Racing F-ZERO with Imitation Learning

Overview
▶ We are motivated by recent success of applying machine learning to play games at a level capable of surpassing human experts. We use the Retro Learning Environment for the SNES game *F-Zero* in order to reproduce works in the Imitation Learning space.
▶ We implement Dataset Aggregation (DAgger) to solve the dataset mismatch problem prevalent in sequential prediction tasks.

Data Acquisition
▶ Retro Learning Environment (RLE), a learning framework based on the Arcade Learning Environment can support SNES games.
http://github.com/nadavbh12/Retro-Learning-Environment
▶ Through 3 human playthroughs, acquired 29,375 RGBA images labeled with controller input

Automatic Player
▶ For the automatic playing of the game we collect aspects of the state from the emulator and base our decisions on a one-step greedy search. We pick among [RIGHT, NOOP, LEFT] and simulate the next 30 frames with that action choice, greedily choosing the action that maximizes our reward, as defined below:

\[
\text{reward} = \alpha(\text{isForward} \times \text{speed}) + \beta(\text{score}) + \gamma(\text{power})
\]  

Training Convolutional Neural Network
▶ Based on NVIDIA autopilot CNN for self-driving cars
▶ 39,366,570 total parameters
▶ Implemented in Keras with TensorFlow backend using 1 GPU
▶ Data split into 80% train and 20% test examples
▶ Images and labels are shuffled prior to training
▶ Raw pixel input of size 224 × 256 × 3
▶ Categorical cross entropy loss function:

\[
H(p, q) = -\sum_x p(x) \log(q(x))
\]

DAgger Algorithm

1. Train CNN on dataset \( D \) as initial policy \( \hat{\pi}_1 \)
2. for \( i = 1 \) to \( N \)
3. begin game
4. while game has not ended
5. run the CNN to obtain new trajectories
6. if pred probs < 0.5 or rand(0-1) < \( \varepsilon \)
7. get \( D_i = \{(s, \pi'(s))\} \) given by automatic player
8. aggregate datasets: \( D \leftarrow D \cup D_i \)
9. end while
10. Train CNN \( \hat{\pi}_{i+1} \) on \( D \)
11. end for

Results and Evaluation
▶ CNN accuracy is 95% on the test set
▶ DAgger run for 30 iterations able to complete full laps
▶ Saliency map visualizes attention over ‘NOOP’ class. Note that as expected the boundaries of the track are most salient. Surprisingly it also picks up the power bar and clock

![Final network architecture. This figure is generated by adapting the code from https://github.com/gwding/draw_convnet](https://github.com/gwding/draw_convnet)