Mass Spectrum Noise Reduction with Neural Networks

Group 8

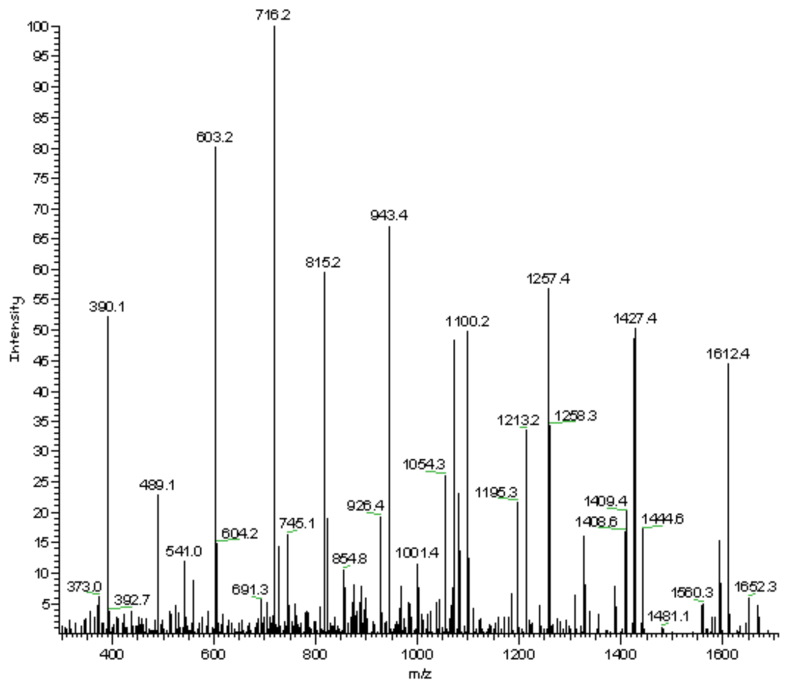
Jonah Kubath
Matt Peter

Abstract

Increasing the rate at which we can identify proteins via their peptide sequences is one of the many computational problems present in the world today. Over the past few decades, protein mass spectrometry has become the primary way of identifying proteins. However, due to the complexity and quantity of proteins, as well as the large amount of “noise” that comes with gathering data, identifying one with an extremely high degree of certainty quickly becomes a daunting task. To solve this problem, large amounts of research have been put into finding ways to decrease the amount of time it takes to identify proteins, while still maintaining a decent level of accuracy. For our project, we decided to specifically focus on finding ways to reduce that large amount of “noise” that is part of the input data using neural networks. Our reasoning behind choosing this path was mostly due to the current success of neural networks in terms of noise reduction in images and audio. Unfortunately, due to the scale and complexity of proteins and the limited time and resources we had, our results did not provide any supporting evidence that neural networks are indeed a viable path to protein identification.

1. Protein Background

In the field of proteomics, “the study of all proteins in a biological system”, mass spectrometry is a technique that is used “to detect, identify and quantitate molecules based on their mass-to-charge (m/z) ratio” [1]. In order to do this, a protein must first be broken down into a set of peptides using enzymes. After this is done, the first stage of mass spectrometry (MS1) takes place, where peptides are further broken down into ions using an ion source and are separated by m/z ratio. In the second stage of mass spectrometry (MS2), ions of a particular m/z ratio are selected and fragmented, creating fragment ions which are then separated and detected [2]. This collection of fragment ions can then be quantified in MS2 data as “spectra”, a collection of “peaks” that shows the ions’ corresponding intensities at various m/z ratios.

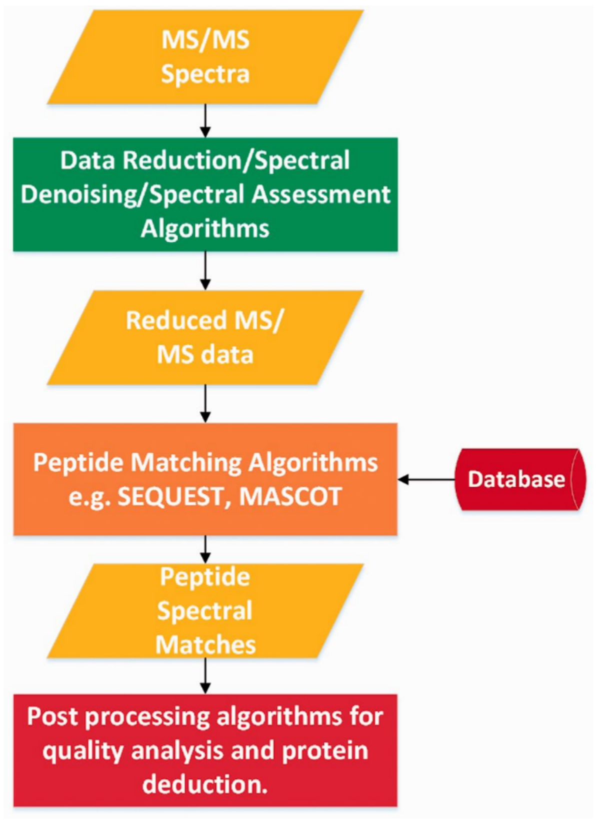


MS2 Data

This MS2 data is then analyzed in order to determine what the original peptides and proteins were. Unfortunately, only a small percentage of the peaks in a spectrum are useful in determining its peptide, as lot of the peaks are either noise or simply not helpful for the given spectrum [3]. This means that a large portion of the time spent analyzing the spectra does not actually improve the end result. In an effort to reduce the amount of time needed to accurately identify a peptide, research has been directed towards finding a way to eliminate this noise without severely reducing the accuracy of the results. One of the algorithms that has been developed to accomplish this is MS-REDUCE.

