Neural Network Handwritten Digit Analysis

Subtitle as needed ***(paper subtitle)***

Jan Olesinski (*Author*)

(of *Affiliation*): Computer Science Student

Brock University

lSt. Catharines, Canada

*Abstract*—The purpose of this paper is to analyze the performance of a neural network that classifies handwritten digits (*Abstract*)

Keywords—Neural Network; Backpropogation; QuickProp; RProp (key words)

# Introduction (*Heading 1*)

The purpose of the neural networks used in this analysis is to classify handwritten digits. The networks take an input which is an 8\*8, black and white image which has been converted into textual form. The image contains a single digit from 0 to 9. The networks then classify the data as belonging one of 10 categories. These categories are numbered 0 to 9 and the category number corresponds to the written number, e.g. if an image is classified as belonging to category 0 then it is a 0. These networks were trained on 700 examples per digit, and tested on 400 examples per digit.

This paper analyzes the performance of unsupervised neural networks using the Backpropogation, QuickProp, and RProp learning rules. The effect of hidden nodes, learning rates, and momentum rates was analyzed. Holdout and Cross-Validation techniques were applied and their effects analyzed.

This paper describes neural networks and the three learning rules stated above. All results were analyzed using statistical methods which are described in this paper.

Furthermore, this paper describes

# Neural Networks

## Feed Forward Neural Networks

A feed forward neural network is a structure through which data is fed to receive a desired output. A feed forward neural network consists of several layers. There is an input layer consists of just raw input. Following the input layer, there are number of “hidden” layers (typically one or two), and an output layer. The hidden and output layers consist of units which are known as perceptrons. A perceptron takes an input and applies an activation function to it. If the activation function returns a value which is above a certain threshold, the perceptron “fires,” outputting a response. The layers in a feed forward neural network are interconnected; thusly the outputs from one layer are fed as inputs into the next layer. Each connection has a weight associated with it. This weight modifies any input into a perceptron in the next layer. The sum of the outputs multiplied by the weights of the associated connections form the input into a perceptron. A feed forward neural network is so called because data is fed forward through the network, and reaches the output layer.

## Learning Rules

Learning in a neural network is generally accomplished by a learning rule. After a forward pass of the network, the learning rule is applied repeatedly over the network. The learning rule takes into account one or several factors (e.g. an error signal, a learning rate, a previous weight change) and then uses those factors to provide better weight values. There are 3 main models of learning when it comes to neural networks. These include supervised, unsupervised, supervised, and reinforced. In supervised learning, the training data includes pairs of input data and a desired output data. Thusly, the desired output of a network can be compared with its actual output. The weights can then be adjusted accordingly so that the actual output matches the desired output as closely as possible. Unsupervised learning happens when a network is provided with an input, but no desired output. The network must find some sort of pattern in the data on its own without any outside aid. Reinforced learning is done when an input is given, and the network performance is given a score based upon its performance. The network uses that score to adjust the network. The learning model decides what kinds of learning rules are appropriate. For example, the backpropogation learning rule is used in supervised learning, but not in unsupervised learning. This paper will focus on supervised learning and three associated learning rules.

## Backpropogation with Momentum

The backpropogation algorithm uses the method of gradient descent to compute the minimum of the error function. A solution to a particular problem is considered to be a combination of weights in the network which minimize the error function.

After a forward pass of the network, where the inputs are propagated forward through the network to achieve an output, the gradient error function with respect to the weights is calculated. This is generally done with mean squared error for a network with a single output, capable of only binary classification. However, it is possible to have a neural network capable of multi-classification. This is done by using multiple output neurons, the neuron that outputs the highest value is considered to classify the input data. In this case, cross-entropy error is more appropriate than mean-squared error. Because cross-entropy error was used in the neural network, it will be explained here. Cross entropy error is given by the formula:

E = -

*Where n = the number of outputs, t = the target value at the output of a perceptron in the output layer, and x = the actual output of a perceptron at the output layer.*

To compute the error of a weight between output and the hidden layer, we use the derivative of the error function with respect to the weight between the output and the hidden layer. The corresponding function is:

