Decentralized Edge Intelligence: A Dynamic Resource Allocation Framework for Hierarchical Federated Learning

Wei Yang Bryan Lim, Jer Shyuan Ng, Zehui Xiong, Jiangming Jin, Yang Zhang, Dusit Niyato, *Fellow, IEEE*, Cyril Leung, Chunyan Miao

Abstract—To enable the large scale and efficient deployment of Artificial Intelligence (AI), the confluence of AI and Edge Computing has given rise to Edge Intelligence, which leverages on the computation and communication capabilities of end devices and edge servers to process data closer to where it is produced. One of the enabling technologies of Edge Intelligence is the privacy preserving machine learning paradigm known as Federated Learning (FL), which enables data owners to conduct model training without having to transmit their raw data to third-party servers. However, the FL network is envisioned to involve thousands of heterogeneous distributed devices. As a result, communication inefficiency remains a key bottleneck. To reduce node failures and device dropouts, the Hierarchical Federated Learning (HFL) framework has been proposed whereby cluster heads are designated to support the data owners through intermediate model aggregation. This decentralized learning approach reduces the reliance on a central controller, e.g., the model owner. However, the issues of resource allocation and incentive design are not well-studied in the HFL framework. In this paper, we consider a two-level resource allocation and incentive mechanism design problem. In the lower level, the cluster heads offer rewards in exchange for the data owners' participation, and the data owners are free to choose which cluster to join. Specifically, we apply the evolutionary game theory to model the dynamics of the cluster selection process. In the upper level, each cluster head can choose to serve a model owner, whereas the model owners have to compete amongst each other for the services of the cluster heads. As such, we propose a deep learning based auction mechanism to derive the valuation of each cluster head's services. The performance evaluation shows the uniqueness and stability of our proposed evolutionary game, as well as the revenue maximizing properties of the deep learning based auction.

Index Terms—Federated Learning, Edge Intelligence, Resource Allocation, Evolutionary Game, Auction

1 Introduction

Today, the predominant approach for Artificial Intelligence (AI) based model training is cloud-centric, i.e., the data owners transmit the training data to a public cloud server for processing. However, this is no longer desirable due to the following reasons. Firstly, privacy laws, e.g., the General Data Protection Regulation (GDPR) [1], are increasingly stringent. In addition, the privacy-sensitive data owners can opt out of data sharing with third parties. Secondly, the transfer of massive quantities of data to the distant cloud burdens the communication networks and incurs unacceptable latency especially for time-sensitive tasks. As such, this necessitates the proposal of Edge Computing [2] as an alternative, in which raw data are processed at the edge of the network, closer to where data are produced.

The confluence of Edge Computing and AI gives rise to Edge Intelligence, which leverages on the storage, communication, and computation capabilities of end devices and edge servers to enable edge caching, model training, and inference [3] closer to where data are produced. One of the enabling technologies [4] of Edge Intelligence is the privacy preserving machine learning paradigm termed *Federated Learning* (FL) [5]. In FL, only the updated model parameters, rather than the raw data, need to be transmitted back to the model owner for global aggregation. The main advantages of FL are: (i) FL enables privacy preserving collaborative machine learning, (ii) FL leverages on the computation capabilities of IoT devices for local model training, thus reducing the computation workload of the cloud, and (iii) Model parameters are often smaller in size than raw data, thus alleviating the burden on backbone communication networks. This has enabled several practical applications, e.g., in the development of next-word-prediction models for text messaging [6], healthcare [7], Unmanned Aerial Vehicles (UAV) sensing [8], and mobile edge computing [9].

However, the FL network is envisioned to involve thousands of heterogeneous distributed devices, e.g., smartphones and Internet of Thing (IoT) devices [10]. In this case, the communication inefficiency remains a key bottleneck in FL. Specifically, node failures and device dropouts due to communication failures can lead to inefficient FL. Moreover, workers, i.e., data owners, with severely limited connectivity are unable to participate in the FL training, thus adversely affecting the model's ability to generalize. As such, solutions from edge computing have recently been incorporated to solve the communication bottleneck in FL. In [4], [11], [12], a *hierarchical* FL (HFL) framework is pro-

WYB. Lim and JS. Ng is with Alibaba Group and Alibaba-NTU Joint Research Institute (JRI), Nanyang Technological University (NTU), Singapore. Emails: limw0201@e.ntu.edu.sg, s190068@e.ntu.edu.sg. Z. Xiong is with Singapore University of Technology and Design, Information Systems Technology and Design (ISTD) Pillar, Singapore. Email: zehui_xiong@sutd.edu.sg. J. Jin is with TuSimple, China. Email: jiangming.jin@outlook.com. Y. Zhang is with College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. Email: yangzhang@nuaa.edu.cn. D. Niyato is with School of Computer Science and Engineering, NTU, Singapore. E-mail: dniyato@ntu.edu.sg. C. Leung is with the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY) and Department of Electrical and Computer Engineering, The University of British Columbia, Canada. Email: cleung@ece.ubc.ca. C. Miao is with SCSE, NTU, Singapore, Alibaba-NTU JRI, LILY, Singapore. E-mail: ascymiao@ntu.edu.sg. Corresponding author: Zehui Xiong.

posed in which the workers do not communicate directly with a central controller, i.e., the model owner. Instead, the local parameter values are first uploaded to edge servers, e.g., at base stations, for intermediate aggregation. Then, communication with the model owner is further established for global aggregation. Besides reducing the instances of global communications with the remote servers of the model owner, this relay approach reduces the dropout rate of devices.

While [11] discusses convergence guarantees and presents empirical results to show that the HFL approach does not compromise on model performance, the challenges of resource allocation and incentive mechanism design have not yet been well-addressed in the HFL framework. In 5G and Beyond networks, the resource sharing and incentive mechanism design for end-edge-cloud collaboration is of paramount importance to facilitate efficient Edge Intelligence [4].

In this paper, we consider a decentralized learning based system model inspired by the HFL. In our system model, there exist data owners, hereinafter referred to as workers, that participate in the FL model training facilitated by different cluster heads, e.g., base stations that support the intermediate aggregation of model parameters and efficient relaying to the model owners (Fig. 1). We consider a two-level resource allocation and incentive design problem as follows:

1) *Lower level (Between workers and cluster heads):* Each worker can freely choose which cluster to join. To encourage the participation of workers, the cluster heads offer reward pools to be shared among workers based on their data contribution in the cluster. For example, a worker that has contributed more data[1] during its local training will receive a larger share of the reward pool. Moreover, the cluster heads offer the workers resource blocks, i.e., bandwidth, to facilitate efficient uplink transmission of the updated model parameters. However, as more workers join a cluster, the payoffs are inevitably reduced due to the division of rewards over a larger number of workers and the increased communication congestion. Thus, the cluster selection strategies of each worker can affect the payoffs of other workers. Accordingly, the workers may slowly adapt their strategies in response to other workers. In contrast to conventional optimization approaches, we use the evolutionary game theory [14] to derive the equilibrium composition of the clusters. Our game formulation enables the bounded rationality and worker dynamics to be captured. Specifically, the workers gradually adapt their strategies in response to other non-cooperative workers. To achieve their objectives, they observe each others' strategies and gradually adjust their strategies accordingly. The solution is therefore, not immediately derived.
2) *Upper level (Between cluster heads and model owners):* There may be multiple model owners in the network that aim to train a model for their respective usage collaboratively with the participation of the workers and cluster heads. However, at any point of time, each worker and cluster head can only participate in the training process with a single model owner. To derive the allocation of cluster head to the model owner, as well as the optimal pricing of the services of the cluster head by the competitive model owners, we adopt a deep learning based auction mechanism which preserves the properties of truthfulness of the bidders, while simultaneously achieving revenue maximization for the cluster heads.

1. Note that this knowledge is available to the cluster heads given that the workers have to report their available resources during the client selection procedure [13]. In this paper, we omit discussions regarding the client selection phase.

The main contributions of our paper are as follows:

1) We propose a joint resource allocation and incentive design framework for the HFL. The "Edge for AI" [15] approach supports decentralized Edge Intelligence, i.e., FL at the edge with reduced reliance on a central controller.
2) We model the cluster selection decisions of the workers as an evolutionary game. Then, we provide proofs for the uniqueness and stability of the evolutionary equilibrium. In contrast to conventional optimization tools which assume that the players are perfectly rational, our model enables us to capture the dynamics and bounded rationality of player decisions.
3) To assign the cluster heads to model owners, we use a deep learning based auction mechanism. In contrast to conventional auctions, the deep learning based auction ensures seller revenue maximization while satisfying the individual rationality and incentive compatibility constraints.

The organization of our paper is as follows. In Section 2, we provide a review of related works. In Section 3, we discuss the system model and problem formulation. In Section 4, we study and analyze the evolutionary game. In Section 5, we discuss the deep learning based auction mechanism. Section 6 provides the performance evaluation and Section 7 concludes the paper.

2 Related Work

FL is a privacy-preserving machine learning paradigm first proposed in [5]. In distributed learning schemes such as FL, the communication cost often dominates the computation cost. In particular, the uplink transmission rate of workers is a major bottleneck in the training process and can lead to the straggler's effect [16]. Several works have proposed a variety of solutions, e.g., model compression techniques such as quantization and subsampling [17], client selection protocols to reduce the occurrences of stragglers [13], as well as Broadband Analog Aggregation (BAA) with over-the-air computation [18].

However, despite the above measures, the FL process is still susceptible to device dropouts. In addition, devices that are located at geographically distant locations are unable to participate in the FL model training process. This

affects the model's ability to generalize well. Recently, edge computing-inspired solutions have been proposed to further enhance the communication efficiency of FL. These methods generally attempt to reduce the reliance of the FL training process on a central controller. In [11], the HFL framework is proposed in which the workers do not communicate directly with a central controller, i.e., the model owner. Instead, the intermediate aggregation of parameters is first conducted at the edge, e.g., with the aid of cluster heads, such as, base stations or other devices. Then, communication with the central controller is established only when there is a need for global aggregation.

