On Modern Deep Learning and Variational Inference

Yarin Gal **Zoubin Ghahramani**
University of Cambridge
{yg279,zg201}@cam.ac.uk

Abstract

Bayesian modelling and variational inference are rooted in Bayesian statistics, and easily benefit from the vast literature in the field. In contrast, deep learning lacks a solid mathematical grounding. Instead, empirical developments in deep learning are often justified by metaphors, evading the unexplained principles at play. It is perhaps astonishing then that most modern deep learning models can be cast as performing approximate variational inference in a Bayesian setting. This mathematically grounded result, studied in Gal and Ghahramani [1] for deep neural networks (NNs), is extended here to arbitrary deep learning models. The implications of this statement are profound: we can use the rich Bayesian statistics literature with deep learning models, explain away many of the curiosities with these, combine results from deep learning into Bayesian modelling, and much more. We demonstrate the practical impact of the framework with image classification by combining Bayesian and deep learning techniques, obtaining new state-of-the-art results, and survey open problems to research. These stand at the forefront of a new and exciting field combining modern deep learning and Bayesian techniques.

1 Introduction

Bayesian modelling and deep learning are fairly antipodal to each other: one pushed forward by theoreticians, while the other by practitioners. Bayesian modelling is based on the vast theory of Bayesian statistics, in which we aim to capture the processes assumed to have generated our data. This often results in interpretable models that can explain the data well, at least when we can perform inference in the models. Deep learning on the other hand is mostly driven by pragmatic developments of tractable models, and has fundamentally affected the way machine learning is used in real-world applications. But unlike Bayesian modelling, deep learning lacks a solid mathematical formalism, and many developments are void of mathematical justification. These are often explained by various metaphors which do not shed much light on the reasons models are built in certain ways.

Consider the following simple example. We are given a dataset of pairs of inputs $\mathbf{x}$ and $\mathbf{y}$, with corresponding outputs $\mathbf{z}$. When should a neural network (NN) be of the form $\mathbf{z} = \mathbf{W}'\sigma(\mathbf{W}_1\mathbf{x} + \mathbf{W}_2\mathbf{y})$ and when should it be of the form $\mathbf{z} = \mathbf{W}'_1\sigma(\mathbf{W}_1\mathbf{x}) + \mathbf{W}'_2\sigma(\mathbf{W}_2\mathbf{y})$? here the $\mathbf{W}$'s are weight matrices and σ is an element-wise non-linear function. In current research the problem of model architecture selection is often solved empirically by a process of trial-and-error.

A much more interesting example is that of stochastic regularisation techniques such as dropout [2], DropConnect [3], Multiplicative Gaussian Noise [4], and many others. In dropout for example the network's units would be multiplied by Bernoulli random variables. This slows down training but circumvents over-fitting and improves model accuracy considerably. Such techniques have had tremendous success in deep learning and are used in almost all modern models [4]. But why do these work so well? In [4] dropout is suggested to work well following a sexual reproduction metaphor. But then why would multiplying a network's units by a Gaussian distribution $\mathcal{N}(1, 1)$ instead of Bernoulli random variables result in the same model performance?[1]

[1] Papers such as [5, 6] attempt to explain dropout as performing stochastic gradient descent on a regularised error function, or as an L_2 regulariser applied after scaling the features by some estimate; these do not explain the multitude of other stochastic regularisation techniques though.

Many other questions are often given metaphorical answers such as these. Why is dropout not used with convolutions in convolutional networks (convnets)? Convnets are popular models for image processing, but in these dropout is used with the deep layers alone, and not with the convolution layers. Some justify this with the claim that there is less co-adaptation in convolution layers, thus dropout is not needed. Other explanations simply state that performing dropout in convolutions results in bad model performance. These arguments do not give any insights into when the technique should be used and how.

