Multilayer Active Learning for Efficient Learning and Resource Usage in Distributed IoT Architectures

Sasho Nedelkoski, Lauritz Thamsen, Ilya Verbitskiy, and Odej Kao
Technische Universitat Berlin, Germany
{nedelkoski, lauritz.thamsen, i.verbitskiy, odej.kao}@tu-berlin.de

***Abstract*—The use of machine learning modeling techniques enables smart IoT applications in geo-distributed infrastructures such as in the areas of Industry 4.0, smart cities, autonomous driving, and telemedicine. The data for these models is continuously emitted by sensor-equipped devices. It is usually unlabeled and commonly has dynamically-changing data distribution, which impedes the learning process. However, many critical applications such as telemedicine require highly accurate models and human supervision. Therefore, online supervised learning is often utilized, but its application remains challenging as it requires continuous labeling by experts, which is expensive. To reduce the cost, active learning (AL) strategies are used for efficient data selection and labeling.**

In this paper we propose a novel AL framework for IoT applications, which employs data selection strategies throughout the multiple layers of distributed IoT architectures. This enables an improved utilization of the available resources and reduces costs. The results from the evaluation using classification and regression tasks and synthetic as well as real-world datasets in multiple settings show that the use of multilayer AL can significantly reduce communication, expert costs, and energy, without a loss in model performance. We believe that this study motivates the development of new techniques that employ selective sampling strategies on data streams to optimize the resource usage in IoT architectures.

***Index Terms*—active learning, edge computing, internet of things, communication efficiency, resource utilization.**

I. Introduction

The recent trend of sensor-equipped IoT devices provides continuous observations and data measurements from the physical world, contributing to the generation of increasingly large volumes of data. IoT applications involving Industry 4.0, smart cities, autonomous driving, and telemedicine [1]–[3] utilize these large amounts of data and a variety of machine learning modeling techniques in order to extract valuable information. These IoT applications are often executed in geo-distributed computing environments. Sensor data are recorded far from the cloud data centers where the analysis is performed. However, the transmission of the data from remote sources to central cluster resources can lead to significant response times. To mitigate these effects, edge and fog computing have become important [4], [5]. With these geo-distributed computing architectures, the available resources between the data sources and clouds are utilized for the execution of parts of data processing pipelines. However, these distributed architectures also bring challenges and open new possibilities for performing machine learning tasks in a distributed manner. For example, an advantage is that the models can be trained on the collected data on seemingly unlimited cloud resources before they are transmitted to the edge devices and utilized for prediction on the data streams in near real-time [6].

At the same time, however, the edge devices such as wearables and mobile phones often record data where the distribution dynamically changes, which is referred to as concept drift [7]. In critical applications involving healthcare or manufacturing, a reduced model performance is not allowed, and thus some form of supervision is required [8]. However, the amount of data points that an expert can label in any given time is limited considering the cost of the labeling process.

Furthermore, many edge devices are limited in their communication due the required energy for data transmissions [9]. That is, the number of data points that can be sent is typically limited. Additionally, resources in IoT architectures that aggregate data streams from multiple edge devices can also become bottlenecks. Congestion occurs as a consequence of multiple applications sharing a single limited link and devices producing high-dimensional data streams (e.g., video) [10]. The labeling budget, communication-limited edge devices, and aggregated links naturally pose constraints in the geo-distributed and heterogeneous IoT environments.

One widely-accepted technique that reduces the cost of the data labeling process is the active learning (AL) [11]. By querying the class label of the most interesting samples based on previously seen data and some selection criteria, AL can produce an approximately optimal hypothesis while requiring a minimum number of labeled data.

We propose a multilayer AL framework that applies AL strategies throughout the layers of distributed IoT architectures by utilizing the sampling strategies. This application of AL addresses the described challenges by reducing the amount of data points that need to be labeled by an expert and thus reduces the data required to train a model. Therefore, the frameworks leads to a smaller amount of samples that need to be transmitted from near-source devices to the cloud.

The main contributions of this paper are as follows:

- We motivate the use of AL in distributed IoT architectures.

