# Large-Scale Visual Active Learning with Deep Probabilistic Ensembles

Kashyap Chitta
Carnegie Mellon University
kchitta@andrew.cmu.edu

Jose M. Alvarez
NVIDIA
josea@nvidia.com

Adam Lesnikowski
NVIDIA
alesnikowski@nvidia.com

## Abstract

*Annotating the right data for training deep neural networks is an important challenge. Active learning using uncertainty estimates from Bayesian Neural Networks (BNNs) could provide an effective solution to this. Despite being theoretically principled, BNNs require approximations to be applied to large-scale problems, and have not been used widely by practitioners. In this paper, we introduce Deep Probabilistic Ensembles (DPEs), a scalable technique that uses a regularized ensemble to approximate a deep BNN. We conduct a series of active learning experiments to evaluate DPEs on classification with the CIFAR-10, CIFAR-100 and ImageNet datasets, and semantic segmentation with the BDD100k dataset. Our models consistently outperform baselines and previously published methods, requiring significantly less training data to achieve competitive performances.*

## 1. Introduction

To obtain high performance on modern computer vision problems, collecting the right data for supervised training of deep models is an important and challenging task. Active learning aims to obtain the smallest possible training set to solve a specific task by selecting the data the model uses for training. To this end, active learning methods use the *uncertainty* of the model as a means of quantifying what it does not know, to select, from a large unlabeled dataset, the data to be annotated [7]. In deterministic neural networks, uncertainty is typically measured based on the output of the last softmax-normalized layer. However, these estimates tend to be overconfident and do not always provide reliable information to select appropriate training data [48, 22]. On the other hand, Bayesian methods provide a principled approach to estimate the uncertainty of a model, and have recently gained momentum, but have not found widespread use in practice [29, 46, 19].

The formulation of a Bayesian Neural Network (BNN) involves placing a prior distribution over all the parameters of the network, and obtaining the posterior given the observed data [38]. The distribution of predictions provided by a trained BNN helps capture the model's uncertainty. However, training a BNN involves marginalizing over all possible assignments of weights, which is intractable for deep BNNs without approximations [21, 4, 17]. Existing approximation algorithms limit their applicability, since they do not specifically address the fact that deep BNNs on large datasets are more difficult to optimize than deterministic networks, and require extensive parameter tuning to provide good performance and uncertainty estimates [39]. Furthermore, estimating uncertainty in BNNs requires drawing a large number samples at test time (typically sequentially), which can be extremely computationally demanding.

In practice, a common approach to estimating uncertainty is based on ensembles [32, 2, 20]. Different models in an ensemble of networks are treated as if they were samples drawn from a BNN posterior. Ensembles are easy to optimize and fast to execute. However, they do not approximate uncertainty in the same manner as a BNN. For instance, the parameters in the first kernel of the first layer of a convolutional neural network may serve a completely different purpose in different members of an ensemble, and therefore, the variance of the values of these parameters after training cannot be compared to the variance we would have by placing distributions over the parameters in a BNN.

In this paper, we propose Deep Probabilistic Ensembles (DPEs), a novel approach to approximate BNNs using ensembles. Specifically, we derive a KL divergence regularization term for ensembles from variational inference, which is a common technique used to train BNNs [3]. We show the potential of our approach for active learning on large scale visual classification benchmarks with up to millions of samples. When applied to active learning on CIFAR-10, CIFAR-100 and ImageNet, DPEs consistently outperform strong baselines. To further show the applicability of our method, we propose a framework to actively annotate data for semantic segmentation, which on BDD100k yields IoU improvements of up to 20% while handling classes that are underrepresented in the training distribution.

Our contributions therefore are (i) we propose KL regularization for training DPEs, which combine the advantages of ensembles and BNNs; (ii) we apply DPEs to active image classification on large scale datasets; and (iii) we propose a framework for active semantic segmentation using DPEs. Our experiments on both visual tasks demonstrate the benefits and scalability of our method compared to existing methods. DPEs are parallelizable, easy to implement, yield high performance, and provide good uncertainty estimates when used for active learning tasks.

## 2. Related Work

**Regularizing Ensembles.** Using ensembles of neural networks to improve performance has a long history [23, 13]. Attempts to regularize ensembles have typically focused on promoting diversity among the ensemble members, through penalties such as negative correlation [34]. While these methods are presented in the perspective of improving the accuracy, our approach improves the ensemble's uncertainty estimates with little to no impact on accuracy.

**Training BNNs.** There are four popular approximations for training BNNs. The first one is *Markov Chain Monte Carlo* (MCMC). MCMC approximates the posterior through Bayes rule, by sampling to compute an integral [41]. This method is accurate but it is extremely sample inefficient, and has not been applied to deep BNNs. The second one is *Variational Inference* which is an iterative optimization approach to update a chosen variational distribution on each network parameter using the observed data [3]. Training is fast for reasonable network sizes, but performance is not always ideal, especially for deeper networks, due to the nature of the optimization objective.

The third approximation is *MC Dropout* [18] (and its variants [49, 1]). These methods regularize the network during training with a technique like dropout [47], and draw Monte Carlo samples *during test time* with different dropout masks as approximate samples from a BNN posterior. It can be seen as an approximation to variational inference when using Bernoulli priors [17]. Due to its simplicity, several approaches have been proposed using MC dropout for uncertainty estimation [29, 30, 16, 19]. However, empirically, these uncertainty estimates do not perform as well as those obtained from ensembles [32, 2]. In addition, MC dropout requires the dropout rate to be very well-tuned to produce reliable uncertainty estimates [39]. The last approximation is *Bayes by Backprop* [4] which involves maintaining two parameters on each weight, a mean and standard deviation, with their gradients calculated by backpropagation. Empirically, the uncertainty estimates obtained with this approach and MC Dropout perform similarly [45].

