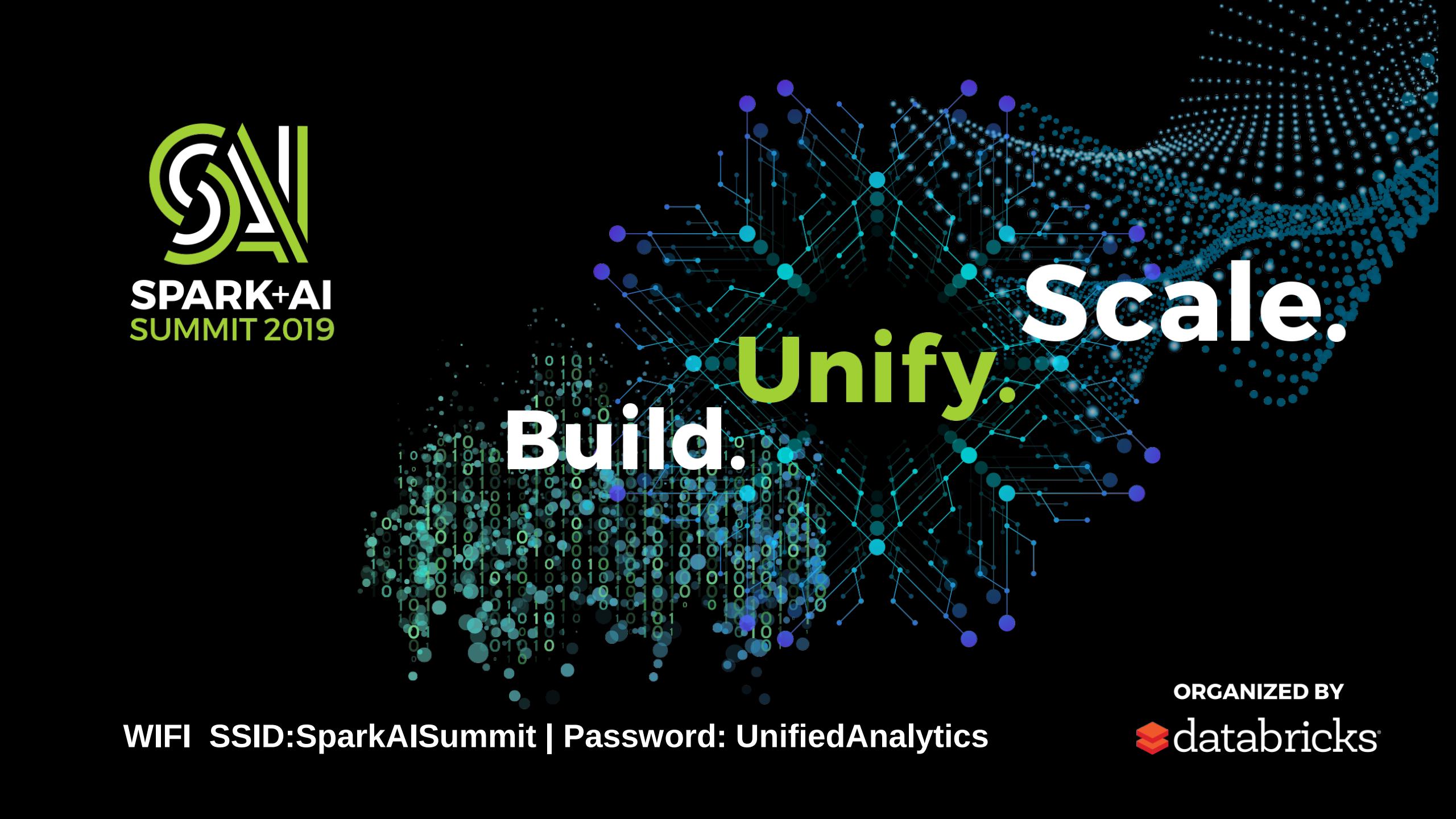

SPARK+AI
SUMMIT 2019
Build.
Unify.
Scale.
WIFI SSID:SparkAISummit | Password: UnifiedAnalytics
ORGANIZED BY
databricks

Game Playing Using AI on Apache Spark

Yuhao Yang, Intel

Acknowledgement: Huang, Shengsheng; Yu, Shan

Agenda

- Analytics Zoo
- Problem statement
- Results and Observations

Analytics Zoo

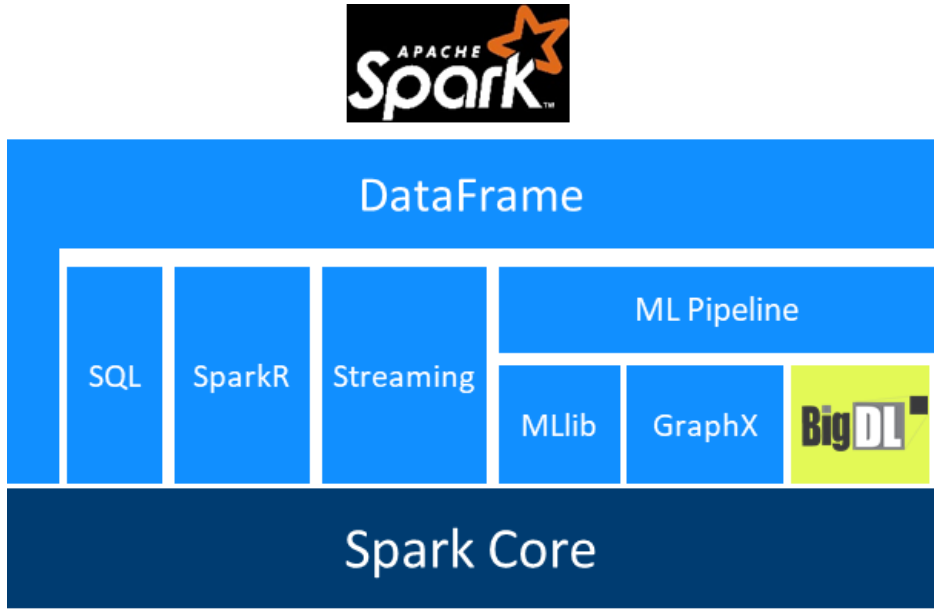
Distributed TensorFlow, Keras and BigDL on Apache Spark

High-Level Pipeline APIs	Distributed TensorFlow and Keras on Spark Native deep learning support in Spark DataFrames and ML Pipelines Model serving API for model serving/inference pipelines
Reference Use Cases	Anomaly detection, sentiment analysis, fraud detection, chatbot, sequence prediction, etc.
Built-In Deep Learning Models	Image classification, object detection, text classification, recommendations, sequence-to-sequence, anomaly detection, etc.
Feature Engineering	Feature transformations for • Image, text, 3D imaging, time series, speech, etc.
Backends	Spark, TensorFlow, Keras, BigDL, OpenVINO, MKL-DNN, etc.

