

The Relative Neuroplasticity of Artificial and Biological Neural Network Configurations

Oliver Balfour

Alfred Deakin High School

May 18, 2018

Abstract

In this experiment the neuroplasticity of neural networks, both artificial and biological, was tested. Several configurations of computing machines, ranging from simple feed-forward Artificial Neural Networks (ANNs) to more complex Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), were tested, with the human brain as a baseline for performance. Also tested was a Support Vector Machine (SVM), a much older algorithm with a mathematical and not biologically-inspired approach to classification, to gauge how far the field of Machine Learning has come in the past decades both in terms of performance and neuroplasticity. Each machine was tested (with minor alterations to the input and output layers of each artificial model to account for differing input sizes) against several different classification and regression problems including but not limited to classifying handwritten digits (using the MNIST dataset) and performing sentiment analysis on highly polarized movie reviews (using the SAR14 IMDb review dataset). The neuroplasticity of each model was then evaluated based on the overall performance of each model in each of the problems tested.

Contents

Introduction	2
Literature Review	2
Background Information	3
Neural Networks	3
Method	4
Results	5
Discussion	5
Conclusion	5
Appendices	5
Appendix I: Implementation Details	5
Glossary	6

Introduction

The human brain is capable of rewiring itself throughout its lifetime (Draganski et al. 2004). This feature, known as neuroplasticity, is the primary force behind the process of learning, and is present in a wide range of animals. It was first discovered in rats in the 1960s (Bennett et al. 1964), and has applications as far as machine learning. In this experiment, the extent to which Artificial Neural Networks (ANNs) exhibit neuroplasticity is tested, by training the same ANNs to perform very diverse tasks.

The aim of this experiment is to determine how far behind in terms of neuroplasticity modern Artificial Neural Networks (ANNs) are when compared with the human brain. This is done by training the same artificial and biological neural network configurations to perform several different classification and regression tasks, and measuring the standard deviation of the error rates across each problem as a measure of neuroplasticity.

It is hypothesised that:

1. The human brain will marginally surpass state-of-the-art ANN configurations, and will perform exceptionally well in all problem domains.
2. The Convolutional Neural Network (CNN) will excel at digit recognition but not at classifying iris genres or performing sentiment analysis.
3. The Recurrent Neural Network (RNN) will perform well on all problems.
4. The standard densely-layered feed-forward Artificial Neural Network (ANN) will perform acceptably on all problems.
5. The Support Vector Machine (SVM) will perform well recognising handwritten digits and classifying iris genres, but will perform sub-par on all other problems.

Literature Review

Whilst there is plenty of literature on machine learning and neuroplasticity as separate topics, there is very little directly relating to both. The only publicly available scientific paper relating directly to both merely documents a method for making an ANN neuroplastic (Perwej and Parwej 2012), however its approach is essentially a synaptic pruning method for reducing overfitting in ANNs identical to the well-established method of neural network dropout (Srivastava et al. 2014). The only other paper referring to both generally is behind a journal paywall.

This investigation focuses on a neglected area of machine learning: as machine learning and artificial intelligence research is primarily focused on researching highly specialised cutting-edge techniques, investigation into the re-usability and adaptivity of neural networks has been left behind. Considering that a true artificial intelligence requires an extremely plastic brain to be capable of learning without requiring different sub-networks for each individual task it is able to perform, this investigation is important as a measure of the level of progress that the field of artificial intelligence has achieved toward an Artificial General Intelligence (AGI).

The techniques used to create neural networks in this experiment, however, are very well documented. The perceptron, an outdated artificial model of a neuron, was documented thoroughly by its creator (Rosenblatt 1958); likewise, the original Support Vector Machine algorithm was well documented (Cortes and Vapnik 1995).