**(Deep) Active Learning.** A comprehensive review of classical approaches to active learning can be found at [44]. Most of these approaches rely on a confusion metric based on the model's predictions, like information theoretical methods that maximize expected information gain [36], committee based methods that maximize disagreement between committee members [10, 15, 37, 9], entropy maximization methods [33], and margin based methods [42, 6, 50, 28]. Data samples for which the current model is uncertain, or most confused, are queried for labeling usually one at a time and, in general, the model is re-trained for each of these queries. In the era of deep neural networks, querying single samples and retraining the model for each one is not feasible. Adding a single sample into the training process is likely to have no statistically significant impact on the performance of a deep neural network and the retraining becomes computationally intractable.

Scalable alternatives to classical active learning ideas have been explored for the iterative batch query setting. Pool-based approaches filter unlabeled data using heuristics based on sample confusion, and then choose what to label based on diversity [11, 51, 56]. More recent approaches explore active learning without breaking the task down using confusion metrics, through a lower bound on the prediction error known as the core-set loss [43] or learning the interestingness of samples in a data-driven manner through reinforcement learning [52, 14, 40]. Results in these directions are promising. However, state-of-the-art active learning algorithms are based on uncertainty estimates from network ensembles [2]. Our experiments demonstrate the gains obtained over ensembles when these are trained with the proposed regularization scheme.

## 3. Deep Probabilistic Ensembles

For inference in a BNN, we can consider the weights $w$ to be latent variables drawn from our prior distribution, $p(w)$. These weights relate to the observed dataset $x$ through the likelihood, $p(x|w)$. We aim to compute the posterior,

$$p(w|x) = \frac{p(w)p(x|w)}{p(x)} = \frac{p(w)p(x|w)}{\int p(w)p(x|w)dw}. \quad (1)$$

It is intractable to analytically integrate over all assignments of weights, so variational inference restricts the problem to a family of distributions $D$ over the latent variables. We then try to optimize for the member of this family that is closest to the true posterior in terms of KL divergence,

$$q^*(w) = \underset{q(w)\in D}{\arg\min}\, KL(q(w)||p(w|x)) \quad (2)$$

$$= \underset{q(w)\in D}{\arg\min}\, \mathbb{E}(\log \frac{q(w)}{p(w|x)}). \quad (3)$$

To make the computation tractable the objective is modified by subtracting a constant independent of our weights $w$. This new objective to be minimized is the negative of the Evidence Lower Bound (ELBO) [3],

$$-ELBO = \mathbb{E}(\log \frac{q(w)}{p(w|x)}) - \log p(x) \tag{4}$$

$$= \mathbb{E}(\log q(w)) - \mathbb{E}(\log p(w|x)) - \log p(x).$$

Substituting for the posterior term from Eq. 1, this can be simplified to

$$-ELBO = \mathbb{E}(\log q(w)) - \mathbb{E}(\log p(w)p(x|w)) \tag{5}$$

$$= \mathbb{E}(\log \frac{q(w)}{p(w)}) - \mathbb{E}(\log p(x|w)) \tag{6}$$

$$= KL(q(w)||p(w)) - \mathbb{E}(\log p(x|w)), \tag{7}$$

where the first term is the KL divergence between the distribution of weights in the approximated BNN $q(w)$ and the prior distribution $p(w)$. The second term is the expected negative log likelihood (NLL) of the data $x$ based on the current parameters $w$. Since the expectation is over all possible assignments of $q(w)$, we seek to make any generic deterministic network sampled from the BNN as close a fit to the data as possible.

The optimization difficulty in variational inference arises partly due to fragile co-adaptations that exist between different parameters in deterministic networks, which can be crucial to their performance [53]. Features typically interact with each other in a complex way, such that the optimal setting for certain parameters is highly dependent on specific configurations of the other parameters in the network.

An ensemble of deterministic networks perfectly exploits co-adaptation, as each network in the ensemble is optimized independently. Co-adaptation makes training easier, but can reduce the ability of deterministic networks to generalize [27]. Popular regularization techniques to improve generalization, such as Dropout, can be seen as a trade-off between co-adaptation and training difficulty [47].

For BNNs, we have the opposite problem: variance in our weight distributions prevents the BNN from exploiting co-adaptations during training, making it harder to optimize through standard variational inference or approximations to it like Bayes by Backprop. While in theory, this should lead to great generalization, it becomes very difficult to tune BNNs to produce competitive results.

**KL regularization.** In this section, we introduce a form of regularization to use the optimization simplicity of ensembles for training BNNs. The main properties of our approach compared to ensembles and BNNs are summarized in Table 1.

For our approach, we write the variational inference minimization objective from Eq. 7 for an ensemble as follows:

$$\mathbf{\Theta}^* = \arg\min_{\mathbf{\Theta}} \sum_{i=1}^{N} \sum_{e=1}^{E} l(y^i, \mathcal{M}_e(\mathbf{x}^i, \Theta_e)) + \beta\Omega(\mathbf{\Theta}), \tag{8}$$

Table 1. Properties of the proposed approach (DPE) compared to BNNs and ensembles. DPEs are simple to optimize like ensembles, and have a training objective similar to BNNs. As a result, we obtain high performance and good uncertainty estimates.

| Model | Co-adaptation | Training | Generalization |
|---|---|---|---|
| Ensemble | High | Easy | Low |
| DPE | Medium | Easy | High |
| BNN | Low | Hard | High |

where $\{(\mathbf{x}^i, y^i)\}_{i=1}^N$ is the training data, $E$ is the number of models in the ensemble, $l$ is the cross-entropy loss for classification, $\Theta_e$ refers to the parameters of the model $e$, and $\Omega$ is our KL regularization penalty over the joint set of all parameters $\mathbf{\Theta}$. By averaging the loss over each independent model, we are calculating the expectation of the ELBO's NLL in Eq. 7 term over only the current ensemble configurations, a subset of all possible assignments of $q(w)$. This is the main distinction between our approach and traditional variational inference.

