Analysis of CIRFAR-10 using Convolution Neural Network

William Duquette and Malcolm McDonough

June 22, 2023

Introduction

It is generally agreed that there are methods for image recognition that are better than others at various tasks. Although there are numerous approaches to conducting image classification, we have chosen two common supervised learning techniques to compare. This paper analyzes two different approaches to image classification: Dimensionality Reduction with a Support Vector Machine and a Convolution Neural Network.

Data

The data set that we will be working with is the CIFAR-10 data set. It contains 60,000 32 x 32 images of various vehicles and animals. All the images in this data set fall within one of 10 categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. In order to work with the data, we need to import the data set using a Keras data loader. We import the data into two sets, a 50,000 image training set and a 10,000 image testing set, separated at random. Then, we print out one of the images at random so that we can get a better sense of the data we are working with.

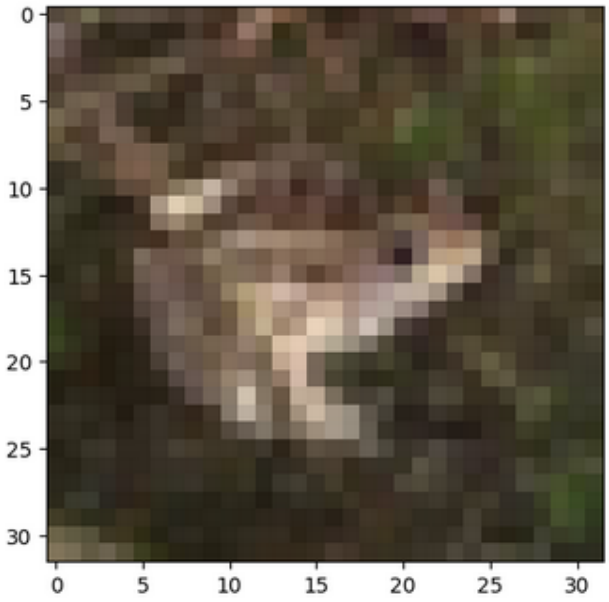

Figure 1: Example Image

Dimensionality Reduction and Support Vector Machine

First, we perform dimensionality reduction on the data and then use a Support Vector Machine (SVM) to classify the data. In order to avoid the curse of dimensionality, we need to reduce the dimensionality of the data, allowing us to more effectively train our model on our data. There are two possible approaches in this instance: Principle Component Analysis or Linear Discriminant Analysis. Both could be used in this instance, but PCA tends to be used in an unsupervised setting (no labels). In this case, we have class labels, which allows us to reduce dimensions in a way that is also beneficial to the classification process. Therefore, we conduct Linear Discriminant Analysis (LDA) as we are working with supervised data. Then we fit an SVM to classify our data.

We perform LDA on our data to maximize the between-class variance and minimize the variance within each class. This helps our model to better differentiate between the different classes of our data set, improving its accuracy. We have to choose how many dimensions we want to reduce our data down to, and in this case, we set the number of components to 9, as we are working with 10 classes. It is best to set the number of components to be one less than the number of classes. To confirm that the dimensions of our data have decreased after running LDA, we can compare the total number of dimensions of a data point from the original data set with a point from the post-LDA data set. As shown below in Figure 2, we have drastically

reduced the dimensionality of the data from 3072 to 9, which will greatly improve our computation time and the accuracy of our model. This is crucial as we will be working with SVMs, which have notoriously slow computation times.

Original Dimensions	3072
Reduced Dimensions	9

Figure 2: Dimensions of Data Before and After Reduction

Now that we have reduced our data's dimensionality, we can begin to fit our SVM. The kernel that we will be using is the Gaussian Radial Basis Function. This kernel is bell-shaped, with 1 being close to whatever landmark we are working with and 0 being far away. In this case, the SVM uses the kernel to transform the data so that it is linearly separable, at which time a Maximum Margin Classifier will be used to find a multi-dimensional linear separator. Once we fit the SVM, we can use it to predict an image's class on the test data set. In Figure 3 we can see the SVM trained and evaluated.

# of Test Samples	10000
# of Errors	6491
% Accuracy	35%

Figure 3: Performance of Support Vector Machine

Unsurprisingly, the SVM is not very successful in predicting an image's class. In fact, the SVM only has an accuracy of 35%. The poor accuracy suggests this data may be too complex to handle with an SVM. So, to continue, we will explore this process using the Convolution Neural Network.

