BABA: Beta Approximation for Bayesian Active Learning

Jae Oh Woo
Samsung SDS Research America
San Jose, CA 95134
jaeoh.woo@aya.yale.edu

Abstract

This paper introduces a new acquisition function under the Bayesian active learning framework, namely BABA. It is motivated by previously well-established works BALD [19], and BatchBALD [27] which capture the mutual information between the model parameters and the predictive outputs of the data. Our proposed measure, BABA, endeavors to quantify the normalized mutual information by approximating the stochasticity of predictive probabilities using Beta distributions. BABA outperforms the well-known family of acquisition functions, including BALD and BatchBALD. We demonstrate this by showing extensive experimental results obtained from MNIST and EMNIST datasets.

1 Introduction

In modern machine learning, as the dataset size is getting bigger and bigger for training a complex model like detection and segmentation, labeling data by humans becomes very expensive. It is necessary to design a systematic way how to prioritize the labeling dataset to save cost. This labeling prioritization process is called active learning, originated from Cohn et al. [8]. The active learning problem is well-aligned with a subset selection problem that can find the most efficient but minimal subset from the data pool [32, 20, 12, 31, 44, 45, 43]. The difference is that active learning is typically an iterative process where a model is trained and a collection of data points is selected to be labeled from an unlabelled data pool.

It is now commonly accepted that standard deep learning models do not capture model uncertainty correctly. The simple predictive probabilities are usually erroneously described as model confidence. So there is a risk that a model can be misdirecting its outputs with high confidence. However, the predictive distribution generated from Bayesian deep learning models better captures the uncertainty from the data [18]. Therefore, we focus on developing an active learning framework in the Bayesian deep neural network model by leveraging the MC dropout method to approximate the Gaussian process [18]. Under this Bayesian deep learning framework, only a limited number of works, including BALD and BatchBALD, have been proposed in the literature [22, 21, 23, 18, 19, 27, 52].

In this paper, we extend and improve recent advances in the Bayesian deep learning regime. We generalize the notion of the joint entropy between model parameters and the predictive outputs by leveraging a point process type entropy [30, 17, 44, 10, 3]. By approximating the marginals using Beta distributions, we then derive an explicit formula of the upper bound of the joint entropy by estimating Beta parameters from Bayesian deep learning models. Then we empirically demonstrate the out-performance of the proposed measure from MNIST and EMNIST datasets.
2 Background

2.1 Problem formulation

We follow the notions introduced in BALD [19] and BatchBALD [27]. We write an unlabeled dataset \( \mathcal{D}_{\text{pool}} \) and the labeled training set \( \mathcal{D}_{\text{training}} \subseteq \mathcal{D}_{\text{pool}} \) in each active learning iteration. We denote by \( \mathcal{D}_{\text{training}}^{(n)} \) if it’s necessary to indicate the specific \( n \)-th iteration step. Given \( \mathcal{D}_{\text{training}} \), we train a Bayesian deep neural network model \( \Phi \) with model parameters \( \omega \sim p(\omega) \). Then for a data point \( x \) given \( \mathcal{D}_{\text{training}} \), the Bayesian deep neural network \( \Phi \) produces the prediction probability:

\[
\Phi(x, \omega) := (P_1(x, \omega), \cdots, P_C(x, \omega)) \in \Delta^C,
\]

where \( \Delta^C = \{ (p_1, \cdots, p_C) : p_1 + \cdots + p_C = 1, p_i \geq 0 \text{ for each } i \} \) and \( C \) is the number of classes. For the final class output \( Y \), it is assumed to be a multinoulli distribution (or categorical distribution):

\[
Y(x, \omega) := \begin{cases} 
1 & \text{with probability } P_1(x, \omega) \\
\vdots & \\
C & \text{with probability } P_C(x, \omega).
\end{cases}
\]

For the sake of brevity, we sometimes omit \( x \) or \( \omega \) by writing \( \Phi(\omega), P_i(\omega), Y(\omega) \) or \( \Phi, P_i, Y \) unless we need further clarifications on each data point \( x \). Under this formulation, the oracle (active learning algorithm) selects a subset of data points to add to the next training set, i.e. at \( (n+1) \)-th iteration, the training set is determined by

\[
\mathcal{D}_{\text{training}}^{(n+1)} = \mathcal{D}_{\text{training}}^{(n)} \cup \{ \text{Next batch selected by Oracle} \}.
\]

Once the next batch is selected, the selected batch will be labeled. This means that the ground truth label information of the selected data is added in training set \( \mathcal{D}_{\text{training}}^{(n+1)} \) in the next round. Then the goal in active learning is to minimize the number of oracle queries until it reaches a certain level of prediction accuracy.

2.2 Examples of acquisition functions

In this section, we list up well-known acquisition functions to obtain our benchmarks, then we shall compare the performances with the proposed measure, BABA in Section 5.

1. **Random**: \( \text{Rand}[x] := U(\omega') \) where \( U(\cdot) \) is a uniform distribution which is independent to \( \omega \). Random acquisition function assigns a random uniform value on \([0, 1]\) to each data point.

2. **Variation Ratio [16]**: \( \text{VarRatio}[x] := 1 - \max_i E P_i \). Variation Ratio captures the uncertainty with respect to the maximal expected predictive probability among all classes.

3. **Predictive entropy [40]**: \( \text{PredEntropy}[x] := - \sum_i (E P_i) \log (E P_i) \). Predictive entropy is the Shannon entropy with respect to the expected predictive probability.

4. **Mean standard deviation (Mean STD) [22, 1]:** \( \text{MeanSTD}[x] := \frac{1}{C} \sum_i \sqrt{E P_i^2 - (E P_i)^2} \). Mean standard deviation captures the average of the standard deviations for each marginal distribution.