*Where xj = the output of perceptron j in the hidden layer, xi = the output of perceptron i in the output layer, and ti = the target output of perceptron I in the output layer*

Lastly, to compute the error between the hidden layer and the output layer, the following derivative of the error with respect to the weight from the input to the hidden layer is given by the formula:

Where = the error signal with respect to the weight

The error at the output layer is first calculated, then the error at the hidden layer(s). Essentially, the error is calculated from the last (output) layer to the first (input) layer. Thusly, the error at the output layer is propagated backwards through the network, hence the name backpropogation. Then, after the error signals for all weights are calculated the change in weight which is given by the formula:

*Where = the current change in weight, = the learning rate, = the momentum rate, = the error signal of the corresponding weight, and = the previous weight change*

Then the ∆w of each weight is added to its corresponding weight, resulting in the new weight.

## QuickProp

QuickProp is a variation of the Backpropogation algorithm. It is known to converge several times faster than Backpropogation.

Quickprop assumes that an approximation of the error surface can be made with a concave-up quadratic curve. Then, the neural net is adjusted to fit the minimum of this curve. QuickProp uses an algorithm inspired by Newton’s Method:

## RProp

RProp is another variant of the Backpropogation algorithm. RProp operates on the principle if the error gradient for a weight has the same sign as it did in the last epoch; the change in weight is increased. If, however, the sign of the weight is different, the change in weight is decreased. Then, if the current weight gradient is positive the change in weight is subtracted from the corresponding weight. If the current weight gradient is negative, the change in weight is added.

## Backpropogation vs. QuickProp vs Rprop

Generally, the QuickProp and Rprop algorithms converge more quickly than the Backpropogation algorithm. Furthermore, they do not rely on a learning rate and momentum rate, reducing the number of parameters than need to be tuned. A problem however, is that since there is a faster convergence, QuickProp and RProp are easier to over-train. Therfore it is a god idea to implement early stopping.

The RProp algorithm seeks to solve the tendency of Backpropogation to oscilate in a local minimum.

QuickProp is notable for converging several times faster than BackPropogation

## Holdout Early Stopping

Early stopping is a measure implemented to solve the problem of over-training. Over-training can cause a network to become overly accustomed to the training set, making it less accurate over inputs which it was not trained on. The idea behind early stopping is to stop training the network when it has been trained over the optimal number of epochs.

A popular and easy to implement early stopping method is the holdout method. The idea is to stop training the network as soon as the error rate begins to increase. This is done by splitting the training data into a training set and a validation set (usually a 70-30 split respectively). The data is trained on the training set, and every few epochs (e.g. every 5 epochs) the network is tested on the validation set. If the error rate is higher than the last time the set was validated, then the network is reverted to the state it was in at the last validation and the training stops.

## Cross-Validation

With K-Fold Cross Validation, the training set is split into k sets (generally k = 10), then a neural network is trained for each k. each neural network uses one of the 10 training set partitions as a validation set. So, for example, neural net 1 would use the first partition, neural net 2 would use the second partition, etc. The most fit, i.e. the set with the least error is chosen as the net to be used on the testing data.

In the trials, cross validation was applied using using the holdout technique. After each of the 10 sets finished training, they were tested for accuracy and the most fit was selected.

# Methods of Analysis

## Analysis

To determine the performance of the Backpropogation, QuickProp, and RProp learning rules, the primary result that will determine network performance will be the amount of errors that result from classification of the testing set. The less errors there are, the more accurate and well-trained the network is.

To determine the effect of the learning rate () and the momentum rate (), a series of values will be tested. Combinations of high and low momentum and learning rates will be tested. These rates will also be tested with both a high and low number of nodes in the hidden layer.

When it comes to convergence, it will be analyzed by early stopping. The epoch on which the training stops will be used as a point of reference to determine the average number of epochs it takes to reach convergence.

The testing data will be tested for normality, to ensure compatibility with the Student’s t and ANOVA tests. If data is not normally distributed, then it will have to be tested with the Mann-Whitney U test. If two samples with different population distributions need comparing, a 1-sample t-test or a 1-sample sign test will be used to compare one sample against the mean of the other sample.

