Deep Reinforcement Learning Recommenders using RayOnSpark

Guoqiong Song

Intel Deep Learning R&D engineer

May 27th 2021

intel.

Agenda

- Why Reinforcement Learning Recommenders?
- What is Analytics Zoo and RayOnSpark?
- How to build reinforcement learning recommenders using RayOnSpark
- Summary

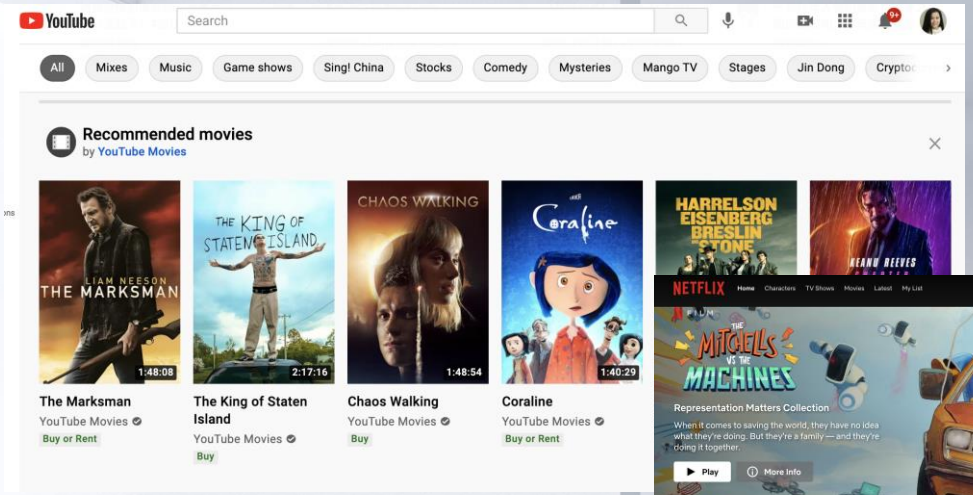
intel.

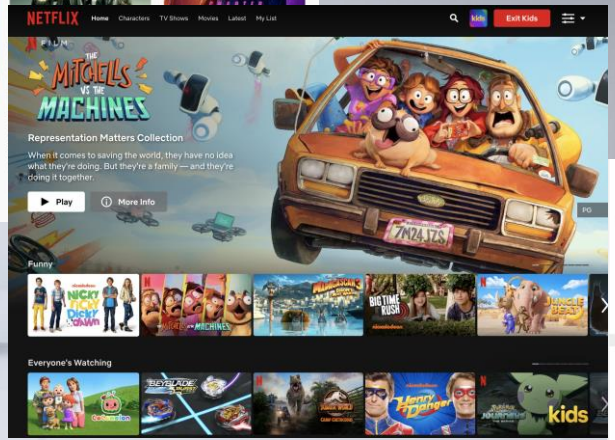
Agenda

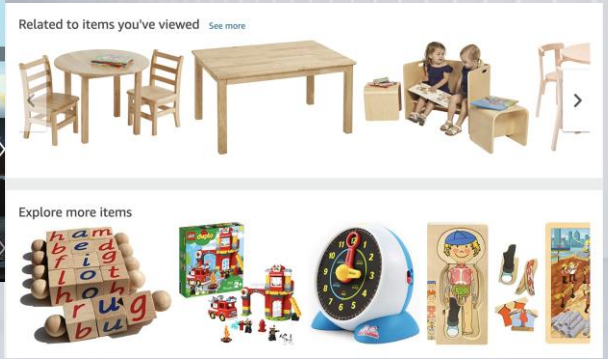
- **Why Reinforcement Learning Recommenders?**
- What is Analytics Zoo and RayOnSpark?
- How to build reinforcement learning recommenders using RayOnSpark
- Summary

intel.

Recommendations everywhere







intel.

