



UNIVERSITY OF EXETER

# Can statistics help us to understand deep learning?

Johannes Smit

May 2019

# Abstract

**To-do:** *All theses/dissertations must include an abstract of approximately 300 words bound in with each copy and placed so as to follow the title page.*

# Acknowledgements

**To-do:** *Hi mum.*

# Contents

<b>Abstract</b>	<b>i</b>
<b>Acknowledgements</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 What is machine learning? . . . . .	1
<b>2 Machine Learning</b>	<b>3</b>
2.1 Neural Networks . . . . .	3
2.1.1 The Neuron . . . . .	3
2.1.2 Neural network layers and Backpropagation . . . . .	5
2.2 Deep Learning . . . . .	6
2.3 Modern techniques/tools . . . . .	7
2.4 Example . . . . .	7
2.4.1 Training a neural network . . . . .	7
2.4.2 Ordering of the datapoints . . . . .	7
<b>3 Opening the Black Box</b>	<b>10</b>
3.1 Previous papers on this subject . . . . .	10
3.2 Regression . . . . .	10
3.2.1 Linear regression . . . . .	10
3.2.2 Stepwise regression . . . . .	11
3.2.3 LASSO . . . . .	11
3.3 Gaussian processes . . . . .	11
3.3.1 Using Gaussian process regression . . . . .	12

<b>4</b>	<b>Conclusion</b>	<b>14</b>
4.1	How well have each of the attempts worked? . . . . .	14
4.2	What could be improved? . . . . .	14
4.3	How useful would more research on this topic be? . . . . .	14
4.4	What should future research on this topic focus on? . . . . .	14
<b>A</b>	<b>Code</b>	<b>15</b>
	<b>Bibliography</b>	<b>16</b>

# Chapter 1

## Introduction

### 1.1 What is machine learning?

**To-do:** *Machine learning is very complex to human understanding  
Machine learning is widely used in many high stakes scenarios — give examples  
It is important to be able to look inside the ‘black box’ of machine learning  
— explain how this would be useful in each of the given examples  
We can use statistical methods to try and recover the information in a  
ML algorithm — give some idea how*

Since the turn of the millennium, machine learning, and particularly deep learning have been able to solve many problems that were once thought impossible for a computer. Machines can now identify objects (Li, Katz and Culler 2018), beat humans at PSPACE-hard games such as Go (Chao *et al.* 2018) and even drive cars (Gerla *et al.* 2014). These tasks are so complex that designing an algorithm by hand to tackle them would be impossible, so instead, we create a set of instructions to train a model. One of these ‘machine learning’ algorithms is the artificial neural network, which takes inspiration from the neural structure of biological brains. Artificial neural networks consist of units called ‘neurons’, which individually perform a simple nonlinear operation, but when connected, can learn complex nonlinear

patterns. Due to the abstracted nature of a machine learning algorithm's calculations, it can be difficult to provide a human understandable explanation for its reasoning, and often it is impossible. 'While an individual neuron may be understood, and clusters of neurons' general purpose vaguely grasped, the whole is beyond. Nonetheless, it works' (Grey 2017).

As these algorithms are placed in more real world scenarios in control of important infrastructure and even human lives, the ability to understand what is happening inside the 'black box' becomes more important. Autonomous vehicles are becoming more ubiquitous, which will be faced with edge cases which no human could have predicted. In cases where a driverless car does something unforeseen, it will be useful to have some understanding of what the algorithm was 'thinking' when it made the decision. Machine learning systems which aid in probation and parole decisions in the U.S. are now being trialled for sentencing, but have come under a lot of criticism for being racially biased (Christin, Rosenblat and Boyd 2015). The question of how an algorithm can be 'held accountable' has been raised (Christin, Rosenblat and Boyd 2015, p. 9), but it is not known what accountability looks like for an unintelligible artificial mind.

# Chapter 2

## Machine Learning

We are in an age where there is so much data that in many cases, hand building a model to analyse, find patterns in and draw conclusions from the data is unfeasible, and so we use computers to analyse it in a process called ‘Machine Learning’ (ML) (Murphy 2012, p. 1). Computers are able to do these tasks by combining information in nonlinear ways, but become more powerful when the output of one nonlinear process is fed into another nonlinear process, abstracting the raw data to a higher level. ‘With the composition of enough such transformations, very complex functions can be learned’ (LeCun, Bengio and Hinton 2015, p. 436).