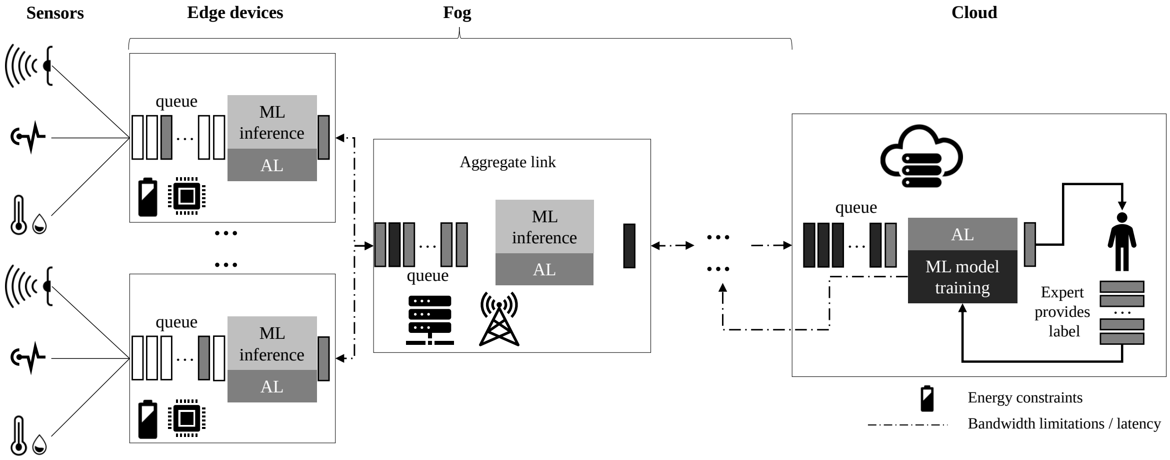

Fig. 1. Multilayer AL architecture for efficient learning in IoT environments.

- We present the multilayer AL framework, which addresses the resource and expert knowledge constraints.
- We show the feasibility of the proposed approach for classification and regression tasks in typical geo-distributed IoT architectures in a comprehensive empirical evaluation.

Outline. The rest of the paper is structured as follows. Section II explains our approach and a multilayer framework for the use of AL for efficient learning and resource usage in IoT applications. Section III presents our evaluation. Section V presents our conclusion.

II. Multilayer AL in IoT Environments

We describe a framework that implements data sampling on multiple layers of the IoT infrastructure. For this, we utilize AL, which is commonly used to train models efficiently with a small amount of selected samples. This is typically necessary as experts are not able to label every instance of data streams. The key idea of our approach is to employ the AL logic on multiple layers of distributed IoT architectures. In Figure 1, we show this multilayer use of AL. Regularly, we have a scenario where the data pass over multiple nodes, but for simplicity of explanation, we assume a three-layer architecture, where N edge devices send data to the cloud through a shared fog network node. At the beginning we assume an inaccurate model is trained on only few labeled data points and is deployed over all displayed levels of the architecture. Each sensor recording forms a feature vector composed of the measurements. In addition, the edge device executes preprocessing steps such as feature extraction. Furthermore, to reduce the latency of the prediction, we use the existing model on the device to compute the prediction. The model is inaccurate and requires more data for training. Due to the communication and energy constraints, the devices are not able to send all of the data to the cloud.

Without loss of generality, we formalize the device constraints as a period of time T_s^e measured in seconds, where it is allowed to send one data point toward the cloud resources. This formalization implies that we need to use a queue Q_e at the edge device, which holds the data samples for T_s^e seconds. To choose which data point needs to be send from Q, the framework can utilize any of the AL strategies for selective sampling [11]. As mentioned, the AL strategy always chooses the data point that is most likely to increase the accuracy of the model or in general to minimize the loss function, which we optimize. This process is repeated every T_s^e seconds to select one data point from the queue; therefore, we ensure that the constraints are satisfied.

In the framework, we consider the AL strategy as a blackbox, which the user can adapt for a particular IoT application. This function receives the queue, applies the sampling strategy, and returns one sample, which is sent to the upper-level resources in the architecture. Next, the already seen samples from the queue are deleted. In this manner, the upper bound on the queue size depends on T_s^e. If the current model is denoted as f, for binary classification problems, we proceed by predicting the class for each sample $x \in Q$. This reveals the probability for the instance x to belong to class 0. The most uncertain sample is then selected. On the other hand, for regression tasks, probabilistic models such as Gaussian processes and Bayesian regression provide confidence estimates at test points in the form of variance for a particular instance x. The most uncertain instance is that with the largest variance from the points in the queue. We then apply a similar logic to every upper layer.