Similarly, [19] proposes that mobile devices can form clusters to participate in self-organized FL. Besides improving communication efficiency, it reduces the likelihood that the training fails due to the unexpected malfunctioning of the central controller. In [19], the cluster head selection algorithm is studied, and the cluster head is chosen based on its social relationship with the other devices. The studies in [4], [12] also propose a collaborative FL, in which the device-to-device (D2D) and device-to-edge (D2E) communication is leveraged to ensure the efficient transmission of model parameters to the model owner.

While the aforementioned studies validate the feasibility of HFL and highlight the advantages of conducting FL in a decentralized manner, thereby reducing the reliance on a central controller, resource allocation and incentive mechanism design have not been addressed in the HFL framework. For example, in a network, the workers are free to join any cluster. In addition, they need to receive some rewards as a compensation for the resources expended during training. Given that the worker decisions are dynamic, it is important to develop a framework that can potentially serve to guide each cluster head's incentive design. As such, in this paper, we propose the evolutionary game to incorporate and analyze the dynamics of cluster selection in HFL.

In addition, the services of the cluster head have to be compensated. Given the competing model owners within a network, we utilize a deep learning based auction mechanism [20] to match the cluster head according to the varying valuations of the model owner.

3 System Model and Problem Formulation

3.1 System Model

We consider a network that consists of a set $\mathcal{N} = \{1, \ldots, n, \ldots, N\}$ of N workers. There exist L distinct model owners, each of which desires to develop an AI model for its own purposes, e.g., traffic crowdsensing [21] or location based recommendation [22]. Given the communication constraints of individual workers, the central controller-reliant conventional FL architecture is prone to high device dropout rates [12]. Moreover, we have J cluster heads, e.g., base stations, employed across the network to facilitate the HFL task, the set of which is denoted by $\mathcal{J} = \{1, \ldots, j, \ldots, J\}$. Each worker can choose to associate with any one $j \in \mathcal{J}$ cluster head.

Without loss of generality, each cluster head $j \in \mathcal{J}$ can only serve a single model owner $l \in \mathcal{L}$ at a time and facilitates the HFL process for a cluster j of $p_j N$ workers, where $\sum_j^J p_j = 1$.

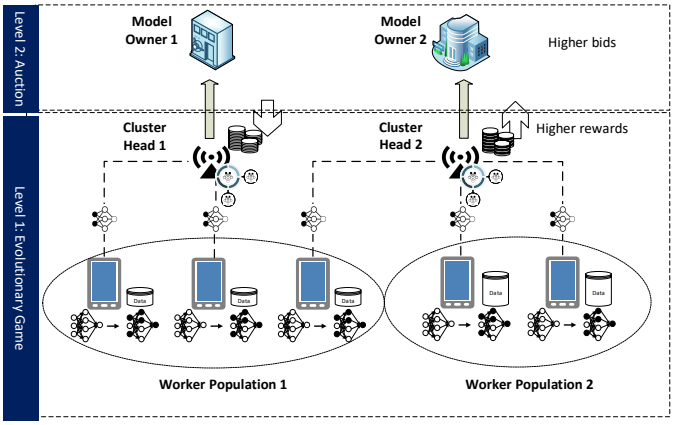


Fig. 1: An illustration of our system model involving two populations of workers. Within each population, all workers have the same data quantities. The workers may choose to join either cluster head. The dynamics is modeled at the first level using the evolutionary game. Cluster head 2 eventually has higher data coverage across the network given that it offers the workers higher rewards. Thus, the services of cluster head 2 are valued higher at the auction.

In HFL, a cluster head j first receives an initial global model, i.e., parameters denoted by the vector $\boldsymbol{w}$, from a model owner that has chosen its services. It then relays the global model to its workers. The FL model training takes place over K iterations to minimize the global loss $F^K(\boldsymbol{w})$ where K is stipulated by the model owner. Each k^{th} iteration, $k \in \{1, \ldots, k, \ldots, K\}$, consists of three steps [17] namely:

1) *Local Computation*: Each worker trains the received global model $\boldsymbol{w}^{(k)}$ locally.
2) *Wireless Transmission*: The worker transmits the model parameter update to its cluster head j.
3) *Intermediate Model Parameter Update*: All parameter updates received from its $p_j N$ workers are aggregated by the cluster head to derive an updated intermediate model $\boldsymbol{w}_j^{(k+1)}$, which is then transmitted back to the worker for the $(k+1)^{th}$ training iteration.

After K iterations, the intermediate model $\boldsymbol{w}_j^{(K)}$ is transmitted to the model owner for global aggregation with the intermediate parameters collected from other clusters. A new set of updated global parameters is derived by the model owner which sends it out to its cluster heads for another round of local model training.

In this paper, we assume that the cluster heads are predetermined, e.g., through the cluster head selection algorithms based on energy efficiency [23], [24], trust [25], and social effects [26]. Instead, we focus our study on a two-level optimization problem as follows: i) in the *lower level*, we adopt an evolutionary game approach to study the dynamics of cluster selection by the workers to derive the dynamics of the composition of each cluster, and ii) in the *upper level*, we adopt a deep learning based auction to value each cluster head's worth to a model owner.

3.2 Lower-Level Evolutionary Game

In the lower level, the cluster formation is derived given J cluster heads. Each cluster head has the objective of attracting more workers to join its cluster, since this ensures that the cluster will have a larger *data coverage* across the network. With a larger data coverage, the cluster value is increased, e.g., due to the fact that the model performance increases with more training data [16].

To encourage the participation of the workers, each cluster head offers a reward pool to be shared by all workers in the cluster. The reward to be distributed to each worker is based on the proportion of the worker's contributions in the cluster, i.e., its data quantity relative to the total amount of data in the cluster. On one hand, a cluster that offers a high reward pool is more attractive to workers. On the other hand, when more workers join that cluster, the reward pool has to be shared among a larger number of workers. Thus, the worker decisions as to which cluster they choose to join are interrelated with the decisions of other workers. We adopt an evolutionary game theory approach to model the dynamics of cluster formation.

3.3 Upper-Level Deep Learning Based Auction

The lower-level evolutionary game gives us the data coverage of each cluster. For example, a cluster head that has greater data coverage will be deemed more valuable to an FL model owner, since the model performance, e.g., inference accuracy, is improved [27]. However, recall from Section 3.1 that there exists more than one model owner in the network seeking to secure the services of the cluster head to facilitate the HFL. In consideration of the competition among model owners, we adopt an auction mechanism in which L model owners bid for the services of each cluster head $j \in \mathcal{J}$. Specifically, we utilize the deep learning based auction mechanism which has the attractive properties of ensuring the truthfulness of the bidders, as well as revenue maximization for the seller (i.e., cluster head), as discussed in Section 5.

4 Lower-Level Evolutionary Game

4.1 Evolutionary Game Formulation

In the following, we formulate the cluster selection as an evolutionary game:

- *Players:* The set of workers $\mathcal{N} = \{1, \ldots, n, \ldots, N\}$ in the FL network are the players of the evolutionary game. For clarity, we use the terms "workers" hereinafter.
- *Population:* We partition the workers into $\mathcal{M} = \{1, \ldots, m, \ldots, M\}$ populations based on the data quantities[2] that each worker owns or the data coverage proportion of the worker using conventional data mining tools, e.g., k-means. The data coverage proportion can be a reflection of the workers' market share in the case when organizations are considered, e.g., based on the proportion of users a bank has, or usage frequency in the case in which individuals are considered, e.g., based on how often the worker uses an IoT device. Each worker of population m owns d_m training data samples, whereas the total data quantity in a population is denoted D_m. We denote the number of workers in each population as $n_m = p_m N$ where $p_m \in [0, 1]$ and $\sum_{m=1}^{M} p_m = 1$. In other words, we have M populations of workers in the network, where all workers within a population own the same number of data samples.
- *Strategy:* The strategy of each worker in population m is the selection of a cluster to join so as to achieve utility maximization. The strategy space of each worker n in population m is denoted by $\mathcal{S}_n^{(m)} = \{a_{n,1}^{(m)}, \ldots, a_{n,j}^{(m)}, \ldots, a_{n,J}^{(m)}\}$ in which $a_{n,j}^{(m)}$ is a binary variable where $a_{n,j}^{(m)} = 1$ represents that the worker n in population m chooses the cluster j, whereas $a_{n,j}^{(m)} = 0$ indicates otherwise.
- *Population Share:* We denote the fraction of population m that selects strategy j, i.e., cluster j, by $x_j^{(m)}$ where $\sum_{j=1}^{J} x_j^{(m)} = 1$. The population state [29] is denoted by the vector $\mathbf{x}^{(m)} = [x_1^{(m)}, \ldots, x_j^{(m)}, \ldots, x_J^{(m)}]^T \in \mathbb{X}$.
- *Payoff:* The expected payoff of each worker is determined by its net utility, which is the difference between the reward that it derives from joining a cluster, and the cost of participating in the FL model training. We further discuss payoffs in Section 4.2.

As an illustration, the system model and game formulation are illustrated in Fig. 1, for the case of two populations. Each worker in population 1 has fewer data samples than each worker in population 2. Each worker in the population can also choose to join either cluster heads. Eventually, each cluster head is associated with a certain level of data coverage, and has its worth evaluated using the auction mechanism discussed in Section 5.

Note that in this paper, we consider that each worker can only join a single cluster, as the worker device is unable to support two instances of model training in parallel. However, our model can be extended to the situation in which each worker can join more than one cluster at a time. In this case, the worker can be modeled to decide, in an evolutionary process, on how its limited resources can be divided among the model owners. Then, $x_j^{(m)} \in [0, 1]$ is denoted to represent the share of resources a worker from population m contributes to cluster head j, where $\sum_{j=1}^{J} x_j^{(m)} = 1$.

4.2 Worker Utility and Replicator Dynamics

The rewards derived by workers of population m, from joining a cluster j for K iterations of FL model training, is given by:

$$p_j^{(m)} = \alpha_j \frac{x_j^{(m)} D_m}{\sum_{m=1}^{M} x_j^{(m)} D_m} + R_j, \tag{1}$$

where α_j is the reward pool to be divided across all workers in cluster j based on their data contributions, $\frac{x_j^{(m)} D_m}{\sum_{m=1}^{M} x_j^{(m)} D_m}$

2. For simplicity, we only consider data quantity as the criterion for clustering here. Note that we may cluster the workers based on other important factors [28] such as geolocation (as a proxy for data distribution) and reputation metrics (as a proxy for quality of contribution). Our work can easily be extended to cover such measures.

is the share of rewards based on the worker's data contribution[3], and R_j is a fixed reward offered to workers in cluster j based on the compensation for the workers' participation costs.

