

Uncertainty approximation in neural networks using parameter-space proximity regularization

Mihály Vetro

*Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
vetromisu@gmail.com*

Gábor Hullám

*Department of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
gabor.hullam@mit.bme.hu*

Abstract—A common question regarding the application of neural networks is whether the predictions of the model are reliable, in other words, what is the degree of uncertainty of our model. Generalizing a neural network into a Bayesian neural network is a frequent choice to quantify uncertainty. This means the extension of scalar weights and biases of the network to random variables. In terms of implementation, there are two main approaches: (1) the assumption of an unknown distribution for the random variables (i.e. weights) which are sampled and then utilized to infer the distribution of the output; (2) the assumption of a prior distribution for the random variables, and search for the output in the form of a similar posterior distribution. A common drawback of both approaches is that their scalability is limited, and generally require more computational resources than a simple neural network. In this paper, we introduce a new parameter sampling method, which maximizes the pairwise distance in parameter space between the models in the ensemble, therefore ensuring model diversity. Results indicate that this method generally needs less samples from the parameter space to be effective, thus it surpasses the other investigated approximation methods in terms of scalability. Finally, we apply our method to a real-life problem related to semantic segmentation of objects relevant in urban driving environments.

Index Terms—uncertainty, Bayesian neural network, Bayesian approach, approximation methods

I. INTRODUCTION

In real-world applications concerning various estimation problems, noise, observation bias and most other sources of uncertainty related to the training data, and therefore the final model, cannot be fully avoided. Thus the demand is continuously rising for more reliable models, and for measuring the output uncertainty and the error probability of a model, especially in safety-critical applications, where it is often obligatory to provide guarantees about the reliability of a given solution. Besides this, uncertainty estimation often comes with a great computational overhead compared to the construction of an ordinary model. Therefore in this paper, we introduce a new, relatively cost-effective method, and compare it to some other known approaches used to estimate uncertainty in predictive models.

This research was supported by the ÚNKP-20-5-BME-92 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund, and the János Bolyai Research Scholarship.

II. APPROXIMATING UNCERTAINTY

In general, uncertainty can be originated from the imperfections of our data (mainly incompleteness and noise), or the limits defined by the parameter space in which we are searching for the solution. Although data uncertainty can be further categorised [3] depending on its source, which can help with the improvement of our model, in this paper we are only focusing on the quantification of the total uncertainty of the model's output. In order to do this, we should consider the output (y) as a random variable (instead of a point estimate), which depends on the input (x) of the model: $p(y|x)$. The learned model is most likely not perfect, and so the distribution of the output variable will also depend on the restrictions introduced by the parameter space and the training data at our disposal. Considering these aspects, we can formulate the final distribution of the output as follows:

$$p(y|x, D) = \int p(y|x, \theta)p(\theta|D)d\theta \quad (1)$$

As we can see in equation 1, the information provided by the training data (D) is taken into consideration via the trainable parameters (θ) of the model. If we know the exact value of θ , then calculating $p(y|x, \theta)$ is basically the same as running inference on the model. The main problem arises when we consider that the training process, i.e. adjusting the parameters based on the data, is usually imperfect, due to the stochasticity of the training method, and there is also some uncertainty related to the training data. Because of this, determining the exact distribution of θ , or more precisely determining $p(\theta|D)$ is infeasible, or at least impractical in most cases. Therefore, our main goal is to provide a good approximation to this distribution, which is computable and preferably has minimal computational overhead compared to an ordinary model, which is typically a maximum a posteriori estimate of θ over $p(\theta|D)$.

III. RELATED WORK

When it comes to approximating the distribution of $p(\theta|D)$, two main approaches exist: (1) the first approach relies on taking samples from an unknown distribution over θ , (2) while the second approach assumes a known prior distribution (e.g. multivariate normal), and searches for the posterior in the

same form. The first approach is followed mostly by various Monte Carlo methods, such as *Stochastic Gradient Langevin Dynamics* (SGLD [2]) or *Anchored Ensembling* [4], while methods taking the second approach are called variational methods, like *Bayes-by-Backprop* (BBB [1]) or *Stein Variational Gradient Descent* (SVGD [7]). The methods using Monte Carlo sampling have the capability to approximate complex (but unknown) distributions, and presumably give a better estimation of the true posterior, while the variational methods have the advantage of a fully known (but more confined) posterior approximation. While both approaches and their methods have their advantages and shortcomings, it is generally true, that they are more complicated and computationally more resource-intensive than a simple neural network created for the same machine learning problem.