The standard approach to training neural networks involves independently regularizing the parameters of each ensemble $\Theta_e$ with $L_1$ or $L_2$ regularization terms. We instead apply the KL divergence term in Eq. 7 to the objective as a regularization penalty $\Omega$, to the distribution of values of a given parameter over all members in the ensemble. If we choose the family of Gaussian functions for $p(w)$ and $q(w)$, this term can be analytically computed by assuming mutual independence between the network parameters and factoring the term into individual Gaussians. The KL divergence between two Gaussians with means $\mu_q$ and $\mu_p$, standard deviations $\sigma_q$ and $\sigma_p$ is given by

$$KL(q||p) = \frac{1}{2}(\log \frac{\sigma_q^2}{\sigma_p^2} + \frac{\sigma_p^2 + (\mu_q - \mu_p)^2}{\sigma_q^2} - 1). \tag{9}$$

We sum up this penalty over all the parameters of the ensemble, using a scaling term $\beta$ to balance the regularizer with the likelihood loss. $\Omega$ can be obtained by removing the terms independent of $q$ and substituting for the chosen values of $\mu_p$ and $\sigma_p$ from our prior into Eq. 9.

As a prior, we use the network initialization technique proposed in [24]. More specifically, for convolutional layers with the ReLU activation, we use zero-centered Gaussian distributions, with variances inversely proportional to the number of kernel parameters. We use fixed variance Gaussians for the batch normalization parameters, with mean 1 for the weights and 0 for the biases. For each parameter $\theta_i$ in a convolutional layer of dimensions $n_{in} \times n_{out} \times w_k \times h_k$,

$$\Omega(\theta_i) = \log \sigma_q^2 + \frac{2}{n_{out} w_k h_k \sigma_q^2} + \frac{\mu_q^2}{\sigma_q^2}. \tag{10}$$

In Eq. 10, the first term prevents extremely large variances compared to the prior, so the ensemble members do not diverge completely from each other. The second term heavily penalizes variances less than the prior, promoting diversity between members. The third term closely resembles weight decay, keeping the mean of the weights close that of the prior, especially when their variance is also low.

## 4. Uncertainty Estimation for Active Learning

In this section, we describe how DPEs can be used for active learning. There are four components in our pipeline:

- A large **unlabeled set** consisting of $N_u$ samples, $U = \{\mathbf{x}_u^j\}_{j=1}^{N_u}$, where each $\mathbf{x}^j \in X$ is an input data point. $X$ is the $D$ dimensional input feature space.
- A **labeled set**, consisting of $N_l$ labeled pairs, $L = \{(\mathbf{x}_l^j, y_l^j)\}_{j=1}^{N_l}$, , where each $\mathbf{x}^j \in X$ is a data point and each $y^j \in Y$ is its corresponding label. For a $K$-way classification problem, the label space $Y$ would be $\{1, ..., K\}$.
- An **annotator**, $\mathcal{A} : X \to Y$ that can be queried to map data points to their true labels
- Our **DPE** $\mathcal{M}$, which is a set of models $\{\mathcal{M}_e : X \to Y\}_{e=1}^{E}$, each trained to estimate the label of a given data point.

These four components interact with each other in an iterative process. In our setup, commonly referred to as *batch mode active learning*, every iteration involves 2 stages, which are summarized in Algorithm 1. In the first stage, the current labeled set $L^{(i)}$ is used to train the ensemble $\mathcal{M}^{(i)}$. The trained ensemble is applied to the existing unlabeled pool $U^{(i)}$ in the second stage, selecting a subset of $b$ samples from this pool to send to the annotator $\mathcal{A}$ for labeling. The growth parameter $b$ may be fixed or vary over iterations. The subset is moved from $U^{(i)}$ to $L^{(i+1)}$ before the next iteration. Training proceeds in this manner until an annotation budget of $B$ total label queries has been met.

Central to the second stage is a acquisition function $\alpha$ that is used for ranking all the available unlabeled data. For this, we employ the ensemble predictive entropy, given by

$$H_{ens} = -s(\sum_{e\in E} \mathbf{p}^{(e)})^T \log s(\sum_{e\in E} \mathbf{p}^{(e)}), \qquad (11)$$

where, $\mathbf{p}$ refers to the unnormalized model prediction vector and $s$ is the softmax function. Since the entropy for each query sample $\mathbf{x} \in U^{(i)}$ is independent of the others, the entire batch of $b$ samples can be added to $L_{i+1}$ in parallel.

In our experiments, to circumvent the need for annotators in the experimental loop, we use datasets for which all samples are annotated in advance, but hide the annotation away from the model. The dataset is initially treated as our unlabeled pool $U$, and labels are revealed to $\mathcal{M}$ as the experiment proceeds.

**Algorithm 1** Active learning with DPEs.

```
$i \leftarrow 0$
Randomly sample $b$ points from unlabeled set $U^{(0)}$
Move these $b$ points to initial labeled set $L^{(0)}$
total_added_samples $\leftarrow b$
while total_added_samples $\leq B$ do
    while val_drop_count $\leq$ patience do                    ▷ stage (i)
        Sample a mini-batch $x$ from $L^{(i)}$
        Forward pass, $\hat{y} = \mathcal{M}^{(i)}(x)$
        Compute gradients $\delta$ of loss $l(y, \hat{y}) + \beta\Omega$
        Update DPE, $\mathcal{M}^{(i)} \leftarrow \mathcal{M}^{(i)} + \delta$
        Periodically check val_acc
        if val_acc < best_acc then
            val_drop_count += 1
        else
            best_acc $\leftarrow$ val_acc
        end if
    end while
    for num_iteration_samples = 1 to $b$ do                    ▷ stage (ii)
        for $\mathbf{x} \in U^{(i)}$ do
            $\alpha(\mathbf{x}) \leftarrow H_{ens}(\mathbf{x})$
        end for
        Move $\arg\max_{\mathbf{x}} \alpha(\mathbf{x})$ to $L^{(i+1)}$
        total_added_samples += 1
    end for
    Update $b$
    $i$ += 1
end while
```

## 5. Experiments

In this section, we evaluate the effectiveness of our approach on active learning for image classification across various levels of task complexity with respect to number of classes, image resolution and quantity of data.