Convolution Neural Network

In order to better classify each of the images, we will use a common technique in the field of image recognition: utilizing a Convolution Neural Network (CNN). A CNN is a deep-learning structure that is frequently used for image recognition, primarily because of the way the convolution layers work in tandem with each other. Essentially, convolution layers are a set of filters that are chosen to move through the image and analyze a small portion of the photo (in our case, a 3 pixel x 3 pixel space); the values within these filters are then used as the weights of the neural network. A detailed discussion of CNNs is not possible within the scope of this paper, however plenty of resources are available which go into great detail. For more information on the topic, we suggest Chapter 14 of Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow by Aurélien Géron.

Since we are working with a Neural Network, it is best practice to have a cross-validation set so that we can test the model against it during training, avoiding over-fitting the network to the data in the training set. The testing set and the cross-validation set both have 10,000 images. Due to the size and complexity of the data set that we are working with, we have to construct a relatively complex CNN. This Neural Network has 13 layers, with the layers doing various tasks. Let us walk through the architecture of this CNN:

1. First, we create a Sequential Keras Model. The first layer of this model is a Convolution layer. We choose a relatively small (3 x 3) filter to start and have 32 of them. For the first convolution layer, we have to define the input shape, which in this case is $[32, 32, 3]$ because the images are 32 x 32 pixels and use 3 colors (red, green, and blue).

2. Next, we include a pooling layer, which uses the default pool size of 2, meaning each spatial dimension will be divided by a factor of 2.

3. We repeat the above structure twice: two convolution layers followed by a pooling layer. One difference between each group of convolution layers is that the number of filters doubles (32, 64, 128) as we move deeper into the neural network towards our output layer. Doubling the number of filters after each pooling layer is widely considered to be best practice because the pooling layer divides each dimension by a factor of 2, which allows us to double the number of feature mappings in the following layer without being concerned about the increasing number of parameters.

4. Next, we have a flattening layer before our dense layers. This must be done because the dense layer expects a 1D array of features for all of the instances. We then construct our fully connected layer with one hidden dense layer and a dense output layer. We also include two dropout layers, which prevents over-fitting to the data. The final activation function used within our dense output layer is a soft-max since we are working with the classification of multiple categories.

Figure 4 is a table that summarizes the Convolution Neural Network, as well as the number of parameters in each layer.

Layer	Output Shape	Param #
Conv2D	$(none, 32, 32, 32)$	896
Max Pooling	$(none, 16, 16, 32)$	0
Conv2D	$(none, 16, 16, 64)$	18496
Conv2D	$(none, 16, 16, 64)$	36928
Max Pooling	$(none, 8, 8, 64)$	0
Conv2D	$(none, 8, 8, 128)$	73856
Conv2D	$(none, 8, 8, 128)$	147584
Max Pooling	$(none, 4, 4, 128)$	0
Flatten	$(none, 2048)$	0
Dropout	$(none, 2048)$	0
Dense	$(none, 128)$	262272
Dropout	$(none, 128)$	0
Dense	$(none, 10)$	1290

Figure 4: Summary of Network

Results

To measure accuracy, we are using sparse categorical accuracy as the classes are mutually exclusive, and the information for the prediction is one integer rather than a whole vector. This measure is extremely similar to the standard accuracy measure and should be thought of as the same in this context. The CNN successfully classifies the images, with approximately 75% accuracy on the test set with 30 epochs. For a data set this complicated, this is an encouraging result. It does not appear that the model is too large or too complex. We do not see good learning achieved quickly, suggesting that the model is the proper size. Looking at the plot below, we see that to prevent over-fitting to the data, we should use no more than 30 epochs to train our model. Once we go over 30 epochs, the accuracy on the cross-validation set decreases or does not change. This can be seen in Figure 5, reflected in the solid blue `val_loss` line, which shows how the accuracy on the validation set begins to decrease as the number of epochs increases over 30.