5. **BALD (Bayesian active learning by disagreement) [21, 19]**:

\[
\text{BALD}[x] := \mathcal{I}(\omega, Y(x, \omega)) \approx \mathcal{I}(\Phi(x, \omega), Y(x, \omega)),
\]

where \( \mathcal{I}(\cdot, \cdot) \) represents a mutual information between random measures. BALD captures the mutual information between the model parameters and the predictive output of the data point. In practice, we calculate the mutual information between the predictive output and the predictive probabilities.

In a multiple acquisition size setting, we simply add the above acquisition functions for each data point \( x_i \):

\[
\text{AcqFunc}[x_1, \cdots, x_n] := \sum_{i=1}^n \text{AcqFunc}[x_i],
\]
where $\text{AcqFunc} \in \{\text{Rand, VarRatio, PredEntropy, MeanSTD, BALD}\}$.

On the other hand, BatchBALD [27] calculates the joint mutual information for many data points. In this way, we can accommodate the dependency between data points. Then we can remove the redundancy in the selection of the next batch.

6. BatchBALD [27]:

$$\text{BatchBALD}[x_1, \cdots, x_n] := I(\omega, Y(x_1, \omega), \cdots, Y(x_n, \omega))$$

$$\approx I(\Phi(x_1, \omega), \cdots, \Phi(x_n, \omega), Y(x_1, \omega), \cdots, Y(x_n, \omega)).$$

In practice, finding the optimal (maximally-valued) selection of BatchBALD is NP-hard. Therefore BatchBALD [27] uses a greedy algorithm that guarantees $1/e$-approximation by proving the submodularity of the mutual information.

Along with the algorithms mentioned earlier, there are many other approaches to finding the next batch, especially in multiple acquisitions and mostly non-Bayesian scenario. e.g., Core-Set method [39], Variational adversarial active learning [42], Margin based adversarial active learning [11], Determinantal point process-based active learning [5], and Wasserstein distance-based active learning [41]. In the following, although we propose and test the selection of the following labeling candidates in a multiple acquisition scenario to illustrate the performance of our approach, the main scope of this paper is to propose a novel acquisition measure for developing a new computational framework by using Beta approximations.

3 Bayesian deep learning model

We adopt the Bayesian neural network framework introduced in Gal et al. [18]. The core idea in the Bayesian neural network is leveraging the dropout feature to generate a distribution of the predictive probability as an output. It turns out that it is equivalent to an approximation to a particular Bayesian model such as Gaussian Process [18, 37].

3.1 Beta approximation

In this section, we consider a Bayesian neural network model $\Phi$ as a random measure, i.e., stochastic process parametrized by $D_{\text{training}}$ over the data set $D_{\text{pool}}$. Given a data point $x \in D_{\text{pool}}$, $\Phi(x, \omega)$ produces a random probability distribution in a simplex $\Delta^C$. This analogy has a close connection with the construction of random discrete distribution, originally introduced by Kingman [25]. After then, random measure construction has been extensively developed in Bayesian nonparametrics, and it is well-known that Dirichlet probability having Beta marginals plays the central role in the
construction of the random discrete distribution \[26\,14\,35\,35\,6\,33\,38\]. It is the main motivation of the Beta approximation.

Although we do not have rigorous probabilistic proof, we may justify the underlying Beta approximation process. Following the approach by Ferguson \[14\], a Dirichlet probability can be constructed through a collection of independent Gamma distributions. On the other hand, each marginal in Gaussian Process (approximated by Bayesian neural network) in the soft-max output having dependent components follows a log-normal distribution (before the normalization, but after the exponentiation in soft-max). Then if we can ignore the dependency of components of log-normal distributions because of the shape similarity between a log-normal distribution and Gamma distribution, the construction of random probability from log-normal distributions would produce an approximated Dirichlet distribution. Then we may assume that the marginal distribution would approximately follow the Beta distribution.

Figure \[1\] shows an example of Beta approximations obtained from MNIST dataset (See Section \[5.1\]). \(P_1, \cdots, P_10\) show each marginal distribution of the predictive probability of each digit. We can visually confirm that the Beta approximation is a reasonable approximation.

In practice, once we estimate the sample mean and sample variance for each marginal, we can estimate two parameters of the Beta distribution by applying the following Lemma.

**Lemma 3.1.** Assume that \(P \sim \text{Beta}(\alpha, \beta)\). If \(\mathbb{E}P = m\) and \(\text{Var}P = \sigma^2\), then
\[
\alpha = \frac{m^2(1 - m)}{\sigma^2} - m, \quad \beta = \left(\frac{1}{m} - 1\right) \alpha.
\]

**Proof.** If \(P \sim \text{Beta}(\alpha, \beta)\), \(\mathbb{E}P = \frac{\alpha}{\alpha + \beta} = m\) and \(\text{Var}P = \frac{\alpha \beta}{(\alpha + \beta)^2 (\alpha + \beta + 1)} = \sigma^2\). Solving the equation with respect to \(\alpha\) and \(\beta\), then the Lemma follows. \(\square\)

### 3.2 Entropy and mutual information in Bayesian neural network

In this section, we briefly discuss the Bayesian neural network from an information-theoretic perspective. We can formulate the Bayesian neural network \(\Phi\) as an encoder-decoder communication channel. The sender sends a message \((x, \omega)\) with a random key \(\omega\) through the channel, then the receiver receives a message \(y(x, \omega)\). Figure 2 illustrates a diagram in this communication process.