**Datasets.** We experiment with three widely used benchmarks for image classification: the CIFAR-10 and CIFAR-100 datasets [31], as well as the ImageNet 2012 classification dataset [12]. CIFAR datasets involve object classification tasks over natural images: CIFAR-10 is coarse-grained over 10 classes, and CIFAR-100 is fine-grained over 100 classes. For both tasks, there are 50k training images and 10k validation images of resolution $32 \times 32$. ImageNet consists of 1000 object classes, with annotation available for 1.28 million training images and 50k validation images of varying sizes and aspect ratios. All three datasets are balanced in terms of the number of training samples per class.

**Network Architecture.** We use ResNet [25] backbones to implement the individual members of the DPE. With ImageNet, we use a ResNet-18 with the standard kernel sizes and counts. We use a variant of the same architecture with pre-activation blocks and less striding for CIFAR

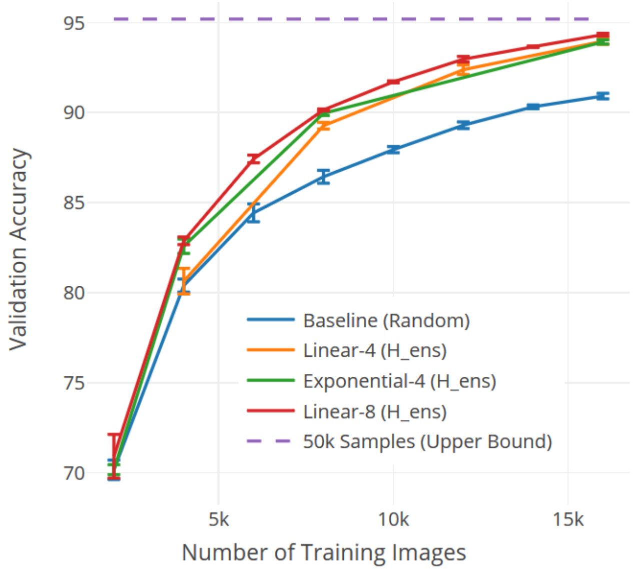

Figure 1. **CIFAR-10:** Effect of varying dataset growth parameter $b$ on validation accuracy using the ensemble first acquisition function. Dashed line is the upper bound as the performance of the model trained with all the data. As shown, regardless of the value of $b$, our approach achieves competitive performance to this upper bound despite using only 32% of the training data.

tasks; with strides of 1 instead of 2 in the initial convolution and first residual block [26].

**Implementation Details.** We retrain the ensemble from scratch for every iteration of active learning, to avoid remaining in a potentially sub-optimal local minimum from the earlier model trained with a smaller quantity of data. Optimization is done using Stochastic Gradient Descent with a learning rate of 0.1, batch size of 32 and momentum of 0.9. We do mean-std pre-processing, and augment the labeled dataset on-line with random crops and horizontal flips.

The KL regularization penalty $\beta$ is set to $10^{-5}$. We use a patience parameter (set to 25) for counting a certain number of epochs with no improvement in validation accuracy, in which case the learning rate is dropped by a factor of 0.1. We end training when dropping the learning rate also gives no improvement in the validation accuracy after a given number of epochs. In each iteration, if the early stopping criteria is not met, we train for a maximum of 400 epochs.

For our initial experiments, we attempted to use ensembles of 2 or 3 models, and found that the regularization penalty (specifically, the third term in Eq. 10, involving the mean square over the variance of the parameters across the DPE) causes instability and divergence in the loss. Though this effect subsides when there are 4 or more members in the ensemble, we found that both the stability and overall performance improve with more ensemble members. For all our experiments, we use ensembles of 8 members, beyond which we observed diminishing returns.

Table 2. **CIFAR-10:** Validation accuracy (in %) of the proposed approach, baselines and state-of-the art methods. The last column refers to the ratio (in %) between with accuracy using 10k samples to the upper bound for each method. As shown, our approach is able to achieve over 96% of the accuracy using only 20% of the labelling effort.

| **Method** | **10k (20%)** | **50k (100%)** | **Ratio** |
|---|---|---|---|
| Core-set [43] | 74 | 90 | 82.2 |
| Ensemble [2] | 85 | **95.5** | 89 |
| Single + Random | 85.2 | 94.4 | 90.3 |
| DPE + Random | 87.9 | 95.2 | 92.3 |
| Single + Linear-8 | 87.5 | 94.4 | 92.7 |
| **Ours** (DPE + Linear-8) | **92** | 95.2 | **96.3** |

## 5.1. Results

In our first experiment, we focus on analyzing variations of the growth parameter $b$. This parameter has large implications on the time needed to reach the final annotation budget $B$ and complete the active learning experiment, since it decides how many times the model must be retrained. For this experiment, we consider the CIFAR-10 dataset and an overall budget of $B = 16k$ samples. As a baseline, we use random sampling as an acquisition function with $b = 2k$, training the DPE 8 times. For reference, we also compute the upper bound achieved by training the DPE with all 50k samples in the dataset. Then, for comparison, we consider three variations of the growth parameter $b$ for our approach, which uses $H_{ens}$ as the acquisition function:

- **Linear-4**: We set $b = 4k$, training the DPE 4 times.
- **Exponential-4**: We initially set $b = 2k$, and then iteratively double its value ($b = 2k, b = 4k$ and $b = 8k$ for the 3 remaining active learning iterations).
- **Linear-8**: We set $b = 2k$, training the DPE 8 times.

