EMBRY-RIDDLE Aeronautical University SCHOLARLY COMMONS

Beyond: Undergraduate Research Journal

Volume 6 Article 4

July 2022

Evaluating the Variable Stride Algorithm in the Identification of Diabetic Retinopathy

Ying Zheng
Embry-Riddle Aeronautical University, zhengy3@my.erau.edu

Brian Danaher
Embry-Riddle Aeronautical University, danaherb@my.erau.edu

Matthew Brown
University of Toledo, Matthew.t.brown.mtb@gmail.com

UNDERGRADUATE RESEARCH JOURNAL
Beyond

Evaluating the Variable Stride Algorithm in the Identification of Diabetic Retinopathy

Ying Zheng, Brian Danaher, Matthew Brown

Abstract

An experiment was performed to investigate a modified pooling method for use in convolutional neural networks for image recognition. This algorithm–Variable Stride–allows the user to segment an image and change the amount of subsampling in each region. This control allows for the user to maintain a higher amount of data retention in more important regions of the image, while more aggressively subsampling the less important regions to increase training speed. Three Variable Stride methods were compared to the preexisting pooling algorithms, Maximum Pool and Average Pool, in three different network configurations tasked with classifying Diabetic Retinopathy images between its early and advanced stages. Each combination was run multiple times and the AUC, Validation Loss, Validation Accuracy, and number of training epochs until convergence of each run was all collected. Maximum Pool and Average Pool were both found to be superior to Variable Stride when deployed in these scenarios.

Nomenclature

AUC = Area under the Receiver Operating Curve
AvgPool = Average Pooling
CNN = Convolutional Neural Network
FPR = False Positive Rate
Maxpool = Max pooling
NNSS = Nevada National Security Site
ReLU = Rectifier function
ROC = Receiver Operating Curve
TPR = True Positive Rate
VS = Variable Stride algorithm
VSR = Variable Stride Right Scheme
VSCe = Variable Stride Center Scheme
VSCu = Variable Stride Custom Scheme

Introduction

The Nevada National Security Site (NNSS) is a U.S. Department of Energy research and development complex that has been instrumental in the development and maintenance of the nuclear weapons of the United States. As full-scale nuclear testing is currently prohibited under international treaties, current experiments performed by the NNSS involve subcritical tests. Quantitative radiography - the science of using x-ray imaging to extract quantitative data (such as density) about an object - is performed by the NNSS using their Cygnus dual-beam radiography source [7]. The Signal Processing and Applied Mathematics research group within the NNSS is currently developing analysis techniques for the resulting image data which involve image classification: a task suited to the deployment of convolutional neural networks. This project was given by the research group to help develop their analysis techniques.

Neural networks are a form of artificial

intelligence which consists of a set of nodes (neurons) connected. When a datum arrives at a node, it is copied and passed to each of the other nodes the current node is connected to. Each connection has a weight assigned to it: a quantity which the datum is multiplied by as it moves through the connection. In a network designed to classify data into a fixed set of categories, the network terminates in a final layer with one node per category. Once passed through a sigmoid or ReLu activation function, the set of outputs form a probability distribution assigning the probability that the input is a member of each category. When a neural network is initially created to perform a task, its weights are randomly generated, and the network will perform as a random classifier. These networks are subsequently improved using an optimization process called training. Network training first involves feeding pre-classified training data into the network and checking how closely the network's predictions match the actual data. A loss function is then constructed using this difference between the classifications outputted by the network and the true classifications. A process known as backpropagation is then employed to compute the gradient of the loss function, on which a variety of algorithms can be deployed to find the extrema of the loss. As the neural network is optimized, the network's classifications get closer to the actual values of the training data and the network becomes more capable at classifying unknown data.

One type of neural network commonly used in image recognition is the convolutional neural network (CNN). These networks contain at least one convolutional layer, as opposed to the dense layers described previously. Within a convolutional layer, a series of filters (also known as kernels) is passed over the dataset to produce an activation map as an output. In the context of image recognition, filters can be created which recognize features such as filled-in regions, lines, simple shapes, different colors, and so on. These filters can be trained in a manner like training dense layers, with the goal of arriving at filters that recognize important parts of the image and pass them forwards. The resulting set of activations, containing the feature maps, is then flattened and fed into standard dense layers for final processing and output.

