Technical Disclosure Commons

Defensive Publications Series

May 31, 2019

ADAPTIVE DEPLOYMENT OF MACHINE LEARNING MODELS IN MOBILE DEVICES

HP INC

Follow this and additional works at: https://www.tdcommons.org/dpubs_series

Recommended Citation
INC, HP, "ADAPTIVE DEPLOYMENT OF MACHINE LEARNING MODELS IN MOBILE DEVICES", Technical Disclosure Commons, (May 31, 2019)
https://www.tdcommons.org/dpubs_series/2239

This work is licensed under a Creative Commons Attribution 4.0 License.
This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.

Adaptive deployment of Machine Learning models in mobile devices

Abstract

This invention aims to allow mobile applications based on Machine Learning to run in different devices with different configurations, by optimizing the tradeoff between accuracy, response time and storage use specifically to each device. It consists in allowing the developer to train multiple models to solve the same task. A subset of trained models, with acceptable accuracy, will then be made available for the application. Whenever the application is installed on the device, it will pre-load each of the available models, initialize it with random parameters, and run some inference steps on the device, in order to measure the computation time. With this information, together with the full size and accuracy of the model , the application can select the most accurate model with acceptable response time (according to the application), or even allow the user to select a model.

Description

Mobile applications are currently evolving, which need to perform intelligent computation of images in real time, to enable a fluid experience for the users to reinvent and improve printing experiences and leverage new lines of products. However, state-of-art research Machine Learning that offer high accuracy solution for such problems have a tradeoff in terms of computational costs and storage use. As mobile applications need to consider the deployment in a broad range of devices, optimizing the tradeoff between accuracy, computing time and model size to each device is crucial to deploy a solution.

To exemplify the process, we show in figure 1 an application should draw the outer rectangle of a scanned page, as to give a proper feedback to the user. Notice that, the response time of this feedback is crucial. To get from the input image (on the left) to the resulting points, different ML models can be trained, often with different setup for the input and output format to these models.

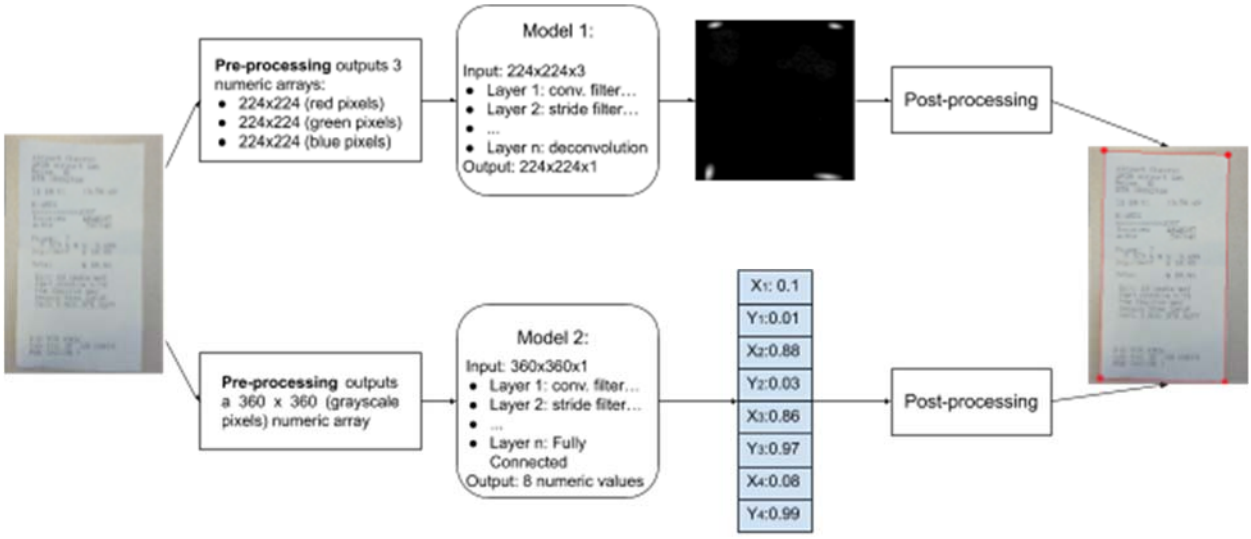

Figure 1: Example of an ML model for solving a corner detection task

One of the challenges is the tradeoff between accuracy and performance - sometimes, a small loss of accuracy compensates the loss of performance. A model that position the corners exactly in the same pixel as the border of the document is desirable, but a couple pixels of distance are acceptable if the response is given in few milliseconds instead of half a second. While measuring accuracy is a matter of a proper engineering process, performance depends a lot of the conditions of the device to be deployed. For this reason, it is desirable to have multiple models fit the needs of a multiple range of users, and the proposed method will solve the problem of selecting the more adequate. In Figure 2, we depict the whole process.

