This article has been accepted for publication in IEEE Transactions on Network Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2023.3276854

Distributed Clustering for Cooperative Multi-Task Learning Networks

Jiani Li, Weihan Wang, Waseem Abbas, *Member, IEEE* and Xenofon Koutsoukos, *Fellow, IEEE*

Abstract—Distributed learning enables collaborative training of machine learning models across multiple agents by exchanging model parameters without sharing local data. Each agent generates data from distinct but related distributions, and multi-task learning can be effectively used to model related tasks. This paper focuses on clustered multi-task learning, where agents are partitioned into clusters with distinct objectives, and agents in the same cluster share the same objective. The structure of such clusters is unknown apriori. Cooperation with the agents in the same cluster is beneficial and improves the overall learning performance. However, indiscriminate cooperation of agents with different objectives leads to undesired outcomes. Accurately capturing the clustering structure benefits the cooperation and offers many practical benefits; for instance, it helps advertising companies better target their ads. This paper proposes an adaptive clustering method that allows distributed agents to learn the most appropriate neighbors to collaborate with and form clusters. We prove the convergence of every agent towards its objective and analyze the network learning performance using the proposed clustering method. Further, we present a method of computing combination weights that approximately optimizes the network's learning performance to determine how one should aggregate the neighbors' model parameters after the clustering step. The theoretical analysis is well-validated by the evaluation results using target localization and digits classification, showing that the proposed clustering method outperforms existing distributed clustering methods as well as the case where agents do not cooperate.

Index Terms—distributed clustering, distributed cooperative learning, multi-agent networks, multi-task learning

1 Introduction

Distributed learning has attracted increasing attention due to the growth of machine learning (ML) applications in distributed devices within multi-agent networks, such as mobile phones, wearable devices, and smart homes [1], [2], [3], [4], [5], [6]. In such networks, multiple agents can operate in a distributed and cooperative manner to achieve a learning task. For example, consider learning the behavior of users in a cellular network based on data generated using various mobile applications. Each user may generate data that follows a distinct distribution, and it is common to learn separate models for each user. However, people may exhibit similar behaviors, and similarities among models commonly exist [7]. In this case, cooperation among agents could be leveraged to promote the learning performance over the network. Given data privacy concerns, cooperation among agents in a network typically relies on exchanging model parameters instead of data. In a distributed learning network, an agent communicates model parameters with its local neighbors and also updates these parameters by incorporating the neighbors' information [8], [9]. It has been demonstrated that such cooperation enables improved learning performance over the network [10]. Furthermore, compared to federated learning [11], distributed learning with fully decentralized networked agents does not require a central server, thus addressing single-point-of-failure and scalability issues.

Clustering is a fundamental problem in data analysis and data mining with a wide variety of applications, including image segmentation, customer segmentation, information management, social networks analysis, and statistical data analysis [12], [13]. The goal is to group data points into clusters such that data objects within the same cluster are much similar to each other compared to data objects in any other cluster. The dataset sizes have multiplied over the years with an expansion of geographically distributed data sources connected through a network. Sometimes it is not feasible due to security, technical or economic reasons to accumulate data at one central site for clustering analysis. As a result, effective approaches to distributed clustering have become of significant importance [14], [15], [16]. There have been several successful approaches for clustering in a centralized setup; however, the topic of distributed clustering is still under development.

At the same time, it is natural to model a distributed learning network using clustered multi-task learning in which agents with similar interests are grouped in the same cluster, and different clusters represent distinct interests [17]. We note that if there are no similarities among the agents, the clustered multi-task network reduces to a non-cooperative network. At the same time, if all agents in the network are similar, the network reduces to a single-task cooperative network. Agents should not be aware of the clustered structure beforehand, and the clustering scheme should adapt to changes and accommodate any new agents. Clustered multi-task learning offers significant practical benefits. For instance, in a mobile application network, people of similar age or profession may exhibit similar preferences

- *Jiani Li (jiani.li@vanderbilt.edu) and Xenofon Koutsoukos (xenofon.koutsoukos@vanderbilt.edu) are with the Department of Computer Science, Vanderbilt University, Nashville, TN, USA. Weihan Wang (wwang103@stevens.edu) is with the Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ, USA. Waseem Abbas (waseem.abbas@utdallas.edu) is with the Department of Systems Engineering, University of Texas at Dallas, TX, USA.*

for particular content. Further, such preferences may deviate from the preferences of people of different ages or professions. Advertising companies, in particular, use such strategies to target better and manage their advertisements. Adaptive clustering allows agents to perform local optimization tasks while learning from neighbors with similar objectives. As a result, an agent is allocated to an appropriate cluster without knowing the clustering structure a priori [16], [18], [19], [20], [21]. Agents can then cooperate with others in the same cluster, thus, making cooperation more beneficial and meaningful. However, as we show in Section 4.1, existing methods typically rely on comparing the Euclidean distance between model parameters to measure their similarities and could result in two agents from different clusters starting cooperation during learning. In particular, such cooperation may continue and could drive agents to converge to a wrong cluster. Our evaluation in Section 5 also shows that such an approach could fail in learning the correct clustering structure.

To address such problems, we propose an *adaptive clustering method by comparing the losses of two agents*. The main idea is that *agents sharing similar objectives have similar underlying data distributions, and hence their models should fit each other's data distributions.* Moreover, to make the cooperation lead to a better learning performance (smaller loss), it is appropriate for an agent to cooperate only with those neighbors who incur a smaller loss than the agent itself. To measure similarity between an agent and its neighbor, we propose that an agent compute the loss by fitting its neighbor's model to its data and then comparing it to the one obtained by fitting the data to its own model. The agent cooperates with a neighbor only if the loss with the neighbor's model is no greater than the agent's own loss. In doing so, it is guaranteed that the loss is reduced in expectation by the cooperation, and agents converge to the objective of the cluster.

The main contributions of the paper are:

- We propose an adaptive clustering method in distributed multi-task learning networks that allows agents to perform the optimization task while simultaneously learn which neighbors are suitable for cooperation.
- We analyze the convergence and learning performance for the proposed method.
- We propose combination weights for aggregating the neighbors' model parameters after recognizing which neighbors to cooperate with using the proposed clustering method that approximately optimizes the network learning performance.
- We evaluate the proposed method for both linear regression and classification problems and compare the results with existing distributed clustering methods. The evaluation results show that the proposed method improves learning performance significantly compared to the non-cooperative approach by correctly estimating the clustering structure. In contrast, other methods fail to achieve the clustering information and exhibit inferior learning performance.

2 Related Work

Multi-Task Learning. Multi-task learning (MTL) deals with the problem of solving multiple related optimization tasks simultaneously to improve the overall performance of the models learned by each task with other auxiliary tasks [22]. There is a large body of related work in the area of MTL with different variations. For example, one area of work is related to deep learning, intending to learn multiple objectives from a shared representation by sharing layers and splitting architecture in the deep neural networks [23], [24], [25], [26], [27], [28]. Such a framework usually assumes that data sets are collected from all the tasks in a central server and uses a single model to do the learning. An example is simultaneously learning the depth and semantics of RGB images [29]. In contrast, in this paper, we consider distributed multi-task learning where networked agents maintain separate data sets and models without a central server [30], [31]. While the first MTL framework is widely used in deep learning, e.g., computer vision and natural language processing, the second framework considered in this paper is naturally suited for distributed learning in multi-agent systems, such as mobile phones, autonomous vehicles, and smart cities [2], [32], [33], [34]. We note that the scope of this work overlaps slightly with [31], which also studies the problem of distributed multi-task learning. However, in [31], the focus is on the resilient aggregation among agents in the presence of adversarial agents. Similarities may exist among agents, and the objective of [31] is to promote such similarities. However, agents may not form clusters, and the objective of [31] is not related to any clustered structure among agents. In this work, we assume the clustered structure exists in the network. Our goal is to make sure agents stop exchanging information with neighbors sharing a different learning objective (i.e., from a different cluster) and are accurately clustered at the end of learning.