Fig. 1 shows the error bars over three trials. Linear-8, which is computationally the most demanding, only marginally outperforms the other approaches. It is able to achieve 99.2% of the accuracy of the upper bound with only 32% of the labelling effort. This is probably due to the combination of two factors: (i) less correlation among the samples added (due to the lower growth parameter), and (ii) better uncertainty estimates in the intermediate stages due to more reinitializations. When the model is only retrained 4 times, both Linear-4 and Exponential-4 data addition achieve similar performance levels. From a computational point of view, the exponential approach is more efficient as the dataset size for the first three training iterations is smaller (2k, 4k and 8k compared to 4k, 8k and 12k).

In a second experiment, we focus on comparing our approach to state-of-the art deep active learning methods.

Table 3. Summary of acquisition functions.

| Function | Formula |
|---|---|
| $H_{ens}$ | $-s(\sum_{e\in E}\mathbf{p}^{(e)})^T\log s(\sum_{e\in E}\mathbf{p}^{(e)})$ |
| $H_{cat}$ | $\sum_{e\in E}(-s(\mathbf{p}^{T(e)})\log s(\mathbf{p}^{(e)}))$ |
| $MI$ | $H_{ens}-H_{cat}$ |
| $V$ | $\sum_{k\in K}[\underset{e\in E}{\mathrm{Var}}(s(\mathbf{p}^{(e)}))]_k$ |
| $VR$ | $1-\frac{f_m}{E}$ where $M=\underset{e\in E}{\mathrm{Mode}}(\underset{k\in K}{\arg\max}\,\mathbf{p}_k^{(e)})$ and $f_m=\sum_{e\in E}(\underset{k\in K}{\arg\max}\,\mathbf{p}_k^{(e)}=M)$ |

Table 4. **CIFAR-10:** Validation accuracy (%) of DPEs with different acquisition functions and labeling budgets. $H_{cat}$ and $VR$ provide marginally better results.

| Acquisition | 4k (8%) | 8k (16%) | 16k (32%) |
|---|---|---|---|
| Random | 80.60 | 86.80 | 91.08 |
| $H_{cat}$ | **82.96** | 89.92 | **94.10** |
| $MI$ | 82.70 | 90.00 | 93.97 |
| $V$ | 82.19 | 90.05 | 93.92 |
| $VR$ | 82.76 | **90.29** | 94.06 |
| **Ours** ($H_{ens}$) | 82.58 | 89.96 | 93.92 |

Table 5. **CIFAR-100:** Validation accuracy (%) of DPEs with different acquisition functions and labeling budgets. $H_{ens}$ and $VR$ provide marginally better results.

| Acquisition | 4k (8%) | 8k (16%) | 16k (32%) |
|---|---|---|---|
| Random | 39.57 | 54.92 | 66.65 |
| $H_{cat}$ | 38.47 | 55.63 | 69.16 |
| $MI$ | 39.96 | 55.50 | 69.01 |
| $V$ | **41.13** | 56.62 | 69.17 |
| $VR$ | 41.08 | **56.97** | 69.45 |
| **Ours** ($H_{ens}$) | 41.05 | 56.91 | **69.79** |

Table 6. **CIFAR-10 and CIFAR-100:** Validation accuracy of the proposed approach compared to standard ensembles. Initial 2k is randomly sampled. DPEs produce consistently better uncertainty estimates and improve active learning performance.

| Task | Data Sampling | 4k (8%) | 8k (16%) | 16k (32%) |
|---|---|---|---|---|
| C-10 | Random | 80.60 | 86.80 | 91.08 |
| | Ensemble | 82.41 | 90.05 | 94.13 |
| | **Ours** (DPE) | **82.88** | **90.15** | **94.33** |
| C-100 | Random | 39.57 | 54.92 | 66.65 |
| | Ensemble | 40.49 | 56.89 | 69.68 |
| | **Ours** (DPE) | **40.87** | **56.94** | **70.12** |

Specifically, we consider the core-set selection [43] and ensembles [2]. The former uses a single VGG-16. The latter an ensemble of DenseNet-121 models. In addition, we consider 'Random' and 'Linear-8' experiments from Fig. 1 with a *single model* instead of the DPE. We use the same architecture (ResNet-18) for either a single model or a member of the DPE. For the baseline, we train the model using $L_2$ regularization instead of KL regularization. Table 2 summarizes our results and those reported in their corresponding papers using 10k samples on CIFAR-10. We also include the upper bound of each experiment. That is, the accuracy obtained when the model is trained using all the data available. As shown, our approach outperforms all the others, not only achieving higher accuracy, but also reducing the gap to the corresponding upper bound.

In a third experiment, we analyze the relevance of the acquisition function, a key component in the active learning pipeline. To this end, we compare $H_{ens}$ (as defined in Eq. 11) with four other acquisition functions: categories first entropy ($H_{cat}$), mutual information ($MI$), variance ($V$) and variation ratios ($VR$). The formulation of these functions and the one used in our approach is listed in Table 3.

Categories first entropy is the sum of the entropy of each of the members of the ensemble. Mutual information, also known as Jensen-Shannon divergence or information radius, is the difference between $H_{ens}$ and $H_{cat}$. This function is appropriate for active learning as it has been shown to distinguish *epistemic uncertainty* (i.e., the confusion over the model's parameters that apply to a given observation) from *aleatoric uncertainty* (i.e., the inherent stochasticity associated with that observation [30]). Aleatoric uncertainty cannot be reduced even upon collecting an infinite amount of data, and can be an irrelevant distractor to an uncertainty based active learning system. Variance measures the inconsistency between the members of the ensemble, which also aims to isolate epistemic uncertainty [46]. Finally, variation ratios is the number of non-modal predictions (that do not agree with the mode, or majority vote) made by the ensemble, and can be thought of as a discretized version of variance.

