Deep Active Learning with Adaptive Acquisition

Manuel Haußmann[1] , **Fred Hamprecht**[1] and **Melih Kandemir**[2]
[1]HCI/IWR, Heidelberg University, Germany
[2]Bosch Center for Artificial Intelligence, Renningen, Germany
manuel.haussmann@iwr.uni-heidelberg.de

Abstract

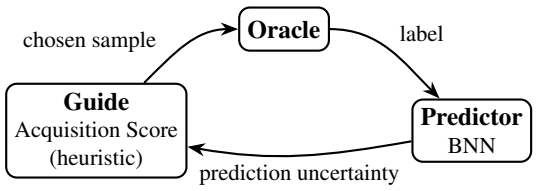
Model selection is treated as a standard performance boosting step in many machine learning applications. Once all other properties of a learning problem are fixed, the model is selected by grid search on a held-out validation set. This is strictly inapplicable to active learning. Within the standardized workflow, the acquisition function is chosen among available heuristics a priori, and its success is observed only after the labeling budget is already exhausted. More importantly, none of the earlier studies report a unique consistently successful acquisition heuristic to the extent to stand out as the unique best choice. We present a method to break this vicious circle by defining the acquisition function as a learning predictor and training it by reinforcement feedback collected from each labeling round. As active learning is a scarce data regime, we bootstrap from a well-known heuristic that filters the bulk of data points on which all heuristics would agree, and learn a policy to warp the top portion of this ranking in the most beneficial way for the character of a specific data distribution. Our system consists of a Bayesian neural net, the predictor, a bootstrap acquisition function, a probabilistic state definition, and another Bayesian policy network that can effectively incorporate this input distribution. We observe on three benchmark data sets that our method always manages to either invent a new superior acquisition function or to adapt itself to the a priori unknown best performing heuristic for each specific data set.

1 Introduction

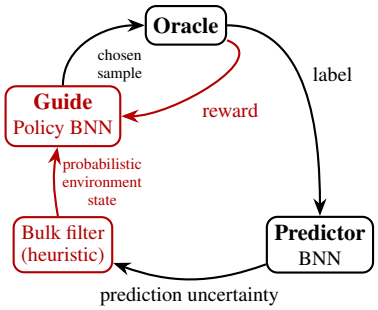
The active learning setup consists of a base predictor that chooses the order in which the data points are supposed to be labeled using an acquisition function. Contrary to the *tabula rasa* ansatz of the present deep learning age, the state of the art in active learning has maintained to use hand-designed acquisition functions to rank the unlabeled sample space. Different studies observed different acquisition functions to perform optimally for specific applications after evaluating various choices. The critical fact is that active learning is meant to address applications where data labeling is extremely costly and it is not possible to know the ideal acquisition function for a given application a priori. Once an acquisition function is chosen and active learning has been performed based on it, the labeling budget is already exhausted, leaving no possibility for another try with an alternative acquisition function. This limitation can only be circumvented by adapting the acquisition function to data during the active learning process, getting feedback from the impact of the previous labeling rounds on model fit. For real-world scenarios to which active learning is applicable, learning also the acquisition function is not only an option driven solely by practical concerns such as avoidance of handcrafting effort, but also an absolute necessity stemming from epistemic limitations.

The acquisition functions in active learning are surrogate models that map a data point to a value that encodes the expected contribution of observing its label to model fit. The founding assumption of the active learning setup is that evaluating the acquisition score of a data point is substantially cheaper than acquiring its ground-truth label. Hence, the acquisition functions are expected to be both computationally cheap and maximally accurate in detecting most information-rich regions of the sample space. These competing goals are typically addressed by information-theoretic heuristics. Possibly the most frequently used acquisition heuristic is *Maximum Entropy Sampling*, which assigns the highest score to the data point for which the predictor reports highest entropy (i.e. uncertainty). This criterion builds on the assumption that the most valuable data point is the one the model is maximally unfamiliar about. While being maximally intuitive, this method remains agnostic to exploiting knowledge from the current model fit about how much the new label can impact the model uncertainty. Another heuristic with comparable reception, *Bayesian Active Learning by Disagreement (BALD)* [Houlsby *et al.*, 2012], benefits from this additional information by maximizing the mutual information between the predictor output and the model parameters. A second major vein of research approaches the active learning problem from a geometric instead of an uncertainty based perspective, e.g. via selection of a core-set [Sener and Savarese, 2018].

None of the aforementioned heuristics has a theoretical superiority that is sufficient to rule out all other options. Maxent strives only to access unexplored data regions. BALD performs the same by also taking into account the expected effect



(a) Active learning pipeline.



(b) Our method.