The cost of workers of population m incurred from joining a cluster j is given by an addition of the computation and communication cost over the K iterations of the model training. The computation cost is as follows:

$$c_m^{cmp} = \eta \kappa \theta_m f_m^2, \tag{2}$$

where η is the unit cost of energy consumption, κ is the coefficient of the value that is determined by the circuit architecture of the worker Central Processing Unit (CPU) [30], θ_m is the number of CPU cycles required to perform local computation, i.e., model training, and f_m^2 refers to the computation capability of the worker which is determined by the clock frequency of the worker CPU. Without loss of generality, we have the computation cost held constant throughout for all workers, i.e., $c_m^{cmp} = c^{cmp}, \forall m \in \mathcal{M}$. To account for the varying computation and communication capabilities, we can straightforwardly extend our work to include multiple heterogeneous populations with varying computation capabilities. For example, if there are Λ varying computation costs, we can have ΛM populations accordingly.

The main benefit of HFL is that devices with communication constraints are able to participate in FL. To facilitate the communication of parameters, the cluster head, e.g., a base station, distributes communication resource blocks to all participants within the cluster. On the one hand, clusters with more communication resources are attractive to participants since the participants can benefit from a higher achievable uplink transmission rate. On the other hand, with more participants attracted to join the cluster, the increased competition for resource blocks leads to more congestion. As such, following [31], we model the disutilities arising from network congestion effects as

$$c_{m,j}^{com}(t) = \zeta_j \left(\sum_{m \in \mathcal{M}} x_j^{(m)}(t) \right)^2, \tag{3}$$

where ζ_j is the congestion coefficient determined by the resource constraints of the cluster head [31], whereas $\left(\sum_{m \in \mathcal{M}} x_j^{(m)}\right)^2$ represents the usage profile across populations in the network for a particular cluster. Specifically, a cluster head with more resources will have a lower congestion coefficient. Moreover, workers in a less-populated cluster experience less congestion.

The total cost of participation incurred by worker n_m (of population m) in cluster j is obtained as

$$c_j^{(m)}(t) = c^{cmp} + c_{m,j}^{com}(t). \tag{4}$$

At time t, the net utility that the workers in class m receive for their participation in cluster j is:

$$u_j^{(m)}(t) = \mathcal{U}\left(p_j^{(m)}(t) - c_j^{(m)}(t) \right), \tag{5}$$

3. In this work, we make a simplifying assumption that the workers are not malicious. In practice, the parameter contributions of the workers can be randomly verified, e.g. detecting model parameters with outlying magnitudes, and malicious workers with anomalous contributions can be penalized from participating in the future FL training.

where we assume $\mathcal{U}(\cdot)$ to be a linear utility function indicating the risk neutrality of workers without loss of generality [8], [16].

Accordingly, at time t, the average utility of workers in population m across all J clusters is

$$\bar{u}^{(m)}(t) = \sum_{j=1}^{J} x_j^{(m)} u_j^{(m)}(t). \tag{6}$$

In practice, information regarding the utility derived from joining different clusters can be exchanged and compared among workers within the network [32]. Workers may thus switch from one cluster to another to seek higher utilities. To capture the dynamics of the cluster selection and model the strategy adaptation process, we define the replicator dynamics [33] as follows:

$$\dot{x}_j^{(m)}(t) = f_j^{(m)}(\mathbf{x}^{(m)}(t)) = \delta x_j^{(m)}(t) \left(u_j^{(m)}(t) - \bar{u}^{(m)}(t) \right), \quad \forall m \in \mathcal{M}, \forall j \in \mathcal{J}, \forall t, \tag{7}$$

where δ refers to the positive learning rate of the population that controls the speed at which workers adapt their strategies. For example, in a network with communication bottlenecks [32] or negative network effects [26], the learning rate tends to be slower as the worker requires more time to collect the information required to change its decision.

The replicator dynamics is a series of ordinary differential equations (ODEs) with the initial condition $\mathbf{x}^{(m)}(0) \in \mathbb{X}$ [34]. Specifically, based on the replicator dynamics, workers in population m can adapt their strategy, i.e., switch from one cluster to another if their utilities are lower than the expected utility. The *evolutionary equilibrium* is a fixed point in (7) that is reached in a particular t when $\dot{x}_j^{(m)}(t) = 0, \forall m \in \mathcal{M}, \forall j \in \mathcal{J}$. In other words, at the evolutionary equilibrium, workers from all clusters derive an identical payoff such that there is no longer a need to deviate from their prevailing clusters.

In a dynamic system, it is of paramount importance that the equilibrium is stable and unique. In terms of stability, an evolutionary equilibrium remains to be $\dot{x}_j^{(m)}(t) = 0$ for all time periods after the equilibrium is first reached. In terms of uniqueness, the same evolutionary equilibrium is reached regardless of the initial conditions. In Section 4.3, we prove the existence, uniqueness, and stability of the solution to (7).

4.3 Existence, Uniqueness, and Stability of the Evolutionary Equilibrium

In this section, we first prove the boundedness of (7) in Lemma 1.

Lemma 1. *The first order derivatives of $f_j^{(m)}(\mathbf{x}^{(m)}(t))$ with respect to $x_v^{(m)}(t)$ is bounded for all $v \in J$.*

Proof. For ease of presentation, we omit the notations of t and (m) in this proof. The first order derivative of $f_j(\mathbf{x})$ with respect to x_v, where $v \in \mathcal{J}$, is given by

$$\frac{\mathrm{d}f_j(\mathbf{x})}{\mathrm{d}x_v} = \delta \left[\frac{\mathrm{d}x_j}{\mathrm{d}x_v} (u_j - \bar{u}) + x_j \left(\frac{\mathrm{d}u_j}{\mathrm{d}x_v} - \frac{\mathrm{d}\bar{u}}{\mathrm{d}x_v} \right) \right]. \tag{8}$$

For ease of notation, denote $A(\mathbf{x}_j) = \sum_{m=1}^{M} x_j D_m$. Next, we derive $\frac{\mathrm{d}u_j}{\mathrm{d}x_v}$ as follows:

$$\frac{\mathrm{d}u_j}{\mathrm{d}x_v} = \alpha_j \left(\frac{\frac{\mathrm{d}x_j}{\mathrm{d}x_v} D_m}{A(\mathbf{x}_j)} - \frac{x_j D_m^2}{A^2(\mathbf{x}_j)} \right) - 2\zeta_j \left(\sum_{m \in \mathcal{M}} x_j \right). \quad (9)$$

It follows that $\left|\frac{\mathrm{d}u_j}{\mathrm{d}x_v}\right|$ and thus $\left|\frac{\mathrm{d}\bar{u}}{\mathrm{d}x_v}\right|$ are clearly bounded $\forall v \in \mathcal{J}$. Therefore, this represents that $\left|\frac{\mathrm{d}f_j(\mathbf{x})}{\mathrm{d}x_v}\right|$ is bounded. This proof also applies to all M populations and T time periods. □

Theorem 1. *For any initial condition $\mathbf{x}^{(m)}(0) \in \mathbb{X}$, there exists a unique evolutionary equilibrium to the dynamics defined in (7).*

Proof. From Lemma 1, we have proven that the replicator dynamics $f_j^{(m)}(\mathbf{x}^{(m)}(t))$ is bounded and continuously differentiable $\forall \mathbf{x}^{(m)}(t) \in \mathbb{X}, \forall m \in \mathcal{M}, \forall j \in \mathcal{J}, \forall t$. Therefore, the maximum absolute value of its partial derivative given in (8) is a Lipschitz constant. According to the Mean Value Theorem, there exists a constant c between $x_1^{(m)}(t)$ and $x_2^{(m)}(t)$ such that $\frac{\left|f_j^{(m)}(x_1^{(m)}(t)) - f_j^{(m)}(x_2^{(m)}(t))\right|}{(x_1^{(m)}(t) - x_2^{(m)}(t))} = \frac{\mathrm{d}f_j(c)}{\mathrm{d}x_v}$. Therefore, we can define the relation

$$\left|f_j^{(m)}(x_1^{(m)}(t)) - f_j^{(m)}(x_2^{(m)}(t))\right| \leq \Gamma \left|x_1^{(m)}(t) - x_2^{(m)}(t)\right|, \quad \forall (x_1^{(m)}, x_2^{(m)}) \in \mathbb{X}, \forall m \in \mathcal{M}, \forall t.$$

where $\Gamma = \max\left\{\left|\frac{\mathrm{d}f_j(c)}{\mathrm{d}x_v}\right|\right\}$. Following the Lipschitz condition [35], this implies that the replicator dynamics, i.e., an initial value problem with $\mathbf{x}^{(m)}(0) \in \mathbb{X}$, in (7) has a unique solution $x_j^{(m)^\star} \in \mathbb{X}$. □

Next, we prove the stability of the evolutionary equilibrium in the following theorem.

Theorem 2. *For any initial condition $\mathbf{x}^{(m)}(0) \in \mathbb{X}$, the evolutionary equilibrium to the dynamics defined in (7) is stable.*

Proof. We define the Lyapunov function

$$G(\mathbf{x}^{(m)}(t)) = \left(\sum_{m=1}^{M} \sum_{j=1}^{J} x_j^{(m)}(t) \right)^2, \quad (10)$$

which is positive definite since:

$$G(\mathbf{x}^{(m)}(t)) \begin{cases} = 0 & \text{if } \mathbf{x}(t) = \mathbf{0} \\ > 0 & \text{otherwise.} \end{cases}$$

Taking the first-order derivative with of $G(\mathbf{x}^{(m)}(t))$ with respect to t,

$$\frac{\mathrm{d}G(\mathbf{x}^{(m)}(t))}{\mathrm{d}t} = 2 \left(\sum_{m=1}^{M} \sum_{j=1}^{J} x_j^{(m)}(t) \right) \left(\sum_{m=1}^{M} \sum_{j=1}^{J} \dot{x}_j^{(m)}(t) \right). \quad (11)$$

Note that at any point of time $\sum_{m=1}^{M} \sum_{j=1}^{J} x_j^{(m)}(t) = M$. Thus, the replicator dynamics have to equate to zero for this to hold, i.e., the net movements and strategy adaptations across clusters are zeroed out in order for the population to remain constant. Specifically,

$$\sum_{m=1}^{M} \sum_{j=1}^{J} \dot{x}_j^{(m)}(t) = 0, \forall t. \quad (12)$$

Therefore, (12) ensures that $\frac{\mathrm{d}G(\mathbf{x}^{(m)}(t))}{\mathrm{d}t} = 0$, which satisfies the Lyapunov conditions required for stability, as defined in the Lyapunov's second method for stability [36]. □

As such, we have proven the uniqueness and stability of the evolutionary equilibrium.