Then the means of each trial will be compared with the t-test other trials to determine whether or not they are significantly different

## Comparison of Means

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

## t-test

The t-test is a statistical test used to determine whether there is in not a statistical difference between the two groups from which two means were sampled.

For example, if one compared the performance of three runs of the QuickProp algorithm with three runs of the Backpropogation algorithm, it could be possible that the results are not statistically significant. If, for example, the Backpropogation algorithm happened to perform unusually well out of pure luck, it would not be reflective of the actual performance of Backpropogation. Therefore, if Backpropogation outperformed QuickProp, it would be incorrect to assume that the performance of Backpropogation is a superior to that of QuickProp. The t-test is used to determine that there is enough variability in each sample to reflect the overall population, avoiding such a scenario.

The t-test uses a null-hypothesis, which states that the means of the two populations are different. If the t-test is successful, then we say that we fail to reject the null-hypothesis. This is because while the null-hypothesis may be true, the t-test does not provide evidence to prove the null hypothesis. It simply ensures that there is no evidence with which to reject the null hypothesis.

There are two versions of the t-test: the 1-sample t-test and the 2-sample t-test. The 1-sample t test is used to compare a sample mean against a set value, while the 2-sample t-test is used to compare the means of 2 samples.

A drawback of the t-test is that requires a sample to be normally distributed.

## Anderson-Darling Test for Normality

The test for normality is used to determine whether or not a sample is normally distributed.

When we test for normality we use a null hypothesis which states that the data follows a normal distribution. There are a number of normality tests available; the one chosen for this study was the Anderson-Darling test for normality.

## Confidence Intervals

A confidence interval for a sample estimates whether the population fits into some sort of category which is based upon the results of the sample.

So if, for example, if 60 out of 100 students preferred pencils to pens, it would be fairly certain that between 51% and 69% of students preferred pencils to pens.

The confidence interval can be used to obtain a population mean. In the statistical tests carried out, a 95% confidence interval was used.

## ANOVA

ANOVA is a statistical test done to determine if there is a statistical difference in the means of several populations. A One-Way ANOVA is test done on a single variable across several, separate population samples. So, for example, a One-Way ANOVA test could be done to determine if there is a difference in the average SAT scores of several states.

An ANOVA is an omnibus test, this means that it simply determines if there is a statistical difference between means of the samples, it does not determine which samples are statistically different from each other, only that at least two samples are different.

ANOVA is another test which requires normality

## 1-Sample Sign test

The 1-sample sign test is the non-parametric equivalent of the 1-sample t-test, it is used to compare the mean of a non-normally distributed sample against a set value.

## Mann-Whitney U-Test

The Mann-Whitney U test is the non-parametric equivalent of the 2-sample t-test, it is used to compare the means of two non-normally distributed samples

# Results

## Important Notes

The Neural Networks were run over 50 epochs. Alpha represents the momentum rate, Eta represents the learning rate, and HL Size represents the size of hidden layer. BP stands for Backpropogation, QP stands for QuickProp,

