# CS559 Machine Learning: Project Proposal

by Ayan Sengupta, Ramana Nagasamudram

## Project Idea

We plan on exploring the task of *One Shot Learning*.

Conventional deep learning relies on having large quantities of data. The idea here is to tackle the problem of learning from as little data as possible. In the one shot classification task, we are given only one example of each class to learn from.

The application area we are interested in is image classification. The model we will build would ideally be able to classify images in a dataset after being trained on only one image from each class.

There are a few ways we can implement such a model. We plan on either using a *Siamese Convolutional Neural Network* or a *Memory Augmented Neural Network*. Papers on these models applied to one shot learning are referenced below.

## Datasets

Omniglot

MNIST

Yale Faces

Image-Net

## Papers

One-Shot Imitation Learning

Matching Networks for One Shot Learning

Siamese Neural Networks for One-shot Image Recognition

One-shot learning by inverting a compositional causal process

## Experiments

We plan on first building a one shot classifier for the Omniglot dataset. We will compare this model to a K-Nearest Neighbors classifier.

We would also like to train our one shot classifier on a minimal subset of the MNIST dataset (10 images - one for each digit) and compare the accuracy with MLE and KNN based classifiers. Here we may consider the task of few shot or $k$ shot learning, wherein we have $k$ examples from each class ($k$ being a small number). A comparison between one shot learning and $k$ shot learning can be made here.

If we have the time, we would like to build one shot classifiers using a few more datasets (especially Yale Faces).

## Software

We will be using Python as our programming language. The libraries we will use are Scikit-learn and Keras. Scikit-learn will be used to implement the simple methods that establish a baseline accuracy for our task. Everything else will be implemented using Keras (with TensorFlow as the backend).

We might use the PyTorch library over Keras if we feel that it might make the implementation better in terms of speed or readability.