![Figure 2: Bayesian neural network channel](image)

Then a natural question is to find the mutual information between the input and the output of the channel related to estimate a capacity of the network communication channel \[9\,15\]. The mutual information is, in fact, one of the acquisition measures, BALD \[21\,19\].

\[
\text{BALD}[x] := I((x, \omega), Y(x, \omega)) = I(\omega, Y(x, \omega))
\]

where \(I(\cdot, \cdot)\) represents a mutual information between two quantities. In practice, controlling \(\omega\) is difficult, but we can control the family of the encoded messages \(\Phi(x, \omega)\) in a tractable manner \[19\,24\,47\]. So by assuming that the encoder approximately keeps the information of the original message, we can approximately estimate the mutual information by calculating the mutual information between the encoded message and the channel output.

\[
\text{BALD}[x] \approx I(\Phi(x, \omega), Y(x, \omega)).
\]

Then another natural question is to calculate the joint entropy of \(\Phi(x, \omega)\) and \(Y(x, \omega)\), denoting by \(\mathcal{J}(\Phi(x, \omega), Y(x, \omega))\). To calculate the joint entropy, we need to note that \(\Phi(x, \omega)\) is on a continuous domain \(\Delta^C\), and \(Y(x, \omega)\) is on a discrete domain \([C] = \{1, \cdots, C\}\). This means that the usual notions of Shannon entropy and differential entropy \[9\] cannot be applied. However, we can apply the notion of the point process entropy which can accommodate both continuous and discrete domain random structures \[30\,17\,34\,10\,3\]. We may write a Janossy density function \[10\] of \((\Phi(x, \omega), Y(x, \omega))\) on \(\Delta^C \times [C]\) as follows:

\[
j(p_1, \cdots, p_C, y = i) = p_i f(p_1, \cdots, p_C),
\]

(1)
where \( f(\cdot) \) is a density function of \( \Phi(\mathbf{x}, \omega) \). Then the joint entropy of \( \Phi(\mathbf{x}, \omega) \) and \( Y(\mathbf{x}, \omega) \) can be defined as

\[
\mathcal{H}(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega)) = -\sum_{i=1}^{C} \int_{\Delta^C} j(p_1, \cdots, p_C, y = i) \log j(p_1, \cdots, p_C, y = i) \, dp_1 \cdots dp_C.
\]

(2)

By plugging (1) into (2), we have the following identity.

\[
\mathcal{H}(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega)) = H(Y(\mathbf{x}, \omega)) + \mathbb{E}_Y \left[ h(\Phi(\mathbf{x}, \omega) | Y) \right],
\]

where \( H(\cdot) \) represents the usual Shannon entropy, and \( h(\cdot) \) represents the usual differential entropy. Here we handle the dependency within the components by approximating \( \Phi(\mathbf{x}, \omega) \) to be Dirichlet distribution, but it is not yet clear for us how to treat the entropy in a more tractable way. Instead, by applying Jensen’s inequality, we may derive an upper bound of the joint entropy:

\[
\mathcal{H}^U(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega)) \leq H(Y) - \sum_i \mathbb{E}_{P_i} \left[ P_i \log \left( \frac{P_i}{\mathbb{E}P_i} f(P_i) \right) \right] = - \sum_i \mathbb{E}_{P_i} \left[ P_i \log (P_i f(P_i)) \right],
\]

where we ambiguously write \( f(\cdot) \) to be a density function for each \( P_i \). And we define a quantity of the upper bound of the joint entropy:

\[
\mathcal{H}^U(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega)) := - \sum_i \mathbb{E}_{P_i} \left[ P_i \log (P_i f(P_i)) \right].
\]

Now we are ready to apply the Beta approximation for each marginal; then, we can directly calculate the upper bound of the joint entropy.

**Theorem 3.2.** Assume that each \( P_i \sim \text{Beta}(\alpha_i, \beta_i) \). Then we can find the explicit formula of the Beta approximated upper joint entropy as follows:

\[
\mathcal{H}^U_{\text{Beta}}(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega)) := - \sum_i \mathbb{E}_{P_i} \left[ P_i \log (P_i f(P_i)) \right]
\]

\[
= \sum_i \left[ \frac{\alpha_i}{\alpha_i + \beta_i} \log B(\alpha_i, \beta_i) - \alpha_i \frac{B(\alpha_i + 1, \beta_i)}{B(\alpha_i, \beta_i)} [\Psi(\alpha_i + 1) - \Psi(\alpha_i + \beta_i + 1)] \right.
\]

\[
- (\beta_i - 1) \frac{B(\alpha_i + 1, \beta_i)}{B(\alpha_i, \beta_i)} [\Psi(\beta_i) - \Psi(\alpha_i + \beta_i + 1)],
\]

where \( B(\cdot, \cdot) \) is the Beta function, and \( \Psi(\cdot) \) is the Digamma function.

**Proof.** See Appendix A.2 for the details. \( \square \)

## 4 Normalized mutual information

Normalized mutual information has been widely used in many applications, including to quantify the quality of the clustering [46, 48, 50, 29, 49]. We propose a novel definition of the normalized mutual information for the Bayesian neural network to leverage it as an acquisition function:

\[
\text{NMI}[\mathbf{x}] := \mathcal{I}(\omega, Y(\mathbf{x}, \omega)) = \frac{\text{BALD}[\mathbf{x}]}{\mathcal{H}(\omega, Y(\mathbf{x}, \omega))} \approx \frac{\text{BALD}[\mathbf{x}]}{\mathcal{H}(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega))}.
\]

In contrast to the typical definition of the normalized mutual information, we apply the absolute value for the denominator as the joint entropy could have negative values, so \( \text{NMI}[\mathbf{x}] \in [0, +\infty] \).

