Beam Search Optimized Batch Bayesian Active Learning

Jingyu Sun[1], **Hongjie Zhai**[2], **Osamu Saisho**[3], **Susumu Takeuchi**[1]
[1] NTT Computer and Data Science Laboratories
[2] NTT Software Innovation Center
[3] NTT Social Informatics Laboratories
jingyu.sun.pu@hco.ntt.co.jp, hongjie.zhai.wv@hco.ntt.co.jp, osamu.saisho.vm@hco.ntt.co.jp,
susumu.takeuchi.sp@hco.ntt.co.jp

Abstract

Active Learning is an essential method for label-efficient deep learning. As a Bayesian active learning method, Bayesian Active Learning by Disagreement (BALD) successfully selects the most representative samples by maximizing the mutual information between the model prediction and model parameters. However, when applied to a batch acquisition mode, like batch construction with greedy search, BALD suffers from poor performance, especially with noises of near-duplicate data. To address this shortcoming, we propose a diverse beam search optimized batch active learning method, which explores a graph for every batch construction by expanding the highest scored samples of a predetermined number. To avoid near duplicate beam branches (very similar beams generated from the same root and similar samples), which is undesirable for lacking diverse representations in the feature space, we design a self-adapted constraint within candidate beams. The proposed method is able to acquire data that can better represent the distribution of the unlabeled pool, and at the same time, be significantly different from existing beams. We observe that the proposed method achieves higher batch performance than the baseline methods on three benchmark datasets.

Introduction

Supervised learning performs well under the scenarios with access to a training set of high quality and quantity. However, as training models have grown more and more complicated in recent years, enormous training data has become required. However, obtaining a sufficient labeled set can be extremely difficult since the annotation processes are usually time-consuming and thus expensive, especially when expertise is required (Settles 2009). A promising way for solving this problem is active learning by iteratively selecting a minimal set of samples for oracles to label and then retraining the model (Ren et al. 2021). This ensures that with this minimized training set, the model can still maintain an acceptable prediction accuracy. Active learning has made a lot of practical impact in various tasks, such as object detection (Li et al. 2021), image classification (Wu et al. 2020), and natural language processing(NLP) tasks (Saisho et al. 2021).

According to the definition of active learning, the most informative samples should be selected for each training iteration from the pool of unlabeled data (Huang, Jin, and Zhou 2010). As a surrogate model in active learning, acquisition functions are responsible for detecting these samples. Multiple information theoretic heuristics can be used to design the acquisition functions that need to be both computationally efficient and accurate in judging the samples' informativeness. Some of them concern the samples that can reduce the model uncertainty most (Seung, Opper, and Sompolinsky 1992), while others take the data distribution into account when selecting the most representative samples (Yu et al. 2018). Recently, attempts have been made by (Kirsch, Rainforth, and Gal 2021) to combine the samples' informativeness and the evaluation dataset's distribution to achieve better performance with noisy data.

Within these heuristics, Bayesian Active Learning by Disagreement (BALD) (Houlsby et al. 2011) performs extraordinarily well by using the mutual information between the model prediction and the model parameters to estimate samples' acquisition scores. However, even though BALD performs well when a single sample is acquired for every training iteration, retraining the model after adding every single labeled sample is undesired due to the expensive computational cost. Instead, batch acquisition that can also achieve a comparable predictive accuracy as single sampling is ideal. BatchBALD (Kirsch, Van Amersfoort, and Gal 2019) selects a batch of multiple informative samples jointly using a greedy algorithm that adds samples into the batch one by one. In BatchBALD, joint mutual information representing the joint informativeness of samples in the batch is calculated after every sample is added to the batch. This approach suffers from two problems. First, as a greedy acquisition algorithm, it selects only one best candidate sample for labeling at each time step. Choosing only one candidate may be suitable for the current time step. However, eventually, another beam, instead of the initial candidate may become the best choice for the whole optimization process. Second, jointly estimating the informativeness of all the combinations of the remaining pool set and existing samples in the batch for every time step of batch construction can be very computationally expensive. Adaptively escaping the near duplicate samples in the unlabeled pool is desirable.

We introduce a diverse beam search algorithm for BALD that selects alternative samples of a predetermined number (beam width) at each time step of batch construction. Ap-

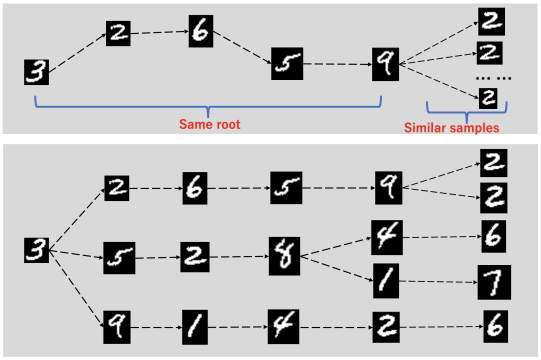

Figure 1: Acquisition batch construction of conventional beam search (top) and beam search with self-adapted constraint (bottom)

parently, a higher beam width gives a better cover of feature representation and leads to a higher possibility of good learning performances. However, higher beam width also means more computational cost; similar to greedy search, adaptively escaping some near duplicate samples is also crucial for reducing the computation cost in beam search optimized active learning. Furthermore, we observe that the result beams usually turn out to be only slightly different and often from a common parent as in Figure 1(top), which is undesirable because the diverse features and multimodal nature of samples cannot be captured and compared for selecting the best batches. To address these issues, we apply a self-adapted constraint within candidate beams helping produce beams that are significantly different from each other as illustrated in Figure 1(bottom). Moreover, the constraint adapts itself in accordance with the density in the pool that intuitively escapes most of the near-duplicating samples from the pool and consequently reduces the computational cost for constructing the beams.