To address the possible network limitations in terms of bandwidth and latency, similarly to the edge devices, we use the selective sampling strategy. We formalize the bandwidth/latency constraints as a time period T_s^f in which we are allowed to send only one sample. Given a queue Q_f, we use the AL strategy to choose the best data point to be forwarded to the cloud. The queue size and T_s^f depend on the available bandwidth and latency.

Once the data are transmitted to the cloud resources, the model needs to be updated. We formalize the budget for the expert that is utilized to provide the labels as a time period T_s^c

in which the expert can provide a single label. Similarly, the queue Q_c holds these instances at the cloud. The process starts when the AL strategy selects a sample from the queue Q_c that needs to be labeled, then the expert provides the label for the instance, and the labeled instance appended to the previously labeled data is used to update the model. Lastly, the model is propagated through the lower levels and the same process continues repeatedly.

Commonly, the AL selective sampling depends on the quality of the model. Consequently, T_s is a lower bound, which also depends on the model's accuracy. When the model is certain in its predictions, the time period after the data point needs to be send increases, which leads to even more efficient usage of the available communication and energy resources.

III. Evaluation

In this section, we describe the datasets and implementation details for the evaluation. Since the existence of labeled data from distributed IoT infrastructures is limited, for the evaluation we utilized well known datasets and distributed them across multiple devices. Further, we discuss the results of these experiments.

A. Datasets

1) Fall Detection Data: The UMA Fall dataset is composed of files that contain the mobility traces generated by a group of 19 experimental subjects which emulated a set of predetermined Activities of Daily Life (ADL) and falls [12].

On the edge device, we perform feature extraction out of a 15-seconds time window, which provided the following features: mean, standard deviation, zero crossings, and histogram computed in equal-sized bins.

2) ECG Data: The MIT-BIH Arrhythmia Database contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory [13].

As a preprocessing, we perform a device feature extraction with a simple three-layer pretrained convolutional neural network by taking the outputs of the last convolutional layer. This provided 256 features from the ECG channels per recording.

3) Synthetic classification data: The synthetic data are created using sci-kit learn, which adapts the algorithm from Guyon and is designed to generate the Madelon dataset [14]. The algorithm enables to generate a random two-class classification problem. Initially, it creates clusters of points normally distributed with $std = 1$ about vertices of an $n\_informative$-dimensional hypercube (the number of informative features) with sides with a length of 2*$class\_sep$ (larger values spread out the clusters/classes and facilitate the classification task) and assigns an equal number of clusters to each class. It can introduce interdependence between these features and adds various types of further noise to the data. Our synthetic dataset contains 150 features; 30 of them are informative.

4) Parkinson's disease telemonitoring: The dataset [15] contains a total of 5875 recordings from 42 subjects. We perform on device feature extraction, in this case, by generating first-order polynomial and interaction features, yielding 23 features.

B. Experiments

Each of the devices, a single aggregate link in the fog, and the cloud displayed in Figure 1 are implemented as separate processes in Python, which share data through bandwidth/latency configurable sockets. The AL logic is independent of the size of the distributed infrastructure as it is distributed over multiple layers and nodes; therefore, the framework achieves a high scalability. As machine learning models for evaluation, we used Random Forest and Bayesian Ridge for classification and regression tasks, respectively. We fixed the parameters for both models across all of the experiments in order to ensure consistency of the results:

- Random Forest: 50 tree estimators with a maximum depth of each tree equal to 5.
- Bayesian Ridge: 300 iterations for training.

For an evaluation of the scalability and changes in the accuracy and communication savings, we experimented with different numbers of edge devices, which record and send data constantly. Further, on the fog node, we configured a bandwidth limit on the ongoing link to the cloud. Lastly, the budget constraint for labeling exists in the cloud. The performances of the models on all datasets were evaluated by running each experiment three times for every combination of the parameters presented in Table I. In order to quantify the performance of the models we split the data into fixed training and test folds. After every iteration the scores are computed using the test datasets.

