Проектна задача по предметот

Агентно базирани системи

Тема:

Parallel Q Network агент во околината CartPole

Бојан Василевски 211165

Содржина

Апстракт

Во овој труд е претставен понов алгоритам за учење со поттикнување, наречен Parallel Q Network (PQN), кој се разликува од стандардните алгоритми преку користење на повеќе паралелни инстанци на средини за тренирање. Со елиминирање на реплеј баферот и таргет мрежите, како и со воведување на нормализација на слоевите, PQN овозможува побрзо и поефикасно учење. Целта на ова истражување е анализа на примената на PQN во околината CartPole-v1, со проценка на неговата ефикасност и стабилност во споредба со традиционалните методи.

Вовед

Учењето со поттикнување (Reinforcement Learning) е област на машинското учење која се фокусира на тоа како агенти можат да учат оптимални политики преку интеракција со средината. Овој тип на учење се применува во различни области како што се роботика, игри, автономни возила и многу други. Стандардните алгоритми за учење со поттикнување, како што е Deep Q-Network (DQN), комбинираат Q-учење со длабоки невронски мрежи, што овозможува справување со задачи со големи просторни состојби. Сепак, овие алгоритми се соочуваат со предизвици во однос на ефикасноста и стабилноста на учењето, особено во посложени средини.

Паралелните методи во учењето со поттикнување, како што е Parallel Q Network (PQN), се предлог за надминување на овие ограничувања. PQN користи повеќе паралелни инстанци на средини за да ја зголеми брзината на собирање податоци и да го забрза процесот на учење. Освен тоа, оваа техника ја елиминира потребата од реплеј баферот и таргет мрежите, а воведувањето на нормализација на слоевите помага во подобрувањето на стабилноста и ефикасноста на учењето.

За експериментите во ова истражување е користена средината **CartPole-v1**, која е популарна во теоретски и практични истражувања на учењето со поттикнување. CartPole е средина која се состои од количка на кој е прикачен столб. Целта на средината е што подолго количката да го балансира столбот во вертикална позиција преку нејзино движење.

Целта на задачата е да се максимизира времетраењето каде што столбот останува исправен. При секое успешно извршување на акција, агентот добива позитивна награда (+1). Ако столбот падне (излегува од предодредената зона или прави прекумерна аголна промена), играта завршува и агентот добива негативен резултат. Поради ова, околината се состои од балансирање на столбот што резултира со награди што треба да се максимизираат преку време.

Сродни истражувања

Бидејќи алгоритамот PQN е приличнно нов алгоритам не постојат многу сродни истражувања за овој алгоритам. Сепак, постојат неколку слични истражувања кои се блиски во контекст на подобрување на ефикасноста и стабилноста на веќе постоечките алгоритми за учење со поттикнување, користејќи паралелизација, елиминирање на реплеј баферот или оптимизација на процесот на учење.

Едно од најблиските истражувања е **Parallel Q Network (PQN) - CleanRL**, кое ја опишува паралелизацијата на Q-учењето и употребата на повеќе агенти кои паралелно учат од различни инстанци на средина. Овој труд покажува како PQN се разликува од традиционалните DQN и други алгоритми со својата брза адаптација на новите податоци и ефективноста во сложени задачи.

Други слични истражувања кои се блиски до PQN, но се фокусираат на различни аспекти на паралелизација и оптимизација на Q-учењето се:

1. Balancing a CartPole System with Reinforcement Learning - Овој труд се фокусира на традиционални методи за учење со поттикнување како што се Q-learning и Deep Q Networks (DQN), но не вклучува паралелизација. Сепак, истиот контекст на примената на околината CartPole овозможува споредба со PQN.
2. A Novel Update Mechanism for Q-Networks Based On Extreme Learning Machines - Овој труд се бави со нов механизам за ажурирање на Q-мрежи, што е релевантно за PQN, бидејќи се фокусира на оптимизација и ефикасност на Q-учењето, иако не користи паралелизација на истиот начин како PQN.

Иако ова поле сè уште е во раниот стадиум на развој, постојат многу истражувања кои се поврзани со алгоритми за паралелно учење и оптимизација на класичните Q-алгоритми, кои можат да бидат корисни за понатамошни анализа и размена на идеи со PQN.

