RL agent to play FIFA 14 skill games

Megha Agarwal[1], **Prashant K. Sharma**[2],**Shinjan Mitra**[3],**Urmi Saha**[4]
173059005[1],173059007[2],17305R005[3],17305R008[4]
{meghaag,prashants,shinjan,urmisaha}@cse.iitb.ac.in
Indian Institute of Technology, Bombay

Abstract

Train an agent to play skill games for FIFA 14. Agents have been demonstrated to take free-kicks in FIFA 18 using deep Q-learning with an average score per kick to be 0.5. We plan to demonstrate the performance of an agent on other types of skill games such as dribbling, passing and increasing the baseline score by at least 10%.

Introduction

One of the most interesting applications of Reinforcement Learning is playing a PC game since it presents several learning challenges that we humans excel at.We have taken up the problem of implementing Q-learning to make an agent play FIFA 14 skill games.

The problem was formulated as a Markov Decision Process with continuous state space, defined as follows

- **agent**: the player which takes an action like shoot, pass, through ball
- **actions**: set of controls pertinent to each game. For e.g. move left, move right, shoot high, shoot low are the actions for the freekick bronze skill game. However, the advanced shooting silver game adds move up and move down as well.
- **reward**: FIFA skill games awards the player points if it is successful in its task like scoring a goal. For our agent, any increment in points is rewarded with +1, while a missed shot is rewarded with -1. Any other action produces 0. The points tally is displayed on the top right of the game window which has to be parsed using optical character recognition.
- **environment**: the game engine from which we extract state information and reward via screenshots
- **state**: a feature vector obtained by sampling the game screenshot

This work builds upon earlier work in pursuit of the same objective, and extends it to new games. We also developed a different model using supervised learning. However, this model was found to not perform as well as the baseline. But we were able to achieve significant performance improvements by modification of the action space.

The report contains detailed description of the problem statement taken up, the performance of the model and also talks about the challenges faced during implementation.

Related Works

This work is heavily based on the project (Trivedi 2018b) which creates an RL agent to play skill games in FIFA 18. This agent was demonstrated to play only the free kick skill game which presents a non-adversarial environment requiring a player controlled by the agent to score a free kick goal. A non-adversarial environment is one where only the agent's action affect the obtained reward. In comparison, an adversarial game has an "opponent", for instance, a shooting game has a goalkeeper which attempts to stop the player from scoring the goal.

(Trivedi 2018a) was an initial attempt by the same author to play FIFA using supervised learning. We attempt to use both techniques to improve the performance of the agent.

(Castellino 2018) extends (Trivedi 2018b) by introducing an LSTM network which trains its model using sequences of frames instead of single screen shots in the above demonstrations. This seems to enable the agent to play shooting skill games with reasonable reward by exploiting the temporal features of the players and the ball.

Methodology

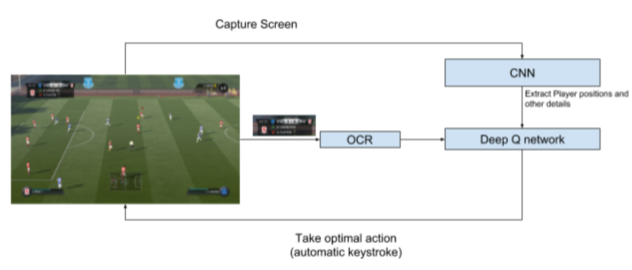


Figure 1: Overview of model

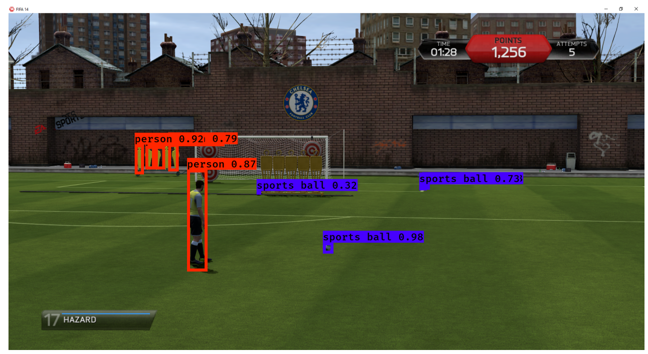
FIFA14 is distributed as an executable without any programmable APIs. Hence the only way to interact with the environment is by actual keypresses, and taking screenshots of the running game. The overview of agent - environment interaction is shown in Fig. 1.

