# DQN AGENT ZA ATARI 2600 TENNIS

Vladimir Popov
SV29/2021

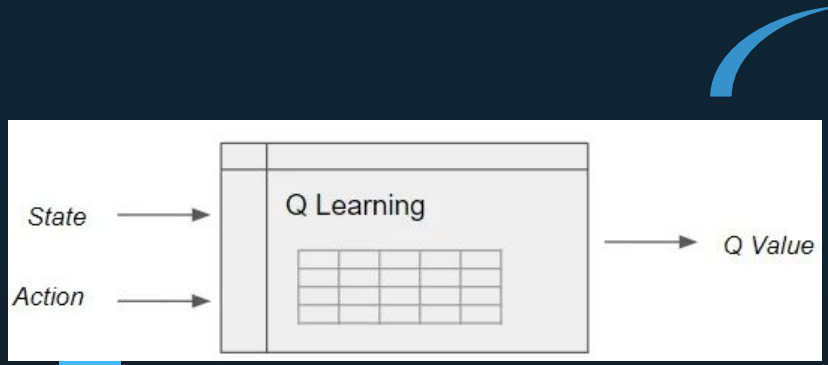
# O IGRI

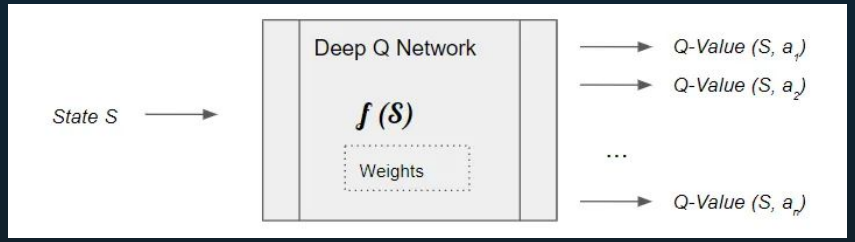
- Teniska pravila
- Igra se jedan set (6 gemova)
- Viewport 210x160x3
- 18 mogućih kontrola
- Okruženje
  - Farama Gymnasium
  - Arcade Learning Environment

| Value | Meaning | Value | Meaning | Value | Meaning |
|---|---|---|---|---|---|
| 0 | NOOP | 1 | FIRE | 2 | UP |
| 3 | RIGHT | 4 | LEFT | 5 | DOWN |
| 6 | UPRIGHT | 7 | UPLEFT | 8 | DOWNRIGHT |
| 9 | DOWNLEFT | 10 | UPFIRE | 11 | RIGHTFIRE |
| 12 | LEFTFIRE | 13 | DOWNFIRE | 14 | UPRIGHTFIRE |
| 15 | UPLEFTFIRE | 16 | DOWNRIGHTFIRE | 17 | DOWNLEFTFIRE |

# O DQN-u

- Metoda za duboko učenje uslovljavanjem
- Aproksimacija Q-vrednosti putem neuronske mreže (tabela -> funkcija -> neuronska mreža)