## Normality Tests

### BP Alpha = 0.8, Eta = 0.2, HL Size = 128



HO is accepted, the data is normally distributed

### BP Alpha = 0.8, Eta =0.2, HL Size = 32



HO is accepted, the data is normally distributed

### BP Log Alpha = 0.2, Eta = 0.8, HL Size = 128



HO is rejected, the data is not normally distributed

### BP LogAlpha = 0.2, Eta = 0.8, HL Size = 32



HO is accepted, the data is normally distributed

### BP Log Alpha = 0.5, Eta = 0.5, HL Size = 128



HO is accepted, the data is normally distributed

### BP Log Alpha = 0.5, Eta = 0.5, HL Size = 32



HO is accepted, the data is normally distributed

### BP TanH Alpha = 0.8, Eta = 0.2, HL Size = 128



HO is rejected, the data is not normally distributed

### BP TanH Alpha = 0.8, Eta = 0.2, HL Size = 32



HO is rejected, the data does not follow a normal distribution

### QP Log HL Size = 128



HO is accepted, the data is normally distributed

### QP Log HL Size = 32



HO is accepted, the data is normally distributed

### QP TanH HL Size = 128



HO is accepted, the data is normally distributed

### QP TanH HL Size = 32



HO is rejected, the data is not normally distributed

### RP Log HL Size = 128



HO is accepted, the data is normally distributed

### RP Log HL Size = 32



HO is rejected, the data is not normally distributed

### RP TanH HL Size = 128



HO is accepted, the data is normally distributed

### RP TanH HL Size = 32



HO is accepted, the data is normally distributed

### Holdout BP Log Alpha = 0.2, Eta = 0.8, HL Size = 128



HO is accepted, the data is normally distributed

### Holdout QP Log HL Size = 128



HO is accepted, the data is normally distributed

### Holdout RP Log HL Size = 128



HO is accepted, the data is normally distributed

### Cross-Validate BP Log Alpha = 0.2, Eta = 0.8, HL Size = 128



HO is accepted, the data is normally distributed

### Cross-Validate QP Log HL Size = 128



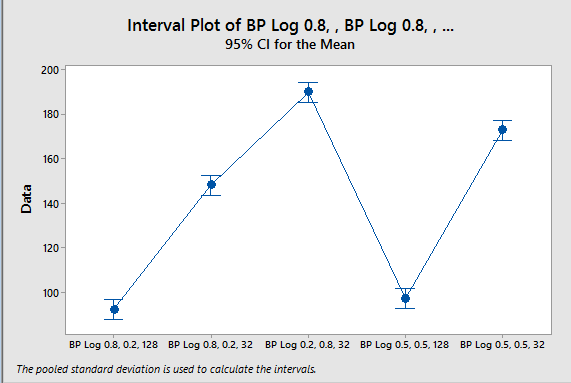
HO is accepted, the data is normally distributed

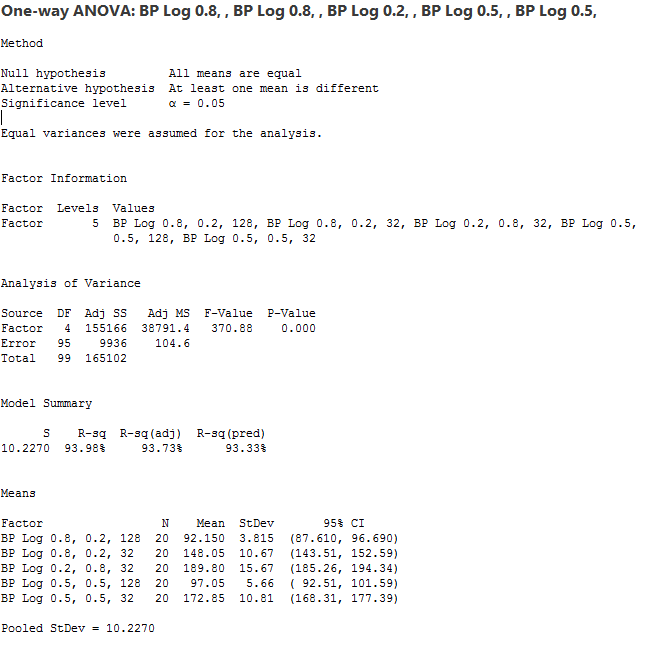
### Cross-Validate RP Log HL Size = 128



HO is accepted, the data is normally distributed

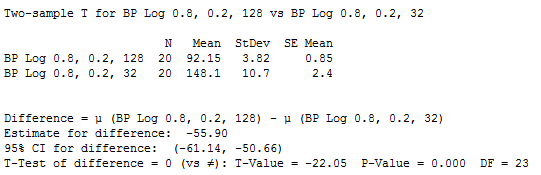
## BackPropogation: ANOVA for normally distributed values





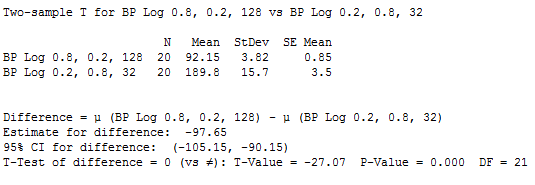
## BackPropogation: Two Sample t-Tests

### BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 vs BackProp Alpha = 0.5, Eta = 0.5, HL Size = 32

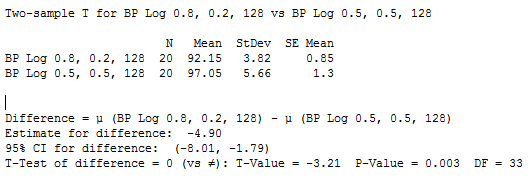


H0 is rejected, The P-Value is less than 0.01, this means that not only is there a difference in the means, but the data is extremely significant. There is sufficient evidence to say there is a difference between the means of the two samples.

### BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 vs BackProp Alpha = 0.2, Eta = 0.8, HL Size = 32

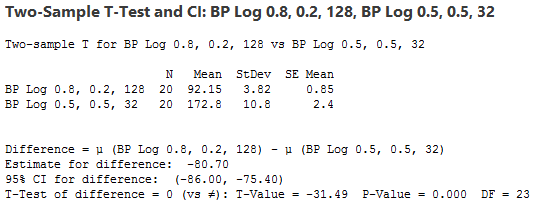


### BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 vs BackProp Alpha = 0.5, Eta = 0.5, HL Size = 128



H0 is rejeced, The P-Value is less than 0.01, this means that not only is there a difference in the means, but the data is extremely significant. There is sufficient evidence to say there is a difference between the means of the two samples

### BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 vs BackProp Alpha = 0.5, Eta = 0.5, HL Size = 32



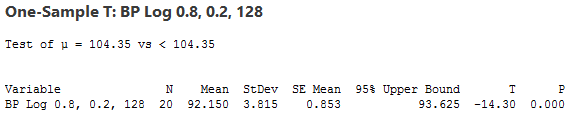
H0 is rejected, The P-Value is less than 0.01, this means that not only is there a difference in the means, but the data is extremely significant. There is sufficient evidence to say there is a difference between the means of the two samples

## Analysis of Two Sample t-Test for Logistic Backpropogation with Varied Learning Rates, Momentum Rates, and the Number of Hidden Nodes and

The network seemed to have the best performance when there was a high number of hidden nodes, a high momentum rate, and a low learning rate. This is because a high learning rate causes the error function to diverge. Too low a learning rate will cause a network to learn slowly. Although the learning rate is low a high momentum rate allows the network to start off slowly and the pick up speed. This allows for sufficient dampening of oscillations as well, allowing the network to converge faster. The network performed best when the momentum rate , the learning rate , and the number of hidden layers = 128. Therefore, the and parameters were used in the next series of tests, while the number of hidden layers was altered.

## 1-Sample t-Test

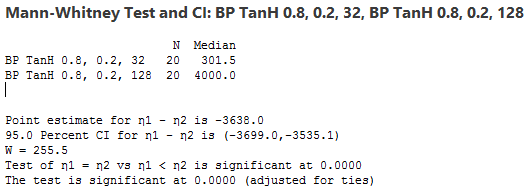
### BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 vs Mean of BackProp Alpha = 0.2, Eta = 0.8, HL Size = 128 (104.35)



HO is rejected, there is sufficient evidence to say that the mean of BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 is less than the mean of BackProp Alpha = 0.2, Eta = 0.8, HL Size = 128

## Backpropogation: Mann Whitney U-Tests

### BP TanH Alpha = 0.8, Eta = 0.2, HL Size = 32 vs BP TanH Alpha = 0.8, Eta = 0.2, HL Size = 128



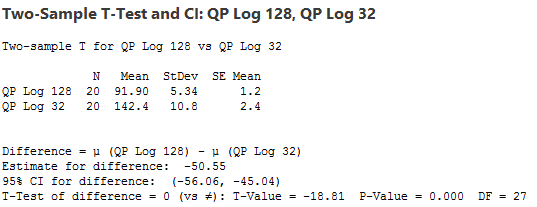
HO is rejected, with the alternative hypothesis being that the mean Backpropogation with , , and Hidden Layers = 32 is less than Backpropogation with , , and Hidden Layers = 128. Therefore, the alternate hypothesis is accepted.