Опис на агентот, методите, експериментални резултати

Во ова истражување, ќе се користи **PQN агент** (Parallel Q Network) од библиотеката **CleanRL** за да се испита неговата ефикасност при извршување на околината **CartPole-v1**. PQN се карактеризира со користење на паралелни инстанци на средини за учење, што го забрзува процесот на учење и го прави поефикасен во однос на традиционалните Q-алгоритми. Овој агент се обучува користејќи неколку паралелни инстанци за да се соберат податоци и да се изврши ажурирање на мрежата без потреба од реплеј бафер или таргет мрежи, што го прави процесот постабилен и побрз.

Методи

1. **Алгоритам:** PQN користи повеќе паралелни агенти кои истовремено учат од различни инстанци на средината. На секоја паралелна инстанца, агентот е одговорен за извршување на акција базирана на својата стратегија и собирање на награди.
2. **Нормализација:** Нормализацијата на слоевите е воведена за да се подобри стабилноста на мрежата и да се забрза процесот на учење. Овој метод помага во контрола на вредностите на активностите во слоевите на мрежата, што овозможува побрзо и поефикасно учење.
3. **Ажурирање на Q-вредности:** PQN користи техники за брзо ажурирање на Q-вредностите преку паралелни инстанци на средини, елиминирајќи ја потребата за реплеј бафер.

Експериментални резултати

Во текот на експериментите, следните метрики ќе бидат користени за мерење на перформансите на агентот и за следење на напредокот на учењето:

1. **Episodic Return:** Оваа метрика ја претставува вкупната награда за секоја епизода. Наградите се доделуваат кога столбот останува исправен, а поголема финална награда значи подобар резултат на агентот.
2. **Losses - Q Value:** Оваа метрика покажуваат разликата помеѓу претпоставените Q вредности и актуелните вредности кои ги добива агентот. Поголемите вредности на оваа метрика покажуваат по ефикасно учење и поголема точност на Q вредностите.
3. **Losses - TD Loss:** Овие метрики ја покажуваат разликата помеѓу предвидените и вистинските вредности на повратот (**return values**), базирано на Bellman-овата еднаквост. Минимализирањето на овие губитоци е клучно за подобрување на стабилноста на учењето и постигнување на оптимални политики.

Податочни множества и алатки

- **Средина:** CartPole-v1, која симулира количка и столб.
- **Технологија:** Python, TensorFlow, PyTorch.
- **Алатки:** TensorBoard за визуелизација на логовите и метриките.

Параметри и прилагодувања

Некои од клучните параметри кои се користат за тренингот на агентот вклучуваат:

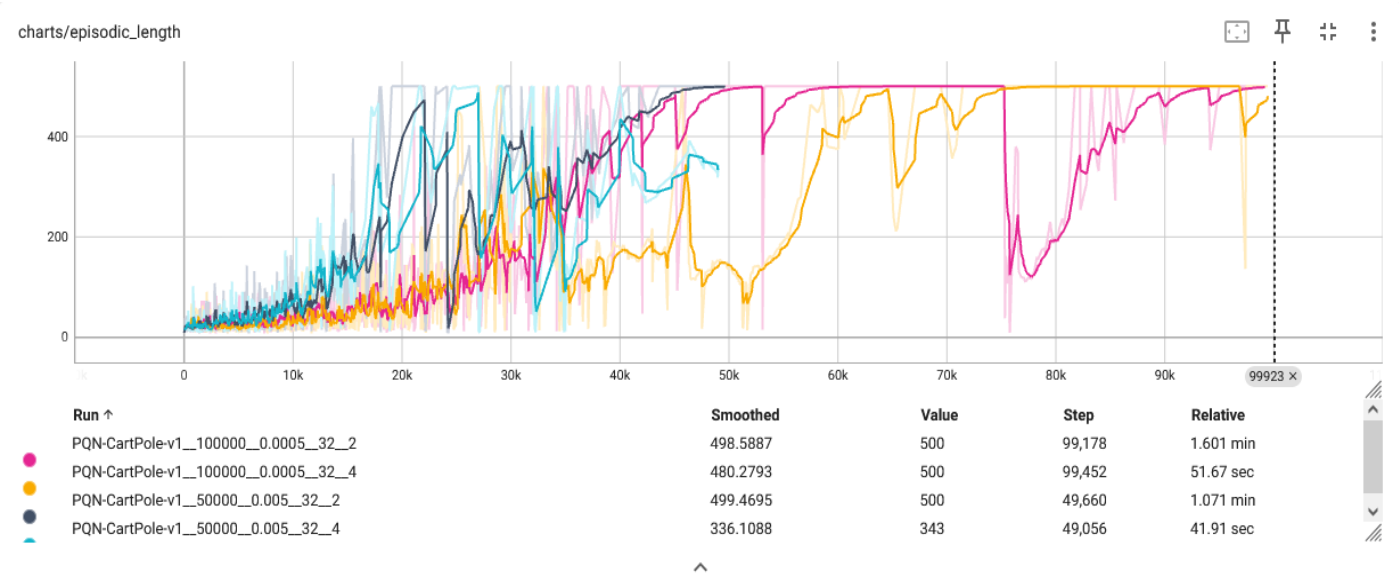
- **Learning Rate:** Одредува колку брзо агентот ќе се прилагодува на средината. Превисока вредност може да доведе до нестабилно учење, додека премала вредност може да забави процесот на учење.