Background Information

Neural Networks

A neural network is essentially a simplified computer abstraction of a brain, with primitive and computationally inexpensive representations of the core functions of neurons in a real life brain. Neural networks are organised into layers, usually at least three, which contain distinct neurons. The first layer takes binary or floating-point input from the host computer program, and the last returns binary output back to the host program. In feed-forward neural networks, the most common type of neural network, signals pass forward through neurons layer by layer without any form of recursion.

A neuron is typically represented by a series of numbers: an activation threshold (bias), and a series of numbers representing the strengths of connections with neurons in the previous layer. Given the outputs (activations) of the neurons in the previous layer (or given the input data from the host program in the case of the input layer), a neuron's output (activation) will equal the sum of all of previous layer's neurons activations multiplied by the connection strengths (weights), with the activation threshold (bias) of the neuron in question being subtracted. In a more traditional perceptron model, the output would be transformed such that if it is positive it is a binary 1, and if it is negative it is a binary 0; however more modern sigmoid models usually transform this value into a floating-point decimal between 0 and 1 using a learning function.

In this experiment, both perceptron and sigmoid neurons are tested, as well as a Support Vector Machine (SVM), which is an older algorithm designed for classification problems.

$$n_a = \begin{cases} 0 & \text{if } \sum_i M_{i_a} M_{i_w} \leq n_b \\ 1 & \text{if } \sum_i M_{i_a} M_{i_w} > n_b \end{cases} \quad (1)$$

Equation 1: Activation (output value) of a perceptron.

$$n_a = \sum_{i=0}^{|M|} (M_{i_a} M_{i_w}) - n_b \quad (2)$$

Equation 2: Activation of an artificial sigmoid neuron before applying a learning function.

In the above equations, M is the set of neurons from the previous layer, M_{i_w} is the weight (or axon-dendrite connection strength) of the i th neuron, and M_{i_a} denotes the activation (output value, a floating-point number from 0 to 1 transformed by a learning function); n is the neuron whose output is being calculated and n_b its bias (activation/firing threshold.) so the above formula boils down to adding all of the connection weights multiplied by the outputs, and then subtracting the bias. In the sigmoid model, a negative value means the neuron does not fire, as the threshold has not been reached, and a positive the opposite. The output is wrapped in a function to make the output smoother and more efficient at learning, but that is an irrelevant implementation detail. In the perceptron model, a negative value is replaced with a 0 and a positive with a 1.

The weights and biases of each neuron and neuronal connection are optimised to serve a specific purpose through the process of training. Usually, these values will start as random numbers, and through a process known as Stochastic Gradient Descent the values are slowly optimised until the neural network model is sufficiently good at its task. (Refer to the glossary on page 7 for more information on Stochastic Gradient Descent.) There are many applications for neural networks: A neural network may be trained to classify images (computer vision and optical character recognition), to estimate data such as house or stock prices given data about their respective markets, or even to control robots. This adaptivity, or neuroplasticity, is investigated in this report.

Method

There are 7 neural networks tested in this experiment, against 4 different classification and regression problems, each in different problem domains. The neural network configurations are listed below:

1. A Support Vector Machine (SVM), to show the inefficiency of traditional mathematical or logical approaches compared to biologically inspired.
2. A densely-layered feed-forward Artificial Neural Network (ANN) using perceptrons instead of sigmoid neurons.
3. A standard densely-layered feed-forward ANN, with 1 hidden layer.
4. A standard ANN like above, but with 2 hidden layers.
5. A Recurrent Neural Network (RNN), with 1 hidden layer.
6. A RNN with 2 hidden layers.
7. And as a baseline for performance and neuroplasticity, several people are used.

Each of the above models will be tested against the following classification and regression problems:

1. Handwritten digit recognition (using the MNIST dataset)¹
2. Iris flower classification (using Fisher’s dataset²)
3. Sentiment analysis (using the SAR14 dataset of IMDb movie reviews)
4. Breast cancer diagnosis (using the University of Wisconsin dataset³)

To make the experiment as fair as possible, the data was delivered to people involved in the experiment in the same way that the data was delivered to all of the artificial models: in numerical form. That way, the same part of the neocortex will be used for all of the problems, making it a fair test of neuroplasticity; and also, the visual and linguistic cortices will not be used, which would otherwise deliver an unfair advantage over the untrained artificial models used in the experiment.

