Project Report/Reflection

Anirudh Satish,
CS 181, HMC,
Prof. Dodds,

Vision and evolution:

Initially, the topic of my project was completely different. I wanted to build a Spotify application that would merge two playlists and choose songs that matched the most. I even spent a good few days getting the authentication set up, with detailed code for the same. However, I later stumbled across Transfer Learning in one of the final questions, and thought that this would be something exciting to tackle, particularly because it was something I had never done before.

Following this, I did some research into Transfer Learning itself, and realized how cool it was, and that it was something I definitely wanted to incorporate into my final project. I was not able to apply this to my original Spotify application, however I did come across a very cool idea, which is what my project evolved into, that being a dog breed detector. The final version I have now is pretty much what I envisioned after the shift from the initial Spotify application to the Detector using Transfer Learning.

I would say that this project was successful in what it was able to achieve in a short two week span. Firstly, I set out to find the breed of the dog that was the face of Doge coin, which as I said in my presentation was one of the main motivators for this project, and I did end up figuring out that it was apparently a Dingo dog, at least according to my model that was 80% confident in this prediction. Secondly, I was able to just play around with cute dog pictures, and see what my model thought, including my pets.

Progress:

After doing my research into Transfer Learning, I found some articles online that talked about using models pre-trained on the ImageNet data set and repurposing them for different use cases. One of these use cases was Dog breed detection, for which I was able to find some implementations online. The gist of this implementation was that it created an instance of a model called Xception, that was trained on the ImageNet data set, and was able to classify 1000 different objects. With the help of online sources, that are referenced below, I was able to write code that removed this top most layer of the model that had 1000 nodes for the 100 different categories, and then use the result of the previous layer as the "features" for my dog breed classifier. This is the crux of Transfer learning, where the representation of the image by a pre-trained model is used as input for a small extension model to use it for a different task. I created a wrapper model that took this representation as input, and had 121 nodes on its last layer, each corresponding to a unique dog breed that I intended for my model to classify. Another important step was

setting all the other layers of the pre-trained model to untrainable, such that only the newly added layers of the "wrapper" model were trained on each epoch.

The model was trained for 50 epochs, and reached an 80% accuracy on the train set, and 90% on the test set, which was quite respectable. In the end, I reached my goals of being able to classify the breeds of dogs when an input image of them was given to the model. I had a lot of fun developing the code/model for this project, and applying it to different dog pictures (and some not dog pictures :)).

Future work on the project:

Currently, because of the way the model is designed, it takes in any picture (which is good) but tries to classify anything as some kind of dog. i.e, if you input a picture of a human, the model would return a label of the dog breed that the human (or the pixels of the picture I guess) most resembles. While this is funny and fun to play around with, ideally this should not be in the final version of the model, or there should be a switch which when turned allows you to only predict dogs for images that have dogs. To solve this, we could use Transfer Learning again to have intermediate layers (after the initial pre-trained, before the dog detector) to test for the presence of a dog in the picture, and based on this, output what is required. Further, with more time I might try different pre-trained models, and a variety of combinations of hyperparameters to try and get the best performing model, as now it was pretty much a guessing game. This is because the training process was quite long, and intensive even with transfer learning, and required large amounts of memory and time.

Potential Different Approaches:

One thing that I would have liked to add was to train a detector from scratch (not using transfer learning) and compare its performance to the model that I have now, to illustrate the advantages of Transfer learning, and potentially highlight the positives and negatives of both sides. Other than that comparison, I do not think there is anything I would change. This was a great project to be able to dive into Transfer Learning, and I was able to do it successfully with my knowledge from the class, and exposure to the different libraries over the course of the semester.

Final Thoughts:

Overall, I am really happy with how this project turned out. Not only was I able to learn about new things and apply them, but I was able to get a system working successfully, and I am really happy about that. I want to potentially explore the Spotify application later during the summer, as it is also really interesting and I can see myself using this a lot as I combine playlists with my friends, who have different music tastes.

References:

1. https://towardsdatascience.com/deep-learning-build-a-dog-detector-and-breed-classifier-using-cnn-f6ea2e5d954a
2. https://medium.com/@iliazaitsev/dogs-breeds-classification-with-keras-b1fd0ab5e49c
3. https://medium.com/@imBharatMishra/dog-breed-classification-with-keras-848b9b1525c1