IV. ENSURING MODEL DIVERSITY

There are two main criteria that an uncertainty-estimation method has to meet: (1) it has to produce a metric, which consistently correlates with the expected error of the model, and (2) this has to be achieved with minimal performance overhead compared to the original estimator on which it is based. The usual approach is to approximate the true posterior distribution of the parameters. In this case, to estimate the uncertainty of the output, such methods often require a large amount of samples from the parameter space. Considering that every sample taken from $p(\theta|D)$ is a whole new model on itself, using a large number of samples greatly increases the computational and storage requirements of the method. Therefore, instead of approximating the true posterior of θ , we are focusing on the approximation of a modified posterior, which aims to efficiently utilize every new sample, minimizing the number of samples required to reliably estimate the output uncertainty. To achieve this, our method maximizes the pairwise distance in the parameter-space between samples by sequentially training new models with the following modified loss function (for the $(m + 1)$ -th model):

$$L_{m+1} = \rho \left(\frac{1}{m} \sum_{i=1}^m \|\theta - \theta_i\|_2 \right)^{-\tau} + \lambda H(y, \hat{y}) \quad (2)$$

Where:

- ρ is the regularization multiplier
- τ is the regularization exponent
- θ_i is the parameter vector of the i -th model from the already trained models
- θ is the parameter vector of the current model
- m is the number of already trained models
- λ is the learning rate
- y and \hat{y} are the true and the predicted output vectors
- H is the cross entropy function

The second half of the loss function is a simple cross-entropy error, while the first half accounts for the parameter space distance regularization. This latter part significantly increases the loss, when the average distance from the previously trained models is low, and its value diminishes as the average distance

gets larger. This way, the training algorithm is forced to find a local minimum of the error function further from the previously created models, which greatly increases overall model diversity. In other words, this results in a model-ensemble, the members of which provide a different, but similarly adequate solution for the given estimation problem. This way, the variance of the members' predictions will be low when the given input is from a "well known"¹ part of the input space, and will be high otherwise. Although this property is generally true for most other known methods, we expect that this can be achieved with fewer models by ensuring model diversity. We refer to our proposed method the Distance Regularized Ensemble Approximation Method (or DREAM for short).

V. PRACTICAL RESULTS

To observe the performance of our method, we have tested it on the CityScapes dataset [5], which is strongly connected to self driving, and object detection. The dataset consists of pictures taken from the windshield camera of a car driving through the urban area of various European cities. These pictures are annotated pixel-by-pixel with various class labels, from which we used the 16 most relevant ones. We achieved this by merging some of the similar classes, and dumping the remaining ones to the "Other" class. Example outputs of a small ensemble consisting of 3 models, trained with the DREAM method are shown in figure 2. Results indicate, that the variance (or uncertainty) of the ensemble's output are noticeably higher in cases when the output is incorrect, compared to the parts of the image where the output is correct. To verify this, the average output variance where the ensemble's prediction is correct should be observed an compared to the output variance of cases in which the prediction is incorrect. Figure 1 shows the variance of the DREAM method for several different ensemble sizes. This demonstrates that there is a consistently large gap between the average variance of the correct and incorrect outputs, even in case of a small sized ensemble.

A. Comparing different methods

With regards to predictive performance, the SGLD and DREAM methods both achieved an overall accuracy of 87%, while the Bayes-by-Backprop model achieved 80% with the same model architecture on the validation dataset (500 images), after being trained on the training dataset (4000 images). Arguably, the main goal of output uncertainty approximation is to estimate the likelihood of the model's prediction being incorrect. This requires that the output uncertainty is relatively high when the prediction of the model is 'wrong', and relatively low when the model provides a correct prediction. Considering the fact that the output variance may be in different ranges for different approaches, in order to enable

¹The part of the input space that we have reliable information about, typically (but not exclusively) close to the known training data.