We report results on three benchmark datasets. Experimental results show that our method consistently outperforms baseline methods including BALD and BatchBALD in terms of both model quality metrics and acceptable computational cost. Moreover, we find that both of these improvements mainly benefit from the self-adapted constraint when compared with regular beam search without any constraint. Overall, our method produces beams of high quality for each time step of batch construction under acceptable computation and memory cost compared with baseline methods.

Preliminaries

In this section, we first revisit the Bayesian neural network in active learning, and then describe how to apply top-s, greedy, and plain beam search techniques (without any constraint) into active learning. We will be using these batch acquisition algorithms for comparison with the proposed method.

Bayesian Active Learning

Bayesian active learning aims at maximizing the information gain from the selected samples while reducing the possible hypotheses of models. The merit of incorporating a Bayesian neural network into active learning is that it can express information gain from labeled samples in terms of the model's predictive entropies, which is more tractable than other approximation approaches (Houlsby et al. 2011).

BALD (Gal, Islam, and Ghahramani 2017) minimizes the uncertainty of the model parameters by selecting samples that can maximize the decrease in expected posterior entropy $H[\omega|\mathcal{D}]$ of model. Here we assume the model's latent parameters, ω, and model's predictions, $p(y|\omega)$. Having an unlabeled pool $\mathcal{D}_{pool}$, and current training set $\mathcal{D}$, we can infer a posterior distribution over the model's parameters as $p(\omega|\mathcal{D})$. The decrease in expected posterior entropy if we try to add a sample (x, y) from the pool $\mathcal{D}_{pool}$ to the training set $\mathcal{D}$ looks like:

$$\arg\max_x \mathbb{H}[\omega|\mathcal{D}] - \mathbb{E}_{y\sim p(y|x,\mathcal{D})}[\mathbb{H}[\omega|y, x, \mathcal{D}]] \quad (1)$$

which equals the conditional mutual information between unknown output $y \sim p(y|x, \mathcal{D})$ and the model parameters ω:

$$\mathbb{I}[\omega; y|x, \mathcal{D}] = \mathbb{I}[y; \omega|x, \mathcal{D}] \quad (2)$$

The right term is easier to compute since we can variationally approximate the distribution over parameters using MC-dropout (Gal and Ghahramani 2016), which is much easier to implement.

Batch Acquisition: Top-S and Greedy Because retraining the model sequentially (after every single sample is added from the pool to the training set) is extremely time-consuming, adding a batch of samples at each acquisition step is desirable. BALD selects a batch for every acquisition step by taking the top s mutual information scored samples (Gal and Ghahramani 2016) (Janz, van der Westhuizen, and Hernández-Lobato 2017):

$$\arg\max_{x_1,\ldots,x_s\subseteq\mathcal{D}_{pool}} \sum_{i=1}^{s} \mathbb{I}(y_i; \omega|x_i, D_{train}) \quad (3)$$

(Kirsch, Van Amersfoort, and Gal 2019) find this naive approach can only assure that the acquired samples are individually informative (top k informative in the pool), but not necessarily jointly informative. They propose BatchBALD to solve this problem by selecting samples whose joint informativeness is high:

$$\arg\max_{x_1,\ldots,x_s\subseteq\mathcal{D}_{pool}} \mathbb{I}(y_1, ..., y_s; \omega|x_1, ..., x_s, D_{train})) \quad (4)$$

As illustrated in Algorithm 1, a greedy algorithm is used to approximately choose samples to construct a batch. Joint mutual entropies are calculated after every sample in the pool is added into the batch sequentially. This ensures that every candidate sample is most informative at least for the current time step. However, it cannot be guaranteed to still be most suitable after the whole batch is constructed.

In light of this, to enhance the batch performance, we try to apply beam search for constructing a batch in our study.

Algorithm 1: Greedy algorithm for batch acquisition

Input: Acquisition batch size s, unlabeled dataset $\mathcal{D}_{pool}$, model parameters ω under the train data $\mathcal{D}^T$ in this acquisition step T, let $\alpha_T(\mathcal{A}) = \mathbb{I}(Y_\mathcal{A}; \omega | X_\mathcal{A}, \mathcal{D}^T)$

$\mathcal{A}^{(0)} = \varnothing$
for $i = 1$ to s **do**
  for all $d \in \mathcal{D}_{pool} \backslash \mathcal{A}^{(i-1)}$ **do**
    $s_d \leftarrow \alpha_T(\mathcal{A}^{(i-1)} \bigcup \{d\})$
  end for
  $\mathcal{A}^{(i)} \leftarrow \mathcal{A}^{(i-1)} \bigcup \{\arg\max_d s_d\}$
end for
return $\mathcal{A}^{(s)}$

Obviously, there will be trade-offs between the batch performance and beam computational cost. Our method attempts to achieve high batch performance while maintaining a low computational cost.

Plain Beam Search for Batch Acquisition

As a widely used heuristic search algorithm, beam search is able to cover more diverse features of data than greedy search. At the same time, it stays computationally tractable compared with exhaustive search, which tries every possible combination of all the samples. We first design a preliminary experiment illustrating how plain beam search works in active learning for comparison with our method.

