrt{\dim \boldsymbol{x}}} \|, 1\} \\ &+ \alpha m (S (\dim \boldsymbol{x})^{-\frac{1}{6}} \sqrt{\log \dim \boldsymbol{x}} T^{\frac{13}{6}} \lambda^{-\frac{1}{6}} w^{\frac{7}{2}} \\ &+ (\dim \boldsymbol{x})^{-\frac{1}{6}} \sqrt{\log \dim \boldsymbol{x}} T^{\frac{8}{3}} \lambda^{-\frac{2}{3}} w^3) \end{aligned} \tag{24}$$

Using Cauchy-Scharwz inequality we can obtain

$$\begin{aligned} R_i(T) \leq& 2m \sqrt{T \sum_{i=1}^{T} \gamma_{t-1}^2 \min\{\| \frac{\nabla f(\boldsymbol{x}; \boldsymbol{\theta_{t-1}})}{\sqrt{\dim \boldsymbol{x}}} \|, 1\}} \\ &+ \alpha m (S (\dim \boldsymbol{x})^{-\frac{1}{6}} \sqrt{\log \dim \boldsymbol{x}} T^{\frac{13}{6}} \lambda^{-\frac{1}{6}} w^{\frac{7}{2}} \\ &+ (\dim \boldsymbol{x})^{-\frac{1}{6}} \sqrt{\log \dim \boldsymbol{x}} T^{\frac{8}{3}} \lambda^{-\frac{2}{3}} w^3). \end{aligned} \tag{25}$$

The first inequality holds due to submodularity of combination reward function and the second ineqaulity holds due to Eq. 24. Using Lemma 6.4 in (Zhou, Li, and Gu 2020), the inequality becomes

$$\begin{aligned} R_i(T) \leq& 3m\sqrt{T} \sqrt{\tilde{d} \log(1 + \frac{Tn}{\lambda}) + 2} \\ &\left[\upsilon \sqrt{\tilde{d} \log(1 + \frac{Tn}{\lambda}) + 2 - 2\log\delta} \right. \\ &\left. + 2\sqrt{\lambda} S + \beta (1 - \frac{\lambda}{Tw}))^J \sqrt{\frac{T}{\lambda}} \right] + m \end{aligned} \tag{26}$$

Taking a high probability with $\delta = \frac{1}{T}$, we obtain the bound

upon expected regret

$$\mathbb{E}[R_i(T)] \leq 3m\sqrt{T}\sqrt{\tilde{d}\log(1+\frac{Tn}{\lambda})+2} \left[\upsilon\sqrt{\tilde{d}\log(1+\frac{Tn}{\lambda})+2+2\log T} + 2\sqrt{\lambda}S + \beta(1-\frac{\lambda}{Tw}))^J\sqrt{\frac{T}{\lambda}}\right] + 2m. \quad (27)$$

References

Abe, N.; Biermann, A. W.; and Long, P. M. 2003. Reinforcement learning with immediate rewards and linear hypotheses. *Algorithmica*, 37(4): 263–293.

Aiello, L. M.; Barrat, A.; Schifanella, R.; Cattuto, C.; Markines, B.; and Menczer, F. 2012. Friendship prediction and homophily in social media. *ACM Trans. Web*, 6(2): 1–33.

Bukaty, P. 2019. *The California Consumer Privacy Act (CCPA): An implementation guide*. IT Governance Publishing. ISBN 9781787781320.

Chen, C.; Chen, Z.; Zhou, Y.; and Kailkhura, B. 2020. FedCluster: Boosting the Convergence of Federated Learning via Cluster-Cycling. In *Proc. IEEE Int. Conf. Big Data*, 5017–5026.

Chen, L.; Xu, J.; and Lu, Z. 2018. Contextual combinatorial multi-armed bandits with volatile arms and submodular reward. In *Proc. Adv. Neural Inf. Process. Syst.*, volume 31, 3247–3256.

European Commission. 2018. 2018 reform of EU data protection rules. https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes_en.pdf. Accessed: 2020-02-06.

Gionis, A. 1999. Similarity Search in High Dimensions via Hashing. In *Proc. Int. Conf. Very Large Data Bases*.

Google. 2018. Evaluation of cohort algorithms for the FLoC API. https://github.com/google/ads-privacy/blob/master/proposals/FLoC/FLOC-Whitepaper-Google.pdf. Accessed: 2020-02-06.

