Rohan Ahluwalia, Summer Intern 2022

Deploying PyTorch Torchscript Models in Micropython

Project Description: Design and implement custom importers that are able to read .pt files and deploy that PyTorch model. From the .pt file, it needs to be able to grab the corresponding python source code and import the relevant packages into the micropython environment.

To achieve the goals set out for the project, there are three sections:

1. Create a micropython interpreter which can deploy micropython scripts in C++
2. Implement a custom loader that can read relevant files from .pt file and deploy them
3. Implement IValue -> mp_obj_t conversion for relevant types (integer, float, tensor ...etc)

While we are continuing to work on this project's sections, the proof of concept for the three sections have been completed.

Section 1: Creating a Micropython Interpreter

In this section, a micropython interpreter was developed that had all of micropython and torch functionality.

Progress:

- Completed implementing micropython functionality in open source code (upytorch directory). https://github.com/nikithamalgifb/upytorch/pull/10
    - In the open source code, we were not able to figure out how to implement the torch functionality, even though we had spent ~4 weeks working on this section. *More notes in the following part.*
- Completed implementing micropython + torch functionality in fb source code
    - The landed diff had all of the required functionality to deploy simple PyTorch models in Micropython.

Notes about open source:

- Open source proved to be difficult to link torch and micropython functionality

- Potential Solution: Need to build PyTorch entirely within and link the completed build to the micropython. Similar to the solution that was in the fbcode, which has a micropython wrapper which processes the torch information. Current strategy of manually inputting the relevant files is not effective.
- Work can be done on ensuring that open source can be effective, however, majority of requirements are completed in internal codebase

Work to be completed:

- Open Source: Can have a working interpreter in open source that has torch functionality

Section 2: Custom Loader

In this section, we worked on deploying a custom loader that works to read the relevant files and import them into the micropython environment.

Progress:

- Completed deploying a simple model into micropython with full functionality: https://github.com/nikithamalgifb/upytorch/pull/11

Work to be completed:

- Continue to work on the .zip functionality to make sure that we have all of the required files in order to run more complex models

Section 3: IValue → mp_obj_t conversion

This aspect converts the IValue (torchscript) to mp_obj_t (micropython). This allows for all of the C++ variables to be used in the micropython environment, such as tensors, floats, ints, ..etc.

Progress:

- Completed Ivalue stubs to allow for testing environment support
- Completed Ivalue support for integers, floats, and lists.

Work to be completed:

- Finish implementation for nested lists, tensors, and other nested functions.

Overall Milestones

- We were able to deploy a simple torch model in micropython on the internal fb code
- Had support for basic IValues (floats, ints, lists)
- Create interpreter class which had required torch and micropython functionality