An efficient cloud-edge framework for computer vision applications

Ehsan Mohyedin Kermani
ehsan.m.kermani@huawei.com

Amin Banitalebi-Dehkordi
amin.banitalebi@huawei.com

Yong Zhang
yong.zhang3@huawei.com

Yuri Mosieyenko
yuri.moiseyenko@huawei.com

Lanjun Wang
lanjun.wang@huawei.com

Jiebo luo
jiebo.luo@huawei.com

Huawei Technologies Canada
Vancouver BC Canada

Extended Abstract

Computer vision has been revolutionized since the emergence of deep learning practices. Deep learning models tend to go deeper and heavier in size to obtain better accuracy, thus become very challenging for edge computing applications. Recently, designing lighter models have started to gain much attention. The status quo approach for deploying deep learning computer vision models does not utilize the computing resources of edge devices such as mobile GPU for inference and treats edge devices as merely sensors that send serialized captured images to cloud. After all the computation is done in cloud, the result is serialized again and sent back to the user, which has the inevitable serialization and deserialization overhead causing major *latency* and can compromise user *data privacy*.

Our work has extended *BranchyNet* with early exist through creating a converter module based on convolutional layers with a tunable number of channels C. For image classification, our framework uses the similarity of the first layer of *ResNet* family and *MobileNet* to make a bridge between heavy models that reside in cloud and lighter models that can reside in edge devices. For training (Fig1], we have optimized the joint loss of our models. For inference (Fig2], the captured image goes through the model in the edge device and outputs a confidence score based on the normalized cross-entropy. If it passes a threshold, which was obtained through hyperparameters search, the result is shown to the user. If the model is not confident, the result of the intermediate computation, which can be an order of magnitude smaller than the original input image, is sent to the cloud and the final result is sent back from cloud to the user.

For our experiments, we have used Huawei's Mate10-pro smartphone with custom mobile GPU with access to Huawei Cloud. To efficiently use its computational resources, we have compiled our customized lighter models via Tensor Compiler Stack (TVM). TVM borrows ideas from low-level compiler techniques through unified Intermediate Representation (IR) of the components and code-generation in order to create optimized kernel codes for the underlying supported hardware.

As a result, up to 95% of our test images were correctly classified at edge and the most challenging 5% were sent to cloud for classification. Moreover, we can save up to 40x bandwidths which lead to lower latency, saving energy consumptions and higher total accuracy. Furthermore, our framework provides much greater data privacy as it does not expose any original captured user data.

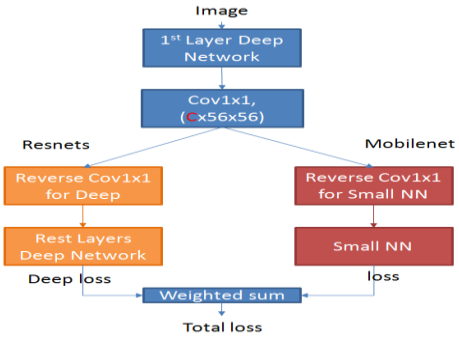

Fig1: Training architecture

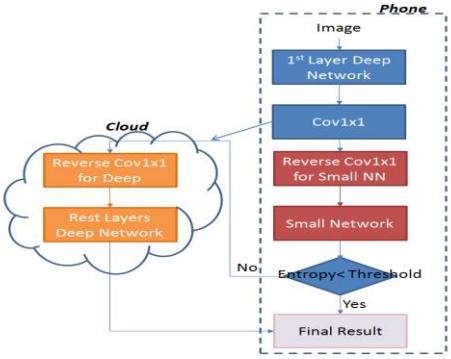

Fig2: Cloud-edge inference