Complex & Intelligent Systems (2023) 9:115–131
https://doi.org/10.1007/s40747-022-00729-2

ORIGINAL ARTICLE

Intelligent depression detection with asynchronous federated optimization

Jinli Li[1] · Ming Jiang[1] · Yunbai Qin[1] · Ran Zhang[2] · Sai Ho Ling[3]

Received: 6 July 2021 / Accepted: 27 March 2022 / Published online: 23 June 2022

Abstract
The growth of population and the various intensive life pressures everyday deepen the competitions among people. Tens of millions of people each year suffer from depression and only a fraction receives adequate treatment. The development of social networks such as Facebook, Twitter, Weibo, and QQ provides more convenient communication and provides a new emotional vent window. People communicate with their friends, sharing their opinions, and shooting videos to reflect their feelings. It provides an opportunity to detect depression in social networks. Although depression detection using social networks has reflected the established connectivity across users, fewer researchers consider the data security and privacy-preserving schemes. Therefore, we advocate the federated learning technique as an efficient and scalable method, where it enables the handling of a massive number of edge devices in parallel. In this study, we conduct the depression analysis on the basis of an online microblog called Weibo. A novel algorithm termed as CNN Asynchronous Federated optimization (CAFed) is proposed based on federated learning to improve the communication cost and convergence rate. It is shown that our proposed method can effectively protect users' privacy under the premise of ensuring the accuracy of prediction. The proposed method converges faster than the Federated Averaging (FedAvg) for non-convex problems. Federated learning techniques can identify quality solutions of mental health problems among Weibo users.

Keywords Depression detection · Asynchronous federated learning · CNN

Introduction

Depression is one primary public healthcare issue, as one of the leading causes of mental disability, it has gradually become a major contributor to psychological diseases globally, especially during the COVID-19 pandemics. Currently, the population with depression in 2019 was estimated to be 0.35 billion, unfortunately, less than 10% of people with depression treated [1]. It has the tendency to become the world's second-largest disease by 2021, and so far, suicide victims are as high as one million every year. In China, depressed patients have been as high as 90 million [2]. In particular, detecting depression in the early stages, so that how to ensure the early stage correct detection of the patients to get hospital treatment becomes a main research direction in the field. Up to the end of 2019, the monthly active users of Weibo reached 516 million, a net increase of approximately 54 million compared to 2018, of which mobile accounts for 94% of total and daily active users reached 222 million [3]. A large number of Weibo users and rich data provide favorable value for the recognition of certain groups. As a social networking tool in China, Weibo is one of the most popular personal and media publishing platforms. Therefore, this study uses data on Weibo to analyzes the feasibility of detecting depression.

✉ Ming Jiang
Jiangvic2021@163.com

Jinli Li
2303610802@qq.com

Yunbai Qin
qinyunbai@mailbox.gxnu.edu.cn

Ran Zhang
zra@deakin.edu.au

Sai Ho Ling
Steve.Ling@uts.edu.au

1 College of Electronic Engineering, Guangxi Normal University, Guilin, China

2 Business and Law School, Deakin University, Geelong, Australia

3 Faculty of Engineering and IT, University of Technology, Sydney, Australia

The most widely used methods of depression detection are based on the standard manuals ICD-10 and DSM-5 and the doctor's clinical experience. The standard manuals are lacking precision or that there is room for incorrect personal diagnosis. The various machine learning techniques are employed for depression detection, there are still some challenges.

- Data are labeled by a single social network is limited, and the training data has a significant influence on the training effect of the machine learning model, so the amount of data needs to be expanded while ensuring data security.
- The increasing number of depressed people puts forward higher requirements on the accuracy of the model.
- With the development of big data, privacy-preserving has attracted attention from the public. How to conduct model training on massive data is an important issue to be solved under the conditions of improving data security and ensure the efficiency of the model.

Federated learning is a newly boomed machine learning framework that can effectively help multiple institutions build a global model under user privacy preservation requirements [4]. In the context of above-mentioned challenges, we aim to develop a federated learning scheme to detect depression. The main contributions in this paper are as follows:

- We propose a new asynchronous federated optimization algorithm with provable convergence for non-convex problems under Weibo users' data.
- We show that our proposed method can effectively protect users' privacy under the premise of ensuring the accuracy of prediction.
- We use 900 users to train the model together, which improves the utilization of resources and the performance of the model.
- The proposed algorithm can reduce communication overhead.

The rest of this paper is organized as follows. Related literature for current work is presented in "Related work". In "Methods", the details of the proposed method are described. "Experiments" evaluates the performance of CAFed in terms of accuracy and convergence for different frameworks, followed by a discussion in "Discussion". "Conclusion and future work", concludes the paper and points out potential future research directions.

Related work

Machine learning

Using social networks for depression detection has become one of the influential approaches [5], however, designing a useful feature sometimes is not straightforward, especially in the presence of a large amount of data. De Choudhury et al. [6] have measured behavioral attributes relating to social engagement, emotions, linguistic styles and network structure. They leverage these behavioral attributes to build a SVM classifier that provides estimates of the risk of depression. However, this method employs crowdsourcing to compile a set of Twitter users, the effort and time spent are not taken into account. Islam et al. [7] propose three types of properties (emotional, temporal, linguistic style), they then apply machine learning approaches such as Decision Tree, KNN, SVM and Ensemble classifier to study each type independently. Their results show that the Decision tree gives a better outcome. Moreover, they focus on analyzing the time patterns of Facebook users. Unfortunately, this study does not identify who the sufferers are and only detects the Facebook comments for depression detection.

Mcmahan et al. [8] have applied natural language processing (NLP) techniques to develop a depression algorithm for the Thai language on Facebook. They extract from the Facebook user behavior features, including a number of posts, interaction with others, privacy settings, and daytime posting. Considering the limited personal information that can be collected, the behavioral attributes obtained by the algorithm do not cover all relevant factors.

Al-Mosaiwi et al. [9] have proposed a novel model based on user-generated content. They use SVM and Naive Bayes to create a model. The most important is that this model can classify the patients into one out of four levels (Minimal, Mild, Moderate, or Severe depression). Different users may have different behaviour on two social networks, and different regions have different linguistic style, it leads to the accuracy is lower than other methods.

Most of the recognition models either act as black boxes or unsupported incremental learning. It is difficult for AI to classify incremental sequence data. Burdisso et al. [10] have shown a novel white-box text classifier to detect depression. They not only build a public dataset but design a new evaluation metric.

Deep learning has made great significant achievements in image and speech recognition. In traditional text classification, if the text is transformed into a word vector, it will often cause the curse of dimensionality. There are some methods to compress dimensions, these methods do not consider context information in the training process of word vectors and are not satisfactory. Therefore, some researchers apply deep learning to solve large-scale text representation, and then use network structures such as CNN or RNN to automatically obtain feature representation [10–12].

Some researchers [11] have studied the convolution kernels of different sizes to extract multiple features in CNN, which generates the penultimate layer and pass into the fully connected layer. Finally, the probability of each class is predicted by the softmax function. The researchers firstly apply

CNN for text classification, the proposed system is composed of four parts: input layer, convolution layer, pooling layer and full connection layer. CNN has obtained better results, although the performance may be slightly worse than RNN on some data sets, the training efficiency of CNN is higher. The model performance is very sensitive to the parameter in neural networks, the parameters adjustment of the neural network is further illustrated in [10].

Wu et al. [13] used recurrent neural networks to compute the posts representation of each user's posts representation. They combine the word representations with other content-based, behaviour and living environment features to build deep neural networks. Trotzek et al. [14] have utilized different word embeddings for training the convolutional neural network and comparing it to a classification based on user-level linguistic metadata. Their experimental results show that an ensemble of both approaches achieves a good performance. Ding et al. [12] presented text-level mining of Sina Weibo data from college students, they use a deep neural network to extract features and propose a deep integrated support vector machine (DISVM) algorithm to detect depression. This approach makes the recognition model more stable and improves accuracy.

Natural language processing

More and more people like social networks such as Weibo, and post, photos and comments on various information on Weibo. It is possible to analyze and detect whether the users have depression tendency from the text information [10]. To translate the information into input vectors available to the classifier, there are two methods to represent words such as one-hot vector representation and distributed representation. In distributed representation, the words are represented in low dimension. Unlike one-hot vector representation, similar features can have similar vectors in distributed representation, and its computational speed is faster than one-hot vector representation. We use a distributed representation of the words in this work. There are many ways to obtain the distributed representation of the words, we use the pre-trained distributional word embeddings of word2vec [15]. There are two methods to train word embeddings in word2vec: continuous bag of words (CBOWs) and skip-gram. The context in CBOWs predicts target words. In contrast, context is predicted by the target words in skip-gram. We use Weibo corpus Weibo word embeddings which are trained with 300 dimensions [16].

Federated learning

Online social networks provide more convenient communication among people, but it also exposes sensitive information about users, the third parties may leak such information.

