Jeremiah Orzales

Professor Dongsheng Che

Machine Learning Internship

December 17th, 2021

Internship Report: Machine Learning

**Introduction:**

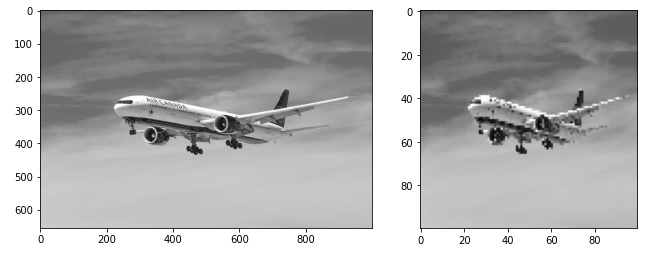
At the beginning of the project, I had almost no knowledge of machine learning and how to program it in Python. It was unclear to me as to how machine learning works in its full detail and what are the tools that I needed to use to start. Researching how machine learning works had to start from somewhere and I decided to do something complex, but also easy for me to tackle. Deciding what type of project that I needed to tackle was simple and very straightforward. By using a dataset, I would have to learn how to use it as the source for a machine learning program to train from. In this case, I used a reasonably large dataset that was broken down in categories. This dataset had images of various types of aircraft and missiles that were taken from Google Images. The categories were airliner, missile, rocket, fighter, drone, and helicopter. All the combined images would reach over one thousand images per category. Next, I decided to use TensorFlow as tool to use for my machine learning program. With TensorFlow, there are a wide range of online information on how to use it and enough troubleshooting information for me to learn from. There were issues that I ran into that were more complicated than the others, but that gave me experience on what to do each time I ran into any issues involving the project.

As I had mentioned, the project had run into problems, from which I could not resolve. This does not mean that I gave up on those problems, but I did gain some insight on how TensorFlow works. For a potentially hardware intensive tool, TensorFlow works well on a computer’s CPU, but it works even better on a Graphics Card. During my research, I attempted to use TensorFlow with a Graphics Card, but there were driver installation issues that I could not resolve. This almost jeopardized the project, because my I had to do multiple reinstallations of Anaconda and various other drivers for my computers. This was one of the issues that I had to move away from and continued with my goal in learning how machine learning works in TensorFlow. Another problem that I ran into was being time efficient and knowing my limitations on matter. During the beginning of my project, I decided to add more to my project by using OpenCV as an additional tool to detect moving aircraft. It was at that time I became aware of the difficulties of adding more responsibilities on top of an existing project. Those were some of the issues that I ran into that were outside of the actual project. With that mentioned, it is now time to discuss the results of my project.

**Module 1 - Data Preparation:**

My project involved a few thousand images of varying aircraft types which I may have no time sift through. Validating which images are good or not was one of the problems that was encountered during the first few days of the project. It was an issue because some of the images were of different file types. The most common and desirable image file type was a JPG file type. By using a try-catch block, we would only pass the undesirable file types and move on with the data processing. Later, I found that each category had some images that were miscategorized or does not belong to the whole dataset at all. For example, there was a picture of a person in the airline category when I was speed-scrolling through the images. However, that problem may have affected a future module involving the Convolutional Neural Network.

The first module was Data Preparation or Data Processing where each category had to be accounted for in the program. By directing the machine learning program to a file path, I was able to save the categories and images into training data. Before doing this, I had to cut down the resolution of each image and turning their color from RGB to Grayscale. Data storage is saved by doing this and in our case the RGB data is not needed. It is also less computationally heavy to process Grayscale images compared to RGB images. The same also goes to the resolution of the images. Since I am using different types of aircraft, there were features on an aircraft that were important for classification. The image size could not be too low, otherwise the program will miss important details on an aircraft. It was found that going lower than a one hundred by one hundred resolution was enough to train the program. Going higher than the specified resolution would take a long time for the Neural Network to train. An image of a sample is shown below from what it was before (left) and after (right) processing.



At first, I worked on the project as modules to divide the work evenly to efficiently work on one thing before moving on to another. This also made it flexible enough for me to program each module to talk to each other through a saved file. The categorical data was saved using Pickle, so processing the data only had to be done, when necessary, prior to creating a Neural Network. We could use the Pickle saved files for any Neural Network program if needed and this made it less computationally expensive to do than doing it every we execute the project.