Figure 1: The proposed pipeline. The standard active learning pipeline is summarized as the interplay between three parts. *(a)* An *oracle* provides a set of labeled data for a *predictor* (here a BNN) to learn on. It in turn provides predictive uncertainties to the *guide*, a usually fixed, hard-coded acquisition function, which in turn communicates to the oracle which points to label next, restarting the cycle. *(b)* We replace the fixed acquisition function with a policy BNN that learns with a probabilistic state and reinforcement feedback from the oracle how to optimally choose the next points (new parts in red). It is thus able to adapt itself flexibly to the data set at hand.

of the newly explored label on the uncertainty of the model parameters. While some studies argue in favor of BALD due to this additional information it enjoys [Srinivas *et al.*, 2012], others prefer to avoid this noisy estimate drawn from an under-trained model [Qiu *et al.*, 2017]. This paper presents a data-driven method that alleviates the consequences of the unsolved acquisition function selection problem. As prediction uncertainty is an essential input to acquisition heuristics, we choose a deep Bayesian Neural Net (BNN) as our base predictor. In order to acquire high-quality estimates of prediction uncertainty with an acceptable computational cost, we devise a deterministic approximation scheme that can both effectively train a deep BNN and calculate its posterior predictive density following a chain of closed-form operations. Next, we incorporate all the probabilistic information provided by the BNN predictions into a novel state design, which brings about another full-scale probability distribution. This distribution is then fed into a probabilistic policy network, which is trained by reinforcement feedback collected from every labeling round in order to inform the system about the success of its current acquisition function. This feedback fine-tunes the acquisition function, bringing about improved performance in the subsequent labeling rounds. Figure 1 depicts the workflow of our method.

We evaluate our method on three benchmark vision data sets from different domains and complexities: MNIST for images of handwritten digits, FashionMNIST for greyscale images of clothes, and CIFAR-10 for colored natural images. We observe in our experiments that the policy net is capable of inventing an acquisition function that outperforms all the handcrafted alternatives if the data distribution permits. In the rest of the cases, the policy net converges to the best-performing handcrafted choice, which varies across data sets and is unknown prior to the active learning experiment.

2 The Model

Our method consists of two major components: a predictor and a policy net guiding the predictor by learning a data set specific acquisition function. As the predictor, described in Section 2.1 we use a BNN, whose posterior predictive density we use to distill the system state. The policy net, another BNN, takes this state as input to decide which data points to request labels for next.[1] We describe this second part of the pipeline in Section 2.2. Since we introduce a reinforcement learning based method for active learning, we refer to it as *Reinforced Active Learning (RAL)*.

2.1 Predictor: Bayesian Neural Network

Let $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)_{n=1}^N\}$ be a data set consisting of N tuples of feature vectors $\mathbf{x}_n \in \mathbb{R}^m$ and labels $\mathbf{y}_n \in \{0,1\}^C$ for a C dimensional binary output label. Parameterizing an arbitrary neural network $f(\cdot)$ with parameters $\mathbf{w}$ following some prior $p(\mathbf{w})$, we assume the following probabilistic model

$$\mathbf{w} \sim p(\mathbf{w}), \quad \mathbf{y}|\mathbf{x}, \mathbf{w} \sim \prod_c^C \mathcal{B}er\big(y_c|\Phi(f_c(\mathbf{x};\mathbf{w}))\big), \qquad (1)$$

where f_c is the cth output channel of the net, $\Phi(u) = \int_{-\infty}^{u} \mathcal{N}(x|0,1)dx$, and $\mathcal{B}er(\cdot|\cdot)$ is a Bernoulli distribution. The calculation of the posterior predictive

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) p(\mathbf{w}|\mathcal{D}) d\mathbf{w} \qquad (2)$$

involves the calculation of the posterior distribution on the latent variables, which can be accomplished by Bayes rule

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathbf{Y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w})}{\int p(\mathbf{Y}|\mathbf{X}, \mathbf{w}) p(\mathbf{w}) d\mathbf{w}}, \qquad (3)$$

for $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ and $\mathbf{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_N\}$. As this is intractable in general we require approximate inference techniques. We aim for high-quality prediction uncertainties. A sampling based approach is not practical for vision-scale applications where neural nets with a large parameter count are

[1] For illustrative purposes we rely on a Central Limit Theorem approach to efficiently marginalize the weights of these BNNs. In general any approach to training a BNN which provides trustworthy predictive uncertainties could be used.