Distributed Clustering over Networks. Clustering is a well-known unsupervised learning technique for grouping a set of data points [35]. In contrast, this paper deals with the distributed clustering problem of a group of networked agents running individual optimization tasks [16], [18], [20], [36]. In such methods, agents perform local tasks while simultaneously learning which neighbors they should cooperate with by measuring their relatedness. Compared to traditional clustering, distributed clustering is more challenging since any clustering error could lead an agent towards an undesired model. Measuring the Euclidean distance between the model parameters of agents is a principle method used in distributed clustering. For example, in [16], if the Euclidean distance is less than a pre-defined threshold, the two agents will be clustered into the same group. Similarly, in [20], an accumulated Euclidean distance within a time-based sliding window is used. Adaptive weights based on Euclidean distance are used in [18] and [36] for distributed clustering. As we discuss in Section 4.1, measuring similarities relying on comparing the Euclidean distance between two model parameters could lead agents to converge to a wrong cluster, as illustrated in our evaluation (Section 5) as well. In contrast, the proposed method can capture accurate clustering information and guarantee that agents cooperate only with

the neighbors sharing the same task. This, in turn, ensures that agents converge to their global minimizers resulting in an improved overall learning performance than without cooperation.

3 Clustered Multi-Task Network

Notation. In this paper, $|A|$ denotes the cardinality of a set A, $\|\cdot\|$ denotes the $2-$norm, $\text{col}\{x_1,\ldots,x_n\}$ denotes the column vector with entries $x_1,\ldots,x_n$ and $\text{diag}\{x_1,\ldots,x_n\}$ denotes the diagonal matrix with entries $x_1,\ldots,x_n$; $\mathbb{E}[\cdot|\xi]$ denotes the expected value of a random variable ξ. If the context is clear, $\mathbb{E}[\cdot]$ is used. In addition, $(\cdot)^\top$ denotes transposition, $\text{Tr}(\cdot)$ is the trace of a matrix, $(M)^{-1}$ is the inverse of a matrix M, and $\otimes$ denotes Kronecker product of two matrices.

We consider a network of N agents modeled by an *undirected graph* $G = (V, E)$, where V represents agents and E represents interactions between agents. A bi-directional edge $(l,k) \in E$ means that agents k and l can exchange information with each other. Note that $(k,k) \in E, \forall k \in V$. The *neighborhood* of k is the set $\mathcal{N}_k = \{l \in V | (l,k) \in E\}$. Every agent is associated with a loss function $r_k(\theta) : \mathbb{R}^d \to \mathbb{R}$, of a vector parameter θ. It is assumed that each $r_k(\theta)$ is strongly-convex and is minimized at the unique point θ_k^*. Agents can then be categorized into $Q \in [1, N]$ mutually-exclusive clusters $\{\mathcal{C}_q; q = 1, 2, \ldots, Q\}$, such that agents in the same cluster share the same minimizer for their individual loss functions. Denote the unique minimizer for cluster $\mathcal{C}_q$ as θ_q^o, then $\theta_k^* = \theta_q^o$ for all $k \in \mathcal{C}_q$. We also assume that the underlying network topology is connected and clusters are inter-connected so that agents may have neighbors from different clusters. Note that agents do not know which of the neighbors belong to the same cluster as themselves. Agents are interested in solving the following optimization problem:

$$\min_{\{\theta_q\}_{q=1}^Q} \sum_{q=1}^{Q} \sum_{k \in \mathcal{C}_q} r_k(\theta_q). \tag{1}$$

To solve (1) in a distributed and cooperative way, we use the *adapt-then-combine* (ATC) diffusion algorithm [30] with a clustering step in-between, which takes the following iterative steps for an agent k at iteration i:

$$\hat{\theta}_{k,i} = \theta_{k,i-1} - \mu_k \widehat{\nabla r_k}(\theta_{k,i-1}), \quad \text{(adaptation)} \tag{2}$$

$$\text{Obtain } \mathcal{N}_{k,i}^+ \text{ by clustering,} \quad \text{(clustering)} \tag{3}$$

$$\theta_{k,i} = \sum_{l \in \mathcal{N}_{k,i}^+} a_{lk}(i) \hat{\theta}_{l,i}, \quad \text{(combination)} \tag{4}$$

where μ_k is the step-size, $\mathcal{N}_{k,i}^+$ is the set of neighbors belonging to the same cluster as agent k at iteration i, $\widehat{\nabla r_k}(\theta_{k,i-1})$ is the stochastic gradient of the loss function $r_k(\cdot)$ at $\theta_{k,i-1}$, and $a_{lk}(i)$ denotes the weight assigned by agent k to l at iteration i that satisfies the following constraints:

$$\sum_{l \in \mathcal{N}_{k,i}^+} a_{lk}(i) = 1, a_{lk}(i) \geq 0, a_{lk}(i) = 0 \text{ if } l \notin \mathcal{N}_{k,i}^+. \tag{5}$$

The ATC algorithm indicates that at each iteration i, agent k minimizes the individual loss function using stochastic gradient descent (SGD) given local data followed by a combination step that aggregates the model parameters of the neighbors in the same cluster according to the weights assigned to them.

The goal of this paper is to solve the optimization problem (1) in a distributed and cooperative way using the ATC diffusion algorithm in (2)-(4) by designing the clustering and combination weights (5) that agents use for cooperation. Cooperation among agents can improve learning if they share common objectives. However, when agents pursue different objectives, indiscriminate cooperation leads to undesired outcomes [37]. Therefore, the agents need to use an accurate clustering method to learn which neighbors share similar objectives. By doing so, agents only cooperate with the neighbors in the same cluster and stop cooperating with those from different clusters, thus ensuring that the cooperation is beneficial.

4 Adaptive Clustering

In this section, we first propose a distributed clustering method that allows agents to learn which neighbors are in the same cluster. Next, we analyze the convergence and learning performance using the proposed clustering method. Finally, we propose a method of computing combination weights to aggregate the model parameters of the neighbors belonging to the same cluster that approximately optimizes (1).

4.1 Clustering hypothesis

Distributed clustering methods are based on measuring similarities among agents. Existing methods rely on comparing the Euclidean distance between two agents' model parameters to determine whether they belong to the same cluster [16], [18], [19], [20], [36]. For example, in [16], the following hypothesis test is used for agent k to determine whether a neighbor l belongs to the same cluster:

$$\|\hat{\theta}_{l,i} - \hat{\theta}_{k,i}\|^2 \underset{\mathbb{H}_1}{\overset{\mathbb{H}_0}{\lessgtr}} d_{k,l}^2, \tag{6}$$

where $\mathbb{H}_0$ denotes the hypothesis that $l \in \mathcal{N}_{k,i}^+$ and $\mathbb{H}_1$ denotes the hypothesis that $l \notin \mathcal{N}_{k,i}^+$, and $d_{k,l} > 0$ is a predefined threshold.