- **Number of Steps:** Бројот на чекори што агентот ги прави пред да ги ажурира своите вредности.
- **Број на паралелни агенти (number of environments):** Бројот на паралелни средини односто агенти што паралелно учат од различни инстанци на средината. Овој параметар има големо влијание на брзината на собирање податоци и на брзината на учењето.
- **Total Timesteps:** колку вкупно чекори ќе трае тренирањето на агентот.

Со користење на овие метрики и параметри, ќе се следи напредокот на агентот и ќе се оцени неговата ефикасност во однос на традиционалните алгоритми за учење со поттикнување.

На следните слики се прикажани наведените метрики за неколку тренирани модели.

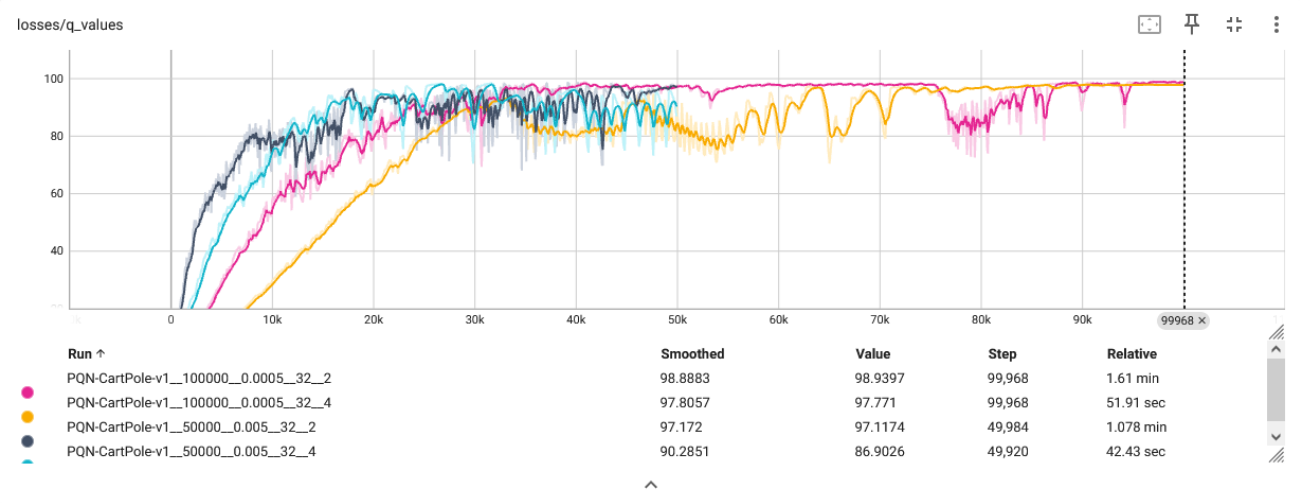
Episodic Return



Run	Smoothed	Value	Step	Relative
PQN-CartPole-v1__100000__0.0005__32__2	498.5887	500	99,178	1.601 min
PQN-CartPole-v1__100000__0.0005__32__4	480.2793	500	99,452	51.67 sec
PQN-CartPole-v1__50000__0.005__32__2	499.4695	500	49,660	1.071 min
PQN-CartPole-v1__50000__0.005__32__4	336.1088	343	49,056	41.91 sec

