Received January 22, 2020, accepted March 3, 2020, date of publication March 9, 2020, date of current version March 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2979315

Batch Active Learning With Two-Stage Sampling

RONGHUA LUO AND XIANG WANG
Guangzhou Higher Education Mega Centre, School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding authors: Ronghua Luo (rhluo@scut.edu.cn) and Xiang Wang (seanwong94@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61372140, and in part by the Guangzhou Science and Technology Project under Grant 201802010073.

ABSTRACT Due to its effectiveness in training precise model using significant fewer labeled instances, active learning has been widely researched and applied. In order to reduce the time complexity of active learning so that the oracle need not wait for the algorithm to provide instance in labeling, we proposed a new active learning method, which leverages batch sampling and direct boundary annotation with a two-stage sampling strategy. In the first stage sampling, the initial seed, which determines the location of boundary annotation, is selected with reject sampling based on the clustering structure of instances to ensure the initial seeds can approximate the distribution of data and with high diversity. In the second stage sampling, by treating the instance sampling as the selection of representative in a local region and maximizing the rewards that can get from selecting a instance as the new representative, we proposed a novel mechanism to maintain local representativeness and diversity of query instances. Compared with the conventional pool-based active learning method, our proposed method does not need to train the model in each iteration, which reduces the amount of calculation and time consumption. The experimental results in three public datasets show that the proposed method has comparable performance with the uncertainty-based active learning methods, which proves that the sampling mechanism in our method is effective. It performs well without retraining the model in each iteration and does not rely on the precision of the model.

INDEX TERMS Active learning, boundary annotation, instance sampling, generative adversarial model.

I. INTRODUCTION

It is well known that the performance of fully supervised learning methods such as deep convolution neural networks (DCNN) strongly depends on a large amount of well annotated data. However, to label a vast amount of data is expensive and time-consuming. Especially in some fields such as medical image analysis and biological information processing, it even requires the oracle to have professional knowledge. Therefore, it is a practical problem to reduce the cost of data annotation. Many methods including semi-supervised learning [1], transfer learning [2] and few shot learning [3] have been proposed for this purpose. Unlike these methods that only use the pre-marked samples in model training, active learning tries to query the most valuable and important unlabeled samples iteratively from the underlying distribution, so that significant fewer training samples are needed in model training. Since active learning algorithm simulates the human learning process to some extent: paying more attention on important instances and learning in a iterative way, it has attract much attention in recent years, and has been widely used in information retrieval [4], [5], object detection [6], speech recognition [7], and text analysis [8]–[10].

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng Xiao.

Active learning generally consists of two modules: query module and training module. In each iteration, the query module draws the most valuable samples that can effectively improve model performance, and then requests the human oracle to label the samples. The training module is responsible for updating the model using the newly labeled samples, so that it can select more desirable unlabeled samples in the next iteration. Although the iterative retrain of model with newly labeled samples can improve the ability of drawing valuable samples, model update and sample selection are quite time-consuming, especially in the case of training a complex model with high-dimension data. The time of waiting for the model to select samples is even longer than that of labeling samples, which is very incommodious when the oracle is human expert. The batch mode active learning [11], which selects a subset of unlabeled samples at each iteration, can solve this problem to some extent. But selecting a batch of best samples for model training is much more difficult than selecting one best sample at each iteration. And the performance of most of the current batch sampling methods in active learning depends heavily on the precision of the model. However, retraining the model using newly annotated instances is the most time comsuming step in active learning. Therefore, besides the batch sampling mode we should consider to simplize the training process with some kind of additional annotation information from the oracle. Annotating the decision boundary along with sample lableling, which has been adopted in [12], is a possible solution. Huijser and van Gemert [12] has proved that the additional annotation of decision boundary can improve the accuracy of the model. But their method does not adopt batch mode, and needs to annotate a sample and its projection point on decision boundary in each iteration, which means that the annotation work is almost twice as much as conventional methods.

To combine the advantages of both batch sampling and decision boundary annotation, we proposed a new batch active learning method which adopts a two-stage sampling strategy, and it is called BALTS. In BALTS, samples are mapped into an embedding, and clustered according to their corresponding latent variables in the embedding. The first stage sampling of BALTS is initial seed sampling which selects a seed using reject sampling and the cluster structure of instances to ensure the initial seeds can approximate the distribution of data and with high diversity. By exploiting the seed, we construct a line that is perpendicular to the decision boundary and sample a sequence of latent variables uniformly on the line. And with the help of a generative adversarial model which converts the latent variables to a row of instances we can let the oracle make boundary annotations. The boundary annotations can help us estimate the uncertainty of unlabeled instance as well as simplify the update of decision boundary by using regression. The second stage sampling is batch sampling which selects a batch of query samples according to multiple criterion including uncertainty, representativeness and diversity. In this paper, by treating the instance sampling as the selection of representative in a local region and maximizing the rewards that can get from selecting a instance as the new representative, we proposed a novel mechanism to maintain local representativeness and diversity of query samples. And the time complex of our batch sampling is $O(N_u)$, where N_u is the number of unlabeled samples.

Compared with current active learning methods our proposed BALTS has the following features:

1) With the direct annotation of decision boundary, BALTS can avoid the complicated model re-training in each iteration, which will reduce much of the computation burden and let the oracle do not need to wait for samples;
2) Based on clustering and reject sampling, BALTS can fast select representative initial seeds with reasonable diversity to generate valuable boundary annotations, which is helpful for decision boundary estimation and property measuring of query instances;
3) By measuring the representativeness of instance according to its neighbors instead of all unlabeled instances and maintaining the local representativeness and diversity by maximizing the gain of adding a new query sample, BALTS has a batch sampling mechanism with time complex of $O(N_u)$.

II. RELATED WORK

Due to its effectiveness, active learning has become a hot research topic in the field of machine learning and computer vision. In this section, we will make a brief review of the researches that are most related to our work.

A. CRITERION FOR SELECTING SAMPLES

Informativeness is the first and widely used criteria for selecting samples. In uncertainty-based active learning, it is believed that samples with high uncertainty is more informative. The uncertainty of sample x can be estimated by the posterior distribution $P(y|x)$, where y is the possible labels of sample x. Since the calculation of difference between posterior distribution is not convenient, entropy is introduced as a measure of uncertainty [13], [14]. And in SVM-based active learning, the uncertainty of sample x can be measured by margin which is the distance between x and the decision boundary [15]. In query-by-committee (QBC) based algorithms [16], the most informative query is considered to be the instance about which the committees most disagree. And vote entropy [17] and average Kullback-Leibler (KL) divergence [10] are the two main approaches for measuring the level of disagreement. However, these approaches are prone to sampling bias [18] and over-fitting [14], because the selection of instances is solely determined by the model trained on a small number of labeled examples.

Another widely used criteria is representativeness or impact, according to which the instance that can represent more unlabeled samples should be selected. More accurately, representativeness means the extent of help for classifying other unlabeled instances when one instance is labeled. Since representativeness can be measured by the similarity between samples, while similarity is the foundation of clustering, some approaches tried to exploit the clustering structure of unlabeled samples and select the representative samples [19], [20]. Xu *et al.* [8] considered that the clustering centers are representative and selected them for labeling, while Nguyen and Smeulders [19] gave priority to samples in dense clusters. Bodo *et al.* [20] proposed an algorithm that exploited graph clustering with normalized cuts to extract representative points. In [21], the instances of each cluster are sorted according to the local density and relative distance, and the top $\sqrt{N}$ instances are considered to be representative ones. However, the quality of clustering results [22] will strongly effect the performance of the selected instances in these approaches.