Preliminary Experiment Design In our preliminary experiment, we apply plain beam search into active learning as a limited-width breadth-first search for a batch of samples maximizing the acquisition scores at each search step. Denote set of $\mathcal{K}$ solutions after time $t-1$ as $\mathcal{B}_{t-1} = \{\mathbf{b}_{t-1}^1, ..., \mathbf{b}_{t-1}^\mathcal{K}\}$, given a set of remaining samples in the pool $\mathcal{D}_{pool}$, search a candidate beam set $\mathcal{B}_t = \{\mathbf{b}_t^k\}_{k=1}^\mathcal{K}$, in which one beam looks like: $\mathbf{b}_t^k = (d_0^k, ..., d_t^k)$:

$$\mathcal{B}_t = \underset{\mathbf{b}^1,...,\mathbf{b}^\mathcal{K} \in \mathbb{B}_t}{\arg\max} \sum \alpha_T(\mathbf{b}^k),\ \alpha_T(\mathbf{b}) = \underbrace{\mathbb{I}(Y_\mathbf{b}; \omega | X_\mathbf{b}, \mathcal{D}_{train}^T)}_{\text{Mutual information scores}}$$

$$\text{s.t.} \underbrace{\mathbf{b}^i \neq \mathbf{b}^j, \forall i \neq j \text{ and } i, j \in [\mathcal{K}]}_{\text{only remove exactly identical beams}} \tag{5}$$

Search is through: $\mathbb{B}_t = \{\mathbf{b}_{t-1}^k \bigcup d \mid \mathbf{b}_{t-1}^k \in \mathcal{B}_{t-1}^k, d \in \mathcal{D}_{pool}\}_{k=1}^\mathcal{K}$, $d = x$ with a search span $\mathcal{K} \times |\mathcal{D}_{pool}|$, on AL acquisition step T.

Approach

In this section, we present a diverse beam search optimized active learning method, which consists of a diverse beam search algorithm and a self-adapted constraint within beams.

Motivation and Overview

In the preliminary experiment that we conducted on ReMNIST digits (Kirsch, Van Amersfoort, and Gal 2019), we directly apply plain beam search (beam width of 32 and acquisition size of 5) to BALD for the batch construction of each acquisition step T. Obviously, if we apply beam search without any constraints, during the whole beam search process for constructing a batch, we can only remove the exactly identical beams. In other words, there could be many nearly identical beams in our result just as illustrated in Figure 1 (top), which leads to a lack of diverse feature representation though our batch construction.

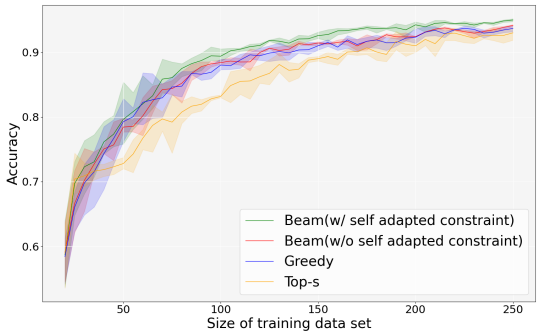

Figure 2: Performance (acquisition size of 5) on ReMNIST of Top-s algorithm, Greedy algorithm, and Beam Algorithm with/without self-adapted constraint: Batch acquisition with plain beam search did not outperform others, while the proposed method (Batch acquisition using beam search with self-adapted constraint) did

Figure 2 shows that batch acquisition using beam search fails to improve the active learning accuracy significantly in comparison with top-s algorithm and greedy algorithm. Again, we infer that this poor performance should be attributed to there being too many similar beams that are inadequate to reflect the whole feature space.

Therefore, intuitively, finding beams that are more diverse, just as shown in Figure 1 (bottom), is desired. Following this intuition, we formalize the task of generating diverse beams in active learning by defining a dissimilarity constraint within beams for each acquisition step T. The main idea is to select a beam element with the dissimilarity constraint conditioned on previous beams to make sure that for each beam search time step t, the generated beams would not be nearly identical. We further extend this idea by designing a self-adapted constraint that not only guarantees the diversity of beams but also takes the data density of the unlabeled pool into account for batch acquisition.

Diverse Beam Search for Active Learning

We want to make every new beam solution more diverse than previous beams, so following the normal M-best solutions (Batra et al. 2012), we should greedily update the m^{th} beam at time t (note that in this paper we denote the time step in beam search as t; while the active learning acquisition time step as T) under the dissimilarity constraint. Having a dissimilarity function $\Pi(\mathbf{b}^m, \mathbf{b}^i)$, we enforce the new beam solution $\mathbf{b}^m$ to be dissimilar to the existing ones $\{\mathbf{b}^i\}_{i=1}^{m-1}$, and the dissimilarity should exceed a threshold k_i.

$$\mathcal{B}_t^m = \underset{\mathbf{b}^m \in \mathbb{B}_t}{\arg\max}\, \alpha_T(\mathbf{b}^k), \alpha_T(\mathbf{b}) = \mathbb{I}(Y_{\mathbf{b}}; \omega | X_{\mathbf{b}}, \mathcal{D}_{train}^T)$$
$$\text{s.t.} \underbrace{\Pi(\mathbf{b}^m, \mathbf{b}^i) \geq k_i, \forall i < m}_{dissimilarity-constrained} \text{ and } m < \mathcal{K} \quad (6)$$

Since data samples forming beams are naturally different from each other in terms of features and surrounding density, the threshold k_i on dissimilarity should be self-adapted with these characteristics.