TABLE I
Parameter settings for the experiments

Name	Values
number of edge devices	1, 4, 8, 16
bandwidth in bytes per second	1000, 25000, 125000
labeling budget in seconds	0.4, 0.8, 2

TABLE II
Saved communication per iteration (KB)

BW (B/s)	Budget (s)	ECG	Fall	Synthetic	Regression
1000	0.4	966	958	468	31
1000	0.8	964	958	469	69
1000	2	964	961	479	184
25000	0.4	164	240	240	31
25000	0.8	305	462	375	70
25000	2	822	1070	842	183
125000	0.4	171	244	242	22
125000	0.8	340	447	384	49
125000	2	869	1068	721	132

Furthermore, we fixed the recording frequency to 100 Hz for each dataset and used them repeatedly in the cases without sufficient recordings. Each trial of the experiment for one of

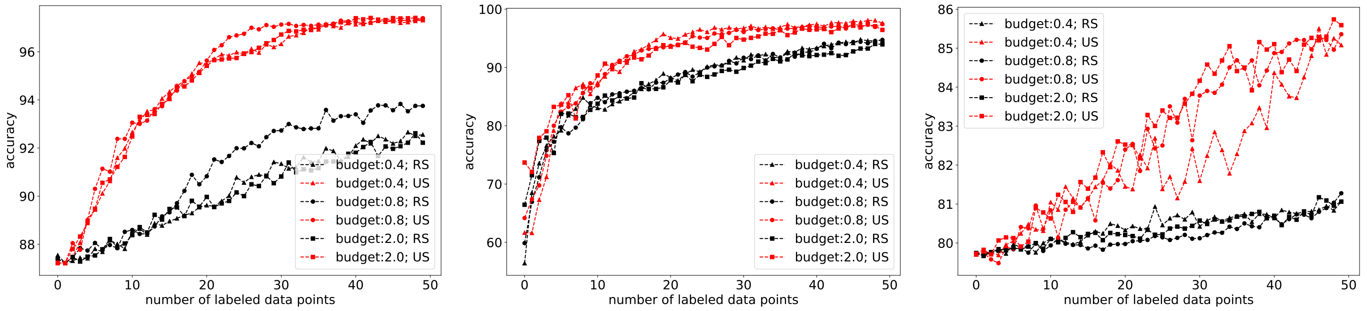

Fig. 2. Accurcy scores of the model in each labeling iteration. The scores are aggregated by the labeling budget for the three classification datasets: left-ECG arrhythmia, middle-Fall detection, right-Synthetic data.

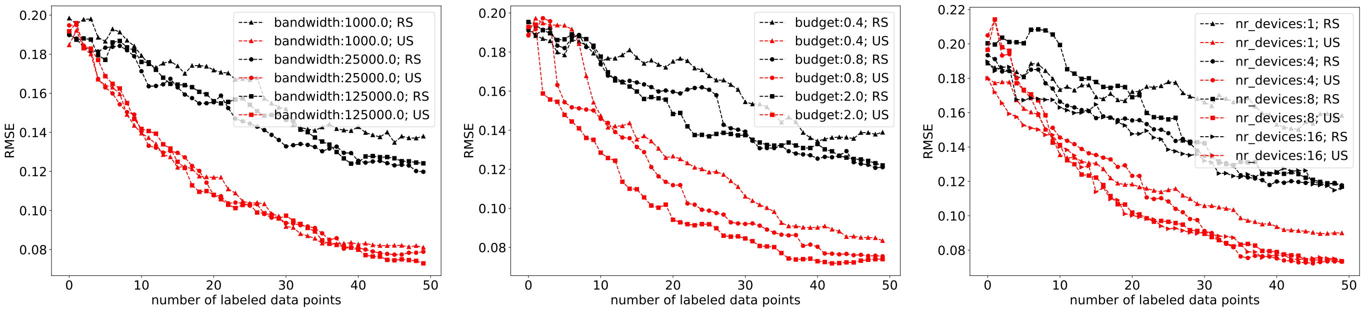

Fig. 3. RMSE scores for of the regression model for each labeling iteration, aggregated for different bandwidths (left), labeling budget (middle), and number of edge devices (right).

the given settings is terminated when maximum of 50 samples are labeled by the expert.