https://github.com/intel-analytics/analytics-zoo/

Bringing Deep Learning To Big Data Platform

- Distributed deep learning framework for Apache Spark*
- Make deep learning more accessible to big data users and data scientists
  - Write deep learning applications as standard Spark programs
  - Run on existing Spark/Hadoop clusters (no changes needed)
- Feature parity with popular deep learning frameworks
  - E.g., Caffe, Torch, Tensorflow, etc.
- High performance (on CPU)
  - Built-in Intel MKL and multi-threaded programming
- Efficient scale-out
  - Leveraging Spark for distributed training & inference

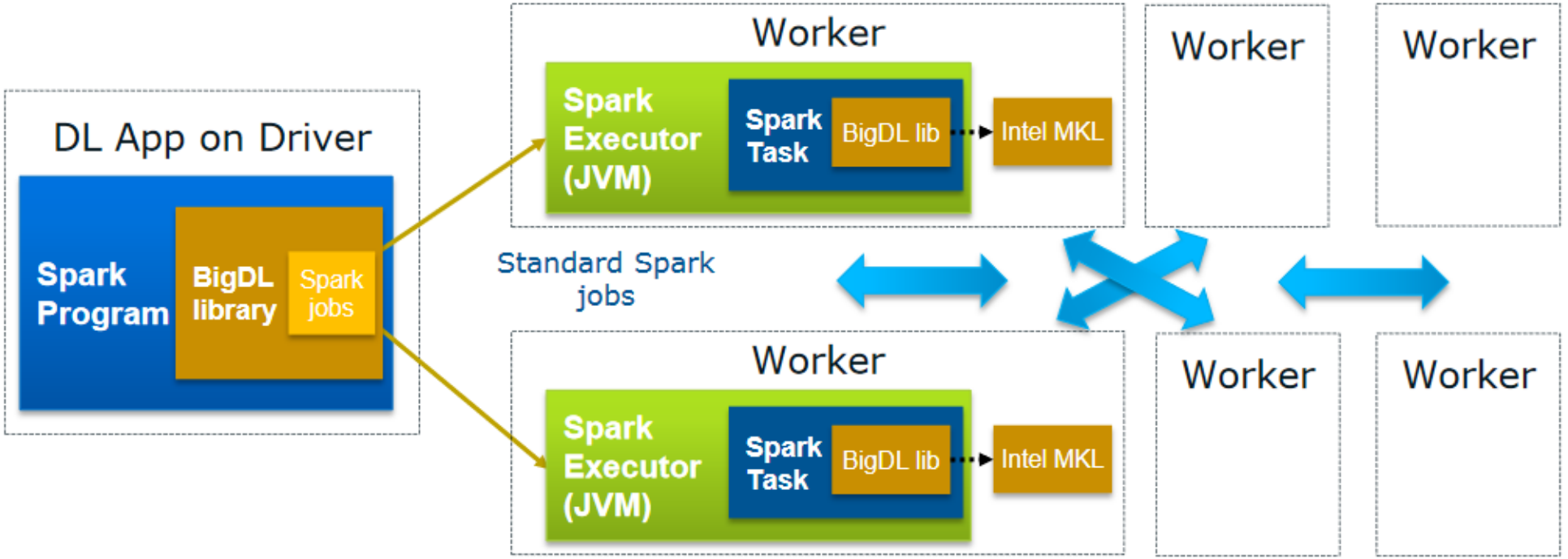


https://github.com/intel-analytics/BigDL

https://bigdl-project.github.io/

Architecture

- Training using synchronous mini-batch
- Distributed training iterations run as standard Spark jobs
- Data parallelism
- All-reduce

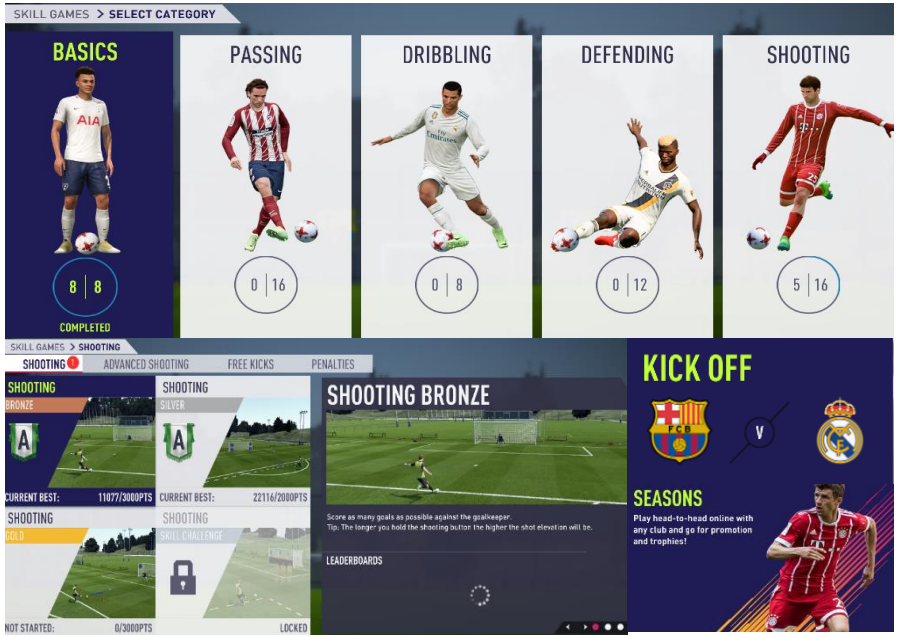


AI-based Game Playing

- Why build AI agents to play game
- Important work/playground for AI game playing:
  - Go: AlphaGo/AlphaZero
  - Atrai Games
  - StarCraft/StarCraft II
- Out attempts to build AI to play game
  - Use AI to play a game that has better connection with real world, hopefully transferable to general AI task.
  - Use Spark/distributed technology for high demands of computation and hybrid and complex computation

