Asynchronous Active Learning with Distributed Label Querying*

Sheng-Jun Huang, **Chen-Chen Zong**, **Kun-Peng Ning** and **Hai-Bo Ye**
College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
Collaborative Innovation Center of Novel Software Technology and Industrialization
MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing, 211106, China
{huangsj, chencz, ningkp, yhb}@nuaa.edu.cn

Abstract

Active learning tries to learn an effective model with lowest labeling cost. Most existing active learning methods work in a synchronous way, which implies that the label querying can be performed only after the model updating in each iteration. While training models is usually time-consuming, it may lead to serious latency between two queries, especially in the crowdsourcing environments where there are many online annotators working simultaneously. This will significantly decrease the labeling efficiency and strongly limit the application of active learning in real tasks. To overcome this challenge, we propose a multi-server multi-worker framework for asynchronous active learning in the distributed environment. By maintaining two shared pools of candidate queries and labeled data respectively, the servers, the workers and the annotators efficiently corporate with each other without synchronization. Moreover, diverse sampling strategies from distributed workers are incorporated to select the most useful instances for model improving. Both theoretical analysis and experimental study validate the effectiveness of the proposed approach.

1 Introduction

Supervised learning usually requires a large set of labeled data to train the prediction model. As the learning algorithms become more and more complicated, the required size of training set gets larger and larger. Meanwhile, labeling data examples is rather expensive, because the annotation process is usually time-consuming and needs high expertise in some difficult tasks [Settles, 2009; Huang *et al.*, 2010]. It is thus a significant challenge to learn with insufficient labeled data.

Active learning is a primary approach to overcome this challenge. It iteratively selects the most useful examples from the unlabeled dataset to query their labels from the oracle [Settles, 2009]. After adding the newly labeled data into the training set, the model can be updated to achieve better performance. The key task in active learning is how to accurately estimate the potential utility of an example on improving the performance, such that the model can be well trained with minimal queries. In the last decades, active learning has been extensively studied with many advanced methods proposed.

While existing active learning methods achieved great success in selecting the most useful instances, they commonly work in a synchronous way. Specifically, in each iteration of active learning, there are three steps performed in order. Firstly, the model is trained based on the currently labeled data. Then, the most useful examples are selected based on the trained model. At last, the labels of the selected instances are queried from the annotators. Each of the three steps depends on the previous step, which implies that they can not be performed asynchronously.

In real applications, there are usually many annotators working together to assign labels for the selected unlabeled instances. Especially in crowdsourcing environments, there are a huge number of online users providing the labeling service simultaneously. On the other hand, the model training and the instance selection are typically computationally intensive, which implies that it will take a long time to produce the candidate instances for labeling in each iteration. As a result, the annotators will have to wait for the next unlabeled example for a long time after each annotation. This will seriously hurt the user experience of annotators, decrease the annotation efficiency, and strongly limit the application of active learning. A more practical mechanism should allow annotators to continuously labeling data no waiting for model training and query selection.

In this paper, we propose a novel approach AAL for Asynchronous Active Learning. Specifically, the proposed approach works in a distributed environment with multiple servers and workers. The servers are responsible for training prediction models, while the workers focus on the active selection of the most useful instances. For each worker, it retrieves the most updated model from the servers, and selects the candidate queries from its own unlabeled data, independently to the other workers. The selected instances are then fed into a query pool shared by all workers. At the same time, all annotators retrieve the instances from the query pool and send them into the labeled pool after label-

*This work was supported by the National Key R&D Program of China (2020AAA0107000), NSFC (62076128) and the Fundamental Research Funds for the Central Universities (NE2019104). Sheng-Jun Huang and Hai-Bo Ye are the corresponding authors.

ing, which are further sent to servers for updating the models. In this way, the servers, the workers and the annotators work asynchronously, while the information is efficiently communicated across them via the two shared data pools. On one hand, the proposed approach eliminates the waiting of annotators by avoiding the synchronization between the three steps of active learning, and thus improves the labeling efficiency; on the other hand, under the multi-server multi-worker framework, the effectiveness of active selection is preserved by increasing the frequency of model updating and incorporating diverse sampling strategies. We perform experiments on multiple datasets with different classification models. Results demonstrate that the proposed approach can achieve comparable performance with synchronous methods while significantly improve the labeling efficiency.