being used. Instead we use variational inference (VI). In order to keep the calculations maximally tractable while benefiting from stochasticity $\mathbf{w}$, we formulate a normal mean-field variational posterior (which could be generalized):

$$q_\theta(\mathbf{w}) = \prod_i \mathcal{N}(w_i|\mu_i, \sigma_i^2), \tag{4}$$

where the tuple (μ_i, σ_i^2) represents the variational parameter set for weight w_i of the network $f(\cdot)$ and $\theta = \{(\mu_i, \sigma_i^2)_i\}$. VI approximates the true intractable posterior by optimizing θ to minimize the Kullback-Leibler (KL) divergence between $q_\theta(\mathbf{w})$ and $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$, which boils down to minimizing the negative evidence lower bound

$$\mathcal{L}_{\text{class}}(\theta; \mathcal{D}) = -\sum_{n=1}^{N} \mathbb{E}_{q_\theta(\mathbf{w})}\big[\log p(\mathbf{y}_n|f(\mathbf{x}_n; \mathbf{w}))\big] + \mathrm{KL}\big(q_\theta(\mathbf{w})||p(\mathbf{w})\big). \tag{5}$$

In this formula, the first term on the r.h.s. maximizes the data fit (i.e. minimizes the reconstruction loss), and the second term penalizes divergence of the posterior from the prior, inducing the Occam's razor principle to the preferred solution.

The modeler has control on the model families of both $q_\theta(\mathbf{w})$ and $p(\mathbf{w})$. Hence, choosing the prior $p(\mathbf{w})$ suitably to the normally distributed $q_\theta(\mathbf{w})$ assures an analytically tractable solution for the $\mathrm{KL}(\cdot||\cdot)$ term.[2] However, the data fit term involves a nonlinear neural net, which introduces difficulties for keeping the calculation of the expectations tractable. A major issue is that we need to differentiate this term with respect to the variational parameters θ, which appear in the density $q_\theta(\mathbf{w})$ with respect to which the expectation is taken. This problem is overcome by the reparameterization trick [Kingma and Welling, 2014], which reformulates $q_\theta(\mathbf{w})$ as a sampling step from a parameter-free distribution and a deterministic variable transformation.

$$\mathcal{L}_{\text{class}}(\theta; \mathcal{D}) = -\sum_{n=1}^{N} \mathbb{E}_{p(\varepsilon)}\big[\log p(\mathbf{y}_n|f(\mathbf{x}_n; \mathbf{w} = \mu + \sigma\varepsilon))\big] + \mathrm{KL}\big(q(\mathbf{w})||p(\mathbf{w})\big), \tag{6}$$

where the variational parameters θ now appear only inside the expectation term, and we could take the gradient of the loss with respect to them and approximate integral in the expectation by Monte Carlo sampling. Although this will provide an unbiased estimator of the exact gradient, due to distorting the global variables of a highly-parameterized system, this estimator will have prohibitively high variance. The remedy is to postpone sampling one step further.

Let the pre-activation and feature map of jth neuron of layer l for data point n be b_{njl} and h_{njl}, respectively. We then have

$$w_{ijl} \sim \mathcal{N}(w_{ijl}|\mu_{ijl}, \sigma_{ijl}^2), \quad b_{njl} = \sum_{i=1}^{I_{l-1}} w_{ijl} h_{nil-1}, \tag{7}$$

as a repeating operation at every layer transition within a BNN.[3] As h_{nil-1} is the sampling output of layer $l-1$, it is a deterministic input to layer l. Consequently, b_{njl} is a weighted linear sum of I_{l-1} independent normal random variables, which is another normal random variable with

$$b_{njl} \sim \mathcal{N}\left(b_{njl} \middle| \sum_{i=1}^{I_{l-1}} \mu_{ijl} h_{nil-1}, \sum_{i=1}^{I_{l-1}} \sigma_{ijl}^2 h_{nil-1}^2\right). \tag{8}$$

We now take separate samples for local variables, further reducing the estimator variance stemming from the Monte Carlo integration. This is known as Variational Dropout [Kingma *et al.*, 2015], as the process performed for the expected log-likelihood term is equivalent to Gaussian dropout with rate $\sigma_{ijl}^2/\mu_{ijl}^2$ for weight w_{ijl}.

Fast Dropout and the CLT Trick

Fast Dropout [Wang and Manning, 2013] has been introduced as a technique to perform Gaussian dropout without taking samples. The technique builds on the observation that b_{njl} is essentially a random variable that consists of a sheer sum of a provisionally large number of other random variables. This is a direct call to the Central Limit Theorem (CLT) that transforms the eventual distribution into a normal density, which can be trivially modeled by matching the first two moments

$$p(b_{njl}) \approx \mathcal{N}(b_{njl}|\phi_{njl}, \lambda_{njl}^2), \quad \text{where}$$

$$\phi_{njl} = \mathbb{E}\left[\sum_{i=1}^{I_{l-1}} w_{ijl} h_{nil-1}\right] = \sum_{i=1}^{I_{l-1}} \mathbb{E}[w_{ijl}]\mathbb{E}[h_{nil-1}],$$

$$\lambda_{njl}^2 = \mathrm{var}\left[\sum_{i=1}^{I_{l-1}} w_{ijl} h_{nil-1}\right] = \sum_{i=1}^{I_{l-1}} \mathrm{var}[h_{nil-1}]\mathbb{E}[w_{ijl}]^2 + \mathrm{var}[w_{ijl}]\mathbb{E}[h_{nil-1}^2].$$

