

# EPSRC Centre for Doctoral Training in Industrially Focused Mathematical Modelling



## Uncertainty quantification for deep neural networks

John Fitzgerald



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
	Background information . . . . .	1
	Glossary of terms . . . . .	2
<b>2</b>	<b>Methods</b>	<b>3</b>
	Compression . . . . .	3
	Sampling and Bayesian inference . . . . .	3
	Adversarial examples . . . . .	4
<b>3</b>	<b>Results</b>	<b>4</b>
<b>4</b>	<b>Discussion, conclusions &amp; recommendations</b>	<b>5</b>
<b>5</b>	<b>Potential impact</b>	<b>6</b>
	<b>References</b>	<b>6</b>



# 1 Introduction

Uncertainty is fundamental to making informed decisions, especially if we choose to automate the process. Often, a single point estimate as output may be insufficient, for example when a doctor makes a recommendation for treating a patient or a computer makes a recommendation about what to do if an object passes in front of a driverless vehicle. The doctor should know the uncertainty in the recommendation, for example if a secondary proposal may have a marginally less optimal outcome but is more likely to succeed.

Deep learning is a technique for predicting complex relationships without requiring careful case-specific fine-tuning, for example determining objects that are contained within an image, or forecasting the future from the present. The National Physical Laboratory (NPL) is the UK's national measurement institute and is involved in a large number of projects where deep learning has a key role to play, including satellite data analysis, X-ray computed tomography, analysis of mass-spectrometric data sets, medical image fusion. One of the main features of NPL's approach to metrology—the science of measurement—is a constant focus on confidence in decision making and a sound understanding of the statistical uncertainty associated with the algorithms and the measurement devices.

How can we tell how uncertain a deep neural network prediction is?

Unfortunately, metrology is an area that has not benefitted from deep learning, with most conventional models only providing a single prediction for a given input. However, deep learning also faces issues around the existence of adversarial examples, small perturbations of real data that fool the model into misclassifying the input. Recent research explores how models may be made more robust to these attacks, as well as how to detect these examples in the first place.

Our aim is to investigate how a combination of (i) model compression (a reduction in the number of parameters used) and (ii) sampling network parameters using probabilistic methods may allow us to tackle both of these issues simultaneously. We apply these techniques to a simple network designed to classify hand-written digits, and attack each network to assess both robustness and the possibility of using uncertainty as a method of detecting adversarial examples.

## Background information

The idea of artificial neural networks (ANNs) began in the 1940s when mathematicians started trying to recreate biological nerve systems [1]. The thought was that, by creating a collection of units—artificial neurons—which may be connected to each other in such a way that the processed output of one unit may be passed on to additional units, information could be processed in a manner akin to the way that synapses in the brain transform and propagate electrical signals in response to external stimuli. The connections between artificial neurons have to be trained somehow by exposure to a dataset, much as neural pathways in our brain are developed by our experiences. The resulting network may then be used to predict from, or classify, previously unseen data.

Contemporary deep learning is the latest development of this field. It utilises a series of mathematical operations to extract abstract features from a given piece of data to then allow classification or forecasting. These operations are typically sequential, with each step in the sequence being called a *layer* of the network. Within the field, *depth* refers to the use of multiple layers in order to allow greater complexity in the data to be understood without requiring enormous numbers of neurons. Originally most neural networks were fully connected, meaning that every neuron in each layer was connected to every neuron of the preceding layer. This meant that the output of every neuron in the preceding layer was passed forward for processing by each neuron of the following layer. However, this resulted in poor performance under shifts in the location (spatial translation) or rotations of objects within images, and high numbers of parameters being required to perform relatively simple tasks.

A considerable boost in the performance of these networks was achieved using *convolutional* layers. These layers have vastly fewer parameters than their fully connected counterparts, and work by applying the same basic transformation to different patches of a piece of data. This means that issues related to spatial shifts were largely overcome, and suddenly the computational expense of training a network was reduced sufficiently to allow rapid

prototyping of different network setups (architectures) by a wider variety of actors. Each convolutional layer passes a variety of these small transformations, each of which is known as a *filter*, over their input. As an input progresses through the network, it is transformed into increasingly abstract elements. In the case of image classification, early filters may learn to perform edge detection, while later filters take the transformed input and learn to recognise the structure typical of eyes.

The network 'learns' by trying to minimise a function of its prediction for a given input and the correct output over a labelled dataset. This function, summed over the given dataset, is often referred to as the *loss*. A typical graph of this loss as the network is trained is displayed in Figure 1: shown in blue is the loss over the training dataset (for which the network parameters change to try and reduce the loss), while in orange is the loss over a validation dataset. The latter is a separate dataset which is not used to train the network, but which is sampled occasionally during training to ensure that the model does not overfit the data.