In this case, we consider k_i to be self-adapted with density around the candidate data point. Let $\Delta(\cdot, \cdot)$ measures similarity between beams, and $\Theta(\mathbf{b}^i)$ be the self-adapted threshold of the new beam solution $\mathbf{b}^i$, when trying to find the m^{th} beam candidate, we can rewrite the constraint as:

$$\underbrace{\mathcal{B}_t^m = \underset{\mathbf{b}^m \in \mathbb{B}_t}{\arg\max}\, \alpha_T(\mathbf{b}^k), \alpha_T(\mathbf{b}) = \mathbb{I}(Y_{\mathbf{b}}; \omega | X_{\mathbf{b}}, \mathcal{D}_{train}^T)}_{\text{Find } m^{th} \text{ new beam solution}}$$
$$\text{s.t.} \underbrace{\Delta(\mathbf{b}^m, \mathbf{b}^i) \leq \Theta(\mathbf{b}^i), \forall i < m}_{\text{Self adapted constraint}} \text{ and } m < \mathcal{K} \quad (7)$$

Self-Adapted Constraint

Both the similarity function Δ and the density threshold function Θ can take various forms - $e.g.$ KL divergence of model prediction, or natural clustering distance on the basis of the raw features of input samples. In our study, we estimate the similarity and constraint with consideration of active learning's iterative training process. In this section, we introduce how the self-adapted constraint works for selecting a beam set with abundant diversity.

Beam Similarity Estimation Since active learning is an iterative learning loop in which the feature space (in view of the model) is actually changing after every training step, we choose the feature space generated by the model to estimate the similarity between samples. As shown in Figure 3, having the discriminative model $\Phi = \theta^{cls}(\theta^{fea})$ consisting of two general layers, the feature extraction layer θ^{fea} ($e.g.$, convolution layers or recurrent neural network (RNN) feature extraction layers), and the classification layer θ^{cla} ($e.g.$, flatten layers, fully connected layers), we estimate the similarity between different data points d_i, d_j in the pool $\mathcal{D}_{pool}$ after every acquisition/training step T, following Gaussian Radial Basis Function (to make similarity vary sharply though data points):

$$\delta_T(d_i, d_j) = \exp(-\gamma ||\theta_T^{fea}(d_i) - \theta_T^{fea}(d_j)||^2)$$
$$d_i, d_j \in \mathcal{D}_{pool} \quad (8)$$

Then, the similarity between beams can be calculated in Hausdorff style, which estimates the smaller similarity in the two oriented similarities (from $\mathbf{b}^m$ to $\mathbf{b}^n$, and the opposite) :

$$\Delta(\mathbf{b}^m, \mathbf{b}^n) = \min[h(\mathbf{b}^m, \mathbf{b}^n), h(\mathbf{b}^n, \mathbf{b}^m)] \quad (9)$$

in which:

$$h(\mathbf{b}^m, \mathbf{b}^n) = \inf_{d_i \in \mathbf{b}^m} \sup_{d_j \in \mathbf{b}^n} \delta(d_i, d_j) \quad (10)$$

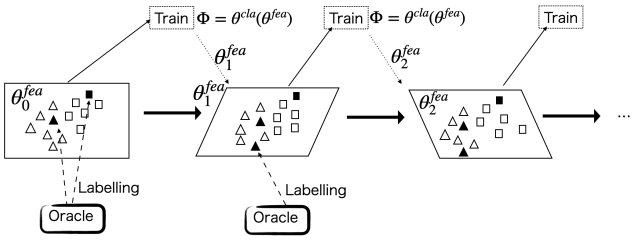

Figure 3: Similarity estimation after every acquisition(retraining) step in active learning process

Density Based Threshold of Similarity In the preliminary experiment, we observe that for some samples ($e.g.$, Label 1 in ReMNIST) the neighbor samples around them obtain very high similarities with themselves, while for other samples ($e.g.$, Label 4 or 5), their neighbors obtained low similarity values. This infers, for different beams, the density around the beam element samples in the unlabeled pool varies sharply. In other words, instead of applying a fixed threshold of similarity for all the beams, a self-adapted constraint in terms of the density around the sample should be used for identifying the near duplicate beams. Moreover, since the samples' feature embeddings are changing during the whole active learning process, the density around a sample in the unlabeled pool, let us say d_i, also varies across every training iteration.

Taking these factors into consideration, we estimate the similarity of a data point d_i on behalf of its surrounding density. First, we find $j - nearest$ neighbors of d_i by using the samples' similarity estimation in Equation. 8. This forms a cluster of samples around d_i, and then we obtain the sample d_r on the radii of the cluster:

$$\underbrace{\underset{d_1,...,d_J \in \mathcal{D}_{pool}}{\arg\max} \sum \delta(d_i, d_j),\ j \in \{1, ..., J\}}_{\text{find j nearest neighbors}} \quad (11)$$

$$d_r = \underbrace{\underset{d \in \{d_1,..,d_J\}}{\arg\min}\ \delta(d_i, d)}_{\text{point on the radii}} \quad (12)$$

We calculate the density of sample d_i as the similarity between it and the its' j^{th} nearest neighbor d_r as follows:

$$\epsilon(d_i) = \delta(d_i, d_r) \quad (13)$$

Threshold of the beam similarity for self-adapted constraint in accordance with the density is calculated as:

$$\Theta(\mathbf{b}^i) = \frac{1}{\gamma} \min\{\epsilon(d_c)\}_{d_c \in \mathbf{b}^i} \quad (14)$$

Here, γ indicates the degree of the diversity on the basis of which we require the learning process to perform the filtering out of near duplicated beams.

By estimating the similarity and self-adapted constraint in this way, when try to filter out the near duplicate beams, we are able to identify the areas of high density and then adapt a tight similarity constraint (compared with low density areas) to the beams containing these samples.

