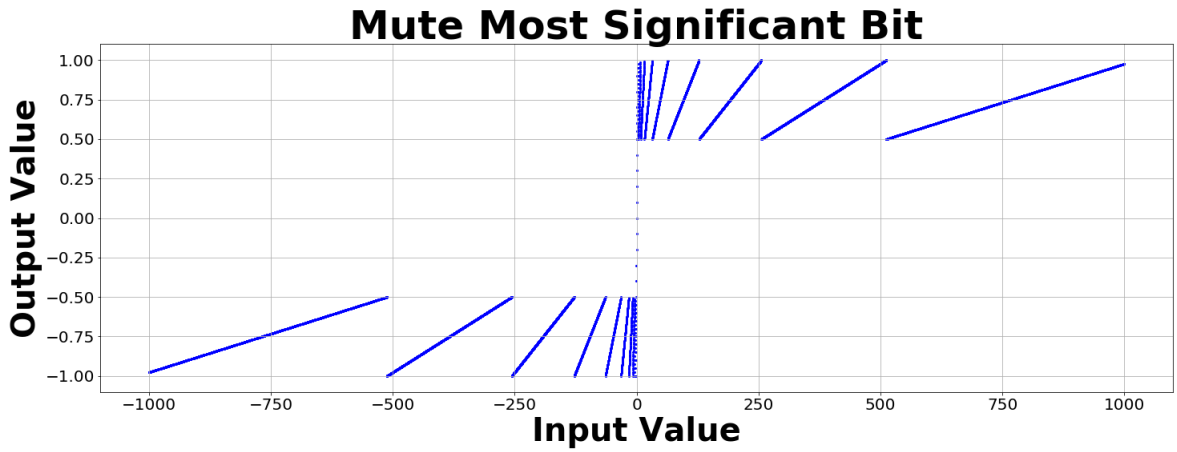
Place for Landon

0.1 Case Study 1: Multilayer Perceptron

The purpose of this case study was to develop an understanding for how the multilayer perceptron (MLP) neural network architecture is affected by a *bit muting* attack function. This type of attack would seek to manipulate the values of a floating-point number at a binary level. In the case of this experiment, we have chosen to model a function that forces the most-significant-bit (MSB) in the exponent of floating-point number to be muted to 0. In each MLP model, numerical values were stored as double precision floating-point numbers as per the IEEE 754 standard.

After a a single floating-point number has it's MSB muted, it is then constrained to lie on the open interval $(-1, +1)$. A map of input values against MSB output values is shown in fig. (0.1). The inputs are a subset of the possible values of a double precision floating-point numbers for visual ease.



0.1.1 Experimental Setup

CNN model, layers, neurons, forward backward propagation what data set you used describe the attack function

The MLP neural network model is composed of multiple layers of *neurons*, that each contain a real-valued floating-point number called an *activation*. Layers of neurons interact with the neurons in adjacent layers in sequence though a series of weighting coefficients, and bias elements, which is then subjected to an activation function. In general, given a

layer of activations $\vec{x}^{(l)}$ (The superscript is used as a layer index), the activations in the next sequential layer, $\vec{x}^{(l+1)}$, are produced with weighting matrix $\hat{W}^{(l)}$, bias vector $\vec{b}^{(l)}$, and activation function f such that:

$$\vec{x}^{(l+1)} = f\Big(\hat{W}^{(l)}\vec{x}^{(l)} + \vec{b}^{(l)}\Big) \tag{1}$$

The recurrence of equation (1) is used to pass information forward through the network. In a network with L layers, raw information is given with $\vec{x}^{(0)}$ and the network's prediction is given by the final layer, $\vec{x}^{(L-1)}$.

The Multilayer Perceptron network must then find a series of parameters, which are the elements in all weighting matrices and all bias vectors, that allows for a given sample to produce a low value for a given *Loss function*. The loss function measures the difference between a networks given output and expected output; with better trained networks producing generally low loss values over a set of samples. The process of finding the values is called *optimization*. There are many different algorithms that perform optimization, and for this experiment, we have chosen to use *Stochastic Gradient Descent* (SGD) due to its widespread use and versatility.

To model a bit-muting attack, we introduce and *attack function* that acts in the matrix-vector product $\hat{W}^{(l)}\vec{x}^{(l)}$ in eqn. (1). When a Boolean *trigger condition* parameter is met, the MSB attack as described above, eqn. (1) is instead replaced with it's attack variant:

$$\vec{x}^{(l+1)} = f\Big(A\big[\hat{W}^{(l)}\vec{x}^{(l)}\big] + \vec{b}^{(l)}\Big) \tag{2}$$

Where A is the attack function, applied element-wise to each object in it's argument. Equation (2) is then producing a perturbation from expected values that is not explicitly acounted for in the optimization procedure.

0.1.2 Impact of muting attack on four metrics

To measure the impact of this type of attack on the image classifier model, we test a baseline classifier against one subset to the

add picture before and after attack

add pictures (results) and draw conclusion