Perhaps surprisingly, we can answer all the questions above using Bayesian statistics and variational inference. Gal and Ghahramani [1] have recently shown that dropout in deep NNs can be cast as a variational approximation to a well known Bayesian model – the deep Gaussian process (GP). Extending on the work we show here that stochastic regularisation techniques in arbitrary neural models can be seen as approximate variational inference in Bayesian NNs. The implications of this result are far-reaching. Since most modern deep learning tools make use of some form of stochastic regularisation, this means that most of modern deep learning performs approximate Bayesian inference, capturing the stochastic processes underlying the observed data. This means that we can use the vast literature of Bayesian statistics with deep learning, explaining many deep learning phenomena with a mathematically rigorous theory, and extend existing tools in a principled way. We can use variational inference in deep learning, combining deep learning tools and Bayesian models in a compositional fashion. We can even assess model uncertainty in deep learning [7] and build interpretable deep learning tools.

In this paper we answer the questions brought above and propose exciting future directions to research. We first survey the results of [1] quickly and then extend these to approximate variational inference in Bayesian NNs, answering the first few questions stated above. Demonstrating the theory's impact on applied machine learning, we combine convolutional neural networks with Bayesian techniques and give new state-of-the-art results in image classification. This answers the last few open questions raised above, explaining how dropout should be used with convolution layers following our mathematically rigorous developments. The framework lays the foundations to a new and exciting field of study combining deep learning and Bayesian techniques. We finish by surveying open problems to research which would stand at its core.

2 Stochastic Regularisation Techniques in Deep Networks and the Gaussian Process

We start by reviewing the main results of [1], relating stochastic regularisation techniques (such as dropout) in deep networks to approximate inference in the Gaussian process. In the process we will be able to shed light on the first question raised above: model architecture selection. We will extend this result in the next section by tying stochastic regularisation in arbitrary network structures to approximate variational inference in Bayesian neural networks instead of Gaussian processes.

Let $\widehat{\mathbf{y}}$ be the output of a NN with L layers and a loss function $E(\cdot, \cdot)$ such as the softmax loss or the Euclidean loss (squared loss). We denote by $\mathbf{W}_i$ the NN's weight matrices of dimensions $K_i \times K_{i-1}$, and by $\mathbf{b}_i$ the bias vectors of dimensions K_i for each layer $i = 1, ..., L$. We denote by $\mathbf{y}_i$ the observed output corresponding to input $\mathbf{x}_i$ for $1 \leq i \leq N$ data points, and the input and output sets as $\mathbf{X}, \mathbf{Y}$. During NN optimisation a regularisation term is often added. We often use L_2 regularisation weighted by some weight decay λ, resulting in a minimisation objective (often referred to as cost),

$$\mathcal{L}_{\text{dropout}} := \frac{1}{N} \sum_{i=1}^{N} E(\mathbf{y}_i, \widehat{\mathbf{y}}_i) + \lambda \sum_{i=1}^{L} \big(||\mathbf{W}_i||_2^2 + ||\mathbf{b}_i||_2^2 \big). \tag{1}$$

With dropout, we sample Bernoulli random variables for every input point and for every network unit in each layer. Each variable takes value 1 with probability p_i for layer i. A unit is dropped (i.e. its value is set to zero) for a given input if its corresponding binary variable takes value 0. We use the same binary variable values in the backward pass propagating the derivatives to the parameters. Other forms of stochastic regularisation correspond to alternative procedures. With Multiplicative Gaussian Noise for example we would multiply each unit by $\mathcal{N}(1, 1)$.

Compared to the non-probabilistic NN, the Gaussian process (GP) is a powerful tool in statistics that allows us to model distributions over functions [8]. Given training inputs $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and their corresponding outputs $\{\mathbf{y}_1, \dots, \mathbf{y}_N\}$, we would like to estimate a function $\mathbf{y} = \mathbf{f}(\mathbf{x})$ that is likely to have generated our observations. For classification with D classes we place a joint Gaussian

distribution over all function values $\mathbf{F} = [\mathbf{f}_1, ..., \mathbf{f}_N]$ with $\mathbf{f}_n = [f_{n1}, ..., f_{nD}]$ and $f_{nd} = f_d(\mathbf{x}_n)$, and sample from a categorical distribution with probabilities given by passing $\mathbf{F}$ through an element-wise softmax:

$$\mathbf{F} \mid \mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}(\mathbf{X}, \mathbf{X})) \tag{2}$$

$$y_n \mid \mathbf{f}_n \sim \text{Categorical}\left(\exp(\mathbf{f}_n) / \left(\sum_{d'} \exp(f_{nd'})\right)\right)$$

for $n = 1, ..., N$ with observed class label y_n, and a covariance function $\mathbf{K}(\mathbf{X}_1, \mathbf{X}_2)$.

