George Mason University

Learning From Data Fall 2024

Computer Exercise #8

Assigned: November 20, 2024
Due Date: December 05, 2024 Part 2

Computer Exercise 8.2 (Image Classification):

Now that you have some familiarity with CNNs, you are now tasked to design an image classifier using the dataset provided with this exercise. You goal is to learn a classifier that you believe will perform the best on a test set that will be used to evaluate your classifer.

(a) The dataset contains labeled images of dogs that are stored in an `.npz` file, a compressed archive format used by NumPy. There are 3,132 color images scraped from the internet and resized to 224×224 to align with the size of the images in the ImageNet dataset. Each image belongs to one of twenty different classes, with class labels zero through 19. The name of the file is `train_data_224.npz` and is 471.5MB in size.

(b) The dataset may be read and put into two arrays, `images` and `labels`, as follows:

```
# Load the .npz file
data = np.load('train_data_224.npz')
# Access the arrays stored in the .npz file
images = data['images']  # Access the images array
labels = data['labels']  # Access the labels array
```

(c) Before preceeding, you may wish to become amiliar with the dataset.

(a) Look at some images,

```
plt.imshow(images[0], interpolation='nearest')
plt.show()
```

(b) Examine the number of samples in each class

```
unique_labels, counts = np.unique(labels, return_counts=True)
# Print the counts
for label, count in zip(unique_labels, counts):
    print(f"Label {label}: {count}")
```

and determine wheter or not the the dataset is balanced, and if there are enough images in each class necessary for the type of classifier you will be learning.

(d) Consider what data preprocessing tasks you think are appropriate for this problem.

(e) You may wish to consider using `tf.keras.preprocessing.image.ImageDataGenerator`.

(f) Your *classic*-style classifier of convolution and pooling layers followed by one or more fully connected layes may not perform as well as you would like, so you may wish to consider using transfer learning.

What you are to submit

Your report should be a Jupyter notebook with the following:

1. A brief discussion of the types of classifiers you considered and how well they worked.
2. A detailed description of your final classifier, the architecture, the preprocessing and how the training is done.
3. A learning curve, the classification accuracy, and a confusion matrix.
4. The code used to train the classifier.
5. **Imporant**: You must include a code cell with the following:

   ```
   model.save("model.keras")  # Saves both architecture and weights
   ```

   This will save the architecture and paraameters of you classifier and allow for the evaluation of your classifier on the test set. You **must** submit this file along with your notebook. Failure to be able to do this will result in a 50% reduction in your grade. You **must also** make sure that this file can be used to evaluate your classifier. To test to make sure that your saved file will work, the Jupyter notebook that will be used to evaluate your classifier will be provided.