We choose to play FIFA

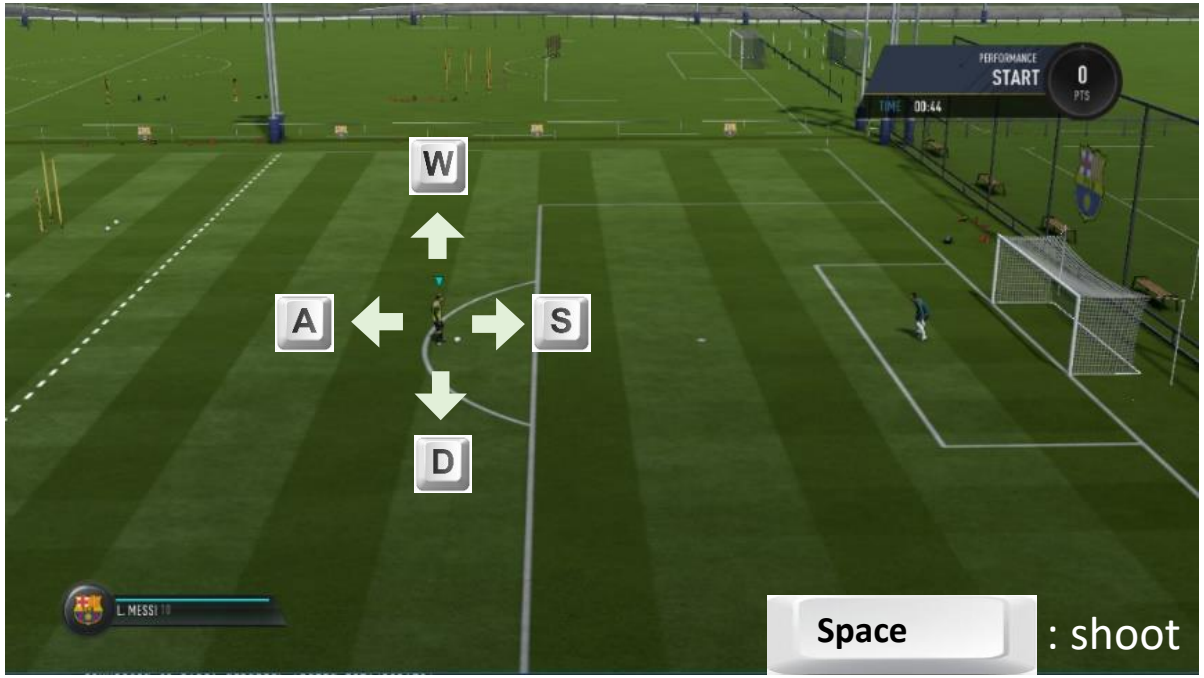
- FIFA18: a real-time 3d modern football simulation video game.
- Why FIFA
  - High resolution screen (1920*1280)
  - Multiple game modes and difficulty levels.
  - Connections to real-world
- DeepMind’s recent attempt on 2v2 soccer game on MuJoCo. Next target of full-court FIFA.



Introduction to Shooting Bronze

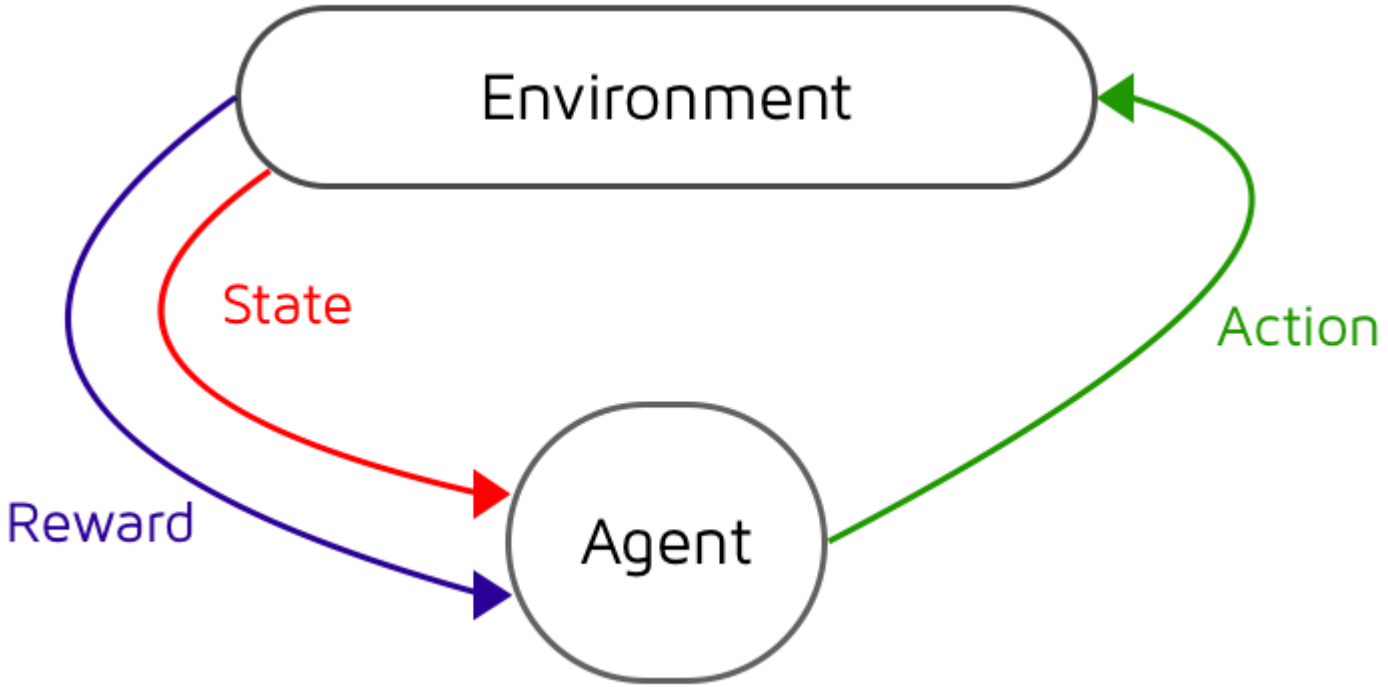
– our major experiment environment

- ***Shooting*** is one of the mini-games in FIFA, ***Bronze*** is the easy level. There're also *silver* and *gold* levels.
- Game mode
  - Player & guard 1v1
  - Control player to shoot
  - Game reset after shoot (different start location)
  - Goal: get more score in 44s
- Evaluation
  - Single shoot:
    - Score <= 200: miss
    - 200 < score < 1200: hit
  - Total score:
    - E (<3000) / D (<4000) / C (<6000) / B (<8000) / A (>8000)
- Keyboard control
  - A/S/W/D: left/right/up/down
  - Space: shoot



