DeepBaller: A NN-Controlled Real-Time Video Game AI

Wyatt Pontius, Kylan Sakata, Pablo Santos, Steven Chen (TA)

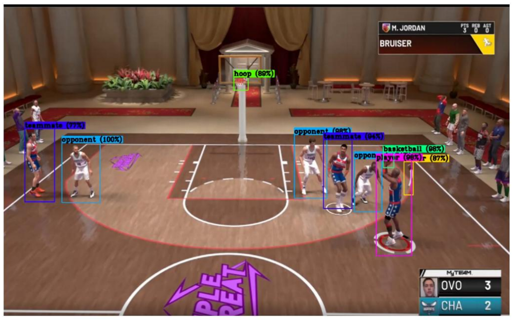
Problem

NBA 2K is a video game where the player takes control of a virtual basketball player on the court. We wanted to create a system - based on neural networks - that could allow a computer to play NBA 2K19 on Xbox without any source code or pre-coded understanding of the game.

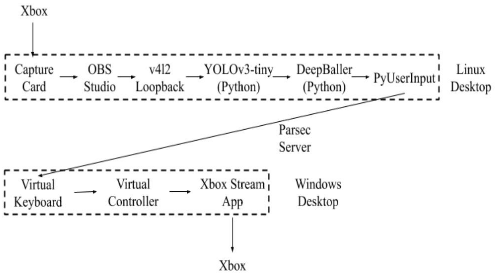
Object Recognition

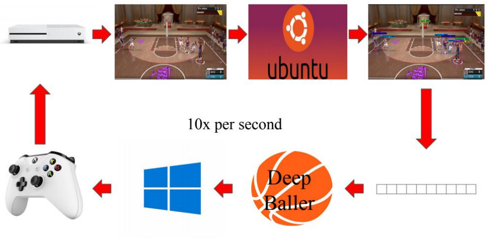
Using YOLOv3-tiny network we trained our own weights file to detect important game classes.

- 9 objects: basketball, player, teammate (2), opponent (3), basketball hoop, and shot meter.
- Hand labeled 2,500 frames of training data
- Bounding box coordinates were then extracted to train on.



System Design



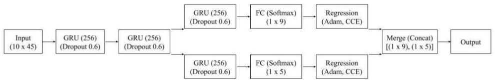


Hardware involved in the project, and their uses:

1. **Xbox One:** Run the NBA 2K game and receive streamed input to control the player
2. **Linux PC:** Capture the Xbox-streamed gameplay, detect the in-game objects, and forward keyboard-to-controller signals to the Windows PC
3. **Windows PC**: Receive keyboard inputs from the Linux PC and forward the inputs to an Xbox controller emulator, which plays the game on Xbox.

Decision Network

We built the agent's decision-making network ourselves, passing a 10 x 45 array representing the bounding boxes (9 objects x 5 coordinates) for each class over the last second (10 FPS). We used shared GRU layers before splitting into two branches to separately predict the agent's movement and action. After passing through one more GRU layer, a FC layer with softmax activation, then the regression layer, the agent takes the movement and action with highest softmax output by signaling through the virtual controller back to the Xbox.



Results and Future Work

While the DeepBaller agent does not crush its opponents, we had a few promising results. The agent showed that it could in real time respond to the changing events of the game by making advantageous passes, putting itself in position to score, and even scoring. However in the end, the lack of training data was the biggest limiting factor of the network's performance.

Challenges we faced:

- Evaluating in real time with negligible lag
- Running our agent program in Linux while communicating with the Xbox through Windows
- Training robust YOLO weights to detect different opponents, configurations, etc.