### 2.1 Neural Networks

#### 2.1.1 The Neuron

A neuron is defined by its weight vector  $\mathbf{w}$ , its bias  $b$  and its activation function  $\phi(\cdot)$ .

The stages of a neuron, seen in Figure 2.1 are:

1. For an input vector  $\mathbf{x} \in \mathbb{R}^n$  and a weight vector  $\mathbf{w} \in \mathbb{R}^n$ , take a weighted sum of the inputs, called a ‘linear combiner’.

$$\text{linear combiner} = \sum_{i=1}^n x_i w_i$$



2. Then, a bias  $b \in \mathbb{R}$  is added, which translates the output to a suitable range. The bias controls how large the weighted sum needs to be to ‘activate’ the neuron, so this is called the pre-activation ( $v$ ).

$$\text{pre-activation} = v = \sum_{i=1}^n x_i w_i + b$$

3. Finally, an ‘activation function’ (or ‘limiter function’)  $\phi(\cdot)$  is applied, which restricts the output to a range and introduces nonlinearity to the system. The activation function must be differentiable if the gradient descent method is used (see Section 2.1.2) as gradient descent relies on calculating the gradient of the error.

$$\text{neuron output} = y = \phi \left( \sum_{i=1}^n x_i w_i + b \right)$$

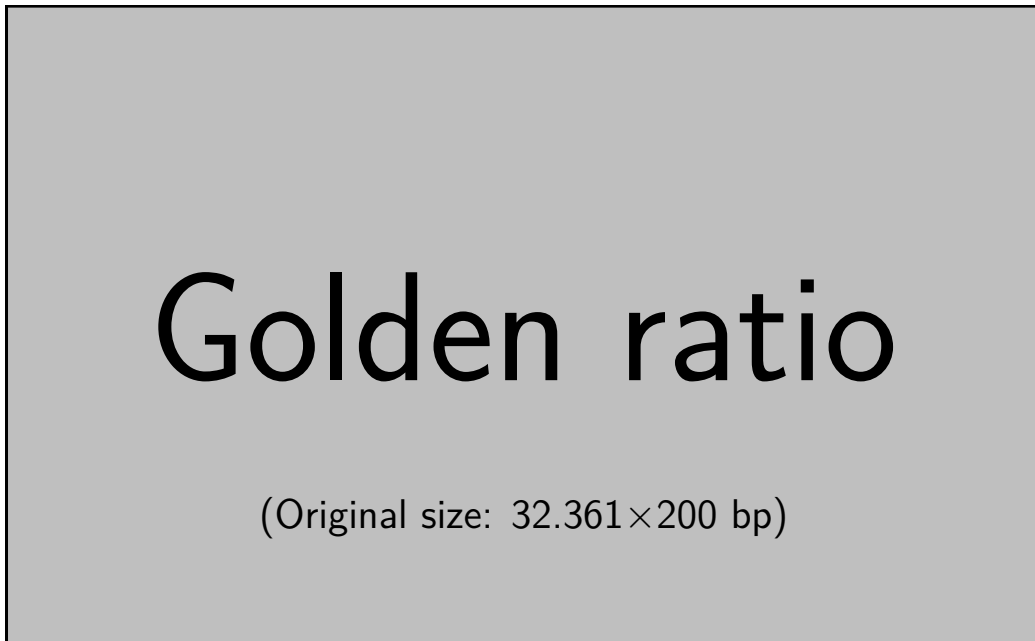


Figure 2.1: A diagram of an example artificial neuron.

## Activation functions

Early examples of artificial neurons were built in hardware with the output being a light that was either on or off. This is equivalent to the  $\text{sign}(\cdot)$  function, which is now only used for binary classification problems. It is not useful for connecting to another neuron, as information about the magnitude of the output is lost. Commonly the sigmoid function  $S(x) = (1 + \exp(-x))^{-1}$ , the  $\tanh(\cdot)$  function or the ‘softmax’ function (a generalisation of the logistic function to vectors) are used. More recently, a commonly used activation function for deep learning (see Section 2.2) is the Rectified Linear Unit (ReLU) function (Ramachandran, Zoph and Le 2017), which is defined as the positive part of its input  $\text{ReLU}(x) = \max(0, x)$ . If the activation function is the identity function, then optimising a neuron is equivalent to performing linear regression.

