Image Rotation Prediction Using Neural Networks

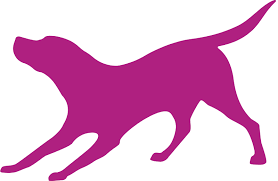
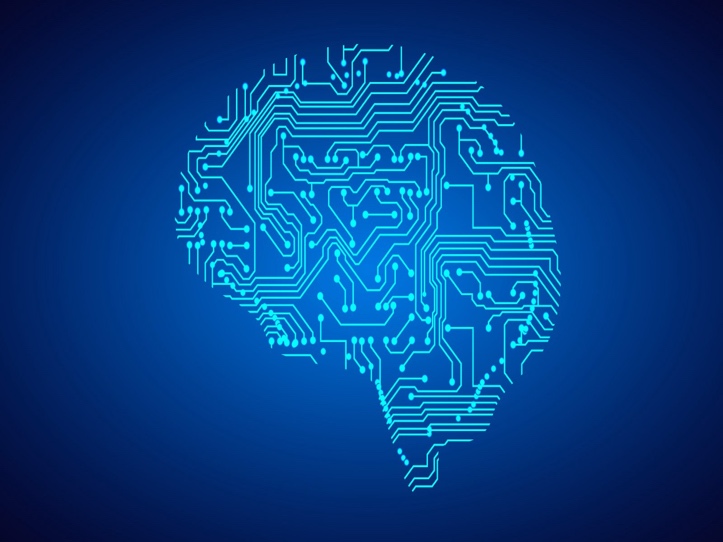


Image rotation prediction problem is solved using two frameworks. First Turi Create which is owned by graphlab and is used to create machine learning models for apple devices. Second is the widely used TensorFlow. Please refer readme.md for testing and evaluating results.

Data Set:

The Data set used to train is from publicly available images. It was hard to get the dataset containing the rotated images with the labels. I have prepared own dataset to train and test images

Turi Create:

Turi create provides a high-level API, which handles the complexity of all the neural networks layers. It chooses the best algorithm for specific problem unless you specify.

It uses vision feature screen model which is just like resnet-50 or squeeze net. Preparing data set was a huge task. For training neural network and giving two images as input was a huge task. I have merged two images one original, and another rotated into a single image file to give input as single file

I have used 4 classes for classification that is 0, 90, 180 and 270 rotation. Separate data set is created to train this model which is saved as coreml\_data. All the data is in greyscale as it restricts the number of inputs to the training. The size of input image is 200 x 400 as two images are merged. Training takes roughly 10 minutes to train 12000 images. Size of trained model is less than 5 mb. Model accuracy is about 0.923.

Note: All the rotations are considered in anticlockwise. It might help to solve confusion between 90- or 270-degree rotations.

TensorFlow:

TensorFlow give more control to the developer and data scientist compared to turi create. All the images are resized to 50 x 50. The data is prepared same like turi create. Original image and its roatation is converted into single image file. The input therefore will be 50 x 100 size image. Out of all data, 80 percent is training data and 20 percent is the test data.

All the data is converted into NumPy array to feed into neural networks. Data is been normalized by dividing 245 as it is the maximum possible pixel value. I tried to use keras but due to less promising result I have created own neural nets.

For Images convolution neural network gives very promising results. Input neural is 5000 as image is of 50\*100. First layer is the convolution layer with 32 filters and kernel size 5 followed by max pooling layer with extract the highest value from the window of given size. Second convolution also follows the same with 64 filters and relu as activation. Lastly SoftMax layer that forma the fully connected layer with the 4 classes. For calculating loss, sparse SoftMax cross entropy with logits is used and to minimize loss Adam optimizer is used.

Model shows the accuracy of 0.954 on the training dataset.