Other measures for instance selection include diversity [23] and generalization error [24]. Diversity criteria means that the selected instances should be diversely distributed to improve the generalization of the model [14]. Generalization error criteria tries to select samples that can minimize the generalization error. In [25], Zhang and Oles adopted Fisher information function to measure the influence of the sample on the model error. Roy and McCallum [26] proposed to select the instance that can minimize the expected error between the current probability distribution and the actual probability distribution.

As mentioned above, each kind of criteria has its own advantage and weakness. Using a single criteria is not enough in many cases, so the combination of multiple criterion has been adopted in active learning algorithms. In [18], informativeness and representativeness were combined and the measure of representativeness took into account the cluster structure of unlabeled instances as well as the class assignments. In [14], by representing the data with a graph, Yang *et al.* evaluated the uncertainty via random walks on the graph and exploited diversity maximization as a constraint for uncertainty sampling. Both [27] and [28] took into account uncertainty, representativeness and diversity in selecting instances. In [27], the representativeness and diversity were maintained using a objective function that maximizes the similarity bewteen the samples in $\mathcal{X}_L \cup \mathcal{X}_S$ and unlabeled samples after a new unlabeled sample is selected, where $\mathcal{X}_L$ is the labeled sample set and $\mathcal{X}_S$ is the current selected sample set, so the time complexity of their method is $O(N_u^2)$ while that of our method is $O(N_u)$. In [28], Wang *et al.* used the probability density function to estimate representativeness, and a threshold of distance was adopted in selecting samples to ensure the diversity. Although their method is effective, the hard setting of the threshold may constraint the generalization of their method.

B. BATCH MODE ACTIVE LEARNING

To avoid stopping annotation work of the oracle frequently when active learning algorithm provides instances one by one, approaches that select a batch of instances for labeling in each iteration have been proposed in recent years.

The early work of batch mode active learning is based on SVM and the most direct way for batch sampling is to increase the number of selected samples in each iteration according the distance between the sample and the decision boundary. However, the theoretical motivation for each selected example to approximately bisect the version space into two halves of equal volume does not hold in this case [29], so some new sampling strategies are proposed for batch mode active learning. In [30], mutual information between the labeled instances and the unlabeled instances were adopted to measure informativeness of a batch of samples, and the maximization of mutual information were converted to a matrix partition problem by employing a Gaussian process framework. Chattopadhyay *et al.* [31] proposed Marginal Probability based Active Learning (MP-AL), which tried to select batches of query instances that can represent best the distribution of the unlabeled instances. But these two methods did not consider the combination of multiple instance selection criterion. In batch mode active learning, some researches also have considered of improving the diversity of selected instances. Brinker [29] proposed adding diversity constraint in sampling and using the angles between the hyperplanes induced by the selected instance to measure diversity, so as to ensure the instances chosen in each of the batches are highly diverse. Hoi *et al.* proposed a batch sampling strategy based on Fisher information matrix [4], [32], which can maximize uncertainty and implicitly minimize redundancy. But their method did not consider the representativeness of instances.

In addition, the current batch mode active learning algorithms need to retrain the model in each iteration, to ensure the selected samples can improve the performance of them mostly. However, model retraining is the most time consuming step in active learning, so we proposed to release the dependency of instance sampling on the precision of model. This will enable us to reduce the time for fine retraining of model in each iteration.

C. ACTIVE LEARNING WITH GENERATIVE ADVERSARIAL NETWORKS

With the development of generative adversarial networks [33], some researchers have introduced generative adversarial models into active learning. According to the idea in margin-based sampling, Ducoffe and Precioso [34] proposed a Deep-Fool based active learning strategy which selects unlabeled samples with the smallest adversarial perturbation, as they thought that adversarial perturbation can approximate the margin. Zhu and Bento [35] combined generative model and the uncertainty sampling to synthesize uncertain samples for model training. Mayer and Timofte [36] adopted GANs to generate high entropy samples and labeled the similar samples searched from the pool rather than directly annotating the synthetic samples. Hence, the quality of new samples is higher and annotations are more reliable compared to [35]. But compared with the conventional method, there is not remarkable improvement of performance in their method. Instead of using the generated samples in model training, Huijser and van Gemert [12] proposed an active learning method which exploits the generated samples to help annotating the decision boundary. After selecting a sample using some sampling strategy, they constructed a line which was perpendicular to the decision boundary and let the oracle indicate the intersection of the line and decision boundary and label the sample as well. The performance of their method is much better than uncertainty-based method. But their method increased the annotation work, which is almost twice as much as conventional methods.

III. BATCH SAMPLING WITH BOUNDARY ANNOTATION

In many practical scenarios, we have a large amount of unlabeled samples and a few labeled samples. Let's denote by $\mathcal{X}_L = \{(x_1, y_1), (x_2, y_2) \ldots (x_{N_l}, y_{N_l})\}$ the set of initial labeled data, by $\mathcal{X}_U = \{x_{N_l+1}, x_{N_l+2} \ldots x_N\}$ the set of unlabeled data, by $\mathcal{X} = \mathcal{X}_L \cup \mathcal{X}_U$ the whole training set and by $x \in \mathcal{X}$ the

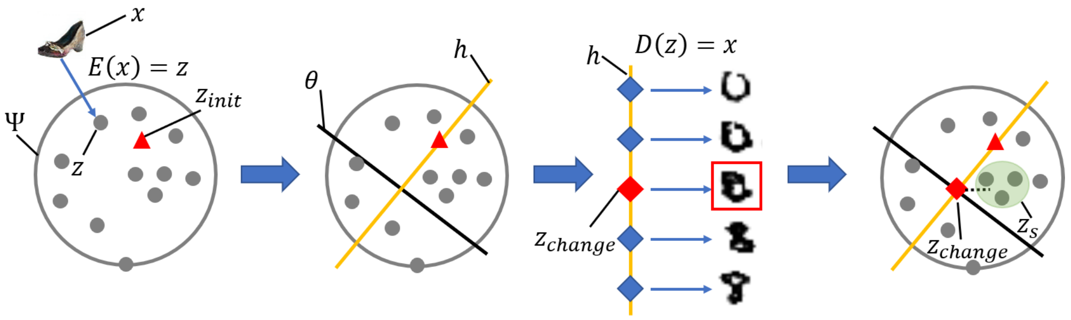


FIGURE 1. Overview of the BALTS method. Ψ is a hypersphere surrounding the set $\mathcal{Z}$ in the embedding.

sample, where N_l is the number of labeled samples, $N = |\mathcal{X}|$ is size of training data, and $N_u = N - N_l$ is the number of unlabeled samples. At every step of active learning, we need to select m examples from $\mathcal{X}_U$ for labeling.

In our method, by making decision boundary annotation similar to [12], we can estimate the decision boundary θ using regression without complex training process. The overview of our method is shown in Fig.1. For each sample $x_i \in \mathcal{X}$, we map it to a latent variable z_i in a embedding by using an inference network $\mathrm{E}(x) = z$. The labeled instance set $\mathcal{X}_L$ is mapped to $\mathcal{Z}_L = \{(z_1, y_1), (z_2, y_2) \ldots (z_{N_l}, y_{N_l})\}$ and the unlabeled set is mapped to $\mathcal{Z}_U = \{z_{N_l+1}, z_{N_l+2} \cdots z_N\}$. We denote all the latent variables by $\mathcal{Z} = \mathcal{Z}_L \cup \mathcal{Z}_U$. And then, the instances in set $\mathcal{Z}$ are clustered in to K clusters $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$ using sparse affinity propagation algorithm [39] based a graph $\mathcal{G}$ constructed on the latent variables in $\mathcal{Z}$.