The rest of this paper is organized as follows. We review related work in Section 2 and introduce the proposed method in Section 3. Section 4 reports the experiments, followed by the conclusion in Section 5.

2 Related Work

Active learning has received great research interests as a primary approach for learning with limited labeled data. The most important branch of research along this topic focuses on designing effective strategies to make sure that the selected instances can improve the model performance most [Fu *et al.*, 2013]. Among these approaches, some of them prefer to sample informative instances to reduce the model uncertainty [Lewis and Gale, 1994; Seung *et al.*, 1992; You *et al.*, 2014], while some others prefer to select representative instances to match the data distribution [Geman *et al.*, 1992; Roy and McCallum, 2001]. There are also some studies trying to combine informativeness and representativeness to achieve better performance [Settles and Craven, 2008; Huang *et al.*, 2010; Huang and Zhou, 2013]. Recently, there are a few attempts to learn the sampling strategy from data, instead of manually designing the selection criteria [Shao *et al.*, 2019; Yan *et al.*, 2020].

As the crowdsourcing becomes a popular setting in many real applications [Li *et al.*, 2021], active learning with multiple annotators has attracted more and more interests recently [Rodrigues *et al.*, 2014; Yan *et al.*, 2011]. Some studies try to select the most cost-effective annotators for the candidate queries, while some others focus on obtaining the ground-truth label from multiple noisy annotations [Zhao *et al.*, 2011]. While previous studies have noticed the importance of active learning in the crowdsourcing environment, they typically work in a synchronous way, and thus can be hardly applied to real tasks due to the serious latency between queries. Some attempts have been made to deal with this problem. Particularly, Authors in [Chakraborty, 2018] proposed a distributed active learning method for image recognition. This method formulates active sample selection as an optimization problem and uses submodular optimization techniques to solve the problem in a distributed setup. However, it only separates the query process from the training process and alternates with the update of the unlabeled video data, and cannot avoid the query latency because of the synchronization between model training and label querying.

Asynchronous mechanisms have been successfully incorporated in many machine learning scenarios [Baytas *et al.*, 2016; Mnih *et al.*, 2016; Terenin *et al.*, 2015]. In [Baytas *et al.*, 2016], an asynchronous distributed coordinate update method is adopted to perform full updates on multi-task learning model vectors. In [Mnih *et al.*, 2016], the author proposes a conceptually simple and lightweight framework for deep reinforcement learning that uses asynchronous gradient descent for optimization of deep neural network controllers. The method in [Terenin *et al.*, 2015] implements asynchronous gibbs sampling by ignoring sequential requirements.

Distributed learning is an advanced technology which usually works along with asynchronous learning [Sohn *et al.*, 2020]. However, these techniques are designed for some specific learning tasks, and cannot be directly applied to overcome the challenges in active learning.

3 The Proposed Approach

3.1 The Framework

In traditional active learning, the algorithm iteratively selects the most useful instances from the unlabeled pool to query their labels. In each iteration, there are in general three steps:

- Model training: the algorithm trains the prediction model by utilizing the currently labeled data.
- Instance selection: the sampling strategy selects the most useful instances based on a specific criterion.
- Label querying: the annotators assign class labels to the selected instances and add them into the labeled set.

This process repeats until enough labeled data queried. While existing active learning methods work in a synchronous way, these three steps depend on each other, and must be performed in order in each iteration. Obviously, in the synchronous learning setting, the selected instances reflect the demands of the latest model, and thus will contribute mostly to the model improving. On the other hand, the model is updated in time once the labeled set is enriched. However, synchronous active learning may suffer from serious latency between queries in real applications.

Nowadays, most data labeling tasks are completed online in a crowdsourcing environment. For example, in image classification tasks, to collect a large set of training data, the unlabeled images are firstly collected, and usually stored in a distributed system. At the same time, there could be thousands of online annotators to label the images simultaneously. In such case, there will be obvious shortcomings for synchronous active learning methods. Firstly, the model training of deep neural networks could be computationally costly, and thus will lead to a long time for selecting the candidate queries after each annotation. As a result, most annotators will have to wait for a long time to receive the next image after each annotation. This will seriously affect the user experience of annotators and significantly decrease the labeling efficiency. Moreover, the synchronous active learning methods try to select the best instances globally from the whole

dataset, and thus need to scan all of the examples on the distributed system. This process will further decrease the efficiency of the label querying.