### 4.1 BABA

Similar to the normalized mutual information, we define a new acquisition function, BABA. BABA is defined as a ratio between BALD and the Beta approximated upper joint entropy:

\[
\text{BABA}[\mathbf{x}] := \frac{\text{BALD}[\mathbf{x}]}{\mathcal{H}^U_{\text{Beta}}(\Phi(\mathbf{x}, \omega), Y(\mathbf{x}, \omega))}.
\]
From the previous section, we have the following relationship:

\[ \mathcal{D} (\Phi(x, \omega), Y(x, \omega)) \leq \mathcal{D}^U (\Phi(x, \omega), Y(x, \omega)) \approx \mathcal{D}_{\text{Beta}} (\Phi(x, \omega), Y(x, \omega)). \]

Then we see that BABA is, in most cases, approximately a lower bound/or an upper bound of the normalized mutual information:

\[ \begin{cases} \text{BABA}[x] \leq \text{NMI}[x] & \text{if } \mathcal{D} (\Phi(x, \omega), Y(x, \omega)) \geq 0, \\ \text{BABA}[x] \geq \text{NMI}[x] & \text{if } \mathcal{D}_{\text{Beta}} (\Phi(x, \omega), Y(x, \omega)) \leq 0. \end{cases} \]

We also note that in the active learning iteration, BABA picks the point when \( \mathcal{D}_{\text{Beta}} (\Phi(x, \omega), Y(x, \omega)) \approx 0 \) with a relatively high BALD value. But we are not fully aware of the underlying information-theoretic interpretation of the high BABA value case. We leave it as an open question.

### 4.2 BatchBABA

Similar to BatchBALD [27], we may extend BABA to a multi-sized acquisition scenario. Following the same process derived in the previous section for BABA, we may define the Beta approximated upper bound of the joint entropy:

\[ \mathcal{D}^U_{\text{Beta}}(x_1, \cdots, x_n) := H(Y(x_1), \cdots, Y(x_n)) + \sum_i [\mathcal{D}^U_{\text{Beta}}(\Phi(x_i), Y(x_i)) - H(Y(x_i))]. \]

Detailed derivations are explained in Appendix A.3. Then we define the measure of BatchBABA as follows:

\[ \text{BatchBABA}[x_1, \cdots, x_n] := \frac{\text{BatchBALD}[x_1, \cdots, x_n]}{\mathcal{D}^U_{\text{Beta}}(x_1, \cdots, x_n)}. \]

In practice, the first term in \( \mathcal{D}^U_{\text{Beta}}(x_1, \cdots, x_n) \) can be calculated by the same way explained in BatchBALD [27] See Equations (10)-(12)], and the second term can be calculated from Theorem 3.2. We, then, apply the same greedy algorithm by replacing the quantity BatchBALD [27] See Algorithm 1] with BatchBABA. We note that we cannot guarantee \( 1/e \)-approximation for BatchBABA since we are not aware of BatchBABA satisfies the sub-modularity or not.

### 5 Experimental results

In this section, we demonstrate the performance of BABA from MNIST and EMNIST datasets. For all experiments, we leverage the same active learning infrastructure implemented for the original BatchBALD [27] See Section 4]. Therefore experimented model architectures are the same as the original BatchBALD setting. We add two more options to apply BABA and BatchBABA algorithms. We use a mix of A100, RTX 3090, GTX 1080Ti GPUs to collect results for all experiments. High memory GPUs are particularly helpful for speeding up experiments with multiple acquisition sizes for BatchBALD and BatchBABA.

In both MNIST and EMNIST experiments, we test two different acquisition scenarios - the size 1 and 20. For the acquisition size of 1, we try Random, Variation Ratios, Predictive Entropy, Mean STD, BALD, and BABA algorithms with 5 repetitions. For the acquisition size of 20, we test Random, BALD, BatchBALD, and BABA, BatchBABA algorithms with 10 repetitions. In our experiments, we generate 100 MC dropout samples to estimate quantities involved in acquisition functions, and we set the early stopping level to be at least 10 epochs longer than experiments used in the original BatchBALD experiments [27]. This allows having the model parameters to converge sufficiently. We note that this longer stopping level is particularly helpful for BALD (and BatchBALD) to improve its performance, so it partially resolves the concern about the noisy estimator raised in [27] See Section 6]. We also present one of the results from a smaller stopping tolerance level in Appendix A.4.

To compare different acquisition methods, we report two metrics in a format of mean ± standard deviation; 1) relative accuracy with respect to random (higher value is better), and 2) relative negative log-likelihood with respect to random (a lower value is better). The detailed formula for these metrics is explained in Appendix A.6.

[https://github.com/BlackHC/BatchBALD](https://github.com/BlackHC/BatchBALD)
5.1 MNIST

MNIST [23] is the most popular and primary dataset to validate the performance of image-based deep learning models initially. The MNIST dataset contains 60,000 balanced training images and 10,000 testing images. For the initial training, we fix a subset to select 2 data points for each class to match the initial training set for all methods.

**Acquisition size** 1. Figure 3 shows the mean accuracy and log-likelihood curves obtained from the experiment with acquisition size 1 in MNIST. Table 1 shows relative accuracy and log-likelihood with respect to the random acquisition function. We may visually and quantitatively confirm that BABA outperforms any other well-known methods in both the accuracy and the negative log-likelihood.

![Figure 3: MNIST with acquisition size 1 - Active learning accuracy (left) and negative log-likelihood (right) curves. Each curve was obtained from the mean of the 5 repeated experiments.](image)

| Method               | Relative Accuracy to Random | Relative Negative Log-Likelihood to Random |
|----------------------|----------------------------|-------------------------------------------|
| Random               | 0 ± 1.78%                  | 0 ± 11.20%                                |
| Variation Ratios     | 3.16 ± 1.36%               | -21.93 ± 11.86%                           |
| Predictive Entropy   | 1.86 ± 1.59%               | -11.44 ± 11.03%                           |
| Mean STD             | 3.42 ± 1.62%               | -21.43 ± 13.80%                           |
| BALD                 | 4.25 ± 1.61%               | -24.67 ± 12.68                            |
| BABA (ours)          | **5.24 ± 1.52%**           | **-38.01 ± 11.77%**                       |

Table 1: Relative accuracy with respect to random (left column) and relative log-likelihood with respect to random (right column) for MNIST with acquisition size 1.