Here, $\mathbb{E}[w_{ijl}] = \mu_{ijl}$ and $\mathrm{var}[w_{ijl}] = \sigma_{ijl}^2$, as determined in Equation 4. We also require the first two moments over of the $h_{nil-1} = r(b_{nil-1})$, for which it suffices to solve

$$\mathbb{E}[h_{nil-1}] = \int r(b_{nil-1}) p(b_{nil-1}) db_{nil-1},$$

$$\mathbb{E}[h_{nil-1}^2] = \int r(b_{nil-1})^2 p(b_{nil-1}) db_{nil-1}.$$

These two are analytically tractable for standard choices of activation functions, such as when $r(\cdot)$ is the ReLU activation and $p(b_{nil-1})$ is a normal distribution [Frey and Hinton, 1999]. Note that b_{nil-1} is either the linear activation of the input layer, a weighted sum of normals, hence another normal, or a hidden layer, which will then similarly follow CLT and therefore already be approximated as a normal. Hence, the above pattern repeats throughout the entire network, allowing a tight closed-form approximation of the analytical solution. Below, we refer to this method as the *CLT trick*.

Closed-Form Uncertainty Estimation with BNNs

Fast Dropout uses the aforementioned CLT trick only for implementing dropout. Here we extend the same method to perform variational inference by minimizing a deterministic loss, i.e. avoiding Monte Carlo sampling altogether. Even

[2] We use $p(w_i) = \mathcal{N}(w_i|0, \alpha^{-1})$ with a fixed precision α.

[3] The same line of reasoning directly applies to convolutional layers where the sum on b_{njl} is performed in a sliding window fashion.

though the CLT trick has previously been used mainly for expectation propagation, its direct application to variational inference has not been investigated prior to our work. Furthermore, the state of the art in deep active learning relies on test-time dropout [Gal *et al.*, 2017], which is computationally prohibitive. Speeding up this process requires parallel computing on the *end-product*, hence reflects additional costs on the user of the model not addressable at the production time. A thus far overlooked aspect of the CLT trick is that it also allows closed-form calculation of the posterior predictive density. Once training is over, we get a factorized surrogate for our posterior. Plugging this surrogate into the predictive density formula, for a new observation $\mathbf{x}^*$ we get

$$\begin{aligned} p(y_c^*|\mathbf{x}^*, \mathcal{D}) &\approx \int \mathcal{B}er\big(y_c^*|\Phi(f_c(\mathbf{x}^*;\mathbf{w}))\big) q_\theta(\mathbf{w}) d\mathbf{w} \\ &\approx \int \mathcal{B}er\big(y_c^*|\Phi(f_c^*)\big) \mathcal{N}\Big(f_c^*|g_c^L(\mathbf{x}^*), h_c^L(\mathbf{x}^*)\Big) df_c^* \\ &= \mathcal{B}er\left(y_c^*\middle|\Phi\left(\frac{g_c^L(\mathbf{x}^*)}{\sqrt{h_c^L(\mathbf{x}^*)+1}}\right)\right), \end{aligned} \tag{9}$$

where the functions $g_c^L(\mathbf{x}^*)$ and $h_c^L(\mathbf{x}^*)$ encode the cumulative map from the input layer to the moments of the top-most layer after repetitive application of the CLT trick across the layers.[4] With $p(y_c^*|\mathbf{x}^*, \mathcal{D})$ is tightly approximated by an analytical calculation of a known distributional form, its high-order moments are readily available for downstream tasks, being active learning in our case.

2.2 Guide: The Policy Net

As opposed to the standard active learning pipeline, our method is capable of adapting its acquisition scheme to the characteristics of individual data distributions. Differently from earlier work on data-driven label acquisition, our method can perform the adaptation *on the fly*, i.e. while the active learning labeling rounds take place. This adaptiveness is achieved within a reinforcement learning framework, where a policy net is trained by rewards observed from the environment. We denote the collection of unlabeled points by $\mathcal{D}_u$ and the labeled ones by $\mathcal{D}_l$. The variables N_u and N_l denote the number of data points in each respective case.