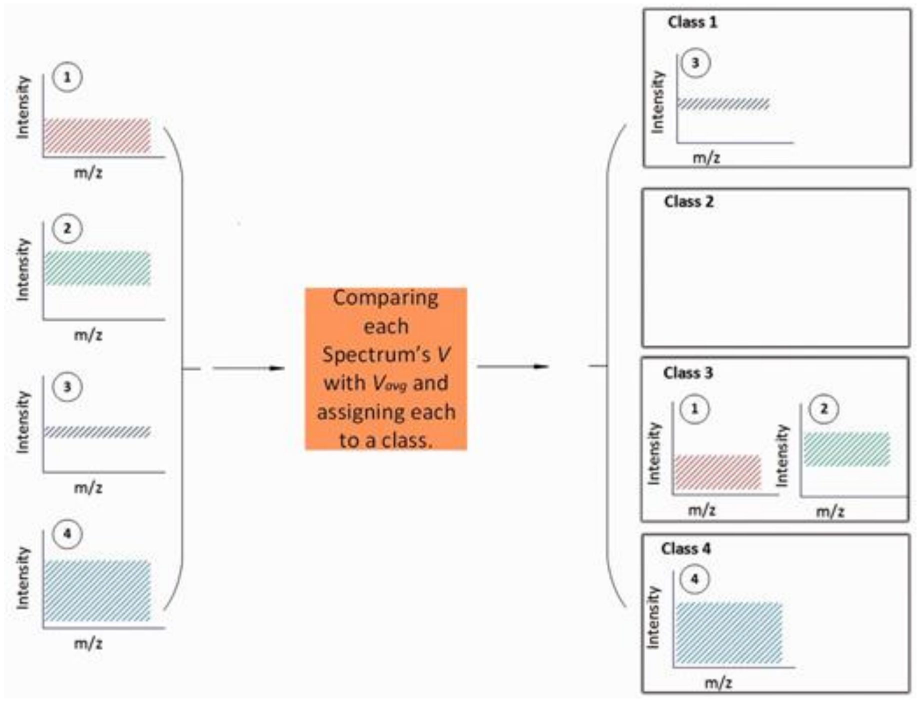
2. MS-REDUCE

MS-REDUCE consists of a three-stage pipeline through which the MS2 data is passed. The three stages of the pipeline are (i) Spectral Classification, (ii) Peak Quantization, and (iii) Weighted Random Sampling.


MS/MS Spectra
Data Reduction/Spectral Denoising/Spectral Assessment Algorithms
Reduced MS/ MS data
Peptide Matching Algorithms e.g. SEQUEST, MASCOT
Database
Peptide Spectral Matches
Post processing algorithms for quality analysis and protein deduction.

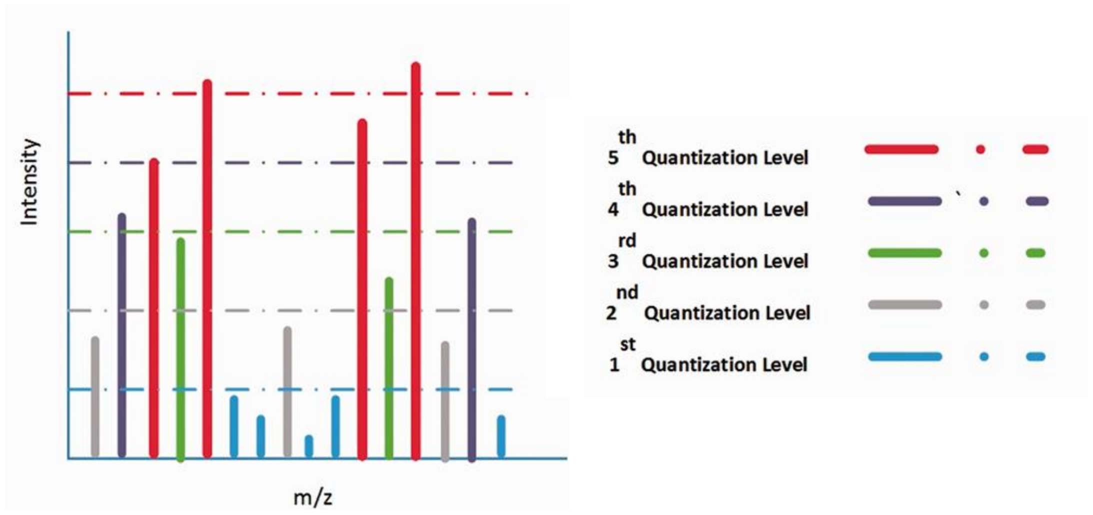
Pipeline for MS-REDUCE algorithm

In the Spectral Classification stage, spectra are placed into one of four classes based on the difference between the average of their highest ten peaks and the average of their lowest ten peaks, also known as their intensity spread. The boundaries of each of the four classes are determined by the average intensity spread over all the spectra in the dataset, with the first class having the smallest intensity spreads and the fourth class having the largest intensity spreads.


