A faster and efficient classification model for edge-devices

Problem Statement

Deep learning models have achieved remarkable accuracies in various computer vision applications. Besides their popularity, one of the key challenges that restrict their uses on devices with low-power, low-computational capability and memory is their high resource requirements during inference. This makes it difficult to deploy trained deep learning models on mobile and other edge-devices. The current capabilities of the edge-devices are beyond their capacity to execute these large models for real-time classification. In the proposed work, the popular pre-trained models will be deployed on mobile devices. Before deploying on mobile devices, the model will go under pruning phase in which the unimportant parameters of the models will be removed to speed up the inference performance of the models. The unimportant parameters will be identified based on feature-maps.

Background

CNNs which are one of the popular DNNs are used extensively in various classification, object, and segmentation applications. CNNs has generally parts feature extraction and classification. In feature extraction various layers are used such as convolutional, batch normalization, pooling, activation etc. The output of feature extraction layers is passed to dense layers for classification. It has been found that convolutional layers and computationally-intensive. Network pruning is one the popular compression and acceleration techniques for pruning unimportant/redundant parameters from the network. Pruning filters from convolutional layers increase the inference performances of the model.

Methodology

The flowchart of the proposed approach is shown in Fig. 1.

Fig. 1 Flowchart of the proposed approach

Step 1: Training the base classifier

First, a CNN will be trained on the selected dataset to gain acceptable training and validation accuracies. For this purpose any deep learning framework can be used like Keras, TensorFlow, PyTorch etc., However, PyTorch is preferred compare to other frameworks due to its flexibility.

Step 2: Prune unimportant filters

Once the base model is trained, now the important part is pruning the filters. It is important to decide the importance of the filters to be pruned form the convolutional layer, There are many criteria based on which the importance of the filter can be decided such their absolute sum, impact on the error etc. After pruning the filters a new model need to be crated and copy the weights from original model to new model of the remaining filters.

Step 3 Fine-tune the pruned model

Pruning parameters from model degrades the performance of the model. In order to compensate the accuracy loss due to the removal of the filters, a pruned model needs to be fine-tuned for more epochs on the same dataset. The process of pruning and fine-tuning can be iterative.

Step 4: Deployment on mobile device

After fine-tuning, to compare the performance of the pruned and effectiveness of the proposed approach, the pruned and original model will be deployed on real mobile-device. TO deploy on the mobile TensorFlow lite or PyTorch can be used.

Experimental Design

Dataset and model

The initial experiments will be performed on standard CIFAR10 and CIFAR100 dataset. There are numbers of classification models, first set of experiments will be performed with VGG16 architecture. After getting satisfactory results on VGG16, the experiments will be extended to more complex architectures like ResNet and DenseNet. In addition to CIFAR dataset any custom dataset can be also be used with transfer learning.

Evaluation Measures

To compare the performance of the original and pruned models various evaluation measures will be used such as accuracy, number of parameters in the original and the pruned model, number of FLOPs, and time taken to process the single image.

Software and Hardware Requirements

To large experiments on large models and datasets, GPU are required. In the absence of physical GPUs, there are many online platforms that could be utilized to conduct the experiments. It should also be noted that initial experiments can be performed on system with low computing power.

Input: Original trained model

Expected Output: A DNN model with improved performance

Mentor Name: Tejalal Choudhary