The experiment happened in two phases for each combination of model (artificial or biological) and problem: the training phase, and the testing phase.

In the training phase, the artificial models are fed data along with the desired results from the respective datasets and left to their own devices. The people involved in the experiment are given the data similarly; however the data is printed and arranged in a more familiar tabular format.

In the testing phase, each model and participant is given data as in the previous stage, however the desired results are omitted, and the models and participants have to make predictions (or classifications, depending on the nature of the problem) based on inferences made from the training data. Note that participants are allowed to keep a copy of the training data for reference.

For each model-problem combination, two metrics were measured:

1. Performance, a measure of the accuracy of the model or participant in the testing phase.
2. Neuroplasticity, calculated as the standard deviation of the performance of the model or participant across all of the problems tested.

¹The MNIST dataset is available at: <http://yann.lecun.com/exdb/mnist/>

²The Iris dataset is available at: <http://archive.ics.uci.edu/ml/datasets/Iris>

³The Breast Cancer diagnosis dataset is available at: [http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic))

$$\sqrt{\frac{(\sum_{i=0}^{|P|} P_i - m)^2}{|P|}} \quad (3)$$

Equation 3: Calculation of neuroplasticity.

In the above equation, P is the set of performance metrics for a model or participant, and m is the mean (average) of the set P .

Results

Discussion

Conclusion

Appendices

Appendix I: Implementation Details

The Python 3 source code files for all of the aforementioned artificial models are available online through the Git repository hosted at following URL: <https://github.com/Tobsta/ScienceFair>

All of the deep learning models tested were implemented in the Python 3 programming language, using the Keras machine learning library with a TensorFlow backend.

Glossary

Activation function: See *Learning Function*

Artificial General Intelligence (AGI): An AGI is an intelligent agent capable of learning to accomplish many general tasks at least as well as a human, as opposed to specialising in a particular task. At this stage, AGIs are mostly a topic of science fiction and radical speculation; however many experts believe that general intelligence is mere decades away. In a survey of prominent AI researchers, the median estimation for when there is a 50% chance of AGIs having been developed was 50 years (Grace et al. 2017).

Artificial Neural Network (ANN): An interconnected network of artificial (ie. simulated) neurons that receive signals (in the form of binary or floating point numbers) as input, which pass through the connections between nodes in the network toward the output layer of neurons which emits a computed result. Simple ANNs typically contain an input layer, one or more hidden layers, and an output layer.

Classification problem: A classification problem is a type of machine learning problem in which a model is trained to classify data according to a few predefined classes. Examples include Optical Character Recognition and Computer Vision problems; the most common of which is handwritten digit recognition.

Convolutional Neural Network (CNN): A neural network in which some layers are not composed of artificial perceptrons.

Cost function: The cost function estimates the accuracy (error frequency) of a model; lower is better.

Dropout: Dropout is a technique to avoid overfitting in ANNs. It works by randomly removing nodes from the neural network. (Srivastava et al. 2014)

Feed-Forward ANN: Feed-forward ANNs are a class of ANN which have distinct layers of neurons whose signals are only fed forward. This significantly reduces the complexity of neural networks, and means that the neural network is not capable of retaining information.

Hidden Layer: Hidden layers are layers of perceptrons in an ANN that do not directly deal with input or output, but instead perform highly

complex logic through their unique combinations of neural connections (weights) and activation thresholds (biases).

Input Layer: The input layer of an ANN is the layer which doesn't take input from the previous layer, but rather from a sample in a dataset specific to the problem domain the ANN is being trained for.

K-Means: K-Means is a clustering algorithm, which attempts to find groups/clusters in a dataset with a known amount of classifications by recursively estimating the centre points (centroids) of each cluster.