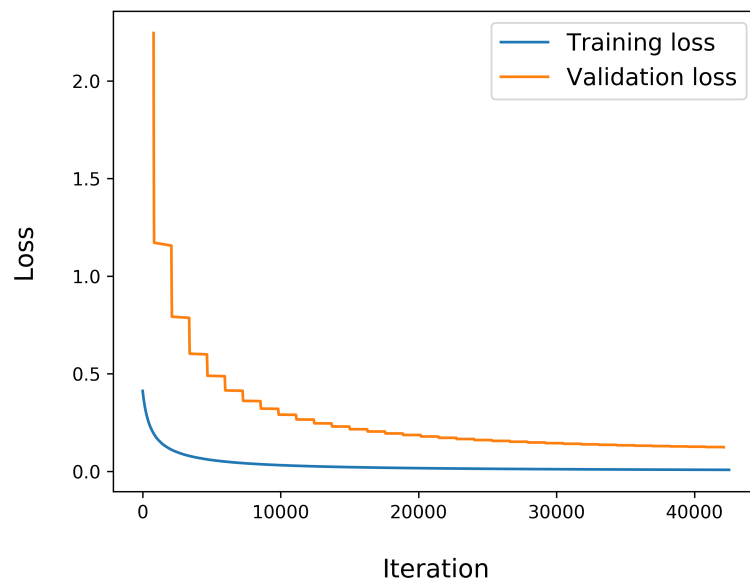


Figure 1 – Typical plot of the value of the loss function over training iterations, having discarded a small initial period

## Glossary of terms

- **Architecture:** A specific setup of the neural network
- **Filter:** A single transformation to be passed over small areas of the input
- **Layer:** A set of transformations processing an input
- **Depth:** The number of layers in a neural network
- **Compression:** A method to reduce the storage size of a network
- **Pruning:** The step of removing parameters from a network judged to be unnecessary by whatever compression technique is being used
- **Bayesian inference:** The use of Bayes' theorem to update the probability of a hypothesis as more information becomes available
- **Adversarial example:** An altered version of a piece of data designed to fool a network
- **Black box case:** A situation in which an attacker solely has access to the output of a network, rather than details about the network itself



## 2 Methods

### Compression

We focus on the pruning stage of compression, reducing the number of parameters

Modern compression techniques are typically made up of three stages: pruning, quantization, and Huffman encoding, as displayed in Figure 2. The latter two stages may be applied to any network, and as such it is typically only the pruning stage which differentiates between methods. This pruning step involves removing some of the

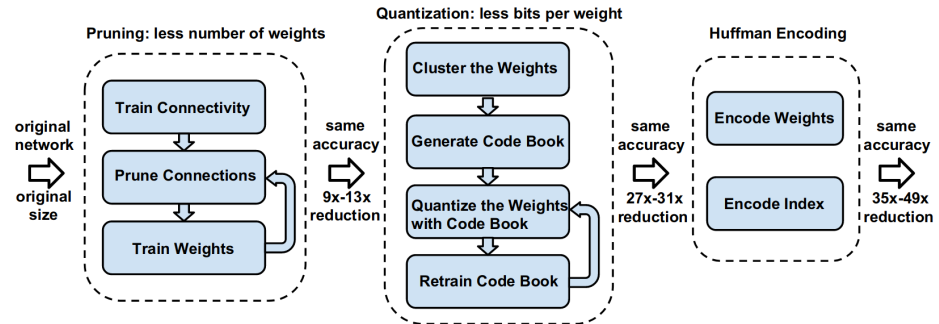


Figure 2 – Deep compression pipeline, figure taken from [2]

parameters of the network that are deemed surplus to requirements. We investigate four different pruning techniques:

- **Magnitude-based weight pruning:** This is the simplest of the four methods, in which the parameters in the network are sorted by magnitude, and a proportion of the smallest then removed. This has often been demonstrated to be the most effective way of compressing the network [3].
- **Activation-based filter pruning:** This method involves sorting the transformations (filters) learnt within convolutional layers by the mean magnitude of their output after processing (their activations) and removing a certain proportion of those with the smallest value.
- **Sensitivity driven regularisation:** This method involves pruning those weights with the smallest impact on the output of the network.
- **Sparse variational dropout:** This is the most complex method. It involves using a probabilistic version of the network during training, then pruning weights which are found to carry minimal importance.

### Sampling and Bayesian inference

Underlying our approach to uncertainty quantification is Bayes theorem, which states that the *posterior* probability of a hypothesis after some new evidence appears is proportional to the likelihood of the evidence being observed under that hypothesis, multiplied by the *prior* estimation of the probability of that hypothesis.

