April 2020

Shared On-device and Server-based Machine Learning Evaluation

Jorim Jaggi

Yannick Stucki

Adrian Roos

Sandro Feuz

Shared On-device and Server-based Machine Learning Evaluation

ABSTRACT

On-device machine learning models can provide inferences quickly; however, such models often have a large size and need to be downloaded to the device. In many domains, such as optical character recognition or translation, the nature of the problem necessitates that several on-device models be made available.

This disclosure describes techniques to split inference computation between the device and a remote server by leveraging the observation that many ML models have domain independent layers that are shared between multiple models and domain specific layers that are specific to individual models. Per techniques described herein, an available smallest intermediate representation from the shared layers is transmitted to the server inference. Alternatively, a bottleneck layer is inserted between the domain-independent and domain-specific layers of the model. The shared layers of the model are on-device while the domain-specific layers are on the server. The use of the smallest intermediate representation eliminates the need to store large models locally, reduces the cost and delays of network communication by minimizing the data transmitted during inference, and allows leveraging the power and flexibility of server-based machine-learning models.

KEYWORDS

- On-device machine learning
- Cloud-based machine learning
- Split model
- Model compression
- Compressed representation

BACKGROUND

Machine learning models are getting larger and more complex, with some models being multiple hundreds of megabytes. While it is possible to deploy large models on a server and utilize server-based inference, this necessitates that the input, e.g., images, speech, etc. be transmitted to the server. With large size inputs, this can cause substantial delay and further requires sensor data to leave the device, which may not be feasible in certain situations, e.g., when the user does not provide permission for the data to leave the device. On-device machine learning models can provide inferences quickly; however, this requires that the large size models need to be downloaded to the device. Further, inference computation using such large size models can be computationally intensive.

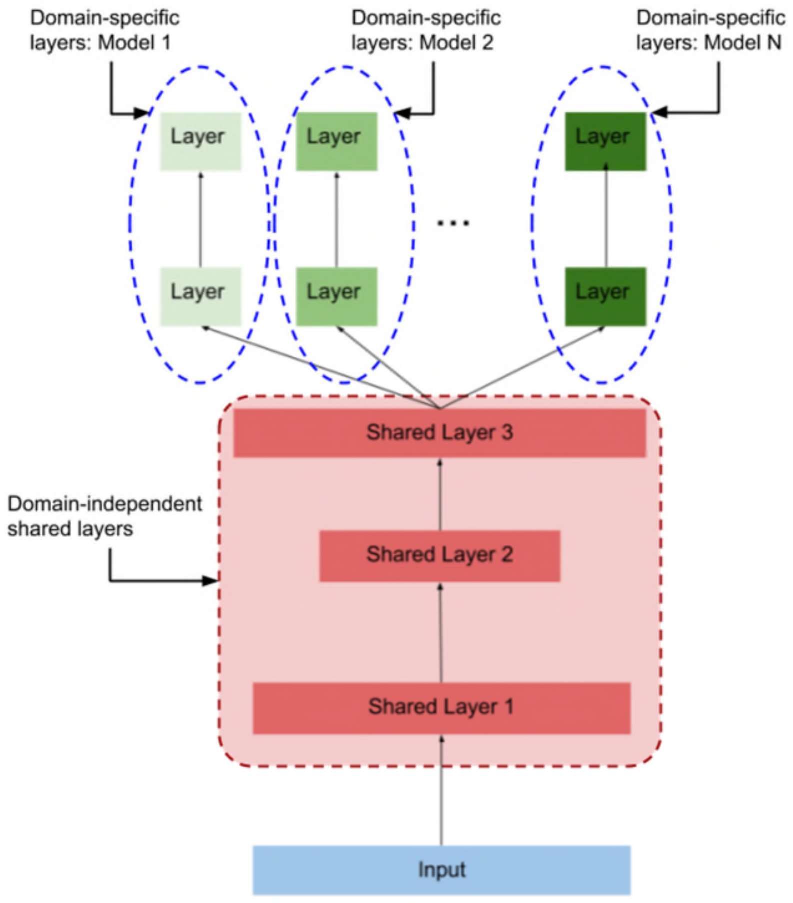
Domain-specific layers: Model 1
Domain-specific layers: Model 2
Domain-specific layers: Model N
Layer
Layer
Layer
Layer
...
Layer
Layer
Shared Layer 3
Domain-independent shared layers
Shared Layer 2
Shared Layer 1
Input

Fig. 1: Example ML models with shared and domain-specific layers

In many domains, such as optical character recognition or translation, the nature of the problem necessitates that several different models be made available on device in order to solve different related subtasks. In many such tasks, machine-learning models are often customized to different inputs. For example, in the domain of optical character recognition (OCR), machine learning models can have script and language dependent components. Similarly, translation or speech recognition models are typically language-dependent, e.g., have a common model architecture and common layers closer to the input, but different, language-dependent model weights for later layers.

An example of machine-learning models with domain-independent shared layers and domain-specific layers is illustrated in Fig. 1. In this example, *N* models are shown. The models have shared, domain-independent layers 1-3 (shown in red). Each model also includes specialized, domain-specific layers (shown in different shades of green). In the model structure, the common layers are typically found close to the input (shown in blue).

DESCRIPTION

This disclosure describes techniques to split inference computation between the device and a remote server by leveraging the model structure in which ML models have domain independent layers that are shared between multiple models and domain specific layers that are specific to individual models, as shown in Fig. 1 above.