Training neural networks is often a very computationally intensive operation. The backpropagation step alone in a standard dense neural network is an $O(n^5)$ operation for n iterations on n layers each with n neurons [3]. Subsampling, also known as pooling, is a technique used in convolutional neural networks which increases the training speed by reducing the size of the images in the dataset while they are in the network. The two standard spatial pooling methods contained in many Python libraries are Average Pooling (AvgPool) and Max Pooling (MaxPool). AvgPool takes in a portion of data and averages the values together into a single data point, MaxPool takes the maximum value of the portion of data and passes this forward into a single data point. In the case of an image with a lot of noise (like static), AvgPool may be ideal because it "smudges" the noise out. In the case of images with fine details, MaxPool may be desirable because it can retain and exaggerate these details, if the pixels in the fine detail have larger pixel values than the neighboring pixels. An illustration of the effects of MaxPool and AvgPool is included in Figure 1.

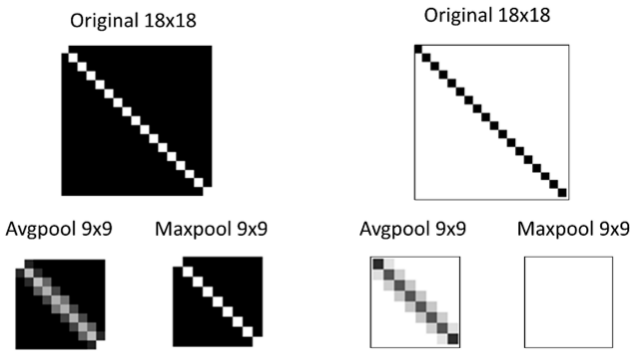

Figure 1: : An illustration of the effects of MaxPool and AvgPool (note that the rgb value for white > black)

The NNSS has been working on deploying an algorithm called Variable Stride (VS), which dynamically subsamples images based

on information density. This algorithm allows the user to section the images into horizontal and vertical strips and determine the pooling stride-the size of the down sampling filter-in each section. By manipulating these parameters, one can isolate the portions of an image which are most important in classification and only minimally down sample these regions, while more aggressively downsizing the less important parts of the image. An implementation of VS is provided in Figure 2.

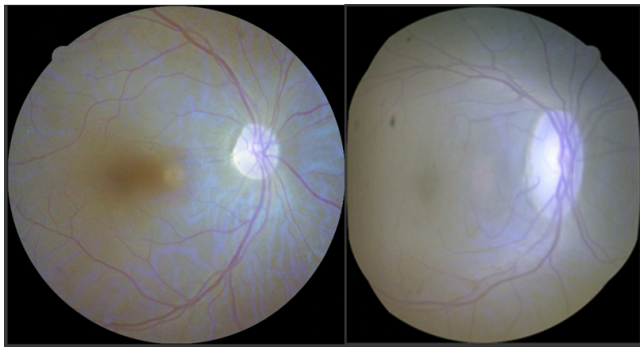

Figure 2: The effect of a Variable Stride scheme applied to a retinal image

This experiment aimed to quantify the performance of the VS algorithm in a binary classification situation and compare the algorithm to AvgPool and MaxPool. The data which was used to evaluate the algorithm is a set of 200 retinal images taken from a publicly available diabetic retinopathy dataset. Diabetic retinopathy causes scarring and other abnormalities in the retinal. It affects diabetic patients and can be identified from retinal images. While the initial dataset is sorted into 4 levels of severity of the disease, the NNSS' pre-processed dataset is taken from two categories: class 1 (being the least severe) and class 4 (being the most severe). The resulting dataset is split evenly into 100 images per category, and the images have been pre-processed to be of a consistent 600x600 initial pixel size. The NNSS found that applying a blue filter to the dataset increased accuracy, so such a filter was applied to this dataset. Additionally, all the images were rotated so that the optic disk lay on the right side of the image. VS is an algorithm that depends on the user to input the pixel strides and image ratio fractions; this rotation ensures that the various features of the eye stay in fixed locations on image so that the user can select specific VS schemes to target certain retinal features.

