ATAI

Federated Transfer Reinforcement Learning

Mikel de la Fuente
Maider Amuchastegui
Jesús Rugarcía

CONTENTS

1. Introduction

In this report, we are going to present what Federated Transfer Reinforcement Learning is, to understand better that, we will focus on explaining what Reinforcement Learning is, how Transfer Learning works and on what Federated Learning is about.
So in the next report not only will we explain these three Machine Learning(ML) models, but also how working together can be useful for autonomous driving, putting them into practice by an example.

First of all, let's introduce what autonomous driving is, it normally refers to self-driving vehicles or transport systems that move without the intervention of a human driver.

In autonomous driving Reinforcement Learning(RL) is commonly used, but its' sequential process of training RL models, makes it a really time-consuming process as well as the fact that RL models stay local and offline, they cannot be re-used or work collaboratively. In order to solve this problem, an online federated RL transfer process may be the solution: despite the environment the participant agents make actions with the knowledge they learn from other agents.

The aforementioned example consists of a real-life collision avoidance system ,which cooperatively learns from scratch to avoid collisions in an indoor environment with obstacle objects. This proposed framework showed a 27% increase in the average distance with obstacles and 42% decrease in the collision counts.

In terms of hardware, LIDAR sensors are used, and digital simulators are used in order to not cause damage in the training process.

DRL, Distributed Reinforcement Learning, has shown a performance improvement by employing distributed architecture for decentralized agents. Most DRL frameworks consider only synchronous learning with a constant environment. It is now common to perform pre-training on simulators, and then transfer the pre-trained model to real-life. This has to be done offline, which may be very time-consuming, and there is lack of feedback and collaborations from the model trained with different real-life scenarios.

To overcome these challenges, an end-to-end training process is proposed which leverages federated learning (FL) and transfer learning to enable asynchronous learning of agents from different environments simultaneously. Specifically, we bridge the pre-training on simulators and real-life fine tuning processes by various agents with asynchronous updating strategies. The proposed framework alleviates the time-consuming offline model transfer process in autonomous driving simulations while allowing heavy loads of training data to stay local in the autonomous edge vehicles. Therefore the framework can be potentially applied to real-life scenarios where multiple

self-driving technology companies collaborate to train more powerful RL tasks by pooling their robotic car resources without revealing raw data information.

2. Reinforcement learning (RL)

Reinforcement Learning is a paradigm of Machine Learning based on the trial and error principle. It is heavily influenced by psychological research, as it uses its own interpretation of rewards and penalizations concepts used in behavioural investigation. Due to the generality of tasks and approaches that can involve RL, it is often thought to be a type of problem rather than a set of techniques.

RL agents learn to take a sequence of decisions autonomously in a dynamic and potentially complex environment without the need for a programmer to establish a specific logic. Instead, the programmer defines a logic of rewards and punishments that affect the validity of a certain behaviour or response for a given task. As in any ML problem, its objective is to maximize rewards or precision for the problem to solve.

In a standard RL model the agent is connected to its environment through perception and action. Implementations often start with a random test and evolve into a complex and effective solution as training progresses. On each training step, the agent receives indications of the state of the environment to generate an action as output. It's behavior should try to take actions that increase the values of the reward over time, following a trial and error approach, creating a map of states and actions.
Formally, the model consists of:

- a discrete set of environment states, S ;
- a discrete set of agent actions, A; and
- a set of scalar reinforcement signals; typically f0; 1g, or the real numbers.

It differs from other ML paradigms as it does not use input/pairs as knowledge for training, and therefore it does not suggest to the agent which is the action that it should take, relying on the reward instead for the agent to gather experience actively. It is more similar to other AI search techniques in that it does require to predefine a set of possible states of the environment, but it does not require to predefine a model of state transitions, and assumes that the entire state space can be stored into memory.

Often, RL consists of a mixture between simulation learning followed by real world training. It is a useful paradigm for resolving complex long term problems within complex environments. However, it also has some problems. It offers solutions to specific problems and may have problems with generalization, and will often need to re-train models to adapt to small changes in the environment if not present on the state model. It also assumes the world to be markovian, which is not.

If we focus on the relationship of reinforced learning and the implementation of autonomous driving, we can say that the RL is widely used in autonomous driving tasks. Training RL models typically involves a multi-step process: pre-training RL models on simulators, uploading the pre-trained model to real-life robots, and fine-tuning the weight parameters on robot vehicles. However, RL applied to autonomous driving often has the following drawbacks:

- Time consuming training.
- Local knowledge, difficult to apply in similar problems.
- Lack of feedback and collaboration from real life applications to models.