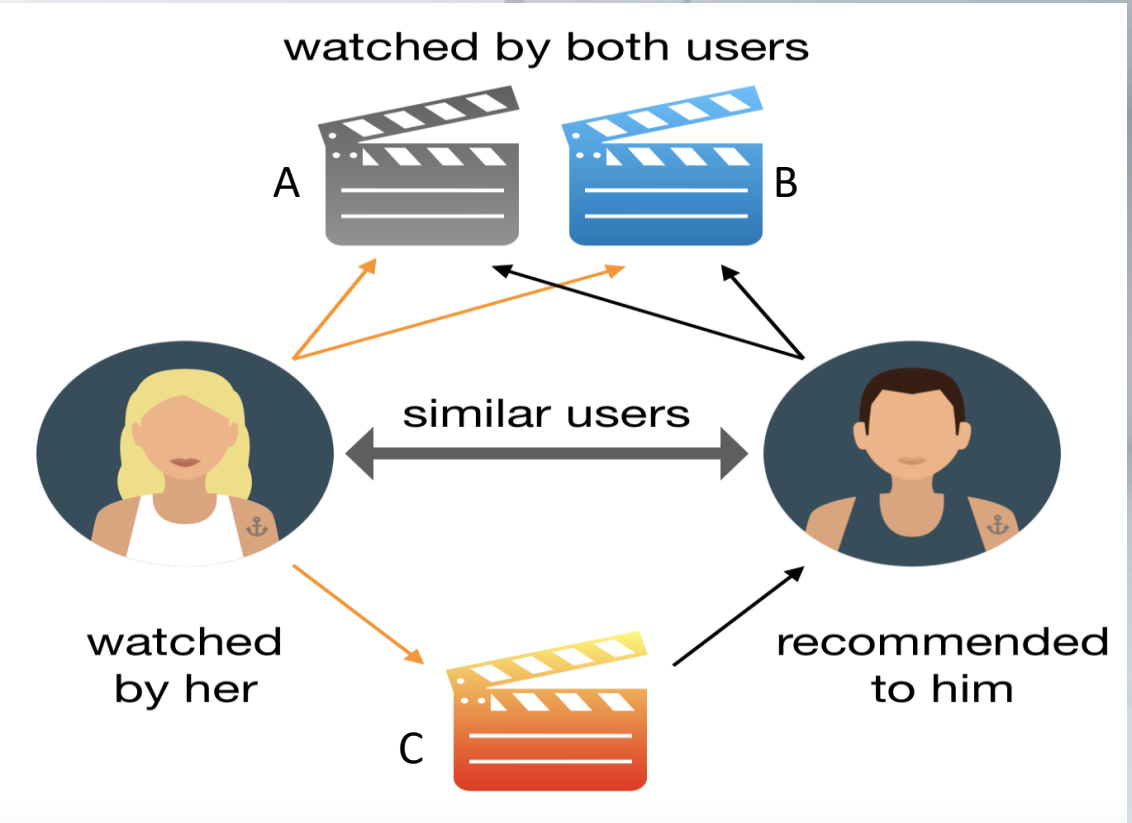
Recommenders everywhere

Users

Items

intel.

