BatchBALD

Efficient and Diverse Batch Acquisition for Deep Bayesian Active Learning

NeurIPS 2019

Andreas Kirsch*, Joost van Amersfoort*, Yarin Gal
OATML - University of Oxford

Email: {andreas.kirsch, joost.van.amersfoort, yarin}@cs.ox.ac.uk
Introduction to Active Learning

- A key problem in Deep Learning is data efficiency
- It is expensive to a-priori obtain a large labelled dataset
- In Active Learning, we instead iteratively acquire labels for only the most interesting datapoints

Goal: achieve particular accuracy (e.g. 95%) with fewest number of labelled data points
Active Learning

Train model on \textit{labelled} training set

Evaluate acquisition function on \textit{unlabelled} pool set using the model

Experts label pool points with highest acquisition score

Add newly \textit{labelled} points to training set

Goal: 95% accuracy

\[ \mathcal{D}_{\text{train}} \text{(labelled)} \]

\[ \mathcal{D}_{\text{test}} \text{(labelled)} \]

\[ \mathcal{D}_{\text{pool}} \text{(unlabelled)} \]
Active Learning

- Train model on **labelled** training set
- Evaluate acquisition function on **unlabelled** pool set using the model
- Experts **label** pool points with highest acquisition score
- Add newly **labelled** points to training set

Goal: 95% accuracy

\[ \mathcal{D}_{train} \quad \text{(labelled)} \]
\[ \mathcal{D}_{test} \quad \text{(labelled)} \]
\[ \mathcal{D}_{pool} \quad \text{(unlabelled)} \]
Acquisition Function

Instead of acquiring a random data point, we want to pick an informative point.

We acquire points using the acquisition function $a$, which takes as input data points and the model parameters:

$$x^* = \arg\max_{x \in \mathcal{D}_{\text{pool}}} a(x, p(\omega | \mathcal{D}_{\text{train}}))$$

Distribution over model parameters
Active Learning

Goal: 95% accuracy

1. Train model on labelled training set
2. Evaluate acquisition function on unlabelled pool set using the model
3. Experts label pool points with highest acquisition score
4. Add newly labelled points to training set

$D_{\text{train}}$ (labelled)

$D_{\text{test}}$ (labelled)

$D_{\text{pool}}$ (unlabelled)
Uncertainty in Deterministic Networks

We have the following two approaches:

1. Variation ratio (the max of the softmax output)
2. Entropy of the softmax output

**Problem:** uncertainty isn’t reliable. Standard deep models extrapolate arbitrarily!
Active Learning - Bayesian Approaches

**Alternative:** use a Bayesian Neural Network

- Dropout as inference method
- Scales well and is easy to implement and train
- Enables more reliable uncertainty quantification:
  - Entropy of *marginalised* softmax output
  - Mutual information between parameters and the true label (BALD)
BALD: “Bayesian Active Learning by Disagreement”

\[
\mathbb{I}(y ; \omega | x, \mathcal{D}_{\text{train}}) = \mathbb{H}(y | x, \mathcal{D}_{\text{train}}) - \mathbb{E}_{p(\omega | \mathcal{D}_{\text{train}})} \left[ \mathbb{H}(y | x, \omega, \mathcal{D}_{\text{train}}) \right].
\]

Intuitively:
1. First term captures general uncertainty of model
2. Second term captures the uncertainty of a given draw of the model parameters

Score is high when model is uncertain (high entropy) and per sample certain (expectation of sample entropy high)
Problems in Practice

- Re-training a deep model is computationally expensive
- Mobilising an expert is time consuming

→ In practice: acquire a batch of data by selecting top \( k \)

\[
\{x_1^*, \ldots, x_b^*\} = \arg\max_{\{x_1, \ldots, x_b\} \subseteq \mathcal{D}_{\text{pool}}} a (\{x_1, \ldots, x_b\}, p(\omega | \mathcal{D}_{\text{train}})).
\]
Active Learning