**Acquisition size** 20. Figure 4 shows the mean accuracy and log-likelihood curves obtained from the experiment with acquisition size 20 in MNIST. Table 2 provides the relative accuracy and relative log-likelihood with respect to random. Even in the multiple acquisition scenario, BABA still outperforms.

![Figure 4: MNIST with acquisition size 20 - Active learning accuracy (left) and negative log-likelihood (right) curves. Each curve was obtained from the mean of the 10 repeated experiments.](image)

Figure 4: MNIST with acquisition size 20 - Active learning accuracy (left) and negative log-likelihood (right) curves. Each curve was obtained from the mean of the 10 repeated experiments.
| Method            | Relative Accuracy to Random | Relative Negative Log-Likelihood to Random |
|-------------------|-----------------------------|-------------------------------------------|
| Random            | 0 ± 1.18%                   | 0 ± 13.79%                                |
| BALD              | 0 ± 2.33%                   | -2.92 ± 16.16%                            |
| BatchBALD         | 3.58 ± 1.32%                | -23.51 ± 15.37%                           |
| BABA (ours)       | 4.13 ± 1.40%                | -36.50 ± 14.51%                           |
| BatchBABA (ours)  | 3.54 ± 1.59%                | -27.81 ± 16.15%                           |

Table 2: Relative accuracy with respect to random (left column) and relative log-likelihood with respect to random (right column) for MNIST with acquisition size 20.

5.2 EMNIST

The EMNIST dataset [7] is a set of handwritten character digits with a 28 × 28 pixel image format aligning with the MNIST dataset. This experiment uses the EMNIST balanced dataset consisting of 47 classes with 112,800 images in the training set and 18,800 images in the test set. For the initial training, we fix a subset to select a 1 data point for each class to match the initial training set for all methods.

**Acquisition size 1.** Figure 5 shows the mean accuracy and log-likelihood curves obtained from the experiment with acquisition size 1 in the EMNIST. Table 3 shows the relative accuracy and relative log-likelihood with respect to random. We note that most of the existing methods fail to perform better than random. It implies that EMNIST active learning problem is a complex problem without any good strategies. BABA still outperforms all other measures, including random.

![Figure 5: EMNIST with acquisition size 1 - Active learning accuracy (left) and negative log-likelihood (right) curves. Each curve was obtained from the mean of the 5 repeated experiments.](image)

| Method            | Relative Accuracy to Random | Relative Negative Log-Likelihood to Random |
|-------------------|-----------------------------|-------------------------------------------|
| Random            | 0 ± 2.59%                   | 0 ± 8.06%                                 |
| Variation Ratios  | -0.08 ± 4.19%               | 6.49 ± 8.63%                              |
| Predictive Entropy| -5.70 ± 4.52%               | 13.77 ± 8.16%                             |
| Mean STD          | 0.55 ± 3.34%                | 1.07 ± 6.77%                              |
| BALD              | -1.07 ± 3.86%               | 4.53 ± 7.27%                              |
| BABA (ours)       | 3.96 ± 4.56%                | -3.53 ± 9.06%                             |

Table 3: Relative accuracy with respect to random (left column) and relative log-likelihood with respect to random (right column) for EMNIST with acquisition size 1.

**Acquisition size 20.** Figure 6 shows the mean accuracy and log-likelihood curves obtained from the experiment with an acquisition size of 20 in the EMNIST. Table 4 shows the relative accuracy and relative log-likelihood with respect to random. We note that BatchBABA outperforms other methods. But in a computational cost perspective, BABA is usually 4 ~ 10 times faster than BatchBALD or BatchBABA depending on GPU environments since it does not consider the dependency between data points. Nevertheless, the performance of BABA is comparable with others.

Table 4: Relative accuracy with respect to random (left column) and relative log-likelihood with respect to random (right column) for EMNIST with acquisition size 20.

| Method            | Relative Accuracy to Random | Relative Negative Log-Likelihood to Random |
|-------------------|-----------------------------|-------------------------------------------|
| Random            | 0 ± 2.39%                   | 0 ± 8.06%                                 |
| Variation Ratios  | -0.08 ± 4.19%               | 6.49 ± 8.63%                              |
| Predictive Entropy| -5.70 ± 4.52%               | 13.77 ± 8.16%                             |
| Mean STD          | 0.55 ± 3.34%                | 1.07 ± 6.77%                              |
| BALD              | -1.07 ± 3.86%               | 4.53 ± 7.27%                              |
| BABA (ours)       | 3.96 ± 4.56%                | -3.53 ± 9.06%                             |

Acquisition size 20.
Figure 6: EMNIST with acquisition size 20 - Active learning accuracy (left) and negative log-likelihood (right) curves. Each curve was obtained from the mean of the 10 repeated experiments.

| Method          | Relative Accuracy to Random | Relative Negative Log-Likelihood to Random |
|-----------------|-----------------------------|------------------------------------------|
| Random          | $0 \pm 3.35\%$             | $0 \pm 7.02\%$                          |
| BALD            | $-5.65 \pm 2.87\%$         | $5.11 \pm 7.39\%$                       |
| BatchBALD       | $3.27 \pm 2.88\%$          | $-4.02 \pm 6.30\%$                      |
| BABA (ours)     | $3.27 \pm 4.38\%$          | $-3.18 \pm 8.05\%$                      |
| BatchBABA (ours)| $3.68 \pm 3.49\%$          | $-5.47 \pm 7.73\%$                      |

Table 4: Relative accuracy with respect to random (left column) and relative log-likelihood with respect to random (right column) for EMNIST with acquisition size 20.

6 Open problems

Throughout this paper, we have made a lot of approximations. Although our proposed measure, BABA (including BatchBABA), shows promising performances in MNIST and EMNIST datasets, there are many open problems to explore further to elaborate/or validate our approach rigorously in the future.

- Beta approximation has been made from heuristic observations by inspecting the marginal distribution of the predictive probability. We conjecture that the Beta approximation phenomenon would be universal, but it lacks the theoretical guarantee on Beta approximation of the marginal distribution.

