Independent Study

Report-3, 28th February

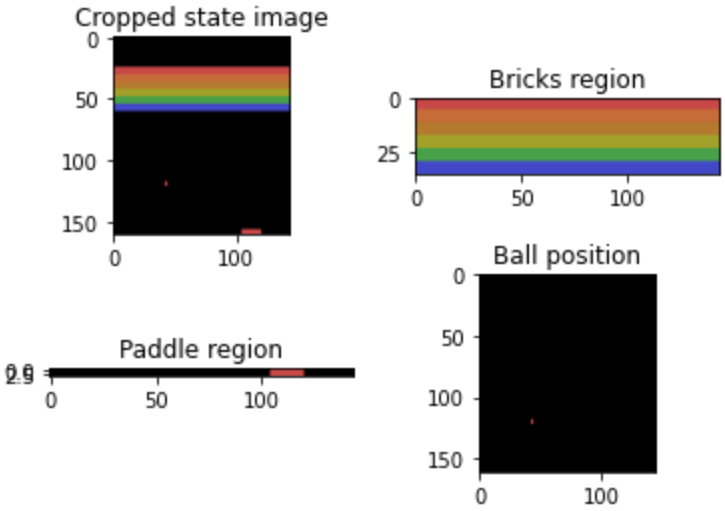
Submitted by: Vaibhav Garg(20171005), Kartik Gupta(20171018)

Feature Extraction for PiRL

Since we won't be feeding the raw image directly to our synthesised program, we had to come up with certain descriptive features that could act as the sensors for our PID controller based program. We decided to capture three important things from the frame:

1. Boolean array indicating whether (i, j)th brick is broken or not.
2. Position of the ball
3. Position of the paddle

OpenAI Gym didn't provide functionality for providing these values, so they had to be manually extracted by separating the image in different regions and some basic image processing. The figure below shows how the features look like.