Intensity
m/z
1
2
3
4
Comparing each Spectrum's V with V_{avg} and assigning each to a class.
Class 1
Class 2
Class 3
Class 4

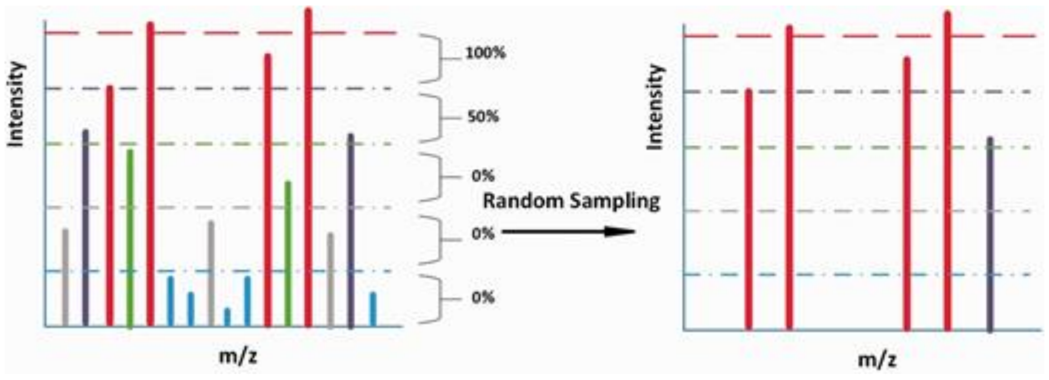
MS-REDUCE Spectral Classification stage

After all the spectra have been classified, their peaks are grouped in the Peak Quantization stage. In this stage, all the peaks in a given spectrum are placed into one of *n* levels, where *n* is determined based on which class the spectra was placed in in the Spectral Classification stage (Ex. Class 1 = 5 Levels; Class 2 = 7 Levels; Class 3 = 9 Levels; Class 4 = 11 Levels). The boundaries of each of the *n* levels is determined by the average intensity of the highest ten peaks in the given spectra, with the lowest level having the lowest intensity peaks and the highest level having the highest intensity peaks.



MS-REDUCE Peak Quantization stage

Finally, in the Weight Random Sampling stage, *p* peaks are selected from a given spectrum, where *p* is determined based on the reduction factor. For example, if the reduction factor is 70% and there are 30 peaks in the given spectrum, 9 of those peaks will be selected in the Weight Random Sampling stage. As far as figuring out which peaks to select, peaks are selected starting from the highest quantization level and continuing with lower quantization levels until the number of peaks needed have been selected. If a given level has less than the remaining number of peaks needed, all the peaks from that level are selected and the amount is subtracted from the remaining needed. If a given level has the exact number of remaining peaks needed, all the peaks from the level are selected and the sampling process is complete. If a given level has more than the remaining number of peaks needed, the amount needed are randomly selected from the all peaks on that layer and the sampling process is complete [3]. The result of this three-stage pipeline is a set of spectra that are reduced to a partially randomized sample of the original spectra with the lowest peaks removed.



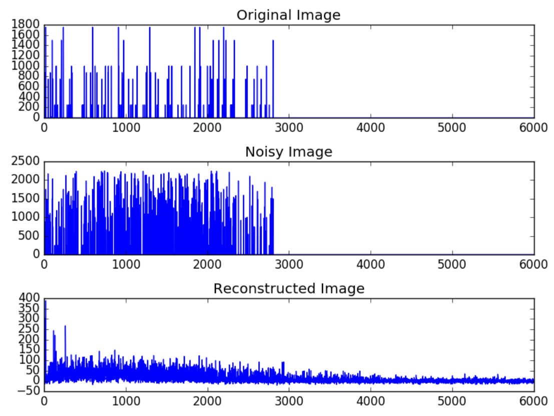
MS-REDUCE Weight Random Sampling stage

3. Generating MS2 Data

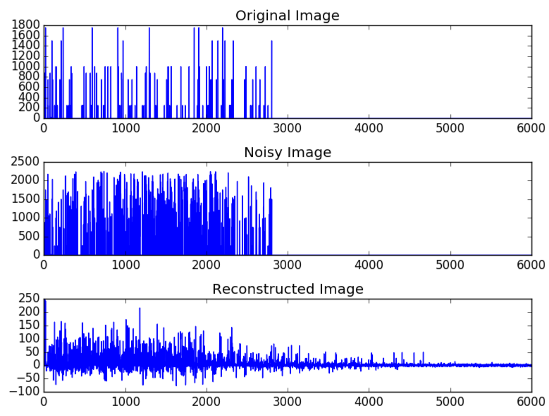
In order for us to train and test our neural network, we needed to develop a process for generating MS2 data both with and without noise. To do this, we started by downloading a FASTA file that contained proteins found in yeast. FASTA files follow a specific text-based format that is used for representing peptide sequences, where each sequence begins with a description containing the protein's ID and name and some other information, followed by the peptide sequence itself [4]. Using our python script *gen_peptide.py*, we took this file and used it to produce two other files: *peptideData.txt*, which contains all possible peptide sequences of length 6 to 50 that are subsequences of full peptide sequences, and *proteinData.txt*, which contains the ID and name that corresponds to each of the peptide sequences. Next, we needed to split up the peptide sequence data into training, validation, and test sets so we could develop our neural network. To do this, we wrote another python script, *split_sets.py*, which iterates through the peptide sequences, placing them in *train.txt*, *valid.txt*, or *test.txt* based on the range a randomly generated number falls in so that roughly a specified percentage of the data will go in each file. Once this was complete, we used an application called *MaSS-Simulator* to generate the corresponding MS2 data for each of the peptide sequences both with and without noise. The MS2 data with noise would be used as input for the neural network while the MS2 data without noise would be used to determine the accuracy of the output of the neural network. After the MS2 data generation process was complete, we used our python script, *noise_reduction_ms2.py*, to train, validate, and test our neural network.