In each iteration, we sample an initial seed z_{init} using reject sampling from a cluster k, which is determined by a probability, to make the initial seeds approximate the distribution of instances better and improve their diversity. We then construct a line h that is perpendicular to the decision boundary through z_{init} and uniformly sampling a row of latent variable $\mathcal{Z}_g$ along the line h. Since line h is perpendicular to θ, there will be a class change point in the row of images corresponding to latent variables in $\mathcal{Z}_g$. We let the oracle annotate the class change point. And then we use the proposed batch sampling method to select a set of m unlabeled latent variables $\mathcal{Z}_S = \{z_{s_1}, z_{s_2}, \ldots, z_{s_m}\}$ from $\mathcal{Z}_U$ with multiple criterion and ask the oracle to label each corresponding instance x_{s_i} of z_{s_i}. Finally, we optimize both classification and regression loss to get the final well trained model, in which the classification loss is to optimize the model with all labeled samples in set $\mathcal{X}_L$, and the regression loss is to fit the boundary annotations in set $\mathcal{A}$. The detailed description of BALTS is shown in Algorithm 1.

A. BOUNDARY ANNOTATION

A heuristic idea for the oracle to annotate the decision boundary is to show the oracle a sequence of instances which have a class change point. However, the decision boundary is in the low dimension embedding where our model is trained. Since the latent variables in the embedding are abstract, the human oracle cannot discriminate these latent variables. Therefore, we need a data transformer between the embedding and the data space. The Generative Adversarial Networks (GANs) developed in recent years [37], [38] can satisfy this requirement. The inference network in GANs can map the instance x from data space to a latent variable, which can be regarded as an encoder $\mathrm{E}(x) = z$, while the generative network in GANs can map a latent variable z to a instance, which can be regarded as a decoder $\mathrm{D}(z) = x$.

In order to ensure that the row of generated instances has a class change point, a feasible method is to convert a sequence of latent variables that are on a line h to instances, and the line h intersects the decision boundary θ. Farther more, if the line h is perpendicular to θ, the instance sequence can more clearly show the class change, and the oracle can label the class change point more accurately.

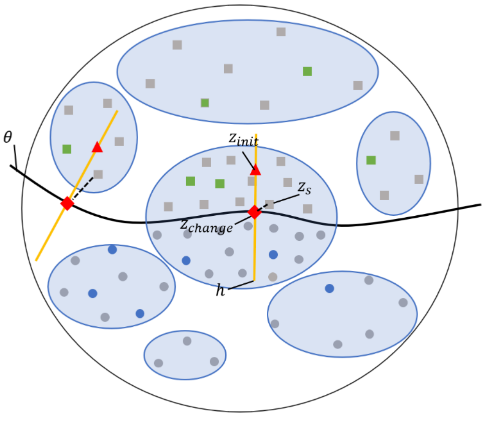


FIGURE 2. Schematic of instance selection with boundary annotation.

In our method, we select an initial seed to construct the line h. We then uniformly sample a row of latent variable $\mathcal{Z}_g$ along the line h, and convert the latent variable sequence $\mathcal{Z}_g$ into an image sequence $\mathcal{X}_g$ using the generative network $\mathrm{D}(z) = x$. Since there will be a class change point in the row of image $\mathcal{X}_g$, we let the oracle annotate the class change point and add it's corresponding latent variable z_{change} to the boundary annotation set $\mathcal{A}$. Fig.2 shows the relationship between initial

Algorithm 1 BALTS

```
Input:
    $\mathcal{X}_L$, the set of labeled training examples
    $\mathcal{X}_U$, the set of unlabeled training examples
    $m$, the number of examples to be labeled each iteration
    $M$, the total number of examples to be labeled
    # initialization
1:  $\mathcal{A} \leftarrow \phi$, $\mathcal{Z} \leftarrow \phi$
    # Train the GANs model
2:  (E($x$), D($z$)) $\leftarrow$ TrainGANs($\mathcal{X}$)
    # mapping images to latent variables using inference network
3:  for $x_i \in \mathcal{X}$ do
4:      $z_i \leftarrow$ E($x_i$)
5:      $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{z_i\}$
6:      $max_i^{LS} \leftarrow 0$
7:  end for
8:  $(\mathcal{Z}_L, \mathcal{Z}_U) \leftarrow \mathcal{Z}$
9:  $\mathcal{G} \leftarrow$ GraphBuilding($\mathcal{Z}$)
10: $\mathcal{C} \leftarrow$ Clustering($\mathcal{Z}, \mathcal{G}$)
11: $\theta \leftarrow$ TrainModel($\mathcal{X}_L$)
12: while $M > 0$ do
        # Step 1: sampling an initial seed with cluster-based strategy
13:     $z_{init} \leftarrow$ InitSampling($\mathcal{C}$)
        # Step 2: ask the oracle to label the boundary annotation
14:     $z_{change} \leftarrow$ BoundaryAnnatation($z_{init}, \theta$)
15:     $\mathcal{A} \leftarrow \mathcal{A} \cup \{z_{change}\}$
16:     $\theta \leftarrow$ update($\mathcal{A}, \theta$)
17:     $\{z_{s_1}, z_{s_2}, .., z_{s_m}\} \leftarrow$ BatchSampling($\mathcal{Z}, \mathcal{A}, \mathcal{G}, max_i^{LS}, m$)
18:     for $z_{s_i} \in \{z_{s_1}, z_{s_2}, .., z_{s_m}\}$
19:         $y_{s_i} \leftarrow$ Labeling($x_{s_i}$)
20:         $\mathcal{Z}_L \leftarrow \mathcal{Z}_L \cup \{z_{s_i}, y_{s_i}\}$
21:         $\mathcal{Z}_U \leftarrow \mathcal{Z}_U \setminus \{z_{s_i}\}$
22:     endfor
23:     $M = M - m$
24: endwhile
25: model $\leftarrow$ TrainFinalModel($\mathcal{A}, \mathcal{Z}_L$)
26: return model
```

seed z_{init}, boundary annotation point z_{change} and the selected sample z_s.

1) INITIAL SEED SAMPLING BASED ON CLUSTERING

The initial seed actually determines the location of boundary annotations. And the location of boundary annotation points in the set $\mathcal{A}$ have strong impact on the performance of method which has been proved in our experiments. When the points in $\mathcal{A}$ can approximate the distribution of instances better and have better representativeness and diversity, our method will have a better performance, so we introduce a clustering-based strategy to select the initial seeds.

