Nicolas Giot

**Lab 2 Mnist in Tensorflow Lite**

Working on this MNIST project using TensorFlow and TensorFlow Lite had a few bumps, but I learned a lot from it. One of the first things that gave me trouble was setting up the Python virtual environment (the .venv). I’ve used environments before, but getting it to work inside VS Code with Jupyter notebooks wasn’t as smooth as I hoped. I wasn’t sure at first if the notebook was actually using the environment I set up, and some packages wouldn’t install correctly or weren’t being found. I had to run a bunch of test commands to figure out if everything was connected right.

Formatting the code also caused some issues. Some of the code I started with had errors, like missing line breaks or things written all on one line, which made it crash. I had to go through and clean up the code just to get it running. It reminded me how important it is to double-check the small stuff like indentation and line spacing, especially when working in notebooks where each cell runs on its own.

Even though it was frustrating at first, it got more interesting once I started building the model and training it. I’ve used PyTorch in the past, so I kind of understood the general process—load data, build a model, train, evaluate—but I hadn’t used TensorFlow before. This project helped me see how TensorFlow handles those steps. It feels more structured than PyTorch, kind of like it expects you to follow a certain flow. Once I got used to it, it made sense and felt solid for training and saving models.

The most new and surprising part was TensorFlow Lite. I hadn’t heard of it before this. I didn’t know there was a smaller, lighter version of TensorFlow for running models on low-power devices. I liked that a lot. After converting the trained model to .tflite, it was actually pretty easy to load it and run predictions using the interpreter. It gave the same kind of results as the regular model, just in a smaller, faster format.

I can see how TensorFlow Lite could be really useful in real-world projects—like if you’re trying to run a model on a phone, a Raspberry Pi, or even a tiny embedded device. It makes the idea of AI on the edge more realistic. You don’t always need a full-size model running in the cloud, especially for simple tasks like digit recognition or other basic vision problems.

Overall, even though I ran into a few technical issues at the start, the project helped me get more comfortable with TensorFlow and showed me something new with Lite. It gave me a better sense of how models can go from training to actually being used in lightweight, real applications. I still like using PyTorch for learning and experimenting, but now I see the value in TensorFlow, especially when it comes to deploying models outside of just a laptop.