**Module 2 - Convolutional Neural Network:**

For the first phases of this module, I used only two of the seven categories within the dataset to test how Convolutional Neural Network works in TensorFlow. By doing so, I was able to see what my program needed to execute efficiently. The Convolutional Neural Network (CNN) is in the feature extraction part of the program where it needs a layer for the Convolution and Max Pooling. Average Pooling could be used any model but for this project, Max Pooling was used instead. Convolution happens before the Pooling it is the one that looks for the features of our aircraft. Pooling come after the Convolution to create a feature map that is less computationally expensive to execute. It takes the values of one Convolutional feature map and take the largest value within the Convolution feature map. This also reduces overfitting where any noise in the image could impact the results of the model. Having more than one set of Convolution and Pooling is possible. In this case, I have two sets of Convolution and Pooling where I flatten their output afterwards.

At the later stages of the project, I had started to use more than two categories after familiarizing myself with CNN and how the Keras and TensorFlow functions work. While building the program’s model, the previous code for using two categorical dataset was incompatible with more categories. It was either an anomaly or a detail that I had missed within the program. Rewriting the CNN module was necessary at that point, and I was able to simplify my code for better troubleshooting. Prior to adding more categories, the program used Binary Cross Entropy as to what it has now which is Sparse Categorical Cross Entropy. This was the start of the problem in the prior code as it only gave the model a constant accuracy per epoch.

Moving on to epochs, running the training data more than it needs is not always good. TensorBoard was used to visualize this so we could understand that in certain circumstances, having more epochs does not mean that the machine learning model will be accurate. The model will reach a point where the accuracy will stay around a percentage rather than go up. For the current machine learning program, twenty-five epochs were sufficient to do the job right. Having thirty to fifty epochs only negatively impact the run time of the program and the model. The plots below are built in the program and show the Loss and Accuracy of each epoch. Accuracy started to platau after the fifteenth epoch.

Chart, line chart

Description automatically generated

**Module 3 - Using the CNN to Classify an Image:**

After creating the CNN model for the machine learning program, it was time to classify an input image. An image must be an aircraft of any type in a JPG format. The input image would go through a series of functions before being used as an input to the CNN model. Grey scaling the input image first and resizing the image to the program’s image size had to be done first. When this was done, it was sent to the CNN-predict-function in TensorFlow for classification. There were a few issues when classifying the images which was expected. One issue was that classification where images were misclassified as something else. Of course, this was to be expected, but there could be underlying issues with the dataset. For example, there were missiles being classified as a helicopter or a drone. Airliners would be classified as drones or helicopters being classified as airliners. The reason could be that the dataset had images that were unrelated to the categories that they are in. Another reason could be that the CNN would need more layers. When building the prediction module, I decided to exclude a dataset from the program. This was the “rocket” category because it resembles a missile in many forms and it shares the same features as a missile, excluding their relative sizes. By excluding the “rocket” category, we reduced misclassifications of other aircraft types. At times, an image of an airliner would be classified as a rocket.

In the prediction module, I have added a function to show an image input to see what is being classified. This confirms what type of aircraft it is and what it should be classified as. It was good addition to the program, since it would be beneficial for a user to recognize what features that are similar to other image input. This way, a person could analyze the mistakes of the machine and make other adjustments in the CNN or the dataset.

**Goals for the Future:**

Continuing this project would be a great way to learn more about TensorFlow and become more advanced in machine learning. There are more to be done with the project, such as running the program with a Graphics Processing Unit instead of the CPU. Also, adding and deleting samples from the dataset for better classification. Doing a project on how to use real time camera feed to classify different aircrafts would be a greater goal for this project. Since this project proved to be fun and beneficial for my own experience, I will continue to develop the machine learning program into something tangible. TensorFlow has proven to be the most powerful tool that I could have ever used in my development experience. Being able to use it in my code along with other data science tools will be useful in my career as a developer.

References:

TensorFlow CNN:

<https://www.tensorflow.org/tutorials/images/cnn>

TensorFlow Python Functions:

<https://www.tensorflow.org/api_docs/python/tf>

Keras Python Functions:

<https://keras.io/api/models/>

MatPlotLib Python Functions:

<https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html>

TensorFlow GPU Support:

<https://www.tensorflow.org/install/gpu>

Keras Image Classification:

<https://keras.io/examples/vision/image_classification_from_scratch/>

Kaggle Aircraft Dataset:

<https://www.kaggle.com/eabdul/flying-vehicles>