As a baseline approach, we utilized random sampling as a strategy for data selection using the same models, which is common practice in most AL studies [11]. In addition, we also carried out experiments to evaluate the number of data points to be labeled without using AL in order to reach the same accuracy of the model as that obtained by the uncertainty sampling. We discuss the obtained results and show the benefits of the approach. With the parameters in Table I, we simulated two different architectures. In one of them, the device can directly communicate to the cloud without bandwidth limitations over the aggregate link. In this case, there is only one device having a bandwidth of 125000. The AL logic is executed only on the device and cloud (the fog node carries out only the forwarding). In the second architecture, many devices send data without limitations, but congestion occurs in the aggregate node.

C. Results

In Figure 2, we show the results for the three classification datasets. The results of all of the experiments using the parameters in Table I are averaged by the labeling budget. When AL is utilized on the cloud to select incoming data points, we achieve a larger accuracy for the three datasets. Additionally, the decrease or increase in the labeling budget does not significantly affect the accuracy, which shows the robustness of the framework. A similar effect is achieved when random sampling is compared to an AL strategy for the regression dataset. Figure 3 shows the iterative accuracy results grouped by bandwidth in the aggregate node, budget, and number of devices. In all of the scenarios, the multilayer AL framework outperforms the baseline random sampling.

TABLE III
PERFORMANCE SCORE FOR DIFFERENT NUMBER OF DEVICES, WHEN THE BANDWIDTH/BUDGET IS FIXED

#Devices	ECG	Fall	Synthetic	Regression (RMSE)
1	97.3	96.6	85.7	0.089
4	97.4	95.7	85.1	0.072
8	97.4	98.5	86.1	0.074
16	97.3	98.4	84.3	0.073

Next, in Table II, we present the amount of saved communication per single iteration. One iteration is counted when the expert labels one data point at the cloud. The presented results are averaged by the available bandwidth in the aggregate link node and labeling budget. For all of the datasets, in each step, depending on the data dimensionality, we save a large amount of data that are not transmitted from the device and fog node. The largest savings in average occur when the bandwidth in the aggregate link has small values (in this case, the fog queue is large), which is expected.

We evaluated the setting where there is no bandwidth limitation on the aggregate link but only on the edge device. The previous discussion and results for the saved communication per iteration applies here as well, since the aggregate link node executes almost the same logic as the edge devices in terms of data selection. The total saved communication highly depends on the labeling budget and the resources available on the edge device that are represented by the period T_s^e.

Lastly, the scalability of the approach is evaluated by experimenting with different numbers of devices and fixed bandwidth/budget. As shown in Table III, the accuracy achieved for 50 labeled samples over the datasets does not decreases when the number of devices is increased, as the sampling is carried out separately at each device in a distributed manner.

All of the results show that the use of the multilayer selective sampling strategies is robust, scalable, and ultimately reduces the cost of expert knowledge. This directly addresses the savings in communication, which also implies savings in energy in the low-power devices.

IV. Related Work

The efficient learning from big unlabeled data in distributed IoT applications is a challenging task. To the best of our knowledge, no studies have been carried out on the problem of efficient learning under resource constraints in distributed IoT architectures utilizing AL strategies. We discuss below closely related studies.

Commonly, security threats to IoT applications needs to be addressed. For the security of a wireless IoT network, it is crucial to detect intrusions. Yang et al. [16] proposed a human-in-the-loop active learning approach for a wireless intrusion detection. By experimental examples, they showed a significant performance improvement of the active learning method over the traditional supervised learning approach in wireless communication architectures. Huang et al. [17] focused on a pool-based active learning using massive high-dimensional unlabeled data. They proposed two strategies that reduce the computation time by approximately 83% compared to that of traditional active learning.

Most previous active learning approaches consider the case of centralized processing, where all of the unlabeled data are supposed to be gathered together in one place. Shen et al. [18] proposed a distributed AL strategy to address the same problem in the increasing number of distributed applications considering that the data are distributed at different nodes over the network. They focus on distributed active learning for the classification problem and demonstrate the effectiveness of the approach on several real datasets.

