Neural Network Function Interpolation

For this project, I used two different versions of the project for demonstration. Both versions used a random distribution of data points. However the first is with a “fuzzy” function, which adds a random value to our spaced points. The second used machine-accurate points with no added randomness.

Our first trial used 7 layers, with 64 neurons apiece. For the fuzzy function, it was impossible to get the error very low, even with many points. At max, I had 50000 points. I attempted many different training “times” (epochs), but found that anything above 500 produced vastly diminishing returns, with a maximum MSE around 0.03, and an average mean squared error (MSE) of O(10-2). For the second function, with no fuzziness, the maximum MSE, even using many data points, was around .02, though the average MSE was better at O(10-4).

I lowered the numbers of layer to 3, and I accordingly upped the nodes to 128, to have fewer but more detailed layers to try and capture the function better. This did not help at all with the fuzzy data, which stayed at the same maximum error and MSE. For the normal function however, this dropped our average MSE to O(10-6), below our goal of O(10-5), without affecting runtimes.

So in conclusion, a few detailed layers was superior. The fuzzy function was by nature harder to approximate, but I found the best possible solution even with only a few data points. For the smooth function the detailed approach made the results significantly better.