Metrics

One way CNNs are evaluated is by examining Receiver Operating Characteristic (ROC) curves. The ROC curve of a binary classifier consists of the true positive rate (TPR) plotted against the false positive rate (FPR) at various classification cutoff probabilities. The Area Under the ROC curve (AUC) is a helpful quantitative measure of the performance of a neural network that is more informative than single-number statistics [2]. A perfect classifier would always guess correctly, and as such would have an AUC of 1 as it is binned at a TPR of 1. A random classifier would guess correctly half of the time, and therefore would have an equal FPR and TPR for all cutoff probabilities. This would be seen graphically as a simple line with a slope of 1 starting at the origin. The corresponding AUC of a random classifier is 0.5.[4]

Other metrics that are commonly used to evaluate CNNs are the validation accuracy, validation loss, and the number of epochs until convergence. In training neural networks, an individual dataset is partitioned into training data to train the network and validation data to test how the network handles new data. Just like with the training data, the validation data has its own accuracy and loss function. The validation accuracy is straightforward and is simply the proportion of data classified correctly. Another justified classification metric, the validation loss function, is created using the difference between the network's predictions and the actual data like the validation accuracy [6]. An increasing test loss is evidence that the network is overfitting and memorizing data instead of learning the actual patterns behind the classification.

Procedure

The experimental procedure for this experiment was designed to evaluate VS under multiple conditions as applied in multiple different configurations. First, three different network structures were selected to analyze the performance of VS across multiple possible implementations. These structures were chosen using a preliminary analysis where 135 network structures were procedurally created and run on the diabetic retinopathy dataset with 160 training images and 40 testing images. The convolutional parts of these networks were generated by creating convolutional layer blocks consisting of a convolutional layer, a dropout function to combat overfitting [8], and a MaxPool layer of which the number of filters (2^n filters were implemented), and the number of layers were altered. The dense part of each network was created by first flattening the data, creating several uniformly sized dense layers, and then creating a single output node. The number of nodes in a dense layer and the number of filters per convolutional layer were set equal for simplicity. Once the validation/test accuracies were collected, the runs were split into groups based on the filter/node count (16, 32, and 64). The three network structures with the lowest validation loss were isolated, and then the dropout rate was changed to find the optimal network structure with the highest validation accuracy and lowest validation loss. These network structures are tabulated in Table 1. Each structure is also visually displayed in Figure 3 using a tool developed by Haris Iqbal[5].

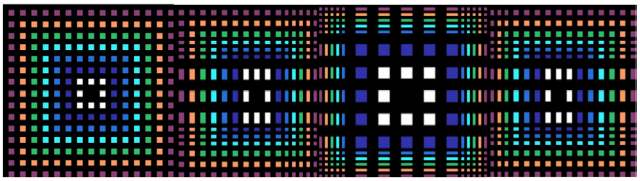

Figure 3: This Figure shows the process of VSR (center left), VSCe (center right), and VSCu (Right) on an original image (left). The original image is 600x600 pixels, and each square is 20x20 pixels in the original image.

Network Structure	Number of nodes and filters	Number of Convo-lutional Layers	Number of Dense Layers	Dropout Rate	Maxi-mum Test Accuracy
Network 1	32	5	4	0.05	0.8
Network 2	16	5	4	0	0.725
Network 3	64	5	1	0.05	0.75

Table 1: Properties of the network structures chosen for the experiment

Once the network structures were selected, a similar procedure was performed to find the three best-performing striding schemes for the experiment. Instead of creating the schemes procedurally, the schemes tested were chosen holistically by visually examining the activation maps of fully trained networks which used MaxPool. An example of a set of activation maps is included in Figure 4.