### 2.1.2 Neural network layers and Backpropagation

**To-do:** *Neurons are connected in layers.*

*Backpropagation is used to optimise the weights.*

*Batch backpropagation of all the datapoints takes too long.*

*Online BP is faster but takes more steps.*

*Stochastic minibatches can be used to speed up calculations.*

The weights in a neural network are optimised using the backpropagation algorithm.

**To-do:** *Fix my explanation of the backpropagation algorithm.*

*[Note: Is it possible to add a title to an enumerate?] For a neural network with  $N$  weights*

1. *Randomly initialise all the weights  $\mathbf{w} = w_1, \dots, w_N$ .*

2. Predict the outputs  $\mathbf{y}^{pred} = y_1^{pred}, \dots, y_n^{pred}$  using the training data  $X = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ .
3. Calculate the error vector  $\boldsymbol{\epsilon} = \mathbf{y}^{pred} - \mathbf{y}^{target}$ . The error only depends on the output of .
4. Calculate the gradient of the error  $\boldsymbol{\Delta} = \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{w}}$ .  $\boldsymbol{\Delta}$  represents the direction in  $N$ -dimensional space .
5. Update the weights  $\mathbf{w} \leftarrow \mathbf{w} - \eta \boldsymbol{\Delta}$ .

Estimating the parameters of an Artificial Neural Network (ANN) using either type of gradient descent is guaranteed to find a local optimum given enough time, but is not guaranteed to converge to a global optimum. It is quite rare for the algorithm to get trapped in a local minimum, but a more common problem is getting stuck at saddle points where the gradient is zero (LeCun, Bengio and Hinton 2015, p. 438).

## 2.2 Deep Learning

**To-do:** *DL is just ML with more layers*

[Note: *Rewrite this to more slowly introduce deep learning and be more generally about machine learning than ANNs*] As computing power has increased exponentially following Moore's law, machine learning has become a more viable and effective method of prediction. In the 21st century, success has been found in increasing the number of layers in a neural network or other machine learning method rather than the complexity of each layer. These 'deep neural networks' 'can implement extremely intricate functions of its inputs that are simultaneously sensitive to minute details' (LeCun, Bengio and Hinton 2015, p. 438).

## 2.3 Modern techniques/tools

Deep Learning (DL) has even made its way into consumer products. Many modern phones have facial recognition software built in, and Artificial Intelligence (AI) voice assistants are becoming more common in homes.

DL is also becoming more accessible, with the release of Google’s ‘Tensorflow’ package for DL (Abadi *et al.* 2016), which is now available as ‘Keras’, a user friendly package for Python (Chollet *et al.* 2015) and R (Allaire and Chollet 2018).

## 2.4 Example

To demonstrate the possibility of using statistical methods to understand the process of deep learning, we use a simple function  $f(x) = x + 5 \sin(x)$ . 256 evenly spread datapoints were taken from this function, and noise following  $\epsilon \sim \mathcal{N}(0, 0.1)$  was applied.

**To-do:** *Possibly also use a more complex example?*

### 2.4.1 Training a neural network

This function was learnt by a neural network with 8 layers, each with 10 neurons and the  $\tanh(\cdot)$  activation function, except for the final layer which used a linear activation function. The result of this learning is seen in Figure 2.2.

**To-do:** *Information on choosing the right size NN*

### 2.4.2 Ordering of the datapoints

Due to the stochastic batch gradient descent used to train the model, if the order of the datapoints is not randomised, then the optimisation algorithm is significantly more likely to get stuck in a local minimum. An example of this



Figure 2.2: The output of the neural network.

is seen in Figure 2.3, where the same neural network has been trained on the example data which has been shuffled, reversed, randomised and separated.

Bengio *et al.* (2009) developed a technique called ‘curriculum learning’ which uses this fact to improve training by starting with easier training examples and slowly working towards more difficult training examples.



# Golden ratio

(Original size:  $32.361 \times 200$  bp)

Figure 2.3: The output of the same neural network trained on the same data but ordered differently.

# Chapter 3

## Opening the Black Box

### 3.1 Previous papers on this subject

### 3.2 Regression

One method of opening the black box of machine learning is to use regression with many different predictors.

#### 3.2.1 Linear regression

In this case,  $x$ ,  $x^2$ ,  $\sin(x)$ ,  $\sin(2x)$ ,  $\sin(x/2)$ ,  $\cos(x)$ ,  $\cos(2x)$  and  $\cos(x/2)$  were used. This produced a complex model with many coefficients close to zero.