4. Testing the Neural Network

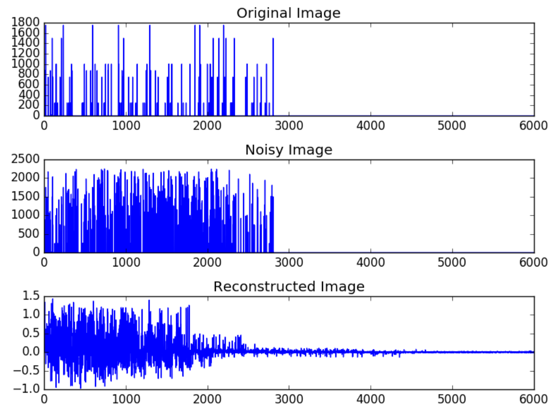
We performed 32 tests with our neural network with varying levels of input data, epochs, and layers. The following are the results of a single test done with each configuration, showing the original data without noise, the original data with noise added, and the neural network's prediction of the noiseless data:



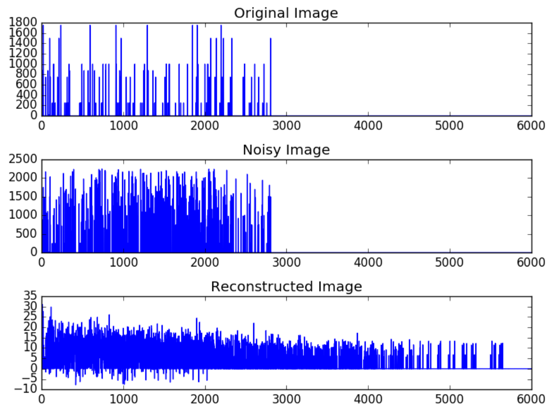
100 Spectra, 5 Epochs, 1 Layer, 100 Nodes



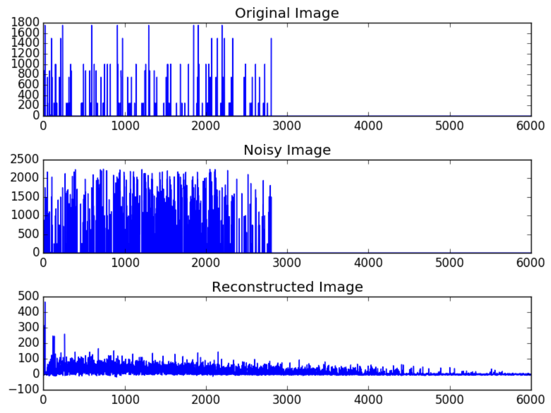
100 Spectra, 50 Epochs, 1 Layer, 100 Nodes



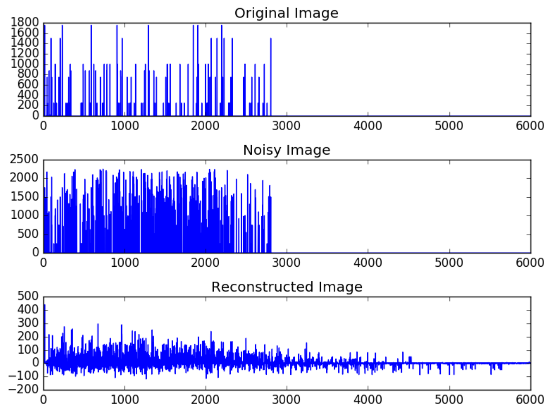
100 Spectra, 500 Epochs, 1 Layer, 100 Nodes



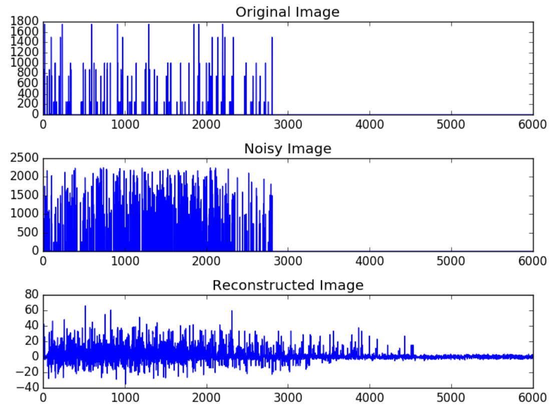
100 Spectra, 5000 Epochs, 1 Layer, 100 Nodes



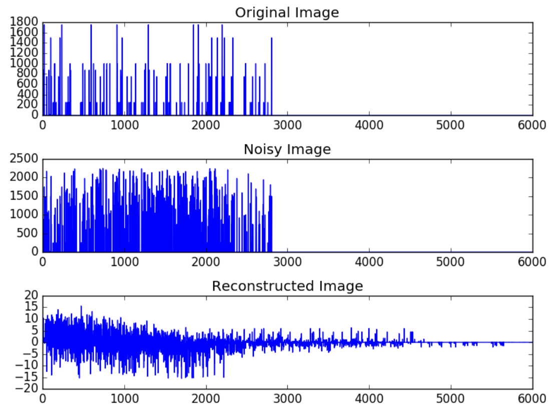
100 Spectra, 5 Epochs, 2 Layers, 50/50 Nodes