State. In active learning, the label acquisition process takes place on the entire unlabeled sample set. However, a feasible reinforcement learning setup necessitates a condensed state representation. To this end, we first rank the unlabeled sample set by an information-theoretic heuristic, namely the maximum entropy criterion. As such heuristics assign similar scores to samples with similar character, consecutive samples in ranking inevitably have high correlation. In order to break the trend and enhance diversity, we follow the ranking from the top and pick up every Kth sample until we collect M samples $\{\mathbf{x}_1, \cdots, \mathbf{x}_M\}$. We adopt the related term from the Markov Chain Monte Carlo sampling literature and refer to this process as *thinning*. Feeding these samples into our predictor BNN (Equation 9), we attain a posterior predictive density estimate for each and distill the state of the unlabeled sample space in the following distributional form:

$$S \sim \prod_{c=1}^{C} \prod_{m=1}^{M} \mathcal{N}\left(g_c^L(\mathbf{x}_m), h_c^L(\mathbf{x}_m)\right), \tag{10}$$

where $g_c^L(\cdot)$ and $h_c^L(\cdot)$ are mean and variance of the activation an output neuron, calculated as in Equation 9.

Action. At each labeling round, a number of data points are sampled from the set $\{\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_M}\}$ according to the probability masses assigned on them by the present policy.

Reward. The straight-forward reward would be the performance of the updated predictor on a separate validation set. This, however, clashes with the constraint imposed on us by the active learning scenario. The motivating assumption is that labels are valuable and scarce, so it is not feasible to construct a separate labeled validation set large enough to get a good guess of the desired test set performance for the policy net to calculate rewards. In our preliminary experiments, we have observed that merging the validation set with the existing training set and performing active learning on the remaining sample set consistently provides a steeper learning curve than keeping a validation set for reward calculation. Hence, we abandon this option altogether. Instead, we propose a novel reward signal

$$R = R_{\text{improv}} + R_{\text{div}}, \tag{11}$$

consisting of the two components detailed below. The first component R_{improv} assesses the improvement in data fit of the chosen point once it has been labeled. From a Bayesian perspective, a principled measure of model fit is the marginal likelihood. For a newly labeled pair $(\mathbf{x}, \mathbf{y})$ the reward is

$$\begin{aligned} R_{\text{improv}} = &\prod_{c=1}^{C} \int \mathcal{B}er\big(y_c|\Phi(f_c(\mathbf{x};\mathbf{w}))\big) q_{\text{new}}(\mathbf{w}) d\theta \\ &- \prod_{c=1}^{C} \int \mathcal{B}er\big(y_c|\Phi(f_c(\mathbf{x};\mathbf{w}))\big) q_{\text{old}}(\mathbf{w}) d\theta, \end{aligned} \tag{12}$$

where $q_{\text{old}}(\cdot), q_{\text{new}}(\cdot)$ are our respective variational posteriors before and after training with the new point. The second component, R_{div}, encourages diversity across the chosen labels throughout the whole labeling round:

$$R_{\text{div}} = \frac{\#\text{unique labels requested}}{\#\text{label requests in this episode}}. \tag{13}$$

Policy net. The policy net $\pi(\cdot)$ is a second BNN parameterized by $\phi \sim p(\phi)$. Compared to the classifier, taking deterministic data points as input, the policy net takes the state S, which follows a $C \cdot M$-dimensional normal distribution. Inputing such a stochastic input into our deterministic inference scheme is straightforward by using the first two moments of the state during the first moment-matching round. The output of the policy net, in turn, parameterizes an M dimensional categorical distribution over possible actions. In order to benefit fully from the BNN character of the policy and to marginalize over the ϕ we again follow the approach we use

[4] Once $g_c^L(\mathbf{x}^*)$ and $h_c^L(\mathbf{x}^*)$ are computed one could also choose a categorical likelihood and approximate the integral via sampling.

Algorithm 1: The RAL training procedure

Input: $\mathcal{D} = \{\mathcal{D}_u, \mathcal{D}_l\}$, labeling budget, state size M, policy π_ϕ, net f_θ, episode length T

```
// Train an initial net
```
train f_θ on $\mathcal{D}_l$ as described in Section 2.1
while *budget available* **do**
```
    // The labeling episode
```
    generate state distribution S_0 from $\mathcal{D}_u$
    for $t \in 1, ..., T$ **do**
        sample A_t via $\pi_\phi(S_{t-1})$
        $\mathcal{D}_l \leftarrow \mathcal{D}_l \cup \{\text{data point selected via } A_t\}$
        $\mathcal{D}_u \leftarrow \mathcal{D}_u \backslash \{\text{data point selected via } A_t\}$
        generate state distribution S_t from $\mathcal{D}_u$
    end
```
    // Update the agent and net
```
    train f_θ on $\mathcal{D}_l$
    compute rewards R_{div}, R_{improv} and returns G_t
    update ϕ via gradient descent on
    $G_t \nabla_\phi \left(\log \pi_\phi(A_t|S_t) + \mathrm{KL}\big(q(\phi)||p(\phi)\big)\right)$
end

for the classifier propagating the moments and first compute M binary probabilities for taking action $\tilde{a}_m$ at time point t

$$p(\tilde{a}_m^t) = \mathbb{E}_{q(\phi)}\left[\mathcal{B}er\big(\tilde{a}_m^t|\Phi(\pi_m(S_t;\phi))\big)\right], \qquad (14)$$

and finally normalize them to choose the action A_t via

$$A_t \sim \mathcal{C}at(\mathbf{a}^t), \qquad \text{where } a_m^t = \tilde{a}_m^t \Big/ \textstyle\sum_j \tilde{a}_j^t,$$

and $\mathcal{C}at(\cdot)$ is a Categorical distribution.