We find that methods based on Euclidean distance between model parameters may lead agents to cooperate with neighbors from another cluster which could prevent agents from converging. Consider the example shown in Figure 1. Assume that k has only one neighbor l, and at iteration i, agent k identifies l to be in the same cluster as $\|\hat{\theta}_{l,i} - \hat{\theta}_{k,i}\|^2 < d_{k,l}^2$. If $\|\hat{\theta}_{l,i} - \hat{\theta}_k^*\|^2 > \|\hat{\theta}_{k,i} - \theta_k^*\|^2$, then $\theta_{k,i}$ as a combination of $\hat{\theta}_{k,i}$ and $\hat{\theta}_{l,i}$ (given (4)) will move away from θ_k^* rendering $r_k(\theta_{k,i}) > r_k(\hat{\theta}_{k,i})$. If l is from another cluster that moves away from θ_k^* yet keeps falling into hypothesis $\mathbb{H}_0$ for agent k, then k will continue to cooperate with l and will fail to converge to θ_k^*.

To ensure that agents converge to their global minimizers using clustering-based cooperation, we propose an alternative hypothesis test as follows:

$$\widehat{r_k}(\hat{\theta}_{l,i}) - \widehat{r_k}(\hat{\theta}_{k,i}) \underset{\mathbb{H}_1}{\overset{\mathbb{H}_0}{\lessgtr}} 0, \tag{7}$$

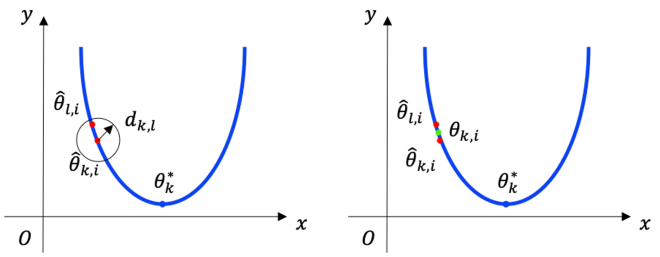


Fig. 1: Example of an undesired outcome due to clustering using hypothesis testing (6): (a) k identifies l to be in the same cluster; (b) $\theta_{k,i}$ as a combination of $\hat{\theta}_{k,i}$ and $\hat{\theta}_{l,i}$ moves away from $\theta^*_{k,i}$ rendering $r_k(\theta_{k,i}) > r_k(\hat{\theta}_{k,i})$.

where $\mathbb{H}_0$ denotes the hypothesis that $l \in \mathcal{N}^+_{k,i}$ and $\mathbb{H}_1$ denotes the hypothesis that $l \notin \mathcal{N}^+_{k,i}$, and $\widehat{r_k}(\cdot)$ is the approximation of $r_k(\cdot)$ with $\mathbb{E}\left[\widehat{r_k}(\theta)\right] = \mathbb{E}\left[r_k(\theta)\right]$. The approximation $\widehat{r_k}(\cdot)$ can be performed using stacked data received in the previous iterations. Alternatively, if we use (mini-) batch gradient descent in the adaptation step, then the batch loss can be used as $\widehat{r_k}(\cdot)$.

The hypothesis test (7) indicates that the clustering is based on measuring the loss by fitting the neighbor's model to an agent's data distribution and comparing the loss with its own loss. Therefore, we have

$$\begin{aligned}\mathbb{E}\left[\widehat{r_k}(\theta_{k,i})\right] &= \mathbb{E}\left[\widehat{r_k}\left(\sum_{l\in\mathcal{N}^+_{k,i}} a_{lk}(i)\hat{\theta}_{l,i}\right)\right] \\ \leq &\mathbb{E}\left[\sum_{l\in\mathcal{N}^+_{k,i}} a_{lk}(i)\widehat{r_k}(\hat{\theta}_{l,i})\right] \leq \mathbb{E}\left[\widehat{r_k}(\hat{\theta}_{k,i})\right],\end{aligned} \tag{8}$$

where the first inequality follows by Jensen's inequality assuming $\widehat{r_k}(\cdot)$ is convex, and the second inequality follows by the proposed clustering hypothesis test (7). Since we assume that $\mathbb{E}\left[\widehat{r_k}(\theta)\right] = \mathbb{E}\left[r_k(\theta)\right]$, it follows from (8) that

$$\mathbb{E}\left[r_k(\theta_{k,i})\right] \leq \mathbb{E}\left[r_k(\hat{\theta}_{k,i})\right]. \tag{9}$$

Thus, clustering using the hypothesis test (7) results in a reduced loss in expectation. Later in Theorem 1, we will show that this results in the convergence of every agent towards their objectives.

4.2 Convergence and learning performance

We first prove the convergence of every agent k towards its objective θ^*_k using the clustering hypothesis test (7), which ensures that agent k converges to the true cluster. Next, we analyze the learning performance over the network using (7). Below, we list the necessary assumptions for the loss functions and gradient noise used in our results that are standard in the literature [10], [16], [38].

Assumption 1. (Loss functions)

(a) Different clusters have distinct minimizers $\theta^o_q \neq \theta^o_r$ if $q \neq r$. Further, the difference between every two minimizers of two distinct clusters are sufficiently large such that $r_k(\theta^o_q) < r_k(\theta^o_r)$ if $k \in \mathcal{C}_q$ and $k \notin \mathcal{C}_r$.

(b) The sequence $\{\theta_{k,i}\}$ for every agent k is contained in an open set over which $r_k(\cdot)$ is always upper-bounded by a scalar r_{inf}.

(c) Each individual loss function $r_k(\theta)$ is strongly convex, twice-differentiable, and has bounded Hessian matrix function, which also implies that it has a Lipschitz continuous gradient, i.e.,

$$r_k(\theta_2) \geq r_k(\theta_1) + \langle \nabla r_k(\theta_1), \theta_2 - \theta_1\rangle + \frac{m}{2}\|\theta_2 - \theta_1\|^2, \quad \forall \theta_1, \theta_2, \text{ for some } m > 0.$$

$$LI_d \leq \nabla^2 r_k(\theta) \leq UI_d, \forall \theta, \text{ for some } 0 \leq L \leq U < \infty.$$

$$\|\nabla r_k(\theta_1) - \nabla r_k(\theta_2)\| \leq U\|\theta_1 - \theta_2\|, \forall \theta_1, \theta_2.$$

(d) Denote the network Hessian function

$$\nabla^2 \mathcal{R}(\Theta) \triangleq \text{diag}\{r_1(\theta_1), \ldots, r_N(\theta_N)\},$$

where $\Theta \triangleq \text{col}\{\theta_1, \ldots, \theta_N\} \in \mathbb{R}^{Nd\times 1}$. It is assumed that $\nabla^2\mathcal{R}(\Theta)$ satisfies the Lipschitz condition:

$$\left\|\nabla^2\mathcal{R}(\Theta_1) - \nabla^2\mathcal{R}(\Theta_2)\right\| \leq \kappa\left\|\Theta_1 - \Theta_2\right\|,$$

for any $\Theta_1, \Theta_2 \in \mathbb{R}^{Nd\times 1}$ and some $\kappa \geq 0$.