Како што можеме да видиме, во една од клучните метрики која претставува добивка (награда) по епизода при тренирање на агентот, моделите со поголем број на **timesteps** и помала вредност на **learning rate** конвергираат кој оптимален резултат (500, што е максимумот што може да се добие за оваа околина), додека кај тие со помала вредност на **timesteps** и поголема вредност на **learning rate** гледаме дека агентот со 2 околини добива оптимални резултати додека пак тој со 4 добива послаби резултати.

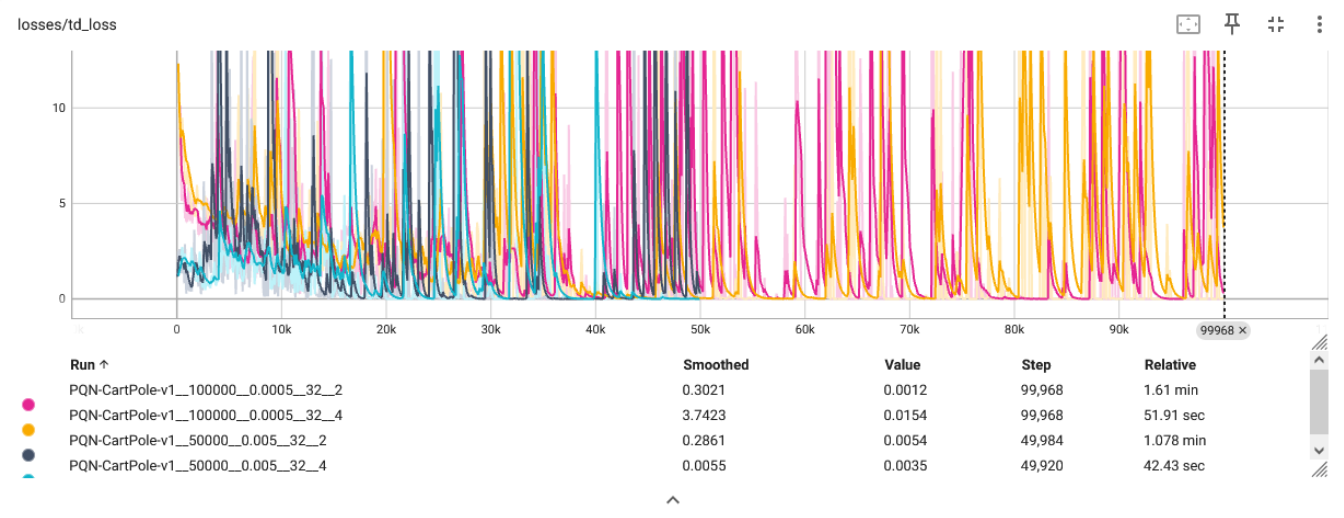
Q Values



Run	Smoothed	Value	Step	Relative
PQN-CartPole-v1__100000__0.0005__32__2	98.8883	98.9397	99,968	1.61 min
PQN-CartPole-v1__100000__0.0005__32__4	97.8057	97.771	99,968	51.91 sec
PQN-CartPole-v1__50000__0.005__32__2	97.172	97.1174	49,984	1.078 min
PQN-CartPole-v1__50000__0.005__32__4	90.2851	86.9026	49,920	42.43 sec

На сликата е претставена друга клучна метрика наречена Q Values. Исто како и претходната метрика, моделите со подолго тренирање добиваат подобри резултати од тие со пократко, додека од тие со пократко моделот со 2 паралелни околини добива подобри резултати од тој со 4. По ова можеме да заклучиме дека за оваа околина по оптимално е да имаме помал број на паралелни околини.