## Analysis of Mann-Whitney U-Tests for Backpropogation

It was shown that when it comes to the Backpropogation algorithm, the TanH activation function performs horribly with a high number of hidden nodes. The network performed significantly better when there was a low number of hidden nodes

## QuickProp: 2-Sample t-Test

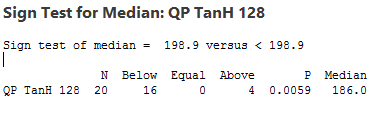
### QP Log HL Size = 128 vs QP Log HL Size = 32 (142.45)



HO is rejected, there is sufficient evidence to say that the mean of QP Log HL Size = 128 is different than the mean of QP Log HL Size = 32

## QuickProp: 1-Sample Sign Test

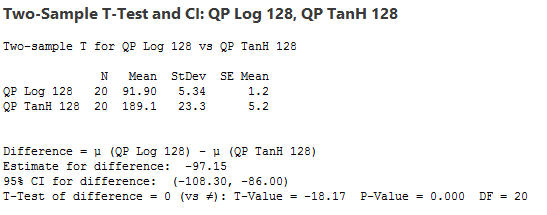
### QP TanH HL Size = 128 vs Mean of QP TanH HL Size = 32 (142.45)



HO is rejected, there is sufficient evidence to say that the mean of QP TanH HL Size = 128 is less than the mean of QP TanH HL Size = 32.

## QuickProp: 2-Sample t-Test Part 2

### QP Log HL Size = 128 vs QP TanH HL Size = 128



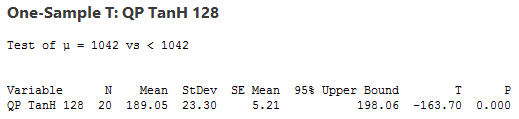
HO is rejected, there is sufficient evidence to say that the mean of QP Log HL Size = 128 is different than the mean of QP TanH HL Size = 128

## Analysis of Logistic and TanH QuickProp

It was shown that when it comes to the QuickProp, the logistic function performs best when there is a high number of hidden nodes. The number of hidden nodes has less impact when the TanH activation function is used, but the network still performs better with a high number of hidden nodes. The Logistic activation function outperformed the TanH activation function.

## RProp: 1-Sample t-Test

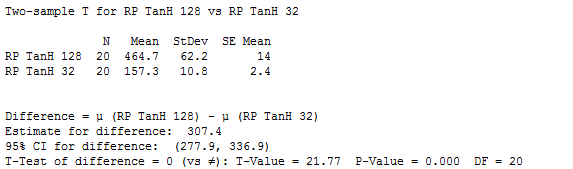
### RP Log HL Size = 128 vs Mean of RP Log HL Size = 32 (142.45)



HO is rejected, there is sufficient evidence to say that the mean of RP Log HL Size = 128 is less than the mean of QP Log HL Size = 32

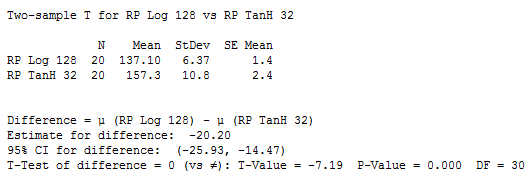
## RProp: 2-Sample t-Test

### RP TanH HL Size = 128 vs RP TanH HL Size = 32



HO is rejected, there is sufficient evidence to say that the mean of RP TanH HL Size = 32 is different than the mean of RP TanH HL Size = 128

