The Relative Neuroplasticity of Artificial and Biological Neural Network Configurations

Oliver Balfour

May 15, 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) an Convolutional Neural Networks (CNNs), were tested, with the human brain as a control. 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. 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), classifying the genus of irises given their petal measurements (the Iris problem), and estimating house prices.

Contents

Introduction

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.

It is hypothesised that: The human brain will marginally surpass state-of-the-art ANN configurations, and will perform well in all problem domains. The Convolutional Neural Network (CNN) will excel at digit recognition but not at estimating house prices or classifying iris genuses. The Recurrent Neural Network (RNN) will perform well on all problems. The basic feed-forward Artificial Neural Network (ANN) will perform acceptably on all problems. The Support Vector Machine (SVM) will perform well on the Iris classification problem but perform badly on all other problems.

Measure the neuroplasticity of various neural network configurations by testing how well they fit to various classification and regression problems.

1. MNIST digit recognition
2. MNIST digit recognition 2

Problems: (all datasets available from UCI) - MNIST digit recognition - Iris classification (Fisher's dataset) - Possibly sentiment analysis, but include disclaimer saying it's not going to fit well to the models due to their simplicity. - Breast cancer diagnosis dataset? Could be interesting - Residential building dataset (construction/sale prices etc.)

Configurations: - Standard feed-forward artificial neural network, 1 hidden layer - Feed-forward ANN, but with 2 hidden layers - Recurrent neural network, 1 hidden layer - RNN, 2 hidden layers - Support Vector Machine (SVM), to show the inefficiency of traditional mathematical or logical approaches compared to biologically inspired - Human Brain as a control variable? Use 2/3 people as test subject

Literature Review

https://www.ijser.org/researchpaper/A-Neuroplasticity-Brain-Plasticity-Approach-to-Use-in-Artificial-Neural-Network.pdf

https://www.sciencedirect.com/science/article/pii/S0893608015002567

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.

$$n_a = \sum_{i=0}^{|M|} (M_{i_a} M_{i_w}) - n_b \tag{1}$$

Each neuron in a network has a bias/threshold associated with it, which dictates the input strength threshold at which the neuron will fire (in our brains this is a constant -55mV, however better results are found with dynamic biases on a neuron by neuron basis.) Each neuron also has a axon-to-dendrite connection to every neuron in the layers beside it, with a weight associated to each connection, to allow for highly complex logic algorithms. These weights can be 0, simulating a nonexistent connection. A neuron, given the outputs of the neurons in the previous (leftward) layer as inputs, will output:

...where m is the set of neurons from the previous layer, M_{i_w} denotes the weight (or dendrite connection strength) of the ith neuron, and M_{i_a} denotes the activation (output value, 0 to 1 depending on whether or not the neuron fired; transformed by a learning function), and 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. A negative value means the neuron does not fire, as the threshold has not beenreached, 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.

What this means, though, is that a neuron is more likely to fire if neurons with stronger connections leading into it also fire. As when creatures reproduce, some random neurons' biases and connection weights change randomly (according to a 'mutation rate'), creatures can randomly evolve intelligent behaviour over time by utilising this system and through survival of the fittest.

However, to actually utilise neural networks the first layer of neurons must have inputs from somewhere, and the last must actually be used as output. As is common practice, the first layer of a neural network is the input layer; transmitting features such as the hunger and thirst levels of animals to their brains in decimal form (0.0 to 1.0,) information about all of the nearby tiles also in numerical form, and some memory inputs saying what actions the animal chose in the past few iterations of the simulation, giving some short term memory to the animal. The last layer has one neuron for each possible action the animal can take (move forward, eat, etc.,) and serves as an output layer. The neuron with the highest activation value (output) in the output layer is selected as the action the creature takes (or in more biological terms, the muscle effector(s) it sends a signal to, if any.) Much like real world brains, a neural network is a black box that only the computer truly understands once initialised, so much care must be taken to ensure optimal results.

Glossary

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.

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.

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 calculating/estimating the centre points (centroids) of each cluster.

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 artificial model of a neuron which has an activation threshold (bias) associated with it.

Recurrent Neural Network (RNN) TODO

Sigmoid function The sigmoid function is a mathematical function that creates an S-shaped curve. Its application to machine learning is as a learning function; it is used to transform the output value of neurons to adapt the rate at which they learn.

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 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.

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

Method

Results

Discussion

Conclusion

Appendices

Appendix I: Source Code

(insert source code link)