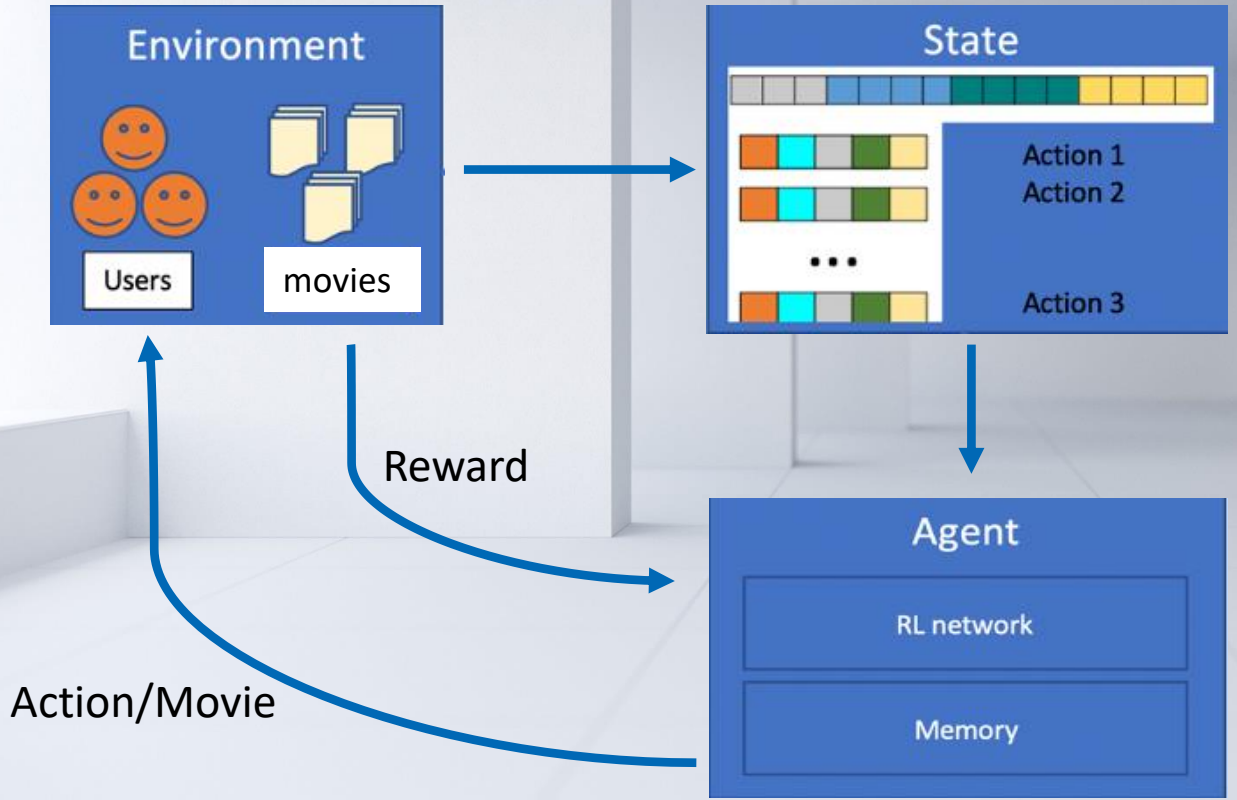
Traditional recommenders challenges



- **Static procedure**
- **Focus on immediate feedback**

intel.

Reinforcement learning recommenders



- Sequential decision process
- Focus on both immediate and long-term rewards

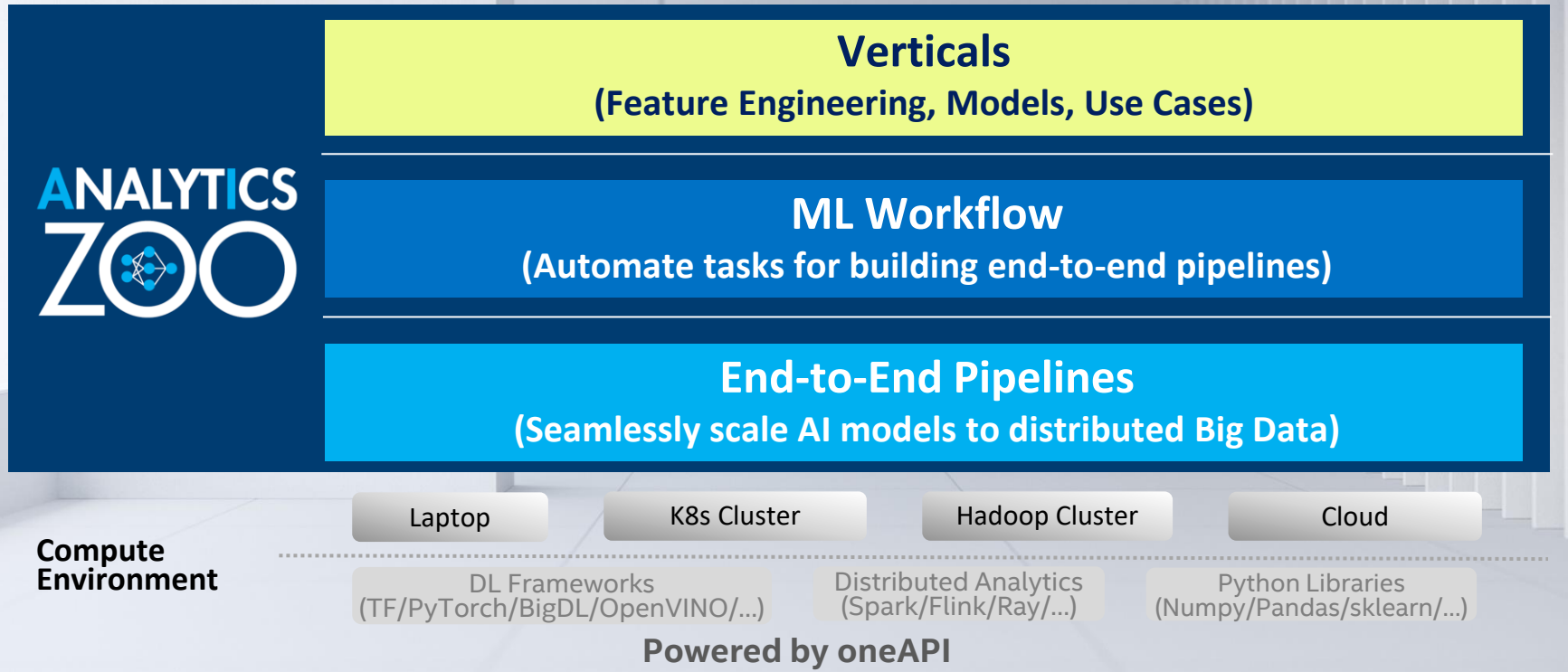
intel.

