Project Overview

“Bangla Currency Recognition” is a project dedicated towards the visually impaired people living in Bangladesh. Unlike many countries, Bangladeshi monetary notes don’t have special features such as raised print (which is easy to feel), tactile marks etc. to assist the visually impaired people. Hence, we have tried to come up with a solution that is realistic in nature and easy to use in everyday life.

The currency recognition problem has been previously tackled using techniques such as optical character recognition (OCR), image processing, etc. Although these techniques are theoretically promising, they are not feasible in real life scenarios as they require images with very high resolution and proper lighting conditions.

Our focus is to provide a solution that will work on images of notes taken by smart-phone devices in everyday conditions. So, we gathered realistic images that are neither properly lightened nor have very high resolution. We built a model using deep convolutional neural network which was trained on such images.Finally, we included our model in an android application. The application will recognize thenotes using smart-phone camera and will play a sound mentioning the value of the note.

Problem Statement

In this project, we are working with 8 classes of monetary notes. The target is to predict the likelihood that a monetary note is from a certain class from the given set of classes. So, it is a multi-class classification problem.

Performance Metric

As the dataset of monetary notes is balanced, we are using “accuracy” as our performance metric.

Data Description

Our dataset consists of images of monetary notes of 8 classes (Tk 2, 5, 10, 20, 50, 100, 500, and 1000). For each class, we have collected 1000 images which have been taken using arbitrary lighting and resolution, which would be the case in real-life scenarios. We have also performed necessary augmentations on these images, such as rotation, flip, scale, translation and shear. These augmentations have been performed using random parameters.

Data Preprocessing

We have collected 8000 images of Bangla taka where we have 1000 images of each note. Out of 8000 images, 4800 (60%) images are in the training set, 1600 (20%) images are in the validation set and the rest 1600 images (20%) are in the test set.

As per using MobileNet like architecture, images are preprocessed as performed in the original MobileNet paper. In the dataset, input images come in different sizes and resolutions so they were resized to 224 x 224 x 3 to reduce size. To produce more images during training and to overcome overfitting, we have used data augmentation. Data Augmentation alters our training batches by applying random rotations, shifting, shearing and zooming. We have also normalized the train images to ensure faster convergence and avoid overfitting.

Model Description

We have used MobileNet which is an efficient convolutional neural network for mobile and embedded vision applications. MobileNet is a light deep CNN.

The MobileNet model is based on depth wise separable convolutions which is a form of factorized convolutions which factorize a standard convolution into a depth wise convolution and a 1×1 convolution called a point wise convolution. For MobileNets the depth wise convolution applies a single filter to each input channel. The point wise convolution then applies a 1×1 convolution to combine the outputs the depth wise convolution. A standard convolution both filters and combines inputs into a new set of outputs in one step. The depth wise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size.

The MobileNet has two part. One is top and another is bottom. The top part is a collection of dense layers. We have discarded the top part of MobileNet because that is used for 1000 categories of ImageNet dataset. The MobileNet has a hyper-parameter named ‘alpha’ which controls the number of layers in the bottom part. The weights of MobileNet used in our model are pre-trained.

Architecture Description

As mentioned earlier, we have discarded the top part of the MobileNet because that is used for 1000 categories of ImageNet dataset and we have 8 categories. We have used our own top part instead of the default top part of the MobileNet.

After the bottom part of the MobileNet, we have used five dense layers. The first dense layer has 1024 nodes, second dense layer has 512 nodes, third dense layer has 512 nodes, fourth dense layer has 256 nodes and the last one has 8 nodes. The last dense layer works as the output layer.

Algorithms and Techniques

Transfer Learning

Transfer learning refers to the process of using the weights of a pretrained network trained on a large dataset applied to a different dataset (either as a feature extractor or by fine-tuning the network). Fine-tuning refers to the process of training the last few or more layers of the pretrained network on the new dataset to adjust the weight. Here weights from a convolutional neural network pretrained on ImageNet dataset is fine-tuned to classify Bangla currencies.

Activation functions

We have used ReLu activation for the first 4 dense layers and Softmax activation for the last dense layer.

Optimizer:

We have used **Adam** as Optimizer.

Loss Function:

We have used **'categorical_crossentropy'** loss function.

Metric:

We have used **‘accuracy’ as** output score.

Hyper Parameter Tuning

We have selected two hyper parameters namely Width multiplier and Resolution multiplier. These two are model shrinking parameters. The base MobileNet architecture is already small and low latency. But many times a specific use case or application may require the model to be smaller and faster. In order to construct these smaller and less computationally expensive models we introduce a very simple parameter α called width multiplier. The role of the width multiplier α is to thin a network uniformly at each layer. This variable ranges from 0 to 1. We have applied 4 values 0.25, 0.5, 0.75 and 1.0.

The second hyper-parameter to reduce the computational cost of a neural network is a resolution multiplier ρ. This reduces the input image resolution when required and makes the computation faster. We used 4 square resolutions 224x224, 192x192, 160x160 and 128x128. Thus combining these two hyper parameters we chose the best model among 4x4=16 models.

Result Description

We ran our architecture for 16 combinations of two hyper parameters. The detailed results are reported in results.docx file. Then we choose the best hyperparameter combination from tuning results providing best validation accuracy. Finally we train our best model using this hyperparameter combination using a larger dataset.

Our best model so far is for Width multiplier: 1.0 and Resolution multiplier: 224. The loss and accuracy were respectively. We ran our model for 10 epochs and would get better results if we could run some more iterations as the model seemed to converging more.

Conclusion

We proposed and developed a mini architecture to detect bangle currency using MobileNet based on tensorflow backend. We trained our model and progressed our work further to build an android app. Our accuracy is satisfactory and further we would extend our procedure to detect other currencies.

Width multiplier	Resolution multiplier	Train score (10 epoch)	Validation score (10 epoch)
1.0	224	0.722049689441	0.6425
1.0	192	0.734472049689	0.57125
1.0	160	0.723602484472	0.6025
1.0	128	0.711956521739	0.62125
0.75	224	0.687888198758	0.58
0.75	192	0.652950310559	0.5575
0.75	160	0.65993788819	0.59125
0.75	128	0.667701863354	0.5525
0.50	224	0.593944099379	0.5275
0.50	192	0.633540372671	0.535
0.50	160	0.621118012422	0.53
0.50	128	0.576863354037	0.48
0.25	224	0.517080745342	.405
0.25	192	0.444875776398	0.30625
0.25	160	0.469720496894	0.32625
0.25	128	0.487577639752	0.3625

Width multiplier	Resolution multiplier	Train score (50 epoch)	Test score
1.0	224	0.9888	0.985