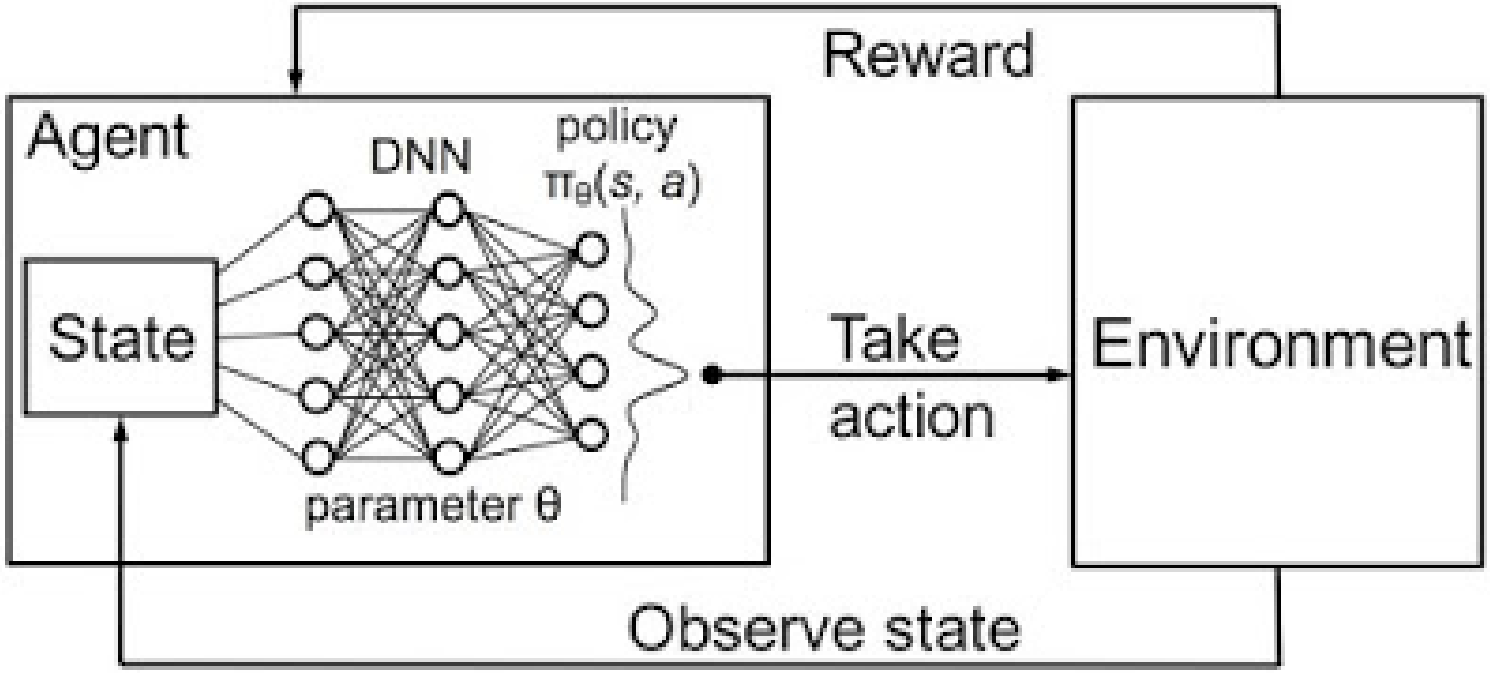
Reinforcement Learning (RL)

- RL is for Decision-making



Deep Reinforcement Learning (DRL)

Agents take actions (a) in state (s) and receives rewards (R)
Goal is to find the policy (π) that maximized future rewards



http://people.csail.mit.edu/hongzi/content/publications/DeepRM-HotNets16.pdf

Approaches to Reinforcement Learning

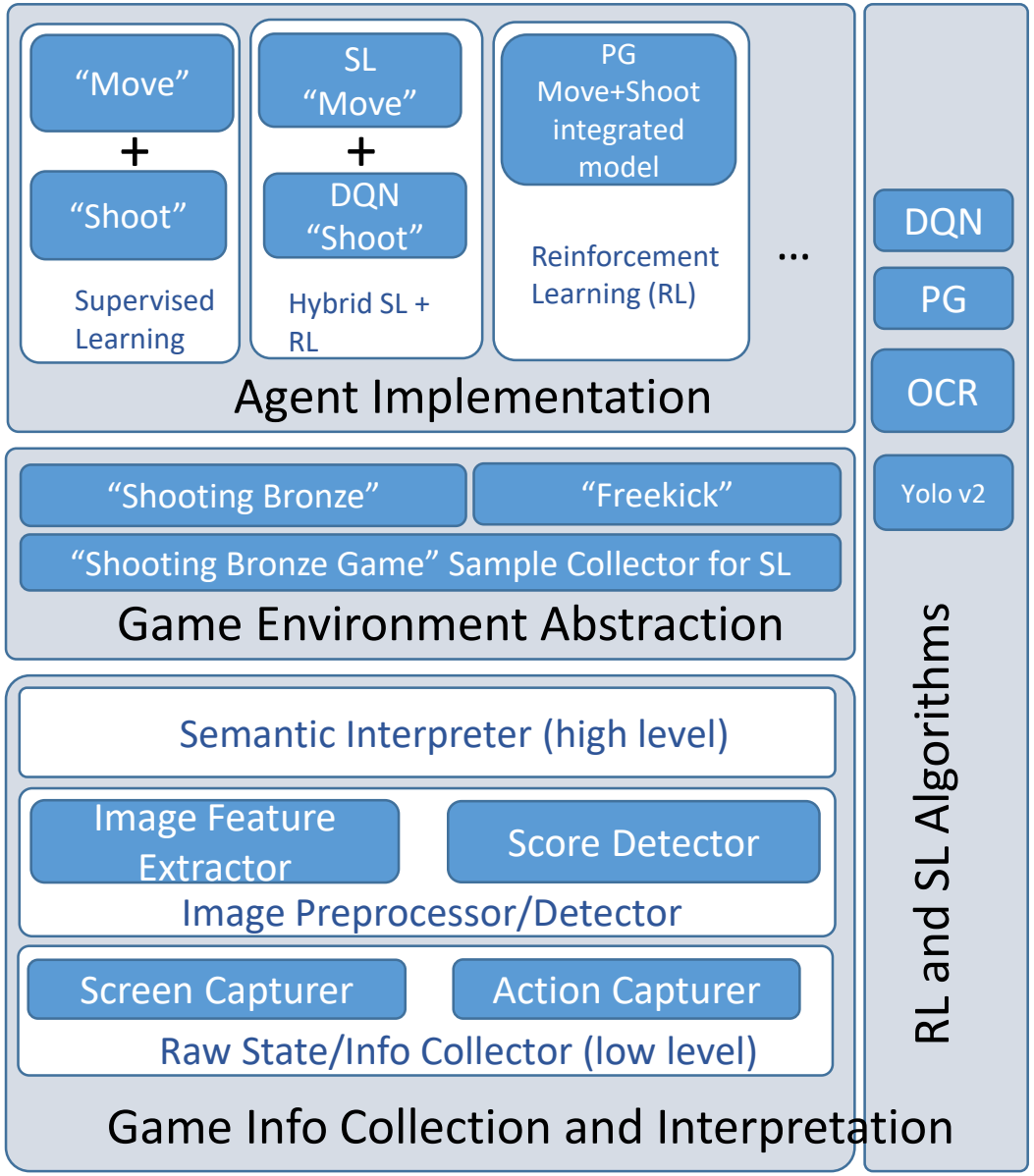
- Value-based RL
  - Estimate the optimal value function Q*(S,A)
  - Output of the Neural network is the value for Q(S, A)
- Policy-based RL
  - Search directly for the optimal policy π^*
  - Output of the neural network is the probability of each action.
- Model-based RL

AI on FIFA platform

A platform for experimenting various agents and algorithms.