Agenda

- Why Reinforcement Learning Recommenders?
- **What is Analytics Zoo and RayOnSpark?**
- How to build reinforcement learning recommenders using RayOnSpark
- Summary

intel.

Analytics Zoo: Software Platform for Big Data AI

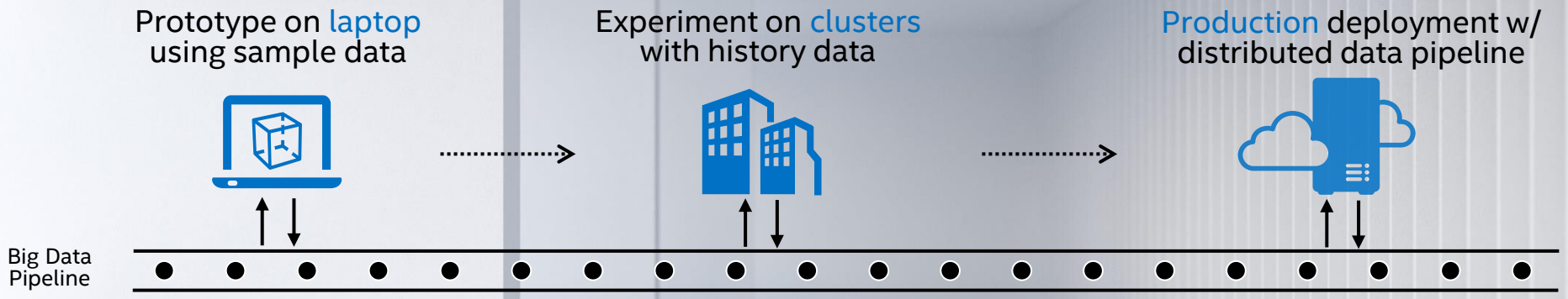


https://github.com/intel-analytics/analytics-zoo

intel.

End-to-End Big Data Analytics and AI

Seamless Scaling from Laptop to Distributed Big Data



- Easily prototype end-to-end pipelines that apply AI models to big data
- "Zero" code change from laptop to distributed cluster
- Seamlessly deployed on production Hadoop/K8s clusters
- Automate the process of applying machine learning to big data

intel.

Analytics Zoo: Open Source Platform for Big Data AI

Scaling End-to-End AI to Distributed Big Data

PPML	Privacy Preserving Data Analytics & ML on SGX
Zouwu	Scalable time series analysis pipeline w/ AutoML
RayOnSpark	Run Ray programs directly on Big Data platform
Cluster Serving	Distributed real-time model serving on Flink
Orca	Seamlessly scale out TF & PyTorch on Spark & Ray

Laptop | K8s Cluster | Hadoop Cluster | Cloud

DL Frameworks (TF/PyTorch/BigDL/OpenVINO/...) | Distributed Analytics (Spark/Flink/Ray/...) | Python Libraries (Numpy/Pandas/sklearn/...)

Powered by oneAPI

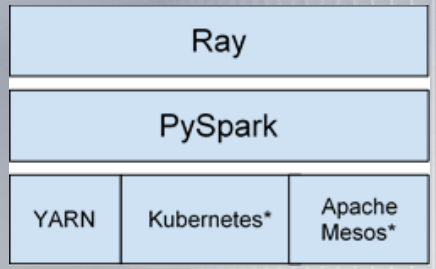
https://github.com/intel-analytics/analytics-zoo