Algorithm. We adopt the episodic version of the standard REINFORCE algorithm [Williams, 1992] to train our policy net. We use a moving average over all the past rewards as the baseline. A labeling episode consists of choosing a sequence of points to be labeled (with a discount factor of $\gamma = 0.95$) after which the BNN is retrained and the policy net takes one update step. We parameterize the policy $\pi_\phi(\cdot|S_t)$ itself by a neural network with parameters ϕ. Our method iterates between labeling episodes, training the policy net π, and training the BNN f until the labeling budget is exhausted. The pseudocode of our method is given in Algorithm 1.

3 Experiments

As RAL is the first method to adapt its acquisition function while active learning takes place, its natural reference model is the standard active learning setup with a fixed acquisition heuristic.[5] We choose the most established two information-theoretic heuristics: Maximum Entropy Sampling (Maxent) and BALD. Gal *et.al.* [2017] already demonstrated how BNNs (in their case with fixed Bernoulli dropout) provide an improved signal to acquisition functions that can be used to improve upon using predictive uncertainty from deterministic nets. We will use our own BNN formulation for both RAL as well as these baseline acquisition functions, to give them access to the same closed-form predictive uncertainty and to to ensure maximal comparability between our model and the baselines by having an absolutely identical architecture and training procedure for all methods. For Maxent one selects the point that maximizes the predictive entropy,

$$\underset{(x,y)\in\mathcal{D}_u}{\arg\max}\ \mathrm{H}[p(y|x,\mathcal{D}_l)], \qquad \text{where}$$

$$\mathrm{H}[p(y|x,\mathcal{D}_l)] = -\sum_{c=1}^{C} p(y=c|x,\mathcal{D}_l)\log p(y=c|x,\mathcal{D}_l),$$

while BALD chooses the point that maximizes the expected reduction in posterior entropy, or equivalently

$$\underset{(x,y)\in\mathcal{D}_u}{\arg\max}\, H[p(y|x,\mathcal{D}_l)] - \mathbb{E}_{p(\mathbf{w}|\mathcal{D}_l)}\big[H[p(y|x,\mathbf{w})]\big].$$

We can compute maximum entropy as well as the first of the two BALD terms in closed form, while we calculate the second term of BALD via a sampling based approach. We also include random sampling as a—on our kind of data rather competitive—baseline and evaluate on three data sets to show the adaptability of the method to the problem at hand.

Experimental Details. To evaluate the performance of the proposed pipeline, we take as the predictor is a standard LeNet5 sized model (two convolutional layers of 20, 50 channels and two linear layers of 500, 10 neurons) and as the guide a policy net consisting of two layers with 500 hidden neurons. We use three different image classification data sets to simulate varying difficulty while keeping the architectures and hyperparameters fixed. MNIST serves as a simple data set containing greyscale digits, FashionMNIST is a more difficult data set of greyscale clothing objects, and CIFAR-10 finally is a very difficult data set given the small classifier depth that requires the classification of colored natural images. The assumption of active learning that labels are scarce and expensive also entails the problem that a large separate validation set to evaluate and finetune hyperparameters is not feasible. Both nets are optimized via Adam [Kingma and Ba, 2015] using their suggested hyperparameters. In general we followed the assumption that an AL setting does not allow us to reserve valuable labeled data for hyper-parameter optimization so that they all remain fixed to the common defaults in the literature. The predictor is trained for 30 epochs between labeling rounds (labeling five points per round), while the policy net gets one update step after each round. To simulate the need to avoid a costly retraining after each labeling round the predictor net is initialized to the learned parameters from the last one, with a linearly decreasing learning rate after each round. In each experiment the state is constructed by ranking the unlabeled data according to their predictive entropy and then taking every twentieth point until $M = 50$ points. Since all three data sets consider a ten class classification problem, the result is a 500 dimensional normal distribution as the input to the policy network. We stop after having collected 400 points starting from an initial set of 50 data points.

Results. We summarize the results in Table 1. RAL can learn to adapt itself to the data set at hand and can always

[5] see github.com/manuelhaussmann/ral for a reference pytorch implementation of the proposed model.