Let's denote the stochastic gradient noise as

$$s_{k,i}(\theta_{k,i-1}) \triangleq \widehat{\nabla r_k}(\theta_{k,i-1}) - \nabla r_k(\theta_{k,i-1}). \tag{10}$$

We stack the noise of every agent into a vector and obtain the network noise denoted by

$$\mathcal{S}_i(\Theta_{i-1}) \triangleq \text{col}\{s_{1,i}(\theta_{1,i-1}), \ldots, s_{N,i}(\theta_{N,i-1})\}.$$

We use the filtration $\{\mathbb{F}_i; i \geq 0\}$ to represent the information flow that is available up to the i-th iteration of the learning process. Then, the conditional covariance of $\mathcal{S}_i(\Theta_{i-1})$ is denoted by

$$\mathcal{V}_{s,i}(\Theta_{i-1}) \triangleq \mathbb{E}\left[\mathcal{S}_i(\Theta_{i-1})\mathcal{S}_i^\top(\Theta_{i-1})|\mathbb{F}_{i-1}\right]. \tag{11}$$

Assumption 2. (Gradient noise) The gradient noise satisfies the following properties:

(a) The stochastic approximations are unbiased estimates of gradients such that

$$\mathbb{E}\left[\mathcal{S}_i(\Theta_{i-1})|\mathbb{F}_{i-1}\right] = 0.$$

(b) The second-order moment of the stochastic gradient process satisfies:

$$\mathbb{E}\left[\left\|\widehat{\nabla r_k}(\theta_{i-1})\right\|^2 \middle| \mathbb{F}_{i-1}\right] \leq \alpha\|\nabla r_k(\theta_{i-1})\|^2 + \sigma_k^2.$$

for some $\alpha \geq 1, \sigma_s > 0$.

(c) The conditional covariance function satisfies the Lipschitz condition:

$$\|\mathcal{V}_{s,i}(\Theta^*) - \mathcal{V}_{s,i}(\Theta_{i-1})\| \leq \beta\left\|\Theta^* - \Theta_{i-1}\right\|^\gamma, \tag{12}$$

for some $\beta \geq 0, 0 < \gamma \leq 4$, with

$$\Theta^* \triangleq \text{col}\{\theta_1^*, \ldots, \theta_N^*\} = \text{col}\{\mathbb{1}_{|\mathcal{C}_q|} \otimes \theta^o_q; q = 1, \ldots, Q\}.$$

(d) The conditional covariance matrix at convergence $\mathcal{V}_s \triangleq \lim_{i\to\infty} \mathcal{V}_{s,i}(\Theta^*) > 0$ is symmetric and positive definite.

Theorem 1. *Under Assumptions 1-2, given sufficiently small step-sizes $\mu_k \in (0, \frac{1}{U\alpha}]$, every normal agent k using the hypothesis test* (7) *for clustering converges towards θ_k^* with*

$$\lim_{i\to\infty} \sup \mathbb{E}\left[r_k\left(\theta_{k,i}\right) - r_k(\theta_k^*)\right] = O(\mu_k). \tag{13}$$

Proof. Let $\delta_k(\theta_{k,i}) = \mathbb{E}\left[r_k(\theta_{k,i}) - r_k(\theta_k^*)\right]$. By recursion (2)-(4) and Assumptions 1-2, using constant step-size $\mu_k \in (0, \frac{1}{U\alpha}]$, it follows from [38](Theorem 4.6) that the following error recursion holds for the SGD step (2):

$$\delta_k(\hat{\theta}_{k,i}) - \frac{\mu_k U\sigma_k^2}{2m} \leq (1-\mu_k m)\left(\delta_k(\theta_{k,i-1}) - \frac{\mu_k U\sigma_k^2}{2m}\right).$$

The hypothesis test (7) requires that k cooperates only with the neighbors incurring a loss $\widehat{r_k}\left(\hat{\theta}_{l,i}\right) \leq \widehat{r_k}\left(\hat{\theta}_{k,i}\right)$, which results in (9). Thus, it follows that $\delta_k(\theta_{k,i}) \leq \delta_k(\hat{\theta}_{k,i})$. Thus,

$$\delta_k(\theta_{k,i}) - \frac{\mu_k U\sigma_k^2}{2m} \leq (1-\mu_k m)\left(\delta_k(\theta_{k,i-1}) - \frac{\mu_k U\sigma_k^2}{2m}\right). \tag{14}$$

Given $\mu_k \in (0, \frac{1}{U\alpha}]$, with $\alpha \geq 1$, $m \leq U$ (given the property of m-strongly convex and U-Lipschitz continuous gradient), it holds that $(1-\mu_k m) \in [0, 1)$. Applying (14) repeatedly through iteration $i \in \mathbb{N}$, we obtain

$$\begin{aligned}\delta_k(\theta_{k,i}) &\leq \frac{\mu_k U\sigma_k^2}{2m} + (1-\mu_k m)^i\left(\delta_k(\theta_{k,0}) - \frac{\mu_k U\sigma_k^2}{2m}\right)\\ &\overset{i\to\infty}{\longrightarrow} \frac{\mu_k U\sigma_k^2}{2m},\end{aligned} \tag{15}$$

and (13) holds accordingly. □

Theorem 1 indicates that every agent converges to the objective of its cluster $\theta_q^o = \theta_k^*$ if $k \in \mathcal{C}_q$. Given Assumption 1(a), at convergence, $r_k(\theta_q^o) < r_k(\theta_r^o)$ if $k \in \mathcal{C}_q$ and $k \notin \mathcal{C}_r$ for any $r \neq q$. For any agent $l \notin \mathcal{C}_q$, suppose $l \in \mathcal{C}_r$, then it holds that $r_k(\theta_k^*) < r_k(\theta_l^*)$. Then, following hypothesis test (7), at convergence, agent k will not cooperate with any neighbor l from a different cluster. Therefore, the network is separated into distinct connected sub-networks (groups) where agents in the same group are from the same cluster.

Assume that there are $G \in [Q, N]$ groups, with N_g agents in group $\mathcal{G}_g$ such that $\sum_{g=1}^G N_g = N$. Then, the clustered multi-task network can be reduced to multiple single-task networks (groups). In each group, let A_g be the $N_g \times N_g$ weight matrix with $A_g \triangleq [\lim_{i\to\infty} a_{lk}(i); l, k \in \mathcal{G}_g]$. Since agents have non-trivial self-loops ($a_{kk}(i) > 0$), it follows that every weight matrix A_g is a left-stochastic and primitive matrix [39]. Then, by the Perron-Frobenius theorem [10], [40], such matrix has a simple eigenvalue at one and all other eigenvalues have magnitude strictly less than one. Moreover, let p_g denote the right eigenvector of A_g that is associated with the eigenvalue at one and normalize its entries to add up to one. Then, the entries of p_g are positive and smaller than one, such that

$$\begin{aligned}&A_g p_g = p_g,\ p_g^\top \mathbb{1} = 1,\ 0 < p_{g,k} < 1,\\ &\text{with } p_g = [p_{g,k}; k = 1, 2, \ldots, |p_g|].\end{aligned} \tag{16}$$

We can then obtain the average learning performance over the network as described in the following theorem.