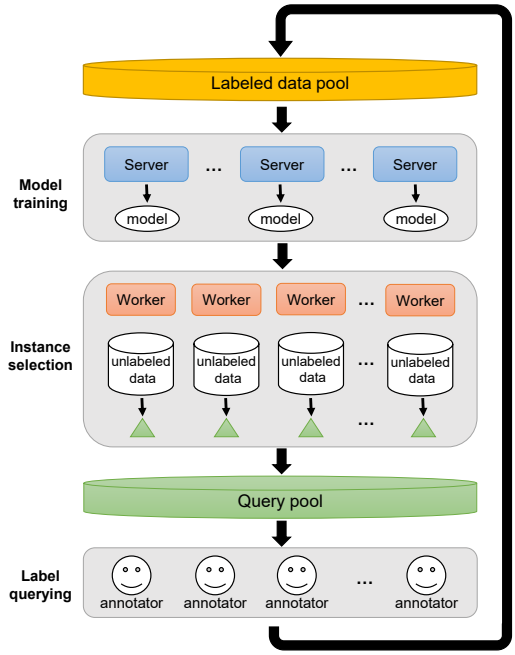


Figure 1: The framework of the proposed approach.

To overcome the above shortcomings, we propose a new approach for asynchronous active learning (AAL for short). The framework is summarized in Figure 1. This framework also consists of the three components of model training, instance selection and label querying, but allows them to work asynchronously without waiting each other. In the AAL framework, we maintain three data pools L, D and S. Specifically, $L = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_{n_l}, y_{n_l})\}$ is the set of n_l labeled examples. $D = D_1 \cup \cdots \cup D_k$ denotes the set of all unlabeled examples, which is a union of k disjoint subsets distributed on k workers respectively. S is the set of selected instances to be queried. There are a set of m servers in the system for model training and k workers for instance selection.

In the model training component, each server tries to independently learn a prediction model from the latest labeled set L. Formally, for the i-th server, we have

$$g_i = \mathcal{A}(L), \tag{1}$$

where $\mathcal{A}$ denotes the algorithm for learning the prediction model, and g_i is the trained model. Note that the algorithm $\mathcal{A}$ varies among servers, i.e., different servers may employ different algorithms. In this paper, we fix $\mathcal{A}$ for all servers for simplicity. During the learning phase, all the servers simultaneously update the models. For a specific server, once its training is finished, it examines the labeled set L, and will launch another round of model updating if L receives some newly labeled data compared to the previous iteration.

In the instance selection component, each worker also works independently to the others, and tries to select the most useful instances from its own unlabeled data. Firstly, it retrieves the latest updated model from $\{g_1, g_2, \cdots, g_m\}$, which is denoted by g^*. As the labeled data is increasing, it can be expected that newer model is better than the old ones. Note that the workers directly retrieve the latest updated model from the servers for active selection, and does not restrict that g^* has utilized all the currently labeled instances. In such a way, the workers do not need to wait the model training when performing instance selection. After that, an active sampling strategy is employed to select the most useful batch of instances from its own unlabeled dataset:

$$S_j = \mathcal{S}_j(g^*, D_j), \tag{2}$$

where $\mathcal{S}_j$ is the active sampling algorithm of the j-th worker, which takes g^* and D_j as the inputs and outputs the selected instance set S_j. Similarly, different workers may use different active sampling strategies for diverse selection. From Figure 1, it can be observed that the candidate query set S is shared by all workers. In other words, all the workers perform active selection to jointly produce the query set. Formally, we have

$$S = S_1 \cup S_2 \cup \cdots \cup S_k. \tag{3}$$

At last, in the label querying component, the annotators randomly pick instances from the query set S, and add them into the labeled set L after labeling. It is easy to observe that if we maintain enough workers in the system, then it can be guaranteed that the query set S will be never empty, which means that all the annotators can work continuously without latency between queries.

In summary, the three components work in an asynchronous way, and communicate with each other efficiently via the two shared data pools. With the proposed AAL framework, the labeling efficiency can be significantly improved by avoiding the synchronization among the internal steps of active learning.

3.2 Multi-Server Multi-Worker Implementation

The asynchronous mechanism significantly improves the efficiency of active learning. But meanwhile, it brings the risk of decreasing the cost-effectiveness of labeling. In the proposed AAL framework, the workers retrieve the latest updated model from the servers, and employ it to actively select the instance for label querying. However, it is not guaranteed that the latest updated model has utilized all of the labeled data without synchronization between the servers and workers. Subsequently, the selected instances may be not the most useful for improving the performance when they are used to update the model.