Our results are listed Tables 4 and 5 for CIFAR-10 and CIFAR-100 respectively. Each result corresponds to the mean of three trials. Intuitively, functions focused on epistemic uncertainty should perform better than simpler ones. However, as shown in our experiments, in practice, there is no significant difference in performance among these acquisition functions. Our choice of using a simple entropy based acquisition function ($H_{ens}$) provides competitive results to the others.

Finally, we focus on comparing DPEs to ensembles trained using standard $L_2$ regularization. To this end, we run experiments on CIFAR-10, CIFAR-100 and ImageNet. For CIFAR, we use an initial growth parameter of 2k samples, and compare performances at 4k, 8k and 16k samples (Exponential-4). For ImageNet, we use a slightly different experimental setting, comparing performances at 2% and 4% of the dataset (25.6k and 51.2k samples), since repeated training takes far too long with more samples. A summary of these results and a baseline using random data sampling

Table 7. **ImageNet:** Top-5 validation accuracy comparing the proposed approach to standard ensembles. Initial 12.8k is randomly sampled. We see a 3%-4% improvement in performance over the random baseline by employing active learning.

| Task | Data Sampling | 25.6k (2%) | 51.2k (4%) |
|---|---|---|---|
| ImageNet | Random | 48.97 | 64.06 |
| | Ensemble | **52.95** | 67.25 |
| | **Ours** (DPE) | 52.89 | **67.28** |

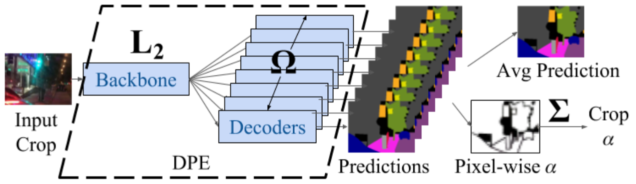

Figure 2. Setup for active segmentation, with a single shared encoder and multiple decoder heads. Uncertainty in the crops is measured by summing the acquisition functions over all pixels.

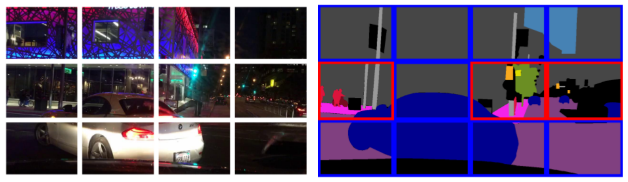

Figure 3. Example of crops queried on a $4 \times 3$ grid for semantic segmentation. Some parts of the image (outlined in red) can be more interesting than others (outlined in blue). Querying labels for crops can lead to a more effective utilization of the annotation budget than querying for images (best viewed in color).

is listed in Tables 6 and 7 for CIFAR and ImageNet respectively. As shown, active learning with DPEs clearly outperforms the baselines, and consistently provides better results compared to ensemble based active learning methods. From these experiments, we can conclude that our proposal provides reliable uncertainty estimates on unlabeled data.

## 6. Active Semantic Segmentation

In this section, we introduce our framework and results for applying DPEs for active semantic segmentation. The goal of semantic segmentation is to assign a class to every pixel in an image. Generating high-quality ground truth at the pixel level is a time-consuming and expensive task. For instance, labeling an image was reported to take 60 minutes on average for the CamVid dataset [5], and approximately 90 minutes for the Cityscapes dataset [8]. Due to the substantial manual effort involved in producing high-quality annotation, segmentation datasets with precise and comprehensive label maps are *orders of magnitude* smaller than classification datasets. Therefore, active learning is an appealing approach to effectively select data to be annotated for this task.

Table 8. **BDD100k:** % mIoU comparing the proposed approach to standard ensembles for active segmentation. Initial 3.3k crops are randomly sampled. In our setup, DPEs improve upon the mIoU of ensembles by up to 1%.

| Data Sampling | 6.7k (8%) | 13.4k (16%) | 26.9k (32%) |
|---|---|---|---|
| Random | 45.37 | 46.92 | 48.41 |
| Ensemble | 45.60 | 47.62 | 49.14 |
| **Ours** (DPE) | **45.80** | **48.10** | **50.12** |

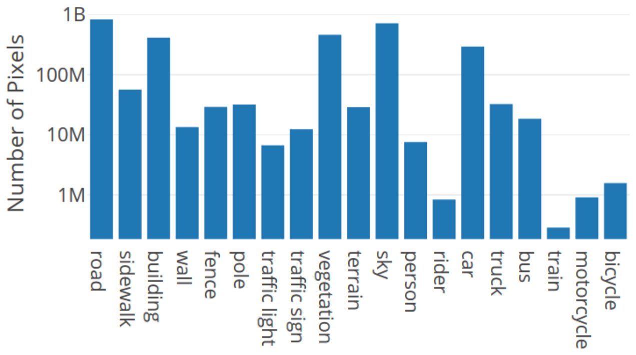

Figure 4. **BDD100k:** Number of pixels of each class, varying from a few million to nearly billions of pixels.

The proposed framework is outlined in Fig. 2. The DPE consists of a shared convolutional backbone as an encoder, and an ensemble of decoders. As all the components are trained jointly, memory requirements scale linearly with the number of decoders, but only for the parameters and activations associated these decoder layers. In our setting, we focus on analyzing uncertainty at object level and thus, we train the decoder with KL regularization while the shared encoder is trained using the common $L_2$ regularization.