For clustering of instances, we adopt sparse affinity propagation [39], which is a kind of graph-based clustering and faster than the original Affinity Propagation (AP) algorithm [40]. In this paper, we first build a graph $\mathcal{G}$ whose vertexes are latent variables in $\mathcal{Z}$. For a latent variable in $\mathcal{Z}$, each of its J nearest neighbors has an edge to it, and for an unlabeled instance in $\mathcal{Z}_U$, it also has an edge to each latent variable in $\mathcal{Z}_L$. The weight of an edge between two vertexes z_i and z_j is defined by

$$w_{ij} = \begin{cases} \exp\left(-\frac{1}{2\sigma^2}||z_i - z_j||^2\right) & \text{if } j \in \text{NB}_i \cup \mathcal{Z}_L \\ 0 & \text{else} \end{cases} \quad (1)$$

where σ is the variance of all instance, and w_{ij} also represents the similarity between z_i and z_j. In AP clustering, since the initial value of w_{ii} represents the preference that z_i is a cluster center, we let w_{ii} be the entropy H_i of z_i, which means that we prefer selecting instances that are nearer to the decision boundary to be the center of clusters. And the calculation of H_i will be given in III-B.

With the K clusters from AP algorithm, we hope to sample more representative seeds in the clusters that are denser and more uncertain. The uncertainty of a clusters can be measured by the distance between its center and the decision boundary or the labeled samples in it. Here, we consider the two measures simultaneously. And calculate the entropy of a cluster k by

$$H_k^c = \frac{n_k^l}{n_k} H_k^l + \frac{n_k^u}{n_k} H_k^\theta \quad (2)$$

where H_k^c is the entropy of cluster k, n_k^l and n_k^u are the sample number of labeled and unlabeled in cluster k respectively, H_k^l and H_k^θ are the entropy of cluster k calculated according to labeled samples and distance from the center of cluster k to decision boundary θ respectively, and

$$H_k^l = -\frac{n_k^+}{n_k^l} \log(\frac{n_k^+}{n_k^l}) - \frac{n_k^-}{n_k^l} \log(\frac{n_k^-}{n_k^l}) \quad (3)$$

$$H_k^\theta = -p_k^\theta \log p_k^\theta - (1 - p_k^\theta) \log(1 - p_k^\theta) \quad (4)$$

where n_k^+ and n_k^- is the number of positive and negative instances in cluster k, and p_k^θ is the uncertainty of labels in cluster k according to margin. And p_k^θ is defined by

$$p_k^\theta = \frac{1}{1 + \exp(-\frac{d_k^{\theta 2}}{2\sigma^2})}$$

where d_k^θ is the distance from the center of cluster k to the decision boundary θ. The representativeness of cluster k is measured by the distance d_k^a from the center of cluster k to the center of all instances. To combine the measures, we assign a weight $\tilde{p}_k$ to cluster k, and

$$\tilde{p}_k = H_k^c + \exp(-\frac{d_k^{a2}}{2\sigma^2})$$

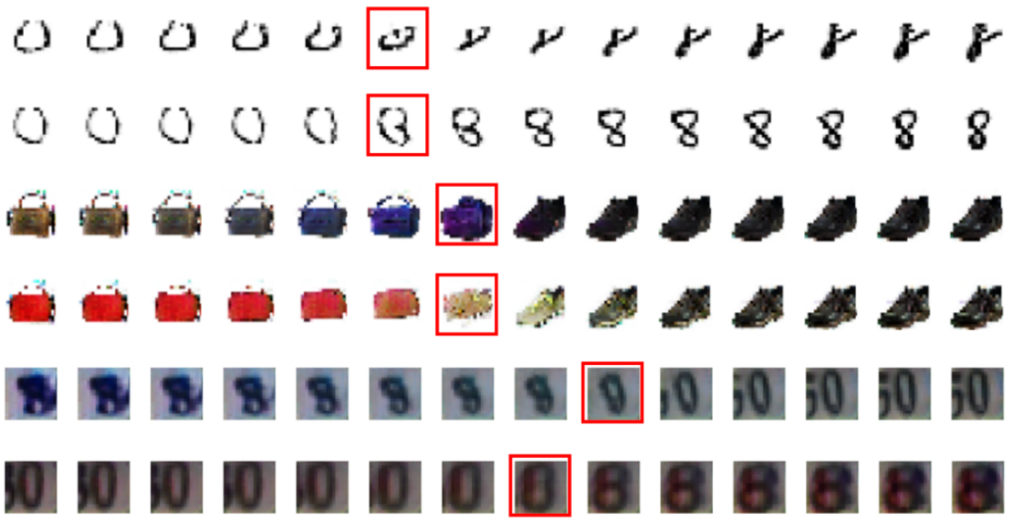


FIGURE 3. Image sequence of different data sets synthesized by the generative adversarial model. The red frame is the category change point in the image sequence.

And the probabilities p_k to sample a seed from cluster k is defined by (5).

$$p_k = \frac{\tilde{p_k} n_k^u}{\sum_{i=1}^{K} n_i^u \tilde{p_i}} \tag{5}$$

After select the cluster k with probability p_k, we use reject sampling to select initial seeds. We first randomly sample points in cluster k, and for a sampled point z_{test} we accept it as an initial seed with the the following probability

$$p_{accept}(z_{test}) = 1 - \exp\left(-\frac{||z_{test} - \mu_{init}^k||^2}{2\sigma_k^2}\right)$$

where σ_k is the variance of instances in cluster k, μ_{init}^k is the mean of initial seeds that have been selected in cluster k. When there is no seed selected in cluster k, μ_{init}^k is set be an instance that is farthest from the center of cluster k. Using reject sampling, which means the larger the distance between z_{test} and μ_{init}^k, the more likely to accept it as an initial seed, is a simple and fast way to improve diversity of initial seeds.

2) DECISION BOUNDARY ANNOTATION

The detailed description of the boundary annotation is shown in algorithm 2. Given the initial seed z_{init}, its projection z_{pr} on the decision boundary θ can be obtained by the equation (6)

$$z_{pr} = p_1 + \frac{(z_{init} - p_1)^T (p_2 - p1)\dot{(}p_2 - p_1)}{||p_2 - p_1||^2} \tag{6}$$

where p_1 and p_2 are any two points on the boundary θ. And function of the line h that pass through z_{init} and z_{pr} can be obtained by equation (7).

$$z = z_{init} + t(z_{pr} - z_{init}) \tag{7}$$

We then sample uniformly on line h to get a sequence vectors $\mathcal{Z}_g = \{z_g^1, z_g^2, \ldots, z_g^n\}$. The sampling range on line

Algorithm 2 Boundary Annotation

```
Input:
    $z_{init}$, the initial point
    $\theta$, the decision boundary of the model
1: $h \leftarrow$ construct vertical line of $\theta$ through $z_{init}$
2: $\mathcal{Z}_g = \{z_g^1, z_g^2, \ldots, z_g^n\} \leftarrow$ uniform sample on $h$
   # mapping $\mathcal{Z}_g$ images using generative network
3: for $z_g^i \in \mathcal{Z}_g$ do
4:     $x_g^i \leftarrow$ D($z_g^i$)
5:     $\mathcal{X}_g \leftarrow \mathcal{X}_g \cup \{x_g^i\}$
6: end for
7: $z_{change}$ $\leftarrow$ask the oracle to annotate category change
     point in $\mathcal{X}_g$
8: return $z_{change}$
```

h is limited by the hypersphere Ψ surrounding all latent variables to ensure that the sampled latent variable sequence is within a reasonable range. We take the center of all latent variables as the center $\bar{z}$ of the hypersphere Ψ, and the largest distance from $\bar{z}$ to latent variables as the radius r. Therefore, the spherical center $\bar{z}$ and radius r of the hypersphere Ψ are defined by

$$\bar{z} = \frac{1}{N}\sum_{i=1}^{N} z_i$$
$$r = \max_{0<i\leq N} ||z_i - \bar{z}||^2 \tag{8}$$

After the hypersphere is defined, we calculate the two intersection points of the line h and the hypersphere Ψ, and sample uniformly on the line segment between the two points to obtain the latent vector sequence $\mathcal{Z}_g$. Finally, the generative network $D(x) = z$ is used to convert elements $z_g^i \in \mathcal{Z}_g$ into the image sequence $\mathcal{X}_g = \{x_g^1, x_g^2, \ldots, x_g^n\}$. And the oracle is required to label the class change points z_{change}. Examples of some generated image sequence for different datasets are shown in Fig. 3.