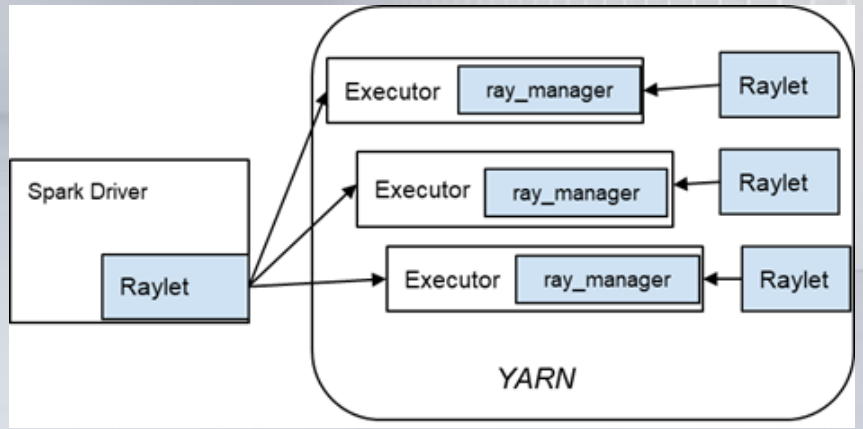
intel.

RayOnSpark

Run Ray Programs Directly on Big Data Platform

- RAY : **distributed framework for emerging AI applications**
- **RayOnSpark**
  - Directly run Ray programs on Big Data cluster
  - Integrate Ray programs into Spark data pipeline





https://medium.com/riselab/rayonspark-running-emerging-ai-applications-on-big-data-clusters-with-ray-and-analytics-zoo-923e0136ed6a

intel.

RayOnSpark

Run Ray Programs Directly on Big Data Platform

```
from zoo.orca import init_orca_context, stop_orca_context
init_orca_context(cluster_mode="yarn", cores=4, memory="10g",
                  num_nodes=2, init_ray_on_spark=True)

#Ray code
@ray.remote
class TestRay():
    def hostname(self):
        import socket
        return socket.gethostname()

actors = [TestRay.remote() for i in range(0, 100)]
print([ray.get(actor.hostname.remote()) for actor in actors])

stop_orca_context()
```

https://medium.com/riselab/rayonspark-running-emerging-ai-applications-on-big-data-clusters-with-ray-and-analytics-zoo-923e0136ed6a

intel.

Agenda

- Why Reinforcement Learning Recommenders?
- What Analytics Zoo and RayOnSpark?
- **How to build reinforcement learning recommenders using RayOnSpark**
- Summary

intel.