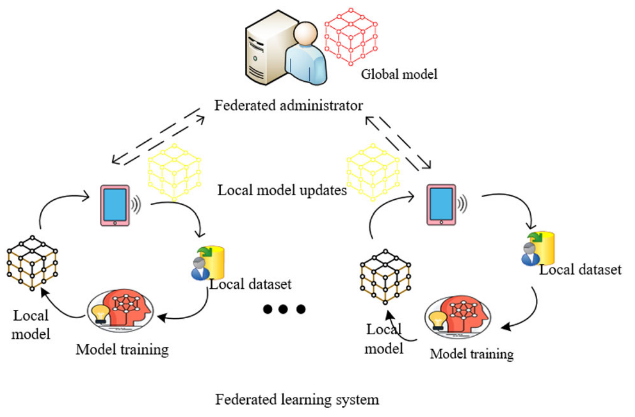


Fig. 1 Architecture for Federated learning system: ①: sending encrypted gradients/model to federated administrator. ②: secure aggregation. ③: sending global model to workers. ④: local model updates

In this paper, we have considered data security and privacy-preserving. Federated learning is a distributed machine learning framework that can effectively solve the problem of data islands, and allow participants to achieve AI collaboration based on not sharing data [4]. Figure 1 shows the federated learning framework, it is composed of server and workers, whose system is similar to parameter servers, but on-device federated learning training tasks are allowed to be executed only when the device is idle, charging, and connected to unmetered networks [17]. Thus, compared to typical distributed machine learning, communication in federated learning needs less costs. Federated learning should handle training data with the following characteristics:

- Non-IID: there are different data distributions on each device [18], i.e., the overall distribution cannot be learned from data on a single device.
- Imbalanced data: Data can be biased to certain labels [17], e.g., users may have different habits or edge devices are monitoring different locations.
- Heterogeneity: Data size and device performance may vary on different local devices [19].

Previous federated learning algorithms such as federated averaging [8] can only train hundreds of devices, but there are many devices in our daily lives. Too many devices training at the same time would congest the network. What's more, different devices have different computational capacity and battery time, and it is difficult for selected devices to synchronize training [18]. Finally, not all devices participate in training tasks, it only selects the subset of available devices. If the number of survived devices is too small, the server must drop the entire epoch.

Xie et al. [20] proposed a novel asynchronous federated optimization algorithm. They apply a mixing hyperparameter to control the tradeoff between the convergence rate and variance reduction and update the global model by weight averaging. However, it is hard to find a suitable mixing hyper-parameter. In addition, FedAvg cannot handle incrementally increased data on devices and data/system heterogeneity that leads to stragglers or dropouts. Therefore,Chen et al. [21] proposed an asynchronous online federated learning framework, in which the local model performances online learning with the continuous streaming local data. What's more,the framework updates the global model in an asynchronous manner to solve the challenges related to different computational loads at heterogeneous edge devices.

There are no well-known studies that have combined federated learning with depression analysis. To overcome the above weakness, we aim to detect depression from Weibo posts. A novel asynchronous federated learning algorithm is proposed to prevent user privacy leaked.

In this paper, we describe in detail the asynchronous federated optimization algorithm based on CNN [22]. Compared with machine learning, the distributed detection model under the federated learning mechanism has dramatically improved the convergence speed and data security, while also ensuring the accuracy of classification.

There are usually words in the collected posts that are not meaningful to our research, such as "this", "that", "there", "oh". So, we processed the collected posts by using the Harbin Institute of Technology's deactivation vocabulary [23] to remove these meaningless words.

The effect of the word vector is related to the size of the corpus, and the corpus of processing task is not enough to support our experiment, we applied a large—scale microblog word vector trained by word2vec model that was published on the Internet [16, 24]. To ensure the reliability of the training results, the features must be normalized to the range of [0, 1]. The advantage of data normalization is that it can eliminate the differences among different dimensions and make data of different dimensions comparable after normalization (Table 1).

Table 1 Notations and terminologies

Notation/tern	Description
n	Number of devices
T	Number of global epochs
H_τ^i	Number of local iterations in the τ^{th} epoch on the ith devices
w_t	Global model in the t^{th} epoch on sever
w_t^k	The k^{th} entry of w_t
$w_{\tau,h}^i$	Model initialized from w_τ, updated in the hth local iteration, on the ith device
w_{back}^i	the model before w_{new}^i is updated, on the ith devices
c	A hyper-parameter
r	A random number
ϑ	A small constant
β	A magnitude coefficient
D^i	Dataset on the ith device
$z_{t,h}^i$	Data(minibatch) sampled from D^i
γ	Learning rate
Server	The place where the training data are placed
Worker	One worker on each device, process that trains the model

Differential privacy

We develop the asynchronous federated learning to protect user privacy leakage, the parameters generated in the algorithm will expose intermediate plaintext contents. Once the model parameter is leaked, some important information may be breached [25, 26].

To solve the problem, researchers proposed the concept of differential privacy (DP) [27]. Differential privacy constitutes a strong standard for privacy protection of algorithms on aggregated databases, it does not require special attack assumptions, nor does it pay attention to the background knowledge of the attacker, so it can ensure that the model does not expose whether a data point is used during training. In the federated averaging scheme, the local models are averaged by the server after each epoch of communications, some researchers apply the randomized mechanism to change the global values [28]. The contribution made by each client is hidden in the model aggregation.

Definition 1 (*A random mechanism* M):$D \rightarrow R$, with domain and range R satisfier (ϵ, δ)- differential privacy, if for any two adjacent inputs d, $d^{'} \in D$ and for any subset of outputs $S \subseteq R$ it holds that $P[M(d) \in S] \leq e^{\epsilon}\Pr\left[M\left(d^{'}\right) \in S\right] + \delta$. In this definition, δ accounts for the probability that plain ϵ — differential privacy is broken [28].

Methods

Initial model

The initial model is one of the important components in the framework of the CAFed depression detection system. Our model is inspired from [29]. The layers present in our CNN architecture are: embedding layer, convolution layer, pooling layer, dropout layer, fully connected layer, and output layer.

The Weibo vector is formed by concatenating all posts of the user. If the dimension of the word vector is d and the length of the nth post of a user u_i is $l_{i,n}$ then the dimension of the word vector matrix is $(l_{i,1} \cup l_{i,2} \cdots l_{i,n}) \times d$. This word vector matrix is input to the first layer of CNN as shown in Fig. 2. Supposing the dimension of the word vector is $\boldsymbol{d}$ and the length of the $\boldsymbol{n}$th post of user $\boldsymbol{u_i}$ is $\boldsymbol{l_{i,n}}$ then the dimension of word vector matrix is $(\boldsymbol{l_{i,1}} \cup \boldsymbol{l_{i,2}} \ldots \boldsymbol{l_{i,n}}) \times \boldsymbol{d}$.

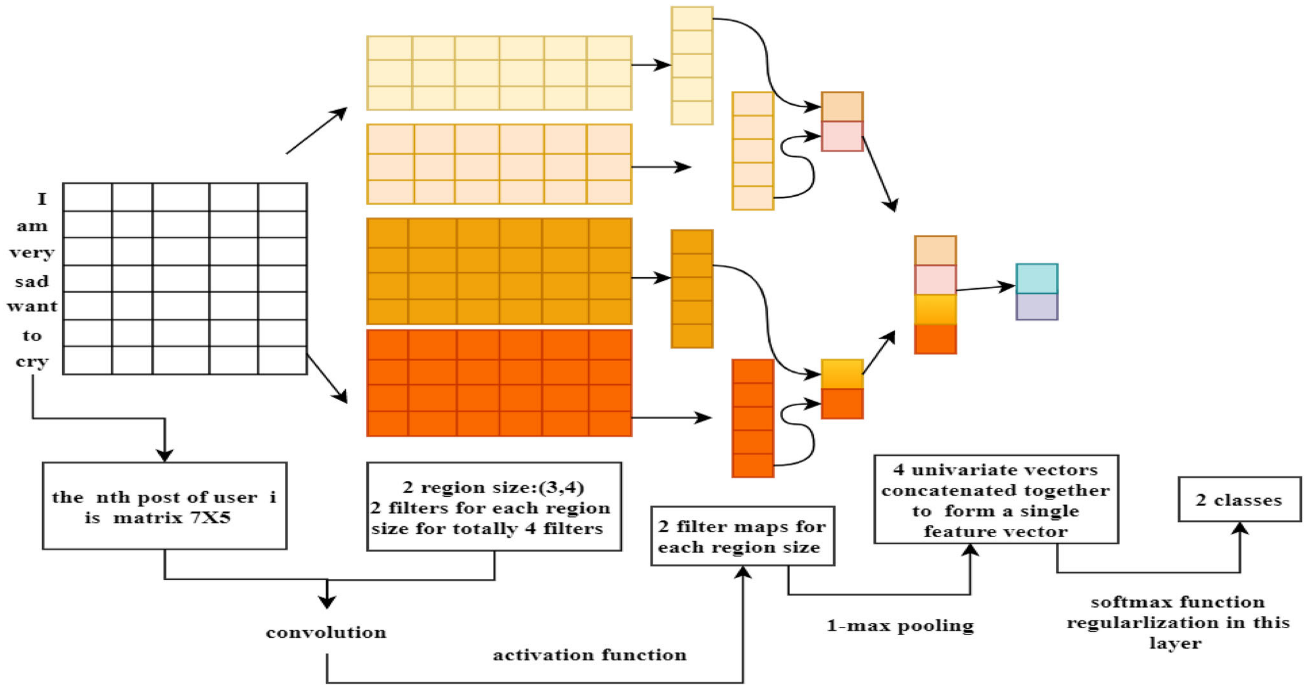

I
am
very
sad
want
to
cry
the nth post of user i is matrix 7X5
2 region size:(3,4) 2 filters for each region size for totally 4 filters
2 filter maps for each region size
4 univariate vectors concatenated together to form a single feature vector
2 classes
convolution
activation function
1-max pooling
softmax function regularlization in this layer

Fig. 2 Structure of the text-CNN

$\text{term}_{l_{i,n}}^{j}$ is denoted as the jth word in the nth post of a user u_i. We present the nth post using $l_{i,n}$ words:

<$\text{term}_{l_{i,n}}^{1}$, $\text{term}_{l_{i,n}}^{2}$, $\ldots \text{term}_{l_{i,n}}^{l_{i,n}}$>. l is comprised of the sequence of words: $<\text{term}_{l_{i,1}}, \text{term}_{l_{i,2}}, \ldots \text{term}_{l_{i,n}}>$. Then, the word vector is represented as

$$W_v = w_{l_{i,1}} \circ w_{l_{i,2}} \circ w_{l_{i,3}} \ldots \circ w_{l_{i,n}} \qquad (1)$$

where $\text{term}_{l_{i,n}}$ is the words of nth post of the user u_i,$w_{l_{i,n}}$ represents the word embedding vector of $\text{term}_{l_{i,n}}$, and the symbol "$\circ$" is the concatenation operator. Next layer is the convolution layer. In this work, rectified linear units (ReLU) are used to get the convolution feature maps, and the filter length can be 3, 4, or 5. We apply maximum pooling to extract the most important feature and then concatenate values in all feature maps to obtain the pooling layer's final feature vector. To avoid overfitting, we add a dropout layer. Next layer is a fully connected layer. Finally, the sigmoid activation function is applied to classify the given Weibo users.