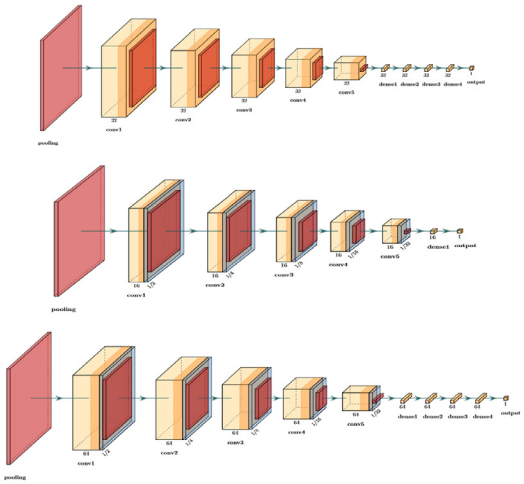

Figure 4: An illustration of the three network structures used in this experiment. Red signifies a pooling layer, yellow a convolutional layer, and blue a dropout layer

Striding schemes were thus created to address the areas most active in the activation maps. The three VS schemes that produced the highest test accuracy on the 64-node network were found to be Variable Stride Right (VSR), Variable Stride Center (VSCe), and Variable Stride Custom (VSCu). The stride methods are detailed in Table 2. The description listed below of each VS method describes the various parts of the striding method. The Vertical fractions divide the original image into separate parts on the vertical axis of the image; in total the fractions will add to 1. Each vertical fraction is

then assigned a vertical stride value at the same index, indicating how many pixels (n-1) the algorithm will skip when minimizing the image. For Example, for VSR, the first 20 percent of the image will have every other pixel be skipped, effectively dividing the length of the first 20 percent of the image by 2. The same process works for the horizontal stride and horizontal fractions.

Properties of the Various Visual Stride Methods	Variable Stride Right (VSR)	Variable Stride Center (VSCe)	Variable Stride Custom (VSCu)
Vertical Stride	[2,3,1,3,2]	[4,1,4]	[2,3,1,3,2]
Vertical Fraction	[⅕, ⅕, ⅕, ⅕, ⅕]	[⅓,⅓,⅓]	[⅕, ⅕, ⅕, ⅕, ⅕,]
Horizontal Stride	[2,3]	[4,1,4]	[3,2]
Horizontal Fractions	[⅔,⅓]	[⅓,⅓,⅓]	[⅕, ⅘]

Table 2: This Figure illustrates the properties of the top three striding methods.

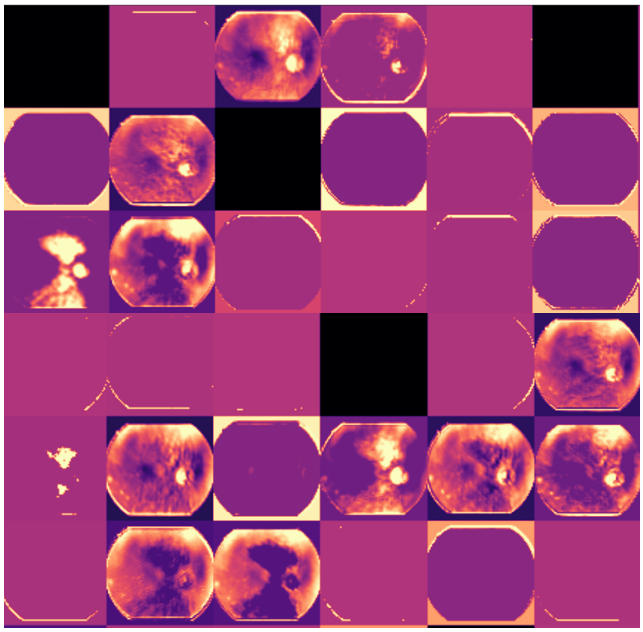

Figure 5" A selection of activation maps for a fully trained CNN on the diabetic retinopathy dataset

VSR was created to capture the right side of the eye, where the optic disk is located. VSCe was created to capture the center of the retina, the detail of which was often captured in the activation maps. VSCu was created as a combination of the two methods. The research team found it very difficult to apply VS to images in a short amount of time; perhaps since the code was written in standard Python instead of a faster language. Due to the long execution time, these striding methods were used to pre-process the images before inputting them in the networks. MaxPool and AvgPool were also implemented as preprocessing layers for consistency. Examples of preprocessed images are included in Figure 6.