### RP TanH HL Size = 128 vs RP TanH HL Size = 32

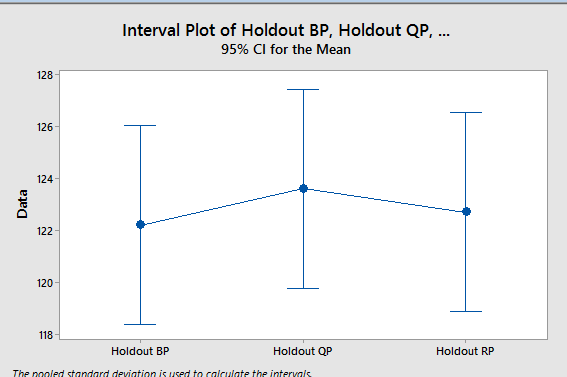


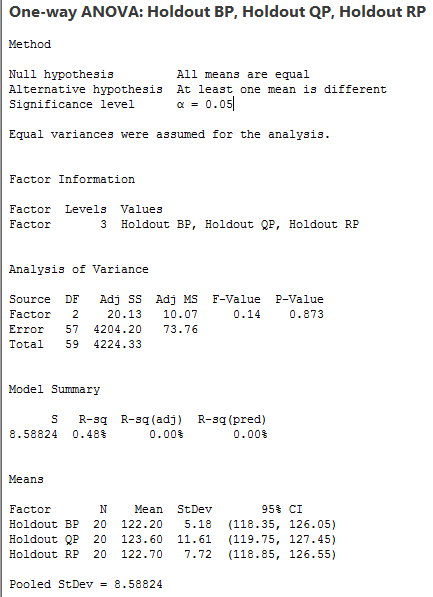
HO is rejected, there is sufficient evidence to say that the mean of RP Log HL Size = 128 is different than the mean of RP TanH HL Size = 32

## Analysis of Logistic and TanH RProp

It was shown that when it comes to the RProp, the logistic function performs best when there is a high number of hidden nodes. The number of hidden nodes has drastic impact when the TanH activation function is used. A High number of nodes results in a high error rate. When the network had a low number of hidden nodes, the performance increase three fold. The Logistic activation function outperformed the TanH activation function.

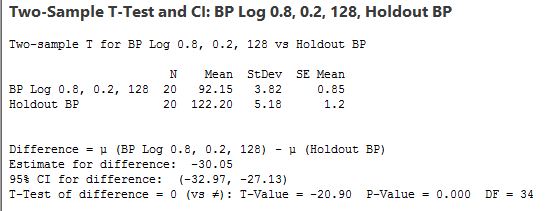
## ANOVA for the Holdout Early-Stopping Method





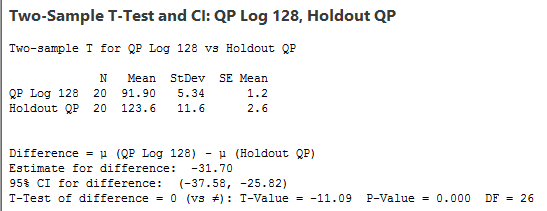
## Holdout vs. Regular: 2-Sample t-Test

### BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 vs Holdout BackProp Alpha = 0.5, Eta = 0.5, HL Size = 128



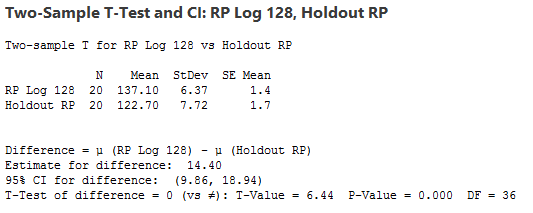
HO is rejected, there is sufficient evidence to say that the mean of BackProp Alpha = 0.8, Eta = 0.2, HL Size = 128 is different than the mean of Holdout BackProp Alpha = 0.2, Eta = 0.8, HL Size = 128

### QuickProp HL Size = 128 vs Holdout QuickProp HL Size = 128



HO is rejected, there is sufficient evidence to say that the mean of QuickProp HL Size = 128 is different than the mean of Holdout BackProp HL Size = 128

### QuickProp HL Size = 128 vs Holdout QuickProp HL Size = 128



HO is rejected, there is sufficient evidence to say that the mean of RProp HL Size = 128 is different than the mean of Holdout RProp HL Size = 128