- The definition of the normalized mutual information in this paper is a novel measure. To make the measure useful in an active learning scenario, we apply the absolute value to make the denominator non-negative, but the value range can be $[0, +\infty]$ instead of $[0, 1]$. This infinite range phenomenon is closely related to the difference between the Shannon entropy and the differential entropy. But we are not fully aware of the underlying meaning of this measure, and it requires further investigation to understand the nature of this new kind of normalized mutual information.

- The computational cost for BatchBABA (and BatchBALD) becomes very expensive as the acquisition size increases. Moreover, BatchBABA has been loosely defined in this paper when we derive the upper bound of the joint entropy. We leave the further improvement of the BatchBABA not only in the accuracy but the computational cost as an open problem.

7 Conclusion

In this paper, we proposed new acquisition functions, BABA and BatchBABA, for deep Bayesian active learning by leveraging a Beta approximation and introducing a new type of normalized mutual information by extending the previous BALD and BatchBALD.

We see that our proposed acquisition measures require further investigations to understand them better. But our experimental results show a promising prospect in this new direction. To the best of our knowledge, we, for the first time, demonstrate the applicability of the normalized mutual information under a deep Bayesian active learning framework. We also look forward to having many follow-up studies with similar lines of broad applications beyond the active learning problems.
Acknowledgments and Disclosure of Funding

The author specially thanks the helpful discussion with and comments from François Baccelli in Inria and The University of Texas at Austin.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. In Gabriel Brostow, Tae-Kyun Kim, Stefanos Zafeiriou, and Krystian Mikolajczyk, editors, Proceedings of the British Machine Vision Conference (BMVC), pages 57.1–57.12. BMVA Press, September 2017.

[2] Evan Archer, II Memming Park, and Jonathan W Pillow. Bayesian entropy estimation for countable discrete distributions. The Journal of Machine Learning Research, 15(1):2833–2868, 2014.

[3] François Baccelli and Jae Oh Woo. On the entropy and mutual information of point processes. In 2016 IEEE International Symposium on Information Theory (ISIT), pages 695–699. IEEE, 2016.

[4] Frits Beukers. Special functions (encyclopedia of mathematics and its applications 71). Bulletin of the London Mathematical Society, 33(1):116–127, 2001.

[5] Erdem Bıyık, Kenneth Wang, Nima Anari, and Dorsa Sadigh. Batch active learning using determinantal point processes. arXiv preprint arXiv:1906.07975, 2019.

[6] Tamara Broderick, Michael I Jordan, Jim Pitman, et al. Beta processes, stick-breaking and power laws. Bayesian analysis, 7(2):439–476, 2012.

[7] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In 2017 International Joint Conference on Neural Networks (IJCNN), pages 2921–2926. IEEE, 2017.

[8] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. Journal of artificial intelligence research, 4:129–145, 1996.

[9] Thomas M Cover. Elements of information theory. John Wiley & Sons, 1999.

[10] Daryl J Daley and David Vere-Jones. An introduction to the theory of point processes: volume II: general theory and structure, volume 2. Springer Science & Business Media, 2007.

[11] Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. arXiv preprint arXiv:1802.09841, 2018.

[12] AP Dvoredsky. Some results on convex bodies and banach spaces. 1961.

[13] Abbas El Gamal and Young-Han Kim. Network information theory. Cambridge university press, 2011.

[14] Thomas S Ferguson. A bayesian analysis of some nonparametric problems. The annals of statistics, pages 209–230, 1973.

[15] Gerald B Folland. Real analysis: modern techniques and their applications, volume 40. John Wiley & Sons, 1999.

[16] L.C. Freeman. Elementary Applied Statistics: For Students in Behavioral Science. For Students in Behavioral Science. Wiley, 1965.

[17] J. Fritz. An approach to the entropy of point processes. Periodica Mathematica Hungarica, 3(1-2):73–83, 1973.

[18] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059. PMLR, 2016.

[19] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In International Conference on Machine Learning, pages 1183–1192. PMLR, 2017.

[20] Dorit S Hochbaum. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In Approximation algorithms for NP-hard problems, pages 94–143. 1996.
[21] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

[22] Michael Kampffmeyer, Arnt-Borre Salberg, and Robert Jenssen. Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–9, 2016.

[23] Kirthevasan Kandasamy, Jeff Schneider, and Barnabás Póczos. Bayesian active learning for posterior estimation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 3605–3611, 2015.

[24] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[25] John FC Kingman. Random discrete distributions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 37(1):1–15, 1975.

[26] John FC Kingman. The population structure associated with the ewens sampling formula. *Theoretical Population Biology*, 11(2):274–283, 1977.

[27] Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. 2019.

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

[29] Stan Lipovetsky. Numerical recipes: The art of scientific computing. *Technometrics*, 51(4):481, 2009.

[30] J. McFadden. The entropy of a point process. *Journal of the Society for Industrial & Applied Mathematics*, 13(4):988–994, 1965.

[31] Vitali D Milman. A new proof of a. dvoretzky’s theorem on cross-sections of convex bodies. *Funkcional. Anal. i Prilozen*, 5:28–37, 1971.

[32] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

[33] Alon Orlitsky, Narayana P Santhanam, and Junan Zhang. Universal compression of memoryless sources over unknown alphabets. *IEEE Transactions on Information Theory*, 50(7):1469–1481, 2004.

[34] F. Papangelou. On the entropy rate of stationary point processes and its discrete approximation. *Probability Theory and Related Fields*, 44(3):191–211, 1978.

[35] Jim Pitman et al. Combinatorial stochastic processes. Technical report, Technical Report 621, Dept. Statistics, UC Berkeley, 2002., 2002.

[36] Jim Pitman and Marc Yor. The two-parameter poisson-dirichlet distribution derived from a stable subprocessor. *The Annals of Probability*, pages 855–900, 1997.