Bayes' theorem allows us to perform *Bayesian inference*, that is, to update our model as new data becomes available. Since the output is considered in terms of probabilities, we can consider measures of the uncertainty of our model. Due to the complexity of neural networks, and the typically poor understanding of exactly how they work, performing inference exactly is normally infeasible. Instead, we deploy approximate inference techniques. These typically rely on sampling from the posterior probability in some manner, to generate an ensemble (a group) of possible models.

We apply two different sampling techniques, called preconditioned stochastic gradient Langevin dynamics (pSGLD) and BAOAB. Both are based on an idea from physics known as Langevin dynamics, which in our situation amounts to adding some random noise to the process of training the network. In doing so, we are able to generate an ensemble of models with which we may perform approximate Bayesian inference. To quantify uncertainty, we use this ensemble to calculate the *mutual information* of a piece of data, which may be

We use sampling techniques to generate a group of possible models

thought of as a measure of how much information the different samples we have generated share—if there is a large difference between the predictions generated from the different samples, it suggests that the output of the baseline model has high uncertainty.

## Adversarial examples

As noted in the introduction, adversarial examples are simply small perturbations of real data that fool the model into misclassifying the input. A typical example is displayed in Figure 3, where to a human the difference is imperceptible but the network becomes completely mistaken with high score.

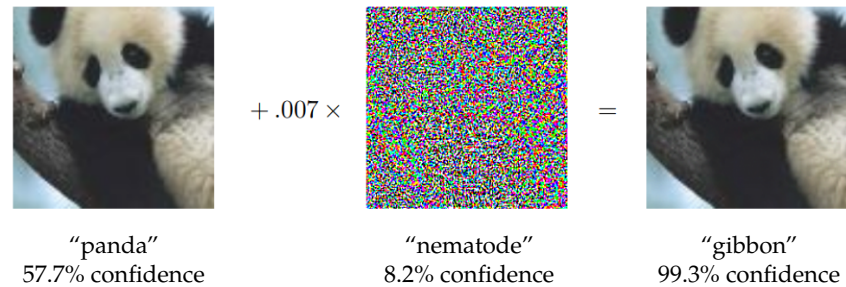


Figure 3 – An example of an adversarial perturbation, figure taken from [4]

Adversarial examples may be generated in a variety of different ways, typically depending on what kind of access to, and knowledge of, the network the attacker possesses. We consider the most generic situation, in which the only information the attacker has is what the output is for a given input: this is known as the *black-box* case. We apply a type of algorithm, called a *genetic algorithm*, to generate adversarial examples in a manner akin to natural selection [5]. We determine the success rates of the attack for networks constructed via compression by limiting the number of times we allow the attack to query the value of the output for its input. We then measure the uncertainty of our model ensembles for this data to see if this provides a method to detect adversarial examples.

## 3 Results

Compression methods seem to provide some additional robustness to attack

We implement the four different compression techniques mentioned in Section 2 along with two different sampling methods. Thus, including the original baseline network, we have five networks each with two different ensembles from which to calculate the uncertainties. We choose to develop a network to classify handwritten numerical digits 0-9, using the canonical MNIST dataset [6]. Our architecture is displayed in Figure 4—for sparse variational dropout the convolutional and fully connected layers are replaced with probabilistic counterparts. An input to the network is processed from the left (where it is simply a black and white image) to the right (where it corresponds to a score for each of the ten digits).

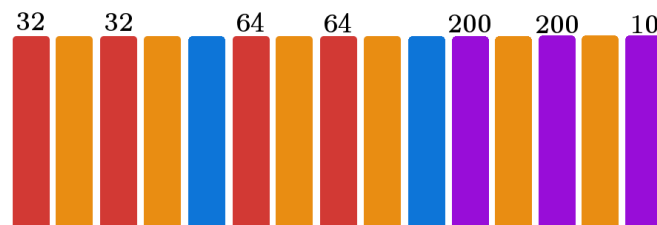


Figure 4 – MNIST neural network architecture. Convolutional layers shown red, ReLU layers in orange, max-pooling in blue, and fully connected layers in purple. Numbers correspond to number of filters for convolutional layers, or neurons (*i.e.* rows of the matrix) for fully connected layers.



We find that two of the compression methods—activation-based filter pruning and sparse variational dropout—seem to result in improved robustness, reducing the proportion of successful attacks by over 10%. However, we find that over half of all attacks succeed: we hypothesise that this may be improved by increasing the compression rates. We also find that sparse variational dropout resulted in the greatest reduction in network size, but still only to 53% of the original (*i.e.* around 2× compression).