TD Loss



Run	Smoothed	Value	Step	Relative
PQN-CartPole-v1__100000__0.0005__32__2	0.3021	0.0012	99,968	1.61 min
PQN-CartPole-v1__100000__0.0005__32__4	3.7423	0.0154	99,968	51.91 sec
PQN-CartPole-v1__50000__0.005__32__2	0.2861	0.0054	49,984	1.078 min
PQN-CartPole-v1__50000__0.005__32__4	0.0055	0.0035	49,920	42.43 sec

Овде можеме да видиме дека за разлика од претходните метрики агентите со поголем број на **total timesteps** добиваат полоши резултати од другите. Иако ова првично звучи негативно тоа не е секогаш случај. Имено, ова значи дека агентот има сеуште простор за учење, бидејќи претходните метрики беа скоро па оптимални.

Тестирање на моделите по тренирање

```
Model: pqn_CartPole-v1_ts-100000_lr-0.0005_ns-32_ne-2.pth achieved average reward Num episodes: 100, avg reward: 500.0
Model: pqn_CartPole-v1_ts-100000_lr-0.0005_ns-32_ne-4.pth achieved average reward Num episodes: 100, avg reward: 422.46
Model: pqn_CartPole-v1_ts-50000_lr-0.005_ns-32_ne-2.pth achieved average reward Num episodes: 100, avg reward: 496.65
Model: pqn_CartPole-v1_ts-50000_lr-0.005_ns-32_ne-4.pth achieved average reward Num episodes: 100, avg reward: 341.71
```

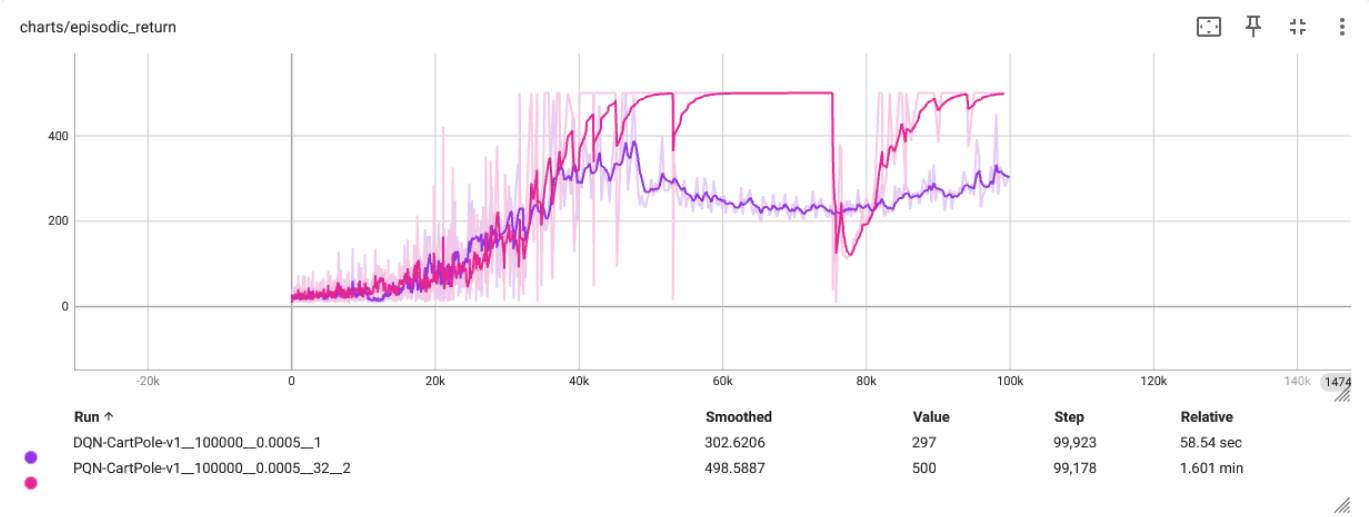
По тренирањето, овие модели беа тестирани во самата околина за да ги видиме нивните резултати по тренирањето. Сите модели беа тестирани во времетраење од 100 епизоди и со лимитрање до добиена награда во вредност од 500 (бидејќи некои модели се толку добро истренирани што добиваа и награда преку 10000 за една епизода). Како и при тренирањето, и при тестирањето беа добиени слични резултати. Најдобар беше истиот модел кој ја имаше максималната можна награда во секоја епизода односно 500. Овој модел користеше 2 паралелни околини за тренирање во времетрање од 100000 **total timesteps**, со вредност од 32 за параметарот **num steps** како и **learning rate** од 0.0005.