Theorem 2. *Under Assumptions 1-2, given sufficiently small step-sizes $\mu_k \in (0, \frac{1}{U\alpha}]$ and the clustering hypothesis test* (7)*, the average learning performance of the network measured by the mean-squared-deviation (MSD) of the estimated parameters is given by*

$$\begin{aligned}&\lim_{i\to\infty} \sup \frac{1}{N}\mathbb{E}\left\|\widetilde{\Theta}_i\right\|^2\\ =&\frac{1}{2N}\sum_{g=1}^G N_g \mathrm{Tr}\left[\left(\sum_{k\in\mathcal{G}_g} p_{g,k}\mu_k H_k\right)^{-1}\left(\sum_{k\in\mathcal{G}_g} p_{g,k}^2\mu_k^2 V_k\right)\right],\end{aligned} \tag{17}$$

where $p_{g,k}$ subjects to (16)*, and*

$$\widetilde{\Theta}_i \triangleq \Theta^* - \Theta_i = \mathrm{col}\{\widetilde{\theta}_{1,i}, \ldots, \widetilde{\theta}_{N,i}\},$$

$$H_k \triangleq \nabla^2 r_k(\theta_k^*), \quad V_k \triangleq \lim_{i\to\infty} \mathbb{E}\left[s_{k,i}(\theta_k^*)s_{k,i}(\theta_k^*)^\top | \mathbb{F}_{i-1}\right].$$

Proof. The average MSD over the network is an average of the MSD of each groups. Then, (17) can be easily derived from the results of [10] (Theorem 11.3). □

4.3 The proposed combination weights

Next, we consider how to optimize the weights $a_{lk}(i)$ in order to optimize (1), which equivalently optimize each individual loss function. Given the clustering hypothesis test (7), at each iteration i, an agent k clusters neighbors in $\mathbb{H}_0$ as $\mathcal{N}_{k,i}^+$ and cooperates only with agents in $\mathcal{N}_{k,i}^+$. Using (4), we get an equivalent problem:

$$\min_{A_k} \left\|\sum_{l\in\mathcal{N}_{k,i}^+} a_{lk}(i)\hat{\theta}_{l,i} - \theta_k^*\right\|^2, \text{ subject to (5)},$$

where $A_k = [a_{1k}(i), \ldots, a_{|\mathcal{N}_{k,i}^+|k}(i)] \in \mathbb{R}^{1\times|\mathcal{N}_{k,i}^+|}$. As in a typical approximation approach [18], we consider

$$\left\|\sum_{l\in\mathcal{N}_{k,i}^+} a_{lk}(i)\hat{\theta}_{l,i} - \theta_k^*\right\|^2 \approx \sum_{l\in\mathcal{N}_{k,i}^+} a_{lk}(i)^2\left\|\hat{\theta}_{l,i} - \theta_k^*\right\|^2. \tag{18}$$

Since the loss functions r_k are assumed to be m-strongly convex, it follows that

$$\|\hat{\theta}_{l,i} - \theta_k^*\|^2 \leq \frac{2}{m}\left(r_k(\hat{\theta}_{l,i}) - r_k(\theta_k^*)\right). \tag{19}$$

Instead of directly minimizing the right side of (18), we consider minimizing its upper bound given in (19). Hence, by combining (18) and (19), we obtain: $\min_{A_k} \sum_{l\in\mathcal{N}_{k,i}^+} a_{lk}^2(i)\left(r_k(\hat{\theta}_{l,i}) - r_k(\theta_k^*)\right)$. Since $r_k(\theta_k^*)$ is small compared to $r_k(\theta_{l,i})$, we consider the following minimization problem:

$$\min_{A_k} \sum_{l\in\mathcal{N}_{k,i}^+} a_{lk}^2(i) r_k(\hat{\theta}_{l,i}), \text{ subject to (5)}. \tag{20}$$

Using the Lagrangian relaxation and the approximation value $\widehat{r_k}(\cdot)$ to replace $r_k(\cdot)$, we obtain the optimal solution

of (20) as an approximate optimization of (1) as

$$a_{lk}(i) = \frac{\widehat{r_k}(\hat{\theta}_{l,i})^{-1}}{\sum_{p\in\mathcal{N}_{k,i}^+} \widehat{r_k}(\hat{\theta}_{p,i})^{-1}}. \tag{21}$$

Combined with (7), we conclude the algorithm for the learning and clustering over networks in Algorithm 1.

Algorithm 1: Distributed learning and clustering over networks

Input: Initialize $w_{k,-1}$ for $k \in \mathcal{C}_q$ and $q = 1, 2, \ldots, Q$.

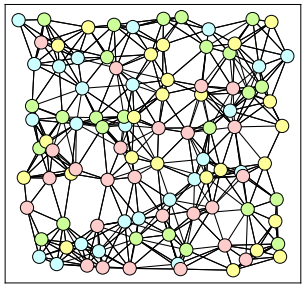
for $i \geq 0$ **do**
  for *every agent* k **do**
    Update $\hat{\theta}_{k,i}$ according to (2).
    Send $\hat{\theta}_{k,i}$ and receive $\hat{\theta}_{l,i}$ from neighbors.
    Cluster neighbors using (7) and obtain $\mathcal{N}_{k,i}^+$.
    Assign weights to neighbors in $\mathcal{N}_{k,i}^+$ according to (21).
    Aggregate neighbors' model parameters according to (4).

5 Evaluation

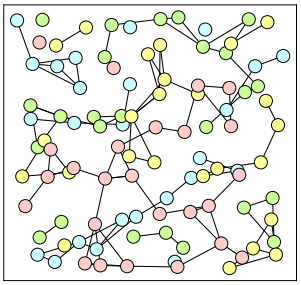
In this section, we evaluate the proposed distributed clustering method with the weights computed using Algorithm 1 and compare the results with the case where the agents (1) do not cooperate with each other (non-cooperative case), (2) cooperate using the average weights ($a_{lk} = \frac{1}{|\mathcal{N}_k|}$), and (3) cooperate using the quadratic distance-based clustering hypothesis test[1]. To avoid the selection of the user-defined parameter $d_{k,l}$ in (6), we instead use $\|\hat{\theta}_{l,i} - \theta_{k,i-1}\|^2 \underset{\mathbb{H}_1}{\overset{\mathbb{H}_0}{\lessgtr}} \|\hat{\theta}_{k,i} - \theta_{k,i-1}\|^2$ as proposed in [36]. Note that the most recent work applying the distance-based method in distributed clustering problem is in [19].

We consider two distributed learning examples: (1) target localization and (2) digit classification. For the classification problem, we use convolutional neural networks (CNNs) that are non-convex models. We show empirically that Algorithm 1 results in better learning performance than the other methods with correct clustering structure being learned, for both convex and non-convex models (such as CNNs). Note that given the setup of the distributed clustering network, agents need to transfer their model parameters to neighbors, a model with millions of parameters will lead to high latency and cost, which is prohibited from a frequent model exchange. As a result, the models and tasks we consider in the paper are lightweight. We note that there is considerable work addressing the multi-task clustering problem in a single server with powerful computational resources [41], [42]. However, these works are different from the distributed setting considered in this paper, making them unsuitable for comparison with ours. Detailed discussion can be found in Section 2.

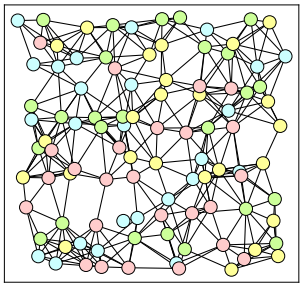
[1]Simulation code can be found in https://github.com/JianiLi/DistributedClustering.