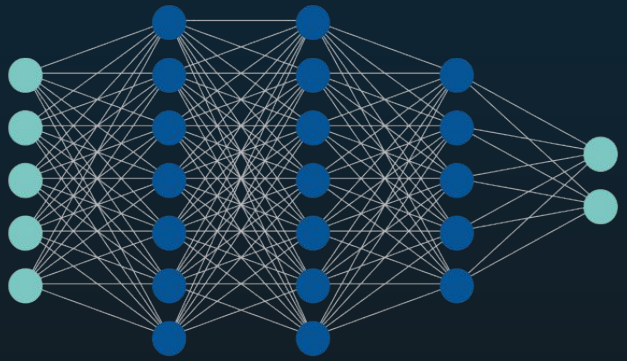



# O DQN-u

- Koraci algoritma
    1. Inicijalizacija replay memorije (kružni bafer, ograničenog kapaciteta)
    2. Inicijalizacija dve neuronske mreže (KNN, policy i target network)
    3. Ponavljamo određeni br. epizoda (do kraja epizode ili određeni broj frejmova)
        4. Za svaki frejm
            a. Izvšavamo izabranu akciju (exploration/exploitation)
            b. Nakon svakog koraka smanjuje se šansa za exploration
            c. Čuvamo (a, s, s‘, r) u replay memoriji
        5. Prilikom treniranja
            a. Uzimamo n nasumičnih “iskustava” iz replay memorije
            b. Na osnovu semplovanih iskustava računa se q-vrednost iz s putem policy mreže
            c. Na osnovu semplovanih iskustava računa se tražena q-vrednost iz s‘ putem target mreže
            d. `q value = reward + discount * expected future reward`
            e. Na osnovu razlike između izračunate q-vrednosti i q-vrednosti iz c. računa se loss nekom od formula
            f. Radi se backpropagation koristeći neki optimizer
        6. Nakon x koraka kopiraj težine iz policy u target mrežu

# ARHITEKTURA MREŽE


210x160x3



Input layer

Hidden layers

Output layer

Q-Value(S, 0)

Q-Value(S, 1)

...

Q-Value(S, 17)

# ARHITEKTURA MREŽE I AGENTA

- 1. Iteracija
  - Previše parametara, dugo se trenira
  - Ideja za mrežu: Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2013). Playing Atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

```
model.add(
    layers.Convolution2D(
        32,
        (8, 8),
        strides=(4, 4),
        activation="relu",
    )
)
model.add(layers.Convolution2D(64, (4, 4), strides=(2, 2), activation="relu"))
model.add(layers.Convolution2D(64, (3, 3), activation="relu"))
model.add(layers.Flatten())
model.add(layers.Dense(512, activation="relu"))
model.add(layers.Dense(256, activation="relu"))
model.add(layers.Dense(actions, activation="linear"))
```

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d (Conv2D) | (None, 51, 39, 32) | 6,176 |
| conv2d_1 (Conv2D) | (None, 24, 18, 64) | 32,832 |
| conv2d_2 (Conv2D) | (None, 22, 16, 64) | 36,928 |
| flatten (Flatten) | (None, 22528) | 0 |
| dense (Dense) | (None, 512) | 11,534,848 |
| dense_1 (Dense) | (None, 256) | 131,328 |
| dense_2 (Dense) | (None, 18) | 4,626 |

**Total params:** 11,746,738 (44.81 MB)
**Trainable params:** 11,746,738 (44.81 MB)
**Non-trainable params:** 0 (0.00 B)

# ARHITEKTURA MREŽE I AGENTA

- 2. Iteracija
  - Optimizacija i pretprocesiranje ulaza, umesto 210x160, slika se pretvara u 80x80 grayscale -> značajno smanjuje broj parametara
  - Istraživanje okruženja pravljenjem random(0, 30) NOOP akcija
  - Frame-skipping: igra vrača svaki n-ti frejm (u ovom slučaju svaki 4.)
  - Max-pooling: Agregira poslednjih n frejmova
  - Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., & Bowling, M. (2017). Revisiting the Arcade Learning Environment: Evaluation protocols and open problems for general agents. arXiv preprint arXiv:1709.06009.