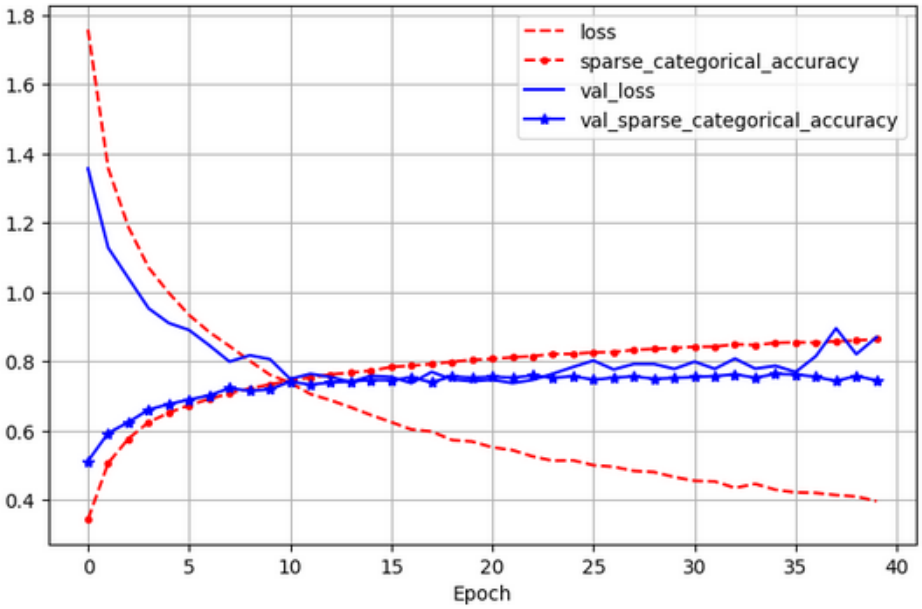

Figure 5: Training Results for CNN showing a decrease in accuracy past 30 epochs

Below is a graph that visualizes the training process and the accuracy obtained across epochs when we train with 30 epochs. Looking at the bottom left of the plot, we can see that the accuracy of our model increased drastically in the first few epochs. It only took six epochs for our model to reach 70% accuracy. The rest of the epochs led to small incremental changes to our model that fine-tuned it so it could become more accurate. You may see that the model's accuracy on the validation set dipped slightly after the 20th epoch, but after ten more epochs, it did increase its accuracy further, albeit only slightly. If computation time is a concern, performing only 15-20 training epochs may be preferable.

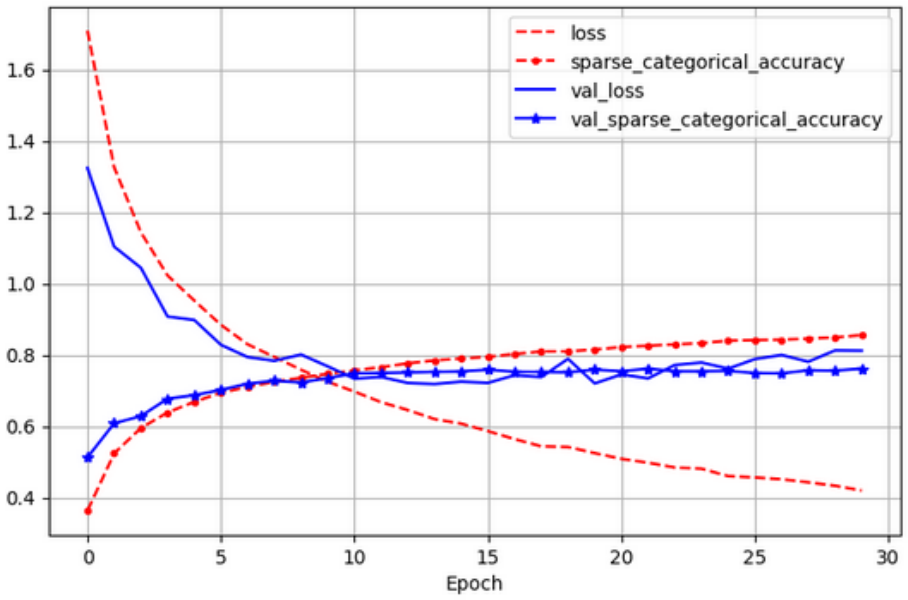

Figure 6: Training Results for CNN

Conclusion

This paper compared different methods for classifying the CIFAR-10 image data set. First, we compared Dimensionality reduction with a Support Vector Machine and a Convolution Neural Network. As we have demonstrated, a Convolution Neural Network correctly classifies the most images, achieving an accuracy of just over 75%, while the SVM approach only achieved an accuracy of approximately 35%. Although the Neural Network was more complicated than the SVM approach, it provided significantly greater accuracy with only a small additional cost in computation time. Therefore, we have concluded that the Neural Network approach is the better of the two approaches tested in this paper.