To achieve both efficiency and effectiveness, on one hand, we propose to employ different sampling strategies for multiple workers to enhance diversity of the active selection; and on the other hand, the model updating frequency is increased to fully utilize the newly labeled data based on multiple servers. Next, we will introduce these two strategies in detail respectively.

Combining different selection criteria for active sampling has been validated to be effective in some previous studies [Demir and Bruzzone, 2014; Wang and Kwong, 2014]. Different selection criteria estimate the potential utility of an

instance from different aspects. By incorporating them together, it is expected to identify the most useful instances more accurately. Intuitively, simultaneously considering different criteria may enhance the diversity of labeled examples and reduce the redundant information, and thus may lead to better performance. In the proposed asynchronous active learning framework, we have multiple workers to perform active selection independently, which provides a straightforward way to enhance the sampling diversity. First of all, the unlabeled data is distributed on multiple workers, which naturally divides the dataset into different groups. In real applications, the data may be collected in different batches with different distributions. In such cases, independently selection from each worker will obtain balanced information from different distributions. In our experiments, we simply divide the unlabeled data into subsets with random partition. One may employ clustering methods to achieve better partitions. In addition, we further enhance the diversity by using different selection strategies in different workers. Specifically, three commonly used strategies, i.e., entropy based sampling [Holub *et al.*, 2008], least confidence sampling [Li and Sethi, 2006] and margin based sampling [Balcan *et al.*, 2007] are employed in our experiments. It is worthy to note that in our experiments, we focus on examining the effectiveness of the asynchronous mechanism instead of the sampling strategy. We thus choose to use the classical strategies because they have been validated to achieve robust performance in many tasks. One may also employ other advanced strategies in our framework, which may lead to better performance.

Next we discuss the multi-server implementation for increasing the model updating frequency. In synchronous active learning, the model is timely updated once a new labeled example comes. It ensures that the model can fully utilize all the supervised information and subsequently the selected instances reflect the demands of the current model. Unfortunately, this property does not remain in the asynchronous active learning setting. Instead, we try to update the models more frequently based on the multi-server mechanism, and thus to exploit the labeled data as timely as possible. Imagine a scenario with three servers, and the time cost of model training is three times of that for querying a label. For the single server case, the first 4 queries are based on the model of version 0. After that, the model is updated after every 3 queries. In contrast, for the multi-server case with 3 servers, the first 4 queries also share the model of version 0, while the following queries always receive a new model. Obviously, the multi-server setting leads to a much higher frequency for model updating, and thus can select the instances for labeling more effectively.

The complete pseudo code of AAL execution process can be found in the supplementary file.

3.3 Theoretical Analysis

In this subsection, we derive an upper bound for the expected risk of the classification model trained with the samples queried with the proposed AAL approach. We denote by f the ground-truth labeling function, and $\mathcal{H}$ a set of hypotheses, where each hypothesis $h \in \mathcal{H}$ is a mapping function from the instance space to the label space. Firstly, we follow [Redko *et al.*, 2017] to give the following definition.

Definition 1. *Given a convex loss function ℓ, the expected risk of the hypothesis h with regard to the data distribution $\mathcal{D}$ is defined as the probability that h disagrees with the labeling function f:*

$$\epsilon_{\mathcal{D}}(h, f) = \mathbb{E}_{x\sim\mathcal{D}}[\ell(h(x), f(x))]. \tag{4}$$

For convenience, we use a simplified denotation $\epsilon_{\mathcal{D}}(h) = \epsilon_{\mathcal{D}}(h, f)$. We denote by $\mathcal{D}$ the generation distribution of the whole data and $\mathcal{Q}$ the distribution of data queried by our algorithm. Following the results in [Redko *et al.*, 2017], we have Lemma 1 to estimate the expected risk of the hypothesis h.

Lemma 1. *Assume that the loss function ℓ is convex, symmetric, bounded, obeys the triangular equality and 1-Lipschtiz. Then the expected risk $\epsilon_{\mathcal{D}}(h)$ can be bounded by:*

$$\epsilon_{\mathcal{D}}(h) \leq \epsilon_{\mathcal{Q}}(h) + W_1(\mathcal{D}, \mathcal{Q}) + \lambda. \tag{5}$$

where $\lambda = \epsilon_{\mathcal{D}}(h^*) + \epsilon_{\mathcal{Q}}(h*)$, h^* is the optimal hypothesis leads to minimal value of the summation error over $\mathcal{D}$ and $\mathcal{Q}$.