**To-do:** *table of linear regression fit*

**To-do:** *This is slightly cheating, but is feasible and a common technique. Find a source for this.*

### 3.2.2 Stepwise regression

It is then possible to use stepwise regression to reduce the number of parameters in this linear model. In this case, the variables were selected using the AIC.

**To-do:** *table of stepwise regression*

While the relevant estimators  $x$  and  $\sin(x)$  were identified and their coefficients fairly accurately estimated, a few other variables were also identified as significant. This method is only likely to work with a perfect or near perfect fit with no noise, which is unrealistic for real applications.

### 3.2.3 LASSO

Alternatively, we can use the Least Absolute Shrinkage and Selection Operator (LASSO) method to select the significant variables from the full model. The LASSO method constrains the sum of the absolute values of the model parameters, regularising the least influential parameters to zero. We vary the hyperparameter  $\lambda$  to change how regularised the coefficients are, as seen in Figure 3.1. We then use k-fold cross validation to find the optimal hyperparameter  $\lambda$ .

**To-do:** *table of LASSO coefficients*

## 3.3 Gaussian processes

We can use Gaussian Processes (GPs) to try and model the output of the ANN. GPs are an extension of the multivariate normal distribution to an infinite dimensional process with a mean function and covariance function instead of a mean vector and covariance matrix.





Figure 3.1: Changing the hyperparameter  $\lambda$ .

### 3.3.1 Using Gaussian process regression

Because GPs are very flexible, applying a GP to the output of the ANN is likely to result in a close fit.



# Golden ratio

(Original size:  $32.361 \times 200$  bp)

Figure 3.2: The fit of the GP.

# Chapter 4

## Conclusion

### 4.1 How well have each of the attempts worked?

**To-do:** *The stepwise regression was not as good as the LASSO technique, which worked quite well.*

### 4.2 What could be improved?

### 4.3 How useful would more research on this topic be?

### 4.4 What should future research on this topic focus on?

# Appendix A

## Code

**To-do:** *Appendix here.*

# Bibliography

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. *et al.* (2016). ‘TensorFlow: A system for large-scale machine learning’. 16, pp. 265–283. arXiv: <http://arxiv.org/abs/1605.08695v2> [cs.DC].
- Allaire, J. J. and Chollet, F. (2018). *keras: R Interface to ‘Keras’*. R package version 2.2.0. URL: <https://CRAN.R-project.org/package=keras>.
- Bengio, Y., Louradour, J., Collobert, R. and Weston, J. (2009). ‘Curriculum learning’. *Proceedings of the 26th annual international conference on machine learning*. ACM, pp. 41–48.
- Chao, X., Kou, G., Li, T. and Peng, Y. (2018). ‘Jie Ke versus AlphaGo: A ranking approach using decision making method for large-scale data with incomplete information’. *European Journal of Operational Research* 265.1, pp. 239–247. DOI: 10.1016/j.ejor.2017.07.030.
- Chollet, F. *et al.* (2015). *Keras*. <https://keras.io>.
- Christin, A., Rosenblat, A. and Boyd, D. (2015). ‘Courts and predictive algorithms’. *Data & CivilRight*.
- Gerla, M., Lee, E., Pau, G. and Lee, U. (2014). ‘Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds’. *2014 IEEE World Forum on Internet of Things (WF-IoT)*. IEEE, pp. 241–246. DOI: 10.1109/WF-IoT.2014.6803166.
- Grey, C. G. P. (2017). *How Machines Learn*. URL: <https://www.youtube.com/watch?v=R90Hn5ZF4Uo>.
- LeCun, Y., Bengio, Y. and Hinton, G. (2015). ‘Deep learning’. *Nature* 521.7553, pp. 436–444. DOI: 10.1038/nature14539.

- Li, T., Katz, R. H. and Culler, D. E. (2018). ‘ConNect: Exploring Augmented Reality Service using Image Localization and Neural Network Object Detection’.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press Ltd. 1104 pp. ISBN: 0262018020. URL: [https://www.ebook.de/de/product/19071158/kevin\\_p\\_murphy\\_machine\\_learning.html](https://www.ebook.de/de/product/19071158/kevin_p_murphy_machine_learning.html).
- Ramachandran, P., Zoph, B. and Le, Q. V. (2017). ‘Searching for Activation Functions’. arXiv: <http://arxiv.org/abs/1710.05941v2> [cs.NE].