Graduate Report on CoreMl – Paladugu Sai Sankirth Reddy

Data equals Knowledge. Data is everywhere and plays a very important part in day-to-day life. The term ‘small data, for example, phone numbers of family, refers to small datasets which humans can remember and analyze. ‘Big data’ refers to large datasets that may be structured or unstructured and require a machine to store and analyze. Data can be in any form like image, video, text, number, and bits.

If we consider our daily activity data for example Weather. Weather plays a crucial role in this world. Due to unpleasant weather, there is a chance of losing many lives. If we can predict the future weather by making assumptions about past weather and taking precautions before the unpleasant weather, there will be a lot of advantages for livelihoods. How can we solve this problem “By training existing data and predicting future data”? Machine learning is the solution, Machine learning is the phenomenon of predicting or classifying future data in terms of old data. In Machine Learning, the most critical assumption is that training and testing datasets should have similar distributions. The model will be effective if the new test data is similar to the past data on which the model was trained. If there are substantial differences between the training data and the testing data, the machine learning algorithm will generate results that are not very accurate. As we know the data can be represented in different formats like images, videos, vectors, texts, etc., Let’s look at data that can be represented by images.

Image data is nothing but scanning a surface with an optical or electronic device, common examples are scanned documents, satellite images, and aerial photographs. Machine learning can also be applied to Image data. For example, if we have tons of mixed animals and humans photographs where we need to classify them. If we can train the machine learning model with existing data of human photographs and classify them as humans same as animals. The model can classify very much faster than manual classification. Above is the most generic example, but there are examples as finding the percentage of rust for a steel component takes a lot of manpower and time, so machine learning will come to the action.

Let us discuss how image data will be represented on the computer. Every image is broken down into pixels. Pixel is the smallest component of the image, and every pixel will have a respective unique value. For an 8-bit image, the pixel value will be between 0 and 255.

Image processing is the phenomenon of converting images into digital forms or matrices, where we can perform operations and we can get useful information from them. Usually, there are 5 important types of image processing Visualization, Recognition, sharpening, Retrieval, and Pattern Recognition. Typically, machine learning needs a pipeline to complete tasks like data preparation, Feature extraction, model training, and Prediction or classification. Initially, ML algorithms need high-quality data to train the model, so that the results will be accurate. We should make sure the images are well processed and in case of classification well classified into respective categories. Then Computer Vision comes into action, it is the field of machine learning which can learn the image data. Preprocessing steps include image data converted into the same format unnecessary parts are cropped in the image and transferred into numbers so that machines can learn, Previously I had discussed pixels, this comes into the picture, so these are features of the ML model. With respect to these features, new data will be predicted or classified.

As we have learned how machine learning is used and how much the data comes into play in building these models. As the amount and variety of data increase, the prediction made by these models also increases. To acquire more precision, we need a large amount of data, after training these data according to our requirements we would deploy this model and use it in our intended field. But usually, such models are very huge, as the amount of data required for that is very large and which tends the models to have a large amount of memory. As such types of models are getting more prevailed and we have been using those in all the websites, for these if we use the computer it can handle such type of request and give us a seamless transaction between this models, As of now Mobile application are regularly used by many, Although the computation power for the mobile devices have been increasing there are still many limitations on those, such as we can extract 100% performance from the mobile devices as these would affect the battery performance of the device such limitation should be taken into account by the mobile application developer also the amount of memory, computation power is also limited compared to computers, the developer should take this into account while building the model.

Although mobile devices are widely used there are many challenges in developing Machine Learning applications (as all applications are using these models to improve their user experience) as such these types of models are very essential.

To tackle these problems and give support to these types of applications in IOS, apple has come up with the CoreML tools, which allow developers to have access to these machine learning tools, and allows them to develop models required for their needs.CoreML is a developer tool that is bundled in the Xcode. You can access coreMl in the xcode developer tool.

CoreML is a development tool that is provided by Apple. inc to help promote and give easy access to the machine learning models, and also help them in building such models. As mentioned above the limitations of mobile devices, coreml helps them in tackling those issues and helps developers in deploying smooth machine learning applications without making the applications too heavy. It not only helps the development of the machine learning model but also converts the other machine learning models to be converted into IOS friendly or swift friendly models, which can directly be used in their app.

One of the features of CoreML is that it would allow large models which have been trained by a large amount of data to be bound in a very small model where the developer can deploy it directly to the users in that app. Where such takes very less memory, thus allows the applications not to be very heavy in this way developers could use whichever-type of models they want to use such as we have google nets, amazon's acs and many such machine learning models which can be converted to iOS-friendly through CoreML.

Another feature of coreML is that it gives us many applications which cannot be obtained coded in the app or are impossible to achieve in the code. It is a prediction-based model where it analyses the data we give and gives out the most appropriate results based on this prediction.

CoreMl is a domain-specific framework, where it uses vision for analyzing images, neural networks for text, speech for converting audio, and Sound analysis for identifying sound in the audio. These are some of the frameworks that core ml uses to train and build the model.