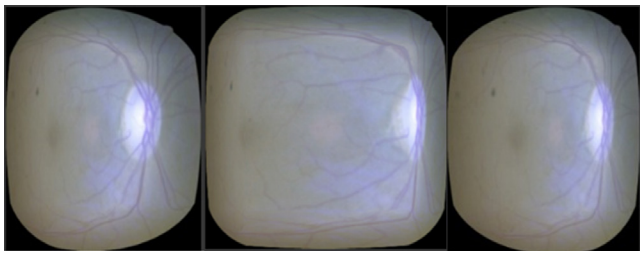

Figure 6: Demonstrations of the three striding methods tested: VSCu (left), VSCe (center), and VSR (right)

Once the striding schemes were chosen, a series of neural network runs were performed to test all 15 combinations of the 3 network structures (16-node, 32-node, and 64-node) and 5 pooling methods (MaxPool, AvgPool, VSR, VSCe, and VSCu). This experiment was conducted across three instances of Google Colab. To minimize the computation time within each instance, four runs were performed for each combination per person, giving a total of 12 runs per combination of striding method and network architecture. Preliminary testing revealed that the MaxPool and AvgPool would often converge faster than the VS methods and consistently within 100 epochs. In the interest of reducing converged runs, the VS methods were run for 150 epochs per run, while the MaxPool and AvgPool methods were run for 100 epochs each. After the end of each epoch, the training losses and accuracies, along with the validation losses and accuracies, were collected for graphical examination. The ROC curves were generated for each run at the final epoch by collecting the false positive rate (FPR) and true positive rate (TPR) data. The networks were determined to have finished their training if the training accuracy reached 100% before the end of the epochs, and the number of epochs a network took to converge was also collected.

Results

Each set of network runs were compared to the others by performing two-tailed t-tests for two independent sample means at a 95% confidence level. Determining which neural network configuration performed best is not a straightforward task, as several metrics must be considered holistically to extract the best overall picture. The t-tests were run on the AUCs, validation losses, validation accuracies, and training speeds of each network. For easy visual inspection, the results are displayed in grids where the intersection between two network structures displays the result of the horizontal set of runs against the vertical set of runs. If the square is gray, there is not a statistically significant result between the two sets of runs. If the square is green, the horizontal set of runs is significantly higher than the vertical, and the horizontal set of runs is significantly less than the vertical set if the square is red. As the two variables – the network structure and striding method - were changed throughout the experiment, results were only gathered from runs within the same network structure to isolate the pooling methods as a variable.

AUC

The overall plot of all the results from the AUC t-tests is included in Figure 7, and the plots within each network structure are in Figure 8. The only statistically significant result within a given network structure was that MaxPool resulted in a higher AUC than Variable Stride center in the 32-node network. On an inter-network level, the 32-node network with MaxPool performed better than a total of four other network configurations. This is likely due to the high standard deviation in the AUC from run to run. As an example, the maximum AUC for the VSR runs on the 32-node network was 0.761 with a minimum of 0.535 and a standard deviation of 0.06. This high standard deviation between runs is evidence that AUC is a poor metric to evaluate the performance between neural networks in the context of this experiment.