Extracted Bricks array

```
[[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
 [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]]
Paddle Index
(array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]), array([104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119]))
Ball Index
(array([118, 118, 119, 119, 120, 120, 121, 121]), array([42, 43, 42, 43, 42, 43, 42, 43]))

Neural Oracle

Oracle is the actual RL algorithm that PiRL tries to generate a DSL code for or basically tries to imitate. As part of our current phase, we were trying to get a decently performing deep RL based Oracle. We tried various experiments to determine the best oracle.

The following table summarises the experiments.
For the last experiment we sent the last 4 frames to the network as a single state. This allowed the network to capture stuff like the ball's movement direction and the effect that the ball touching a brick has (since the brick disappears on the next frame).

Input	Past Frames	Architecture
Features	1	Fully Connected Network
Full Image Frame	1	CNN
Full Image Frame	4	CNN

We found that out of all the above models, the last model trained the best. But the training takes really long, hence we have not been able to ascertain if any of the first 2 models will catch up later in the training.
We found pretrained weights for the last model online. We were able to verify that it works really well. It is able to consistently clear all the bricks in each level.

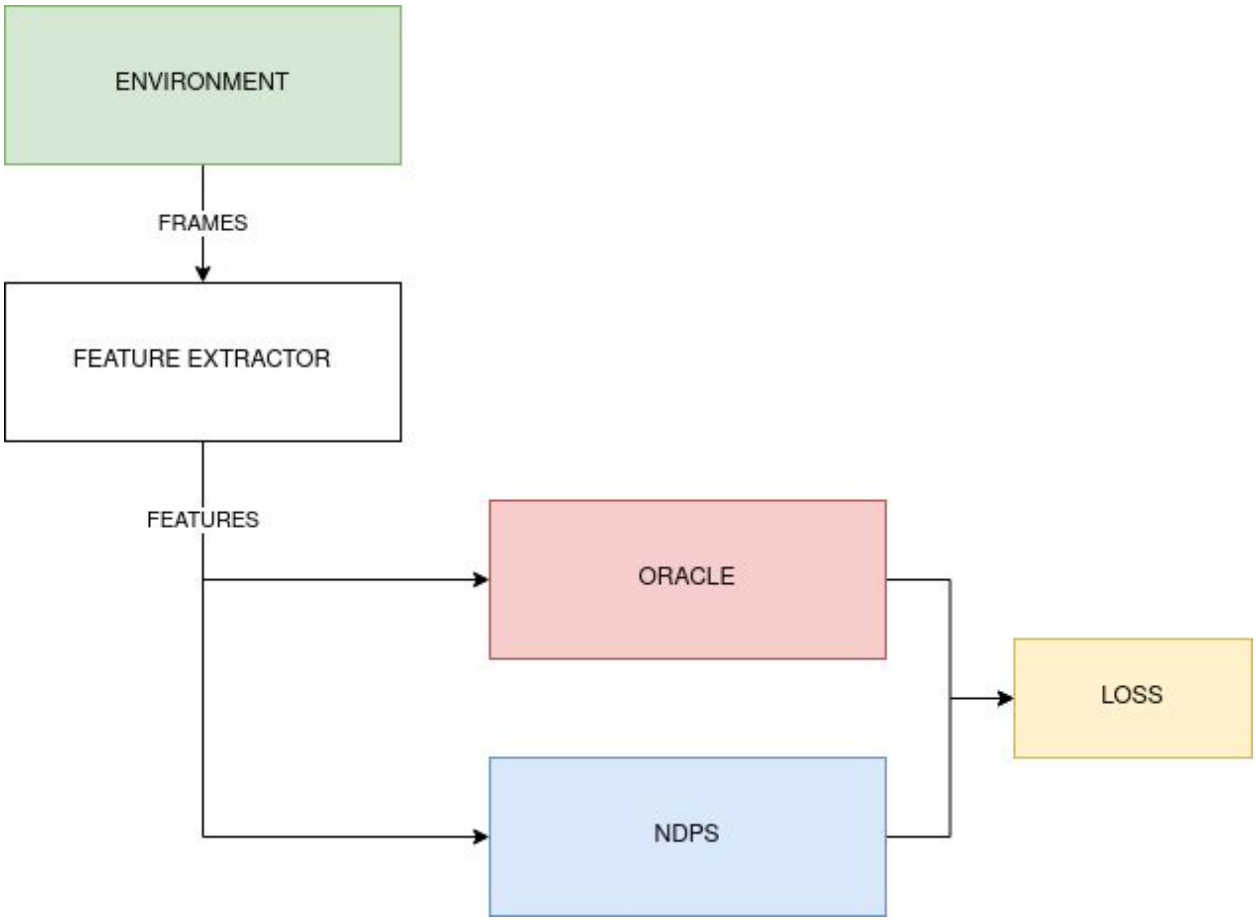
Results

The pretrained model consistently gets a really good score of 358, across 100 runs.

Using Oracle with PiRL

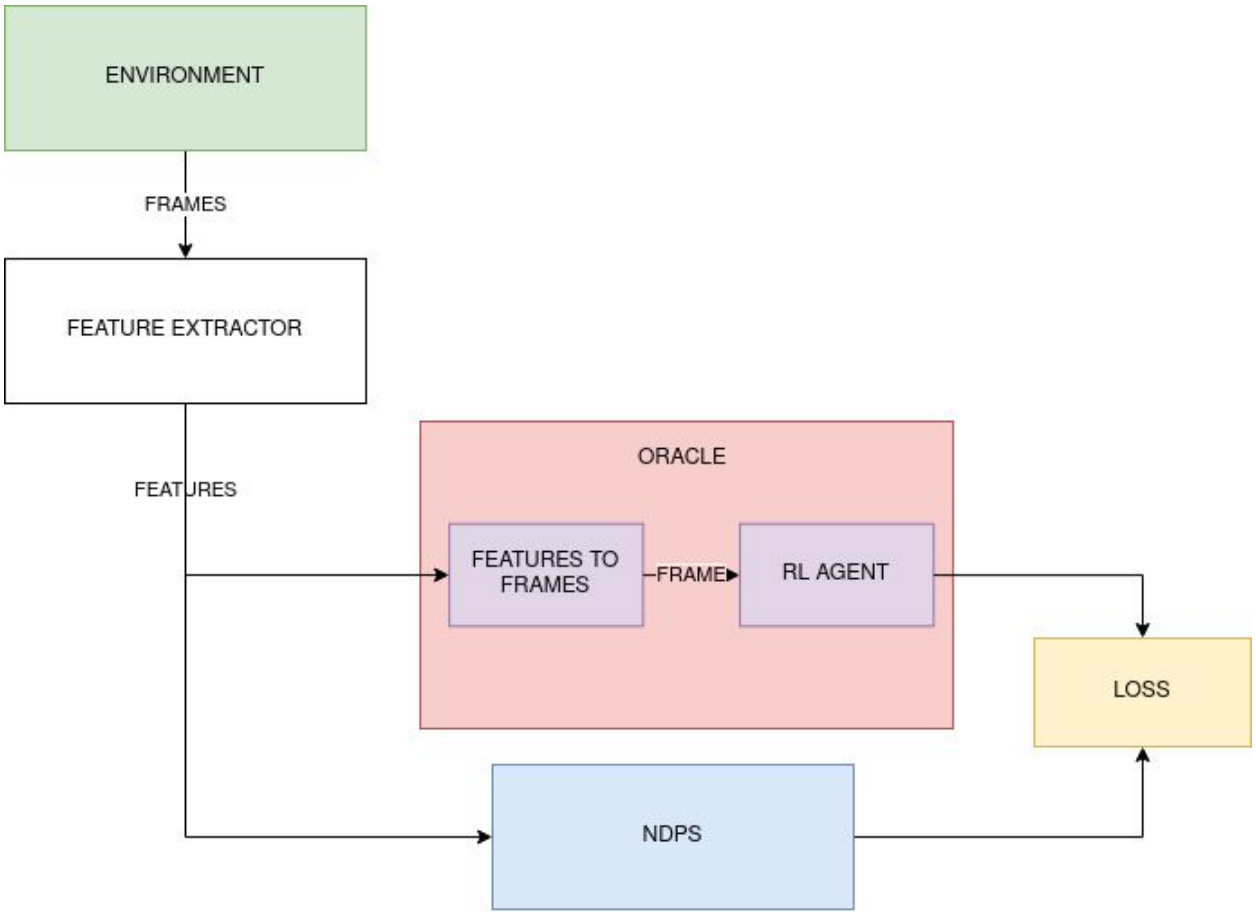
Features as Input

1. Get frames from the environment.
2. Extract features from the frame.
3. Generate history using these features from the oracle, and use that to guide from the NDPS algorithm.



Frames as Input

1. Get frames from the environment.
2. Extract features from the frames.
3. The oracle has a submodule which converts features back to frames for feeding into the RL agent.
    a. Direct frames are not used so that our oracle also has essentially the same amount of information to work with as the NDPS. In case our features miss out on some information, the submodule will ensure that the reconstruction does not add any extra information.
4. The output from the oracle is used to guide the NDPS algorithm.



Code

GitHub public repository link: https://github.com/VAIBHAV-2303/PiRL

Next Phase

1. We plan to start writing the code for Neural Directed Program Search which will try to generate a program which imitates the Neural Oracle explained above.
2. We will be parallely investigating other 2 oracle models, since they directly work with less information from features.