[37] C Rasmussen and C Williams. Gaussian processes for machine learning. adaptive computation and machine learning, 2006.

[38] Narayana P Santhanam, Anand D Sarwate, and Jae Oh Woo. Redundancy of exchangeable estimators. *Entropy*, 16(10):5339–5357, 2014.

[39] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

[40] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[41] Changjian Shui, Fan Zhou, Christian Gagné, and Boyu Wang. Deep active learning: Unified and principled method for query and training. In *International Conference on Artificial Intelligence and Statistics*, pages 1308–1318. PMLR, 2020.

[42] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5972–5981, 2019.
Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.

Daniel A Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014.

Daniel A Spielman and Jae Oh Woo. A note on preconditioning by low-stretch spanning trees. *arXiv preprint arXiv:0903.2816*, 2009.

Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002.

Dimitris G Tzikas, Aristidis C Likas, and Nikolaos P Galatsanos. The variational approximation for bayesian inference. *IEEE Signal Processing Magazine*, 25(6):131–146, 2008.

JV White, Sam Steingold, and CG Fournelle. Performance metrics for group-detection algorithms. *Proceedings of Interface 2004*, 2004.

Dedy Rahman Wijaya, Riyanarto Sarno, and Enny Zulaika. Information quality ratio as a novel metric for mother wavelet selection. *Chemometrics and Intelligent Laboratory Systems*, 160:59–71, 2017.

Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. Practical machine learning tools and techniques. *Morgan Kaufmann*, page 578, 2005.

David H Wolpert and David R Wolf. Estimating functions of probability distributions from a finite set of samples. *Physical Review E*, 52(6):6841, 1995.
A Appendix

A.1 Derivation of Janossy density and more details about joint entropy

The Janossy density function resides in a combination of continuous and discrete domains \cite{10}. For the Janossy density of \((\Phi (x, \omega), Y(x, \omega))\) on \(\Delta^C \times [C]\), we may follow the classical approach:

\[
\mathbb{P}(P_1 \in [p_1 + dp_1], \ldots, P_C \in [p_c + dp_c], Y = i) \\
\approx \mathbb{P}(Y = i|P_1 = p_1, \ldots, P_C = p_C) \mathbb{P}(P_1 \in [p_1 + dp_1], \ldots, P_C \in [p_c + dp_c])
\]

where \(f(\cdot)\) is a density function of \(\Phi (x, \omega)\). So we may write the Janossy density of \((\Phi (x, \omega), Y(x, \omega))\) as follows:

\[
j(p_1, \ldots, pc; y = i) = p_i f(p_1, \ldots, pc).
\]

The joint entropy of \(\Phi (x, \omega)\) and \(Y(x, \omega)\) can be defined as

\[
\hat{H}(\Phi (x, \omega), Y(x, \omega)) = -\sum_{i=1}^{C} \int_{\Delta^C} j(p_1, \ldots, pc; y = i) \log j(p_1, \ldots, pc; y = i) \, dp_1 \cdots dp_c. \tag{4}
\]

We note that

\[
\int_{\Delta^C} p_i f(p_1, \ldots, pc) \, dp_1 \cdots dp_c = \int_{[0, 1]} p_i f(p_i) \, dp_i = EP_i.
\]

We may split the Janossy density into two pieces:

\[
\hat{H}(\Phi (x, \omega), Y(x, \omega)) = H(Y(x, \omega)) + \mathbb{E}_{Y} [h(\Phi (x, \omega) | Y)].
\]

A.2 Proof of Theorem 3.2

Assume that \(P \sim \text{Beta}(\alpha, \beta)\). Then we have a density function of \(P\) as

\[
f(x) = \frac{x^{\alpha - 1} (1 - x)^{\beta - 1}}{B(\alpha, \beta)}.
\]

Then it suffices to prove the following two lemmas. We note that similar calculations have been done in \cite{51}\cite{2}.

**Lemma A.1.** Assume that \(P \sim \text{Beta}(\alpha, \beta)\).

\[
\mathbb{E}[P \log P] = \frac{\mathcal{B}(\alpha + 1, \beta)}{\mathcal{B}(\alpha, \beta)} [\Psi(\alpha + 1) - \Psi(\alpha + \beta + 1)].
\]

**Proof.**

\[
\mathbb{E}[P \log P] = \frac{1}{\mathcal{B}(\alpha, \beta)} \int_{0}^{1} (p \log p) p^{\alpha - 1} (1 - p)^{\beta - 1} \, dp
\]

\[
= \frac{1}{\mathcal{B}(\alpha, \beta)} \int_{0}^{1} p^{\alpha} \log p (1 - p)^{\beta - 1} \, dp
\]

\[
= \frac{1}{\mathcal{B}(\alpha, \beta)} \int_{0}^{1} \frac{d}{d\alpha} p^{\alpha} (1 - p)^{\beta - 1} \, dp
\]

\[
= \frac{1}{\mathcal{B}(\alpha, \beta)} \frac{d}{d\alpha} \int_{0}^{1} p^{\alpha} (1 - p)^{\beta - 1} \, dp
\]

\[
= \frac{1}{\mathcal{B}(\alpha, \beta)} \frac{d}{d\alpha} \mathcal{B}(\alpha + 1, \beta)
\]

\[
= \frac{\mathcal{B}(\alpha + 1, \beta)}{\mathcal{B}(\alpha, \beta)} [\Psi(\alpha + 1) - \Psi(\alpha + \beta + 1)].
\]

Note that we may interchange the differentiation and the integral operator by applying Lebesgue’s dominated convergence theorem \cite{13}. The derivative of the Beta function can be found at \cite{4}. \[\square\]
Lemma A.2. Assume that $P \sim \text{Beta}(\alpha, \beta)$.