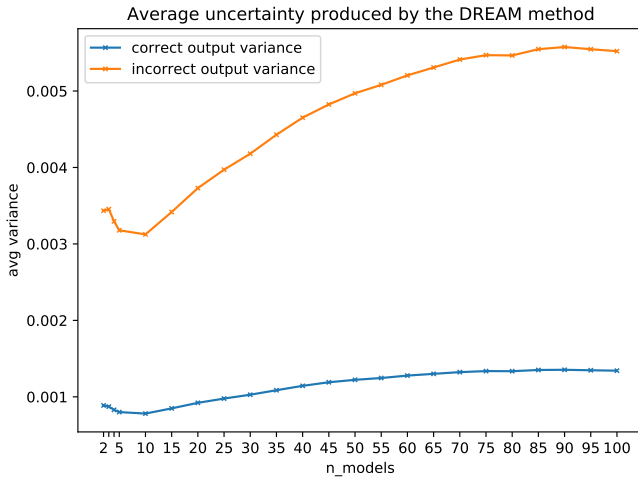


Fig. 1. Average output variance of the DREAM method on the CityScapes validation dataset for correct and incorrect predictions given various ensemble sizes.

their comparative analysis we compute the ratio between the two average variance values. This can be expressed as follows:

$$m(x_n) = \frac{U_{incorrect}(x_n)}{U_{correct}(x_n)} \quad (3)$$

Where $U_{incorrect}(x_n)$ is the average output uncertainty (variance) of an ensemble consisting of n models, produced by the x method on incorrect predictions, and $U_{correct}(x_n)$ is the average output uncertainty (variance) of an ensemble consisting of n models, produced by the x method on correct predictions. This metric indicates the amount of information that a method’s output uncertainty conveys about the likelihood of the model’s prediction being correct or incorrect. The comparison between three different methods (including DREAM) based on this metric is shown in figure 3. Results indicate that the $m(DREAM_n)$ value is consistently high even with smaller sized ensembles, surpassing the two other approaches on the CityScapes validation dataset. This further ensures the correlation between model error and output variance with a small ensemble size suggested by figure 2, and indicates that the DREAM method provides a more useful uncertainty estimation even with low model counts, compared to the other investigated methods (SGLD and Bayes-by-Backprop) on the CityScapes dataset.

B. Knowledge distillation

There is a fairly known approach for minimizing the size and resource requirements of estimators called *knowledge distillation* [6]. In short, this method relies on forming a large dataset by taking random samples from the output of the original, and then constructing a smaller, more efficient estimator on that dataset. Generally, this presents two challenges in our case: (1) first, we have to acquire appropriate samples from the input space, (2) and second, we must train our distilled estimator to approximate both the predictions and

output variance of the original sample. The first challenge is generally considered hard, because in most cases it is infeasible to reliably model the input distribution to take samples from it. Fortunately, the CityScapes dataset contains 20,000 unlabeled images taken from various cities with the same setup, which therefore can be considered as samples taken from the very same (or at least similar) distribution from which the training and validation samples were taken. As for the second challenge, we decided to train two separate distilled models, from which one will provide the prediction and the other will estimate the output variance. We used a fully Convolutional Neural Network (CNN) architecture for this task, which consists of three main components: the encoder (3 convolutional layers with a filter count of 32, 64 and 128 respectively), the middle layers (2 convolutional layers with both having a filter count of 256 by default), and the decoder (3 convolutional layers with a filter count of 128, 64 and 32 respectively). Among these, the vast majority of the parameters are related to the middle layers, which usually also handle most of the logic contained within the model, therefore we used the filter count of these middle layers as a measure of both the models’ size and complexity. We trained our distilled models on an ensemble consisting of 25 models for every method, and achieved basically the same predictive performance as the original ensemble in every one of the three tested methods (Bayes-by-Backprop, SGLD, DREAM) with a distilled model having a middle layer filter count of 256, thus this model was selected to handle predictions. As for the output uncertainty estimation, we tested multiple different filter counts, and observed their $m(x)$ value for the predictions of the previously mentioned model. This is presented in figure 4. This shows us that a relatively accurate estimation can be achieved for the output variance of the ensembles with a filter count of 256 for the SGLD and BBB methods, and 512 for the DREAM method respectively. In addition, having a higher filter count will most likely result in overfitting, and therefore worsen the overall performance on the validation data. Considering this notion, it should also be noticed, that while the model achieved an adequate distillation for the DREAM method with a middle layer filter size of 256 for the predictive model, and 512 for the uncertainty estimator model respectively, its $m(x)$ value still falls behind of a DREAM-ensemble consisting of 3 models, which all have a middle layer filter size of 256. Therefore, while knowledge distillation is a viable option for most ensemble-based methods, in case of the CityScapes dataset and the DREAM method, it was outperformed by the original ensemble with a similar overall parameter count.