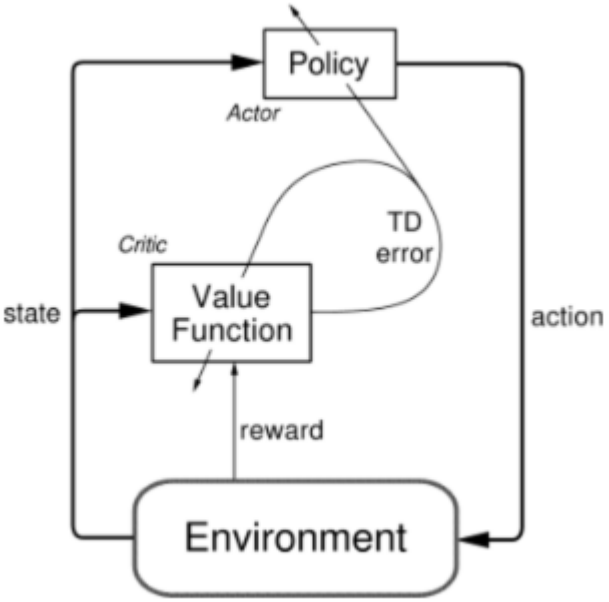
RL in autonomous driving with DDPG

Autonomous driving problems all share one fundamental characteristic, the complexity of a changing environment such as a motorway, which can have dynamic obstacles like pedestrians, other vehicles or road obstacles, such as fallen trees or gaps. Therefore, Reinforcement learning algorithms can be a good strategy to tackle this problem.

One such algorithm is Deep Deterministic Policy Gradient (DDPG). High sensitivity hardware sensors are utilized to collect precise data about the environment, such as LIDAR. This data can then be utilized to feed the algorithm the current state of the environment.

DDPG mixes deterministic policy gradient algorithms with Q-networks. This means that DDPG maps actions to environment states in a deterministic way instead of a stochastic one. In other words, this algorithm does not work with continuous values of the environment perception signals, but maps states using discrete measurements, thus reducing the data processed.

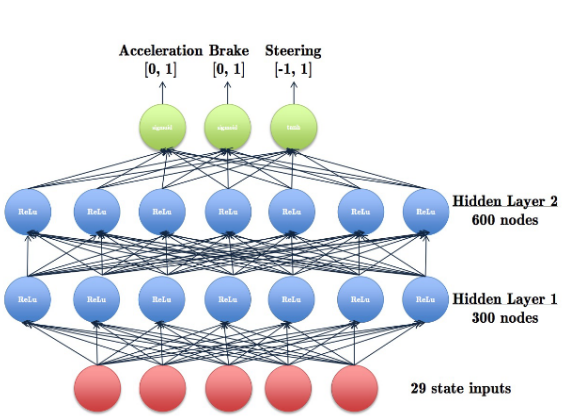
There are two main elements. The action policy selects action for the agent to take based on the current environment state, while the critic uses a Q-value function to assess its performance.



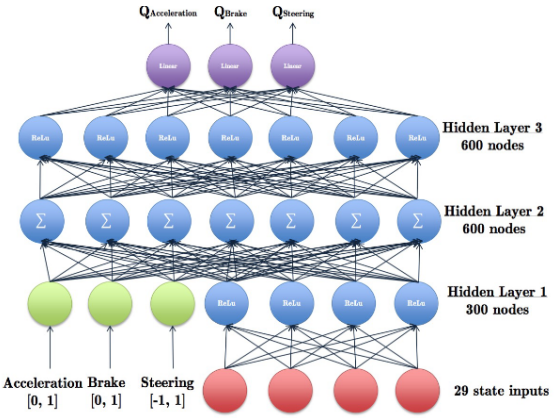
Actions have three dimensions :

- Acceleration, valued from 0(no acceleration) to 1(full acceleration).
- Brake, value from 0(no brake) to 1(full brake).
- Steering, valued from -1 (max steering to the left).

The Q-function gives a reward value for each of the dimensions of the taken action, thus telling the actor exactly which side of it's behaviour to reinforce.



(a) Actor Network



(b) Critic Network

3. Transfer Learning (TL)

Transfer learning (TL) is a research problem in machine learning (ML). It focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. For example, in training a classifier to predict whether an image contains food, you could use the knowledge it gained during training to recognize drinks.

The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the source task. There are three common measures by which transfer might improve learning. First is the initial performance achievable in the target task using only the transferred knowledge, before any further learning is done, compared to the initial performance of an ignorant agent. Second is the amount of time it takes to fully learn the target task given the transferred knowledge compared to the amount of time to learn it from scratch. Third is the final performance level achievable in the target task compared to the final level without transfer.