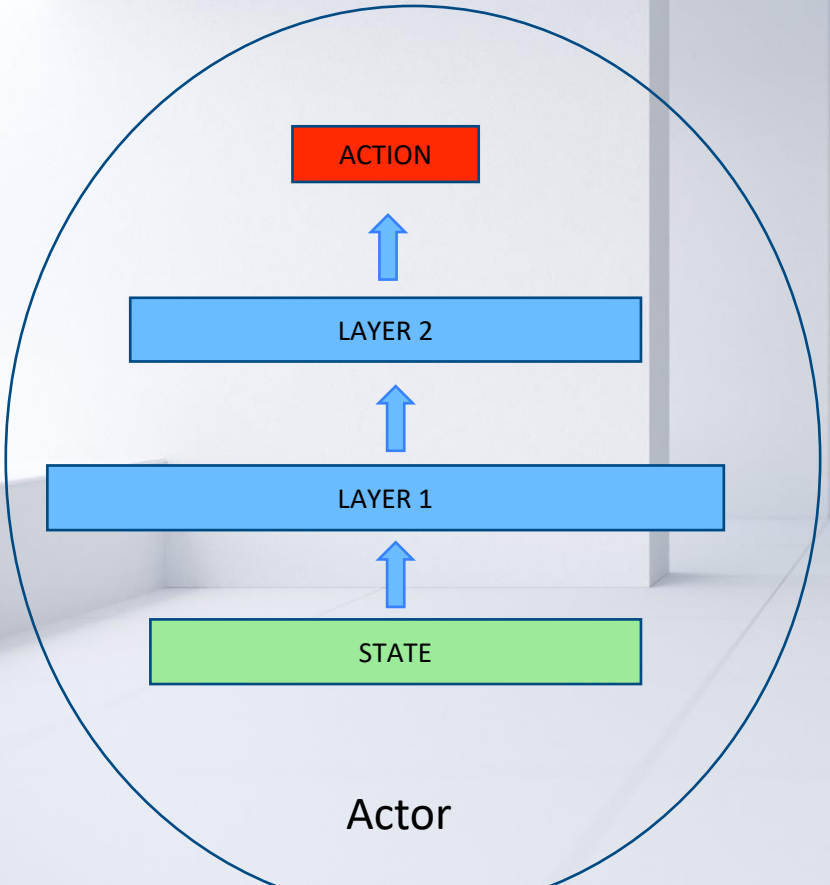
Build MovieEnv

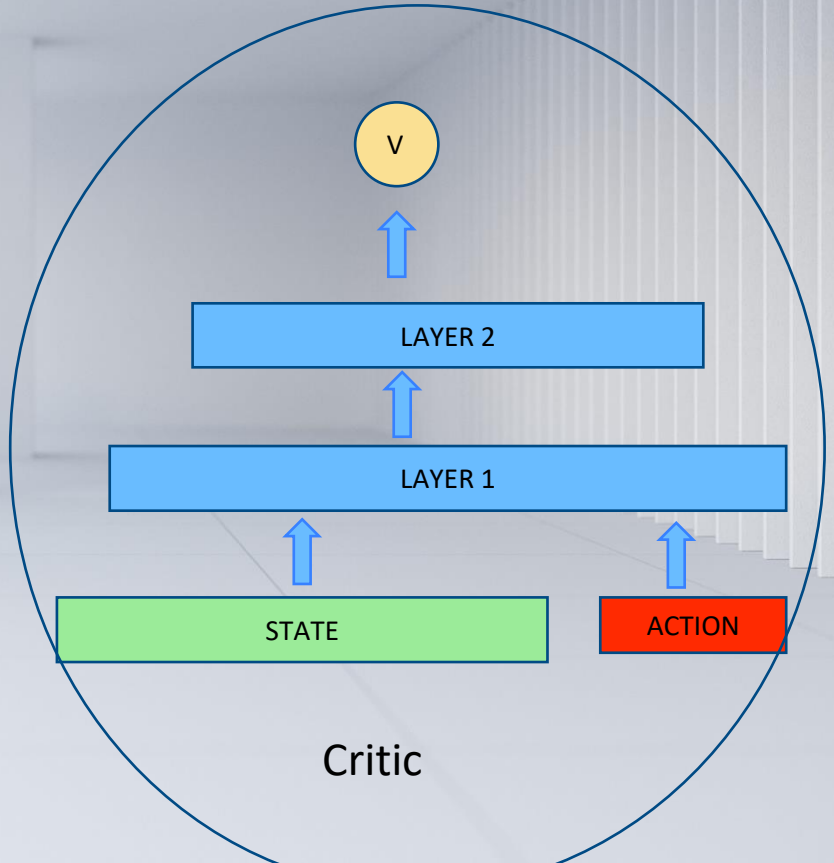
- Environment: MovieEnv(gym.Env)
- Agent: recommender
- Observation/States:
  - User embeddings, movie history embeddings
- Action: movie recommended
- Reward: rate + diversity
- Episode: user session

  20 steps

```
class MovieEnv(Env):
    def __init__(self, config):

    def step(self, action):
        ...
        return obs, reward, done, self.info

    def _get_reward(self, action):

        similarity = spatial.distance.cosine(m_action, m_vecs) if mid in
self.movies.keys()  else 0
        reward = rate + (1-similarity)
```

A2C


ACTION
LAYER 2
LAYER 1
STATE
Actor


V
LAYER 2
LAYER 1
STATE
ACTION
Critic

intel.

Trainer with RayOnSpark

```
import ray.rllib.agents.a3c.a2c as a2c

init_orca_context(cores=8, memory="10g",
                  num_nodes=2, init_ray_on_spark=True)

env_conf= EnvConfig()
trainer_conf = a2c.A2C_DEFAULT_CONFIG.copy()
trainer_conf["env_config"] = env_conf._values
trainer_conf["env"] = MovieEnv
trainer = a2c.A2CTrainer(config=trainer_conf)
for i in range(1, 1000):
    result = trainer.train()
    if i % 50 == 0:
        checkpoint = trainer.save()

stop_orca_context()
```

intel.