1. Screenshot of the running game is taken using grabscreen function.
2. This image is resized and fed to MobileNet CNN which generates a feature map.
3. The feature map is a 128 dimensional vector which is then input to a fully connected neural network, giving action probabilities as output. For the started free kick agent, there are 4 actions, hence we obtain 4 output probabilities.
4. Q-learning selects the argmax from this output vector. We use ϵ-greedy Q-learning so that exploration and exploitation go on infinitely.
5. The chosen action is an integer value. This is converted to actual keypresses to give controls to FIFA.
6. Rewards are acquired by optical character recognition of the points bar on the top right corner. This is enabled by the pytesseract library.

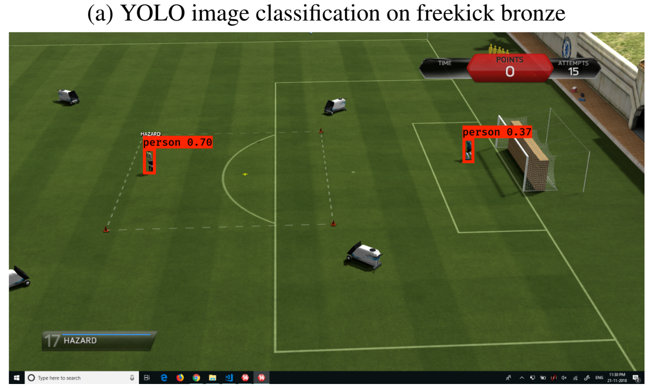
Our objective was to train the agent to play different types of games and test alternative models to train the agent. We chose the Free Kick Bronze and Advanced Shooting Silver skill games for this purpose. We ran several experiments on a a couple of different models with variations of parameters for both.

- The original code by (Trivedi 2018b), modified to work on FIFA 14. The modifications that had to be made were:
  - Shifting the pixels that had to be captured for reading the reward
  - Pressing the key **Q** instead of Enter for restarting the game.

  The baseline model was also tested after increasing the action space to 8 actions from 4. The move_left and move_right actions kept the key pressed for 0.25 sec. We added 2 more actions for each of them which kept the key pressed for 0.1 sec and 0.4 sec.
- Changing a portion of the Q-learning network to speed up action selection. The original code would run tf.Session for every action selection which was taking at least 10 sec to give output, since it would initialise GPU handles for Tensorflow. This was modified to call tf.Session only once at initialisation. The action selection time came down to within 1 sec.
- Using YOLO((Redmon and Farhadi 2018) image classifier to detect only the player, as illustrated in Fig. **??** and the ball, and feeding the vector of bounding boxes to the Q-learning approximation network. The center of the bounding boxes were calculated resulting in the vector [x1, y1, x2, y2] which was finally given as input to. This reduced the dimensionality of the input layer from 128 to 4. We expected this to greatly reduce computation time and improve accuracy since it will only act on the useful features. We noticed that YOLO was detecting multiple objects in the background as *person* along with the player, so we chose the detection with the highest confidence. This vector served as input to the Q-learning network which was now a fully connected neural net with 1 hidden layer of 8 neurons and an output layer with 4 neurons in the case of freekick and 5 in the case of advanced shooting.



(a) YOLO image classification on freekick bronze



(b) YOLO image classification on advanced shooting silver

Figure 2: YOLO image classification

Each experiment was run for at least 500 epochs and the average number of goals scored per game was recorded. Here, an epoch is considered to have ended when a shot on goal is taken, as that will either result in a goal or a failure.