The proof of Lemma 1 can be found in the supplementary file. From Lemma 1, we know that the expected risk of h over distribution $\mathcal{D}$ is upper bounded by the expected risk over the distribution $\mathcal{Q}$ of queried data, the Wasserstein distance $W_1(\mathcal{D}, \mathcal{Q})$ between the two distributions and λ. In our setting, the queried data consists of examples selected from workers with different sampling strategies. Formally, we notate it as follows:

$$\mathcal{Q} = \mathcal{Q}_1 \vee \mathcal{Q}_2 \vee ... \vee \mathcal{Q}_k \tag{6}$$

where $\mathcal{Q}_i$ denotes the distribution of data sampled by the i-th worker. Then we can further get the following theorem.

Theorem 1. *Let $\hat{\mathcal{Q}}, \mathcal{Q}^*$ be the distributions of data selected by the worst and the best query strategies among all workers, respectively. The expected risk $\epsilon_{\mathcal{D}}(h)$ can be bounded by:*

$$\begin{aligned}\epsilon_{\mathcal{D}}(h) \leq & \lambda + \epsilon_{\mathcal{Q}}(h) + \mathbb{E}_{x\sim\mathcal{D}}\ell(h(x), h^*(x)) - \mathbb{E}_{x\sim\mathcal{Q}}\ell(h(x), h^*(x)) \\ \leq & \hat{\lambda} + \epsilon_{\hat{\mathcal{Q}}}(h) + W_1(\mathcal{D}, \mathcal{Q}^*)\end{aligned}$$

where $\hat{\lambda} = \epsilon_{\mathcal{D}}(h^*) + \epsilon_{\hat{\mathcal{Q}}}(h^*)$, and h^* is the hypothesis leads to the minimal value of $\epsilon_{\mathcal{D}}(h) + \epsilon_{\hat{\mathcal{Q}}}(h)$.

The proof of Theorem 1 can be found in the supplementary file. Intuitively, based on Theorem 1, if we approximate the distance between the two distributions as $\epsilon_{\hat{\mathcal{Q}}}(h) - \epsilon_{\hat{\mathcal{D}}}(h)$, then it can be easily concluded that the expected risk of the final model will be upper bounded by the one trained with the worst sampling strategy among all workers. This implies that if we launch decent sampling strategies on each worker, it can be expected that the asynchronous active learning will improve the model effectively.

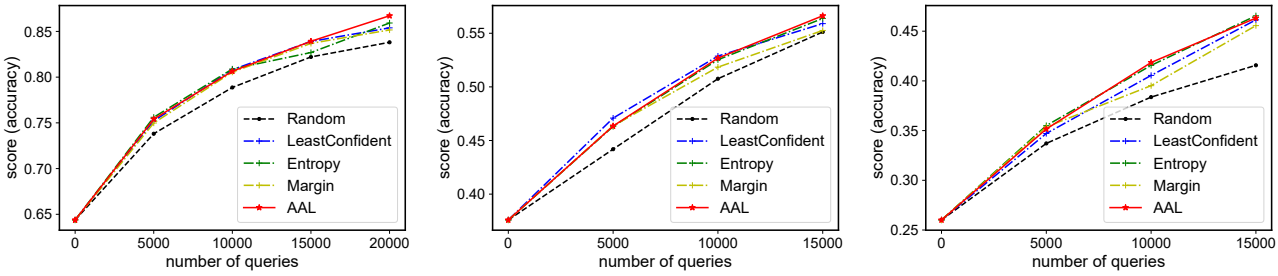
4 Experiments

4.1 Settings

The experiments are performed on three data sets: Cifar-10, Cifar-100 and mini-ImageNet. mini-ImageNet is a subset of the ImageNet dataset. It samples 600 images from each of 100 classes to form a set with 60,000 images. We randomly