Next, we discuss the procedures to derive the equilibrium cluster data coverage based on the replicator dynamics in (7). In contrast to the population evolution algorithm [32] which involves the intervention of a centralized controller, e.g., in disseminating information of potential payoffs that can be derived from joining a particular cluster, we consider the implementation of a decentralized cluster selection algorithm in Algorithm 1.

At the initialization phase, workers in each population m are randomly assigned to j clusters, where $m \in \mathcal{M}$, $j \in \mathcal{J}$. At each time period t, the workers compute their utilities and the average utility of workers in the population. Note that in practice, the workers may not have the complete information of all workers belonging to the same population in a large network. As such, its knowledge of the average utility in the population is based on the worker's "best guess", i.e., the expected average utility. This procedure is simply a comparison between (i) the worker's own utility from joining a particular cluster j, i.e., $u_j^{(m)}(t)$ and (ii) the expected average utilities of other workers from the same population which have chosen to join other clusters, i.e., $E(\bar{u}^{(m)}(t))$. Thereafter, the evolution of population state can be derived following the replicator dynamics.

The output of Algorithm 1 is the population state that is observed after t_{max} iterations. Then, we are eventually able to derive the data coverages of the cluster head, i.e., the proportion of data across the network that each cluster head can cover, as follows:

$$D_j = \sum_{m=1}^{M} x_j^{(m)}(t_{max}) D_m. \quad (13)$$

5 Deep Learning Based Auction for Valuation of Cluster Head

5.1 Auction Formulation

Based on the cluster formations from the evolutionary game, we are able to derive the data coverage D_j of the cluster head $j \in \mathcal{J}$.

As each cluster head can only offer its services to a single model owner, i.e., the workers' participation in the FL model training, the model owners need to compete for the services of the cluster heads. Each model owner $l \in \mathcal{L}$ has different preference for the accuracy of their models, e.g., applications for accident warning and prediction [37] require higher accuracy than the route planning and navigation systems. Following the work in [38], the FL model accuracy A_l of model owner l, can be expressed as a power law function that is denoted as follows:

$$A_l(\mu_l) = \sigma - v\mu_l^{-r}, \quad (14)$$

where σ, v, and r are calibrable parameters depending on the model to be trained. μ_l is the data coverage required by model owner l to achieve its required model accuracy, σ is

Algorithm 1 Cluster Selection for HFL

Input: Worker and cluster characteristics
Output: $D_j, \forall j \in \mathcal{J}$

Initialization: Workers in population m each assigned to a random cluster
while $t < t_{max}$ **do**
  for $m \in \mathcal{M}$ **do**
    Payoff Computation
    for $j \in \mathcal{J}$ **do**
      Derive $u_j^{(m)}(t) = p_j^{(m)}(t) - c_j^{(m)}(t)$
    end for
    Compute $E(\bar{u}^{(m)}(t)) = E(\sum_{j=1}^{J} x_j^{(m)} u_j^{(m)}(t))$
    Cluster Selection
    for $j \in \mathcal{J}$ **do**
      Derive $\dot{x}_j^{(m)}(t) = \delta x_j^{(m)}(t) \left(u_j^{(m)}(t) - \bar{u}^{(m)}(t)\right)$ and $x_j^{(m)}(t)$
    end for
  end for
end while
for $j \in \mathcal{J}$ **do**
  Compute $D_j = \sum_{m=1}^{M} x_j^{(m)}(t_{max}) D_m$
end for

the upper bound of the accuracy that can be derived from historical data, whereas v and r are the fixed parameters of the function.

In general, when the requirement for data coverage of the model owner μ_l is larger, the model owner has more incentive to pay a higher price for the services of the cluster heads which has more data coverage. In contrast, a model owner that already has some pre-existing training data d_l will have less incentive to bid for the services of a cluster head. Therefore, the valuation b_l of model owner l for the services offered by the cluster heads can be expressed as $b_l = \mu_l - d_l$.

In order to maximize the revenue of the cluster heads and to ensure that the services from the cluster heads are allocated to the model owners that value them most, we model the allocation problem as multiple rounds of single-item auctions. In this auction, the cluster heads are the sellers, i.e., auctioneers, while the model owners are the buyers, i.e., bidders. The cluster head with the highest amount of data coverage is the first to auction its services to the model owners. All model owners submit their bids to compete for the service. Then, the cluster head collects the bid profile $(b_1, \ldots, b_l, \ldots, b_L)$ to decide on the winning model owner $l^\star$ and the corresponding payment price $\theta_{l^\star}$. After each round of auction, the winning model owner has higher data coverage, i.e., higher d_l. Thus, its valuation in the next round of auction naturally has to be updated and decreases. The auction ends when all cluster heads have been allocated to the model owners. Note that the model owners may participate in more than one round of auction to fulfill their data coverage requirement, e.g., if the data coverage that single cluster has is insufficient to fulfill its needs.

Accordingly, the utility of the model owner in each round of the auction is as follows

$$u_l = \begin{cases} b_l - \theta_{l^\star} & \text{if the model owner wins the bid,} \\ 0 & \text{otherwise.} \end{cases} \quad (15)$$

An optimal auction [39] has two characteristics:

1) *Individual Rationality (IR):* By participating in the auction, the model owners receive non-negative payoff, i.e., $u_l \geq 0$.
2) *Incentive Compatibility (IC):* There is no incentive for the model owners to submit bids other than their true valuations, i.e., the bidders always bid truthfully.

In each round of the auction, in order to determine the payment price $\theta_{l^\star}$ of the winning model owner, traditional auction schemes such as the first-price auction and second-price auction (SPA) may be adopted. However, each of these traditional auction schemes has its own drawbacks.

The traditional first-price auction, in which the highest bidder pays the exact bid it submits, maximizes the revenue of the seller but does not ensure that the bidders submit their true valuations. On the other hand, the SPA, in which the highest bidder pays the price offered by the second highest bidder, ensures that the bidders submit their true valuations, i.e., ensures IC, but does not maximize the revenue of the seller. Therefore, in order to ensure that both conditions of truthfulness and revenue maximization of the seller are satisfied, we design an optimal auction using the Deep Learning approach with reference to the study in [20].

5.2 Deep Learning Based Auction for Valuation of Cluster Heads

In this section, we illustrate the neural network architecture for the design of an optimal auction. Following the procedure in [20], we describe the neural network architecture (Fig. 2) which renders the design of an optimal auction. Then, we elaborate on the proposed implementation of multiple round single-item auctions for the valuation of the services of cluster heads.

By adopting the SPA scheme to determine the payment price of the winning model owner, the revenue of the cluster head is not maximized, especially when the bid of the second highest bidder is low. Thus, in order to maximize the revenue of the cluster heads, the monotonically increasing functions are applied to the bids of the model owners to transform the bids into transformed bids, which are used to determine the allocation and the corresponding payment of the model owners in the network. The input bids and the transformed bids of model owner l are denoted as b_l and $\bar{b}_l$ respectively. The transform function for the bids submitted by model owner l is denoted as ϕ_l. In order to determine the allocation and conditional payment of the model owners, the SPA with zero reserve price (SPA-0) is applied to transform the bids. The reserve price is the minimum price that the cluster head requires to offer its service. The SPA-0 allocation rule and the SPA-0 payment rule of the model owner l are represented by $g_l^0(\bar{\boldsymbol{b}})$ and $\theta_l^0(\bar{\boldsymbol{b}})$, respectively, where $\bar{\boldsymbol{b}}$ is the vector of the transformed bids. The SPA-0 allocation rule $g_l^0(\bar{\boldsymbol{b}})$ determines the winning model owner which has the highest bid if the bid is greater than zero.

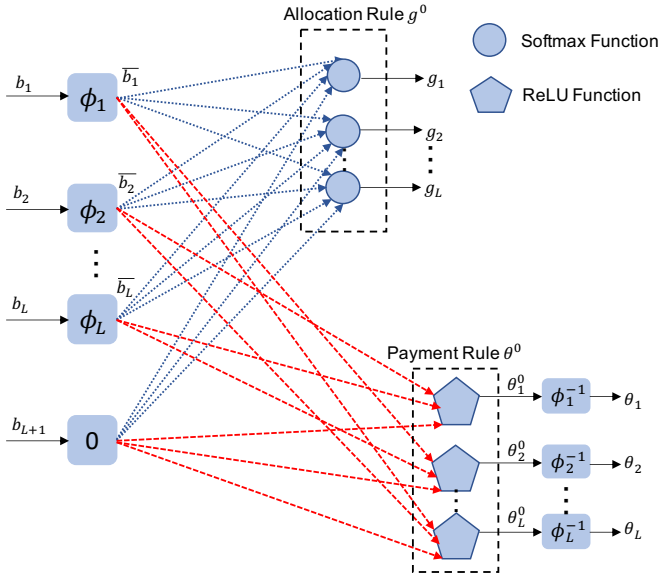


Fig. 2: Neural Network Architecture for the Optimal Auction.

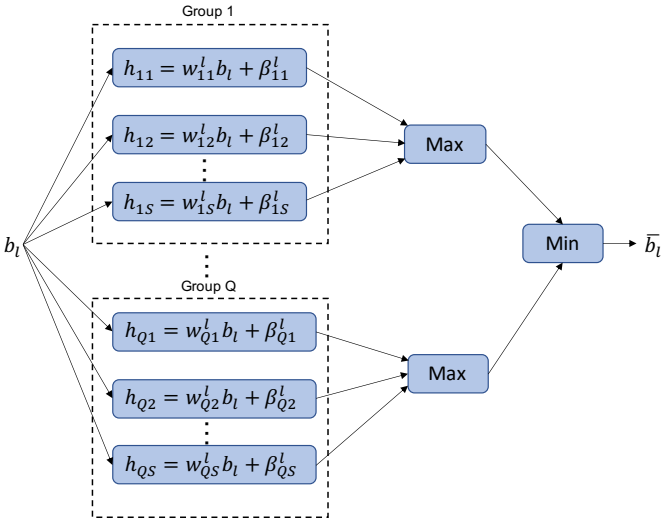


Fig. 3: Monotone Transform Functions.