- It interacts with FIFA game by capturing screenshot as input, and issue keyboard input to control the game

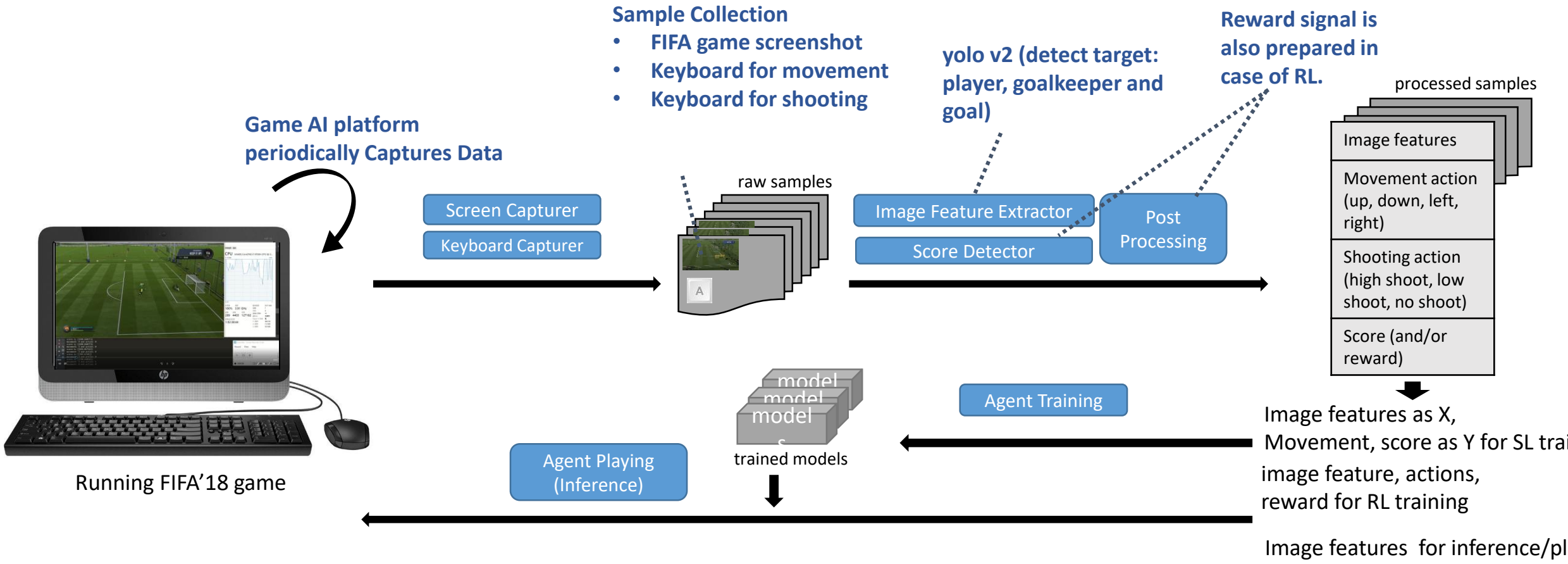
- Major components
  - **Game Info Collection and Interpretation**: Hierarchical game info processing, extraction and understanding
  - **Game Environment Abstraction**: 2 available – “Shooting Bronze”, “Free-kick”
  - **Agent Implementation**: 3 available – SL, RL and hybrid SL+RL.
    - SL: Supervised Learning
    - RL: Reinforcement Learning



Notes 1: This is the score calculated by the game, considering scoring rate in ~45 secs， the direction of scoring and total number of shooting

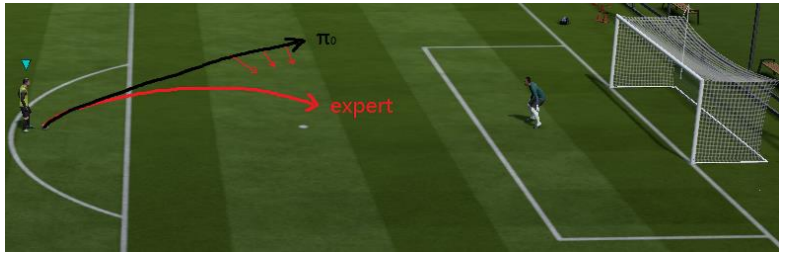
Notes 2: This is calculated as $\frac{number\ of\ successfull\ shooting}{total\ number\ of\ shooting\ attempts}$