Results and Discussions

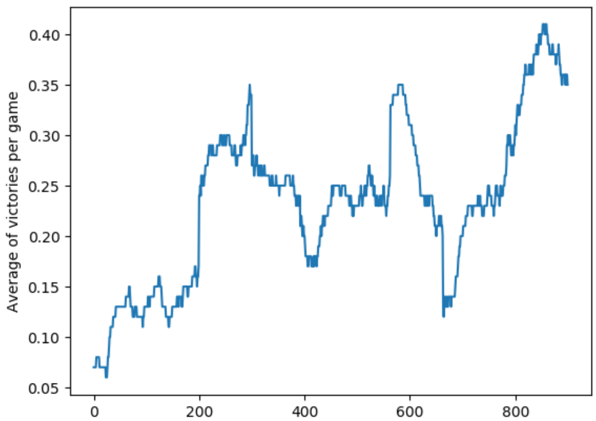


Figure 3: Baseline agent training

The agent was first trained using the model defined by (Trivedi 2018b). We used this as baseline for all the subsequent performance evaluations. The plot of the training till 1000 epochs is shown in Fig. 3. We notice steady improvement of the agent's goal-scoring rate as expected. We noticed that the game background is very important for the agent. We accidentally changed the background midway during learning which resulted in extremely poor performance after the change (after 680 epochs) as can be seen from the dip in Fig. 3.

Figure 4: Advanced shooting silver

Figure 5: Cropped image fed to CNN

This same model was trained anew on an adversarial game i.e. the advanced shooting silver. A snapshot from the game is shown in Fig. 4. This game involves the agent having to score a goal from distance while the goalkeeper will attempt to block the shot. Unlike the free kick game where the agent did not have to make decisions quickly, here the actions have to be faster otherwise the keeper will come out and kick the ball away. Additionally, we provided cropped image input to the CNN so that feature detection would be easy as shown in Fig. 5

The performance plot of this model over 1000 epochs is shown in Fig. 6. This agent also achieves a max performance of 0.6 goals per game which is quite appreciable.

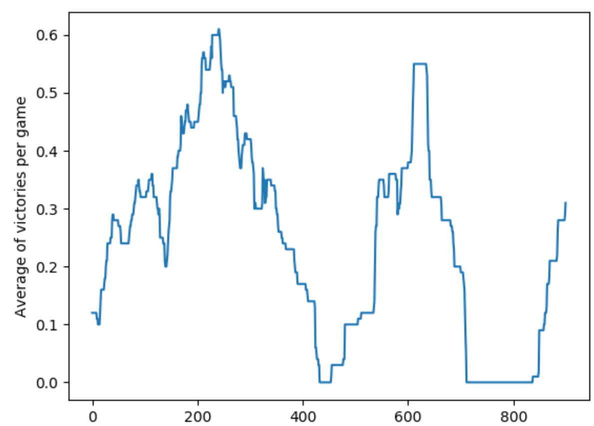


Figure 6: Performance of baseline agent on advanced shooting silver

The performance of the YOLO-based model was unexpectedly quite poor compared to the baseline. This is illustrated in Fig. 7 and Fig. 8. Not only did it not reach the same performance as the baseline model, but it was also slower to generate actions. This was due to a bottleneck with YOLO itself since the Q-learning network had a mere 65 trainable parameters for advanced shooting. We infer that this may be due to the model only being able to detect players and the ball. Without detecting the goal, it may not be able to decide the direction of taking shots, hence it was not able to converge to an optimum. Also for advanced shooting, it seemed to converge to a policy where the agent was only moving right until it moves out of the shooting area. This may have happened since the ball sometimes rolled into the net and gave it positive reward. It may have not received similar returns by shooting since the goalie would stop the shot.