The SPA-0 payment rule $\theta_l^0(\bar{\boldsymbol{b}})$ determines the conditional payment price θ_l of model owner l by applying the inverse transform function which is represented by ϕ_l^{-1}.

Theorem 3. *For any set of strictly monotonically increasing function $\{\phi_1, \ldots, \phi_l, \ldots, \phi_L\}$, an auction which is defined by the allocation rule $g_l = g_l^0 \circ \phi_l$ and the payment rule $\theta_l = \phi_l^{-1} \circ \theta_l^0 \circ \phi_l$ satisfies the properties of IC and IR, where g^0 and θ^0 represent the SPA-0 allocation rule and the SPA-0 payment rule, respectively [20].*

Based on the theorem, for any choices of strictly monotonically increasing transform functions, the proposed auction with the allocation rule g_l and conditional payment rule θ_l satisfy the characteristics of the optimal auction, i.e., IR and IC. Therefore, the monotone transform functions are used in the neural network to ensure the IR and IC properties of the auction. In addition to this, the cluster heads want to maximize their revenues. Based on the allocation and the conditional payment rules, the revenue of the cluster head is determined. In particular, the revenue of the cluster head is equivalent to the payment price of the winning model owner. Thus, the objective of the cluster heads is to maximize their individual revenues while fulfilling the properties of IR and IC of the optimal auction. In order to do so, the neural network architecture learns the appropriate transform functions for the optimal auction to minimize the loss function which is defined as the expectation of the negated revenue of the cluster head. The minimization of the loss function is equivalent to the maximization of the revenue of the cluster head. With this, the optimal auction design based on the neural network architecture maximizes the revenues of the cluster heads while satisfying the necessary and sufficient conditions for IC and IR.

The algorithm for the implementation of the optimal auction based on the neural network architecture is illustrated in Algorithm 2.

In the following, we discuss the three important functions in the neural network architecture

1) the monotone transform function ϕ_l,
2) the allocation rule g_l,
3) the conditional payment rule θ_l.

Algorithm 2 Algorithm for Deep-Learning Based Optimal Auction.

Input: Set of cluster heads $\mathcal{J} = \{1, \ldots, j, \ldots, J\}$, bids of model owners $b^i = (b_1^i, \ldots, b_l^i, \ldots, b_L^i)$
Output: Revenue of the cluster heads

while $\mathcal{J} \neq \emptyset$ **do**
  Identify the cluster head the highest data coverage, D_j
  Initialization: $\mathbf{w} = [w_{qs}^l] \in \mathbb{R}_+^{I \times QS}$, $\boldsymbol{\beta} = [\beta_{qs}^l] \in \mathbb{R}^{I \times QS}$
  Deep-Learning Based Optimal Auction:
  while Loss function $\hat{R}(\mathbf{w}, \boldsymbol{\beta})$ is not minimized **do**
    Compute the transformed bids $\bar{b}_l^i = \theta_l(b_l^i) = \min_{q \in \mathcal{Q}} \max_{s \in \mathcal{S}} (w_{qs}^l b_l + \beta_{qs}^l)$
    Compute the allocation probabilities $g_l(\bar{\mathbf{b}}) = softmax_l(\bar{b}_1, \ldots, \bar{b}_l, \ldots, \bar{b}_{L+1}; \gamma)$
    Compute the SPA-0 payments $\theta_l^0(\bar{\mathbf{b}}) = ReLU(\max_{s \neq l} \bar{b}_s)$
    Compute the conditional payments $\theta_l = \phi_l^{-1}(\theta_l^0(\bar{\mathbf{b}}))$
    Compute the loss function $\hat{R}(\mathbf{w}, \boldsymbol{\beta})$
    Update the network parameters $\mathbf{w}$ and $\boldsymbol{\beta}$ using the SGD solver
  end while
  Update the data coverage of the winning model owner $d_l^{new} = d_l^{old} + D_j$
  Update the valuation of the winning model owner $b_l = \mu_l - d_l$
  Remove the cluster head from set $\mathcal{J}$
end while
return The revenue gain by the cluster heads

5.3 Monotone Transform Functions

In the auction, the valuation, i.e., bid b_l of each model owner l is the input to the transform function ϕ_l. The transform function maps the input to its transformed bid, $\bar{b_l} = \phi_l(b_l)$. Each transform function ϕ_l is modelled as a two-layer feed forward network which consists of the min and max operators over several linear functions, as shown in Fig. 3. There are Q groups of S linear functions $h_{qs}(b_l) = w^l_{qs} b_l + \beta^l_{qs}$, $\mathcal{Q} = \{1, \ldots, q, \ldots, Q\}$, $\mathcal{S} = \{1, \ldots, s, \ldots, S\}$, $w^l_{qs} \in \mathbb{R}^+$ and $\beta^l_{qs} \in \mathbb{R}$ are the positive weight and bias respectively. With these linear functions, the transform function ϕ_l of each model owner l is defined as follows:

$$\phi_l(b_l) = \min_{q \in \mathcal{Q}} \max_{s \in \mathcal{S}} (w^l_{qs} b_l + \beta^l_{qs}). \tag{16}$$

Based on the parameters for the forward transform function ϕ_l, the inverse function ϕ_l^{-1} can be derived as follows:

$$\phi_l^{-1}(y) = \max_{q \in \mathcal{Q}} \min_{s \in \mathcal{S}} (w^l_{qs})^{-1} (y - \beta^l_{qs}). \tag{17}$$

5.4 Allocation Rule

The allocation rule in the neural network architecture is based on the SPA-0 allocation rule. In particular, the data coverage D_j of the cluster head j is allocated to the model owner with the highest transformed bid if the transformed bid is more than zero. Otherwise, the cluster head does not sell its service to any model owner. In order to model the competition among the model owners, we use a *softmax* function on the vector of transformed bids $\bar{\mathbf{b}} = (\bar{b}_1, \ldots, \bar{b}_l, \ldots, \bar{b}_L)$ and the additional dummy input $\bar{b}_{L+1} = 0$ in the allocation network. The output of the *softmax* function is a vector of allocation probabilities, which is represented by $\mathbf{g} = (g_1, \ldots, g_l, \ldots, g_L)$. The *softmax* function used in the neural network architecture is defined as follows:

$$\begin{aligned} g_l(\bar{\mathbf{b}}) &= softmax_l(\bar{b}_1, \ldots, \bar{b}_l, \ldots, \bar{b}_{L+1}; \gamma) \\ &= \frac{e^{\gamma \bar{b}_l}}{\sum_{l=1}^{L+1} e^{\gamma \bar{b}_l}}, \ \gamma > 0, \forall l \in \mathcal{L}. \end{aligned} \tag{18}$$

The parameter γ in the *softmax* function measures the quality of the approximation where the higher the γ, the more accurate the approximation of the function. However, a better quality of approximation results in a less smooth allocation function.

5.5 Conditional Payment Rule

The conditional payment rule determines the price θ_l that needs to be paid by the winning model owner l. The conditional payment rule is carried out in two steps. Firstly, the SPA-0 payment θ^0_l for each model owner l is calculated. Specifically, the SPA-0 payment θ^0_l is the maximum of the transformed bids of other model owners and zero which is determined by using the *ReLU* activation unit function as follows:

$$\theta^0_l(\bar{\mathbf{b}}) = ReLU(\max_{s \neq l} \bar{b}_s), \ \forall l \in \mathcal{L}. \tag{19}$$

The $ReLU(x) = \max(x, 0)$ activation function guarantees that the SPA-0 payment of each model owner is non-negative. Secondly, based on the SPA-0 payment θ^0_l, the conditional payment θ_l of model owner l is calculated as follows:

$$\theta_l = \phi_l^{-1}(\theta^0_l(\bar{\mathbf{b}})), \tag{20}$$

where the inverse transform function from Equation (17) is applied to the SPA-0 payment of the model owner l.

5.6 Neural Network Training

The aim of the neural network is to optimize the weights and biases of the linear functions in the neural network such that the loss function is minimized. In the neural network, the loss function is defined as the expectation of the negated revenue of the cluster head. The loss function of the neural network is formulated based on the inputs, i.e., the training dataset and the outputs, i.e., the allocation probabilities and the conditional payments of the model owners. The training dataset of the neural network consists of the bidders' valuation profiles of which the bidders' valuation profile i is denoted as $\mathbf{b}^i = (b^i_1, \ldots, b^i_L)$, $\mathcal{I} = \{1, \ldots, i, \ldots, I\}$ where I is the size of the training dataset. b^i_l is the valuation of model owner l for the data coverage of cluster head is drawn from a valuation distribution function $f_B(b)$. Since the valuation b^i_l of the model owner l depends on the data coverage requirement μ_l and the current amount of data coverage d_l of the model owner, i.e., $b^i_l = \mu^i_l - d^i_l$, the distribution function $f_B(b)$ can be determined based on the distribution of the data coverage requirement, which is represented by $f\mu(\mu)$. In our work, we assume that the data coverage requirement of the model owners follows a uniform distribution, i.e., $\mu \sim U[\mu_{min}, \mu_{max}]$.

The parameters of the monotone transform functions, i.e., weights w^l_{qs} and biases β^l_{qs} are the entries of matrices $\mathbf{w}$ and $\boldsymbol{\beta}$. The matrices are needed to determine the allocation probability and conditional payment of model owner l, which are represented by $g_l^{(\mathbf{w},\boldsymbol{\beta})}$ and $\theta_l^{(\mathbf{w},\boldsymbol{\beta})}$ respectively.