Figure 7: The overall AUC results of the networks

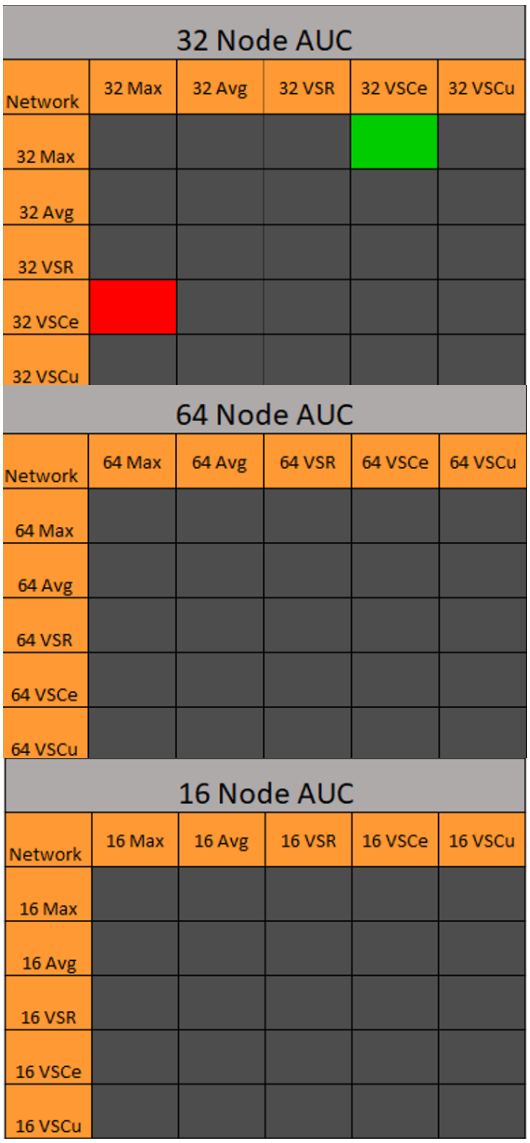

Figure 8: The AUC results from the individual intra-network runs

Accuracy

The overall plot of all the results is included in Figure 10, and the plots within each network structure are in Figure 9. In a similar way to the AUC, the only statistically significant intra-network result was that MaxPool achieved higher validation accuracy than variable stride right. Accuracy gives even sparser results than AUC; the only other significant result was that the 16-node network with VSR performed better than the 32 node network with VSR. Validation accuracy suffers from a high standard deviation just like AUC, making it a poor indicator of performance in this experiment.