We fine-tuned the model by the following methods [23], respectively:

- CNN-rand: all words are randomly initialized and then modified during training.
- CNN-static: all words presented by pre-trained word vectors, including the unknown ones that are randomly initialized are kept static, and only other parameters of the model are learned.
- CNN-nostatic: similar to above but the pre-trained vectors are fine-tuned.

FedAvg model

The federated averaging framework comprises two main parts: server optimization and local training.

First, the server sends the latest model to the client, and the client starts updating the model with local data and uploads the parameters updated to the server. The server aggregates the received parameters, when a certain number of clients are satisfied, generates the latest model, and repeats the steps until the model convergence or the maximum number of iterations is reached. The detailed FedAvg is shown in Algorithm 1.

Algorithm 1 Federated Averaging (FedAvg) [4]

```
Input: $k \in [n]$
Initialize $w_0$
for all epoch $t \in [T]$ do
  Randomly select a group of $k$ workers
  for all $i$ in parallel do
    Receive the latest global model $w_t$ from the server
    $w_{t,0}^{i} \leftarrow w_t$
    for all local iteration $h \in [H_t^i]$ do
      Randomly sample $z_{t,h}^{i}$
      $w_{t,h}^{i} = w_{t,h-1}^{i} - \gamma \nabla l(w_{t,h-1}^{i}; z_{t,h}^{i})$
    end for
    Push $w_{t,H_t^i}^{i}$ to the server
  end for
  Update the global model:
    $w_t = \frac{1}{k} \sum_i w_{t,H_t^i}^{i}$
end for
```

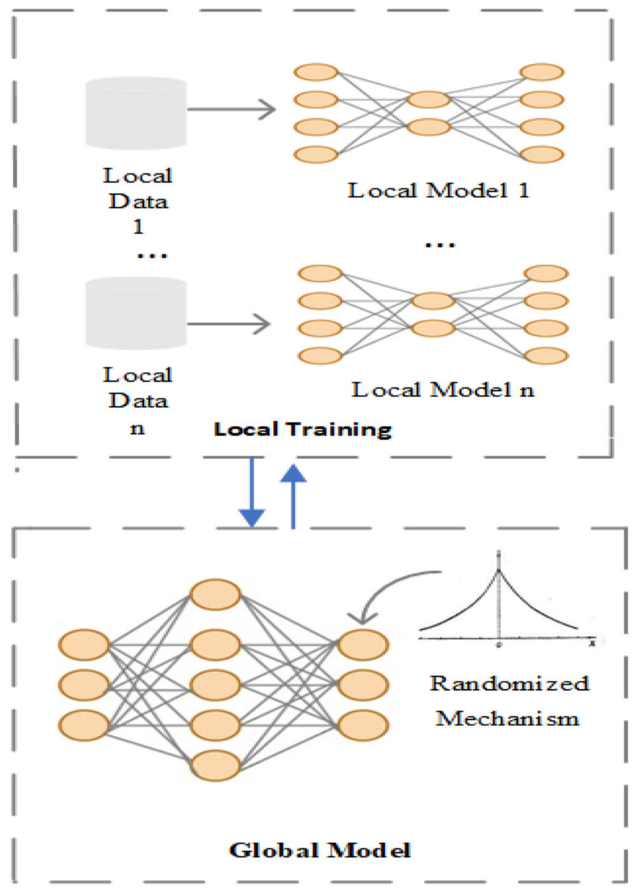


Fig. 3 Architecture for CAFed system: the devices use the local data to train the model, after receiving the parameters transmitted by the server. Differential privacy is added before global model updated on the server

CAFed model

The asynchronous federated learning framework comprises two main parts: local training and global training.

During the local training, each device updates the local model according to the global model sent by the server. The server updates the global model immediately after receiving the parameters uploaded by any device. Different from the federated average algorithm, noise is added before updating the global model to protect the parameters from leakage. Figure 3 shows a flowchart of CAFed model.

We consider federated learning with clients (devices) in a horizontal federated learning system, D^i is the local data on the i th device. Z^i is a sample from the i th device. The overall objective is to train a global model using the distributed local models from all the devices. To do so, we consider Eq. (2) as the final goal of our optimization.

$$\min F(w) = \min\left(\frac{1}{n}\sum_{i\in[n]} E_{z^i\sim D^i}\left(w;\, z^i\right)\right) \tag{2}$$

A simple premise of federated optimization has global epochs, the server receives a local model w^i_{new} from worker i.

$$g_t = w^i_{back} - w^i_{new} \tag{3}$$

$$w^k_{t+1} = w^k_t - \partial_k g^k_t \tag{4}$$

The detailed algorithm shows in Algorithm 2. Given that most devices are edge devices, gradients cannot be directly uploaded to the server. We upload the device's updated model to the server and extract the w^i_{back} from w_{back}. We save every model that ith device uploads to the server in w_{back}, w^i_{back} is the model before w^i_{new} is updated, on the ith device, so that server can calculate the gradient that ith device updated.

```
Algorithm 2 CAFed
Server Process
Initialize $w_0$, $w^i_{back}$=$w_0$,$i \in \{1,2, \ldots, n\}$
Select $n$ clients, and sends them the latest global model with time stamp
for all epoch $t \in [T]$ do
  Receive the pair $(w^i_{new}, \tau)$ from worker $i$
  $g_t = w^i_{back} - w^i_{new}$
  for all $g^k_t \in g_t$ do
    $s(t-\tau) = \sum_{u=j}^{t-1} f_{\{g^k_u \neq 0\}}$
    if $s(t-\tau) = 0$ then $\partial_k \leftarrow 1$ else
      $\partial_k \leftarrow \frac{1}{s(t-\tau)}$
    $\mathrm{w}^{\mathrm{k}}_{\mathrm{t+1}} = \mathrm{w}^{\mathrm{k}}_{\mathrm{t}} - \partial_{\mathrm{k}}(\mathrm{g}^{\mathrm{k}}_{\mathrm{t}} + \beta\mathrm{N}(0,\sigma^2))$
  end for
end for
Worker Processes
for all $i \in n$ in parallel do
  if $\gamma < \frac{1}{(1+e^{-v})}$ then receives the pair of
  global model and its time stamp $(w_{t+1}, t)$ from the server
  $\tau \leftarrow t$, $w^i_{\tau,0} \leftarrow w_{t+1}$
  for all local iteration $h \in [H^i_\tau]$ do
    randomly sample $z^i_{\tau,h} \in D^i$
        $w^i_{\tau,h} = w^i_{\tau,h-1} - \gamma\nabla f(w^i_{\tau,h-1}; z^i_{\tau,h})$
  end for
  Push $(w^i_{\tau,H^i_\tau}, \tau)$ to the server
end for
```

Due to the unreliable connection and low communication efficiency of edge devices, we use Eq. (3) to indirectly calculate the gradient of the device i, so as to avoid the adverse impact of the direct upload gradient on the server. Equation (4) represents the global model updated after the server receives the local model.

Remark 1 Intuitively, large staleness results in a greater error when updating the global model. When one device pulls w, it receives the associated timestamp τ, and updates the w. Meanwhile, intermediate updates (pushed by other devices) increase the timestamp of the server to t. Our solution is spired from [25]. Instead of computing the same staleness for each g_t^k of the update g_t, we define an adaptive staleness for each parameter w as Eq. (5):

$$s(t-\tau) = \sum_{u=\tau}^{t-1} f_{\{g_u^k \neq 0\}} \tag{5}$$

where f_A is the indicator function of condition A equal to 1 if condition A is true and 0 otherwise. As shown in Eq. (6), The staleness $s(t-\tau)$ is computed individually for each parameter by counting the number updates applied to it since the last pull by the worker.

$$\partial_k^i = \begin{cases} \frac{1}{s(t-\tau)}, \ s(t-\tau) \neq 0 \\ 1, \text{ otherwise} \end{cases} \tag{6}$$

The communication is critical in federated learning, to reduce bandwidth consumption, we attempt to deal with the problem by reducing the total number of communications rounds as follows. All devices participating in the training, and the server immediately updates the global model whenever it receives a local model. However, as is shown in Fig. 4 the server can only accept the model from one device for one parameter updated at a time, which means that two or more workers push parameters to the server, the server can only select one of them to update the global model. To improve communication, we choose whether to push at any given time a probabilities choice. When one device wants to push, it generates a pseudo-random number and compares it with other quantities. If the number is larger than that quantity, the data is dropped, otherwise, it is transmitted [25]:

More formally, if one device

$$r < \frac{1}{1+e^{-v}} \tag{7}$$

where $r \in [0, 1]$ is a random number, v is a hyper-parameter. In practice, the right-hand side of inequation (7) is a Sigmoid function. Thus, the right-hand slide lies in (0, 1).