The objective of the training is to find the optimal weight $\mathbf{w}^*$ and bias $\boldsymbol{\beta}^*$ matrices that minimize the loss function of the neural network, i.e., the expectation of the negated revenue of the cluster head j. Specifically, the approximation of the loss function, $\hat{R}$ is defined as follows:

$$\hat{R}(\mathbf{w}, \boldsymbol{\beta}) = -\frac{1}{I} \sum_{l=1}^{I} g_l^{(\mathbf{w},\boldsymbol{\beta})}(\mathbf{b}^i) \theta_l^{(\mathbf{w},\boldsymbol{\beta})}(\mathbf{b}^i). \tag{21}$$

For the optimization of the loss function $\hat{R}(\mathbf{w}, \boldsymbol{\beta})$ over the parameters $(\mathbf{w}, \boldsymbol{\beta})$, a stochastic gradient descent (SGD) solver is used.

6 Performance Evaluation

In this section, we present the performance evaluation of the evolutionary game based cluster formation and deep learning based auction for the valuation of cluster data coverage. Unless otherwise stated, the simulation parameters are as shown in Table 1. Note that we use the terms "cluster" and "cluster heads" interchangeably.

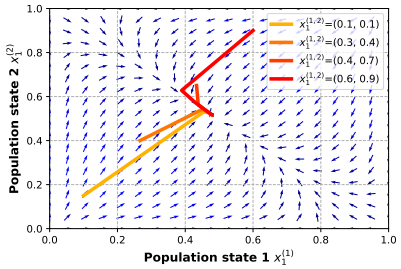


Fig. 4: Phase plane of the replicator dynamics.

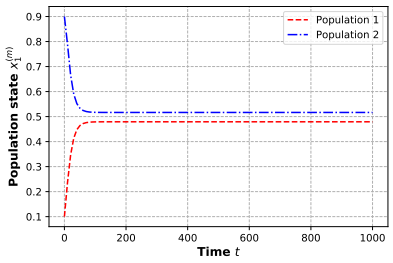


Fig. 5: Evolutionary equilibrium of population states for cluster 1.

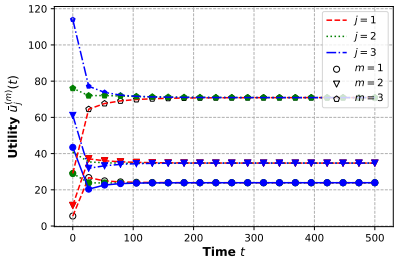


Fig. 6: Evolution of population utilities.

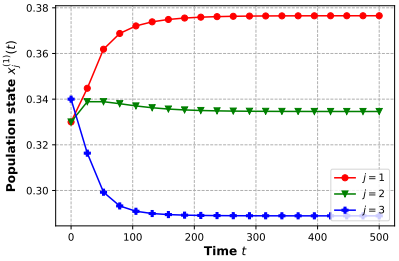


Fig. 7: Evolution of population states for population 1.

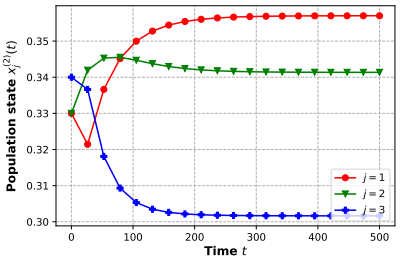


Fig. 8: Evolution of population states for population 2.

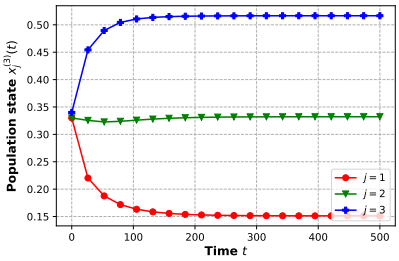


Fig. 9: Evolution of population states for population 3.

TABLE 1: Simulation Parameters [38], [40].

Parameter	Values
Total number of workers in the network N	90
Number of data samples of population m D_m	$[2400, 4800]$
Reward pool offered in cluster α_j	$[100, 300]$
Fixed rewards offered in cluster R_j	80
Congestion coefficient ζ_j	$[10, 20]$
Computation cost c^{cmp}	0.1
Rate of strategy adaptation δ	0.001
Total Number of Model Owners, L	10
Size of Training Dataset, I	1000
Number of Groups for Linear Function Q	5
Linear Functions in Each Group S	10
Learning Rate	0.001
Quality of Approximation	1000, 2000
Range of the model owners' data coverage requirement, μ_l	$\sim U[0.5, 0.9]$, $\sim U[0, 0.4]$

6.1 Lower-level Evolutionary Game

For the first part of our simulations, we analyze the lower-level evolutionary game. We consider a network which consists of 90 workers. The workers have different data quantities that follow a uniform distribution. Using the k-means algorithm, we derive 3 populations[4] of 30 workers each based on the data quantities that they possess. In the first population $m = 1$, each worker has 80 data samples. In the second population $m = 2$, each worker has 100 data samples. In the third population $m = 3$, each worker has 160 samples. Without loss of generality, the data samples that each worker owns are assumed to be characterized by the populations that they belong to. Thus, the populations are arranged in the ascending order based on the data samples each worker has.

Besides, there exist 3 cluster heads in the network with each offering different reward pools α_j, as well as congestion coefficients ζ_j. Recall from Section 4.2 that a higher value of α_j indicates that a cluster offers a larger reward pool for the workers to share, whereas a higher value of ζ_j represents that a cluster head has more limited communication resources. The clusters are arranged in the ascending order based on the reward pool they offer, i.e., cluster 3 offers the highest reward pool to its workers.

Accordingly, in each time period, workers in the populations choose one of the clusters to join. Then, following Algorithm 1, the strategy adaptation is performed and evolved such that the workers evaluate their payoffs and churn to another cluster with higher payoffs with some probabilities. Eventually, the evolutionary equilibrium is achieved.

6.1.1 *Stability and Uniqueness of the Evolutionary Equilibrium*

To demonstrate the uniqueness of the evolutionary equilibrium, i.e., the solution to the replicator dynamics defined in (7), we first derive the phase plane of the replicator dynamics in Fig. 4. For ease of exposition, population 3 is excluded initially. As such, only the first and second populations of workers are considered to choose among the three clusters to join. Figure 4 shows the population states of populations 1 and 2, i.e., the proportion of workers in each population that join cluster 1. We consider varying initial conditions in Fig. 4 and plot the corresponding dynamics. For example, for the first condition, we have 10% of the workers from both

4. For ease of exposition, we use only three populations for comparison. Our model can easily be extended to multiple populations.

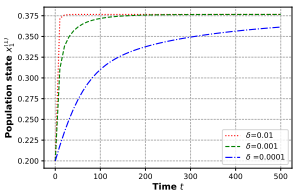


Fig. 10: Evolutionary dynamics under different learning rates.

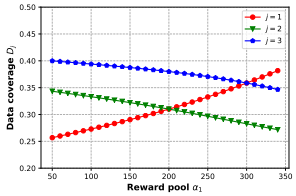


Fig. 11: Data coverage vs. varying fixed rewards in cluster 1.

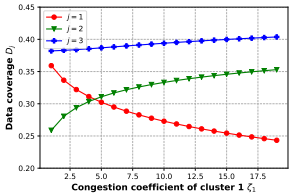


Fig. 12: Data coverage vs. varying congestion coefficient in cluster 1.

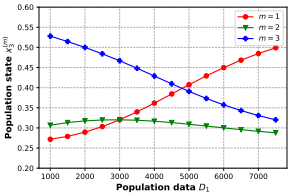


Fig. 13: Population states in cluster 3 vs. varying population data for population 1.

populations choose cluster 1. Clearly, despite varying initial conditions, the evolutionary equilibrium always converges to a unique solution.

To evaluate the stability of this evolutionary equilibrium, we consider that 30% and 70% of workers in populations 1 and 2 respectively are initially allocated to cluster 1. Then, we plot the evolution of population states in Fig. 5. We observe that the proportion of workers from population 2 joining cluster 1 declines as more workers from population 1 joins the cluster. This is due to the division of rewards and congestion effects as more workers join the cluster. Eventually, the evolutionary equilibrium is reached and the population states no longer change.

Next, we consider the situation in which there are three populations and three clusters. We set the initial conditions such that a third of the workers from each population are assigned to each cluster initially. Then, we plot the evolution of utilities in Fig. 6. Specifically, within each population, we plot the utilities derived by workers which have chosen each of the three clusters. We observe that for each population, the utilities derived from choosing the varying clusters eventually converges with time. This implies that an evolutionary equilibrium is reached whereby at the equilibrium, the workers no longer have the incentive to adapt their cluster selection strategies. Moreover, at the equilibrium, workers which belong to population 3 derive the greatest utilities, given that they are compensated for having larger data shares in the clusters.

In Figs. 7-9, we plot the evolution of population states of each population respectively. We observe that cluster 3, which offers the highest reward pool for distribution across workers, has the largest proportion of workers from population 3. The reason is that the workers from population 3 have the largest number of data samples. Hence, they can have the largest shares of the large reward pool if they join cluster 3. In contrast, the lowest proportion of workers from population 1 joins cluster 3 because they have the lowest reward shares. However, there is an upper limit to how many workers can join cluster 3. Even though cluster 3 offers the highest reward pool, the distribution of rewards and congestion effects set in eventually when more workers have joined the cluster. Thus, workers of population 3 may also join clusters 1 and 2 as well when this occurs.

6.1.2 *Evolutionary Equilibrium Under Varying Parameters and Conditions*

In this section, we vary the simulation parameters to study the evolutionary equilibrium under varying conditions. In Fig. 10, we vary the learning rate of the population, which controls the speed of strategy adaptation. Naturally, when the learning rate is low, the evolutionary equilibrium can only be reached after a longer time period. However, we can observe that the stability of the evolutionary equilibrium is not compromised. The speed of convergence depends how fast the workers can observe and adapt their strategies, e.g., they have more accurate information about the system.

In Fig. 11, we vary the reward pool offered by cluster 1 between $[50, 350]$ while keeping the reward pool for clusters 2 and 3 constant at 200 and 300 respectively. Then, we plot the changes in data coverage of each cluster, i.e., how much data coverage a cluster has as a result of worker contribution. Naturally, the data coverage of cluster 1 increases with an increment of the reward pool. The reason is that the cluster is more attractive to workers as shareable rewards increase. We further note that at the points $\alpha_1 = 200$ and $\alpha_1 = 300$, the data coverage for cluster 1 is identical to those of clusters 2 and 3 respectively. The reason is that at these points, the cluster 1 is identical to the corresponding clusters and thus, workers are indifferent between choosing the cluster to join.