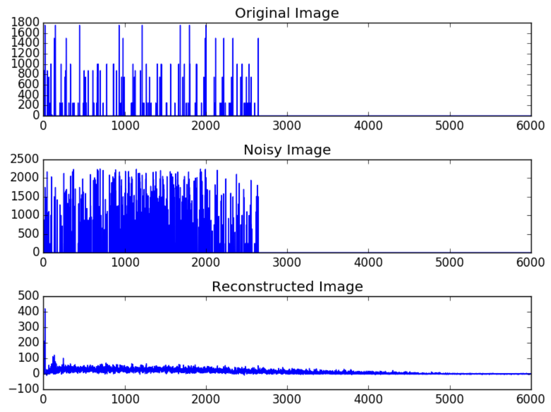
100 Spectra, 50 Epochs, 2 Layers, 50/50 Nodes



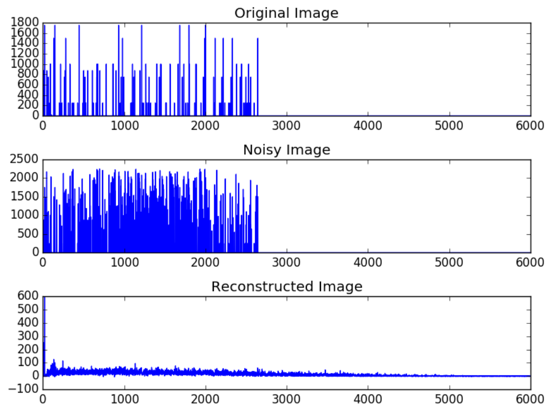
100 Spectra, 500 Epochs, 2 Layers, 50/50 Nodes



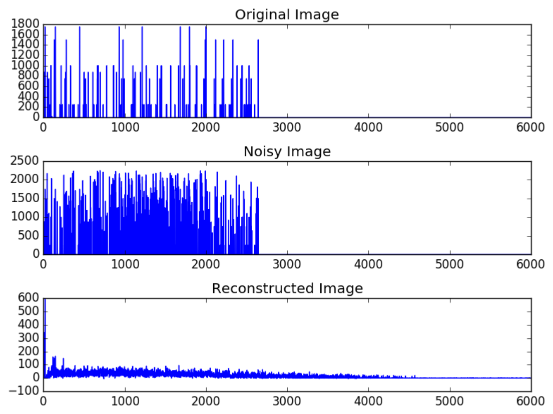
100 Spectra, 5000 Epochs, 2 Layers, 50/50 Nodes



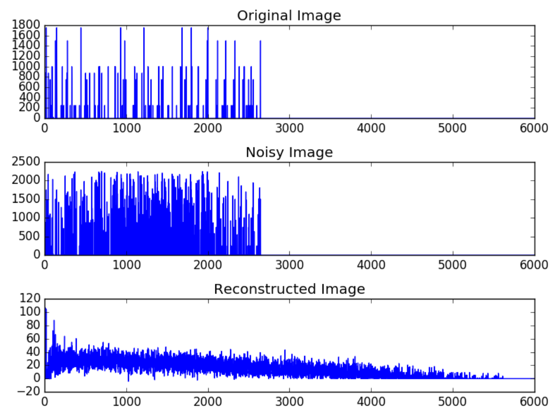
500 Spectra, 5 Epochs, 1 Layer, 100 Nodes



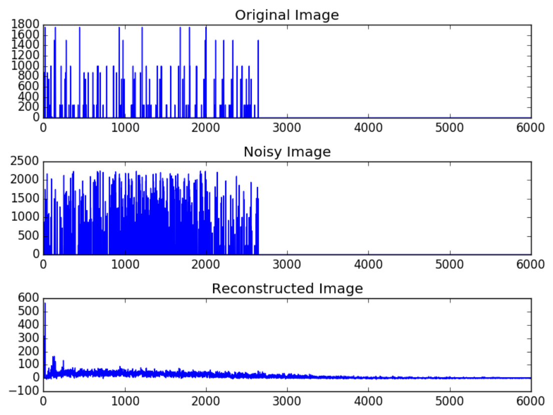
500 Spectra, 50 Epochs, 1 Layer, 100 Nodes



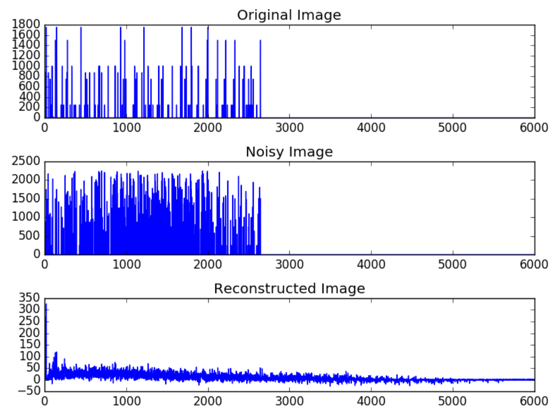
500 Spectra, 500 Epochs, 1 Layer, 100 Nodes



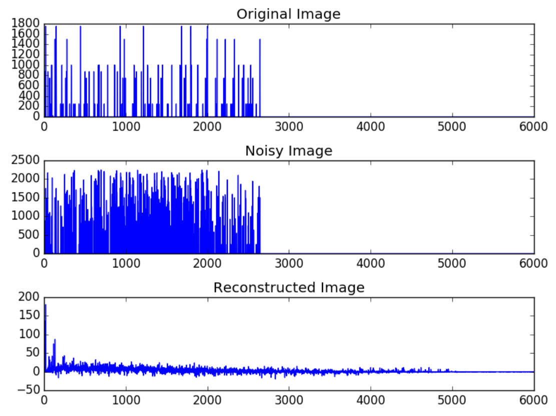
500 Spectra, 5000 Epochs, 1 Layer, 100 Nodes