Algorithm 2: Beam search with self-adapted constraint for batch acquisition in active learning

Input: Beam size $\mathcal{K}$, acquisition batch size s, unlabeled dataset $\widehat{\mathcal{D}_{pool}}$, model parameters ω under the train data in this step, for beam $\mathbf{b}$, let $\alpha_T(\mathbf{b}) = \mathbb{I}(Y_{\mathbf{b}}; \omega | X_{\mathbf{b}}, \mathcal{D}^T_{train})$

Output: Acquired batch $\mathcal{A}_{acquired} = \{d_1, d_2, ..., d_s\}$

1: $\mathscr{B}^{(0)} = \{(\varnothing, 0)\}$
2: **for** $i = 1$ to s **do**
3: $\quad \mathscr{B}^{(i)} = \{\}$
4: $\quad$ **for all** $(\mathbf{b}^{parent}, s) \in \mathscr{B}^{(i-1)}$ **do**
5: $\quad\quad$ **for all** $d \in \widehat{\mathcal{D}_{pool}} \backslash \mathbf{b}^{parent}$ **do**
6: $\quad\quad\quad$ **if** $\mathbf{b}^m = \mathbf{b}^{parent} \bigcup \{d\}$ not in $\mathscr{B}^{(i)}$ **then**
7: $\quad\quad\quad\quad$ **if** $\Delta(\mathbf{b}^m, \mathbf{b}^j) \leq \Theta(\mathbf{b}^j)\ \forall (\mathbf{b}^j, s) \in \mathscr{B}^{(i)}$ **then**
8: $\quad\quad\quad\quad\quad \mathscr{B}^{(i)} \leftarrow \mathscr{B}^{(i)} \bigcup \{(\mathbf{b}^{parent} \bigcup \{x\}, \alpha_T(\mathbf{b}^m)\}$
9: $\quad\quad\quad\quad$ **end if**
10: $\quad\quad\quad$ **end if**
11: $\quad\quad$ **end for**
12: $\quad$ **end for**
13: $\quad \mathscr{B}^{(i)} \leftarrow$ take top-$\mathcal{K}$ scored $(\mathcal{A}, scores)$ from $\mathscr{B}^{(i)}$
14: **end for**
15: $(\mathcal{A}_{acquired}, score) \leftarrow$ take top-1 scored $(\mathcal{A}, scores)$ from $\mathscr{B}^{(k)}$
16: **return** $\mathcal{A}_{acquired}$

Efficient Implementation

Next, we design a quick adapting algorithm to conduct the scoring process more efficiently. First, we conduct data reduction on the unlabeled pool set to reduce the pool size to $\widehat{\mathcal{D}_{pool}} = \frac{|\mathcal{D}_{pool}|}{\lambda_2 \mathcal{K}}$ by a greedy search for samples of low similarities in terms of δ with samples that are already in the reduced pool. Here let $\mathcal{K}$ be the beam width in the diverse beam search and λ_2 be degree of diversity. Generally, our method is formalized in Algorithm 2. If we compare it with Algorithm 1, we find two major differences:

(1) Beam search is applied by keeping $\mathcal{K}$ candidates for every beam search step t. Here suppose the scoring function $\alpha_T(\mathcal{B})$ performs under the same time cost. Since we reduce the pool size to $\widehat{\mathcal{D}_{pool}}$ and $\lambda_2 > 1$, we will not consume more time for beam scoring than the greedy algorithm.

(2) Note that we put the self-adapted constraint in line 7, before calculating the score of beams. This indicates that we escape a bunch of beams without the time-consuming joint mutual information calculation, which will significantly reduce the processing time of each active learning iteration.

Experiments

We evaluate the performance of our method and other baseline methods including BALD (top-s acquisition algorithm) and BatchBALD (greedy acquisition algorithm). We also do the ablation study on self-adapted constraint, beam width, and beam diversity. Here we report results on different sizes of training set size.

Active Learning Performance

The datasets, MNIST((LeCun et al. 1998)), ReMNIST((Kirsch, Van Amersfoort, and Gal 2019)), and CIFAR-10, are used in our experiments.

The network architectures and the training procedure are set identically to (Kirsch, Van Amersfoort, and Gal 2019). Correspondingly, we choose 10 MC dropout samples for ReMNIST and 100 of them for MINIST and CIFAR-10. We use a convolutional neural network (CNN) consisting of two blocks of convolution (with 32 and 64 5×5 filters), dropout, max pooling, and ReLU. Following these two blocks, there are a fully connected layer of 128 hidden units and MC dropout. All dropouts are with a probability of 0.5. All models are optimized with the Adam optimizer with a learning rate of 0.001 and betas(0.9, 0.999). The initial samples are randomly selected but of the same number per class. We also stop early after three epochs with declining accuracy on validation set for avoiding overfitting. Each experiment is repeated five times with different seeds and different initial samples. Median of these five trials and the lower/upper quartiles are used to draw the accuracy figures. All beam searches are conducted with a width of 32.

ReMNIST We first examine our method's performance on ReMNIST in which there are many near duplicate data points containing isotropic Gaussian noise with standard deviation of 0.1. We use a validation set of 2000 samples, a balanced test set of 20,000, and randomly select 20 samples for initial training set (2 per class).

The test accuracy of batch acquisition size 5 and 10 are shown in Figures 2 and 4, respectively. Our method can significantly outperform all other batch acquisition algorithms in terms of accuracy. The performance is consistent across all acquisition steps and is more obvious when we increase the acquisition size. As also demonstrated in (Kirsch, Van Amersfoort, and Gal 2019), a dataset with near duplicate samples such like ReMNIST leads to poor performance if we do not consider batch diversity. We can infer from our result that the proposed method can achieve abundant beam diversity while being applied to these datasets.

MNIST and CIFAR-10 For MNIST and CIFAR-10, we use a validation set of 1000 samples and a balanced test set of 10,000 (1000 per class). We set acquisition size of 10 for both of them. The training set is initialized with 20 samples (2 per class). We tune γ and set it as 1.1.

