Noise in neural networks - Steven Senger

It is common practice to add noise to a neural network. With these projects, we'll explore some ways to add noise to a neural network to potentially increase its functionality.

Project 1:
Any digital computer is constrained to the computational power of a Turing Machine (TM). However, if we augment the computation of a TM with real-valued noise, we can give it indirect access to what we call an advice string, that may aid in computation. While fully fleshing that out may be a bit complicated, we can explore a simpler, related phenomenon. Suppose we have a single-input perceptron that outputs a 0 if its input is less than 1/2, and a 1 otherwise. Now suppose that we want to use this perceptron to detect whether or not its input is above or below a different real number, say, e/π, and we are not allowed to scale the inputs. Imagine that it is already at the lowest detectable resolution. One technique to get the output we want is to add some time-varying random noise to the input, and have our output be determined by averaging multiple "samples," or by making a decision based on observations with different noisy inputs. The idea is that the resolution errors will, with high probability, be averaged out. We could then try to use this as a basis for more complicated increases in resolution of an array of perceptrons.

Project 2:
In practice, we set the parameters of a neural network with a small set of training data, and then unleash our beast on the wild. However, if our training data set is too small, or has too little variation, it can lead to a problem called overfitting. In this project, we will take a small set of training data for a neural network and generate a bigger set of training data by perturbing the original training data by some random noise. However, this has to be done with care, as adding too much noise, or adding noise in a reckless way, could lead to other unwanted consequences.