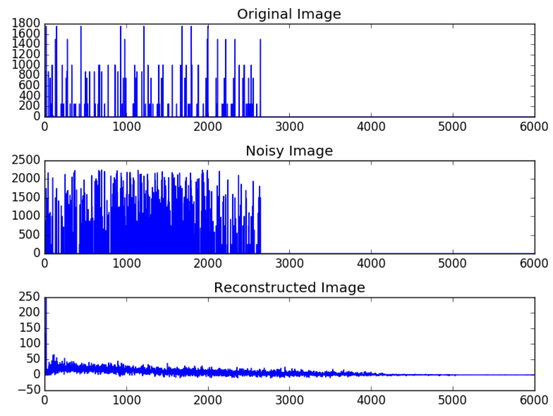
500 Spectra, 5 Epochs, 2 Layers, 50/50 Nodes



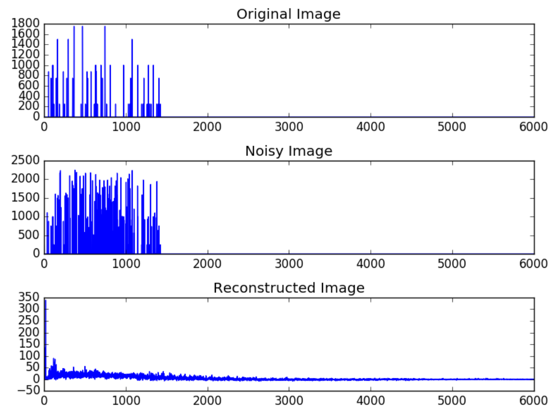
500 Spectra, 50 Epochs, 2 Layers, 50/50 Nodes



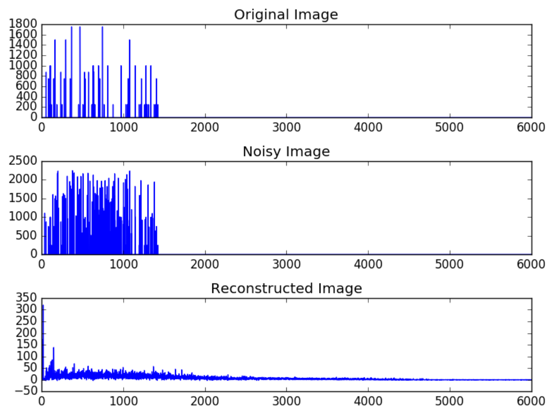
500 Spectra, 500 Epochs, 2 Layers, 50/50 Nodes



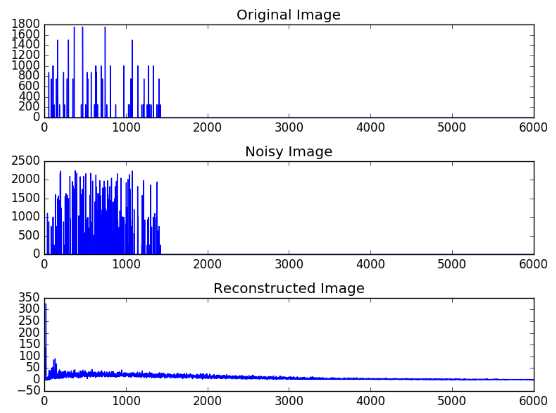
500 Spectra, 5000 Epochs, 2 Layers, 50/50 Nodes



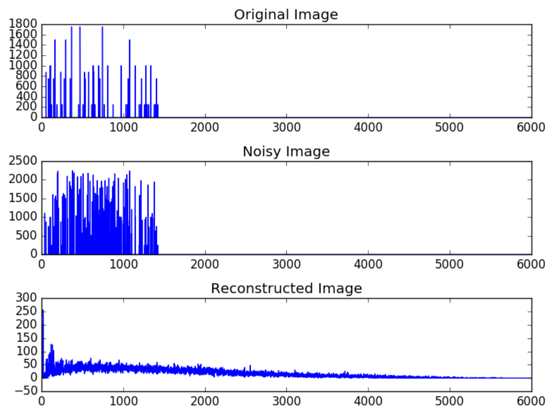
1000 Spectra, 5 Epochs, 1 Layer, 100 Nodes



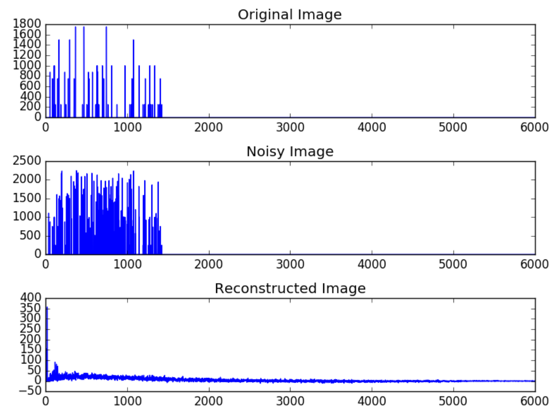
1000 Spectra, 50 Epochs, 1 Layer, 100 Nodes