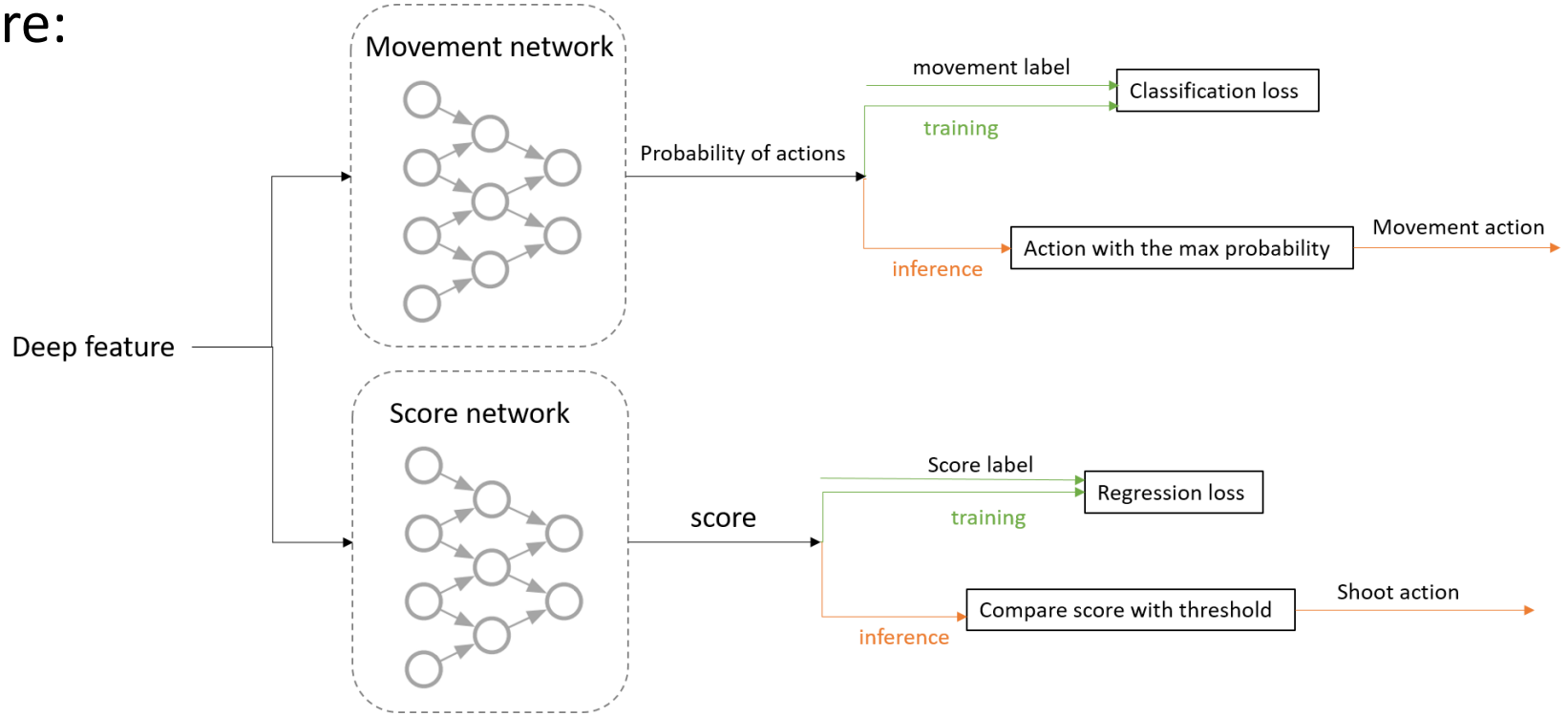
Internal Workflow



Agents implementation (SL Agent)

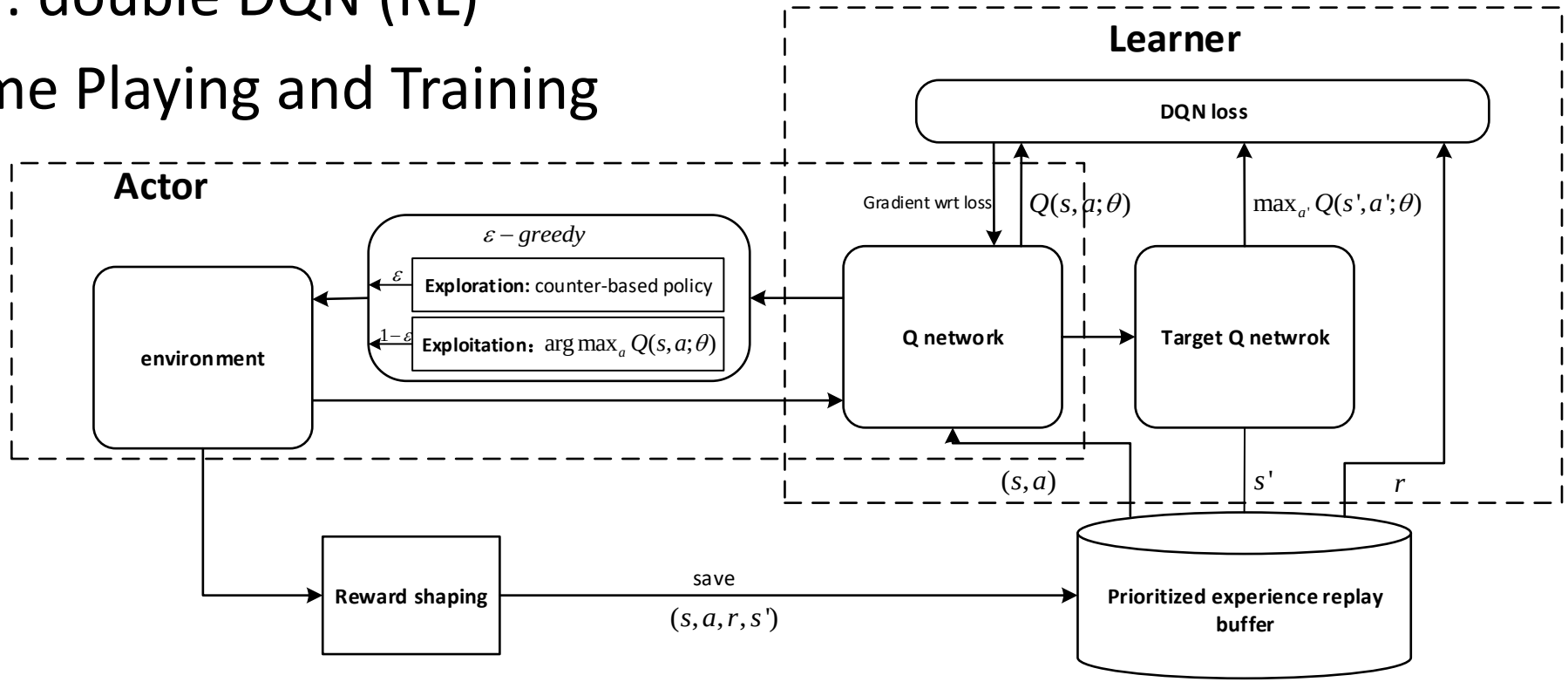
- Imitation learning
  - Behavior cloning: supervised learning by classification.
  - DAgger (Dataset Aggregation)
- Structure:





Agents implementation (SL+RL Agent)

- Movement predictor: pre-trained model with imitation learning (SL)
- Shoot predictor: double DQN (RL)
- Distributed Game Playing and Training



Demo

- Human Demo (the demonstrator used to train SL agent)
- SL agent Demo (plays better than the human it learnt from)
- SL+RL agent Demo

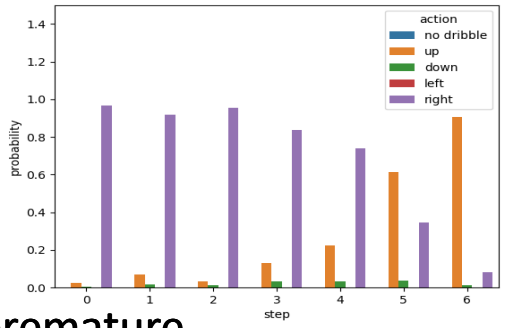
Typical trajectory analysis (SL+RL agent)