Train model on labelled training set

Evaluate acquisition function on unlabelled pool set using the model

Experts label pool points with highest acquisition score

Add newly labelled points to training set

Goal: 95% accuracy

\( D_{\text{train}} \)
\( D_{\text{test}} \)
\( D_{\text{pool}} \)
Naively acquiring the top k scoring points can lead to redundant acquisitions!
BALD vs BatchBALD

Lack of Diversity

Performs worse than random on Repeated MNIST
BatchBALD I - Introduction

→ Score batches of points **jointly**

Joint entropy: Hard to compute

\[ a_{\text{BatchBALD}} (\{x_1, \ldots, x_n\}, p(\omega|D_{\text{train}})) = \mathbb{H}(y_1, \ldots, y_n | x_1, \ldots, x_b, D_{\text{train}}) - \mathbb{E}_{p(\omega|D_{\text{train}})} [\mathbb{H}(y_1, \ldots, y_n | x_1, \ldots, x_b, \omega)] . \]

Expectation over entropy: easy to compute
BALD

Dark areas are counted double

BatchBALD

No double-counting with BatchBALD

\[ \sum_{i=1}^{b} \mathbb{I}(y_i; \omega | x_i, D_{\text{train}}) \]

\[ \mathbb{I}(y_1, \ldots, y_b; \omega | x_1, \ldots, x_b, D_{\text{train}}) \]
We derive an MC estimator for the first term (implicitly conditioned on x):

\[ \mathbb{H}(y_1, \ldots, y_n) \approx - \sum_{\hat{y}_{1:n}} \left( \frac{1}{k} \sum_{j=1}^{k} p(\hat{y}_{1:n} | \hat{\omega}_j) \right) \log \left( \frac{1}{k} \sum_{j=1}^{k} p(\hat{y}_{1:n} | \hat{\omega}_j) \right). \]

- k - the number of samples of the parameter distribution
- \( \hat{y} \) - a configuration of a batch of labels
- Size of \( \hat{y} \) grows exponentially with \( |\mathcal{D}_{pool}|^b \)
BatchBALD III - Greedy Approximation

To avoid combinatorial explosion: we grow the acquisition batch one by one.

\[
\frac{1}{k} \sum_{j=1}^{k} p(\hat{y}_{1:n} | \hat{\omega}_j) = \frac{1}{k} \sum_{j=1}^{k} p(\hat{y}_{1:n-1} | \hat{\omega}_j) p(\hat{y}_n | \hat{\omega}_j) = \left( \frac{1}{k} \hat{P}_{1:n-1} \hat{P}_n^T \right)_{\hat{y}_{1:n-1}, \hat{y}_n}
\]
Greedy Approximation and Submodularity

- We can compute the greedy joint entropy exact for the first 4 points, afterwards we use the Monte Carlo estimator.
- We prove in the paper that the greedy approximation is a submodular function and therefore its error is bounded by $1 - 1/e$. 
Consistent dropout

There is a lot of variance in dropout uncertainty estimates. By using the same dropout masks across the unlabelled set we reduce variance:
Results MNIST
Results EMNIST
Diversity
Results CINIC-10 (CIFAR+ImageNet)
Conclusion

BatchBALD improves over BALD:

1. Performs as well with acquisition size 10 as BALD with 1
2. Much lower total experiment time than BALD
3. Performs well on datasets with repetition
4. Performs well when the number of classes is high
Future Work

- Estimating joint entropy with dropout samples is noisy
- Training a deep model on a small dataset with dropout is difficult
  - Uncertainty of deterministic ResNet is (surprisingly) good in practice
- In Active Learning the labelled set does not follow the original data distribution
  - This leads to difficulties with imbalanced datasets
  - Random acquisition doesn’t have this problem
Thanks!

Andreas Kirsch*  Joost van Amersfoort*  Yarin Gal

Email: {andreas.kirsch, joost.van.amersfoort, yarin}@cs.ox.ac.uk