Our predictive probability

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{f}^*) p(\mathbf{f}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) \mathrm{d}\mathbf{f}^* \tag{3}$$

is intractable for our model. To approximate it we could condition the model on a finite set of random variables $\boldsymbol{\omega}$. We make a modelling assumption and assume that the model depends on these variables alone, making them into sufficient statistics in our approximate model.

The predictive distribution for a new input point $\mathbf{x}^*$ is then given by

$$p(y^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(y^*|\mathbf{f}^*) p(\mathbf{f}^*|\mathbf{x}^*, \boldsymbol{\omega}) p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y}) \, \mathrm{d}\mathbf{f}^* \mathrm{d}\boldsymbol{\omega}.$$

The distribution $p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y})$ cannot usually be evaluated analytically as well. Instead we define an approximating *variational* distribution $q(\boldsymbol{\omega})$, whose structure is easy to evaluate.

We would like our approximating distribution to be as close as possible to the posterior distribution obtained from the full Gaussian process. We thus minimise the Kullback–Leibler (KL) divergence: $\mathrm{KL}(q(\boldsymbol{\omega}) \,||\, p(\boldsymbol{\omega}|\mathbf{X}, \mathbf{Y}))$, resulting in the approximate predictive distribution

$$q(y^*|\mathbf{x}^*) = \int p(y^*|\mathbf{f}^*) p(\mathbf{f}^*|\mathbf{x}^*, \boldsymbol{\omega}) q(\boldsymbol{\omega}) \mathrm{d}\mathbf{f}^* \mathrm{d}\boldsymbol{\omega}. \tag{4}$$

Minimising the Kullback–Leibler divergence is equivalent to maximising the *log evidence lower bound*,

$$\mathcal{L}_{\text{VI}} := \int q(\boldsymbol{\omega}) p(\mathbf{F}|\mathbf{X}, \boldsymbol{\omega}) \log p(\mathbf{Y}|\mathbf{F}) \mathrm{d}\mathbf{F} \mathrm{d}\boldsymbol{\omega} - \mathrm{KL}(q(\boldsymbol{\omega})||p(\boldsymbol{\omega})) \tag{5}$$

with respect to the variational parameters defining $q(\boldsymbol{\omega})$.

Gal and Turner [9] have shown that the Gaussian process can be approximated by defining $\boldsymbol{\omega} = \{\widehat{\mathbf{M}}_1, \widehat{\mathbf{M}}_2\}$ to be an approximating distribution over the spectral frequencies and their coefficients in a Fourier decomposition of our function:

$$\mathbf{f} \mid \mathbf{x}, \boldsymbol{\omega} \sim \sqrt{\frac{1}{K}} \widehat{\mathbf{M}}_2 \sigma\big(\widehat{\mathbf{M}}_1 \mathbf{x} + \mathbf{m}\big)$$

with $\sigma(\cdot)$ determined by the covariance function $\mathbf{K}(\cdot, \cdot)$. Gal and Ghahramani [1] have shown that (5) results in dropout's objective when approximating the integral with Monte Carlo integration with a single sample[2] $\widehat{\boldsymbol{\omega}} \sim q(\boldsymbol{\omega})$ and using approximating distribution $q(\boldsymbol{\omega})$ of the form

$$\boldsymbol{\omega} = \{\widehat{\mathbf{M}}_i\}_{i=1}^L \tag{6}$$

$$\widehat{\mathbf{M}}_i = \mathbf{M}_i \cdot \text{diag}([\mathbf{z}_{i,j}]_{j=1}^{K_i}) \tag{7}$$

$$\mathbf{z}_{i,j} \sim \text{Bernoulli}(p_i) \text{ for } i = 1, ..., L, \; j = 1, ..., K_{i-1} \tag{8}$$

given some probabilities p_i and matrices $\mathbf{M}_i$ being variational parameters (with dimensions $K_i \times K_{i-1}$). The $\text{diag}(\cdot)$ operator maps vectors to diagonal matrices whose diagonals are the elements of the vectors. The binary variable $\mathbf{z}_{i,j} = 0$ corresponds to unit j in layer $i - 1$ being dropped out as an input to the i'th layer. This results in an identical model structure and optimisation objective to (1), in effect resulting in the same model parameters that best explain the data. Other stochastic regularisation techniques are obtained for alternative choices of $q(\boldsymbol{\omega})$.