(a)



(b)



(c)

Fig. 2: Target localization (nodes in the same color share the same target): (a) Initial network; (b) Final network by Algorithm 1; (c) Final network by the distance-based clustering method.

5.1 Target localization

Target localization is a linear regression problem in which the objective is to estimate the location of a target by minimizing the squared error loss of noisy streaming sensor data [17]. We consider a network of 100 agents randomly distributed in a planar region $\mathcal{W} = [0, 10] \times [0, 10] \in \mathbb{R}^2$ as shown in Figure 2a. An edge between two agents means that they are neighbors. Agents with index numbers from 0-24, 25-49, 50-75, 75-99 share the same target, which are indicated using the same color. However, the agents do not know this clustering information beforehand. The four target locations in $\mathbb{R}^2$ are: $(100, 200), (200, 100), (100, 100), (200, 200)$ respectively. At each iteration, every agent k has a noisy observation (streaming data) of the distance $d_k(i)$ and the unit direction vector $u_{k,i}$ pointing from x_k to its target based on built-in sensors. Let $\theta_k \in \mathbb{R}^2$ denote the estimation of the target location for agent k, then the loss is given by $r_k(\theta_{k,i}) = \mathbb{E}\left[\|d_k(i) - (\theta_{k,i} - x_k)^\top u_{k,i}\|^2\right]$. The approximation $\widehat{r_k}(\theta_{k,i})$ is computed by the last 10 stacked streaming datapoints. The distance measurement data has noise variance $\sigma_{d,k}^2 \in [0.1, 1]$, and the unit direction vector

This article has been accepted for publication in IEEE Transactions on Network Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2023.3276854

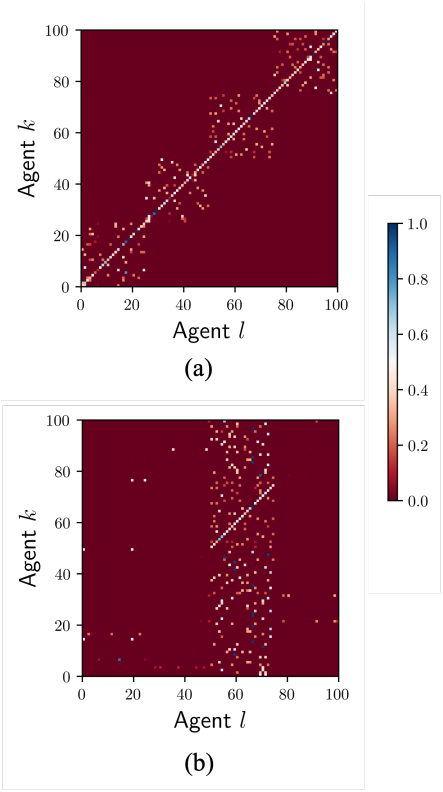


Fig. 3: Target localization: average weight matrix over time $\frac{1}{T+1}\sum_{i=0}^{T} a_{lk}(i)$: (a) Loss-based; (b) Distance-based

has additive white Guassian noise with diagnonal covariance matrices $R_{u,k} = \sigma_{u,k}^2 I_2$, with $\sigma_{u,k}^2 \in [0.01, 0.1]$ for different k. The step-size is set to be $\mu = 0.1$ for every agent.

The results are given in Figure 2 - Figure 5. From Figure 2, the loss-based clustering method (Algorithm 1) results in a clustered network at the end of the simulation (Figure 2(b)), and no agents in two different clusters end up being neighbors of each other. On the other hand, the distance-based method fails to correctly capture the clustering information (Figure 2(c)). Further, as shown in Figure 3, in the proposed method agents cooperate only with neighbors in the same cluster. Using distance-based clustering, the agents failed to identify those belonging to the same cluster. Figure 4 shows the learning loss using different clustering methods, where solid lines are the average losses over the network and the shadow area is the range between the minimum and the maximum loss of the networked agents. Our method achieves the minimum average loss and the learning performance is better than the non-cooperative case. Figure 5 shows the target's estimation as a function of time. It can be found that our method achieves smoother and more accurate estimations than the non-cooperative case, whereas the average and distance-based methods fail to learn the correct targets.

5.2 Digit classification

We consider a network of twenty agents performing digit classification tasks. We use a complete graph to model the network topology such that every agent is connected to all the other agents. An agent does not know which of its

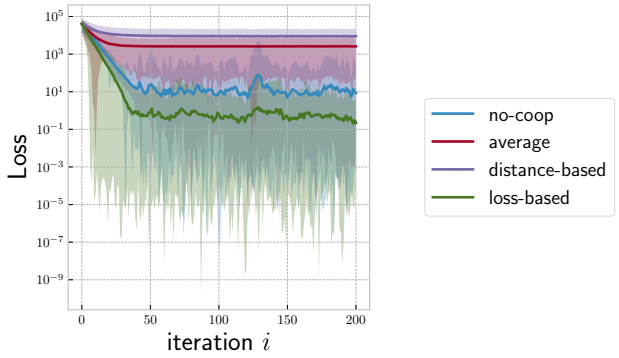


Fig. 4: Target localization: learning losses for different methods.

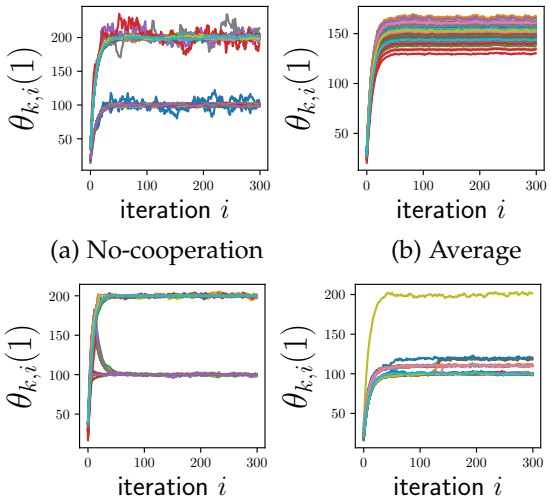


Fig. 5: Target localization: estimation $\theta_{k,i}$ (1st dimension) of every agent k (each line represents an agent) for different methods.

neighbors are performing the same task as the agent itself. Agents indexed by 0-9 have access to the MNIST dataset[2] [43] and agents indexed by 10-19 have access to the synthetic dataset[3] that is composed by the generated images of digits embedded on random backgrounds [44]. All the images are preprocessed to be 28×28 grayscale images. We use a CNN model of the same architecture for each agent and cross-entropy-loss. The CNN architecture we use is the same as in [31] and is listed in Table 1. We consider that agents have access to uneven sizes of training data such that each agent receives $100 - 1000$ training data and 200 testing data from the corresponding dataset for each iteration, similar to the real-world examples. We use mini-batch gradient descent with batch size of 64. The approximation $\widehat{r}_k(\cdot)$ is computed using the mini-batch loss. The step-size $\mu = 0.001$ is set for every agent.

The results are given in Figure 6 - Figure 7. From Figure 6, using the proposed method, the clustering structure is correctly learned and agents cooperate only with neighbors

[2]http://yann.lecun.com/exdb/mnist
[3]https://www.kaggle.com/prasunroy/synthetic-digits

in the same cluster and stop cooperating with the agents from the other cluster. In the case of distance-based clustering, agents do not cooperate with any neighbors and task is reduced to the non-cooperative case. Figure 7 shows the training and testing loss using different clustering methods, where solid lines are the average losses over the network and the shadow area is the range between the minimum and the maximum loss of the networked agents. The proposed method achieves the minimum average training and testing loss, and the learning performance is better than the non-cooperative case.