B. BATCH SAMPLING WITH MULTIPLE CRITERION

In our method, samples in set $\mathcal{X}_S$ are selected by considering multiple criterion including uncertainty, representativeness and diversity.

Generally, in batch sampling we should select a set of instances $\mathcal{Z}_S$ that can maximize the following objective function.

$$Q^*(\mathcal{Z}_S) = \max_{\mathcal{Z}_S \subset \mathcal{Z}_U} (\alpha H^*(\mathcal{Z}_S) + (1-\alpha)V^*(\mathcal{Z}_S)) \tag{9}$$

where $H^*(\mathcal{Z}_S)$ is the informativeness of $\mathcal{Z}_S$, and $V^*(\mathcal{Z}_S)$ is the diversity and representativeness constraint on $\mathcal{Z}_S$. However, to get the global optimum $\mathcal{Z}_S$ in equation (9) is a NP-hard problem, so we adopt a greedy method instead. And we change the objective function to equation (10), so that we can greedily select the m best instances in each iteration.

$$Q(z_i) = \alpha H_i + (1-\alpha)V(z_i) \tag{10}$$

where H_i measures the uncertainty of z_i and $V(z_i)$ is a function of the diversity and local representativeness constraints. We define $V(z_i)$ by equation (11).

$$V(z_i) = \frac{1}{|\mathrm{NB}_i|} \sum_{z_j \in \mathrm{NB}_i} H_j[\beta w_{ij} - \max_{z_k \in \mathcal{Z}_L \cup \mathcal{Z}_S} (w_{jk})] \tag{11}$$

where β is a parameter to adjust the influence of local representativeness on the final objective function, $\mathrm{NB}_i \subseteq (\mathcal{Z}_U - \mathcal{Z}_S)$ is a set of J nearest neighbors of z_i. And we use the graph $\mathcal{G}$ built in initial seed sampling to help calculate $V(z_i)$. The description of our greedy batch sampling method is shown in algorithm 3.

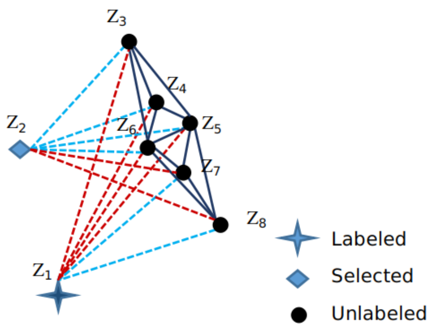


FIGURE 4. An example graph for selecting instance.

Function $V(z_i)$ means the reward that we will get if neighbors of z_i select z_i as their representative instead of former representative which is nearest to them in $\mathcal{Z}_L \cup \mathcal{Z}_S$. This mechanism can not only measure the representativeness of z_i according to its similarity with its neighbors, but also can maintain the diversity of $\mathcal{Z}_S$. Because when neighbors of z_i is farther from the instances that have been labeled, $\max_{z_k \in \mathcal{Z}_L \cup \mathcal{Z}_S}(w_{jk})$ will be smaller, and we can get larger reward if other conditions are the same. Fig.4 shows an

Algorithm 3 Batch Sampling in Greedy

```
Input:
    $\mathcal{Z}$, latent vector set of instance
    $\mathcal{G}$, a graph, the weight of edge between node $i$ and $j$ is $w_{ij}$
    $max^{LS}$, a vector whose element $max_i^{LS} = \max_{z_k}(w_{ik})$
        s.t. $z_k \in \mathcal{Z}_L \cup \mathcal{Z}_S$
    $\mathcal{A}$, the set of annotated category change points
    $m$, the number of examples to be labeled each iteration
1: Initialize $\mathcal{Z}_S \leftarrow \phi$
2: Calculate entropy $H_i$ for each $z_i \in \mathcal{Z}_U$
3: while $|\mathcal{Z}_S| < m$ do
4:     $Q_{max} \leftarrow 0$
       # Calculate $Q(z_i)$
5:     for $z_i \in \mathcal{Z}_U - \mathcal{Z}_S$ do
6:         $V(z_i) \leftarrow 0$
7:         for $z_j \in \mathrm{NB}_i$ do
8:             $V(z_i) \leftarrow V(z_i) + H_j[\beta w_{ij} - max_j^{LS}]$
9:         end for
10:        if $Q_{max} < \alpha H_i + (1-\alpha)\frac{V(z_i)}{|\mathrm{NB}_i|}$ do
11:            $Q_{max} \leftarrow \alpha H_i + (1-\alpha)\frac{V(z_i)}{|\mathrm{NB}_i|}$
12:            $v \leftarrow i$
13:        end if
14:    end for
15:    $\mathcal{Z}_S \leftarrow \mathcal{Z}_S \cup \{z_v\}$
16:    for $z_i \in \mathcal{Z}_U - \mathcal{Z}_S$ do
17:        Update neighbor set $\mathrm{NB}_i$ of $z_i$
18:        $w_{iv} \leftarrow ||z_i - z_v||^2$
19:        $max_i^{LS} \leftarrow \max(max_i^{LS}, w_{iv})$
20:    end for
21: end while
22: return $\mathcal{Z}_S$
```

example of the graph. In Fig.4, there are 8 latent variables $z1, \ldots z_8$, $w_{64} = w_{54}$, $w_{67} = w_{57}$ and $J = 3$. If we let $H_i = 1$ be the same for all instances and $\beta = 1$, when we select z_6 as the next instance, we have

$$V(z_6) = \frac{1}{3}[(w_{65} - w_{52}) + (w_{64} - w_{42}) + (w_{67} - w_{71})]$$

when we select z_5 as the next instance, we have

$$V(z_5) = \frac{1}{3}[(w_{65} - w_{62}) + (w_{54} - w_{42}) + (w_{57} - w_{71})]$$

It is obvious that $V(z_6) < V(z_5)$, and z_5 will be added to $\mathcal{Z}_S$ according to the value of $V(z_i)$. From the example we can see that $V(z_i)$ can maintain both representativeness and diversity constraints.

In order to reduce the running time of each iteration, so that the oracle need not wait for instances while labeling, we only update the parameters of the decision boundary to fit boundary annotations in $\mathcal{A}$ using regression in our method. Therefore, the decision boundary is less precise than the conventional method. Without retraining the model in each iteration, we prefer to estimate the uncertainty of instance according to the labeled instances and boundary annotations directly.