По прегледот на овие метрики можеме да заклучиме дека агентот прикажан со розеви линии на графиците добива најдобри резултати. Овој агент ги има следните хиперпараметри:

- **total timesteps** – 100000
- **learning rate** – 0.0005
- **num steps** – 32
- **num environments** – 2

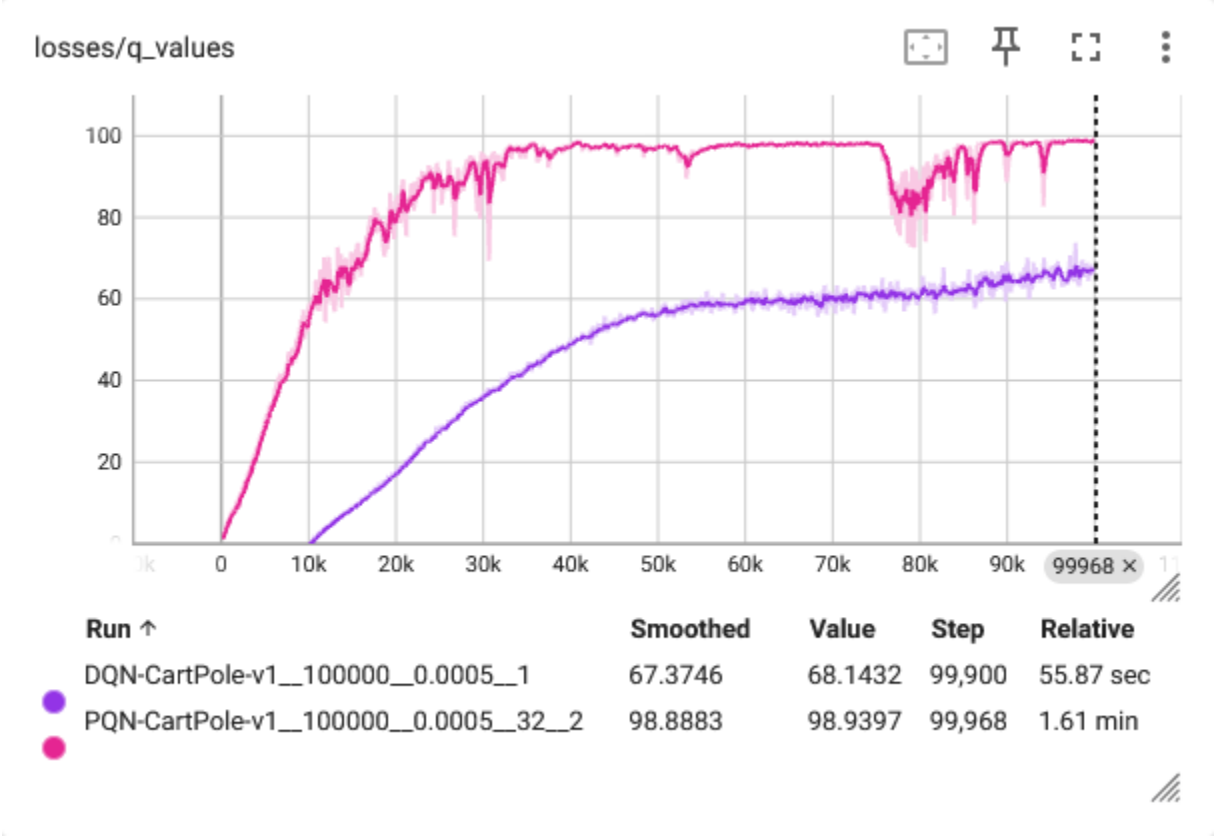
Следно, ќе го споредиме овој агент со друг агент кој е обичен DQN агент и ги користи истите хиперпараметри.