We apply the randomized mechanism into federated learning [30] a Gaussian white noise with the mean of 0 and the variance of 1 is added to the server process in Eq. (8).

Considering the effect of a randomized mechanism for federated learning, we set a tunable parameter β. The updated model of the server is shown in Eq. (8) (Fig. 4).

$$w_{t+1}^k = w_t^k - \partial_k(g_t^k + \beta N(0, \sigma^2)) \tag{8}$$

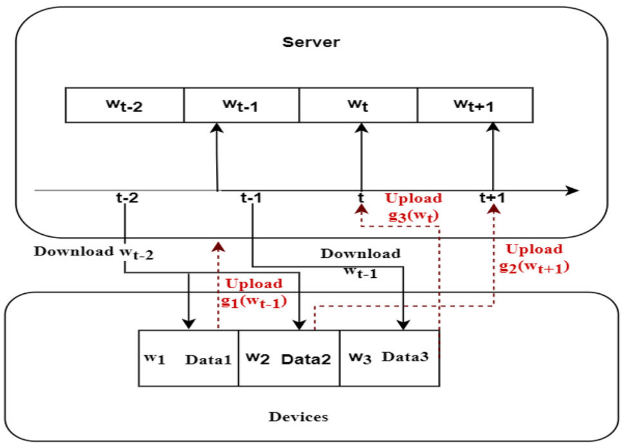


Fig. 4 Updating Process of Global Model. At time $t-2$, the server distributes model w_{t-2} to the device 1 and device 2. Then, these two devices start to train their models. At time $t-1$, device 1 finishes its local training and uploads its local model $g_1(w_{t-1})$ to the central server for aggregation

Assumptions and lemmas

First, we introduce some definitions and assumptions [31] for our convergence analysis.

Definition 2 (*Smoothness*) A differentiable function f is L-smooth if for $\forall x$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2L}\|\nabla \mathrm{f}(x_k)\|^2 \tag{9}$$

Assumption 1 (*Bounded age*) All staleness variables $s(t-\tau)$ are bounde:

$$s(t-\tau) \leq M \tag{10}$$

Assumption 2 (*Strong convexity*) A differentiable function f is μ – strongly convex if for $\forall x$,

$$f(x_{k+1}) \leq f(x_k) - \frac{\mu}{2}\|\nabla \mathrm{f}(x_k)\|^2 \tag{11}$$

Assumption 3 Let z_t^i be a sample from the ith device's local data uniformly at random. The variance of stochastic gradients in each device is bounded:

$$E\left\|\nabla F_i\left(w_t^i, z_t^i\right) - \nabla F_i(w_t^i)\right\|^2 \leq \sigma^2 \text{ for k } = 1, \ldots, n. \tag{12}$$

Lemma 1 (*Results of one step SGD*) Assume Assumption 1 and 2, that

p_i is the weight of the ith device.

If $\gamma \leq \frac{1}{4L}$, we have

$$\begin{aligned}E\|v_{t+1}^{-} - w^*\|^2 \le{}& (1 - \eta_t \mu) E\|w_t^{-} - w^*\|^2 \\ &+ r^2 E\|(g_t - g_t^{-})\|^2 + 6L\eta_t^2 \varnothing \\ &+ 2E\sum_{k=1}^{n} p_k \|w_t^{-} - w_k^i\|^2\end{aligned} \tag{13}$$

where $\varnothing = F^* - \sum_{k=1}^{n} p_k F_k^* \ge 0$

Lemma 2 (*Bounding the variance*) Assume the above Assumption 3 holds, it follows that

$$E\|(g_t - g_t^{-})\|^2 \le \sum_{k=1}^{n} p_k^2 \sigma_k^2 \tag{14}$$

Lemma 3 (*Bounding the divergence of* $\{w_t\}$), Assume $\|\nabla \mathrm{F}(w_t)\|^2 \le \varepsilon^2$, that η_t is non-increasing and $\eta_t \le 2\eta_{t+E}$ for all $t \ge 0$. It follows that

$$E\left[\sum_{k=1}^{n} \|w_t^{-} - w_k^t\|^2\right] \le 4\eta_t^2 (E-1)^2 \varepsilon^2 \tag{15}$$

Convergence guarantees

Based on the above assumptions, we show the convergence rate and how the staleness affects the convergence. With some modification, we adapted further the proof of the convergence by [31–33] and lead to the following theorem, which indicates the convergence rate and our algorithm's linear property of speed-up.

$$g_t = \frac{1}{n}\sum_{i=1}^{n} p_i \nabla l(w_t^i, z_t^i) \tag{16}$$

$$g_t^{-} = \sum_{i=1}^{n} p_i \nabla F_i(w_t) \tag{17}$$

By substituting Lemmas 1 and 2 into Lemma 3, we can get

$$E\|w_{t+1}^{-} - w^*\|^2 \le (1 - \eta_t \mu) E\|w_t^{-} - w^*\|^2 + B \tag{18}$$

where $B = \sum_{k=1}^{n} p_k^2 \sigma_i^2 + 6L\phi + 8(E-1)^2 \varepsilon^2$

Let $\Delta_t = E\|w_{t+1}^{-} - w^*\|^2$, from lemma 1, lemma 2 and lemma3, it follows that

$$\Delta_{t+1} \le (1 - \eta_t \mu)\Delta_t + \eta_t^2 B$$

For a diminishing step size, $\eta_t = \frac{\beta}{t+\gamma}$, it is discretionary, so β and γ need to ensure $\eta_1 = \frac{\beta}{1+\gamma} \le \frac{1}{4L}$ and $\frac{\beta}{t+\gamma} \le 2$. We prove $\Delta_t \le \frac{\upsilon}{\gamma+t}$, where $\upsilon = \max\left\{\frac{\beta^2 B}{\beta\mu - 1}, (\gamma+1)\Delta_1\right\}$

$$\begin{aligned}\Delta_{t+1} &\le (1 - \eta_t \mu)\Delta_t + \eta_t^2 B \\ &= \left(1 - \frac{\beta\mu}{\gamma+t}\right)\left(\frac{\upsilon}{t+\gamma}\right) + \left(\frac{\beta^2 B}{(t+\gamma)^2}\right) \\ &= \frac{t+\gamma-1}{(t+\gamma)^2}\upsilon + \left[\left(\frac{\beta^2 B}{(t+\gamma)^2}\right) - \left(\frac{\beta\mu - 1}{(t+\gamma)^2}\right)\upsilon\right] \\ &\le \frac{\upsilon}{t+\gamma+1}\end{aligned}$$

Then by the strong convexity

$$E\left[F(w_t^{-})\right] - F^* \le \frac{L}{2}\frac{\upsilon}{t+\gamma}$$

Completing the proof.

From the perspective of the sever, the update procedure of parameter w for Algorithm 2 can be written as

$$g_t = \underbrace{\frac{1}{H_\tau^i}\frac{\gamma}{s_t}\sum\nolimits_{k=1}\underbrace{\left(\sum\nolimits_{h=1}^{H_\tau^i}\left(\frac{1}{z_{\tau,h}^i}\left(\sum\nolimits_{z=1}^{z_{\tau,h}^i}\nabla l(w_{\tau,h,k}^i; z_{\tau,h}^i)\right)\right)\right)}_{\boxed{\text{calculated a parameter of } i\text{th device}}}}_{\boxed{\text{aggregated all parameters of } i\text{th device}}} \tag{19}$$

$$\mathrm{w}_{t+1} = \underbrace{\mathrm{w}_t - \mathrm{g}_t}_{\boxed{\text{updated in the server}}} \tag{20}$$

where $w_{\tau,h,k}^i$ denotes kth parameter initialized from w_τ, updated in the hth local iteration, on the ith device. We use the staleness sequence$\{s_t\}$, $t = 0, 1, \ldots, T$ in Algorithm 2. Where s_t is the sum of staleness for all parameters on the ith device.

Note that γ is modulated in (19), one can verify that the convergence rate is faster than in (16). From Theorem 1 proposed by Lei et al. [34] with some modification, we have the following theorem, which indicates the convergence rate and our algorithm's linear speedup property.

Theorem 1 Let C_1, C_2, C_3, C_4 be certain positive constants depending on the loss function $L(\mathrm{w})$. Under certain commonly used assumptions proposed by Lei et al. [34] and the theorem proposed by [35], we can achieve a convergence rate of

$$\frac{1}{\sum_{t=1}^{T} 1/s_t}\sum_{t=1}^{T}\frac{1}{s_t}E\left(\|\nabla L(w_t)\|^2\right) \le 2\frac{\sqrt{\frac{2c_1c_2}{z_{\tau,h}^i}\sum_{t=1}^{T}\left(\frac{1}{s_t^2}\right)}}{\sum_{t=1}^{T}\frac{1}{s_t}} \tag{21}$$

where T is the total epoch number, if

$$\gamma = \sqrt{\frac{c_1 z_{\tau,h}^i}{\sum_{t=1}^{T}\left(\frac{2}{s_t^2}c_2\right)}} \tag{22}$$

under the prerequisite that

$$\gamma \le \frac{c_2}{c_3 s_t \sum_{j=t-M}^{t-1}\frac{1}{s_j^2}}, \forall t \in [M, T] \tag{23}$$

and

$$c_3\frac{\gamma}{s_t} + c_4 M\frac{\gamma^2}{s_t}\sum_{k=1}^{M}\frac{1}{s_{t+k}} \le 1, \forall t \tag{24}$$

When selecting small enough γ, Eqs. (19) and (20) can always be satisfied, it means that the LHS of (21) is guaranteed to converge. We also note that the relationship between staleness and convergence rate can be found by observing the RHS (21).