Recommend items

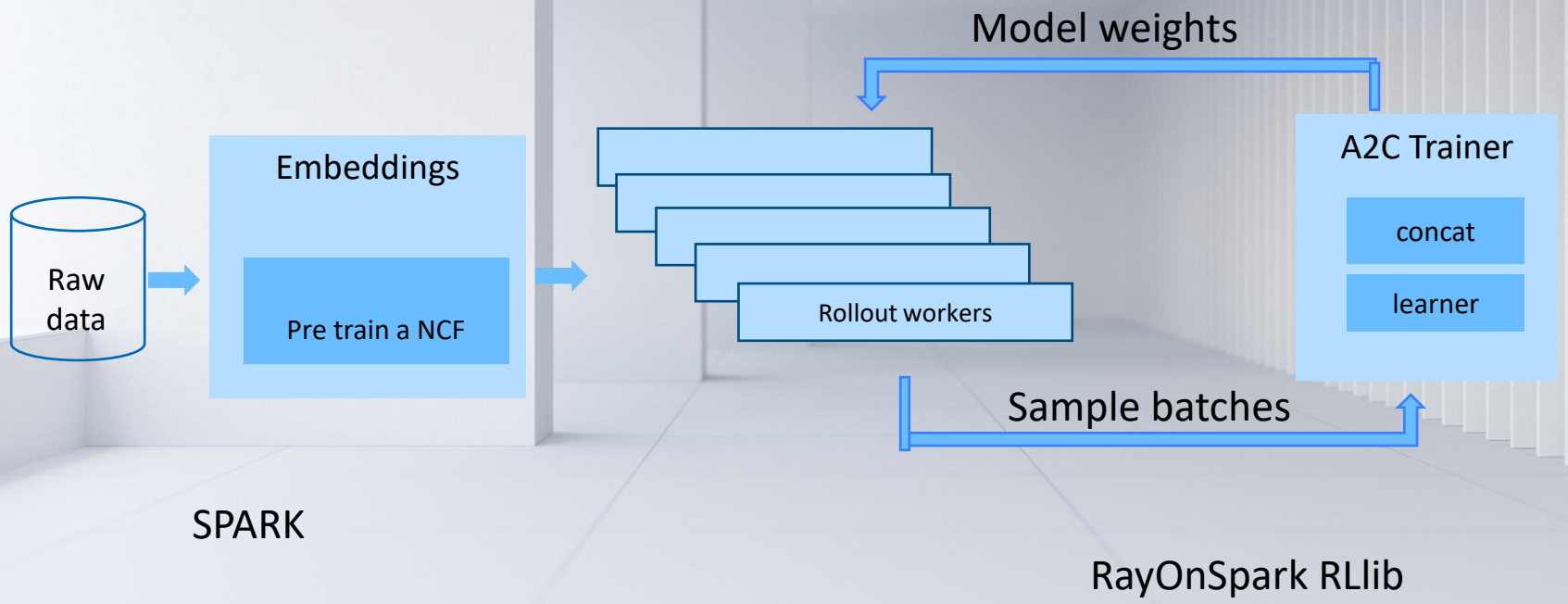
```
trainer.restore(path_to_checkpoint)
for i in range(100):
    obs = env.reset()
    while not done:
        action = trainer.compute_action(obs)
        obs, reward, done, info = env.step(action)
        episode_reward += reward

        print("episode_reward", episode_reward)
```

https://medium.com/riselab/rayonspark-running-emerging-ai-applications-on-big-data-clusters-with-ray-and-analytics-zoo-923e0136ed6a