For each latent point $z_i \in \mathcal{Z}_U$ we estimate it's margin by equation (12).

$$\begin{aligned} d_i^l &= \min(\frac{1}{2}\max(d_i^+, d_i^-), d_i^b) \\ d_i^+ &= \min_{z_j \in \mathcal{Z}_L \wedge y_j = 1}(||z_i - z_j||) \\ d_i^- &= \min_{z_j \in \mathcal{Z}_L \wedge y_j = -1}(||z_i - z_j||) \\ d_i^b &= \min_{z_j \in \mathcal{A}}(||z_i - z_j||) \end{aligned} \tag{12}$$

With the estimated margin of $z_i \in \mathcal{Z}_U$, we define it's entropy by

$$H_i = -p_i \log(p_i) - (1 - p_i)\log(1 - p_i) \tag{13}$$

where p_i represents the uncertainty of z_i's label, and

$$p_i = \frac{1}{1 + \exp(-\frac{{d_i^l}^2}{2\sigma^2})} \tag{14}$$

By measuring the local representativeness which only considers the relationship between neighbors of z_i and using the utility of changing delegates to constrain diversity, the time complexity of our batch sampling algorithm is $T((|\mathrm{NB}_i|+1)\cdot|\mathcal{Z}_U|)$. Since $|\mathrm{NB}_i|$ is much smaller than $|\mathcal{Z}_U|$, the time complexity of our batch sampling algorithm is $O(|\mathcal{Z}_U|)$, which is better than the methods measuring global representativeness whose time complexity is $O(|\mathcal{Z}_U|^2)$ [27].

C. UPDATING AND FINAL TRAINING OF THE MODEL

In each iteration, we only use the boundary annotations in the set $\mathcal{A}$ to fit the linear regression model, and obtain an approximate decision boundary. So that we can let line h cross the decision boundary θ to make the image sequence have class change point that can be labeled. And we train a final precise model by optimizing the classification loss and the regression loss after the all instances are labeled.

For updating the regression model $f(x) = \omega^T + b$, we use a simple square loss $\mathcal{L}_{regress}$ to fit it.

$$\mathcal{L}_{regress} = \frac{1}{|A|}\sum_{z \in \mathcal{A}}\left(\omega^T z + b\right)^2 \tag{15}$$

For the training of the final model, we simultaneously optimize the classification loss on the labeled instances $\mathcal{Z}_L$ and use the regression loss to fit decision boundary annotations in $\mathcal{A}$. In order to make a fair comparison with other methods, we adopt a linear model for testing, and define the decision boundary θ by $\omega^T z + b = 0$.

For the classification loss $\mathcal{L}_{class}$, we use a standard SVM hinge loss over the labeled samples (z, y) in $\mathcal{Z}_L$.

$$\mathcal{L}_{class} = \frac{1}{|\mathcal{Z}_L|}\sum_{(z,y)\in\mathcal{Z}_L}\max\left(0, 1 - y(\omega^T z + b)\right) \tag{16}$$

And we use a simple square loss defined in (15) to fit the model $\omega + b$ to the annotations in $\mathcal{A}$. The final loss $\mathcal{L}$ is defined by

$$\mathcal{L} = (1-\gamma)\mathcal{L}_{class} + \gamma\mathcal{L}_{regress} + \lambda||\omega||^2 \tag{17}$$

where γ is a parameter to adjust the weight of $\mathcal{L}_{class}$ and $\mathcal{L}_{regress}$, λ controls the influence of the regularization term $||w||^2$.

IV. EXPERIMENTS

A. DATASETS AND MODELS

We evaluated our method on three public datasets: MNIST [41], SVHN [42] and Shoe-Bag [12] dataset. MNIST contains 60,000 binary digit images with a resolution of 28 × 28. In MNIST, the train set includes 50,000 images and test set includes 10,000 images. Digit images in SVHN dataset are collected from the house numbers of Google Streetview. It has 73257 train and 26032 test images with a resolution of 32 × 32. Shoe-Bag dataset has 40,000 train images and 14,000 test images with a resolution of 64 × 64. And we adopt the ALI (Adversarially learned inference) [37] as the generative adversarial model in our proposed method.

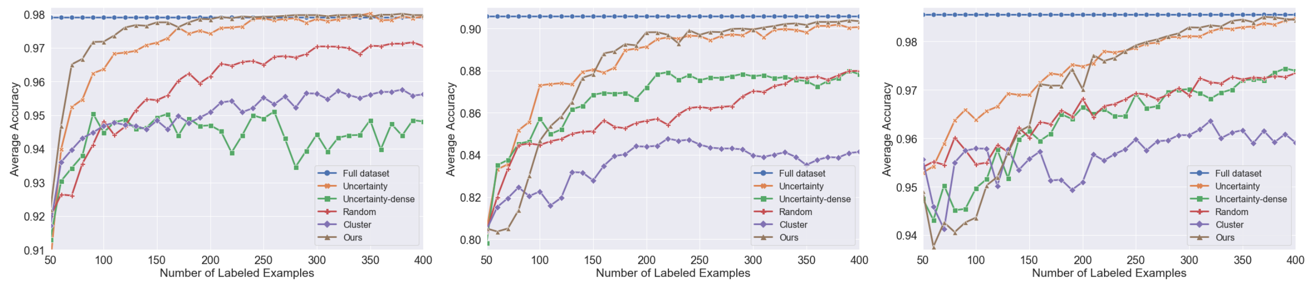
B. EVALUATED ALGORITHMS

In the experiment, we test and compared our method with the following methods:

1) Full trained: A method using SVM as classifier that are trained with all the instances in train set.
2) Random sampling method: A method that selects samples randomly from an unlabeled sample set.
3) Uncertainty-based method [43]: Samples are selected based on uncertainty which is the most typical margin-based method. According to the benchmarked tests in [44], although the uncertainty-based method is very simple and proposed in 1994, it even has a better performance than many well-designed new methods.
4) Uncertainty-dense [45]: This method has considered informativeness and representativeness in sample selection, and it prefers selecting sample in the region with higher density.
5) Clustering based method [46]: This method takes into account the representativeness of the samples by clustering all the unlabeled samples and selects the cluster center for labeling.

C. EXPERIMENTAL SETUP

For each dataset, we used the method in [37] to train the ALI model. We trained an embedding with 100 dimensions for SVHN dataset and MNIST dataset and trained an embedding with 256 dimensions for more complicated Shoe-Bag dataset. We set dropout = 0.4 for the layers of the discriminator. We ran each algorithm for 5 times in each dataset and took the average of the 5 experiments as the final result. Each experiment starts with an initial set of 50 random labeled samples. In each iteration, we selected 10 samples from the unlabeled sample set for labeling, and then added them to



(a) MNIST 0 vs. 8 (b) SVHN 0 vs. 8 (c) Shoe-Bag

FIGURE 5. Precision with different number of labeled instances. We compared our method with the methods tagged as Uncertainty [43], Uncertainty-dense [45], Cluster [46] and Random which selects samples randomly.

TABLE 1. Evaluation of algorithms using AULC in three data sets. The best results are shown in bold.

Methods	Datasets	MNIST 0 vs. 8			SVHN 0 vs. 8			Shoe-Bag		
	Batch size	1	5	10	1	5	10	1	5	10
Ours		**342.3±0.32**	**68.3± 0.06**	**34.2±0.02**	**310.9±0.67**	**61.7±0.15**	**30.8±0.1**	339.6±1.4	67.9±0.22	34.0±0.14
ADBA		340.5±0.40	68.2±0.10	34.1±0.02	305.3±0.80	60.8±0.39	30.5±0.15	**342.1±1.04**	**68.4±0.18**	**34.2±0.05**
Uncertainty		340.4±0.52	68.0±0.11	34.0±0.012	309.5±0.51	61.6±0.29	**30.8±0.09**	341.1±0.31	68.2±0.08	34.1±0.03
Uncertainty-dense		331.2±1.13	65.9±0.11	33.0±0.10	305.5±2.18	60.8±0.40	30.7±0.20	337.3±0.58	67.4±0.10	33.7±0.08
Random		334.6±2.10	67.1±0.12	33.0±0.10	301.9±1.76	60.4±0.22	30.4±0.20	336.7±0.63	67.4±0.30	33.7±0.08

labeled sample set $\mathcal{Z}_L$ until the size of the labeled sample set is 400. After each iteration, we used the current labeled samples to train the model, then evaluated the accuracy of the model on the test set. For the loss function, we set the equilibrium factor to 0.2 and the regularization coefficient to 1.