Remark 2. We found that the RHS of (21) is of the form $h\left(x_1,\ldots,x_T\right) = O\left(\frac{\sqrt{x_1^2+x_2^2\cdots x_T^2}}{x_1+x_2+\cdots x_T}\right)$[35] by letting$x_t = \frac{1}{s_t}$. If $x_1 = x_2 = \cdots = x_T$, h is minimized. We consider the ideal assumption of the staleness, namely, we take s_t as a constant$s$, we have

$$\frac{1}{T}\sum_{t=1}^{T}E\left(\|\nabla L(w)\|^2\right) \le 2\frac{\sqrt{2c_1c_2}}{\sqrt{z_{\tau,h}^i T}} \tag{25}$$

Thus, the convergence rate is roughly $O(1/\sqrt{z_{\tau,h}^i T})$, where T is the total number of epochs, $z_{\tau,h}^i$ denotes the mini-batch size on the ith device. We can draw a conclusion that a linear speedup can be achieved in our Algorithm 2, according to Theorem 1 from [30].

Proof. from (24) we have

$$c_3\frac{\gamma}{s_t} + c_4 M\frac{\gamma^2}{s_t}\sum_{k=1}^{M}\frac{1}{s_{t+k}}$$

$$= c_{3H_\tau^i}\frac{\gamma}{H_\tau^i s_t} + c_4(z_{\tau,h}^i)^2 M\frac{\gamma}{z_{\tau,h}^i s_t}\sum_{k=1}^{M}\frac{\gamma}{z_{\tau,h}^i s_{t+k}}$$

$$\le 1$$

With (22) we have.

$$\frac{\gamma^3}{z_{\tau,h}^i s_t}c_3\sum_{j=t-M}^{t-1}\frac{1}{s_j^2} \le \frac{\gamma^2}{z_{\tau,h}^i s_t^2}c_2, \forall t \tag{26}$$

Note that the upper bound of staleness is M in our setting. Then it follows from Theorem 1 proposed by [30] that

$$\frac{1}{\sum_{t=1}^{T} 1/s_t}\sum_{t=1}^{T}\frac{1}{s_t}E\left(\|\nabla L(w_t)\|^2\right)$$

$$\le \frac{c_1 + \left(\sum_{t=1}^{T}\frac{\gamma^2}{z_{\tau,h}^i s_t^2}c_2 + \frac{\gamma^3}{z_{\tau,h}^i s_t}c_3\sum_{j=t-M}^{t-1}\frac{1}{s_j^2}\right)}{\sum_{t=1}^{T}\frac{\gamma}{s_t}}$$

$$\le \frac{c_1 + \gamma^2\sum_{t=1}^{T}\frac{2}{z_{\tau,h}^i s_t^2}c_2}{\sum_{t=1}^{T}\frac{\gamma}{s_t}}$$

$$= \frac{2\sqrt{c_1\sum_{t=1}^{T}\frac{2}{z_{\tau,h}^i s_t^2}c_2}}{\sum_{t=1}^{T}\frac{1}{s_t}}$$

$$= \frac{2\sqrt{\frac{2c_1c_2}{z_{\tau,h}^i}\sum_{t=1}^{T}\left(\frac{1}{s_t^2}\right)}}{\sum_{t=1}^{T}\frac{1}{s_t}}$$

completing the proof.

Experiments

Data analysis

We experimented on Weibo user's posts for depressive behavioural exploration and detection. Datasets were collected from the social network. The greatest challenge for data collection is to find positive class among Weibo users. Some researchers issue survey/questionnaires to users, but it is not sure that the answers given by patients to the questionnaire are accurate. De Choudhury et al. [32] have employed crowdsourcing to collect comments from several hundred Twitter users who report that they have been diagnosed with clinical MDD using the CES-D2 (Center of Epidemiologic Studies Depression Scale) screening test. However, this method needs to spend money, Crowdsourcing, for example, requires a certain amount of money, so relatively few users are collected.

In this paper, the real users who have experienced medical treatment and non-depression users were screened out through questionnaire consultation. With the permission of the users, we adopted the crawler method to obtain the related Weibo users' data by signing confidentiality agreements and

Table 2 Experimental data set introduction

Total sample	Depression samples		Number of normal users	
	Training sample size	Test sample size	Training sample size	Test sample size
900	253	74	467	106

Table 3 Evaluation results for CNN

Method	Precision (%)	Recall (%)	F-measure (%)	Accuracy (%)
CNN + rand	91.76	80.88	85.94	86.11
CNN + static	90.01	83.33	86.96	83.33
CNN + nostatic	87.50	100	93.33	87.50

Table 4 Comparison results for FedAvg

Method	Precision (%)	Recall (%)	F-score (%)	Accuracy (%)
FedAVg + rand	89.47	85.00	87.18	83.33
FedAvg + static	81.82	81.82	81.82	73.33
FedAvg + nostatic	100	81.82	90.00	86.67

privacy norms. We adopted the crawler method to collect 1000 Weibo users' data, which includes nickname, number of Weibo, number of fans, number of followers, and content of Weibo. We collected a user's data for one year. We also exclude those vague expressions, for example, 'I seem to have depression,' or 'I have been depressed for a long time' and' suspected that I was depressed', etc. We divide the data set into two groups. One group is used to count the posts posted by users, and the other is used to count the number of posts by users, the number of fans, the number of likes, the number of comments, and the number of followers. Details of the experimental data set are shown in Table 2. The evaluation matrices parameters (precision, recall and F-measure) have been used to execute these classifiers. It has been conducted based on four different ways [7]. True Positive (TP) is the depression cases that are positive and anticipated as positive and anticipated as positive. True Negative (TN) is the depression cases that are negative and anticipated as negative. False Negative (FN) is the depression cases that are positive but anticipated to be negative. False Positive (FP) is the depression cases that are negative but anticipated to be positive. All the evaluation metrics are defined as follows.

Experiment results

1. CNN
   In this experiment, the word embedding is trained on the posts, where are 900 users. The word embedding is fine-tuned during training to improve performance. According to our experiments, we set the length of the ordered vector set $L = 27{,}941$, the output dimension of word2vec $d = 300$. In addition, we also adjust the batch size which is a parameter used with the Stochastic Gradient Descent (SGD) method in the model training phase. Here, the SGD updates the weights in the CNN after each set of batch size are examined. A large batch size increases the chance of overfitting, while a smaller batch size may result in a longer time for convergence [11]. Therefore, we empirically set batch size = 32, which achieves a good balance among the two factors mentioned above. We use l_2 regulation to avoid overfitting. The other parameters used in our experiments are as follows: the number of filters: 128, dropout: 0.5, and loss function: binary cross-entropy. The results are shown in Table 3.
2. Federated averaging

Table 5 Comparison results on CAFed

Method	Precision (%)	Recall (%)	F-measure (%)	Accuracy (%)
CAFed + rand	86.90	85.25	73.20	67.92
CAFed + static	80.00	89.75	84.59	80.00
CAFed + nostatic	90.00	81.82	85.26	86.67

Table 6 Magnitude of randomization vs. testing accuracy on Weibo

Randomization	CAFed-rand (%)	CAFed-static (%)	CAFed-nostatic (%)
Nonrandomized			
$\beta = 0$	85.27	80.00	86.67
Randomized			
$\beta = 0.0001$	83.67	78.00	84.00
$\beta = 0.001$	81.25	76.27	81.62
$\beta = 0.01$	78.63	74.00	78.00
$\beta = 0.05$	73.00	71.23	74.00

We apply the FedAvg to detect depression and use the CNN model of the first experiment as the network structure. Considering the lack of a large enough data set and hardware devices problems, after many experiments, we finally determined that the training set is partitioned onto n = 10 devices, each of the n = 10 partition has 72 Weibo users in each round. For any worker, the minibatch size for SGD is 16. The detailed FedAvg is shown in Algorithm 1 and the results are shown in Table 4.

3. CNN asynchronous federated
   To make a comparison with the previous two experiments, we also adopted the CNN model as the network architecture. The detailed CAFed is shown in Algorithm 2, and the results are shown in Table 5. In each experiment, the training set is also partitioned onto $n = 10$ devices. We evaluate the magnitude of normal noise in the randomization mechanism on the model. Comparative experiments are conducted to analyze the effect of the magnitude of noise on model performance. The results are shown in Table 6 with both randomized and nonrandomized settings. For the randomized version, four values of magnitude are chosen, i.e., 0.0001, 0.001, 0.01 and 0.05.
4. Experiments on MNIST
   To make our proposed algorithm CAFed more reliable, we evaluate the performance on an extensive training set, composed by the MNIST training dataset (i.e., with 60,000 images). The data is shuffled, and then partition into 100 devices, each receiving 600 images. We divided the Non-IID data. Firstly, we sort the data by digital label, divide it into 200 shares of size 300, and assign each of 100 devices 2 shards. This is a pathological Non-IID partition of the data because many devices will only have examples from two digits. Both of these partitions are balanced.

Table 7 shows the test set accuracy with different methods. We use a convolutional neural network (CNN) with three convolutional layers and dropout layers followed by one fully connected layer.