REFERENCES

- [1] C. Blundell, J. Cornebise, K. Kavukcuoglu and D. Wierstra, “Weight Uncertainty in Neural Networks,” 2015.
- [2] A. Korattikara, V. Rathod, K. Murphy and M. Welling, “Bayesian Dark Knowledge,” 2015.
- [3] E. Hüllermeier and W. Waegeman, “Aleatoric and Epistemic Uncertainty in Machine Learning: An Introduction to Concepts and Methods,” 2020.
- [4] T. Pearce, F. Leibfried, A. Brintrup, M. Zaki and A. Neely, “Uncertainty in Neural Networks: Approximately Bayesian Ensembling,” 2020.

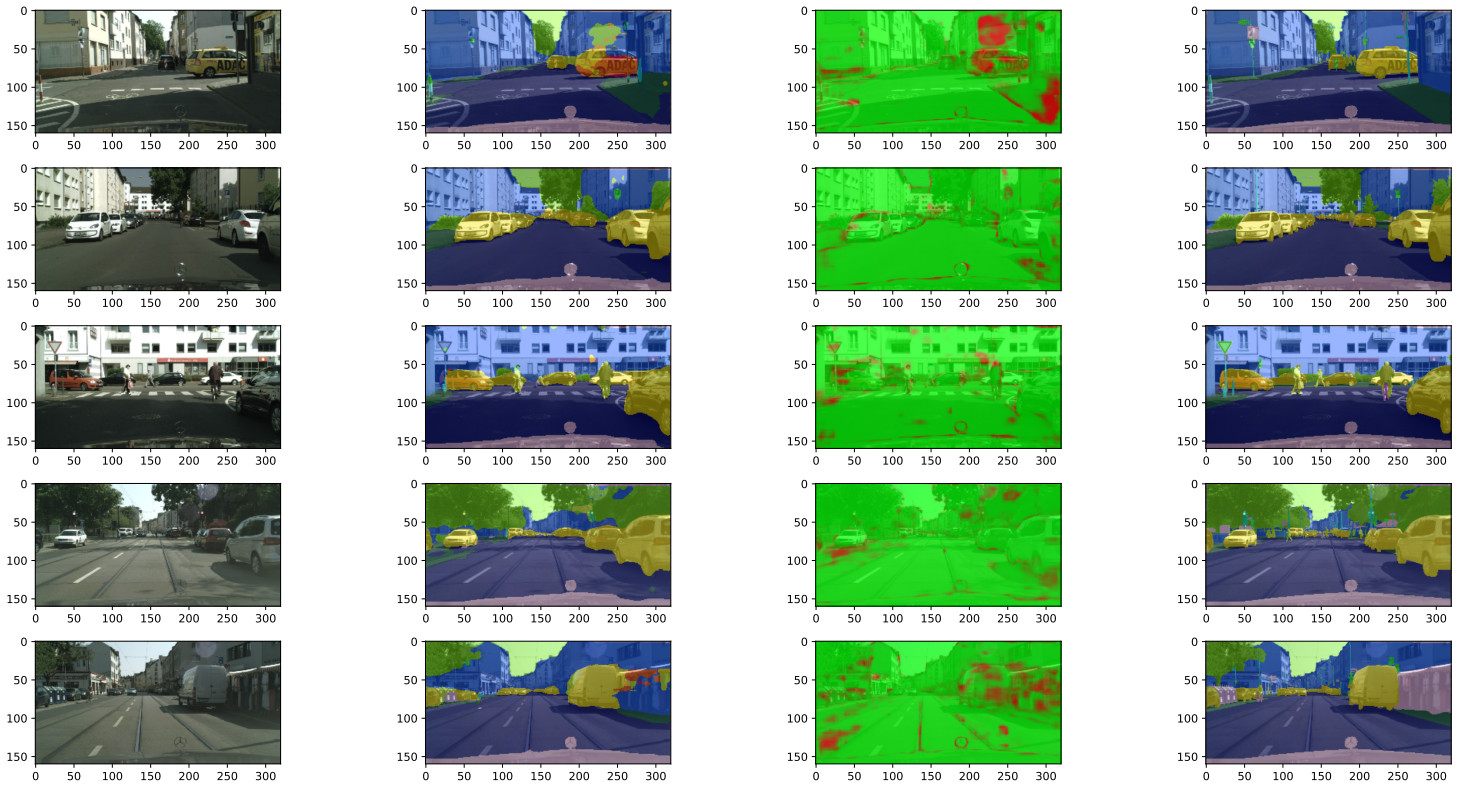


Fig. 2. Example outputs of the DREAM method applied on the CityScapes validation dataset, evaluated by an ensemble of 3 models. The columns from left to right: input, output, output variance (red: high, green: low), expected output.