## Holdout vs. Regular: ANOVA

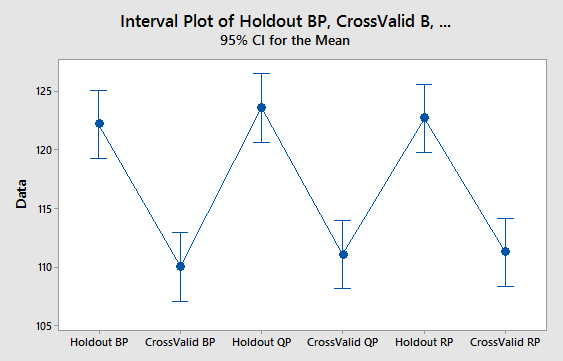
## 

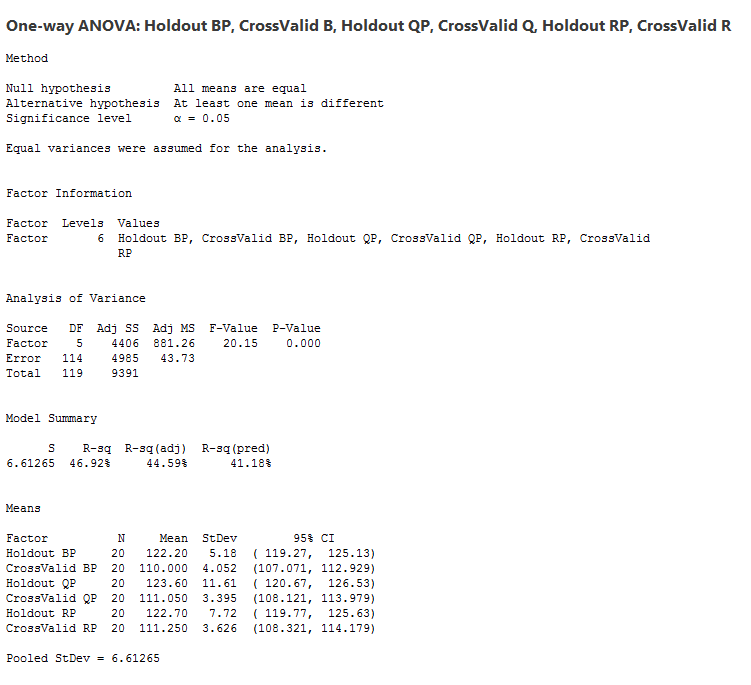
## Holdout vs. Regular: Analysis

Strangely, when it came to regular Backpropogation, the holdout method was substantially less effective. The case was the same for QuickProp. With RProp, however, the holdout method outperformed the regular algorithm.

According to the results in sections O, P, and Q; the most effective learning algorithms were regular Backpropogation and QuickProp, followed by BackPropogation Holdout, followed by RProp Holdout, followed by QuickProp HoldoutH, followed by regular RProp.

## Cross-Validate vs. Holdout: ANOVA





## Cross-Validate vs. Holdout: ANOVA

Crossvalidate outperforms holdout every time. According to the ANOVA results, it seems that Cross-Validation has a substantially positive impact on neural net performance. A drawback of Cross-Validation is that it takes a substantial amount of time to run.

# Conclusions

It was shown that, in general, a relatively low learning rate and high momentum provided the best results when using Backpropogation with momentum.

When the activation function is Logistic a high number of hidden nodes seems to generally provide better performance than a low number. The reverse was is for the TanH activation function. Generally, the Logistic function seems to outperform the TanH function.

The Holdout Method seems to be less efficient than the regular learning algorithm which runs for a set number of epochs, except in the case of RProp. This is perhaps due to the number of epochs, the regular algorithms ran over 50 epochs. If the number of epochs was higher, perhaps Holdout would be a better option.

The Cross-Validation method increased performance. It seems to minimize the amount of error by returning the most fit neural network of 10.

The most effective algorithms seem to be the regular QuickProp and the Backpropogation (with a low learning rate and a high momentum) algorithms.

##### References

1. [1]D. Wackerly, W. Mendenhall, R. Scheaffer, L. Bain and M. Engelhardt, Mathematical statistics and probability, 1st ed. Andover: Brooks/Cole Cengage Learning, 2010.
2. [2]H. Abdi, B. Edelman and D. Valentin, Neural networks, 1st ed. Thousand Oaks, Calif. [u.a.]: Sage, 2001.
3. [3]R. Amardeep, "Training Feed forward Neural Network With Backpropogation Algorithm", International Journal Of Engineering And Computer Science, 2017.
4. [4]R. Rojas, Neural Networks: A Systematic Introduction, 1st ed. Berlin: Springer-Verlag,