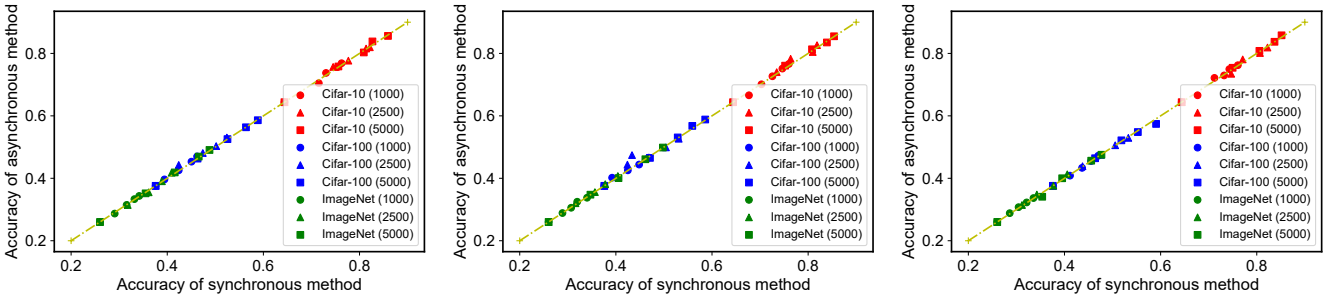
Method		Entropy			Least Confident			Margin			AAL		
#Servers		1	2	4	1	2	4	1	2	4	1	2	4
Dataset	Cifar-10	0.819	0.832	0.841	0.817	0.833	0.845	0.818	0.834	0.844	0.818	0.833	0.843
	Cifar-100	0.353	0.362	0.369	0.356	0.370	0.383	0.349	0.357	0.365	0.357	0.365	0.371
	ImageNet	0.386	0.399	0.408	0.388	0.397	0.406	0.379	0.389	0.399	0.394	0.403	0.410

Table 1: The average performance of different methods with different numbers of servers.



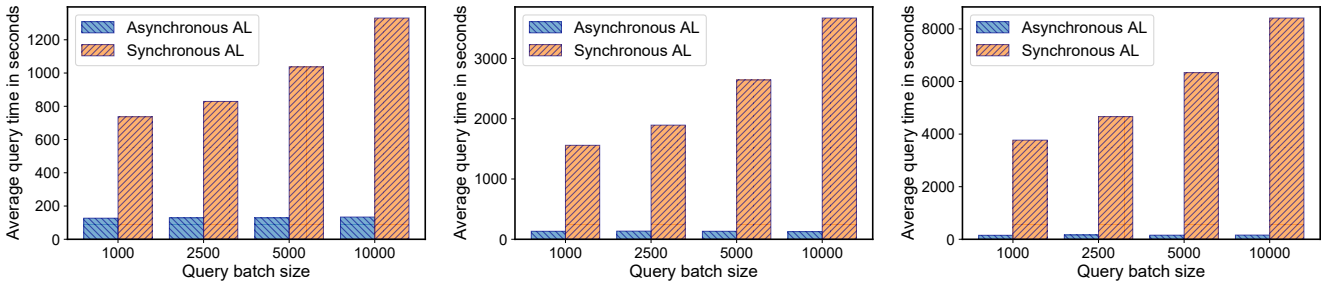
(a) Cifar-10 with batch size of 5000 (b) Cifar-100 with batch size of 5000 (c) mini-ImageNet with batch size of 5000

Figure 2: Performance comparison of different methods with a single server.



(a) Entropy based sampling (b) Least confidence sampling (c) Margin based sampling

Figure 3: Performance comparison between synchronous and asynchronous mechanisms with different sampling strategies.



(a) Cifar-10 (b) Cifar-100 (c) Mini-ImageNet

Figure 4: Comparison of average query time between synchronous and asynchronous methods on different datasets.

sample 50,000 images as the training set and the rest 10,000 images as test set. The images are resized into 84×84.

Unlike most active learning studies that focus on designing new selection criteria, the key contribution of this work is to reduce query latency with the asynchronous framework. We thus employ some classical and robust selection criteria to validate the effectiveness of the asynchronous querying mechanism. Therefore, in the experiments, we compare with the following sampling strategies:

- Random sampling: it selects instances at random.
- Entropy base sampling: it selects the instances with largest entropy of the predictions [Holub *et al.*, 2008].

- Least Confidence sampling: it selects the instances with the lowest prediction confidence [Li and Sethi, 2006].
- Margin based sampling: it selects the instances with minimal margins [Balcan *et al.*, 2007].
- AAL: the proposed approach. It employs the entropy, least confidence and margin methods on different workers to select the instances.