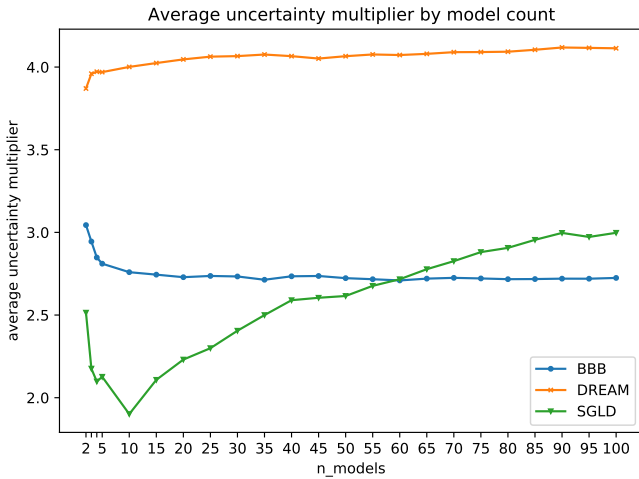


Fig. 3. The $m(x_n)$ value of three different methods by ensemble size applied on the CityScapes validation dataset.

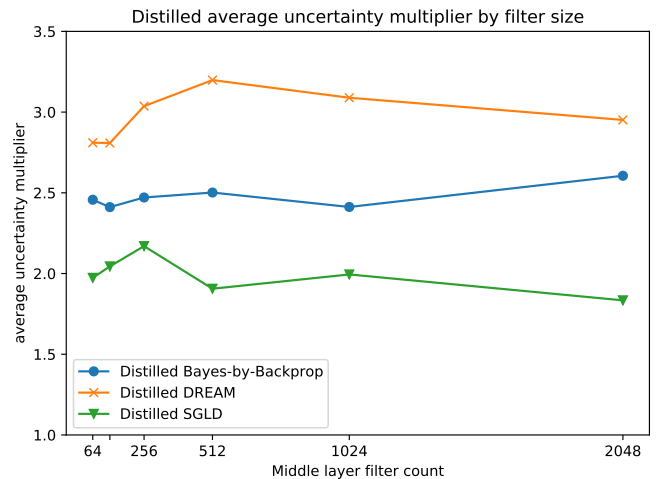


Fig. 4. The $m(x)$ value of the distilled models on three different methods by middle layer filter count on the CityScapes validation dataset, distilled from an ensemble of 25 models.

- [5] 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," 2016.
- [6] T. Jiaxi, S. Rakesh, Z. Zhe, L. Dong, S. Anima, H. C. Ed and J. Sagar, "Understanding and Improving Knowledge Distillation," 2020.
- [7] Q. Liu and D. Wang, "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm," 2019.