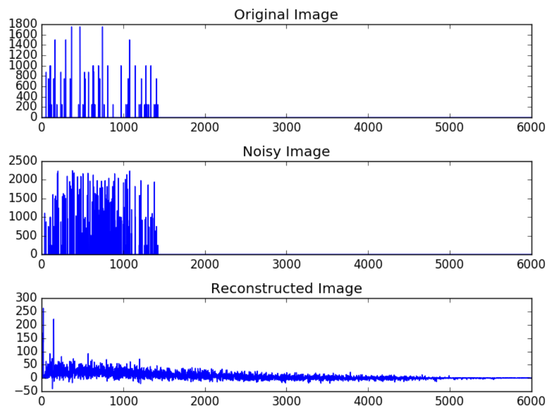
1000 Spectra, 500 Epochs, 1 Layer, 100 Nodes



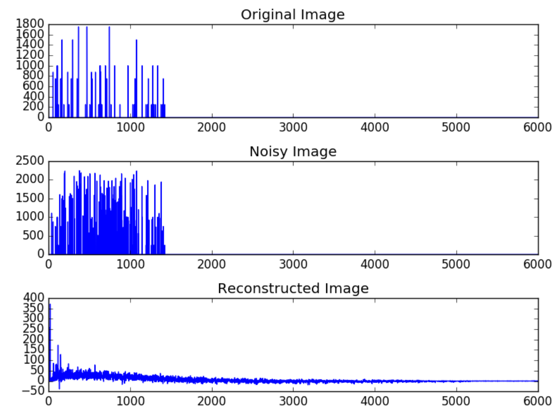
1000 Spectra, 5000 Epochs, 1 Layer, 100 Nodes



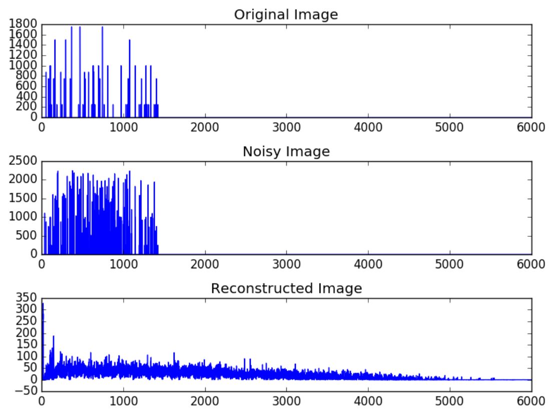
1000 Spectra, 5 Epochs, 2 Layers, 50/50 Nodes



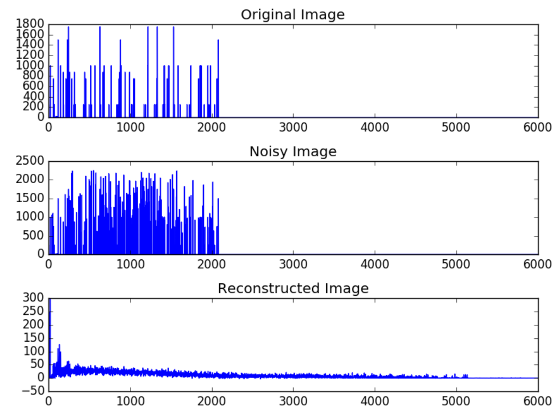
1000 Spectra, 50 Epochs, 2 Layers, 50/50 Nodes



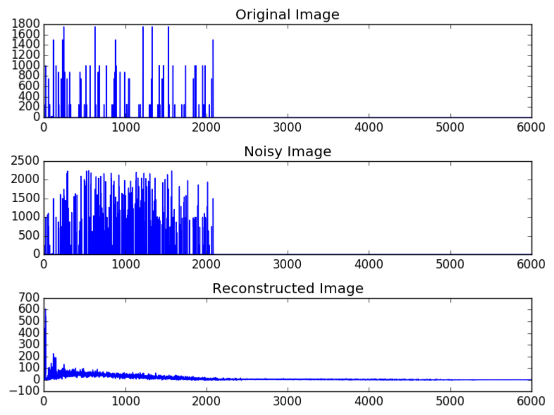
1000 Spectra, 500 Epochs, 2 Layers, 50/50 Nodes



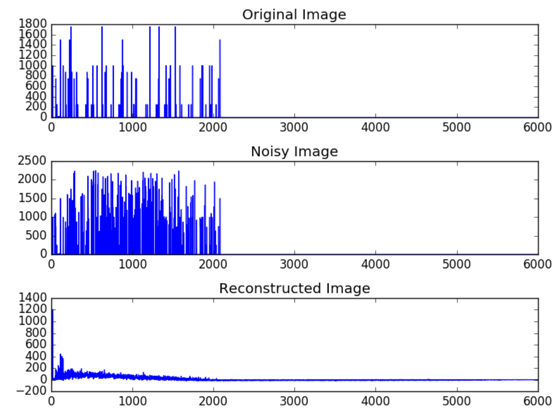
1000 Spectra, 5000 Epochs, 2 Layers, 50/50 Nodes



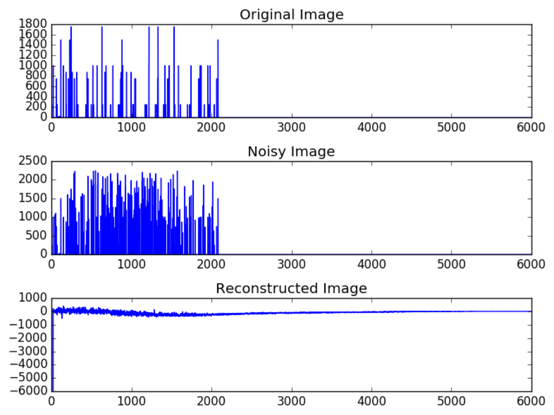
5000 Spectra, 5 Epochs, 1 Layer, 100 Nodes