In reinforcement learning (RL) problems, inclined agents take sequential actions with the goal of maximizing a reward signal, which can be delayed in time. However, when RL agents begin to learn a clean sweep, mastering difficult tasks is often time-consuming or infeasible, and therefore a significant amount of current RL research is focused on improving the speed of learning by exploiting experience in the domain with different amounts of knowledge provided by humans.

Learning transfer in RL is an important topic to address at this time for three reasons. First, in recent years, RL techniques have achieved remarkable successes in difficult tasks that other machine learning techniques cannot or are ill equipped to tackle. Second, classic machine learning techniques, such as rule induction and classification, are mature enough that they can now be easily leveraged to help with TL. Third, promising initial results show that these transfer methods are not only possible, but can also be very effective in accelerating learning.

4. Federate Learning (FL)

Federated learning is a machine learning technique that trains an algorithm across multiple decentralized edge devices or servers holding local data samples, without exchanging them. This approach stands in contrast to traditional centralized machine learning techniques where all the local datasets are uploaded to one server, as well as to more classical decentralized approaches which often assume that local data samples are identically distributed. For example, mobile phones collectively study a shared prediction model, while keeping the device's training data local instead of uploading and storing it.

Federated learning involves training statistical models over remote devices or siloed data centers, such as mobile phones or hospitals, while keeping data localized. Training in heterogeneous and potentially massive networks introduces novel challenges that require a fundamental departure from standard approaches for large-scale machine learning, distributed optimization, and privacy-preserving data analysis.

Federated learning is also used in self-driving cars. Self-driving cars encapsulate many machine learning technologies to function: computer vision for analyzing obstacles, machine learning for adapting their pace to the environment (e.g., bumpiness of the road). Due to the potential high number of self-driving cars and the need for them to quickly respond to real world situations, traditional cloud approach may generate safety risks. Federated learning can represent a solution for limiting volume of data transfer and accelerating learning processes.

5. Federated Transfer Reinforcement Learning

The algorithms explained above can be used together to further augment the benefits of the features they provide. One example of the usage of the algorithm in conjunction is the implementation of Autonomous driving algorithms.

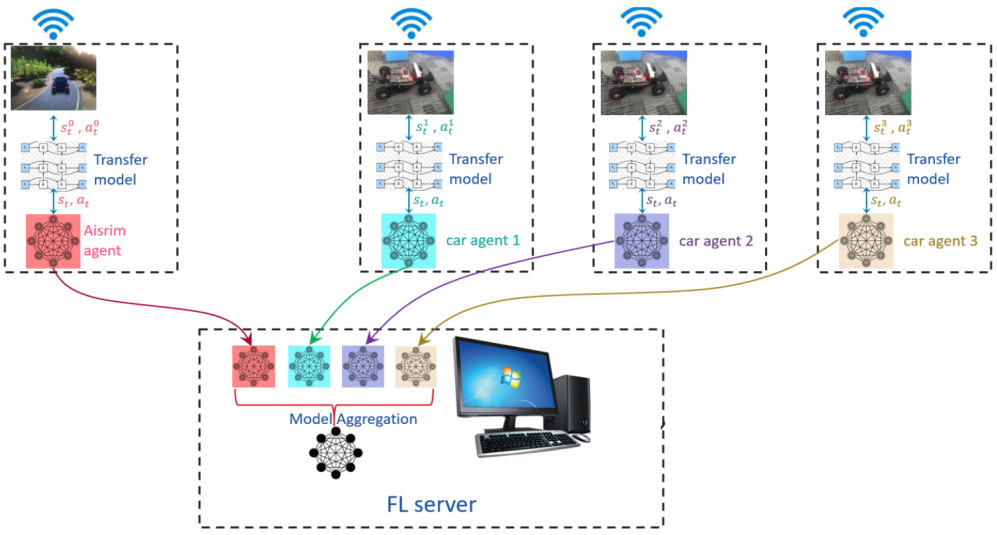
In this example, researchers designed a framework to operate with the three models. The framework is capable of transferring the knowledge obtained through different implementations of Reinforcement Learning agents in real time on the foundation of Federated Learning.

FTRL Framework in autonomous driving

The FTRL framework is used for collision avoidance. FTRL framework is not designed for any specific RL method. However, in order to thoroughly describe the framework and validate its effectiveness, Deep Deterministic Policy Gradient (DDPG) is chosen to be the RL implementation.

The model has RL agents that interact with an stochastic environment in discrete time. At each time step, each agent makes observations, takes actions and receives rewards, as in standard RL models. Actions are determined by a policy that maps states to actions. The reward policy is based on data collected from a LIDAR sensor, in which actions that result in collisions and get the smallest distance to obstacles obtain greater rewards.