The developments above shed some light on the problem of network structure design. Each GP covariance function has a one-to-one correspondence with the combination of both NN non-linearities and weight regularisation. So given a dataset of pairs of inputs $(\mathbf{x}, \mathbf{y})$, and outputs $\mathbf{z}$, a NN of the form $\mathbf{z} = \mathbf{W}' \sigma(\mathbf{W}_1 \mathbf{x} + \mathbf{W}_2 \mathbf{y}) = \mathbf{W}' \sigma([\mathbf{W}_1; \mathbf{W}_2]^T [\mathbf{x}; \mathbf{y}])$ would correspond to a GP covariance

[2] Using stochastic optimisation the new noisy objective would converge to the same optima as (5).

that concatenates both inputs. We would have that two outputs $\mathbf{z}_1, \mathbf{z}_2$ (with corresponding inputs $(\mathbf{x}_1, \mathbf{y}_1)$ and $(\mathbf{x}_2, \mathbf{y}_2)$) are very *similar* when both $\mathbf{x}_1$ is similar to $\mathbf{x}_2$ and $\mathbf{y}_1$ is similar to $\mathbf{y}_2$. On the other hand, the form $\mathbf{z} = \mathbf{W}_1'\sigma(\mathbf{W}_1\mathbf{x}) + \mathbf{W}_2'\sigma(\mathbf{W}_2\mathbf{y})$ corresponds to a sum of two covariance functions (and, in fact, simply a sum of two functions). In this case two outputs $\mathbf{z}_1, \mathbf{z}_2$ would be *dissimilar* when both $\mathbf{x}_1$ is dissimilar to $\mathbf{x}_2$ and $\mathbf{y}_1$ is dissimilar to $\mathbf{y}_2$. Input similarity here is defined simply by the covariance value on the pair of inputs.

3 Stochastic Regularisation in Arbitrary Networks as Approximate Variational Inference in Bayesian Neural Networks

We now link the derivation above to variational inference in Bayesian NNs, extending on [1] to arbitrary network structures. The derivation gives us tools to answer the other questions discussed in the introduction – the principles underlying stochastic regularisation techniques. In the next section we will give a concrete example of the framework demonstrating its practical impact in image classification with a Bayesian convnet. These do not necessarily have a corresponding GP interpretation, but can be modelled with the following Bayesian NN interpretation.

One defines a Bayesian NN by placing a prior distribution over the weights of an arbitrary NN. Given weight matrices $\mathbf{W}_i$ and bias vectors $\mathbf{b}_i$ for layer i, we often place standard matrix Gaussian prior distributions over the weight matrices, $p(\mathbf{W}_i)$:

$$\mathbf{W}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

We assume a point estimate for the bias vectors for simplicity. Denote the random output of a NN with weight random variables $(\mathbf{W}_i)_{i=1}^L$ on input $\mathbf{x}$ by $\widehat{\mathbf{f}}\big(\mathbf{x}, (\mathbf{W}_i)_{i=1}^L\big)$. We assume a softmax likelihood given the NN's weights:

$$p\big(y|\mathbf{x}, (\mathbf{W}_i)_{i=1}^L\big) = \text{Categorical}\left(\exp(\widehat{\mathbf{f}}) / \sum_{d'} \exp(\widehat{f}_{d'})\right)$$

with $\widehat{\mathbf{f}} = \widehat{\mathbf{f}}\big(\mathbf{x}, (\mathbf{W}_i)_{i=1}^L\big)$ a random variable.