For the compared baseline methods, we also implement the asynchronous version with the proposed framework. It is worthy to note that we focus on validating the effectiveness of the proposed asynchronous mechanism, but not the selection criterion itself. So we choose to implement the commonly used strategies. We launch 10 workers in all experiments, and their sampling strategies are randomly assigned with entropy, least confidence and margin based methods. VGG-16 is employed as the classification model for all experiments. In the following subsections, the experiments are performed in the single-server and multi-server settings, respectively.

4.2 Results with Single Server

We first examine the effectiveness of the asynchronous mechanism with a single-server setting. For each dataset, we examine the results with the query batch size varies from $\{1000, 2500, 5000\}$. A larger or smaller query batch size is allowed, here is just to facilitate experimentation. Figure 2 presents the comparison results of different methods when batchsize is 5000. The remaining experimental results can be found in the supplementary file. Each subfigure plots the accuracy curves.

First of all, we observe that the random sampling losses its edge to other active sampling methods on all datasets, which is as expected. The performances of baseline methods, i.e., Entropy, Least confidence and Margin, are generally mixed. A specific method may achieve better performance on some datasets while worse performance on others. The proposed AAL method achieves decent performance in most cases. It is comparable to the best one among the baselines. Noticing that our method does not requires the synchronization between model training and label querying, these results validate that the proposed asynchronous mechanism can still achieve effective performance.

We further perform experiments to compare the synchronous and asynchronous active learning more intuitively. For the active sampling strategies, entropy sampling, least confidence sampling and margin sampling, we implement two versions for each of them, one in a synchronous way and the other in an asynchronous way as proposed in our AAL framework. Then on each of 3 datasets, we run both the 2 versions to record their performance with 3 different batch sizes. So after the experiments, we have in all $2 \times 3 \times 3 = 18$ performance records for each sampling strategy. Based on these records, we show the scatter plots in Figure 3, each subfigure corresponds to a sampling strategy. The horizontal axis represents the accuracy of synchronous method while the vertical axis represents the accuracy achieved by asynchronous method. A marker point above the diagonal line indicates that the asynchronous method achieves better performance than the synchronous method on the corresponding dataset and with the specific batch size. From the figures we can observe that all the points are closely distributed along the diagonal line, indicating that the synchronous and asynchronous methods achieve comparable performance. This phenomenon validates that although the proposed AAL framework relax the synchronization, it still can performs competitively with the synchronous method.

At last, we examine the key advantage of the asynchronous active learning approach, i.e., whether the proposed method can reduce the query latency by avoiding the synchronization among the different components. In Figure 4, we present the average time between two queries for both synchronous and asynchronous methods on different datasets. It can be observed that the proposed asynchronous active learning approach is much more efficient than the synchronous method in all cases. The time cost of synchronous method increases significantly when the query batch size gets larger. In contrast, the time cost of the proposed AAL method remains low for large batch sizes.

4.3 Results with Multiple Servers

In this subsection, we further examine the effectiveness of the multi-server setting. As discussed previously, by using multiple servers to simultaneously training the prediction models, the frequency of model updating could be increased. And thus it is expected that the selected instances could be more useful for improving the model. We perform the experiments with 1, 2 and 4 servers for all of the four active learning strategies, respectively. Note that the entropy, least confidence and margin method are all implemented in the asynchronous way with the proposed framework. Due to space limitation, the performance curves of 4 methods on 3 datasets with different numbers of servers are plotted in the supplementary file.

Here we calculated the average accuracy over the whole performance curves for each case, and summarized the results in Table 1. It can be observed that the performances are consistently improved by using more servers for all cases. These results validate that by utilizing multiple servers to update the models more frequently, the active selection is more effective as it captures the demands of models more timely.

5 Conclusion

In this paper, we propose a novel framework with multi-server multi-worker structure for asynchronous active learning. On one hand, the servers independently training the prediction models with a increasing update frequency; on the other hand, the workers perform active selection from their own unlabeled data with diverse sampling strategies. By maintaining two shared pools of candidate queries and labeled data, the three internal components of active learning, i.e., model training, instance selection and label querying, can work efficiently without synchronization with each other, and thus avoid the serious query latency of existing active learning methods. Both theoretical analysis and extensive experiments demonstrate that the proposed approach can achieve both effectiveness and efficiency. In the future, we plan to incorporate more advanced sampling strategies in the AAL framework, and examine the performance of the proposed asynchronous mechanism with more workers on larger datasets.

