ECE 532: Term Project: Classifying Images in CIFAR-10 Dataset Using Supervised Algorithms

Final Report

Written by:
Nicholas Chelales

(a) N. Chelales

University of Wisconsin-Madison
October 21, 2020

Contents

1 Project Dataset

The project dataset to be used is the CIFAR-10 dataset (found here). The data is publicly available and consists of 10,000 32x32 RGB images, each belonging to mutually exclusive classes (1 of 10 possible options, see the dataset). The goal is to test the three different algoithms, K-Nearest Neighbor, Kernel Based SVM, and convolutional neural networks and measure their performance in classifying the images as one of the 10 possible classes.

2 Algorithm Investigation

2.1 K-Nearest Neighbors

The k-nearest neighbors algorithm works by making a prediction based off of the closest K training examples in the N-dimensional feature space (where N is the number of dimensions in the feature set). There are many different distance functions that are used in the k-nearest neighbors classification such as city block (Manhattan), euclidean distance, cosine distance, and mahanblois distance. A subset of these distance metrics will be selected and the performance will be compared among these metrics. In addition to distance metrics, the number of neighbors (K) will be a second parameter and performance will be compared among different number of neighbors to see the best performing K.

2.2 Kernel Based SVM

The kernel based SVM is a linear classifier which uses a kernel to convert a N-dimensional space into a higher dimensional space, making it easier to distinguish between different classes compared to lower dimensional spaces that can sometimes be more difficult to separate with a linear boundary. There are many different types of kernels, such as a linear kernel, polynomial kernel, and and radial basis kernel. These different kernels will then be compared and performance will be evaluated to find the most optimal parameter. In addition, optimization and regularization parameters (such as gamma) can be used as parameters that can be tuned and evaluated.

2.3 Convolutional Neural Network

A convolutional neural network is a neural network that uses a filter that is convolved (filtering) the input image in order to get a result that can then be classified as one of the possible classes. The neural network is trained and the filter kernel is tuned appropriately to get an end result kernel that is applied across each layer. There can be many layers in a neural network, so the number of layers will be one of the parameters that is explored to see the most optimal number of layers. Another parameter that will be explored is the loss function, which is used to help tune the filter and avoid overfitting. There are different loss functions that can be used, and several functions will be evaluated to determine which is the most effective.

2.4 Performance Evaluation

Performance will be evaluated using three different loss functions: 0/1 Loss, squared loss, and hinge loss. These loss functions will be used to get a representation of performance on training data (for baseline model performance), and will be used with k-fold cross validation on the testing data to get a measure of the robustness of each model/parameter experiment.

2.5 Features and Feature Selection

There are several possible features from an image that can be used as possible inputs for the classification algorithm. Primarily, the raw input pixel values contain significant information that can be used for inputs to the models (although there is room to add features by preprocessing some of the images with filters such as horizontal or veritcal line filters, extracting color schemes in the HSV or LAB colorspace, augmenting the dataset, etc. if necessary to achieve better results). For K nearest neighbors, the raw pixel values will be used as the features, and each test image will be compared with the all training images using the specified distance function. For SVM, we can treat the input data as the matrix of values in the image, flattened into a 1D array (basically like reading a single vector of values, rather than a matrix). For the convolutional neural network, the features will also be the raw input pixel values, as the network will use these values as it builds the optimal filter.

2.6 Timeline

2.6.1 Week 1 Oct 26th

-Build project environment in PyCharm -Download and Import Data

2.6.2 Week 2 Nov 2

-Implement and test KNN algorithm

2.6.3 Week 3 Nov 9th

-Implement Convolutional Neural Network

2.6.4 Week 4 Nov 16th

-Implement Kernel Based SVM -Submit Update 1

2.6.5 Week 5 Nov 23rd

-Continue to implement Kernel Based SVM

2.6.6 Week 6 Nov 30th

-Buffer week for debugging/unexpected problems -Submit Update 2

2.6.7 Week 7 Dec 7th

-Write Report

2.6.8 Week 8 Dec 14th

-Submit Report

2.7 GitHub Page

Access here