We are interested in finding the most probable weights that have generated our data – the posterior over the weights given our observables $\mathbf{X}, \mathbf{Y}$: $p\big((\mathbf{W}_i)|\mathbf{X}, \mathbf{Y}\big)$ (we write $(\mathbf{W}_i)$ to denote $(\mathbf{W}_i)_{i=1}^L$ for brevity). This posterior is not tractable in general, and we use variational inference to approximate it as was done in [10, 11, 12, 13]. We need to define an approximating variational distribution $q\big((\mathbf{W}_i)\big)$, and then minimise the KL divergence between the approximating distribution and the full posterior:

$$\begin{aligned}&\text{KL}\big(q\big((\mathbf{W}_i)\big) \,||\, p\big((\mathbf{W}_i)|\mathbf{X}, \mathbf{Y}\big)\big) \propto \\ &\quad - \int q\big((\mathbf{W}_i)\big) \log p\big(\mathbf{Y}|\mathbf{X}, (\mathbf{W}_i)\big) + \text{KL}\big(q\big((\mathbf{W}_i)\big) \,||\, p\big((\mathbf{W}_i)\big)\big). \end{aligned} \tag{9}$$

We define our approximating variational distribution $q(\mathbf{W}_i)$ for every layer i as

$$\mathbf{W}_i = \mathbf{M}_i \cdot \text{diag}([\mathbf{z}_{i,j}]_{j=1}^{K_i}) \tag{10}$$

$$\mathbf{z}_{i,j} \sim \text{Bernoulli}(p_i) \text{ for } i = 1, ..., L,\ j = 1, ..., K_{i-1} \tag{11}$$

with $\mathbf{z}_{i,j}$ Bernoulli distributed random variables and variational parameters $\mathbf{M}_i$. Approximating eq. (9) with Monte Carlo integration over $\mathbf{z}_i$ we can continue the derivation following [1] recovering the dropout model where dropout is performed before every layer $\mathbf{W}_i$. Defining $\mathbf{z}_{i,j} \sim \mathcal{N}(1, 1)$ instead of Bernoulli we recover Multiplicative Gaussian Noise. Approximating the last term in eq. (9) results in L_2 regularisation over the weight matrices [7, appendix, section A].

This derivation sheds light on the second question raised in the introduction – the reasons dropout and its stochastic regularisation variants work so well. The results above suggest that dropout works well because it approximately integrates over the model's parameters. It also explains why Multiplicative Gaussian Noise works well – it simply corresponds to an alternative approximating variational distribution.

Dropout and its stochastic regularisation variants are often assessed by setting the weight matrices to their mean at test time. Combining Bayesian techniques and deep learning however, the derivation directs us towards a different testing technique. Predictions in the Bayesian model follow equation (4) replacing the posterior $p\big((\mathbf{W}_i)|\mathbf{X}, \mathbf{Y}\big)$ with the approximate posterior $q\big((\mathbf{W}_i)\big)$. We can

approximate the integral with Monte Carlo integration:

$$p(y^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) \approx \int p(y^*|\mathbf{x}^*, (\mathbf{W}_i)) q\big((\mathbf{W}_i)\big) \approx \frac{1}{T} \sum_{t=1}^{T} p(y^*|\mathbf{x}^*, (\mathbf{W}_i)_t) \tag{12}$$

with $(\mathbf{W}_i)_t \sim q\big((\mathbf{W}_i)\big)$. We refer to this as MC dropout when $q(\cdot)$ is defined with Bernoulli random variables. This equation has been suggested before as model averaging in the discrete case of Bernoulli random variables [4, section 7.5]. It was suggested that setting the random weight matrices to their mean approximates this quantity following the empirical observation that performance does not suffer much. We will next use our results to answer the last question raised in the introduction – how to use stochastic regularisation with convolutions.

4 Example Application: Bayesian Convolutional Neural Networks

Convolutional neural networks (convnets) have become an immensely popular image processing tool. These work extremely well for tasks ranging from classification and object recognition to segmentation [14]. These tools are built of several convolution layers (which preserve spatial information in the image) followed by inner-product layers. Stochastic regularisation techniques (dropout in particular) are often used with these tools, but only with the inner-product layers at the end of the model. Dropout has never been used in convnets with convolution layers, perhaps because empirical results with the standard dropout approximation suggested deteriorated performance (as can also be seen in our experiments in the appendix). Standard dropout approximates model output during test time by weight averaging. Instead, using the Bayesian methodology we evaluate the predictive mean – and average stochastic forward passes through the model (eq. (12)). Following the Bayesian interpretation of the model we obtained a significant improvement in classification accuracy, and managed to demonstrate new state-of-the-art results on the CIFAR-10 dataset. Full experiment details are given in the appendix. It is interesting to note that Srivastava et al. [4, section 7.5] gave empirical results suggesting that the standard dropout approximation is equivalent to MC dropout, and it seems that most research has followed this approximation. The results we have shown suggest the contrary however.