In Fig. 12, we vary the congestion coefficient of cluster 1 between $[2, 18]$ while keeping the other clusters' constant. Then, we plot the changes in data coverage of each cluster. We observe that as the congestion coefficient increases, the cluster has lower data coverage. Instead, the workers that used to join the cluster 1 adapt their strategies and churn to clusters 2 and 3. This is given that with a large congestion coefficient, the cluster head has more limited communication resources and can no longer support as many workers without them having to incur larger communication costs, e.g., due to device interference.

In Fig. 13, we vary the data quantities of workers in population 1 while keeping the other populations constant. Then, we plot the population states of all populations with respect to participation in cluster 3. Specifically, the figure shows the proportion of workers from each population which have joined cluster 3 as the data quantities of population 1 vary. Clearly, as the data owned by workers in population 1 increases, more workers from population 1 are able to join cluster 3, which is the cluster with the largest reward pool as mentioned in Section 6.1.1. The reason is

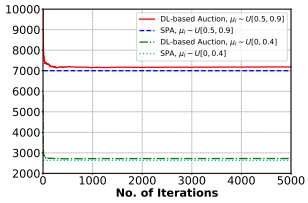


Fig. 14: Revenue of cluster head 1 under different distribution of model owners.

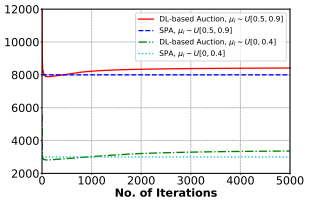


Fig. 15: Revenue of cluster head 2 under different distribution of model owners.

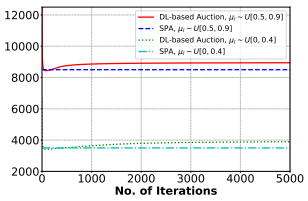


Fig. 16: Revenue of cluster head 3 under different distribution of model owners.

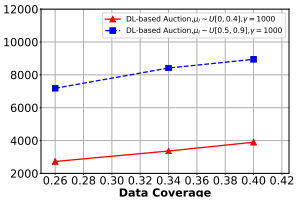


Fig. 17: Revenue vs data coverage of cluster heads.

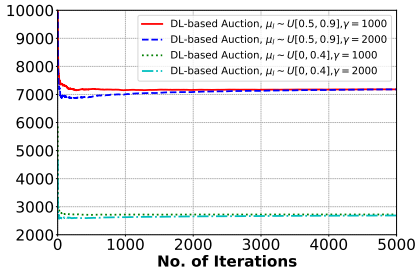


Fig. 18: Revenue of cluster head 1 under different approximation qualities.

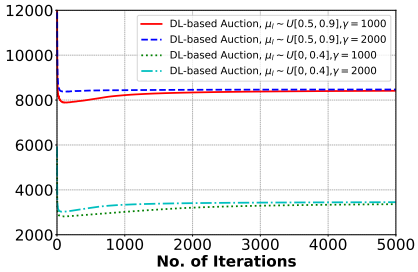


Fig. 19: Revenue of cluster head 2 under different approximation qualities.

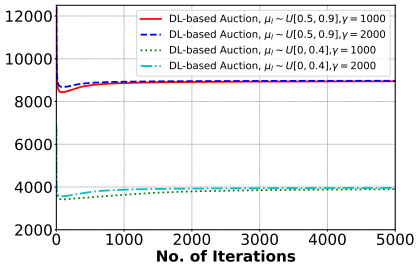


Fig. 20: Revenue of cluster head 3 under different approximation qualities.

that with more data, the workers in the population can gain a larger proportion of the pooled rewards, relative to that of workers from other populations.

6.2 Upper-Level Deep Learning Based Auction

In this section, we perform simulations to evaluate the performance of the Deep-Learning based auction. For comparison, the classic SPA is chosen as the baseline scheme. The TensorFlow Deep Learning library is used to implement the optimal auction design. We consider a network with the formations of 3 clusters and 10 model owners. We first evaluate the performance of the Deep-Learning based auction against the traditional SPA. Then, we proceed to study the impacts of (i) data coverage of the cluster heads, (ii) model owners with varying distribution for data coverage requirement, and (iii) different quality of approximation, on the revenues of the cluster heads.

6.2.1 Evaluation of the Deep Learning Based Auction

From Figs. 14 to 16, we observe that the revenue of the cluster heads determined by the deep-learning based auction is consistently higher than that of the conventional SPA scheme. The reason is that the SPA scheme guarantees IC, but does not guarantee that the revenue of the seller is maximized. While preserving the properties of IC and IR of the traditional auction, the deep-learning based auction maximizes the revenue earned by the cluster heads by providing their services to the model owners that value their services the most.

The revenues of the cluster heads are affected by the amount of data coverage. In Fig. 17, we observe that when the data coverage of the cluster head is higher, the revenue earned is also higher. In particular, when the model owners have high requirement for data coverage, i.e., $\mu_l \sim U[0.5, 0.9]$, cluster head 3 with the highest data coverage of 0.4 earns a revenue of 8944 whereas cluster head 1 with the lowest data coverage of 0.26 earns a revenue of 7182. This is due to the fact that the cluster head with the highest data coverage is allowed to conduct auction its services first. Hence, it will be able to offer its service to the model owner with the highest data coverage requirement in which the model owner is willing to pay the highest price as compared to other model owners in the network. Intuitively, this serves to compensate the cluster head for the higher rewards expense it incurs.

We examine the impacts on the cluster heads' revenues when they are presented with model owners with data coverage requirement of different distribution ranges. Specifically, model owners can take on two distribution ranges, i.e., $\mu_l \sim U[0, 0.4]$ and $\mu_l \sim U[0.5, 0.9]$. In Fig. 14, cluster head 1 which has a total data coverage of 0.26 earns a revenue of 2724 when model owners have data coverage requirements that range between 0 and 0.4. On the other hand, when model owners have higher range of data coverage requirements, i.e., between 0.5 and 0.9, cluster head 1 earns a higher revenue of 7182. Since the model owners with higher data coverage requirement value the cluster head more, they have more incentive to pay a higher price which results in the higher revenue earned by the cluster head. Similar trends are observed for cluster head 2 and cluster head 3 in Fig. 15 and Fig. 16 respectively.

To further evaluate the performance of the deep-learning based auction, we consider the cluster heads' revenues under different quality of approximation, γ. The quality of approximation is used in the *softmax* function in the

determination of the winning model owner. We consider two values for the quality of approximation, i.e., 1000 and 2000. We observe in Fig. 18 to 20, the revenue of the cluster head increases slightly when the quality of approximation is higher. Specifically, given the model owners with a high requirement for data coverage, i.e., $\mu_l \sim U[0.5, 0.9]$, the revenues of cluster head 3 are 8944 and 8969 when the values of γ are 1000 and 2000, respectively. This follows that with a greater value of approximation quality, the neural network is able to solve the optimization problem which maximizes the revenue of the cluster heads.

7 Conclusion

In this paper, we proposed a resource allocation and incentive mechanism design framework for HFL. We considered a two-level problem and leveraged on the evolutionary game theory to derive the equilibrium solution for the cluster selection phase. Then, we introduced a deep learning based auction mechanism to value the cluster head's services. The performance evaluation shows the uniqueness and stability of the evolutionary equilibrium, as well as the revenue maximizing property of the auction mechanism. In the future work, we will consider social network effects and their impact on the cluster selection decisions of the workers, as in [26]. Moreover, we may also account for the existence of malicious workers.

Acknowledgment

This work was supported in part by Alibaba Group through Alibaba Innovative Research (AIR) Program and Alibaba-NTU Singapore Joint Research Institute (JRI), by National Research Foundation, Singapore, under its AI Singapore Programme under AISG awards AISG2-RP-2020-019 and AISG-GC-2019-003, in part by WASP/NTU under Grant M4082187 (4080), in part by the Singapore Ministry of Education (MOE) Tier 1 (RG16/20), in part by the National Natural Science Foundation of China under Grant 62071343, and in part by SUTD under Grant SRG-ISTD-2021-165. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

References

[1] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *I*, 2019.

[2] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[3] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, and P. Hui, "A survey on edge intelligence," *arXiv preprint arXiv:2003.12172*, 2020.

[4] W. Y. B. Lim, J. S. Ng, Z. Xiong, D. Niyato, C. Leung, C. Miao, and Q. Yang, "Incentive mechanism design for resource sharing in collaborative edge learning," *arXiv preprint arXiv:2006.00511*, 2020.

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

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

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

[8] W. Y. B. Lim, J. Huang, Z. Xiong, J. Kang, D. Niyato, X.-S. Hua, C. Leung, and C. Miao, "Towards federated learning in uav-enabled internet of vehicles: A multi-dimensional contract-matching approach," *arXiv preprint arXiv:2004.03877*, 2020.

[9] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

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

[11] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8866–8870.

[12] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Wireless communications for collaborative federated learning in the internet of things," *arXiv preprint arXiv:2006.02499*, 2020.

[13] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[14] J. W. Weibull, *Evolutionary game theory*. MIT press, 1997.

[15] E. Peltonen, M. Bennis, M. Capobianco, M. Debbah, A. Ding, F. Gil-Castiñeira, M. Jurmu, T. Karvonen, M. Kelanti, A. Kliks *et al.*, "6g white paper on edge intelligence," *arXiv preprint arXiv:2004.14850*, 2020.

[16] W. Y. B. Lim, Z. Xiong, C. Miao, D. Niyato, Q. Yang, C. Leung, and H. V. Poor, "Hierarchical incentive mechanism design for federated machine learning in mobile networks," *IEEE Internet of Things Journal*, 2020.

[17] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[18] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020.

[19] L. U. Khan, M. Alsenwi, Z. Han, and C. S. Hong, "Self organizing federated learning over wireless networks: A socially aware clustering approach," in *2020 International Conference on Information Networking (ICOIN)*. IEEE, 2020, pp. 453–458.