Table 7 Magnitude of randomization vs testing accuracy on MNIST

Randomization	FedAvg (%)	CAFed (%)
Nonrandomized		
$\beta = 0$	72.23	83.26
Randomized		
$\beta = 0.0001$	70.26	81.75
$\beta = 0.001$	68.14	80.00
$\beta = 0.01$	67.75	78.13
$\beta = 0.05$	65.00	76.00

Analysis of experimental results

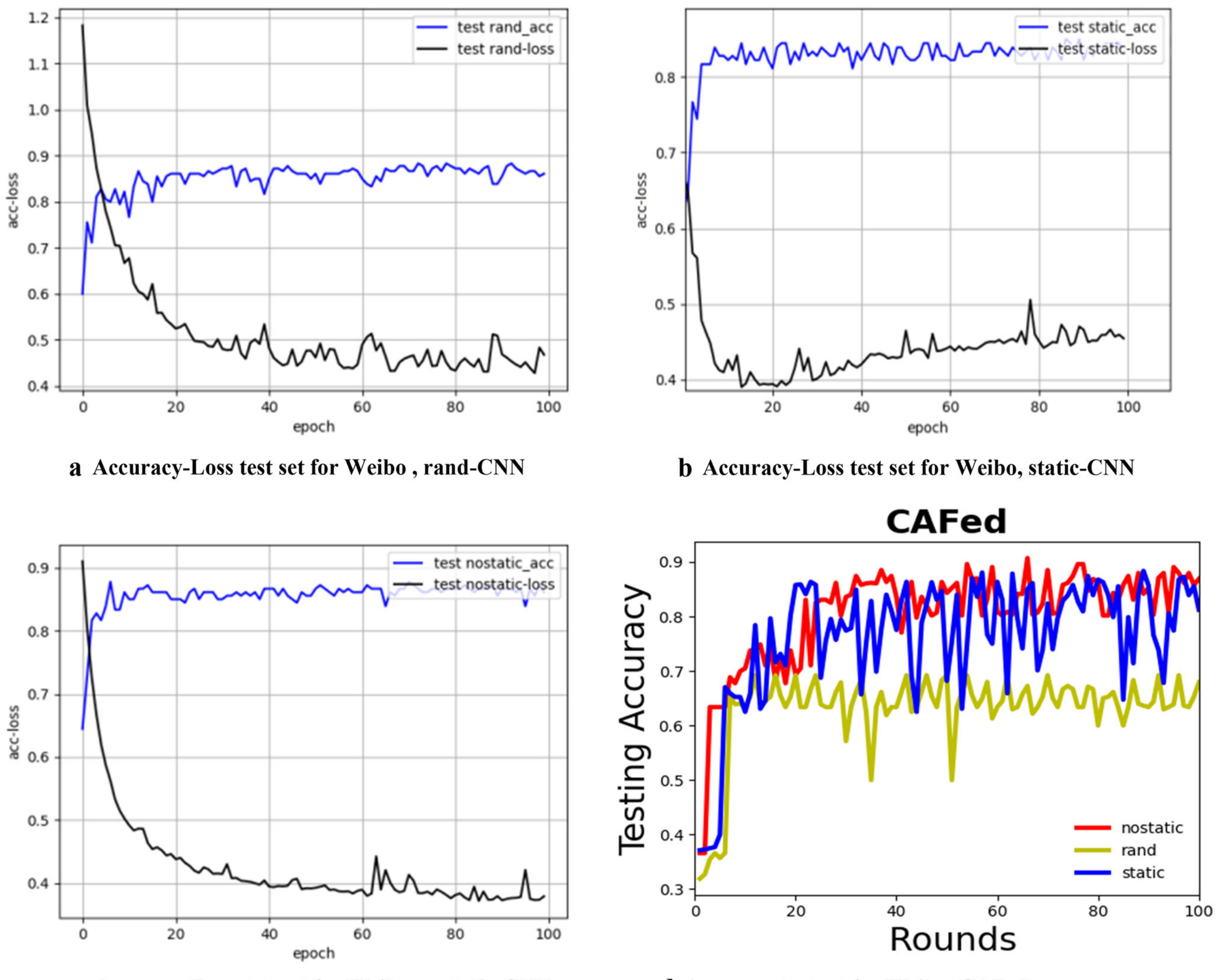
For all the algorithms under investigation, we compute the number of global epochs by counting how many times the global model is updated. The total number of global epochs $T = 200$ in Algorithm 1, and $T = 100$ in Algorithm 2. In Fig. 5, we show how to word embedding type affects the performance of the model. It can be noted that CNN-nostatic is performing better than other models.

From Fig. 6, we can observe that the performance of the FedAvg-nostatic model is better than other models. At the same time, we find that the curve of the federated average model fluctuates more than that of the machine learning model. We think there are two reasons: first, the data set is not large enough, each device has only a small amount of data, which affects the performance of the model. Second, deep learning method is not concerned with data security and preserve protection. Deep learning method is to pull all data together and use all data to train a modelM_{SUM}. A federated learning system is a learning process in which the data owners collaboratively train a model M_{FED}, in which process any worker does not expose its data to a server or other workers. From the inequality (27) [4], we can conclude that the performance of M_{FED} is not well than M_{SUM}. $\mathrm{ACC}_{\mathrm{FED}}$ and $\mathrm{ACC}_{\mathrm{sum}}$ represent the accuracy of Federated learning and deep learning, respectively. Let δ be a non-negative real number, if

$$|\mathrm{ACC}_{\mathrm{FED}} - \mathrm{ACC}_{\mathrm{SUM}}| < \delta \tag{27}$$

we say federated learning has $\boldsymbol{\delta - accuracy}$ loss.

From Fig. 7, we can observe that our proposed method is performing better than FedAvg when differential privacy is not added. After the addition of differential privacy, it is obvious that the performance of the model decreases, but the privacy of users is protected. As shown in Table 6, a small noise did not cause a too big effect on the performance of the model. With a large noise, the performance of the model becomes worse. We observed that the difference in accuracy between a no-private and a private model is about



a Accuracy-Loss test set for Weibo , rand-CNN

b Accuracy-Loss test set for Weibo, static-CNN

c Accuracy-Loss test set for Weibo, nostatic-CNN

d Accuracy test set for Weibo, CAFed

Fig. 5 Test set accuracy and convergence for Text-CNN, **a**–**c** use different fine-tuned model. **d** Describes test set accuracy for Algorithm 1, with a different fine-tuned model. Noted that the accuracy of nostatic-CNN is high than other models. Actually, they are centralized frameworks, namely, all participants' data are sent to a central server, so it is very effective. However, it violates data privacy as all users' data are exposed to the server

2%, therefore, we should make a trade-off between accuracy and privacy protection.

In Table 7, we observed that the effect of the magnitude on the test accuracy under the different model structure, and because of the added noise, models need more running rounds to converge.

In Figs. 6 and 7, we show CAFed and FedAvg convergence when the number of global epochs grows. Obviously, CAFed converges faster than FedAvg. Because FedAvg has to wait for some devices respond in each epoch, while CAFed only needs one device's response to move on to the next epoch. What's more, in each global epoch, FedAvg has more communications compared with CAFed. Overall, with the same amount of communication overhead, CAFed converges faster than FedAvg.

In Fig. 8, we show the test set accuracy-loss performance by use of MNIST with different methods. As shown in Fig. 8, CNN makes more progress in each round, however, the performance of the federated average model is not as good as the asynchronous federated model, and more training rounds are needed. Although our model has a higher accuracy than FedAvg, the curve fluctuation is relatively large. It is believed that the global model can be aggregated by the proposed method. The asynchronous federated model updates the global model immediately after receiving any parameter uploaded by the client, however, the performance of the