The loss used to train semantic segmentation models is computed independently at each pixel. Therefore, training these models from partial annotations of an image is straightforward. In our experiments, as shown in Fig. 3, we split a training image into a $4 \times 3$ grid of cropped sections and use our acquisition function to make queries for crops rather than querying for entire images. To measure the uncertainty of a crop, we sum the acquisition function $\alpha$ over each pixel in the crop.

### 6.1. Experiments

We set up the active segmentation experiments in a similar way to the Exponential-4 scheme we used for classification, with iterations at 8%, 16% and 32% of the data, retraining from scratch for every new iteration.

**Dataset.** We experiment with the BDD100k [55] dataset, which consists of 7000 training images and 1000 validation images of resolutions $1280 \times 720$ provided with fine pixel annotation over 19 classes. While the classes were balanced in the classification datasets, this task involves significant

Table 9. **BDD100k:** Active learning results after training with 26.9k image crops (32% of the dataset), showing % IoU for each class and mean IoU. For comparison, we include the upper bound results obtained using a model trained with all 84k training crops. For IoU differences of at least 1%, we mark the better result in **bold**. We observe improvements in performance on foreground classes (person through bicycle), especially those with very few instances in the training distribution (like motorcycle).

| **Data Sampling** | road | sidewalk | building | wall | fence | pole | traffic light | traffic sign | vegetation | terrain | sky | person | rider | car | truck | bus | train | motorcycle | bicycle | **mIoU @ 26.9k** | **mIoU @ 84k** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 84k Crops (Upper Bound) | 92.1 | 56.3 | 81.6 | 20.7 | 40.3 | 42.5 | 44.9 | 43.6 | 84.3 | 45.0 | 93.7 | 56.1 | 25.1 | 86.6 | 38.9 | 56.9 | 0.0 | 38.6 | 42.1 | - | 52.1 |
| Random | 91.2 | 54.7 | 80.2 | 17.3 | **38.0** | 38.4 | 40.8 | 40.3 | 83.6 | 43.3 | 92.9 | 53.6 | 18.5 | 84.8 | 33.3 | 47.2 | 0.0 | 16.9 | 44.9 | 48.4 | - |
| **Ours** (DPE + $H_{ens}$) | 91.3 | 55.0 | 80.3 | 18.4 | 36.4 | 38.2 | 40.0 | 39.9 | 83.5 | 42.6 | 93.0 | **55.1** | **20.1** | 85.5 | 34.1 | **52.2** | 0.0 | **36.8** | 50.0 | **50.1** | - |
| Target Signs (DPE + $V_w$) | 91.1 | 55.2 | 80.1 | **22.0** | 36.9 | 38.7 | 40.5 | **41.9** | 83.6 | 41.3 | 93.0 | 53.2 | 16.9 | 85.3 | 31.8 | 48.0 | 0.0 | 34.1 | 49.8 | 49.7 | - |

class imbalance in the labels, as shown in Fig. 4.

**Network Architecture.** We use a fully convolutional encoder-decoder network in our experiments [35]. For the encoder, we dilate the last 2 blocks of a ResNet-18 backbone so that the encoded features are 1/8 of the input resolution [54]. For each decoder, we use two $1 \times 1$ convolutional layers of 64 and 19 kernels each.

**Implementation Details.** Since we have 12 crops per image, we perform active learning over the unlabeled set of 84k crops. In comparison to the classification networks, we use a smaller batch size of 8, learning rate of 0.0001, $\beta$ of $10^{-4}$, and patience of 10 epochs. Our network is initialized by pre-training on the same 19-class segmentation task with the Cityscapes dataset [8]. We resize the Cityscapes images to a resolution of $1536 \times 768$, and train on random crops of size $320 \times 320$ for 100 epochs, with a batch size of 16 and an exponentially decaying learning rate initialized at 0.001.

Though it is common to use tricks such as data augmentation, class-balanced sampling, class-wise weighting of losses, or auxiliary losses at intermediate layers in semantic segmentation, we avoid these to maintain simplicity in our setup. To verify that our training setup and hyperparameter choices still perform fairly well for this task, we train a fully-supervised version of our model on the entire dataset, which achieves a mean IoU of 52.1%. This is just 1% less than the baseline result provided with the BDD100k dataset release, which place the mean IoU for the DRN-D-22 model (architecturally similar to ours) at 53.2% [55].

**Results.** The mean segmentation IoUs over 3 trials are shown in Table 8. As with classification, we observe clear overall benefits of DPEs in the semantic segmentation setup, with nearly a 2% improvement in mean IoU over the random baseline and 1% over ensembles at 26.9k training crops. DPEs recover 96.2% of the performance of the upper bound with just 32% of the labeling effort.

**Targeting Specific Classes.** We compare our class-wise performance at 26.9k crops to the random baseline and fully-supervised upper bound in Table 9. We observe the largest margins of improvement with our method for foreground classes with fewer instances in the training data. On the motorcycle class, of which there are only 240 training instances, we see a difference in IoU of nearly 20%.

We now consider a special use case where the performance in specific classes is critical. To that end, we define an acquisition function suitable for targeting specific classes based on a class weighting vector, $\mathbf{w}$. This class-weighted acquisition function is defined as follows:

$$V_w = \sum_{k \in K} \mathbf{w}_k [\underset{e \in E}{\text{Var}}(s(\mathbf{p}^{(e)}))]_k. \qquad (12)$$

Table 9 summarizes the results using $V_m$ for targeting the traffic sign class. Specifically, we set $\mathbf{w}_k$ for traffic sign to 1 and the remaining weights to 0. As shown, this results in a 2% improvement in IoU over the $H_{ens}$ acquisition function for the traffic sign class, without affecting significantly the IoU improvements on other classes (even improving by 3.6% on the wall class). Though the 2% gap does not seem as significant a change (motorcycle still improves over random by 17%), we note that the gap to the upper bound for traffic signs is improved to 1.7% from 3.7%, which is a 54.1% relative performance gain.