Per techniques described herein, an available smallest intermediate representation from the shared layers is transmitted to the server inference. Alternatively, a bottleneck layer is inserted between the domain-independent and domain-specific layers of the model in order to force a small intermediate representation. The shared layers of the model are on-device while the domain-specific layers are on the server. The use of the smallest intermediate representation eliminates the need to store large models locally, reduces the cost and delays of network communication, keeps the raw input data safely on-device, and allows leveraging the power and flexibility of server-based machine-learning models.

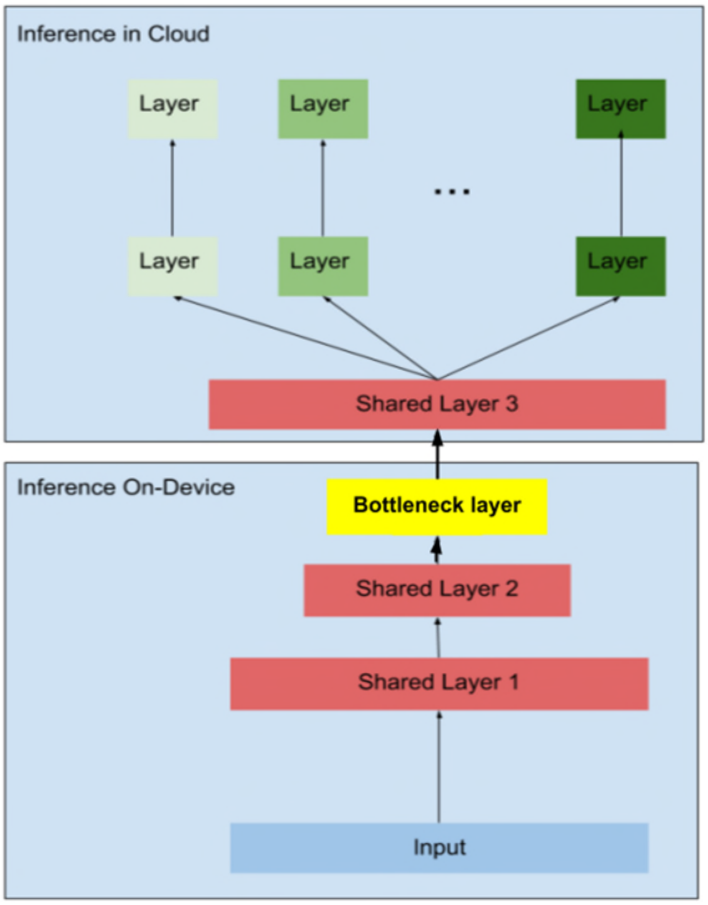

Fig. 2: Splitting machine learning models into on-device and server-based components

Fig. 2 illustrates an example of splitting inference computation between a device and a server. The model is analyzed to find the smallest intermediate representation amongst the shared layers. In the example illustrated in Fig. 2, shared layer 2 is the smallest intermediate layer. If suitable, output of the layer can be sent directly to the server. Alternatively, as illustrated in Fig. 2, the model can be rewritten to insert an even smaller bottleneck layer (shown in yellow). The bottleneck layer (and the layers above) can be trained from the finished model by freezing all remaining weights. Alternatively, the whole model can be retrained either from scratch or using teacher-forcing on a prior version of the model.

In this manner, each ML model is split into two sub-models. A first sub-model (shared between all models) includes the section from the input up to and including the bottleneck layer and is deployed on-device. A second sub-model includes a model-specific section above the bottleneck layer and is deployed on a server. The output of the bottleneck layer is a compressed representation of the input which can be sent over the network to the server. Such transmission is fast and relatively inexpensive.

Splitting the model in this manner avoids the need for transmittal of the entire input, e.g., a large image, which may be slow and incur a large cost, while leveraging the availability of multiple, powerful, domain-specific models on the server. Further, the burden of hosting multiple domain-specific models is shifted from the device to the server where more resources are available.

The split models are used with specific user permission. If the user provides permission for server-based inference, the intermediate representation of the input is sent to the server, which then provides an inference to the user device. If the user denies permission, or requests that all computations be performed locally, split models are not used, and instead, the complete model is provided on device.

CONCLUSION

This disclosure describes techniques to split inference computation of deep machine learning models between the device and a remote server by leveraging the observation that many ML models have domain independent layers that are shared between multiple models and domain specific layers that are specific to individual models. Per techniques described herein, an available smallest intermediate representation from the shared layers is transmitted to the server inference. Alternatively, a bottleneck layer is inserted between the domain-independent and domain-specific layers of the model. The shared layers of the model are on-device while the domain-specific layers are on the server. The use of the smallest intermediate representation eliminates the need to store large models locally, reduces the cost and delays of network communication, and allows leveraging the power and flexibility of server-based machine-learning models.

REFERENCES

[1] Matsubara, Yoshitomo, Sabur Baidya, Davide Callegaro, Marco Levorato, and Sameer Singh. "Distilled Split Deep Neural Networks for Edge-Assisted Real-Time Systems." In *Proceedings of the 2019 Workshop on Hot Topics in Video Analytics and Intelligent Edges*, pp. 21-26. 2019.

[2] Amir Erfan Eshratifar, Amirhossein Esmaili, and Massoud Pedram, “BottleNet: A Deep Learning Architecture for Intelligent Mobile Cloud Computing Services,” *arXiv preprint arXiv:1902.01000 (2019)*.