Learning function: A learning function is used to transform the output value of neurons to adapt the rate at which they learn.

Machine Learning: Machine learning is a subfield of computer science that deals with artificial intelligence, pattern recognition and the creation of inference engines.

Neuroplasticity: The ability of a brain or machine learning model to adapt to different problem domains without specialisation.

One-Hot Encoding: One-hot encoding is the process of converting a class ID to an array of numbers equal to the number of classes, populated with zeros, aside from the number at the index of the class ID, which is a one. To illustrate, the classes "Cat", "Dog", and "Horse" could be represented as [1,0,0], [0,1,0], and [0,0,1] with one-hot encoding. One-hot encoding is much easier for an ANN to work with, especially in an output layer.

Output Layer: The output layer of an ANN is the layer which calculates the final result. In classification problems, each neuron in the output layer will typically map to a sample class. The result of the output layer is then compared to the one-hot encoded class of the sample in the cost function to determine the error rate.

Overfitting: If a model is overfitted, it is so fine-tuned to its training dataset that it performs poorly during validation and testing.

Perceptron: A perceptron is a common, but outdated, artificial model of a neuron which has a binary output as opposed to a floating-point one. This means that they are harder to train, as a small change in inputs creates a much larger change in output.

Recurrent Neural Network (RNN): TODO

Regression problem: A regression problem is a style of problem where a model is trained to find the relationship between input and output data. One example of a regression problem is estimating the selling price of a house given data on it (number of bedrooms, size of the block, etc.)

ReLU ReLUs (Rectified Linear Units) are a simple activation (learning) function that returns the input or 0, whichever is greater.

Sigmoid function: The sigmoid function is a learning function that creates an S-shaped curve.

Stochastic Gradient Descent (SGD): A method of minimising the error function of a neural network model (or other machine learning model) given multivariate input, in which the input parameters are viewed as axes on an n-dimensional graph with the cost function as the Y-axis. SGD attempts to find the global minimum of the cost function without calculating it for every possible combination of parameters by rolling the ball (vector representing the weights and biases of the neural network) down the hill (this analogy only really works with 2 parameters as the X and Z axes and the cost function as the Y axis, but it can be expanded to work with n dimensions.) Local minimums are avoided by the learning rate hyper-parameter, indicative of the length of the step taken (ideally toward the global minimum) in each iteration of the SGD algorithm. The direction gradient toward the local minimum is estimated by calculating the partial derivative of the cost function with respect to each parameter, and determining which parameter, plus-minus the learning rate hyper-parameter, has the lowest result of the cost function.

Softmax: Softmax is an activation (learning) function that scales all outputs from a layer from 0 to 1, such that the sum of all outputs is 1. It is typically used for the output layer of a classification neural network, as a kind of floating-point one-hot encoding method.

Support Vector Machine (SVM): SVMs are an older method of machine learning which uses... (TODO)

Bibliography

- Bennett, Edward L et al. (1964). “Chemical and anatomical plasticity of brain”. In: *Science* 146.3644, pp. 610–619.
- Cortes, Corinna and Vladimir Vapnik (1995). “Support-vector networks”. In: *Machine learning* 20.3, pp. 273–297.
- Draganski, Bogdan et al. (2004). “Neuroplasticity: changes in grey matter induced by training”. In: *Nature* 427.6972, p. 311.
- Grace, Katja et al. (2017). “When will AI exceed human performance? Evidence from AI experts”. In: *arXiv preprint arXiv:1705.08807*.
- Perwej, Yusuf and Firoj Parwej (2012). “A Neuroplasticity (Brain Plasticity) Approach to Use in Artificial Neural Network”. In: *International Journal of Scientific & Engineering Research*.
- Rosenblatt, Frank (1958). “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65.6, p. 386.
- Srivastava, Nitish et al. (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning* 15. URL: <https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf>.