Ancud - MarioKart64 presentation @ PyData

What is TensorKart

Tensorkart is a project by kevinhughes27, which uses a Convolutional Neural Network in Tensorflow, which is trying to learn to play MarioKart64. (originally for the Nintendo64)

How does that work

1. **Install TensorKart** and all dependencies. Turned out to be pretty difficult
2. **Record gameplay** - Use `record.py` to record data. Data in this case is a screenshot of the game, and the state of the gamepad in this moment. Screenshots are saved in a folder and there's also a .csv file, which contains the correspoding controller input to each screenshot.
3. **Prepare the data** - This creates two `.npy` (python-numpy) files. `X.npy` is a 3d numpy array, containing pixel data for each pixel for each screenshot. `Y.npy` contains the correspoding inputs.
4. **Train** - feed the `X.npy` data to the neural network and try to predict the values from `Y.npy`. Result is a 5-tuple, which is interpreted as controller input.
5. **Play** - Uses a custom input plugin for the emulator, so that you can get controller input both from the neural network and from the player controller.

How do I use this?

Project directory

Most of the files you'll need are in

`~/Developement/multitensorkart/ternsorkart`

Once there, activate the virtual environment

`source ../.vTensorKart/bin/activate`

2 people against eachother for more gameplay data

- Experiment a bit to find which is Gamepad 1 and Gamepad 2. You will need this information in order to launch a 2 people recording.
- Use the bash script `./recordTwoPlayers` to start the recording and follow the instructions.
- **Game window has to be on main monitor** (If using multiple displays)
- **Game window has to be in TOP-LEFT corner** (ALWAYS)
- During the recording the UI updating on the Recorders will stop in order to save performance.
- Change the name of the recordings so that not to overwrite data

Playing against the AI

- Start `play.py`
- **Game window has to be on main monitor** (If using multiple displays)
- **Game window has to be in TOP-LEFT corner** (ALWAYS)
- Environment will navigate itself and select 1v1 Versus race
- Then when the neural network starts, the player can choose a character and a track
- Neural Network is *relatively* good at:
  - Luigi's Raceway
  - Kalimari Desert
  - Royal Raceway

Technical Details

What is Tensorflow

TensorFlow is an open-source software library for machine learning. It's designed as a library for numerical computation using dataflow graphs, which means that Tensorflow builds a graph of all the operations, and feeds inputs into the first node of the graph. Tensorflow is well suited for neural networks, but it can preform a number of machine learning tasks.

Neural Network Structure

- 5 convolutional layers, with increasing amount of filters
- Pooling layer between 2nd and 3rd convolutional layers
- Dropout layer after the 5th conv. layer
- 4 fully connected layer, with dropouts

Limitations

Data

Collecting data takes a lot of time. You have to play Mario Kart a lot to collect enough data, so that the neural network will be able to learn how to drive well. Data takes a lot of place. For each recording we're saving 400-800 screenshots and corresponding inputs. 1 recording is between 100-200 MB.

Preparing the data in numpy arrays for training takes some time, but the main issue here is HDD and RAM capacity. Cince there's no jpg compression, image data to numpy array takes more place, than the actual image, and it's a single file. So with 15-20 recordings we have a 7.8GB file.

Another problem with this file is RAM. The data file is being loaded once in memory. If the file was larger or we had less memory, there would be problems. It is important for large amounts of data, to have a proper infrastructure, so that you don't have to load all the data to the main memory.

Training

Training takes a long time. Tensorflow has a CUDA capability, so it can utilise the GPU to speed up the training process a lot. Unfortunately most old GPUs don't have the required CUDA capability, so old systems train only on CPU, which takes exponentially longer.

Luckily for us, we had a machine with a good GPU, where training with 15-20 samples took 30 minutes. In comparison, 30 Minutes with CPU on a Thinkpad W520 is 4 samples.

General Problems

One problem is that when turning players do sudden sharp turns, i.e. they hold down the left stick only for a short time. A potential problem is that the recorder misses some of those turns and so the neural network is trained in certain situations not to take a turn.

Another problem might be the structure of the network. We're not really sure if this is the proper model for the task. Problems which we have identified is that the model doesn't 'see walls'. If it misses a turn or doesn't see the road it bumps into a wall and can't recover. If there's a cliff next to the road instread of a wall, it seems to be better.

Not enough data? Deep learning is all about data. Data collection is not easy, and there are also certain infrastructure problems like loading all the data in memory at once, or taking too much HDD space and so forth.