For both of the two datasets and ReMNIST, our method dominates other batch acquisition algorithms as shown in Table. 1. Moreover, compared with BatchBALD, the total time including acquisition and training time decreased about 30% for all of these datasets. This should be attributed to the constraint for escaping about half of the samples during the pool resizing and acquisition.

Ablation Study

Next, we discuss how the self-adapted constraint and beam width influence the experimental results.

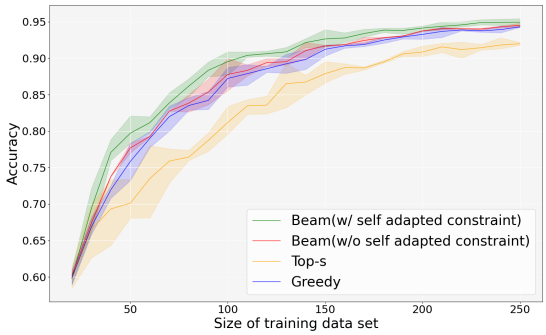

Figure 4: Performance (acquisition size of 10) on ReMNIST of Top-s Algorithm, Greedy algorithm (BatchBALD) and Beam algorithm with/without self-adapted constraint: the proposed method (Batch acquisition using beam search with self-adapted constraint) outperforms others

	MNIST	ReMNIST	CIFAR-10
Random	$0.891^{\pm 0.013}$	$0.887^{\pm 0.011}$	$0.292^{\pm 0.017}$
top-s	$0.922^{\pm 0.011}$	$0.910^{\pm 0.009}$	$0.295^{\pm 0.012}$
greedy	$0.941^{\pm 0.008}$	$0.924^{\pm 0.007}$	$0.301^{\pm 0.011}$
Proposed	$\mathbf{0.948^{\pm 0.006}}$	$\mathbf{0.939^{\pm 0.005}}$	$\mathbf{0.309^{\pm 0.012}}$

Table 1: Average accuracy (over five runs) when training set size reaches 200

Dataset	Beam optimized active learning	
	w/o constraint	w/ constraint
ReMNIST	0.918 ± 0.006	$\mathbf{0.927 \pm 0.005}$
MNIST	0.928 ± 0.004	$\mathbf{0.931 \pm 0.004}$
CIFAR-10	0.289 ± 0.006	$\mathbf{0.306 \pm 0.005}$

Table 2: Average accuracy (over five runs): Learning with constraint outperforms that without any constraint

Ablation Study on Self-Adapted Constraint We examine the performance of beam search optimized active learning with/without self-adapted constraint. In Table 2, we report the average accuracy of our model when the training set size reaches 150. The results indicate the effectiveness of self-adapted constraint on all of these datasets.

Ablation Study on Beam Width We examine the performance of beam search optimized active learning with self-adapted constraint, but under different beam width as shown in Figure 5. We set acquisition size of 10, MC dropout number of 10 as well as other parameters all fixed as in the section Active Learning Performance. We train our model with different beam width 2^n where $n \in \{0, 1, 2, 3, 4, 5, 6\}$ on ReMNIST and report the average accuracy. Beam search with higher beam width performs better in general. However, after increasing beam width to 32, no more obvious enhancement is observed.

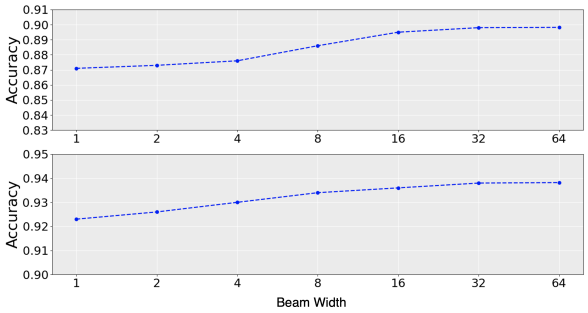

Figure 5: Ablation study on beam width when training set size reaches 100 (top) and 200 (bottom)

γ	100 data points acquired	
	Acquisition Size 5	Acquisition Size 10
0.001	0.878 ± 0.006	0.865 ± 0.004
0.1	0.882 ± 0.004	0.869 ± 0.003
0.3	0.887 ± 0.003	0.879 ± 0.006
0.5	0.897 ± 0.007	0.893 ± 0.006
0.9	$\mathbf{0.897 \pm 0.006}$	$\mathbf{0.894 \pm 0.005}$
1.2	0.897 ± 0.008	0.893 ± 0.006
1.4	0.889 ± 0.006	0.878 ± 0.006
100	0.601 ± 0.006	0.506 ± 0.005

Table 3: Average accuracy (over three runs): Beam diversity strength affects model performance

Ablation Study on Beam Diversity We examine the performance of beam search optimized active learning with self-adapted constraint, but under different beam diversity that is adjusted by the parameter γ. We set acquisition size of 5 and 10, MC dropout number of 10 as well as other parameters all fixed as in the section Active Learning Performance. As shown in Table 3, we train our model with different beam diversity by changing γ where $\gamma \in \{0.001, 0.1, 0.3, 0.5, 0.9, 1.2, 1.4, 100\}$ on ReMNIST, and report the average accuracy when the size of training dataset reaches 100. We find that γ in range (0.5, 1.2) works well without an obvious difference in performance, but with a value too large like 100, the model accuracy drops even more sharply than in greedy search.

Discussion

In this section, we analyze the possible performance bound of the proposed method theoretically and give suggestions for setting proper hyperparameters that are related to both the model performance and computation cost.

