CS 4033 Project 1

Your Name

1 Design Choices Made by the Assignment (Steps 2-3)

The following choices were made by the assignment specification:

- **Learning paradigm:** Supervised learning, as labeled class data is provided.
- **Task type:** Binary classification — each sample belongs to one of two classes.
- **Architecture type:** Feedforward neural network with error backpropagation.
- **Loss signal:** Classification error between predicted and true class label.
- **Datasets:** Nine fixed datasets across Gaussian 2D, Gaussian 3D, and crescent moon distributions.

2 Design Choices For Us to Consider (Steps 4-5)

The following choices were considered, as well as what we what we plan on doing:

- **From Scratch:** Do it from scratch in C. Does not seem like we need something complex for this.
- **Hidden layers:** One hidden layer, as the universal approximation theorem guarantees a single hidden layer is sufficient to approximate any continuous function given enough neurons. Also simplifies things.
- **Hidden neurons:** 8 neurons per hidden layer.
- **Activation function:** Sigmoid, $\sigma(x) = \frac{1}{1+e^{-x}}$, natural probabilistic interpretation for binary classification. Also simple derivative for backprop.
- **Learning rate:** $\eta = 0.1$ Just a guess
- **Weight initialization:** Uniform random in $[-1, 1]$. Starting small.

3 Design Choices After Implementation (Steps 7-8)

- **From Scratch:** Yup, did it from scratch.
- **Hidden layers:** Yup, ended up doing 1 hidden layer since its simple. Made it easy to hard-code backprop formulas.
- **Hidden neurons:** 32 neurons, actually, since it was easy to add more and in initial testing it seemed to improve things.
- **Activation function:** Sigmoid, since again it was simple for backprop.
- **Learning rate:** $\eta = 0.1$ Seems a good rate to have chosen.
- **Weight initialization:** Uniform random in $[-1, 1]$. Didn't really experiment too much with this, but seems fine.
- **Epoch Count:** Decided to go with baseline of 10,000 but can stop early if validation shows no loss improvement after 500 epochs.
- **Train/Validate/Test Division:** Decided to go with a 60/20/20 split since this seemed to be better than others we tried like 75/15/15.

4 Data Collection (Step 9)

For each of the nine datasets, the network was trained over 20 independent runs, each with a different random weight initialization, keeping everything else the same. The data was, as mentioned before, split 60% training, 20% validation, and 20% test. The validation set was used to monitor overfitting, and we only tested after the entire training run was complete.

The following data was collected per run:

- Training loss per epoch, to observe convergence behavior.
- Training, validation, and test accuracy at the end of training.

We report mean and standard deviation of test accuracy across the runs. Since we randomly initialize the weights, we can't tell if our ANN is good or not from just a single run.

We also provide a graph of the loss across epochs.

We hope to determine if our ANN is any good from this data, specifically the mean and standard deviation across runs. As well we hope to see whether our ANN converges quickly, or in a consistent manner.

5 Data Collection

Dataset	Mean Accuracy	Std Dev
Gaussian 2D Wide	0.98	0.01
Gaussian 2D Narrow	0.91	0.03

Table 1: Test accuracy across all datasets, 10 runs each