D. EXPERIMENTAL RESULTS

1) PREDICTION PRECISION AND CONVERGENCE

We compared the precision of the algorithm mentioned above with different number of labeled instances. The experimental results are shown in Fig.5 in which the algorithms are tagged in short as Uncertainty [43], Uncertainty-dense [45] and Cluster [46] respectively.

From the experimental results in these three datasets, we can see that only about 400 labeled samples are needed for the efficient active learning method to obtain a model with precise prediction on test data. Our method and uncertainty-based method are obviously better than random-based method, which shows that the sampling mechanism in our method is effective, it performs well without retraining the model in each iteration, that means it does not rely on the precision of the model in selecting valuable samples. Clustering based method (Cluster) [46] only considers the representativeness of the samples, so the samples selected by this method are more and more redundant, which has little effect on the performance improvement of the model. The same problem exists in uncertainty-dense method, which always selects samples in dense areas, does not consider the diversity of samples, and cannot guarantee the global precision of decision boundary. Our method is comparable with uncertainty-based method in performance, especially after a certain number of iterations, our method can make the model converge faster.

2) COMPARATIVE STUDY USING AULC

We compared our method with the methods mentioned above and ADBA [12] which combine the decision boundary annotation and uncertainty sampling using the Area Under the (accuracy) Learning Curve (AULC) measure [45]. The Area under the Learning Curve is computed by integrating over the test accuracy scores of N active learning iterations using the trapezoidal rule:

$$\text{AULC} = \sum_{i=1}^{N} \frac{1}{2}(\text{acc}_{i-1} + acc_i) \tag{18}$$

where acc_0 is the test accuracy of the initial classifier before the first query. The AULC score measures the informativeness per annotation and is high for active learning methods that quickly learn high-performing models with few queries, i.e. in few iterations.

In the experiments, the size of initial labeled samples is 50, and the algorithm stops learning when the size of the labeled sample set reaches 400. We tested the methods with the batch sizes of 1, 5 and 10, and the number of iterations is 350, 70 and 35 respectively. The highest value of AULC is also 350, 70 and 35, which will reach when the test accuracy is 100%. The AULC values with different batch size in three datasets are shown in Table 1.

Table 1 shows that our method is superior to other methods on the MNIST and SVHN datasets, and slightly worse than ADBA on Shoe-Bag. But with batch sampling in our method,

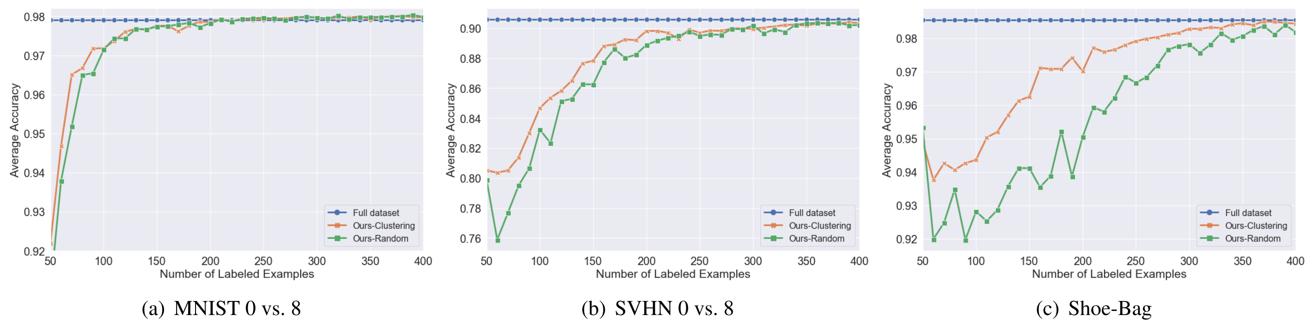


FIGURE 6. Comparison of initial seed sampling methods. Ours-Random selects initial seed randomly, and Ours-Clustering selects initial seed by our proposed method.

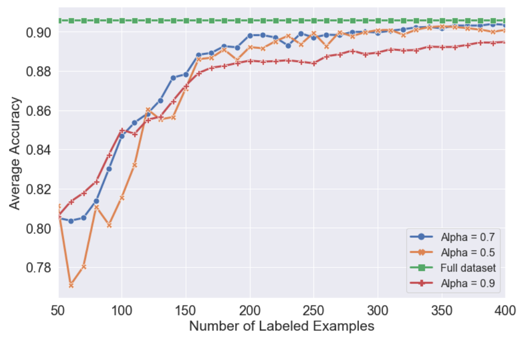


FIGURE 7. Evaluation of influence of parameter α in SVHN dataset.

the number of annotations of our method is much less than that of ADBA. Because with one boundary annotation, our method selects a batch of samples, while ADBA selects one sample.

E. EVALUATING OF DIFFERENT SETTINGS

We first tested the performance of our method with different initial seed sampling method including Ours-Random which select seed randomly and Our-Clustering which is the method adopted in our other experiments. Fig. 6 shows the evaluation results, from which we can see that the initial seed sampling method will impact the learning precision of the proposed method and the clustering-based method is better than random sampling.

We then tested the selection of parameter α in equation (9), and the result is shown in Fig. 7. From result we can see that when α is around 0.7 the performance of our method is better.

V. CONCLUSION

In this paper, we proposed an active learning method BALTS, which leverages batch sampling and direct boundary annotation with a two-stage sampling strategy. The first stage sampling of BALTS is initial seed sampling which selects a seed to enable decision boundary annotation. To ensure the selected seeds can approximate the distribution of data and with high diversity, clustering strategy and reject sampling are adopted in initial seed sampling. With direct boundary annotation, BALTS can avoid the complicated model re-training in each iteration, which will reduce much of the computation burden and let the oracle do not need to wait for samples.The second stage sampling is batch sampling which selects a batch of query samples according to a well-designed objective function that combines multiple criterion including uncertainty, representativeness and diversity. In this paper, we proposed a novel mechanism to maintain local representativeness and diversity of query samples by maximizing the reward of adding a new query sample. And the time complex of our batch sampling is $O(N_u)$. The experimental results proved that our proposed method perform well in three datasets without retraining the model in each iteration.

REFERENCES

[1] Y. Cheng, W. Xu, and Z. He, "Semi-supervised learning for neural machine translation," in *Joint Training for Neural Machine Translation*. Cham, Switzerland: Springer, 2019, pp. 25–40.

[2] U. Côté-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, and B. Gosselin, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, no. 4, pp. 760–771, Apr. 2019.

[3] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 403–412.

[4] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Batch mode active learning with applications to text categorization and image retrieval," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1233–1248, Sep. 2009.

[5] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu, "Semi-supervised SVM batch mode active learning for image retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Anchorage, AK, USA, Jun. 2008, pp. 24–26.