First, as described above, when try to find the m^{th} solution during every search iteration, we consider maximizing the score of each beam under the self-adapted constraint as follows:

$$\max_{\mathbf{b}^m \in \mathbb{B}_t} \alpha_T(\mathbf{b}^k)$$
$$\text{s.t.} \Delta(\mathbf{b}^m, \mathbf{b}^i) \leq \Theta(\mathbf{b}^i), \forall i < m \qquad (15)$$

We refer to the above formulation as $DivBeam(\Delta, \Theta)$. We can find out the accuracy bound of the proposed method by studying the Lagrangian relaxation of $DivBeam(\Delta, \Theta)$, in which Lagrange multipliers $\boldsymbol{\lambda} = \{\lambda_i | i \in 1, 2, ..., m-1\}, \boldsymbol{\lambda} \geq 0$ are designed for the penalty of the dissimilarity constraint.

$$\mathcal{L}(\boldsymbol{\lambda}) = \max_{\mathbf{b}^m \in \mathbb{B}_t} \alpha_T(\mathbf{b}^k) - \sum_{i=1}^{m-1} \lambda_i(\Delta(\mathbf{b}^m, \mathbf{b}^i) - \Theta(\mathbf{b}^i)) \tag{16}$$

Intuitively, by solving the Lagrangian dual problem $\min_{\boldsymbol{\lambda} \geq 0} \mathcal{L}(\boldsymbol{\lambda})$, we can find a upper-bound on the value of $DivBeam(\Delta, \Theta)$, while $\mathcal{L}(\cdot)$ is concave in $\boldsymbol{\lambda}$. In other words, $\min_{\boldsymbol{\lambda} \geq 0} \mathcal{L}(\boldsymbol{\lambda}) \geq DivBeam(\Delta, \Theta)$. As presented by (Zhao, Luh, and Wang 1999), assume both Δ and Θ can be approximated into high-order potentials, eventhough the dual problem for solving this case is nondifferentiable, a surrogate subgradient of $\boldsymbol{\lambda}$ can be used to estimate the final model performance. (Vijayakumar et al. 2018) tunes $\boldsymbol{\lambda}$ with fixed values directly for maximizing $\mathcal{L}(\cdot)$ representing a linear trade-off between model score and diversity of solutions. In our method, $\boldsymbol{\lambda}$ is considered to be a very large value forcing the model to consider diversity of beam solutions first. This can considerably save computing cost for escaping a lot of score computation. Instead of tuning $\boldsymbol{\lambda}$, we tune the diversity strength γ for scaling the count of beams solutions that we are going to make escape from score computing.

Analysis of Hyperparameters

Following the analysis above, we discuss the impact of the hyperparameters.

Strength of Diversity The proposed method trades off between the diversity of beam solutions and the mutual information score. Hyper-parameter γ is responsible for adjusting the diversity strength. If γ is set too high, diversity of beams can over-power the mutual information score. Since the result beam can only contain a determined beam size $\mathcal{K}$, an oversized γ may lead to ignoring several informative data points and result in lower model accuracy. Moreover, since more diversity in the selected beams means more data points that are similar to the existing ones will escape from the mutual information computing, a high γ also leads to a lower computation cost. On the other hand, if γ is set too low, it may lead to many near-duplicate solutions in the final beam set and result in poor performance during active learning. Lower diversity that escapes fewer beam solutions leads to higher computation costs of mutual information calculation.

Although we assume setting an appropriate value for γ is a task-dependent problem, in our ideal application, we set γ to make sure that we can make at least $1 - 1/\mathcal{K}$ candidate beam solutions escape. This can ensure our method maintains a nearly equivalent computing cost compared with the baseline solutions. The additional computation cost for computing the similarities of beam solutions can be offset by a higher escaping rate or the implementation of data reduction.

Beam Size Beam size trades off between model performance and computation cost. A big beam size can explore the search space better in contrast with a small one, but it also causes longer computation time. Besides, increasing beam size does not lead to a higher upper bound of the method, but gives a better chance to reach it. With diversity strength following the settings in our experiment, a beam size in the range 30 to 40 is desired.

Batch Size If batch size increases too much, differences between beams naturally become unobvious; thus, the effectiveness of our method will decrease correspondently. At the same time, our method will consume more time. But we consider that there should be a tradeoff between the model's time-consuming and human labor for annotating.

Related Work

Different heuristic approaches have been proposed for maintaining the samples' diversity in batch acquisition of active learning. (Azimi et al. 2012) leverages the availability of high-quality and efficient sequential active-learning policies. (Hoi, Jin, and Lyu 2006) maximizes the Fisher Information of a classification model. (Guo and Schuurmans 2007) formulates the diverse instance selection task as a continuous optimization problem while taking unlabeled samples into consideration. (Janz, van der Westhuizen, and Hernández-Lobato 2017) restricts the MC dropout inference samples while calculating the acquisition scores for more batch diversity. (Ash et al. 2020) detects samples which are disparate and high-magnitude embedded in a hallucinated gradient space for incorporating both predictive uncertainty and sample diversity into selected batches.

Some researchers have investigated generating diverse beams (or other structured outputs) from probabilistic models. Theoretically, (Batra et al. 2012) formalizes diverse M-best tasks as the DivMBest problem. Extending DivMBest, (Gimpel et al. 2013) introduces a family of dissimilarity functions for Machine Translation for generating diverse translation beams. (Kirillov et al. 2015) considers the DivMBest problem as a greedy approximate task. In NLP tasks, (Li et al. 2016b) produces diverse decoding for RNNs. (Li et al. 2016a) produces more diverse beams by maximizing mutual information in RNNs. (Li and Jurafsky 2016) increases diversity in the N-best list by discouraging beams from the same root. (Vijayakumar et al. 2018) incorporates diversity constraints with groups of candidate sequences during decoding for generating beams that are significantly different from each other. (Cohen and Beck 2019), and (Ott et al. 2018) improves beam search in terms of degradation.