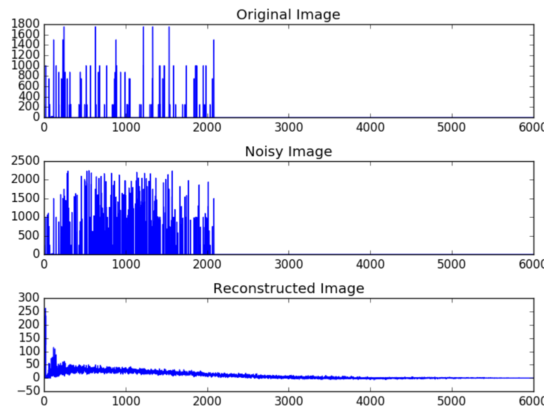
5000 Spectra, 50 Epochs, 1 Layer, 100 Nodes



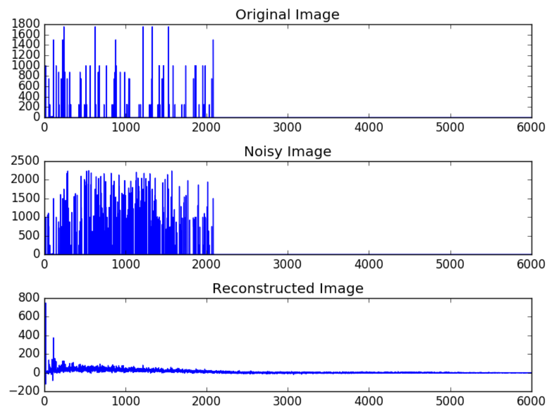
5000 Spectra, 500 Epochs, 1 Layer, 100 Nodes



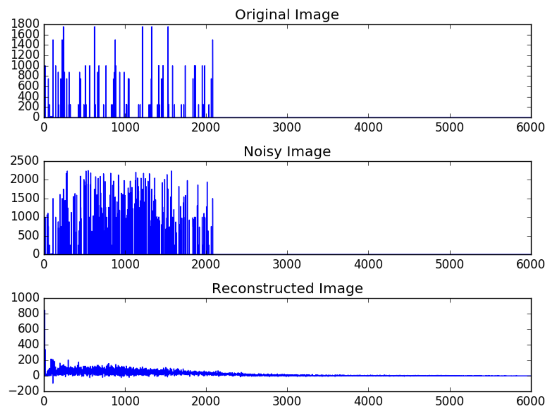
5000 Spectra, 5000 Epochs, 1 Layer, 100 Nodes



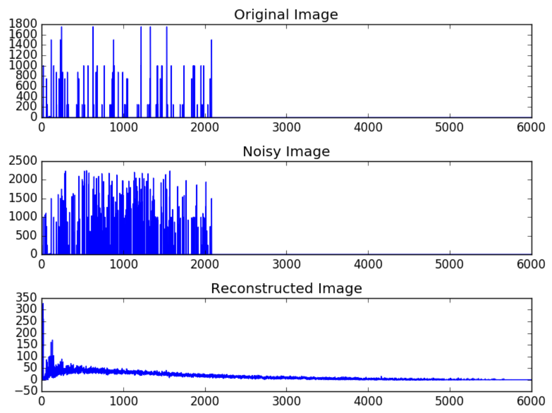
5000 Spectra, 5 Epochs, 2 Layers, 50/50 Nodes



5000 Spectra, 50 Epochs, 2 Layers, 50/50 Nodes



5000 Spectra, 500 Epochs, 2 Layers, 50/50 Nodes



5000 Spectra, 5000 Epochs, 2 Layers, 50/50 Nodes

Unfortunately, most of the results turned out to be very basic and didn't seem to have changed much from the starting state, and the ones that did have more variation weren't very accurate when it came to predicting the original data. This was also apparent when the training was being done, as the MSE obtained from the validation data often failed to vary much from the starting value. However, given the scale and complexity of this data and our limited time and resources, it's possible that we didn't provide the neural network with enough time or resources to properly develop, and that increasing these could give us more accurate results. Future research could be devoted to performing larger tests and finding ways to increase the rate at which they are performed.

References

[1] “Overview of Mass Spectrometry | Thermo Fisher Scientific - US.” *Overview of Mass Spectrometry | Thermo Fisher Scientific - US*, www.thermofisher.com/us/en/home/life-science/protein-biology/protein-biology-learning-center/protein-biology-resource-library/pierce-protein-methods/overview-mass-spectrometry.html.

[2] “Tandem Mass Spectrometry.” *Wikipedia*, Wikimedia Foundation, 18 Mar. 2019, en.wikipedia.org/wiki/Tandem_mass_spectrometry.

[3] Gul Awan, Muaaz, and Fahad Saeed. “MS-REDUCE: an Ultrafast Technique for Reduction of Big Mass Spectrometry Data for High-Throughput Processing.” *OUP Academic*, Oxford University Press, 21 Jan. 2016, academic.oup.com/bioinformatics/article/32/10/1518/1743195.

[4] “What Is FASTA Format?” *Zhang Lab*, zhanglab.ccmb.med.umich.edu/FASTA/.