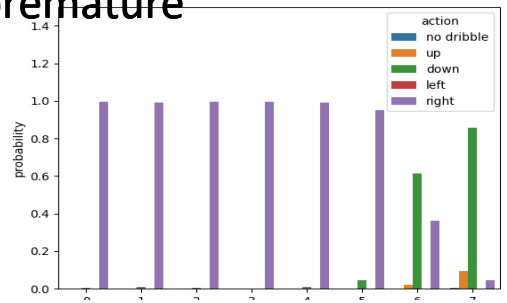
Typical trajectories

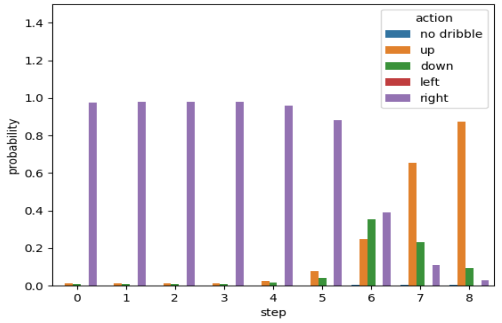
Movement policy (SL)

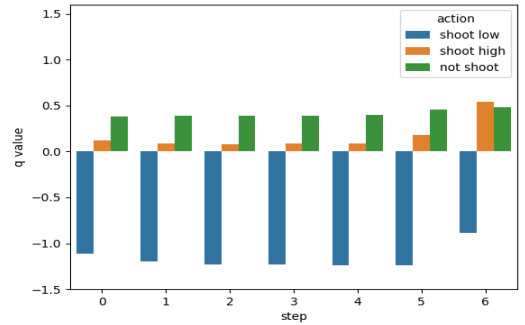
Shoot Q-value (RL)

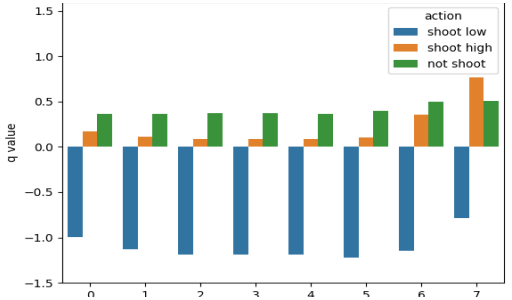
premature

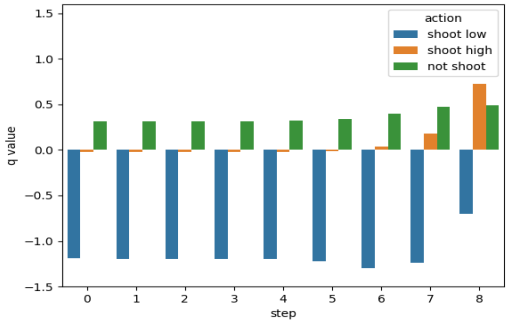












Result

		score	Score_ratio	Convergence speed
Human	beginner	5846.69	50%	-
	master	10112.78	92%	-
	demonstrator	7284.98	84.96%	-
Agent	SL	10345.18	92.54%	-
	RL	5606.31	40.25%	1069.5 epochs
	SL+RL	10514.43	95.59%	749.6 epochs

- SL+RL agent has the best performance and compare to the master level human players.
- SL agent outperforms its demonstrator.
- Performance of RL agent is similar to beginner human players

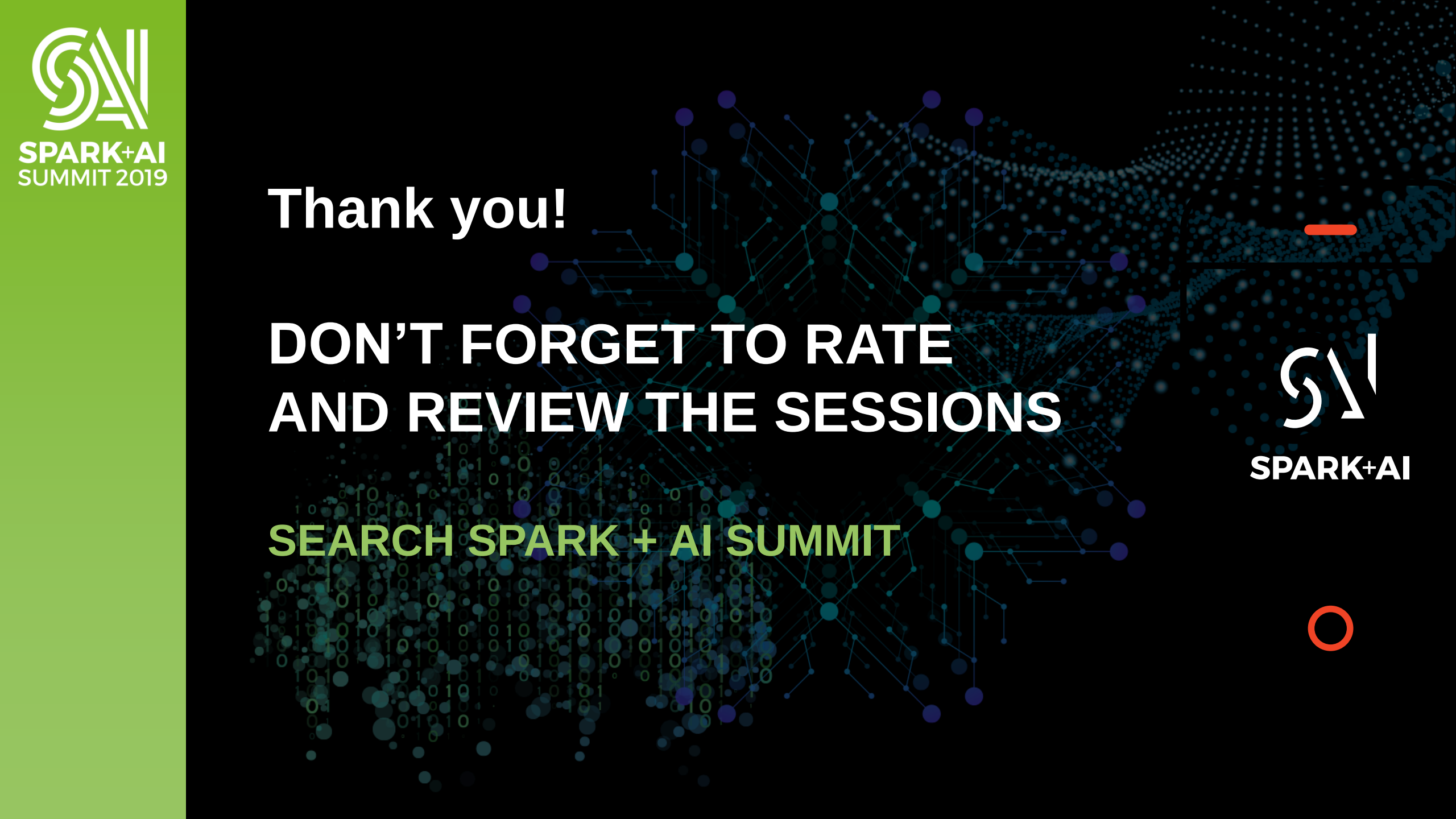
Key Takeaways

- RL + SL
- Spark is useful for supervised Learning and imitation learning training, especially when train data is very large (which is very possible if scenario is very complex, e.g. full-court).
- Latency problem on small batch updates can be mitigated without hurting agent performance, as long as the (RL) algorithms are carefully chosen and tuned.