## 7. Conclusion

In this paper, we introduced Deep Probabilistic Ensembles (DPEs) for uncertainty estimation in deep neural networks. The key idea is to train ensembles with a novel KL regularization term as a means to approximate variational inference for BNNs. Our results demonstrate that DPEs improve performance on active learning tasks over baselines and state-of-the-art active learning techniques on three different image classification datasets. Importantly, contrary to traditional Bayesian methods, our approach is simple to integrate in existing frameworks and scales to large models and datasets without impacting performance. Moreover, when applied to active semantic segmentation, DPEs yield up to 20% IoU improvement in underrepresented classes.

## References

[1] A. Atanov, A. Ashukha, D. Molchanov, K. Neklyudov, and D. Vetrov. Uncertainty Estimation via Stochastic Batch Normalization. *ArXiv e-prints*, 2018. 2

[2] W. H. Beluch, T. Genewein, A. Nrnberger, and J. M. Khler. The power of ensembles for active learning in image classification. In *CVPR*, 2018. 1, 2, 5, 6

[3] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational Inference: A Review for Statisticians. *ArXiv e-prints*, 2016. 1, 2, 3

[4] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. In *ICML*, 2015. 1, 2

[5] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV*, 2008. 7

[6] C. Campbell, N. Cristianini, and A. J. Smola. Query learning with large margin classifiers. In *ICML*, 2000. 2

[7] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine Learning*, 1994. 1

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 7, 8

[9] A. Culotta and A. McCallum. Reducing labeling effort for structured prediction tasks. In *AAAI*, 2005. 2

[10] I. Dagan and S. P. Engelson. Committee-based sampling for training probabilistic classifiers. In *ICML*, 1995. 2

[11] B. Demir, C. Persello, and L. Bruzzone. Batch-mode active-learning methods for the interactive classification of remote sensing images. *IEEE Trans. on Geo. and Rem. Sen.*, 2011. 2

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 4

[13] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems, LBCS-1857*, pages 1–15. Springer, 2000. 2

[14] M. Fang, Y. Li, and T. Cohn. Learning how to Active Learn: A Deep Reinforcement Learning Approach. *ArXiv e-prints*, 2017. 2

[15] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 1997. 2

[16] Y. Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016. 2

[17] Y. Gal and Z. Ghahramani. Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. *ArXiv e-prints*, 2015. 1, 2

[18] Y. Gal, R. Islam, and Z. Ghahramani. Deep bayesian active learning with image data. *ArXiv e-prints*, 2017. 2

[19] Y. Gal and L. Smith. Sufficient Conditions for Idealised Models to Have No Adversarial Examples: a Theoretical and Empirical Study with Bayesian Neural Networks. *ArXiv e-prints*, 2018. 1, 2

[20] Y. Geifman, G. Uziel, and R. El-Yaniv. Boosting Uncertainty Estimation for Deep Neural Classifiers. *ArXiv e-prints*, 2018. 1

[21] A. Graves. Practical variational inference for neural networks. In *NIPS*. 2011. 1

[22] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On Calibration of Modern Neural Networks. *ArXiv e-prints*, 2017. 1

[23] L. K. Hansen and P. Salamon. Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(10):993–1001, Oct. 1990. 2

[24] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 3

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[26] K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. *ArXiv e-prints*, 2016. 5

[27] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv e-prints*, 2012. 3

[28] A. J. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification. In *CVPR*, 2009. 2

[29] A. Kendall, V. Badrinarayanan, and R. Cipolla. Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. *ArXiv e-prints*, 2015. 1, 2

[30] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NIPS*, 2017. 2, 6

[31] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 4

[32] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, 2017. 1, 2

[33] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *SIGIR*, 1994. 2

[34] Y. Liu and X. Yao. Ensemble learning via negative correlation. *Neural Networks*, 12(10):1399 – 1404, 1999. 2

[35] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 8

[36] D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 1992. 2

[37] A. McCallum and K. Nigam. Employing em and pool-based active learning for text classification. In *ICML*, 1998. 2

[38] R. M. Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1995. 1

[39] I. Osband. Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout. In *NIPS Workshops*, 2016. 1, 2

[40] K. Pang, M. Dong, Y. Wu, and T. Hospedales. Meta-Learning Transferable Active Learning Policies by Deep Reinforcement Learning. *ArXiv e-prints*, 2018. 2

[41] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2005. 2

[42] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. 2000. 2

[43] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. *ICLR*, 2018. 2, 5, 6

[44] B. Settles. Active learning literature survey. Technical report, 2010. 2

[45] A. Siddhant and Z. C. Lipton. Deep Bayesian Active Learning for Natural Language Processing: Results of a Large-Scale Empirical Study. *ArXiv e-prints*, 2018. 2

[46] L. Smith and Y. Gal. Understanding Measures of Uncertainty for Adversarial Example Detection. In *UAI*, 2018. 1, 6

[47] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *ICML*, 2014. 2, 3

[48] C. Szegedy, G. Inc, W. Zaremba, I. Sutskever, G. Inc, J. Bruna, D. Erhan, G. Inc, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 1

[49] M. Teye, H. Azizpour, and K. Smith. Bayesian Uncertainty Estimation for Batch Normalized Deep Networks. *ArXiv e-prints*, 2018. 2

[50] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2002. 2

[51] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin. Cost-effective active learning for deep image classification. *IEEE Trans. Cir. and Sys. for Video Technol.*, 2017. 2

[52] M. Woodward and C. Finn. Active One-shot Learning. *ArXiv e-prints*, 2017. 2

[53] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014. 3

[54] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 8

[55] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling. *ArXiv e-prints*, 2018. 7, 8

[56] Y. Zhang, M. Lease, and B. C. Wallace. Active discriminative text representation learning. In *AAAI*, 2017. 2