References

[Balcan *et al.*, 2007] Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *International Conference on Computational Learning Theory*, pages 35–50. Springer, 2007.

[Baytas *et al.*, 2016] Inci M Baytas, Ming Yan, Anil K Jain, and Jiayu Zhou. Asynchronous multi-task learning. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 11–20. IEEE, 2016.

[Chakraborty, 2018] Shayok Chakraborty. Distributed active learning for image recognition. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1833–1841. IEEE, 2018.

[Demir and Bruzzone, 2014] Beguem Demir and Lorenzo Bruzzone. A multiple criteria active learning method for support vector regression. *Pattern recognition*, 47(7):2558–2567, 2014.

[Fu *et al.*, 2013] Yifan Fu, Xingquan Zhu, and Bin Li. A survey on instance selection for active learning. *Knowledge and information systems*, 35(2):249–283, 2013.

[Geman *et al.*, 1992] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

[Holub *et al.*, 2008] Alex Holub, Pietro Perona, and Michael C Burl. Entropy-based active learning for object recognition. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE, 2008.

[Huang and Zhou, 2013] Sheng-Jun Huang and Zhi-Hua Zhou. Active query driven by uncertainty and diversity for incremental multi-label learning. In *2013 IEEE 13th International Conference on Data Mining*, pages 1079–1084. IEEE, 2013.

[Huang *et al.*, 2010] Sheng-Jun Huang, Rong Jin, and Zhi-Hua Zhou. Active learning by querying informative and representative examples. In *Advances in neural information processing systems*, pages 892–900, 2010.

[Lewis and Gale, 1994] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer, 1994.

[Li and Sethi, 2006] Mingkun Li and Ishwar K Sethi. Confidence-based active learning. *IEEE transactions on pattern analysis and machine intelligence*, 28(8):1251–1261, 2006.

[Li *et al.*, 2021] Shao-Yuan Li, Sheng-Jun Huang, and Songcan Chen. Crowdsourcing aggregation with deep bayesian learning. *SCIENCE CHINA-INFORMATION SCIENCES*, 64(3), 2021.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[Redko *et al.*, 2017] Ievgen Redko, Amaury Habrard, and Marc Sebban. Theoretical analysis of domain adaptation with optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 737–753. Springer, 2017.

[Rodrigues *et al.*, 2014] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. Gaussian process classification and active learning with multiple annotators. In *International conference on machine learning*, pages 433–441, 2014.

[Roy and McCallum, 2001] N Roy and A McCallum. Toward optimal active learning through sampling estimation of error reduction. int. conf. on machine learning, 2001.

[Settles and Craven, 2008] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, 2008.

[Settles, 2009] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[Seung *et al.*, 1992] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294, 1992.

[Shao *et al.*, 2019] Jingyu Shao, Qing Wang, and Fangbing Liu. Learning to sample: an active learning framework. *arXiv preprint arXiv:1909.03585*, 2019.

[Sohn *et al.*, 2020] Jy-yong Sohn, Dong-Jun Han, Beongjun Choi, and Jaekyun Moon. Election coding for distributed learning: Protecting signsgd against byzantine attacks. *Advances in Neural Information Processing Systems*, 33, 2020.

[Terenin *et al.*, 2015] Alexander Terenin, Daniel Simpson, and David Draper. Asynchronous gibbs sampling. *arXiv preprint arXiv:1509.08999*, 2015.

[Wang and Kwong, 2014] Ran Wang and Sam Kwong. Active learning with multi-criteria decision making systems. *Pattern Recognition*, 47(9):3106–3119, 2014.

[Yan *et al.*, 2011] Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G Dy. Active learning from crowds. 2011.

[Yan *et al.*, 2020] Yi-Fan Yan, Sheng-Jun Huang, Shaoyi Chen, Meng Liao, and Jin Xu. Active learning with query generation for cost-effective text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6583–6590, 2020.

[You *et al.*, 2014] Xinge You, Ruxin Wang, and Dacheng Tao. Diverse expected gradient active learning for relative attributes. *IEEE transactions on image processing*, 23(7):3203–3217, 2014.

[Zhao *et al.*, 2011] Liyue Zhao, Gita Sukthankar, and Rahul Sukthankar. Robust active learning using crowdsourced annotations for activity recognition. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.