[6] R. Moskovitch, N. Nissim, D. Stopel, C. Feher, R. Englert, and Y. Elovici, "Improving the detection of unknown computer worms activity using active learning," in *Proc. Annu. Conf. Artif. Intell.* Cham, Switzerland: Springer, 2007, pp. 489–493.

[7] K. Malhotra, S. Bansal, and S. Ganapathy, "Active learning methods for low resource end-to-end speech recognition," in *Proc. Interspeech*, Sep. 2019, pp. 2215–2219.

[8] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang, "Representative sampling for text classification using support vector machines," in *Proc. Adv. Inf. Retr., 25th Eur. Conf IR Res. (ECIR)*, Pisa, Italy, Apr. 2003, p. 11.

[9] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Nov. 2001.

[10] A. K. Mccallum, "Employing em in pool-based active learning for text classification," in *Proc. 15th Int. Conf. Mach. Learn.* San Mateo, CA, USA: Morgan Kaufmann, 1998, pp. 350–358.

[11] Y. Guo and D. Schuurmans, "Discriminative batch mode active learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 593–600.

[12] M. Huijser and J. C. V. Gemert, "Active decision boundary annotation with deep generative models," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5286–5295.

[13] A. J. Joshi, F. Porikli, and N. Papanikolopoulos, "Multi-class active learning for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 2372–2379.

[14] Y. Yang, Z. Ma, F. Nie, X. Chang, and A. G. Hauptmann, "Multi-class active learning by uncertainty sampling with diversity maximization," *Int. J. Comput. Vis.*, vol. 113, no. 2, pp. 113–127, Jun. 2015.

[15] M.-F. Balcan, A. Broder, and T. Zhang, "Margin based active learning," in *Proc. Int. Conf. Comput. Learn. Theory*. Cham, Switzerland: Springer, 2007, pp. 35–50.

[16] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by committee," in *Proc. 5th Annu. Workshop Comput. Learn. Theory*, 1992, pp. 287–294.

[17] I. Dagan and S. P. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Proc. Mach. Learn.* Amsterdam, The Netherlands: Elsevier, 1995, pp. 150–157.

[18] S.-J. Huang, R. Jin, and Z.-H. Zhou, "Active learning by querying informative and representative examples," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 1936–1949, Oct. 2014.

[19] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, New York, USA, 2004, pp. 79–86, doi: 10.1145/1015330.1015349.

[20] Z. Bodo, Z. Minier, and L. Csato, "Active learning with clustering," *J. Mach. Learn. Res.*, vol. 16, pp. 127–139, Jan. 2011.

[21] M. Wang, F. Min, Z.-H. Zhang, and Y.-X. Wu, "Active learning through density clustering," *Expert Syst. Appl.*, vol. 85, pp. 305–317, Nov. 2017.

[22] S. Dasgupta and D. Hsu, "Hierarchical sampling for active learning," in *Proc. 25th Int. Conf. Mach. Learn.*, New York, USA, vol. 2008, pp. 208–215, doi: 10.1145/1390156.1390183.

[23] Y. C. Wu, "Active learning based on diversity maximization," *Appl. Mech. Mater.*, vols. 347–350, pp. 2548–2552, Aug. 2013.

[24] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," *J. Artif. Intell. Res.*, vol. 4, pp. 129–145, Mar. 1996.

[25] T. Zhang and F. Oles, "A probability analysis on the value of unlabeled data for classification problems," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 1191–1198.

[26] N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction," in *Proc. 18th Int. Conf. Mach. Learn.*, San Francisco, CA, USA, 2001, pp. 441–448.

[27] L. Shi, Y. Zhao, and J. Tang, "Batch mode active learning for networked data," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 2, pp. 1–25, Feb. 2012, doi: 10.1145/2089094.2089109.

[28] M. Wang, Y.-Y. Zhang, and F. Min, "Active learning through multi-standard optimization," *IEEE Access*, vol. 7, pp. 56772–56784, 2019.

[29] K. Brinker, "Incorporating diversity in active learning with support vector machines," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2003, pp. 59–66.

[30] Y. Guo, "Active instance sampling via matrix partition," in *Proc. 23rd Int. Conf. Neural Inf. Process. Syst.*, Red Hook, NY, USA, vol. 1, 2010, pp. 802–810.

[31] R. Chattopadhyay, Z. Wang, W. Fan, I. Davidson, S. Panchanathan, and J. Ye, "Batch mode active sampling based on marginal probability distribution matching," *ACM Trans. Knowl. Discovery Data*, vol. 7, no. 3, pp. 1–25, Sep. 2013, doi: 10.1145/2513092.2513094.

[32] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu, "Batch mode active learning and its application to medical image classification," in *Proc. 23rd Int. Conf. Mach. Learn.*, New York, NY, USA, 2006, pp. 417–424, doi: 10.1145/1143844.1143897.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[34] M. Ducoffe and F. Precioso, "Adversarial active learning for deep networks: A margin based approach," 2018, *arXiv:1802.09841*. [Online]. Available: http://arxiv.org/abs/1802.09841

[35] J.-J. Zhu and J. Bento, "Generative adversarial active learning," 2017, *arXiv:1702.07956*. [Online]. Available: http://arxiv.org/abs/1702.07956

[36] C. Mayer and R. Timofte, "Adversarial sampling for active learning," in *Proc. Workshop Appl. Comput. Vis.*, 2020, pp. 3071–3079. [Online]. Available: http://arxiv.org/abs/1808.06671

[37] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," 2016, *arXiv:1606.00704*. [Online]. Available: http://arxiv.org/abs/1606.00704

[38] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," in *Proc. 5th Int. Conf. Learn. Represent.*, Toulon, France, 2017, pp. 1–18.

[39] Y. Jia, J. Wang, C. Zhang, and X.-S. Hua, "Finding image exemplars using fast sparse affinity propagation," in *Proc. 16th ACM Int. Conf. Multimedia (MM)*, 2008, pp. 639–642.

[40] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[41] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[42] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2011, pp. 1–9.

[43] D. D. Lewis, "A sequential algorithm for training text classifiers," *ACM SIGIR Forum*, vol. 29, no. 2, pp. 13–19, Sep. 1995.

[44] Y. Yang and M. Loog, "A benchmark and comparison of active learning for logistic regression," *Pattern Recognit.*, vol. 83, pp. 401–415, Nov. 2018.

[45] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2008, pp. 1070–1079.

[46] X. Shen and C. Zhai, "Active feedback in ad hoc information retrieval," in *Proc. 28th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2005, pp. 59–66.

RONGHUA LUO was born in Hunan, China, in 1975. He received the B.Sc., M.S., and Ph.D. degrees in computer science from the Harbin Institute of Technology, China, in 1998, 2001, and 2005, respectively.

Since 2006, he has been a Teacher with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. He is currently an Associate Professor at the South China University of Technology. He has published more than 60 articles and holds 12 patents. His current research interests include machine intelligence, image analysis, and machine vision. He was a recipient of the Best Young Researcher Award of Chinese Association for Artificial Intelligence. He has undertaken two projects supported by the Nature Science Foundation of China and several projects supported by the Guangdong Science and Technology Department.

XIANG WANG was born in Chongqing, China, in 1994. He received the B.S. degree in electrical engineering and automation from the Chongqing University of Posts and Telecommunications, China, in 2017. He is currently pursuing the M.S. degree with the South China University of Technology, China. His research interests include machine learning, data mining, deep learning, and active learning.