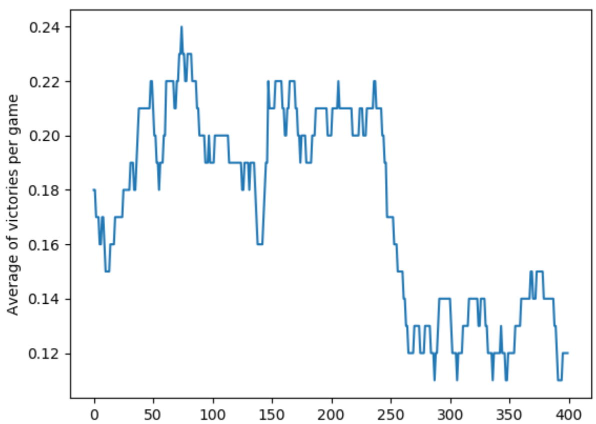


Figure 7: Performance on freekick using YOLO

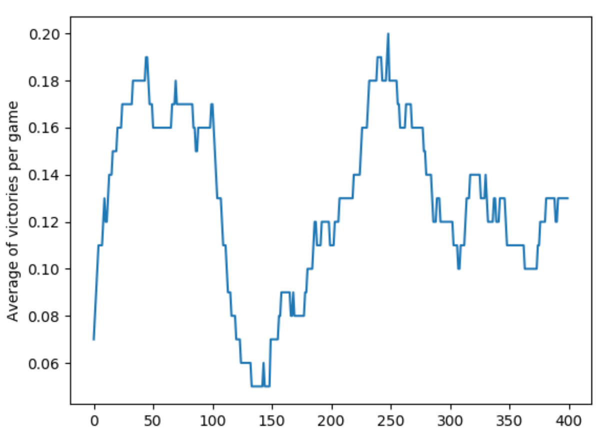


Figure 8: Performance on advanced shooting using YOLO

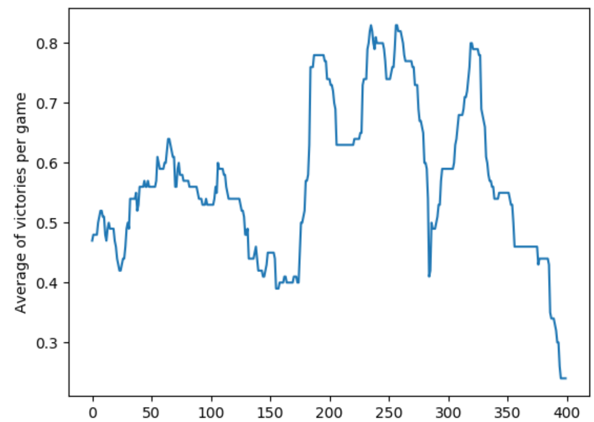


Figure 9: Performance on freekick using 8 actions

The result of extending the baseline model by adding more granular actions is show in Fig. 9

Challenges

- We started training the agent using FIFA 18 which is a later release of the same game but with enhanced graphics and user customization. However we had to shift to FIFA 14 due to the following issues:
  - The game and agent had to run on the same system so that keypresses could be sent to the game window. Since we were constrained to use only one system, the graphics demands of FIFA 18 left very little processing resources for the CNN. Given our resource constraints (a Core i3 laptop with nVIDIA GeForce 940MX), each action selection was taking too long leading to episode termination in case of the adversarial games and overall lengthy training times (over 10 hours for 500 epochs).
  - If FIFA 18 is started before the agent code then the agent is not able to acquire CUDA handle and throws the following error. tensorflow/stream_executor/cuda/cuda_dnn.cc:373] Could not create cudnn handle: CUDNN_STATUS_INTERNAL_ERROR However if the game is started after the agent, then it works fine.
- Agent code was running extremely slow due to a line which was creating a new tf.Session on every action. Changed this to a session creation at init. This solved the issue and action selection came down to 1 sec.

Conclusion

The project was aimed at developing an agent to play most of the FIFA 18 skill games. We had to move to FIFA 14 due to resource constraints and were able to demonstrate effectiveness of reinforcement learning in playing adversarial games. Given greater computing resources with lesser prediction times, we believe the agent would perform far better, since in most case it was unable to react before the adversary had defeated it. Our attempt at using supervised learning to explicitly provide the agent with position coordinates did not work out as expected. However we were able to obtain significant performance improvements by simply expanding the action space.

There is much to done if we are to develop an agent that is capable of playing all the skill games and ultimately the full game itself. We think that extending the algorithm to support multi-granularity action spaces may hold the key to this. Maybe future work on the lines of (Lee et al. 2018)

References

Castellino, L. 2018. Reinforcement learning with fifa and keras. *Medium*.

Lee, K.; Kim, S.-A.; Choi, J.; and Lee, S.-W. 2018. Deep reinforcement learning in continuous action spaces: a case study in the game of simulated curling. In *International Conference on Machine Learning*, 2943–2952.

Redmon, J., and Farhadi, A. 2018. Yolov3: An incremental improvement. *arXiv*.

Trivedi, C. 2018a. Deepgamingaififann: Building a deep neural network to play fifa 18. *Towards Data Science*.

Trivedi, C. 2018b. Deepgamingaififarl: Using deep q-learning in fifa 18 to perfect the art of free-kicks. *Towards Data Science*.