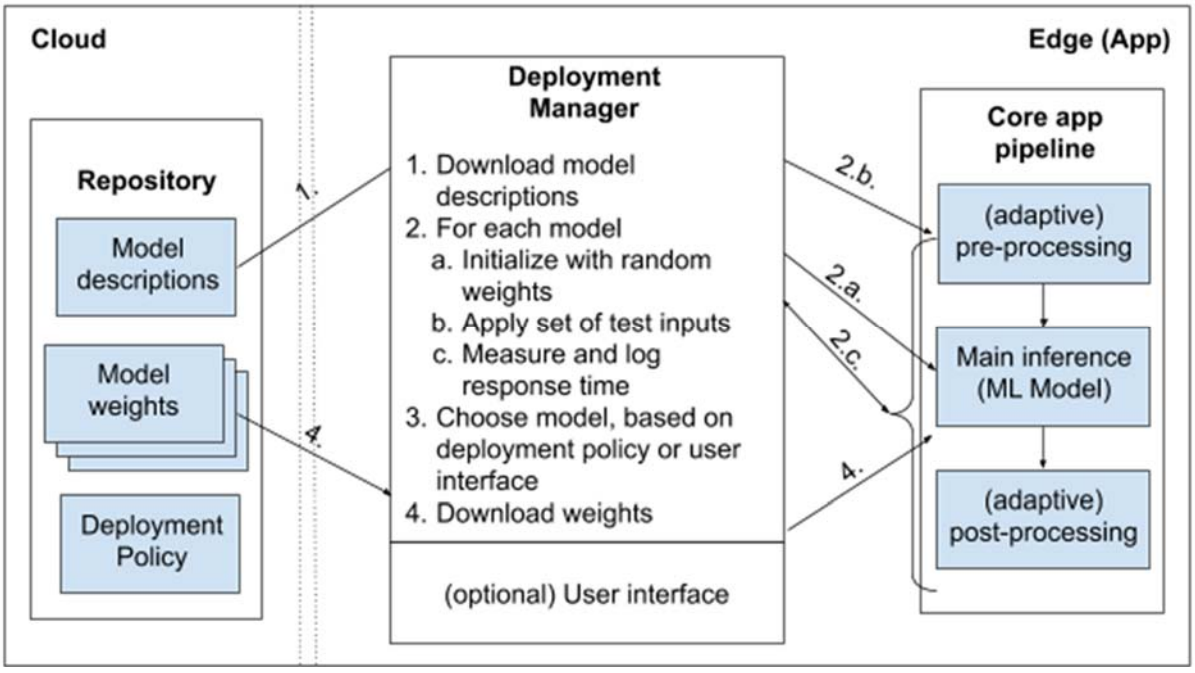

Figure 2: Process of adaptive model deployment

The first part of the process is, make different models available in a model repository to solve the same task. This repository keeps the description of the model separated of the numeric parameters adjusted during training. In general, the representation size of the weights of a deep network model is some orders of magnitude larger than the description of the model. Also, the repository describes a deployment policy, which keeps general information about the models and the acceptable conditions for each task.

Inside the app, the deployment manager is the module responsible for deciding which model (or models) are adequate to the device. It starts by downloading the description of all the models in the repository, selecting those that attend the size restriction established in the policy (according with the total storage of the device and the free space). For each description, the manager creates a new model, following the architectural details but initializing the weights with random, non-zero, values. This will allow testing the computational complexity of the model without having to download the whole set of weights, reduces both the traffic data and the storage necessary to the process.

The next step of the manager consists in applying a sequence of inputs through the main pipeline (pre-processing, core model, post processing), recording the time necessary to compute each of the steps. If any of the steps exceeds the time limit, the computation should be stopped - and retried as many times as expressed in the descriptor - if all attempts exceed the limit, that model should be disconsidered. At the end of the process, the manager should have a list of the remaining models, ordered by processing time. Again, based on the deployment policy, one of the models can be selected (faster model, more accurate with time below threshold), or a user interface can be called so the user can choose the preferred experience.

After the choice is made, only the weights of the actual chosen model are downloaded from the repository, and then loaded into the application. The deployment manager keeps also regularly monitoring changes in the repository and measuring the response time of the other modules. The whole process of time measurement/model choice may be performed again, every time the repository is updated, or whenever the application starts to show signals of slower response time.

The main advantage of this process is the personalization. Users with a high-end device can enjoy an experience with the best accuracy, while users in the mid-range can still have an application that attends the purposes it was developed for. Moreover, it accelerates the development of Machine Learning-based applications, by automating the test of different models in different devices, by isolating the development and deployment stages, and allowing also the over-the-air update of models without upgrading the application.

Disclosed by Rafael Borges, João Melo, Ricardo Piccoli, Vinicius Lafourcade and Ricardo Ribani, HP Inc.