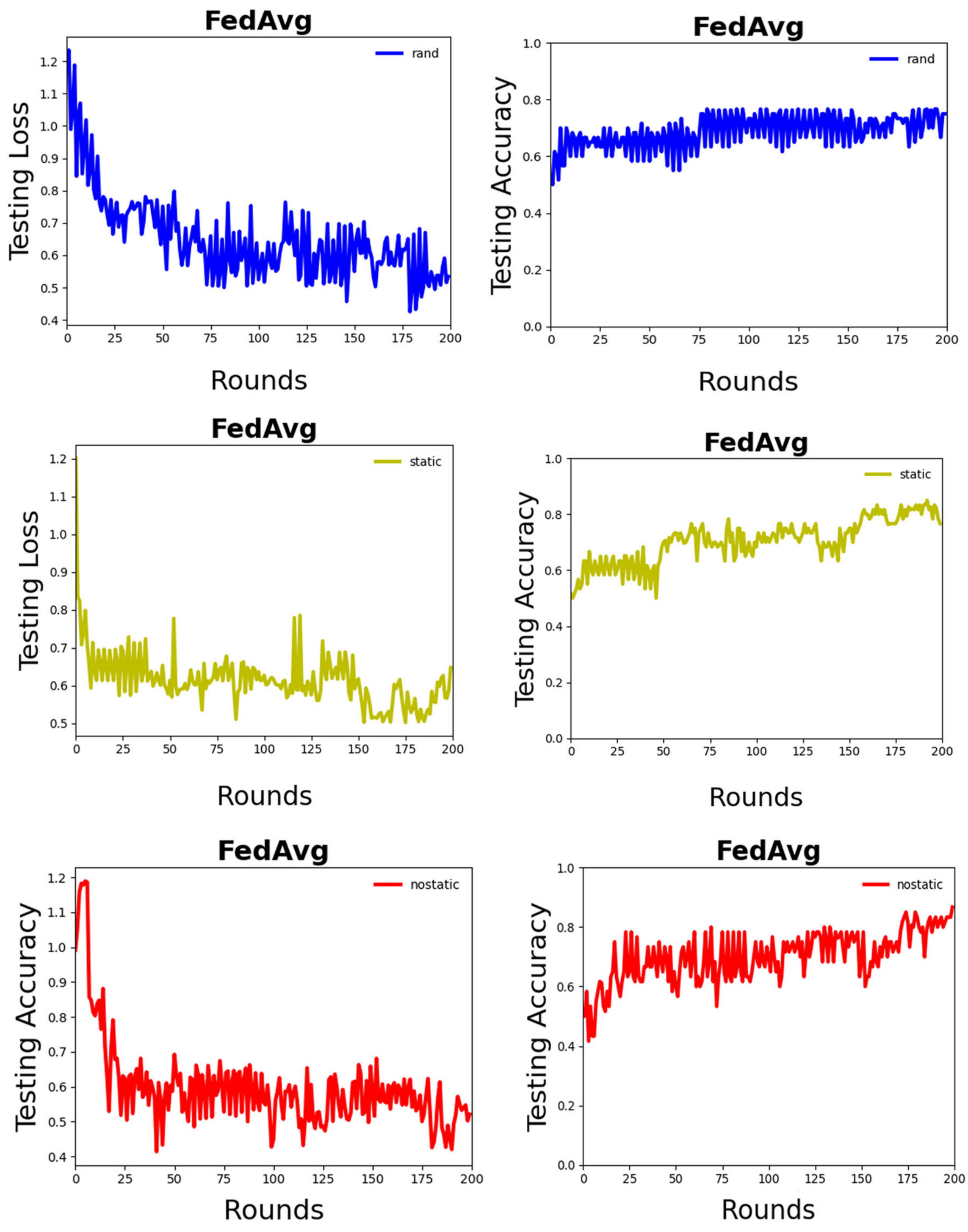


Fig. 6 The left column is test set convergence, and the right is the accuracy of test set for Weibo. It is obvious that the performance of FedAvg-nostatic is better than other fine-tuned model

model is affected by the different data quality and device environment of each client. On the other hand, we think that the inclusion of differential privacy has also affected the accuracy of the model.

A comparison among different deep learning frameworks is provided in Table 8. In summary, the centralized framework is very effective with higher accuracy than a distributed framework, but it violates data privacy as all devices' data are exposed to the server, and if the central server fails, the whole network stops working. In the distributed framework,

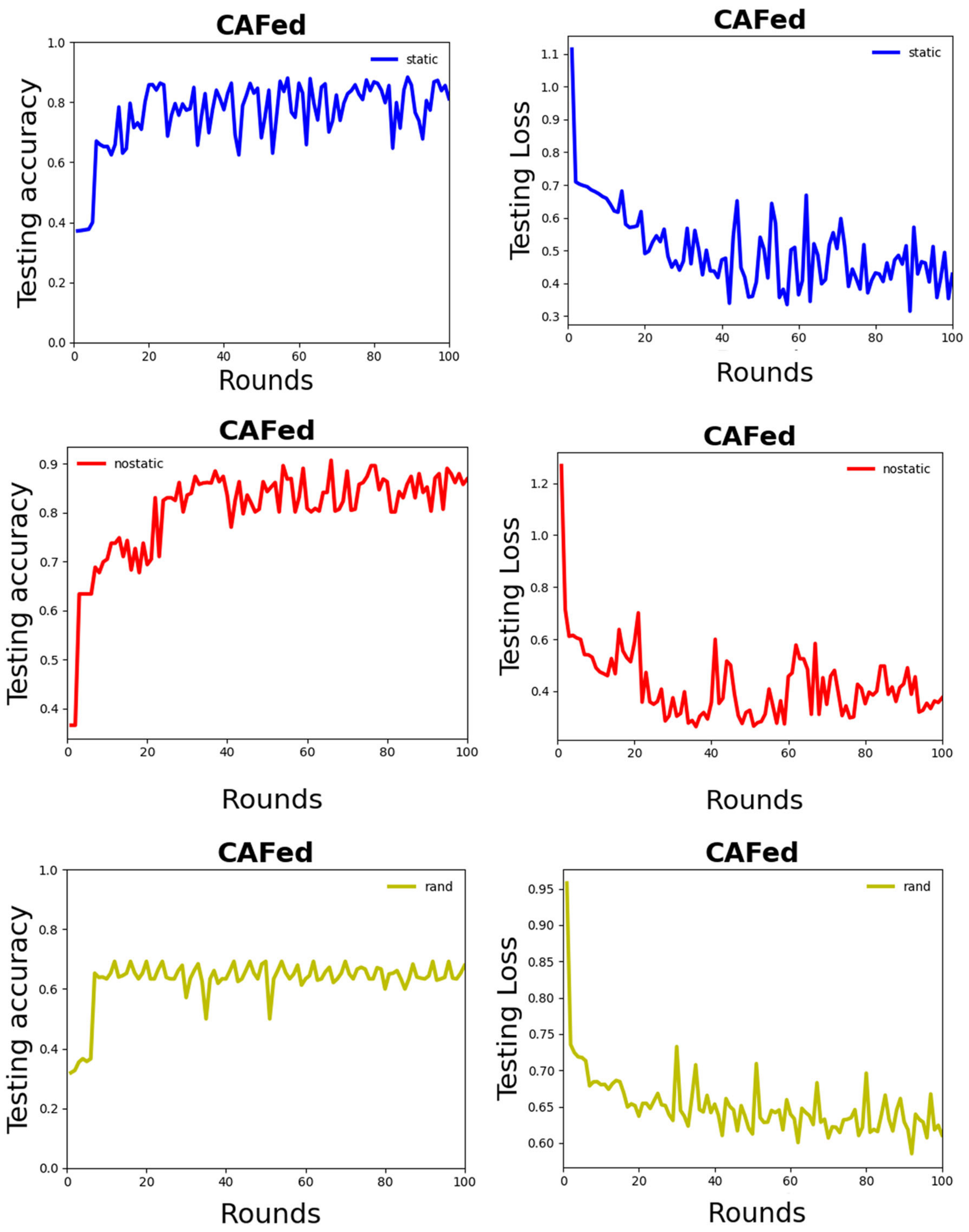


Fig. 7 The left column is test set accuracy, and the right is the convergence of test set, with nostatic fine-tuned model. Note that CAFed needs less training rounds and the accuracy of CAFed model is high than FedAvg

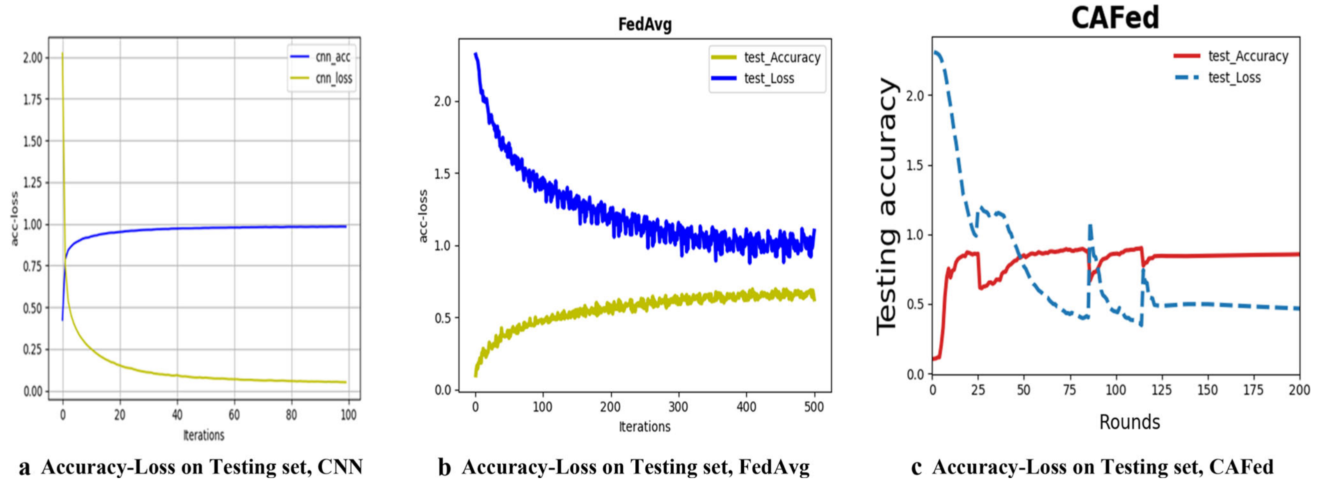


Fig. 8 Testing set accuracy-loss for the MNIST, with different fine-united model

Table 8 Comparative results analysis on Weibo

Frameworks	Technique used	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
Centralized framework	CNN + rand	86.11	91.67	80.88	85.94
	CNN + static	83.33	90.01	83.33	86.96
	CNN + nostatic	87.50	87.50	100	93.33
Distributed framework	FedAvg + rand	83.33	89.47	85.00	87.18
	FedAvg + static	73.33	81.82	81.82	81.82
	FedAvg + nostatic	86.67	100	81.82	90.00
	CAFed + rand	67.92	86.90	85.25	73.20
	CAFed + static	80.00	80.00	89.75	84.59
	CAFed + nostatic	86.67	90.00	81.82	85.26

devices collaboratively train a model by sharing local model updates to avoid privacy leakage.