Conclusion

In this work, we introduced a diverse beam search optimized active learning method, in which a self-adapted constraint is applied within candidate beams to acquire significantly different samples and better represent the distribution of the unlabeled pool in the dataset. Experiment results show that our method consistently outperforms other baseline methods and the plain beam search across all the experiments of batch mode active learning without too much extra computational cost substantially.

References

Ash, J. T.; Zhang, C.; Krishnamurthy, A.; Langford, J.; and Agarwal, A. 2020. Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds. *ArXiv*, abs/1906.03671.

Azimi, J.; Fern, A.; Fern, X. Z.; Borradaile, G.; and Heeringa, B. 2012. Batch Active Learning via Coordinated Matching. In *International conference on machine learning (ICML)*.

Batra, D.; Yadollahpour, P.; Guzman-Rivera, A.; and Shakhnarovich, G. 2012. Diverse m-best solutions in markov random fields. In *European Conference on Computer Vision (ECCV)*, 1–16.

Cohen, E.; and Beck, C. 2019. Empirical Analysis of Beam Search Performance Degradation in Neural Sequence Models. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 1290–1299. PMLR.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International conference on machine learning (ICML)*, 1050–1059. PMLR.

Gal, Y.; Islam, R.; and Ghahramani, Z. 2017. Deep bayesian active learning with image data. In *International Conference on Machine Learning (ICML)*, 1183–1192. PMLR.

Gimpel, K.; Batra, D.; Dyer, C.; and Shakhnarovich, G. 2013. A Systematic Exploration of Diversity in Machine Translation. In *EMNLP 2013 - 2013 Conference on Empirical Methods in Natural Language Processing*.

Guo, Y.; and Schuurmans, D. 2007. Discriminative Batch Mode Active Learning. In *Advances in Neural Information Processing Systems (NIPS)*, 593–600. Citeseer.

Hoi, S. C.; Jin, R.; and Lyu, M. R. 2006. Large-scale text categorization by batch mode active learning. In *Proceedings of the 15th international conference on World Wide Web*, 633–642.

Houlsby, N.; Huszár, F.; Ghahramani, Z.; and Lengyel, M. 2011. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*.

Huang, S.-J.; Jin, R.; and Zhou, Z.-H. 2010. Active learning by querying informative and representative examples. *Advances in neural information processing systems (NeurIPS)*, 23: 892–900.

Janz, D.; van der Westhuizen, J.; and Hernández-Lobato, J. M. 2017. Actively Learning what makes a Discrete Sequence Valid. *ArXiv*, abs/1708.04465.

Kirillov, A.; Savchynskyy, B.; Schlesinger, D.; Vetrov, D. P.; and Rother, C. 2015. Inferring M-Best Diverse Labelings in a Single One. *2015 IEEE International Conference on Computer Vision (ICCV)*, 1814–1822.

Kirsch, A.; Rainforth, T.; and Gal, Y. 2021. Active Learning under Pool Set Distribution Shift and Noisy Data. *International Conference on Machine Learning (ICML)*.

Kirsch, A.; Van Amersfoort, J.; and Gal, Y. 2019. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32: 7026–7037.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, B. 2016a. A Diversity-Promoting Objective Function for Neural Conversation Models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 110–119. San Diego, California: Association for Computational Linguistics.

Li, J.; Galley, M.; Brockett, C.; Gao, J.; and Dolan, W. B. 2016b. A Diversity-Promoting Objective Function for Neural Conversation Models. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Li, J.; and Jurafsky, D. 2016. Mutual Information and Diverse Decoding Improve Neural Machine Translation. *ArXiv*, abs/1601.00372.

Li, Y.; Fan, B.; Zhang, W.; Ding, W.; and Yin, J. 2021. Deep active learning for object detection. *Information Sciences*, 579: 418–433.

Ott, M.; Auli, M.; Grangier, D.; and Ranzato, M. 2018. Analyzing Uncertainty in Neural Machine Translation. *ArXiv*, abs/1803.00047.

Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Gupta, B. B.; Chen, X.; and Wang, X. 2021. A Survey of Deep Active Learning. *ACM Comput. Surv.*, 54(9).

Saisho, O.; Ohguro, T.; Sun, J.; Imamura, H.; Takeuchi, S.; and Yokozeki, D. 2021. Human Knowledge Based Efficient Interactive Data Annotation via Active Weakly Supervised Learning. In *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, 332–335.

Settles, B. 2009. Active learning literature survey. *Computer sciences technical report 1648, University of Wisconsin-Madison*.

Seung, H. S.; Opper, M.; and Sompolinsky, H. 1992. Query by committee. In *Conference on Learning Theory (COLT)*.

Vijayakumar, A. K.; Cogswell, M.; Selvaraju, R. R.; Sun, Q.; Lee, S.; Crandall, D. J.; and Batra, D. 2018. Diverse Beam Search for Improved Description of Complex Scenes. In *Advancement of Artificial Intelligence (AAAI)*.

Wu, J.; Sheng, V. S.; Zhang, J.; Li, H.; Dadakova, T.; Swisher, C. L.; Cui, Z.; and Zhao, P. 2020. Multi-label active learning algorithms for image classification: Overview and future promise. *ACM Computing Surveys (CSUR)*, 53(2): 1–35.

Yu, H.; Yang, X.; Zheng, S.; and Sun, C. 2018. Active learning from imbalanced data: A solution of online weighted extreme learning machine. *IEEE transactions on neural networks and learning systems*, 30(4): 1088–1103.

Zhao, X.; Luh, P. B.; and Wang, J. 1999. Surrogate gradient algorithm for Lagrangian relaxation. *Journal of optimization Theory and Applications*, 100(3): 699–712.