V. Conclusion

This study addressed an important challenge for machine learning applications in distributed IoT architectures, which is to efficiently learn models from concept drifting data under communication and expert budget labeling constraints. We addressed this problem with the proposed multilayer AL framework, which propagates the logic of selective sampling through the layers of the distributed IoT architectures. Furthermore, we demonstrated the approach with concrete implementations for classification and regression problems in typical IoT architectures. The evaluation with synthetic and real-world data showed that the approach outperformed the baseline random sampling over all of the datasets. We achieved significant savings in communication in a range from 50% to 80%, while maintaining the same or achieving a better accuracy of the model over multiple settings and experiments, which also implies savings in energy.

Finally, we believe that this study is a general step towards new possibilities for selective sampling techniques for data streams in distributed IoT architectures and thus opens up many interesting research questions.

Acknowledgments

This work has been supported through grants by the German Ministry for Education and Research (BMBF) as Berlin Big Data Center BBDC (funding mark 01IS14013A and 01IS18025A).

References

[1] N. Jazdi, “Cyber Physical Systems in the Context of Industry 4.0,” in *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*. IEEE, May 2014, pp. 1–4.

[2] P. M. Kumar and U. D. Gandhi, “A novel three-tier internet of things architecture with machine learning algorithm for early detection of heart diseases,” *Computers & Electrical Engineering*, vol. 65, pp. 222–235, 2018.

[3] Y. Goldberg, “A primer on neural network models for natural language processing,” *Journal of Artificial Intelligence Research*, vol. 57, pp. 345–420, 2016.

[4] A. V. Dastjerdi and R. Buyya, “Fog computing: Helping the internet of things realize its potential,” *Computer*, vol. 49, no. 8, pp. 112–116, Aug 2016.

[5] M. Satyanarayanan, “The emergence of edge computing,” *Computer*, vol. 50, no. 1, pp. 30–39, Jan 2017.

[6] J. Ren, Y. Guo, D. Zhang, Q. Liu, and Y. Zhang, “Distributed and efficient object detection in edge computing: Challenges and solutions,” *IEEE Network*, vol. 32, no. 6, pp. 137–143, November 2018.

[7] L. Cohen, G. Avrahami-Bakish, M. Last, A. Kandel, and O. Kipersztok, “Real-time data mining of non-stationary data streams from sensor networks,” *Inf. Fusion*, vol. 9, no. 3, pp. 344–353, Jul. 2008.

[8] A. Holzinger, “Interactive machine learning for health informatics: when do we need the human-in-the-loop?” *Brain Informatics*, vol. 3, no. 2, pp. 119–131, 2016.

[9] B. Martinez, M. Monton, I. Vilajosana, and J. D. Prades, “The power of models: Modeling power consumption for iot devices,” *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5777–5789, 2015.

[10] P. Porambage, J. Okwuibe, M. Liyanage, T. Taleb, and M. Ylianttila, “Survey on multi-access edge computing for internet of things realization,” *IEEE Communications Surveys & Tutorials*, vol. PP, 06 2018.

[11] B. Settles, “Active learning,” *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[12] E. Casilari, J. A. Santoyo-Ramón, and J. M. Cano-García, “Umafall: A multisensor dataset for the research on automatic fall detection,” *Procedia Computer Science*, vol. 110, pp. 32–39, 2017.

[13] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, “Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals,” *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[14] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, “Result analysis of the nips 2003 feature selection challenge,” in *Advances in neural information processing systems*, 2005, pp. 545–552.

[15] K. Bache and M. Lichman, “UCI machine learning repository,” 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[16] K. Yang, J. Ren, Y. Zhu, and W. Zhang, “Active learning for wireless iot intrusion detection,” *CoRR*, vol. abs/1808.01412, 2018.

[17] E. Huang, H. Pao, and Y. Lee, “Big Active Learning,” in *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec 2017, pp. 94–101.

[18] P. Shen, C. Li, and Z. Zhang, “Distributed active learning,” *IEEE Access*, vol. 4, pp. 2572–2579, 2016.