Discussion

To better understand our proposed algorithm, we apply the following techniques for depression detection, such as text-based CNN technology, FedAvg and CAFed. Table 8 shows the results of various models. It is clearly observed that the nostatic mode performs better than the other modes, it not only uses a pre-trained model, but also continuously adjusts the word vectors during training. The static mode uses a pre-training model, but does not adjust the word vector during the training process; the random mode does not use the pre-training model, but uses random initialization, and then continuously adjust word vectors in the training process. Although the F1 score of the FedAvg algorithm is best in a distributed framework, it usually requires more communication rounds to achieve better results, and CAFed can also achieve the same effect as FedAvg in nostatic mode, and not so many communication rounds. The most important is that CAFed also considers the user's privacy and security, and makes a trade-off between the privacy protection and the accuracy. Systematically, CAFed has the following advantages compared to FedAvg (Federated Average):

1. Convergence rate
   For asynchronous federated learning (Fedasync), when the server receives a local model uploaded by any worker, it updates the global model immediately. Unlike synchronous federated learning (Fedsync), the server needs to wait for a subset of available workers to push parameters before aggregating them. In general, CAFed converges much faster than FedAvg.
2. Communication costs
   All devices participate in training, the server can only accept one device's parameters to update in each global

epoch. In order to reduce the communication costs, we only select one of the devices to push, and at the same time, avoid network congestion. Unlike FedAvg, if the number survived devices are too small, the entire global epoch including all the received updates may be dropped by the server.

3. Scalability
The federated averaging can only process hundreds of devices in parallel at a time, which may also cause network congestion, while the asynchronous algorithm not only does not cause network congestion but can also process many devices. Due to the limited data set we collected; this advantage was not obvious in our experiments.

Conclusion and future work

In this paper, we propose a novel asynchronous federated optimization algorithm on Non-IID training data. We have proved that a linear speedup can be achieved in our algorithm. As shown in the experiments, the proposed CAFed system outperforms other baselines approaches, we have considered users' privacy. It is regarded to be significant in regard to the recognition of depressed patients without revealing their privacy. After all, no one wants their privacy to be exposed, especially some sensitive information. Moreover, we consider that these results can help develop traditional detection methods, enabling those suffering from depression to be detected and receive treatment as soon as possible.

To better understand the proposed algorithm, we apply text-based CNN technology, FedAvg and CAFed for depression detection in this paper. Table 8 shows the results of various models with different frameworks. It is reasonable that text-based CNN achieves 87.50% of accuracy. Because it applies the traditional data transactions models, specifically, one party collects and transfers data to another party, and other parties will be responsible for cleaning and fusing the data. Finally, a third party will take the integrated data and build models for other parties to use [15]. So, we need to push the data to a third party to ensure the accuracy of the model during the entire data transmission processing. Unlike text-based CNN, the parties did not expose the data to the server or other parties, so the other two models are slightly less accurate compared to the first model. The deep learning technology guarantees the accuracy of the model, it does not consider data security and privacy. Federated learning technology can not only help multiple parties to build a sharing model but also strengthen data privacy and security [13].

We propose a novel asynchronous federated optimization algorithm for Weibo users' data. It is proved that a linear speedup can be achieved in our algorithm. As shown in the experiments, the proposed CAFed system outperform other baseline approaches, the current system achieved 86.67% recognition rate of depression, which can effectively distinguish between depression and normal individuals in practice. Most importantly, we have considered users' security and privacy. We believe that it is significant to recognise depressed patients without revealing their privacy. These results can help develop traditional detection methods, enabling those suffering from depression to be detected and receive treatment as soon as possible. Our future work includes: (1) Considering behavior-based features as additional information sources to further boost up the performance; (2) taking into account incentive mechanism module to improve the proposed algorithm; (3) extending the proposed approach to effectively detect other important mental illness, such as anxiety and bipolar disorders [36].

References

1. W. H. Organization (2019) The world health report—mental health. https://www.who.int/topics/depression/zh/
2. I. o. Depression (2019) 2019 China depression blue paper. https://www.medsci.cn/article/show_article.do?id=1db71861e8f6
3. S. Technology (2019) 2019 Weibo Q4 and annual financial report. https://tech.sina.com.cn/i/2020-02-26/doc-iimxxstf4598954.shtml
4. Yang Q, Liu Y, Chen T, Tong Y (2019) Federated machine learning: concept and applications. ACM Trans Intell Syst Technol 10(2) (**art. no. 12.**)
5. Zafar A, Chitnis S (2020) Survey of depression detection using social networking sites via data mining. In: 2020 10th international conference on cloud computing, data science & engineering (confluence), p 88–93
6. Hu Q, Li A, Heng F, Li J, Zhu T (2015) Predicting depression of social media user on different observation windows. In: 2015 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology (WI-IAT), vol 1, p 361–364
7. Islam MR, Kabir MA, Ahmed A, Kamal ARM, Wang H, Ulhaq A (2018) Depression detection from social network data using machine learning techniques. Health Inf Sci Syst 6(1):8
8. Brendan McMahan H, Moore E, Ramage D, Hampson S, Agüera y Arcas B (2016) Communication-efficient learning of deep networks from decentralized data. arXiv:1602.05629. Accessed 01 Feb 2016

9. Al-Mosaiwi M, Johnstone T (2018) In an absolute state: elevated use of absolutist words is a marker specific to anxiety, depression, and suicidal ideation. Clin Psychol Sci 6(4):529–542
10. Burdisso SG, Errecalde M, Montes-y-Gomez M (2019) A text classification framework for simple and effective early depression detection over social media streams. Expert Syst Appl 133:182–197
11. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv:1408.5882. Accessed 01 Aug 2014
12. Ding Y, Chen X, Fu Q, Zhong S (2020) A depression recognition method for college students using deep integrated support vector algorithm. IEEE Access 8:75616–75629
13. Wu MY, Shen C-Y, Wang ET, Chen ALP (2020) A deep architecture for depression detection using posting, behavior, and living environment data. J Intell Inf Syst 54(2):225–244
14. Trotzek M, Koitka S, C. M. J. I. T. o. K. Friedrich, D. Engineering (2018) Utilizing neural networks and linguistic metadata for early detection of depression indications in text sequences. p 1-1
15. Mikolov T, Sutskever I, Chen K, Corrado G, Dean G (2013) Distributed representations of words and phrases and their compositionality. arXiv:1310.4546. Accessed 01 Oct 2013
16. Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, Xiaoyong Du (2018) Analogical reasoning on chinese morphological and semantic relations, ACL
17. Liu Y, Yuan X, Xiong Z, Kang J, Wang X, Niyato D (2020) Federated learning for 6G communications: challenges, methods, and future directions. China Commun 17(9):105–118
18. Li Y, Sahu AK, Talwalkar A, Smith V (2020) Federated learning: challenges, methods, and future directions. IEEE Signal Process Mag 37:50
19. Konečný J, Brendan McMahan H, Ramage D, Richtárik P, (2016) Federated optimization: distributed machine learning for on-device intelligence. arXiv:1610.02527. Accessed 01 Oct 2016
20. Xie C, Koyejo S, Gupta I (2019) Asynchronous federated optimization. arXiv:1903.03934. Accessed 01 Mar 2019.
21. Chen Y, Ning Y, Slawski M, Rangwala H (2019) Asynchronous online federated learning for edge devices with non-IID data. arXiv:1911.02134. Accessed 01 Nov 2019
22. Zhang Y, Wallace B (2015) A sensitivity analysis of (and Practitioners' Guide to) convolutional neural networks for sentence classification. arXiv:1510.03820. Accessed 01 Oct 2015
23. https://github.com/goto456/stopwords
24. Joulin A, Grave E, Bojanowski P, Mikolov T (2016) Bag of tricks for efficient text classification. arXiv:1607.01759. Accessed 01 July 2016
25. Hardy C, Merrer EL, Sericola B (2017) Distributed deep learning on edge-devices: feasibility via adaptive compression. In: 2017 IEEE 16th international symposium on network computing and applications (NCA), 2017, p 1–8
26. Carlini N, Liu C, Erlingsson Ú, Kos J, Song D (2018) The secret sharer: evaluating and testing unintended memorization in neural networks. arXiv:1802.08232. Accessed 01 Feb 2018
27. Cynthia D, Aaron R (2013) The algorithmic foundations of differential privacy. Found Trends Theor Comput Sci 9:211–407
28. Bu Z, Dong J, Long Q, Su WJ (2019) Deep learning with Gaussian differential privacy. arXiv:1911.11607. Accessed 01 Nov 2019
29. Fan Y, Yu R, Li J, Zhu J, Li X (2020) EEG-based mild depression recognition using multi-kernel convolutional and spatial-temporal feature. In: 2020 IEEE international conference on bioinformatics and biomedicine (BIBM), 2020, p 1777–1784
30. Ji S, Pan S, Long G, Li X, Jiang J, Huang Z (2019) Learning private neural language modeling with attentive aggregation. In: 2019 International joint conference on neural networks (IJCNN), p 1–8
31. Li X, Huang K, Yang W, Wang S, Zhang Z (2019) On the convergence of FedAvg on non-IID data. arXiv:1907.02189. Accessed 01 July 2019. https://ui.adsabs.harvard.edu/abs/2019arXiv190702189L.
32. De Choudhury M, Gamon M, Counts S, Horvitz E (2013) Predicting depression via social media
33. Odena A (2016) Faster asynchronous SGD. arXiv:1601.04033. Accessed 01 Jan 2016
34. Lei J, Shanbhag UV (2017) Asynchronous schemes for stochastic and misspecified potential games and nonconvex optimization. arXiv:1711.03963. Accessed 01 Nov 2017
35. Zhang W, Gupta S, Lian X, Liu J (2015) Staleness-aware Async-SGD for distributed deep learning. arXiv:1511.05950. Accessed 01 Nov 2015
36. Rush AJ, Gullion CM, Basco MR, Jarrett RB, Trivedi MH (2009) The inventory of depressive symptomatology (IDS): psychometric properties. Psychol Med 26(3):477–486