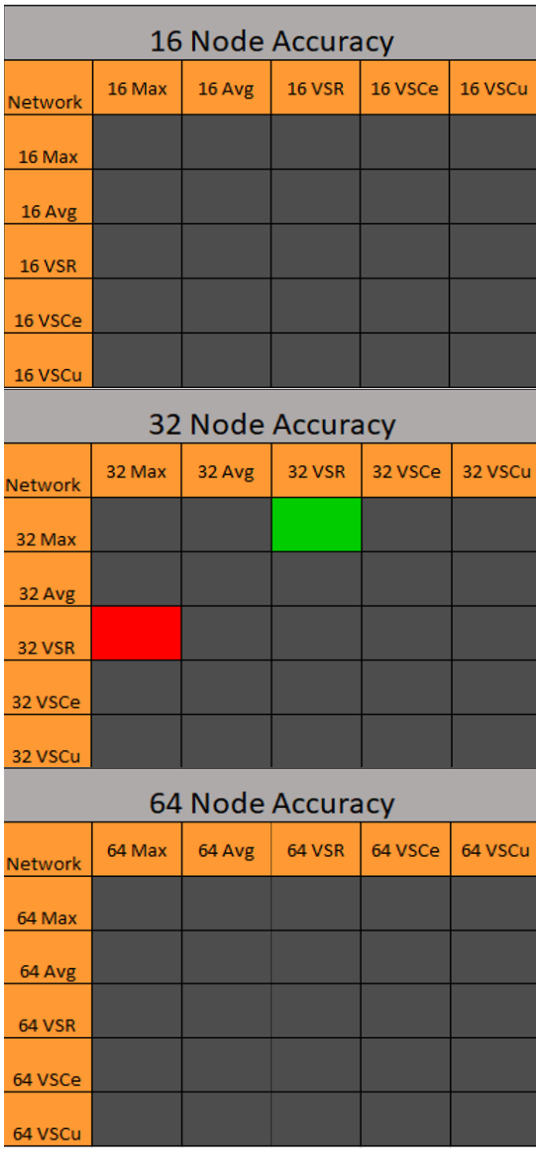

Figure 9: The accuracy results from the individual intra-network runs

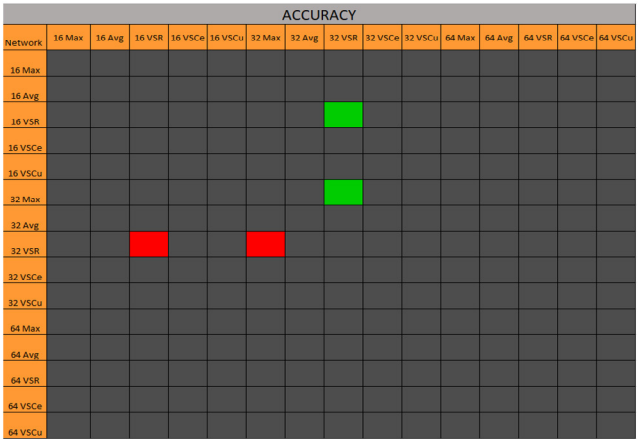

Figure 10: The overall accuracy results of the network runs

Loss

The overall plot of all the results is included in Figure 11, and the plots within each network structure are in Figure 12. More significant results exist for this metric than for the others, especially in the 16-node network. In all intra-network significant cases except for two, MaxPool or AvgPool outperform the VS methods. In these remaining two cases, the VSR method gives a significantly lower loss than the VSCe and VSCu schemes.

Figure 11: The overall loss results of the network runs

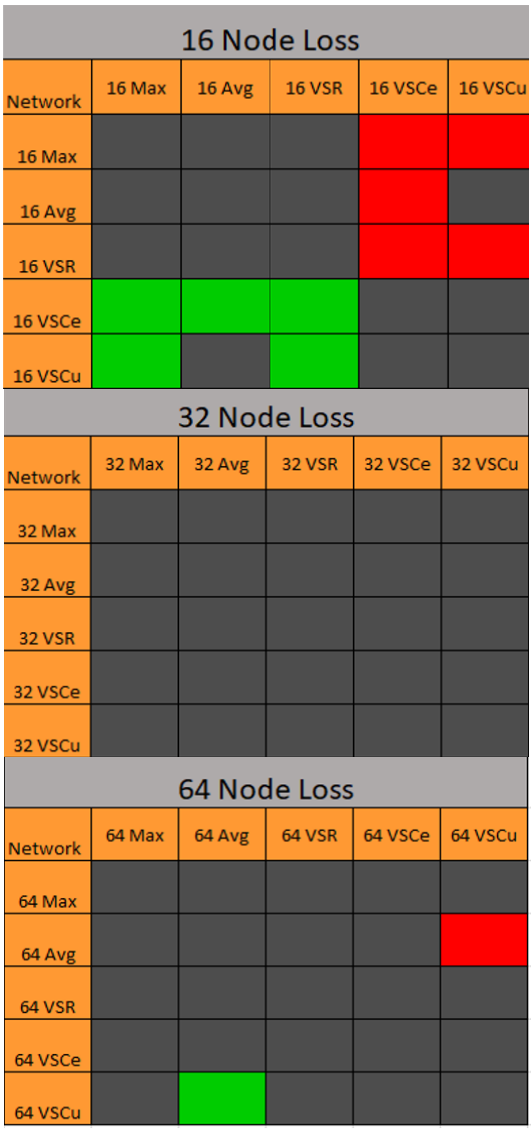

Figure 12: the loss results from the individual intra-network runs

Epochs

The overall plot of all the results is included in Figure 14, and the plots within each network structure are in Figure 13. To not skew the data, network runs which failed to converge were not included in this analysis and as such the number of runs in each combination was sometimes less than 12. The t-tests were updated accordingly. The number of epochs until converge, the training speed, resulted in by far the largest number of significant results. Out of the 12 significant intra-network results, MaxPool or AvgPool train faster in 10 instances. None of the VS methods beat MaxPool or AvgPool, and only outperform the other VS schemes or schemes in other network structures. Considering that the preferred outcome of VS was to trade off accuracy for speed, these results demonstrate that VS does not cause the networks to complete training faster than MaxPool or AvgPool.