[20] P. Dütting, Z. Feng, H. Narasimhan, D. C. Parkes, and S. S. Ravindranath, "Optimal Auctions through Deep Learning," *arXiv preprint arXiv:1706.03459*, 2017.

[21] M. Elloumi, R. Dhaou, B. Escrig, H. Idoudi, and L. A. Saidane, "Monitoring road traffic with a uav-based system," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2018, pp. 1–6.

[22] M. Xie, H. Yin, H. Wang, F. Xu, W. Chen, and S. Wang, "Learning graph-based poi embedding for location-based recommendation," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 15–24.

[23] M. Handy, M. Haase, and D. Timmermann, "Low energy adaptive clustering hierarchy with deterministic cluster-head selection," in *4th international workshop on mobile and wireless communications network*. IEEE, 2002, pp. 368–372.

[24] H. Junping, J. Yuhui, and D. Liang, "A time-based cluster-head selection algorithm for leach," in *2008 IEEE Symposium on Computers and Communications*. IEEE, 2008, pp. 1172–1176.

[25] R. Ferdous, V. Muthukkumarasamy, and E. Sithirasenan, "Trust-based cluster head selection algorithm for mobile ad hoc networks," in *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, 2011, pp. 589–596.

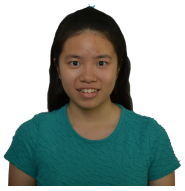
[26] G. Zhang, K. Yang, and H.-H. Chen, "Socially aware cluster formation and radio resource allocation in d2d networks," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 68–73, 2016.

[27] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet of Things Journal*, 2020.

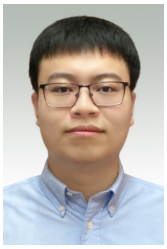
[28] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gurel, B. Li, C. Zhang, D. Song, and C. Spanos, "Towards efficient data valua-

tion based on the shapley value," *arXiv preprint arXiv:1902.10275*, 2019.
[29] K. Zhu, E. Hossain, and D. Niyato, "Pricing, spectrum sharing, and service selection in two-tier small cell networks: A hierarchical dynamic game approach," *IEEE Transactions on Mobile Computing*, vol. 13, no. 8, pp. 1843–1856, 2013.
[30] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *arXiv preprint arXiv:1911.02417*, 2019.
[31] X. Gong, L. Duan, X. Chen, and J. Zhang, "When social network effect meets congestion effect in wireless networks: Data usage equilibrium and optimal pricing," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 2, pp. 449–462, 2017.
[32] D. Niyato and E. Hossain, "Dynamics of network selection in heterogeneous wireless networks: An evolutionary game approach," *IEEE transactions on vehicular technology*, vol. 58, no. 4, 2008.
[33] J. Hofbauer and K. Sigmund, "Evolutionary game dynamics," *Bulletin of the American mathematical society*, vol. 40, no. 4, pp. 479–519, 2003.
[34] X. Gao, S. Feng, D. Niyato, P. Wang, K. Yang, and Y.-C. Liang, "Dynamic access point and service selection in backscatter-assisted rf-powered cognitive networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8270–8283, 2019.
[35] J. Engwerda, *LQ dynamic optimization and differential games*. John Wiley & Sons, 2005.
[36] S. Sastry, "Lyapunov stability theory," in *Nonlinear Systems*. Springer, 1999, pp. 182–234.
[37] R. Ke, Z. Li, J. Tang, Z. Pan, and Y. Wang, "Real-time traffic flow parameter estimation from uav video based on ensemble classifier and optical flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 54–64, 2018.
[38] Y. Zou, S. Feng, D. Niyato, Y. Jiao, S. Gong, and W. Cheng, "Mobile Device Training Strategies in Federated Learning: An Evolutionary Game Approach," in *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, 2019, pp. 874–879.
[39] R. B. Myerson, "Optimal auction design," *Mathematics of operations research*, vol. 6, no. 1, pp. 58–73, 1981.
[40] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

Wei Yang Bryan Lim is currently an Alibaba Talent Programme PhD candidate with the Alibaba-NTU Joint Research Institute (JRI), Nanyang Technological University (NTU), Singapore. Prior to joining the JRI in 2019, he graduated with two First-Class Honors in Economics and Business Administration (Finance) from the National University of Singapore (NUS). During his PhD candidature, he has won 2 Best Paper Awards in the IEEE Wireless Communications and Networking Conference (WCNC) and the ACM Mobicom-Dronecom workshop. He regularly serves as a reviewer in leading journals and flagship conferences and is currently the assistant to the Editor-in-Chief of the IEEE Communications Surveys & Tutorials.



Jer Shyuang Ng graduated with Double (Honours) Degree in Electrical Engineering (Highest Distinction) and Economics from National University of Singapore (NUS) in 2019. She is currently an Alibaba PhD candidate with the Alibaba Group and Alibaba-NTU Joint Research Institute, Nanyang Technological University (NTU), Singapore. Her research interests include incentive mechanisms and edge computing.



Zehui Xiong (S'17) is currently an Assistant Professor in the Pillar of Information Systems Technology and Design, Singapore University of Technology and Design. Prior to that, he was a researcher with Alibaba-NTU Joint Research Institute, Singapore. He received the PhD degree in Nanyang Technological University, Singapore. He was the visiting scholar at Princeton University and University of Waterloo. His research interests include wireless communications, network games and economics, blockchain, and edge intelligence. He has published more than 100 research papers in leading journals and flagship conferences and 4 of them are ESI Highly Cited Papers. He has won 5 Best Paper Awards in international conferences and technical committee. He is now serving as the editor or guest editor for many leading journals including IEEE Transactions. He is the recipient of the Chinese Government Award for Outstanding Students Abroad in 2019, and NTU SCSE Best PhD Thesis Runner-Up Award in 2020. He is the Founding Vice Chair of Special Interest Group on Wireless Blockchain Networks in IEEE Cognitive Networks Technical Committee.

Jiangming Jin graduated from Singapore Nanyang Technological University in 2013 and obtained his PhD degree in Computer Engineering. Before that, he obtained his Bachelor's degree in University of Electronic Science and Technology of China in 2008. Dr. Jiangming Jin started his career in J. P. Morgan (Singapore & Beijing). In J. P. Morgan, he worked with the most sophisticated and large-scale financial computing system and also worked with JPM Credit Portfolio Group in credit derivative calculations. Dr. Jin begins a venture journey in autonomous driving at TuSimple (NASDAQ: TSP) in 2017. In TuSimple, he has served successively as Director of HPC and Senior Expert of Engineering to oversee projects of High Performance Robotic Middleware and Heterogeneous Computing Systems. Dr. Jiangming Jin has wide and deep knowledge in Hi-Tech areas including Fintech, AI Chips and Software, Big Data and Machine Learning Systems, 5G and IoT. In such areas, Dr. Jin has published over 20 research papers in top conferences/journals and applied over 10 US/CN patents. Dr. Jin also served as session-chair or keynote-speaker in several AI Summits/Forums. Dr. Jin is the member of IEEE, executive member of CCF-TCHPC and CCF-TCARCH. Dr. Jin also holds several advisory positions in Universities and Non-Profit Institute such as Shanghai Tech University, Tianjin University, and Shenzhen Fintech Association.

Yang Zhang (M'11) is currently an associate professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. He received B. Eng. and M. Eng. from Beijing University of Aeronautics and Astronautics (Beihang University) in 2008 and 2011, respectively. He obtained a Ph.D. degree in Computer Engineering from Nanyang Technological University (NTU), Singapore, in 2015. He is an associate editor of EURASIP Journal on Wireless Communications and Networking, and technical committee member of Computer Communications.

Dusit Niyato (M'09-SM'15-F'17) is currently a professor in the School of Computer Science and Engineering and, by courtesy, School of Physical & Mathematical Sciences, at the Nanyang Technological University, Singapore. He received B.E. from King Mongkuk's Institute of Technology Ladkrabang (KMITL), Thailand in 1999 and Ph.D. in Electrical and Computer Engineering from the University of Manitoba, Canada in 2008. He has published more than 380 technical papers in the area of wireless and mobile networking, and is an inventor of four US and German patents. He has authored four books including "Game Theory in Wireless and Communication Networks: Theory, Models, and Applications" with Cambridge University Press. He won the Best Young Researcher Award of IEEE Communications Society (ComSoc) Asia Pacific (AP) and The 2011 IEEE Communications Society Fred W. Ellersick Prize Paper Award. Currently, he is serving as a senior editor of IEEE Wireless Communications Letter, an area editor of IEEE Transactions on Wireless Communications (Radio Management and Multiple Access), an area editor of IEEE Communications Surveys and Tutorials (Network and Service Management and Green Communication), an editor of IEEE Transactions on Communications, an associate editor of IEEE Transactions on Mobile Computing, IEEE Transactions on Vehicular Technology, and IEEE Transactions on Cognitive Communications and Networking. He was a guest editor of IEEE Journal on Selected Areas on Communications. He was a Distinguished Lecturer of the IEEE Communications Society for 2016-2017. He was named the 2017, 2018, 2019 highly cited researcher in computer science. He is a Fellow of IEEE.

Cyril Leung received the B.Sc. (First Class Hons.) degree from Imperial College, University of London, U.K., and the M.S. and Ph.D. degrees in electrical engineering from Stanford University. He has been an Assistant Professor with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, and the Department of Systems Engineering and Computing Science, Carleton University. Since 1980, he has been with the Department of Electrical and Computer Engineering, University of British Columbia (UBC), Vancouver, Canada, where he is a Professor and currently holds the PMC-Sierra Professorship in Networking and Communications. He served as an Associate Dean of Research and Graduate Studies with the Faculty of Applied Science, UBC, from 2008 to 2011. His research interests include wireless communication systems, data security and technologies to support ageless aging for the elderly. He is a member of the Association of Professional Engineers and Geoscientists of British Columbia, Canada.

Chunyan Miao received the BS degree from Shandong University, Jinan, China, in 1988, and the MS and PhD degrees from Nanyang Technological University, Singapore, in 1998 and 2003, respectively. She is currently a professor in the School of Computer Science and Engineering, Nanyang Technological University (NTU), and the director of the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY). Her research focus on infusing intelligent agents into interactive new media (virtual, mixed, mobile, and pervasive media) to create novel experiences and dimensions in game design, interactive narrative, and other real world agent systems.