5 Where Next?

The framework above lays the foundations of a new and exciting field of study combining modern deep learning and Bayesian techniques in a practical way. Many open problems arise within it, and here we survey directions for future research.

1. **Deep learning can be explained by a mathematically rigorous theory.** We already saw examples of this above – explaining the theory behind the performance of dropout and its variants. The vast literature of Bayesian statistics contains many tools that can be used to analyse deep learning models, and Gaussian processes form only a small part of it.
2. **Deep learning can be extended in a principled way.** Understanding the underlying principles leading to good models allows us to improve upon them. For example, alternative approximating distributions to the ones discussed above would translate to new stochastic regularisation techniques. These can range from simple distributions to complex ones, tying different parameters to force correlations between the weights. Model compression can be achieved by forcing weights to take values from a discrete distribution over a continuous base measure of "hyper-weights" for example.
3. **Deep learning uncertainty.** Initial research in [7] assessed the performance of dropout in terms of the predictive mean and variance. Even though the Bernoulli approximating distribution is a crude one, the model outperformed its equivalents in the field. But different non-linearity–regularisation combinations correspond to different Gaussian process covariance functions, and these have different characteristics in terms of the predictive uncertainty. Understanding the behaviour of different model structures and the resulting predictive mean and variance are of crucial importance to practitioners making use of dropout's uncertainty.
4. **Deep learning can make use of Bayesian models.** A much more interesting application of the theory above is the combination of techniques from the two fields: deep learning and Bayesian modelling. Bayesian models, often used in data analysis, strive to describe data in an interpretable way – a property that most deep learning models lack. Using the theory above we can combine deep learning with interpretable Bayesian models and build

hybrid models that draw from the best both worlds have to offer. For example, in the fields of computational linguistics and language processing we often look for models that can explain the linguistic phenomena underlying our data. Current deep learning methods work well modelling the data and have improved considerably on previous research – partly due to their tractability and ability to go beyond the bag-of-words assumptions. But the models are extremely opaque and have not been able to explain the linguistic principles they use. Interleaving Bayesian models with deep ones we could answer many of these open problems.

5. **Bayesian models can make use of deep learning.** The field of Bayesian modelling can benefit immensely from the simple data representations obtained from deep learning models. Sequence data, image data, high dimensional data – these are structures that traditional Bayesian techniques find difficult to handle. Many unjustified simplifying assumptions are often used with these data structures: bag-of-words assumptions, reducing the dimensionality of the data, etc. Interpreting deep learning models as Bayesian ones we can combine these easily and in a principled way. Further, models can be built in a compositional fashion by propagating derivatives, forming small building blocks that can be assembled together by non-experts.
6. **Unsupervised deep learning.** One last problem discussed here is the design of unsupervised models. Bayesian statistics lends itself naturally to data analysis and unsupervised data modelling. With the Bayesian interpretation of modern deep learning tools new horizons open and new tools become available to solve this laborious task.

6 Conclusions

We have presented new theoretical developments casting modern deep learning techniques as approximate variational inference, and demonstrated the impact of the theory through an application in image classification. The theory lays the foundations to a series of exciting new research problems, of which only a small fraction was discussed above.

References

[1] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Insights and applications. In *Deep Learning Workshop, ICML*, 2015.

[2] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[3] L Wan, M Zeiler, S Zhang, Y LeCun, and R Fergus. Regularization of neural networks using dropconnect. In *ICML-13*, 2013.

[4] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[5] Stefan Wager, Sida Wang, and Percy S Liang. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 351–359, 2013.

[6] Pierre Baldi and Peter J Sadowski. Understanding dropout. In *Advances in Neural Information Processing Systems*, pages 2814–2822, 2013.

[7] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *arXiv:1506.02142*, 2015.

[8] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006. ISBN 026218253X.