$$E[P \log(1 - P)] = \frac{B(\alpha + 1, \beta)}{B(\alpha, \beta)} \left[ \Psi(\beta) - \Psi(\alpha + \beta + 1) \right].$$

Proof.

$$E[P \log(1 - P)] = \frac{1}{B(\alpha, \beta)} \int_0^1 (p \log(1 - p)) \rho^{\alpha - 1} (1 - p)^{\beta - 1} dp$$

$$= \frac{1}{B(\alpha, \beta)} \int_0^1 p^{\alpha} \left(\log(1 - p)\right) (1 - p)^{\beta - 1} dp$$

$$= \frac{1}{B(\alpha, \beta)} \frac{d}{d\beta} \int_0^1 p^{\alpha} (1 - p)^{\beta - 1} dp$$

$$= \frac{1}{B(\alpha, \beta)} \frac{d}{d\beta} B(\alpha + 1, \beta)$$

$$= \frac{B(\alpha + 1, \beta)}{B(\alpha, \beta)} \left[ \Psi(\beta) - \Psi(\alpha + \beta + 1) \right].$$

Then the Theorem 3.2 follows by rearranging terms after applying Lemma A.1 and Lemma A.2.

A.3 Beta approximated upper joint entropy in BatchBABA

To define the BatchBABA, we may find an upper bound of the joint entropy as follows:

$$S_1(\Phi(x_1), \cdots, \Phi(x_n), Y(x_1), \cdots, Y(x_n))$$

$$= H(Y(x_1), \cdots, Y(x_n), Y) + E_Y [h(\Phi(x_1), \cdots, \Phi(x_n)|Y(x_1), \cdots, Y(x_n))$$

$$\leq H(Y(x_1), \cdots, Y(x_n)) + \sum_i E_Y [h(\Phi(x_i)|Y(x_i))]$$

$$\leq H(Y(x_1), \cdots, Y(x_n)) + \sum_i \left[ S_1^{\beta}(\Phi(x_i), Y(x_i)) - H(Y(x_i)) \right]$$

$$\approx H(Y(x_1), \cdots, Y(x_n)) + \sum_i \left[ S_1^{\beta}(\Phi(x_i), Y(x_i)) - H(Y(x_i)) \right] =: S_1^{\beta}(x_1, \cdots, x_n),$$

where the first inequality comes from the property that conditioning reduces the entropy [9], and the second inequality comes from the identity [3].

A.4 Short early stopping tolerance level

In this section, we present an experimental result of EMNIST dataset with smaller early stopping to illustrate the performance difference of BALD. In this experiment, we set the early stopping tolerance level to be 3, and the model was trained from scratch, i.e., starting from an empty initial training set. In contrast to the result in Figure 5 and Table 3, the performance of BALD shows the second worst. This experimental result implies that BALD is very sensitive to the convergence of model parameters. Nevertheless, we also note that BABA still outperforms other methods.

| Method     | Relative Accuracy to Random | Relative Negative Log-Likelihood to Random |
|------------|----------------------------|-------------------------------------------|
| Random     | 0 ± 3.79%                  | 0 ± 8.34%                                 |
| Variation Ratios | -5.06 ± 4.96%             | 17.35 ± 9.21%                             |
| Predictive Entropy | -6.22 ± 6.54%             | 17.21 ± 9.92%                             |
| Mean STD   | -1.09 ± 5.97%              | 11.96 ± 12.39%                            |
| BALD       | -5.76 ± 4.81%              | 12.71 ± 10.23                             |
| BABA (ours)| 6.89 ± 4.81%               | -3.39 ± 9.11%                             |

Table 5: EMNIST dataset with acquisition size 1. We set the early stopping condition to be 3 epochs.
A.5 Ablation study

One might be interested in the standalone effect of the Beta approximated joint entropy as a denominator. We include the experimental results by using the following acquisition measure excluding BALD:

$$\text{InvBeEnt}[x] := \frac{1}{|\mathcal{H}_{\text{Beta}}(\Phi(x, \omega), Y(x, \omega))|}.$$ 

We use the same experimental setting of the experiment used in Section 5.2. As the results indicate, the normalized mutual information quantity is a better and meaningful measure to capture the uncertainty. Each standalone acquisition, BALD or InvBeEnt, fails to perform better than random.

A.6 Explanations on reporting metrics

In each table, we report the mean and the standard deviation of the relative accuracy or the relative log-likelihood with respect to the random method results. We describe the calculation details. Let $N$ be the number of repeated experiments. For each accumulated dataset size $t$ and the random acquisition function $R$, denote by $R_i(t)$ the accuracy for the random acquisition at the size $t$ for the $i$-th experiment. Then we compute the geometric mean of the series $R_i(t)$ for each $t$. This is our baseline accuracy to compare with others.

$$\text{GM}_R(t) := \exp \left( \frac{1}{N} \sum_i \log R_i(t) \right).$$
For each observed experiment series $S_i(t)$, we compute the relative accuracy as follows:

$$\text{RelAcc}_{S_i}(t) := \log \left( \frac{S_i(t)}{\text{GM}_R(t)} \right).$$

Then for each method $S \in \{\text{Rand}, \text{VarRatio}, \text{PredEntropy}, \text{MeanSTD}, \text{BALD}, \text{BABA}, \text{BatchBALD}, \text{BatchBABA}\}$, we report the total mean and standard deviation of relative accuracies by writing $\text{Mean}(S) \pm \text{SD}(S)$ obtained from

$$\text{Mean}(S) := \frac{1}{NT} \sum_{i,t} \text{RelAcc}_{S_i}(t),$$

$$\text{SD}(S) := \sqrt{\frac{1}{NT} \sum_{i,t} (\text{RelAcc}_{S_i}(t))^2 - (\text{Mean}(S))^2},$$

where $T$ is the number of all observed cumulative acquisition data-size points. Similarly, we do the same calculations for the log-likelihood.