premature

Future Work

- Better performant RL agents
- New hybrid agent implementations and more data for imitation learning.
- More difficult game modes (“Shooting Silver/Gold”, Full-court)
  - Need small object detection – (ball, path markers, goal keeper, etc.)
  - More complex scenes means more semantic info to process in game states or complicate keyboard actions.
  - Need to handle sparse direct reward signals, i.e. scoreboard – may need indirect performance indicators such as holding time


SPARK+AI
SUMMIT 2019
Thank you!
DON’T FORGET TO RATE
AND REVIEW THE SESSIONS
SEARCH SPARK + AI SUMMIT
SPARK+AI

Acknowledgements

(Alphabetical)

Huang, Shengsheng(Shane)

Yu, Shan

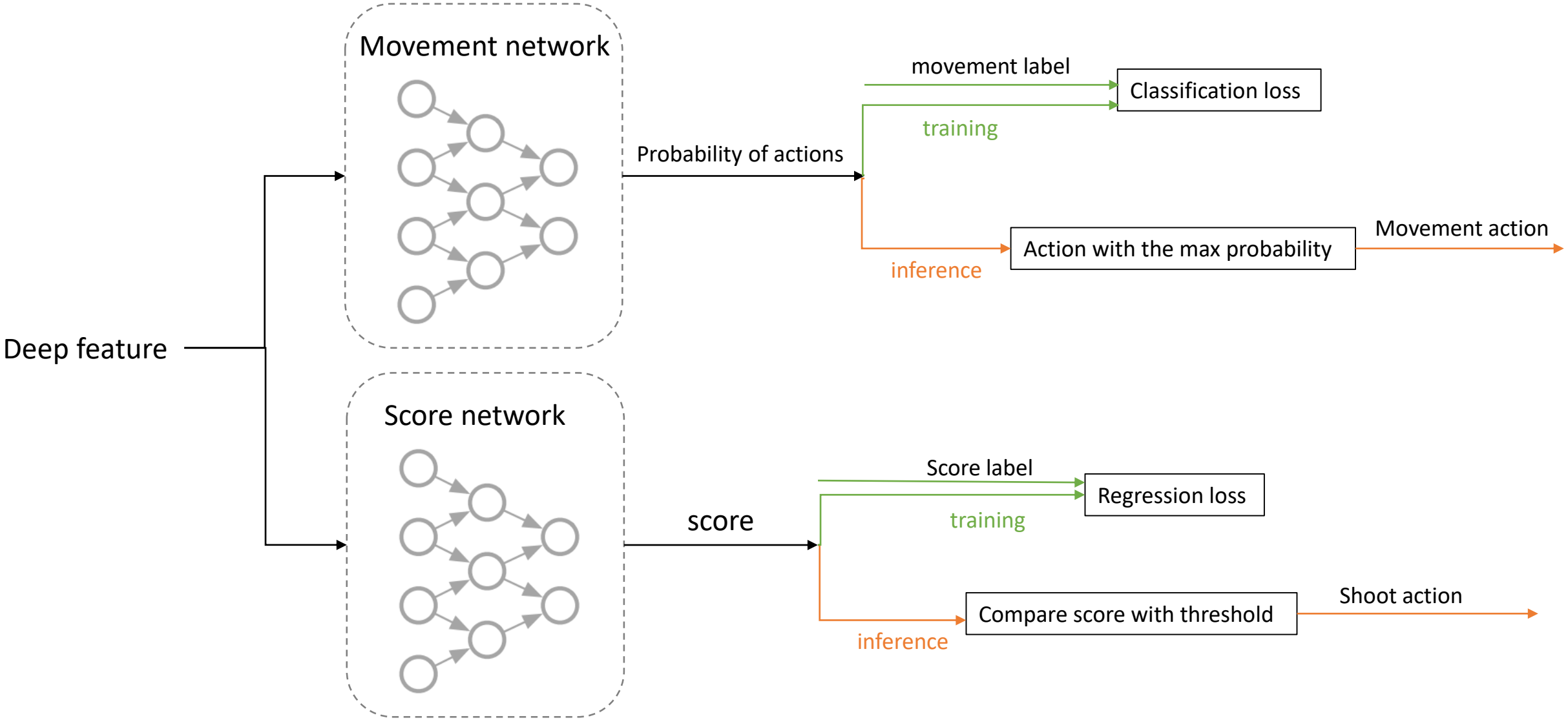
Appendix

miscellaneous

- Some tricks employed to make SL+RL more steady
  - In exploration stage of epsilon-Greedy, use a trick to prevent shooting happen too early.
  - Update DQN network every n-samples instead of 1-sample (Train at the end of each round of shooting instead of each step)
  - Reward shaping

$$R(s_t, a_t, a_{t+1}) = \begin{cases} -1 \text{ s.t. } s_t = \textit{initial state} \text{ and } a_t = \textit{shoot} \\ -0.1 \text{ s.t. } s_t \neq \text{terminal and } a_t \neq \textit{shoot} \\ 1 \text{ s.t. } s_{t+1} = \textit{goal} \\ -1 \text{ s.t. } s_{t+1} = \text{not goal} \end{cases}$$

SL Algorithm Chart



System performance

Module	GPU	Tensorflow	ZOO (tfpark)
Feature_extractor			
movement_predictor	-		
Score_predictor (SL)	-		
dqn_actor (RL)	-		

- For inference only: analytics-zoo(w/ tfpark)'s overhead is ignorable
- For training:
  - For SL, training overhead does not matter, and distributed training had advantage for even larger dataset
  - For RL,
    - On-policy algorithms may not be a good fit using Spark due to the delay (frequent update on small batches make the overhead obvious).
    - We tried do update every n-samples for DQN. Delay can be tolerated and the scores is still not bad.

Tfpark Intro

- Do we need this?