Table 2 lists the final learning losses for different methods for the two tasks. It is clear from the table that the proposed loss-based method achieves the best learning results, as measured by the final learning loss, for both tasks. In Table 3, we present the average precision, recall, and F1-scores for digit classification across multiple agents. These scores are calculated using micro averaging, a method that averages the true positives, false negatives, and false positives for each class in the multi-class classification problem run by each agent. We found that the proposed method outperforms the other methods in terms of precision, recall, and F1-scores for the digit classification task.

TABLE 1: CNN architecture for digit classification.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 28, 28]	320
ReLU-2	[-1, 32, 28, 28]	0
MaxPool2d-3	[-1, 32, 14, 14]	0
Conv2d-4	[-1, 64, 14, 14]	18,496
ReLU-5	[-1, 64, 14, 14]	0
MaxPool2d-6	[-1, 64, 7, 7]	0
Conv2d-7	[-1, 64, 7, 7]	36,928
ReLU-8	[-1, 64, 7, 7]	0
MaxPool2d-9	[-1, 64, 3, 3]	0
Linear-10	[-1, 128]	73,856
ReLU-11	[-1, 128]	0
Linear-12	[-1, 10]	1,290

TABLE 2: Final learning losses for different methods.

Method	Target Localization	Digit Classification
No-coop	8.431e0	9.396e-4
Average	8.906e3	2.702e-3
Distance-based	2.584e3	1.003e-3
Loss-based (ours)	**2.226e-1**	**7.911e-4**

TABLE 3: Final average precision, recall, and F1-scores for different methods for digit classification.

Method	Average Precision	Average Recall	Average F1-score
No-coop	0.8674	0.8653	0.8627
Average	0.8367	0.8251	0.8213
Distance-based	0.8721	0.8701	0.8678
Loss-based (ours)	**0.9227**	**0.9226**	**0.9190**

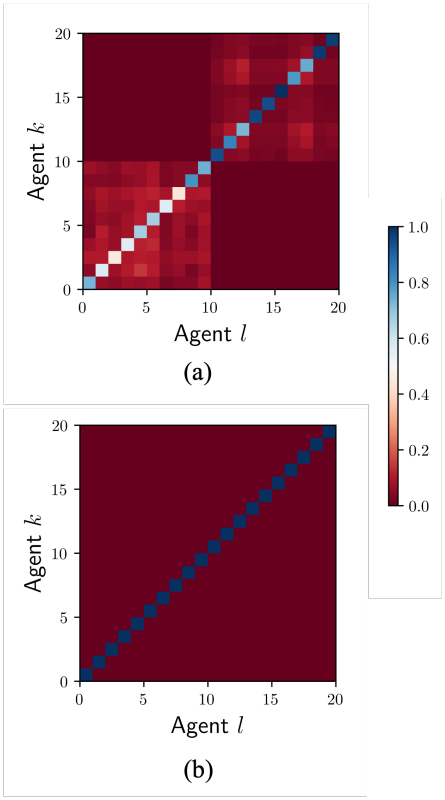


Fig. 6: Digit classification: average weight matrix over time $\frac{1}{T+1}\sum_{i=0}^{T} a_{lk}(i)$: (a) Loss-based; (b) Distance-based

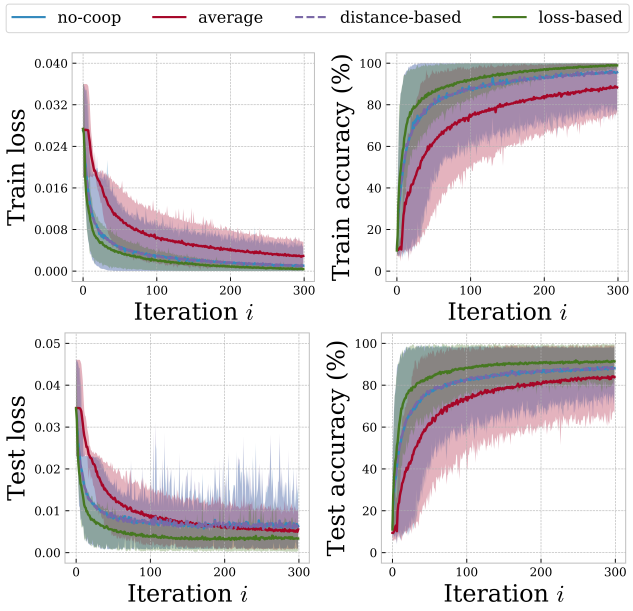


Fig. 7: Digit classification: learning performance for different methods.

6 Conclusion

Accurately capturing the clustering structure in a distributed multi-task learning network has great significance in practice. For example, advertising companies can use such information to target clusters of potential customers who may be interested in a particular product. This paper proposes an adaptive clustering method for distributed clustered multi-task learning network with fully decentralized agents. We prove the convergence of agents using the proposed method for clustering towards their objectives and analyze the learning performance. We also present a method of computing combination weights for aggregating the neighbors' model

parameters to approximately optimizes network learning performance. In the evaluation, we show that existing clustering methods fail to capture the correct clustering structure and result in worse learning performance. In contrast, the proposed method accurately learns the clustering structure and cooperation within such clusters improves the network learning performance compared to the case where agents do not cooperate with each other.

REFERENCES

[1] Dixon Vimalajeewa, Chamil Kulatunga, Donagh Berry, and Sasitharan Balasubramaniam. A service-based joint model used for distributed learning: Application for smart agriculture. *IEEE Transactions on Emerging Topics in Computing*, pages 1–1, 2021.

[2] Kai Yang, Tao Jiang, Yuanming Shi, and Zhi Ding. Federated learning via over-the-air computation. *CoRR*, abs/1812.11750, 2018.

[3] Tian Wang, Yan Liu, Xi Zheng, Hong-Ning Dai, Weijia Jia, and Mande Xie. Edge-based communication optimization for distributed federated learning. *IEEE Transactions on Network Science and Engineering*, 9(4):2015–2024, 2022.

[4] Guanghui Wen, Jian Qin, Xingquan Fu, and Wenwu Yu. DLSTM: distributed long short-term memory neural networks for the internet of things. *IEEE Transactions on Network Science and Engineering*, 9(1):111–120, 2022.

[5] Hui Lu, Chengjie Jin, Xiaohan Helu, Xiaojiang Du, Mohsen Guizani, and Zhihong Tian. Deepautod: Research on distributed machine learning oriented scalable mobile communication security unpacking system. *IEEE Transactions on Network Science and Engineering*, 9(4):2052–2065, 2022.

[6] Jakub Konecný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016.

[7] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S. Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4424–4434, 2017.

[8] Ali H. Sayed, Sheng-Yuan Tu, Jianshu Chen, Xiaochuan Zhao, and Zaid J. Towfic. Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior. *IEEE Signal Processing Magazine*, 30(3):155–171, 2013.

[9] Muhammed O. Sayin, N. Denizcan Vanli, Suleyman Serdar Kozat, and Tamer Basar. Stochastic subgradient algorithms for strongly convex optimization over distributed networks. *IEEE Transactions on Network Science and Engineering*, 4(4):248–260, 2017.