Episodic Return



На сликата можеме да видиме дека **PQN** агентот скоро и конвергира кон оптимална вредност додека **DQN** агентот има многу полоши резултати, односно **PQN** агентот добива подобри резултати за околу 60% во однос на **DQN** агентот.

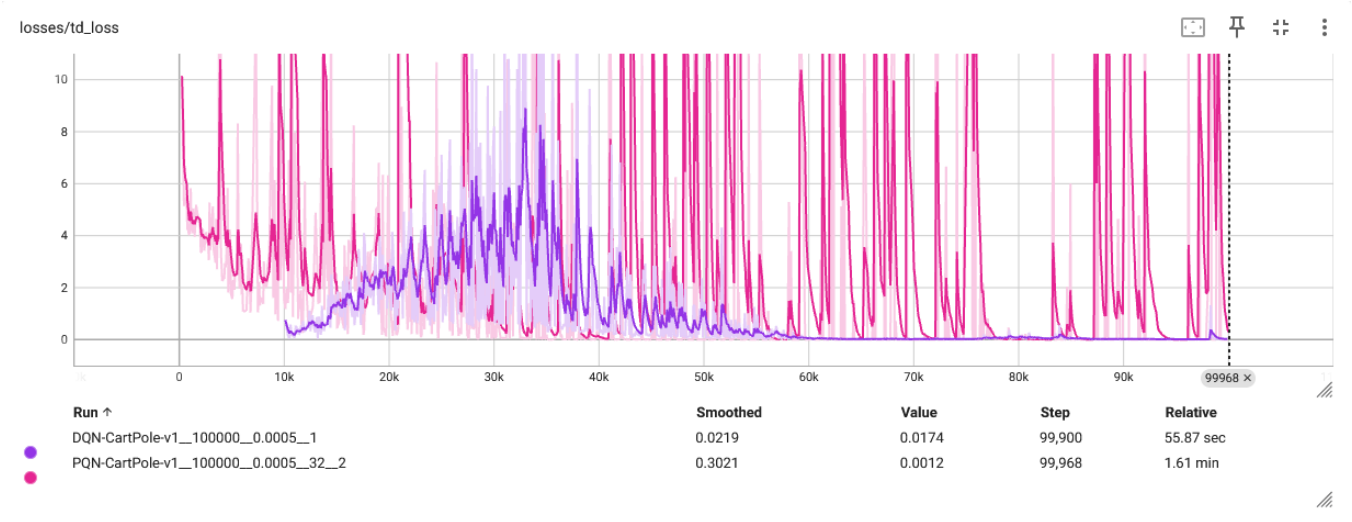
Q Values



Run ↑	Smoothed	Value	Step	Relative
DQN-CartPole-v1__100000__0.0005__1	67.3746	68.1432	99,900	55.87 sec
PQN-CartPole-v1__100000__0.0005__32__2	98.8883	98.9397	99,968	1.61 min

Исто како и претходно, **PQN** агентот е подобар, овојпат за околу 40%.

TD Loss



Run ↑	Smoothed	Value	Step	Relative
DQN-CartPole-v1__100000__0.0005__1	0.0219	0.0174	99,900	55.87 sec
PQN-CartPole-v1__100000__0.0005__32__2	0.3021	0.0012	99,968	1.61 min

Во оваа метрика е подобар **DQN** агентот но како што беше наведено претходно **PQN** агентот добива поголема вредност на губиток поради тоа што тој сеуште има простор за учење и тренирање.

Заклучок

Преку овој труд беше прикажан релативно нов алгоритам сферата на учење со поттикнување наречен **Parallel Q Network (PQN)**. Главната разлика помеѓу овој алгоритам и традиционалните алгоритми како што е Deep Q Network (DQN) е тоа што овде наместо една околина се тренираат повеќе агенти во различни околини во исто време кои комуницираат меѓу себе и разменуваат резултати и искуства што помагаат да се добијат оптимални резултати. Иако овој алгоритам е релативно нов и сеуште не е усовршен, тој сепак постигнува поефикасно тренирање и подобри резултати од традиционалните алгоритми како што е DQN.

Изворен код

https://github.com/BVasilevski/Agent-Based-Systems