[9] Yarin Gal and Richard Turner. Improving the Gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015.

[10] Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13. ACM, 1993.

[11] David Barber and Christopher M Bishop. Ensemble learning in Bayesian neural networks. *NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES*, 168:215–238, 1998.

[12] Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pages 2348–2356, 2011.

[13] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with Bernoulli approximate variational inference. *arXiv:1506.02158*, 2015.

[16] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] Yann LeCun and Corinna Cortes. The mnist database of handwritten digits, 1998.

[18] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 1(4):7, 2009.

[19] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[20] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[21] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. *arXiv preprint arXiv:1409.5185*, 2014.

[22] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

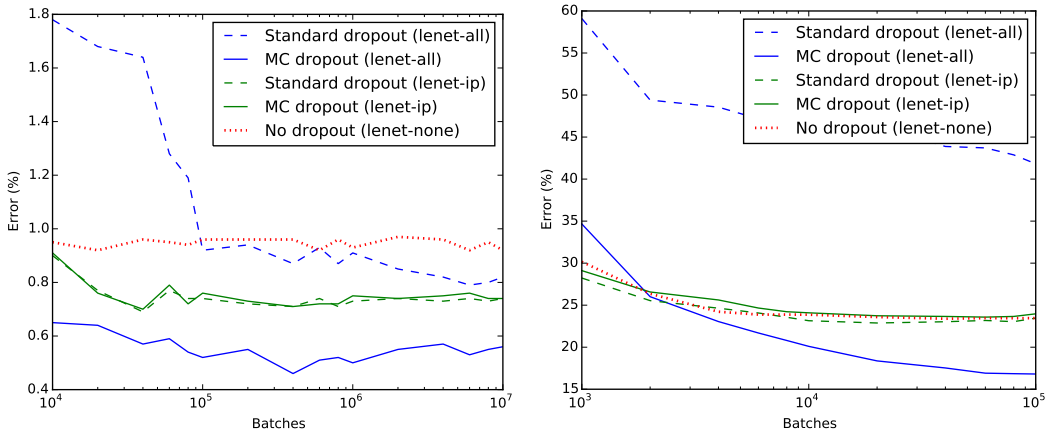
A Empirical Evaluation

We give some results assessing our Bayesian counterpart to the convolutional neural network model in image classification, with a full study of the model given in [15]. By interleaving Bayesian techniques into deep learning we attain a considerable improvement in performance for simple models, and get state-of-the-art performance with complex ones. We compare the simple LeNet network structure [16] on MNIST [17] and CIFAR-10 [18] with and without a Bayesian treatment, and test various existing complex model architectures in the literature with the added benefit of our Bayesian techniques. All experiments were done using the Caffe framework [19], requiring identical training time to that of standard convnets, with the configuration files available online at `http://mlg.eng.cam.ac.uk/yarin/`.

A.1 Bayesian Convolutional Neural Networks

We implement a Bayesian convnet by putting a distribution over the convolution layers, and approximating the posterior with variational inference. Relying on the theoretical developments above, this is equivalent to performing a stochastic regularisation technique after the convolution layers and assessing the model at test time with eq. (12). We choose dropout as our stochastic regularisation technique and assess the model by averaging stochastic forward passes through the network. On both the MNIST dataset and CIFAR-10 dataset this results in a considerable improvement in test accuracy compared to existing techniques in the literature.

We refer to the Bayesian convnet implementation with dropout used after every parameter layer as "lenet-all", and compare it to the traditional use of dropout after the fully connected inner-product layers at the end of the network alone. We refer to this model as "lenet-ip". Additionally we compare to LeNet as described originally in [16] with no dropout at all, referred to as "lenet-none". We evaluate each dropout network structure (lenet-all and lenet-ip) using two testing techniques. The first is using weight averaging, the standard way dropout is used in the literature (referred to as "Standard dropout"). This involves multiplying the weights of the i'th layer by p_i at test time. We use the Caffe reference implementation for this [19]. The second testing technique involves averaging T stochastic forward passes through the model. We do this following the Bayesian interpretation of dropout derived in eq. (12). This technique is referred to here as "MC dropout". The technique has been motivated in the literature before as model averaging, but never used with convnets. In this experiment we average $T = 50$ forward passes through the network. We stress that the purpose of this experiment is not to achieve state-of-the-art results on either dataset, but rather to compare the different models with different testing techniques. Full experiment set-up is given in [15].