[10] Ali H. Sayed. Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7(4-5):311–801, 2014.

[11] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016.

[12] Adil Fahad, Najlaa Alshatri, Zahir Tari, Abdullah Alamri, Ibrahim Khalil, Albert Y Zomaya, Sebti Foufou, and Abdelaziz Bouras. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE Transactions on Emerging Topics in Computing*, 2(3):267–279, 2014.

[13] Dongkuan Xu and Yingjie Tian. A comprehensive survey of clustering algorithms. *Annals of Data Science*, 2(2):165–193, 2015.

[14] N Karthikeyani Visalakshi and K Thangavel. Distributed data clustering: a comparative analysis. In *Foundations of Computational Intelligence, Volume 6*, pages 371–397. Springer, 2009.

[15] Aditya Bhaskara and Maheshakya Wijewardena. Distributed clustering via lsh based data partitioning. In *International Conference on Machine Learning*, pages 570–579, 2018.

[16] Xiaochuan Zhao and Ali H. Sayed. Distributed clustering and learning over networks. *IEEE Transactions on Signal Processing*, 63(13):3285–3300, July 2015.

[17] Jie Chen, Cédric Richard, and Ali H. Sayed. Diffusion LMS for clustered multitask networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, pages 5487–5491, 2014.

[18] Jie Chen, Cédric Richard, and Ali H. Sayed. Diffusion LMS over multitask networks. *IEEE Transactions on Signal Processing*, 63(11):2733–2748, June 2015.

[19] Qing Shi, Feng Chen, Xinyu Li, and Shukai Duan. Distributed adaptive clustering learning over time-varying multitask networks. *Information Sciences*, 567:278–297, 2021.

[20] S. Khawatmi, A. M. Zoubir, and A. H. Sayed. Decentralized clustering over adaptive networks. In *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 2696–2700, Aug 2015.

[21] Sheng-Yuan Tu and Ali H. Sayed. Distributed decision-making over adaptive networks. *IEEE Transactions on Signal Processing*, 62(5):1054–1069, 2014.

[22] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[23] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Philip S. Yu. Learning multiple tasks with multilinear relationship networks. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems, Long Beach, CA, USA*, pages 1594–1603, 2017.

[24] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA*, pages 3994–4003, 2016.

[25] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple NLP tasks. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark*, pages 1923–1933, 2017.

[26] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA*, pages 7482–7491. IEEE Computer Society, 2018.

[27] Van Thiem Pham, Nadhir Messai, and Noureddine Manamanni. Impulsive observer-based control in clustered networks of linear multi-agent systems. *IEEE Transactions on Network Science and Engineering*, 7(3):1840–1851, 2020.

[28] Xiaojun Zhou, Yuan Gao, Chaojie Li, and Zhaoke Huang. A multiple gradient descent design for multi-task learning on edge computing: Multi-objective machine learning approach. *IEEE Transactions on Network Science and Engineering*, 9(1):121–133, 2022.

[29] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017.

[30] Roula Nassif, Stefan Vlaski, Cédric Richard, Jie Chen, and Ali H. Sayed. Multitask learning over graphs: An approach for distributed, streaming machine learning. *IEEE Signal Processing Magazine*, 37(3):14–25, 2020.

[31] Jiani Li, Waseem Abbas, and Xenofon D. Koutsoukos. Byzantine resilient distributed multi-task learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[32] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[33] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *21st European Symposium on Artificial Neural Networks, Belgium, April*, 2013.

[34] Yiqiang Chen, Jindong Wang, Chaohui Yu, Wen Gao, and Xin Qin. Fedhealth: A federated transfer learning framework for wearable healthcare. *CoRR*, abs/1907.09173, 2019.

[35] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[36] Xiaochuan Zhao and Ali H. Sayed. Clustering via diffusion adaptation over networks. In *2012 3rd International Workshop on Cognitive Information Processing*, pages 1–6, May 2012.

[37] Jie Chen, Cédric Richard, and Ali H. Sayed. Multitask diffusion adaptation over networks. *IEEE Transactions on Signal Processing*, 62(16):4129–4144, Aug 2014.

[38] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization

methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[39] Ali H. Sayed. Adaptive networks. *Proceedings of the IEEE*, 102(4):460–497, 2014.

[40] Abraham Berman and Robert J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*, volume 9 of *Classics in Applied Mathematics*. SIAM, 1994.

[41] Xi Peng, Jiashi Feng, Shijie Xiao, Wei-Yun Yau, Joey Tianyi Zhou, and Songfan Yang. Structured autoencoders for subspace clustering. *IEEE Transactions on Image Processing*, 27(10):5076–5086, 2018.

[42] Xi Peng, Yunfan Li, Ivor W. Tsang, Hongyuan Zhu, Jiancheng Lv, and Joey Tianyi Zhou. XAI beyond classification: Interpretable neural clustering. *Journal of Machine Learning Research*, 23:6:1–6:28, 2022.

[43] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[44] Prasun Roy, Subhankar Ghosh, Saumik Bhattacharya, and Umapada Pal. Effects of degradations on deep neural network architectures. *arXiv preprint arXiv:1807.10108*, 2018.

Jiani Li is currently working as a Research Scientist at Meta. She received her Ph.D. degree in Computer Science from the Electrical Engineering and Computer Science Department, Vanderbilt University, Nashville, TN, USA in 2021. Her research focuses on resilient multi-agent distributed systems and resilient design of cyber-physical systems with machine learning components.

Weihan Wang is a Ph.D. candidate in the Computer Science Department at Stevens Institute of Technology, NJ, USA. He is currently working as a research intern at OPPO US Research Center. His research interests lie in the area of simultaneous Simultaneous localization and mapping, Robotics, and multi-agent systems.

Waseem Abbas is an Assistant Professor in the System Engineering Department at the University of Texas at Dallas, TX, USA. Previously, he was a Research Assistant Professor at the Vanderbilt University, Nashville, TN, USA. He received Ph.D. (2013) and M.Sc. (2010) degrees, both in Electrical and Computer Engineering, from Georgia Institute of Technology, Atlanta, GA, and was a Fulbright scholar from 2009 till 2013. His research interests are in the areas of control of networked systems, resilience and robustness in networks, distributed optimization, and graph-theoretic methods in complex networks.

Xenofon Koutsoukos is a Professor and the Chair of the Department of Computer Science and a Senior Research Scientist with the Institute for Software Integrated Systems (ISIS), Vanderbilt University, Nashville, TN, USA. He received his PhD in Electrical Engineering from the University of Notre Dame in 2000, and was a Member of Research Staff at the Xerox Palo Alto Research Center (PARC) (2000–2002). His research work is in the area of cyber-physical systems with emphasis on learning-enabled systems, formal methods, distributed algorithms, security and resilience, diagnosis and fault tolerance, and adaptive resource management. He has published more than 300 journal and conference papers and he is co-inventor of four US patents. Prof. Koutsoukos was the recipient of the NSF Career Award in 2004, the Excellence in Teaching Award in 2009 from the Vanderbilt University School of Engineering, and the 2011 NASA Aeronautics Research Mission Directorate (ARMD) Associate Administrator (AA) Award in Technology and Innovation. He was named a Fellow of the IEEE for his contributions to the design of resilient cyber-physical systems.