All agents share the same model structure, including both those acting in simulated environments and those acting in real life environments. They communicate with the FL server through a wireless router, sharing their experience for it to be aggregated to generate the FL model. Then this model gets asynchronously updated to all the agents.



The basic training process is as follows:

1. Online transfer process. Since distributed RL agents are acting in various environments, a knowledge transfer process is needed when each RL agent interacts with its specific environment.
2. Single RL agent training and inference. This process serves as a standard RL agent training and inference process.
3. FedAvg process. All the useful knowledge of distributed RL agents is aggregated by FedAvg process of the RL models.

Compared to previous models that do not include the three ML paradigms, FTRL obtains significantly better results and takes less time to train.

Training DDPG algorithm from scratch on real-life autonomous cars may take too much time. But with a common DDPG model pre-trained, each car can make reasonable action corresponding to the LIDAR data, which however it can still improve. In this fine-tune processes of any real-life agent, the training time of each DDPG agent is divided into three stages, with each containing 2500 discrete time steps. The inference of the pre-trained model happens when the number of the experience buffer is smaller than 2500. Each time step takes 0.25 seconds, and stage I and stage II have range [625, 1250) and [1250, 1875) seconds, respectively. Since all cars may be running in different environments, the rewards may be of different scales.

In an experiment, researchers kept track of the cumulative summation of relative performance for different stages, and presented the results of different application settings, including DDPG results on single RC cars, FTRL-DDPG results with the federation of three RC cars (FTRL-DDPG) and FTRL-DDPG results with the federation of three RC cars and Airsim platform (FTRL-DDPG-SIM).

In the table below are represented the average LIDAR distances and collision number results of three cars on the test experiments. For each approach on each car, 50 cycles in the race are executed.

	car1		**car2**		**car3**	
	avg_dist	coll_no	avg_dist	coll_no	avg_dist	coll_no
DDPG	0.39	18	0.29	31	0.38	24
FTRL-DDPG	0.42 (7.7%)	**9** (50%)	0.37 (27.6%)	27 (12.9%)	**0.51** (34.2%)	17 (29.2%)
FTRL-DDPG-SIM	**0.45** (15.4%)	12 (33.3%)	**0.39** (34.5%)	**16** (48.4%)	0.50 (31.6%)	**13** (45.8%)

The better result for each car it's on bold denote. It can be seen that a better policy is capable of fulfilling collision avoidance tasks with greater average distance and less collision number.

It can be easily seen that for each car, the average distance and collision numbers of FTRL-DDPG and FTRL-DDPG-SIM are much less than the corresponding results of DDPG, which demonstrate the effectiveness of FTRL-DDPG-SIM.

6. Conclusions

Reinforcement learning, Transfer learning and Federated learning are well known in ML investigation, each providing its own set of benefits. However, the integration of the three of them yields better results, as proved in with the FTRL framework. However, more investigation is necessary to further develop the field of this promising integration.

If we focus on the differents application settings that we mentioned before, used on autonomous driving areas, with the capabilities of transferring online knowledge from simulators to real-life cars, FTRL-DDPG-SIM performs better than both single execution of single RL agents and federation model with identical RL agents with better training speed and performance.

7. References

- Federated Transfer Reinforcement Learning for Autonomous Driving
  Authors: Xinle Liang, Yang Liu, Tianjian Chen, Ming Liu, Qiang Yang.

- Deep Reinforcement Learning framework for Autonomous Driving
  Authors: Sallab, Ahmad EL; Abdou, Mohammed; Perot, Etienne; Yogamani, Senthil.

- Transfer Learning for Reinforcement Learning Domains: A Survey
  Authors: Mathew E. Taylor, Peter Stone

- What is reinforcement learning? The complete guide
  Web: Deepsense.ai

- Federated Learning: Challenges, Methods, and Future Directions
  Web: ieeexplore.ieee.org

- Transfer learning
  Web: wikipedia.org

- Federated learning
  Web: wikipedia.org

- Emergent tool use from multi-agent autocurricula
  Authors: Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, Igor Mordatch.

- Reinforcement Learning: A survey
  Authors: Leslie Pack Kaelbling, Michael L. Littman, Andrew W. Moore.

- Autonomous Driving: An Overview
  Web: zf.com

- Transfer Learning
  Authors: Lisa Torrey, Jude Shavlik.

- 5 Applications of Federated Learning
  Web: hitechnectar.com

- Deep Reinforcement Learning for Autonomous Driving
  Authors: Sen Wang, Daoyuan Jia, Xinshuo Weng.