Seamless Model Flighting For On-device Machine Learning Models

ABSTRACT

On-device machine learning models play a key role in applications such as virtual assistants, document scanners, photo editors, facial recognition applications etc. on modern smartphones and other devices by eliminating the dependency on network connectivity and delivering a consistent and satisfactory user experience. However, the embedded model architecture makes the task of flighting a new machine learning model challenging. In model flighting, a small portion of the user base is exposed to the new version of the model, while the rest of the user base continues to use the old version of the model that is already part of the application. The performance of both models is compared and the model that has better performance is chosen for deployment in the main application. This disclosure describes an approach to seamlessly conduct flighting for machine learning models embedded in smart devices on hundreds of thousands smart devices instead of a much smaller sample of in-house devices in a lab environment. Using the architecture described in this disclosure, model flighting can be achieved without a user downloading and updating to a new version of the application.

KEYWORDS

- Model flighting
- Machine learning
- Model testing
- Model update
- On-device model
- A/B Testing
- Smart Device
BACKGROUND

Machine learning plays a key role in applications such as virtual assistants, document scanners, photo editors, facial recognition applications etc. on modern-day smartphones and other smart devices. In such applications, low response time is one of the key drivers of user satisfaction. To achieve a low response time, machine learning models are embedded in many applications that execute locally on a smart device. Having the machine learning models embedded in the smart devices eliminates the dependency on connectivity and helps deliver a consistent and satisfactory user experience. However, the embedded model architecture makes the task of flying a new machine learning model quite challenging.

Model flying is a key activity that is performed before a model is deployed in the main application. In model flying, a small portion of the user base is exposed to the new (test) version of the model, while the rest of the user base continues to use the old version of the model that is already a part of the application. The performance of both models is compared and the model that has better performance is chosen for deployment in the main application. In many applications, machine learning models are updated periodically to introduce new functionality or to account for advances in the underlying algorithms. Hence, ensuring seamless model flying is a priority for data scientists and developers that build the applications.

Model flying is challenging in applications that have machine learning models embedded and available for local execution on smart devices. When models are executed on a server, e.g., through a web service, flying can be conducted seamlessly since both the production model as well as the new candidate (test) model can be made available on the server. However, when the model is an on-device model, a new version of the model can only be shipped with a new version of the app.
Currently, model flighting is conducted by testing a new version of the app (with the new model) on in-house devices that are available with the testing team. However, the test sample is typically quite small and may not sufficiently simulate some operations or scenarios experienced by the end users.

DESCRIPTION

This disclosure describes an approach for flighting a machine learning model seamlessly, before rolling it out to the main application on the distribution platform. The approach described in this disclosure proposes a change in the application architecture. The proposed application architecture, illustrated in Figure 1, takes a machine learning model driven approach.

![Application architecture for model flighting](image)
Per the new architecture, a server is provided that includes a model repository and a flighting service. The model repository is used for managing various versions of a machine learning model, e.g., a production version, and one or more test/candidate versions, on the server. The flighting service selects the appropriate model for flighting and communicates with a model manager on the smart device.

The application architecture on the smart device includes a model manager component, model runtime engine component, and an action controller component. The model manager is used to manage the embedded local models (stored in the local model store) and to perform model flighting. The model runtime engine is used for running the machine learning model and transforming the output into predefined action signals.

For example, if the machine learning model is a PyTorch model, then the model runtime engine invokes the PyTorch engine and runs the model. The action controller is used for management and conversion of action signals into the related actions and task completion. Additionally, the action controller is also used to manage the basic routine task workflow.

In this architecture, when a new model is to be flighted, the model manager detects it by communicating with the flighting service on the server and downloads the new model from the model repository on the server, without updating other portions of the app. The flighting service on the server side can control which users or devices are eligible for flighting, e.g., based on permitted client data, to adequately cover the different scenarios to be tested.

On devices that are eligible for flighting, the model manager downloads the new candidate model and switches to it for flighting the new functionality. The results from model flighting are communicated by the model manager to the flighting service on the server. The results are collated and the performance of the new model is compared, e.g., against the
performance of the existing model in production (or other candidate models). For example, a metric such as user engagement time on the application can be used to compare the performance of the existing model and the new model.

A key benefit of this architecture is that it gives the developers substantial flexibility to do model flighting on a very large number (e.g., hundreds of thousands) smart devices that have the app installed, rather than a much smaller (e.g., a few hundred) in-house or test devices in the lab. This can be achieved without the user downloading and updating to a new version of the application. Furthermore, this approach also provides better A/B testing data, since the operations are on the same device and come from the same user.

CONCLUSION

This disclosure describes an approach to seamlessly conduct flighting for machine learning models embedded in smart devices on hundreds of thousands smart devices instead of a much smaller sample of in-house devices in a lab environment. Using the architecture described in this disclosure, model flighting can be achieved without a user downloading and updating to a new version of the application.