	MNIST	FASHIONMNIST	CIFAR-10
Random	10.41 ± 2.28	24.64 ± 0.48	69.78 ± 0.69
Maxent	8.61 ± 1.25	25.72 ± 1.28	69.80 ± 0.32
BALD	6.91 ± 0.23	26.85 ± 0.49	69.69 ± 1.69
RAL (ours)	$\mathbf{6.81 \pm 0.99}$	$\mathbf{23.69 \pm 0.73}$	$\mathbf{68.96 \pm 1.03}$

Table 1: Results. The table gives the average final error ($\pm$ one standard deviation over five runs) after labeling 400 labeled points.

outperform the baselines. Note that our central goal is to evaluate the relative performance of RAL and the baselines in these experiments and not the absolute performance. For a real world application one would use deeper architectures for more complex data sets, incorporate pretrained networks from similar labeled data sets, and use data augmentation to make maximal use from the small labeled data. Further benefits would come from using semi-supervised information, e.g. by assigning pseudo-labels to data points the classifier assigns a high predictive certainty [Wang *et al.*, 2017]. Such approaches would significantly improve the classifier performance for all models, but since they would blur the contribution of the respective acquisition function, we consciously ignore them here. Note that although RAL uses a thinned Maxent ranking to generate its state, it can improve upon that strategy in every case. An ablation study showed that while the thinning process can improve over the plain Maxent in some settings if one were to use it as a fixed strategy, it is not sufficient to explain the final performance difference between RAL and Maxent. REINFORCE owes its success to the bulk filtering step, which substantially facilitates the RL problem by filtering out a large portion of the search space. The simplified problem can thus be improved within a small number of episodes. More interactions with the environment would certainly bring further improvement at the expense of increasing labeling cost. We here present only a proof-of-concept for the idea that can improve on the feedback-free AL even within limited interaction rounds. Further algorithmic improvements are worthwhile investigating as future work, such as applying TRPO [Schulman *et al.*, 2015] or PPO [Schulman *et al.*, 2017] in place of vanilla REINFORCE.

4 Related Work

The gold standard in AL methods has long remained to base on hard-coded and hand-designed acquisition heuristics (see [Settles, 2012] for a review). A first extension is to not limit oneself to one heuristic, but to learn how to choose between multiple ones, e.g. by a bandit gorithm [Baram *et al.*, 2004; Chu and Lin, 2016; Hsu and Lin, 2015] or a Markov Decision Process [Ebert *et al.*, 2012]. However this still suffers from the problem of being limited to existing heuristics.

A further step to gain more flexibility is to formulate the problem as a meta-learning approach. The general idea [Fang *et al.*, 2017; Konyushkova *et al.*, 2017; Pang *et al.*, 2018] is to use a set of labeled data sets to learn a general acquisition function that can either be applied as is to the target data set or finetuned on a sufficiently similar set of labeled data. Our approach differs from those attempts insofar as we learn the acquisition function based solely on the target data distribution while the data is labeled. If we take the scarcity of labels serious we can't allow ourselves the luxury of a separate large validation set to adapt a general heuristic. A separate large enough validation set also could not outperform the simple ablation study of allowing simpler acquisition functions that do not need a separate data set to instead combine that set with the labeled data they are training on. This is simply due to that as long as little labeled data is available the gain from being able to learn from extra data tends to outweigh the benefit one would get by a complicated acquisition function, and as soon as data becomes more abundant the effectiveness of any active learning method sharply. We therefore discard them from comparative analysis. A related area is the field of metareasoning [Callaway *et al.*, 2018], where an agent has to learn how to request based on a limited computational budget.

Alongside the sampling-based alternatives for BNN inference, which are already abundant and standardized [Blundell *et al.*, 2015; Gal and Ghahramani, 2016; Kingma *et al.*, 2015; Louizos *et al.*, 2017; Molchanov *et al.*, 2017], deterministic inference techniques are also emerging. While direct adaptations of expectation propagation are the earliest of such methods [Gast and Roth, 2018; Hernández-Lobato and Adams, 2015], they do not yet have a widespread reception due to their relative instability in training. This problem arises from the fact that EP do not provide any convergence guarantees, hence an update might either improve or deteriorate the model fit on even the training data. Contrarily, variational inference maximizes a lower bound on the log-marginal likelihood. Early studies exist on deterministic variational inference of BNNs [Haussmann *et al.*, 2019; Wu *et al.*, 2019]. However, neither quantifies the uncertainty quality by using the posterior predictive of their models for a downstream application. Earlier work that performs active learning with BNNs does exist [Hernández-Lobato and Adams, 2015; Gal *et al.*, 2017; Depeweg *et al.*, 2018]. However, all of these studies use hard-coded acquisition heuristics.

Our state construction method that forms a normal distribution from the posterior predictives of data points shortlisted by a bootstrap acquisition criterion is novel for the active learning setting. Yet, it has strong links to model-based reinforcement learning methods that propagate uncertainties through one-step predictors along the time axis [Deisenroth and Rasmussen, 2011].