Of the two sampling methods used, we find that BAOAB performs best when used to calculate the uncertainty of the network. Not only do misclassified examples typically have higher uncertainty than those correctly classified, but the difference between the uncertainty in adversarial examples and that for real data is considerably more dramatic.

For each of the five networks, we generate 1000 adversarial examples (see Figure 5 for some representative cases) using a genetic algorithm. These are then fed into ensembles of the model generated by each sampling technique. Two interesting results present themselves: firstly, the ensemble prediction (the most common prediction of the group of models) is actually often correct, showing the attack fails for most samples. Secondly, the mean uncertainty of the adversarial data for a specific combination of compression and sampling techniques (sensitivity based regularisation and BAOAB) is over 154000 times larger than that for unaltered test data.

Mean uncertainty for adversarial examples has been found to be up to 154000× larger than for normal data



Figure 5 – Some adversarial examples


## 4 Discussion, conclusions & recommendations

Deep learning is increasingly being applied within industry to a wide range of problems, yet the issues of quantifying the uncertainty present in the output and susceptibility to adversarial attack remain. We have examined the potential for a combination of compression and sampling to overcome these issues.

Our preliminary results suggest that this combination holds great promise, as with a negligible (or beneficial) effect on accuracy, the network appears to become more robust to attack, and we may have means to detect successful attacks. The improvement in robustness from compression was significant but not drastic—further work should test the relation between compression rate and robustness under attack. Nonetheless, our experiments have revealed multiple avenues for further study.

We observe that different compression techniques brought different benefits. In particular, the improvement in robustness from activation-based filter pruning (despite only slightly compressing the network) and sparse variational dropout (the compression technique supplying the greatest reduction in number of parameters) support the hypothesis that

Compression and inference together may improve network resilience



adversarial examples target areas of input space away from the true data manifold, meaning that pruning unnecessary parameters reduces the size of space that may be exploited to fool the network. The considerably higher uncertainty of adversarial examples for the network trained using sensitivity driven regularisation further supports this idea, as it promotes the removal of parameters that don't significantly affect the output of the network, thus improving the networks conception of the true data manifold and so increasing the specificity required of the attack.

We recommend that future work includes an investigation into how compression techniques may be combined to provide all of the respective benefits of the individual methods. This may result in high-accuracy, increased robustness, networks of comparably small size, which would further allow for greater ease of application of Bayesian inference techniques. As there are an increasing number of such inference techniques available, a thorough investigation of how they differ, and a theoretical understanding of which may be best for uncertainty quantification is certainly required. Furthermore, since only a single adversarial attack method has been considered, discovering how other methods perform against compressed networks, and how effective uncertainty may be for detecting these attacks, may also lead to a greater understanding of why these weaknesses exist in the first place.

## 5 Potential impact

Our investigation will assist NPL in implementing deep learning across a broad array of fields where uncertainty and security are crucial. We have found that compression techniques carry additional benefits beyond a reduction in computer memory requirements and computational expense, with potential improvements in accuracy and robustness to attack. Additionally we have demonstrated that contemporary sampling methods provide the means to gauge measures of uncertainty that are useful both for model confidence as well as the detection of adversarial examples.

Stephane Chretien, Senior Research Scientist at NPL said: *"The contributions by John Fitzgerald to the reliable implementation of state of the art algorithms for Deep Neural Network compression and Bayesian analysis form a key step in the development of future capabilities in the area of Uncertainty Quantification for Machine Learning at the National Physical Laboratory. The report and the codes provided during this InfoMM placement demonstrate that a sound understanding of the mathematical underpinnings of Deep Learning is the only solution to solving difficult industrial challenges in Deep Learning.*

*His work provides a clear account of the various attempts at compressing the information captured by deep networks proposed in the literature. It also addresses how compression can be a tool of choice for robustness to adversarial attacks, which is so important for Industry 4.0 and Advanced Manufacturing. The stream of research from which John's work draws is still undergoing strong changes at a fast pace, due to new theoretical insights from leading experts worldwide. Assessing the various ideas proposed in these recent papers, as remarkably performed by John, gives a better understanding of which of these approaches might have the greatest impact in the industrial context. The work completed by John will definitely become one of the main building blocks in our future reflections on this fast evolving topic and will help NPL make better choices concerning which architectures and algorithms need to be implemented in practice."*

## References

- [1] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- [2] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
- [3] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.
- [4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [5] Moustafa Alzantot, Yash Sharma, Supriyo Chakraborty, and Mani Srivastava. Genattack: Practical black-box attacks with gradient-free optimization. *arXiv preprint arXiv:1805.11090*, 2018.
- [6] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.