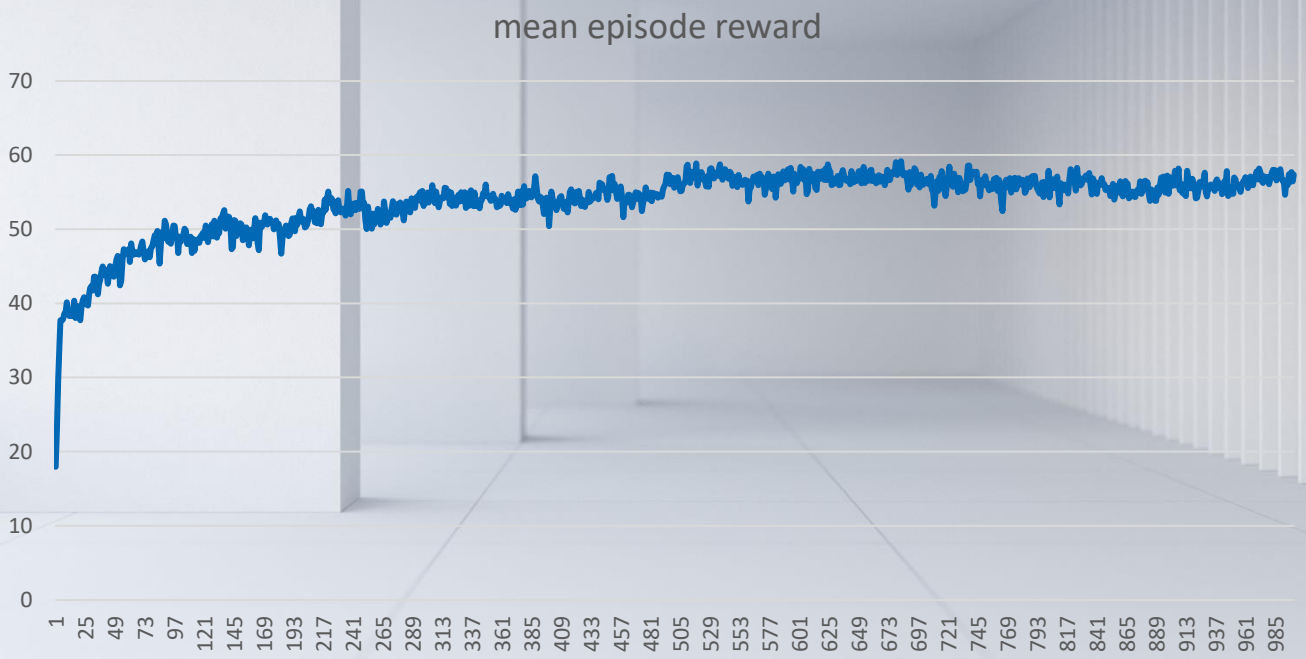
Analytics Zoo API in blue

intel.

Training with RayOnSpark



Initial Results



Agenda

- Why Reinforcement Learning Recommenders?
- Why Analytics Zoo and RayOnSpark?
- How to build reinforcement learning recommenders using RayOnSpark
- Summary

intel.

Summary

Built RL recommenders using RayOnSpark

- RL Recommenders capture dynamic nature of user-item interaction and focus on both immediate and long term reward
- Could maintain recommendation accuracy and diversity by adding similarity in reward
- Train different model components separately substantially reduces training time

Open Source Website

- Project repo: https://github.com/intel-analytics/analytics-zoo
- Use cases: https://analytics-zoo.readthedocs.io/en/latest/doc/Application/powered-by.html

intel.

Thank You

intel.

intel.

Notices & Disclaimers

- Performance varies by use, configuration and other factors. Learn more at www.intel.com/performanceIndex
- Performance may vary based on the specific game title and server configuration. To reference the full list of Intel Server GPU platform measurements, please refer to http://www.intel.com/content/www/us/en/benchmarks/server/graphics/IntelServerGPU
- All product plans and roadmaps are subject to change without notice.
- Intel technologies may require enabled hardware, software or service activation.
- No product or component can be absolutely secure.
- Your costs and results may vary.
- Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.
- All product plans and roadmaps are subject to change without notice.
- © Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others. Intel Server GPU TCO analysis is based on internal Intel research. Pricing as of 10/01/2020. Analysis assumes standard serving pricing, GPU list pricing, and software pricing based on estimated Nvidia software license costs of $1 per year for 5 years.
- Intel Server GPU Performance may vary based on the specific game title and server configuration. To reference the full list of Intel Server GPU platform measurements, please refer to http://www.intel.com/content/www/us/en/benchmarks/server/graphics/IntelServerGPU
- Video game footage courtesy of Tencent Games and Gamestream.
- LEGO STAR WARS TITLES : © Lucasfilm Entertainment Company Ltd. or Lucasfilm Ltd. & ® or TM as indicated. All rights reserved.
- LEGO, the LEGO logo and the Minifigure are trademarks of The LEGO Group. © The LEGO Group. All rights reserved.
- “DiRT4”™ : © 2017 The Codemasters Software Company Limited ("Codemasters"). All rights reserved. "Codemasters"®, “EGO”®, the Codemasters logo, and “DiRT”® are registered trademarks owned by Codemasters. “DiRT4”™ and “RaceNet”™ are trademarks of Codemasters. All rights reserved. Under licence from International Management Group (UK) Limited. All other copyrights or trademarks are the property of their respective owners and are being used under license. Developed by Codemasters.