Torch Deploy

This is an experimental feature to embed multiple python interpreters inside the torch library, providing a solution to the 'GIL problem' for multithreading with the convenience of python and eager or torchscripted pytorch programs.

libinterpreter

This is an internal library used behind the scenes to enable multiple python interpreters in a single deploy runtime. libinterpreter.so is DLOPENed multiple times by the deploy library. Each copy of libinterpreter exposes a simple interpreter interface but hides its python and other internal symbols, preventing the different python instances from seeing each other.

CPython build

Torch Deploy builds CPython from source as part of the embedded python interpreter. CPython has a flexible build system that builds successfully with or without a variety of dependencies installed - if missing, the resulting CPython build simply omits optional functionality, meaning some stdlib modules/libs are not present.

Currently, the torch deploy build setup assumes the full CPython build is present. This matters because there is a hardcoded list of python stdlib modules that are explicitly loaded from the embedded binary at runtime.

rebuilding CPython after installing missing dependencies

Because CPython builds successfully when optional dependencies are missing, the cmake wrapper currently doesn't know if you need to rebuild CPython after adding missing dependencies (or whether dependencies were missing in the first place).

To be safe, install the complete list of dependencies for CPython for your platform, before trying to build torch with USE_DEPLOY=1.

If you already built CPython without all the dependencies and want to fix it, just blow away the CPython folder under torch/csrc/deploy/third_party, install the missing system dependencies, and re-attempt the pytorch build command.