5 Conclusion

We introduce a new reinforcement based method for labeling criterion learning. It is able to learn how to choose points in parallel to the labeling process itself, instead of requiring large already labeled subsets to learn on in an off-line setting beforehand. We achieve this by formulating the classification net, the policy net as well as the state probabilistically. We demonstrate its ability to adapt to a variety of qualitatively different data set situations performing similar to or even outperforming handcrafted heuristics. In the future work, we plan to extend the policy net with a density estimator that models the input data distribution so that it can also take the underlying geometry into account, making it less dependent on the quality of the probabilities.

References

[Baram *et al.*, 2004] Y. Baram, R. Yaniv, and K. Luz. Online choice of active learning algorithms. *JMLR*, 2004.

[Blundell *et al.*, 2015] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *ICML*, 2015.

[Callaway *et al.*, 2018] F. Callaway, S. Gul, P.M. Krueger, T.L. Griffiths, and F. Lieder. Learning to select computations. 2018.

[Chu and Lin, 2016] H. Chu and H. Lin. Can active learning experience be transferred? In *ICDM*, 2016.

[Deisenroth and Rasmussen, 2011] M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *ICML*, 2011.

[Depeweg *et al.*, 2018] S. Depeweg, J. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In *ICML*, 2018.

[Ebert *et al.*, 2012] S. Ebert, M. Fritz, and B. Schiele. Ralf: A reinforced active learning formulation for object class recognition. In *CVPR*, 2012.

[Fang *et al.*, 2017] M. Fang, Y. Li, and T. Cohn. Learning how to active learn: A deep reinforcement learning approach. In *EMNLP*, 2017.

[Frey and Hinton, 1999] B. J. Frey and G. E. Hinton. Variational learning in nonlinear gaussian belief networks. *Neural Computation*, 1999.

[Gal and Ghahramani, 2016] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

[Gal *et al.*, 2017] Y. Gal, R. Islam, and Z. Ghahramani. Deep Bayesian active learning with image data. In *ICML*, 2017.

[Gast and Roth, 2018] J. Gast and S. Roth. Lightweight probabilistic deep networks. In *CVPR*, 2018.

[Haussmann *et al.*, 2019] M. Haussmann, F.A. Hamprecht, and M. Kandemir. Sampling-free variational inference for bayesian neural networks by variance backpropagation. *UAI*, 2019.

[Hernández-Lobato and Adams, 2015] J.M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *ICML*, 2015.

[Houlsby *et al.*, 2012] N. Houlsby, F. Huszar, Z. Ghahramani, and J.M Hernández-Lobato. Collaborative gaussian processes for preference learning. In *NIPS*, 2012.

[Hsu and Lin, 2015] W. Hsu and H. Lin. Active learning by learning. In *AAAI*, 2015.

[Kingma and Ba, 2015] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Kingma and Welling, 2014] D.P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[Kingma *et al.*, 2015] D.P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *NIPS*, 2015.

[Konyushkova *et al.*, 2017] K. Konyushkova, R. Sznitman, and P. Fua. Learning active learning from data. In *NIPS*. 2017.

[Louizos *et al.*, 2017] C. Louizos, K. Ullrich, and M. Welling. Bayesian compression for deep learning. In *NIPS*, 2017.

[Molchanov *et al.*, 2017] D. Molchanov, A. Ashukha, and D. Vetrov. Variational dropout sparsifies deep neural networks. In *ICML*, 2017.

[Pang *et al.*, 2018] K. Pang, M. Dong, Y. Wu, and T. Hospedales. Meta-learning transferable active learning policies by deep reinforcement learning. *arXiv preprint*, 2018.

[Qiu *et al.*, 2017] Z. Qiu, D.J. Miller, and G. Kesidis. A maximum entropy framework for semisupervised and active learning with unknown and label-scarce classes. *IEEE transactions on neural networks and learning systems*, 2017.

[Schulman *et al.*, 2015] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *ICML*, 2015.

[Schulman *et al.*, 2017] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint*, 2017.

[Sener and Savarese, 2018] O. Sener and S. Savarese. Active learning for convolutional neural networks: Acore-set approach. In *CVPR*, 2018.

[Settles, 2012] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2012.

[Srinivas *et al.*, 2012] N. Srinivas, A. Krause, S.M. Kakade, and M.W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 2012.

[Wang and Manning, 2013] S. Wang and C. Manning. Fast dropout training. In *ICML*, 2013.

[Wang *et al.*, 2017] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[Williams, 1992] R.J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

[Wu *et al.*, 2019] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J.M. Hernández-Lobato, and A.L. Gaunt. Deterministic variational inference for robust bayesian neural networks. *ICLR*, 2019.