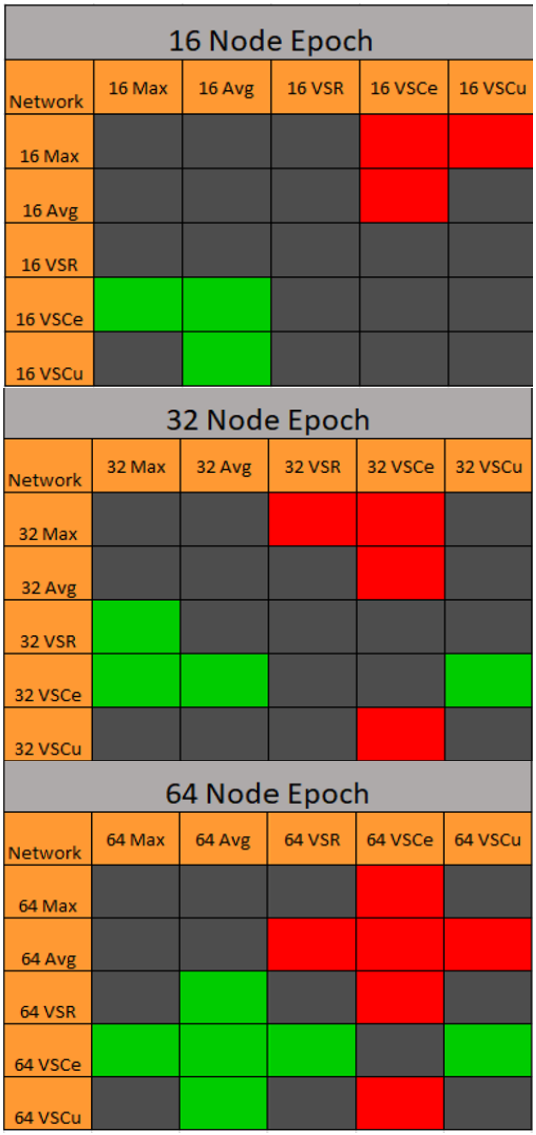

Figure 13: The training time results from the individual intra-network runs

Figure 14: The overall stopping times of the individual network runs

Conclusion

This research found that none of the VS algorithms consistently performed on par or better than MaxPool or AvgPool. While most of the metrics collected returned few statistically significant results, the majority of the significant results of the AUC, loss function, and test accuracy had the VS schemes perform worse than MaxPool or AvgPool. The VS algorithms were also found to consistently train the networks slower than MaxPool or AvgPool. One potential reason for this failure is that the indicators of diabetic retinopathy cover the eye in a near-uniform pattern such that one region of the eye is not significantly more important than the other. This occurrence would make VS inherently unsuited for use in identifying the disease. One other possible explanation for the failure of VS is that the dataset is too small for the network structure and overfitting consistently occurs. This is supported by the fact that all loss functions, regardless of striding method or network structure, were not minimized and instead increased as the networks were trained: a potential indicator of overfitting.

References

[1]Basavarajaiah, M., "Maxpooling vs minpooling vs average pooling," *Medium*, 2019.

[2]Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7), 1145–1159. https://doi.org/10.1016/s0031-3203(96)00142-2

[3]Daniely, A., Linial, N., Shalev-Schwartz, S., "From Average Case Complexity to Improper Learning Compexity," *The Hebrew University*, Jerusalem, Israel, 2014.

[4]Fawcett, T., "An Introduction to ROC Analysis," *Pattern Recognition Letters*, Palo Alto, CA, 2005.

[5]Iqbal, H. (2018). *PlotNeuralNet*. Github. Retrieved from https://github.com/HarisIqbal88/PlotNeuralNet

[6]Janocha, K., & Czarnecki, W. M. (2017). On loss functions for deep neural networks in classification. *Schedae Informaticae,* 1/2016. https://doi.org/10.4467/20838476si.16.004.6185

[7]J. Smith, R. Carlson, and et al., Cygnus Dual Beam Radiography Source, in Pulsed Power Conference, 2005 IEEE, 2005, pp. 334–337.

[8]Ying, X. (2019). An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168, 022022. https://doi.org/10.1088/1742-6596/1168/2/022022