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

Huang, L.; Shea, A. L.; Qian, H.; Masurkar, A.; Deng, H.; and Liu, D. 2019. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *J. Biomed. Inform.*, 99: 103291.

Internet Archive. 2021. https://archive.org/details/stackexchange. Accessed: 2021-02-06.

Jafari, O.; Maurya, P.; Nagarkar, P.; Islam, K. M.; and Crushev, C. 2021. A Survey on Locality Sensitive Hashing Algorithms and their Applications. *arXiv preprint arXiv:2102.08942*.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2019. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*.

Lai, F.; Zhu, X.; Madhyastha, H. V.; and Chowdhury, M. 2021. Oort: Efficient federated learning via guided participant selection. In *Proc. USENIX OSDI*, 19–35.

Li, L.; Chu, W.; Langford, J.; and Schapire, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proc. ACM WWW*, 661–670.

Liu, W.; Chen, L.; Chen, Y.; and Zhang, W. 2020a. Accelerating federated learning via momentum gradient descent. *IEEE Trans. Parallel Distrib. Syst.*, 31(8): 1754–1766.

Liu, Y.; Huang, A.; Luo, Y.; Huang, H.; Liu, Y.; Chen, Y.; Feng, L.; Chen, T.; Yu, H.; and Yang, Q. 2020b. Fedvision: An online visual object detection platform powered by federated learning. In *Proc. AAAI Conf. Artif. Intell.*, volume 34, 13172–13179.

Manku, G. S.; Jain, A.; and Sarma, A. D. 2007. Detecting Near-Duplicates for Web Crawling. In *Proc. ACM WWW*, 141–150.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1): 415–444.

Nemhauser, G. L.; and Wolsey, L. A. 1978. Best algorithms for approximating the maximum of a submodular set function. *Math. Operations Res.*, 3(3): 177–188.

Ouyang, X.; Xie, Z.; Zhou, J.; Huang, J.; and Xing, G. 2021. ClusterFL: a similarity-aware federated learning system for human activity recognition. In *Proc. ACM MobiSys*, 54–66.

Sundermeyer, M.; Schlüter, R.; and Ney, H. 2012. LSTM neural networks for language modeling. In *Proc. Interspeech*.

Uddin, M. P.; Xiang, Y.; Lu, X.; Yearwood, J.; and Gao, L. 2020. Mutual Information Driven Federated Learning. *IEEE Trans. Parallel Distrib. Syst.*, 32(7): 1526–1538.

Wang, H.; Kaplan, Z.; Niu, D.; and Li, B. 2020. Optimizing federated learning on non-IID data with reinforcement learning. In *Proc. IEEE Conf. Comput. Commun.*, 1698–1707.

Wang, S.; Tuor, T.; Salonidis, T.; Leung, K. K.; Makaya, C.; He, T.; and Chan, K. 2019. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commu.*, 37(6): 1205–1221.

Xia, W.; Quek, T. Q. S.; Guo, K.; Wen, W.; Yang, H. H.; and Zhu, H. 2020. Multi-Armed Bandit-Based Client Scheduling for Federated Learning. *IEEE Trans. Wireless Commun.*, 19(11): 7108–7123.

Yang, T.; Andrew, G.; Eichner, H.; Sun, H.; Li, W.; Kong, N.; Ramage, D.; and Beaufays, F. 2018. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*.

Yelp. 2004. Yelp open dataset. https://www.yelp.com/dataset. Accessed: 2020-02-06.

Yu, H.; Yang, S.; and Zhu, S. 2019. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proc. AAAI Conf. Artif. Intell.*, volume 33, 5693–5700.

Zhao, Z.; Gao, M.; Luo, F.; Zhang, Y.; and Xiong, Q. 2020. LSHWE: improving similarity-based word embedding with locality sensitive hashing for cyberbullying detection. In *Int. Joint Conf. Neural Netw.*, 1–8.

Zhou, D.; Li, L.; and Gu, Q. 2020. Neural contextual bandits with UCB-based exploration. In *Proc. Int. Conf. Mach. Learn.*, volume 119, 11492–11502.

Zhou, J. T.; Zhang, L.; Jiawei, D.; Peng, X.; Fang, Z.; Xiao, Z.; and Zhu, H. 2021. Locality-Aware Crowd Counting. *IEEE Trans. Pattern Anal. Mach. Intell.*

Zhou, L. 2015. A survey on contextual multi-armed bandits. *arXiv preprint arXiv:1508.03326*.