In the Output part the coreml gives out an API model where it has been trained and can help us in calculating the probabilities as it is a precision model, it calculates the probability of the data getting matched to the labels we have given, in such way whichever the label has the highest probability than it would give out that result, it not necessarily be right all the time.

The precision of the model depends on the type of data we give to train and how well they have been classified, based on these factors the precision of the model will be decided. If we have different types of data, and the amount of data is large it would be possible to train the model to be more precise which helps it in increasing its precision.

So far, we had an overview of what is machine learning, how much is data important for these machine learning models, and the basic overview of what is CoreMl. From here on out we would be going a bit deep into each type in CoreMl.

In CoreMl we have a total of 13 types of models in which each type specialized in their respective cases, each type made be useful in a couple of other types depending on the type of data we give to train.

Of those 13 types in coreMl, Image Classification is one such type where they need images as input and output would be in the format of strings. Where the developer trains the data which is images he has in the coreML (where it uses image classification), Apple has given many resources for such operation where we have a vision which helps in re-sizing the images in such a way that it would be accepted by the model which is a native library which comes with xcode if we want we can use many different third party resources for such operations but it is noted that the performance of native library folder would be more higher has they are tailored made for that particular application.

We have Object Detection in CoreML, which helps the data to classify the objects in the images, we have a similar way in the image processing, but object detection learns a lot more details than image classification and only focuses on the types of objects which is not the case for image classification. We can use vision with object Detection which helps in improving the focus and helps indirectly identifying the object.

We also have style Transfer which helps the developers in making different shades or getting cool filters to their images, which is also used with the help of vision to have live access. As vision helps all image Machine learning models to have live access and decrease the time for an image to get to the datamodel, in this way it helps in giving out more user-friendly performance.

We Hand-Pose Classification is a model where we need to train the model with images of people giving out different hand gestures and categorizing them into different groups. This model helps in recognizing different hand phoses given.

Now we have Action Classification, where it detects all the objects in the video and tries to recognize them depending on the type of actions the object takes in that video, where we would be training the data with multiple videos performing different actions and labeling them according to that.

We have Hand Action Classification, which takes the hand action and recognizes it, First, we need to train the model with multiple videos by labeling them into a different category and training them, accuracy depends on the type of data and amount of data

As such we have Hand Classification, Activity Classification, Sound Classification, Text Classification, Word Tagging, Tabular Classification, and Tabular classification are some of the models which are some of the machine learning coreml models which find with the bundle of Xcode.

All the above models we have discussed are only the models which are native to Xcode, we have many coreMl models which are known as the third party which would be like the native application. The usage of such models is like the CoreMl (Create ML) models which are generated through developer tools in the xcode.

These models give a coreml API that is IOS(Swift) friendly, where developers can easily integrate these into their apps.

To give a more accurate explanation of how CoreML works let me give an example of my Graduation project, were asked to build an application that detects the rust and coercion percentage in the railroad boxcar.

In this, the first step would be classifying the model which means categorizing or deciding on the categories in the data which we give as a label in a later stage. After categorizing the model into five different types we started collecting the data through different online sources.

For the five categories I have decided on Great_Condition where the rust and coercion are very minilar or less than 10%, Good_condition, where we have rust and coercion, are 10-20%, for average_condition we have 20-40%, In faulty we find images with 40-60%, worst_condition we have more than 60% of rust and coercion.

After deciding on the above characterization, I have started collecting the data (that is images) from many online sources. In this way, I was able to collect 500+ images that have been evenly distributed between three categories.

It is always important to note that the data which we are giving should have labels and the data on each label should be evenly distributed, if we have 1000 images in one category and 10 in another category then the model would be getting very biased which would be giving out the wrong predictions.

As coreMl model works on the bases of prediction as it is a prediction model and decides in which category that image may fit. As such it is important to have equal distribution among different categories in the Coreml data set.

Also, when training the model we need to give a unique test data to validate your model, if you won't give the unique test data then validating your model would be very difficult, and very difficult to give out accurate results on how that model works, if you're not satisfied with the model you just train you could train another model with similar or different settings and data.

While training the model we have many options such as iteration where if we increase the number of iterations the model would be learning a lot more from the data, if we give less our model would not be learning anything in such case, we need to be mindful and tactful how many iterations we need to give it to the model to achieve an accurate result.

After we get the expected results, we would be going into a preview session to check how well our application performs, if it is satisfactory then we can go to the output section and export it and it would be ready to use in our app or else we could repeat the above steps.

This is the way coreMl works with the native library in XCode. Also for other models (third party), we would be getting directly from their particular website after downloading we could directly use it but it's difficult to find the exact model that we are looking for and have desired type of data and class labels are next to impossible

References:

1. **https://developer.apple.com/documentation/coreml**

2. **https://developer.apple.com/documentation/swift**

3. **https://apple.github.io/turicreate/docs/userguide/style_transfer/**

4. **https://developer.apple.com/documentation/technologies (total documentation)**