(a) MNIST (b) CIFAR-10

Figure 1: **Test error for LeNet with dropout applied after every weight layer ("lenet-all" – our Bayesian treatment of the convnet model, blue), dropout applied after the fully connected layer alone ("lenet-ip", green), and without dropout ("lenet-none", dotted red line).** Standard dropout is shown with a dashed line, MC dropout is shown with a solid line. Note that although Standard dropout lenet-all performs very badly on both datasets (dashed blue line), when evaluating *the same network* with MC dropout (solid blue line) the model outperforms all others.

Krizhevsky et al. [14] and most existing convnets literature use Standard dropout after the fully-connected layers alone, equivalent to "Standard dropout lenet-ip" in our experiment. Srivastava et al. [4, section 6.1.2] use Standard dropout in all convnet layers, equivalent to "Standard dropout lenet-all" in our experiment. Srivastava et al. [4] further claim that Standard dropout results in very close results to MC dropout in normal NNs, but have not tested this claim with convnets.

Figure 1 shows classification error as a function of batches *on log scale* for all three models (lenet-all, lenet-ip, and lenet-none) with the two different testing techniques (Standard dropout and MC dropout) for MNIST (fig. 1a) and CIFAR-10 (fig. 1b). It seems that Standard dropout in lenet-ip results in improved results compared to lenet-none, with the results more pronounced on the MNIST dataset than CIFAR-10. When Standard dropout testing technique is used with our Bayesian convnet (with dropout applied after every parameter layer – lenet-all) performance suffers. However by averaging the forward passes of the network the performance of lenet-all supersedes that of all other models ("MC dropout lenet-all" in both 1a and 1b). Our results suggest that MC dropout should be carried out after all convolution layers.

A.2 MC Dropout in Standard Convolutional Neural Networks

We evaluate the use of Standard dropout compared to MC dropout (our Bayesian technique) on existing convnet models previously published in the literature. The recent state-of-the-art convnet models use dropout after fully-connected layers that are followed by other convolution layers, suggesting that improved performance could be obtained with MC dropout.

We evaluate two well known models that have achieved state-of-the-art results on CIFAR-10 in the past several years. The first is Network in network (NIN) [20]. The model was extended by [21] who added multiple loss functions after some of the layers – in effect encouraging the bottom layers to explain the data better. The new model was named a Deeply supervised network (DSN). The same idea was used in [22] to achieve state-of-the-art results on ImageNet.

We assess these models on the CIFAR-10 dataset, as well as on an augmented version of the dataset for the DSN model [21]. We replicate the experiment set-up as it appears in the original papers, and evaluate the models' test error using Standard dropot as well as using MC dropout, averaging $T = 100$ forward passes. MC dropout testing gives us a noisy estimate, with potentially different test results over different runs. We therefore repeat the experiment 5 times and report the average test error. We use the models obtained when optimisation is done (using no early stopping). We report standard deviation to see if the improvement is statistically significant.

Test error using both Standard dropout and MC dropout for the models (NIN, DSN, and Augmented-DSN on the augmented dataset) are shown in table 1. As can be seen, using MC dropout a statistically significant improvement can be obtained for all three models (NIN, DSN, and Augmented-DSN), with the largest increase for Augmented-DSN. It is also interesting to note that the lowest test error we obtained for Augmented-DSN is 7.51. Our results suggest that MC dropout should be used even with standard convnet models.

	CIFAR Test Error (and Std.)	
Model	**Standard Dropout**	**MC Dropout**
NIN	10.43	$\mathbf{10.27 \pm 0.05}$
DSN	9.37	$\mathbf{9.32 \pm 0.02}$
Augmented-DSN	7.95	$\mathbf{7.71 \pm 0.09}$

Table 1: **Test error on CIFAR-10 with the same networks evaluated using Standard dropout versus MC dropout** ($T = 100$, averaged with 5 repetitions and given with standard deviation). MC dropout achieves consistent improvement in test error compared to Standard dropout. The lowest error obtained is 7.51 for Augmented-DSN.