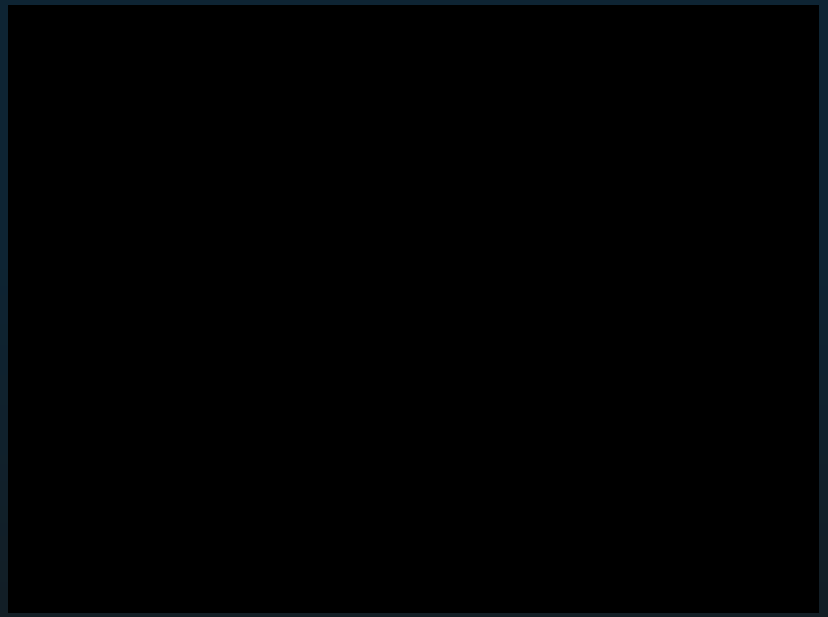
```
model.add(
    layers.Conv2D(32, 8, strides=4, activation="relu", input_shape=(4, 84, 84))
)
model.add(layers.Conv2D(64, 4, strides=2, activation="relu"))
model.add(layers.Conv2D(32, 8, strides=4, activation="relu"))
model.add(layers.Flatten())
model.add(layers.Dense(512, activation="relu"))
model.add(layers.Dense(actions, activation="relu"))
```

| Layer (type) | Output Shape | Param # |
|---|---|---|
| lambda (Lambda) | (None, 84, 84, 4) | 0 |
| conv2d (Conv2D) | (None, 20, 20, 32) | 8,224 |
| conv2d_1 (Conv2D) | (None, 9, 9, 64) | 32,832 |
| conv2d_2 (Conv2D) | (None, 1, 1, 32) | 131,104 |
| flatten (Flatten) | (None, 32) | 0 |
| dense (Dense) | (None, 512) | 16,896 |
| dense_1 (Dense) | (None, 18) | 9,234 |

**Total params:** 198,290 (774.57 KB)
**Trainable params:** 198,290 (774.57 KB)

# ARHITEKTURA MREŽE I AGENTA

- 2. Iteracija

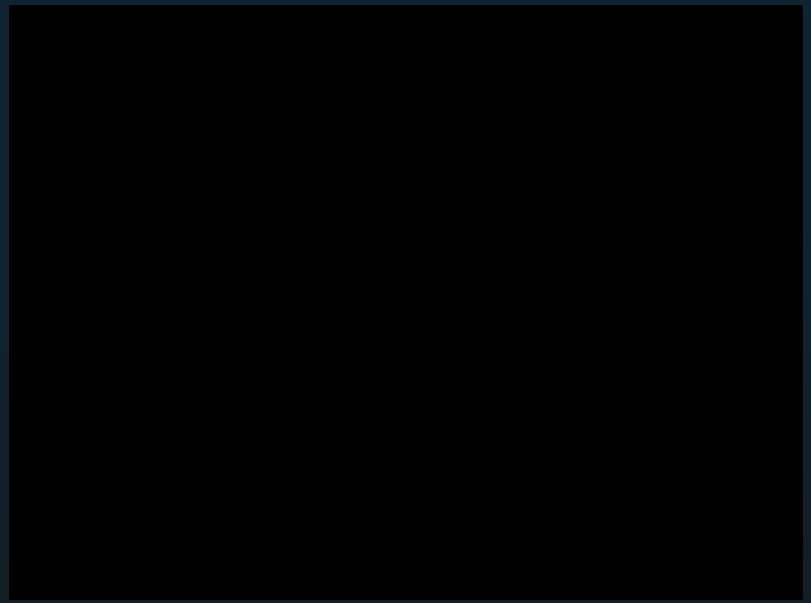


# ARHITEKTURA MREŽE I AGENTA

- 3. Iteracija
  - Kažnjavanje agenta za svaki frejm -0.01

# ARHITEKTURA MREŽE I AGENTA

- 4. Iteracija
  - Fiksiranje akcije FIRE (serviranje) na početku svakog poena (optimizacija u odnosu na tu akciju kao jedinu tačnu)



# TRENIRANJE

- Izbor parametara

```
class AgentParams:
    learning_rate = 0.0001
    seed = 8
    gamma = 0.99  # Disount factor (zanemarivanje nagrade)
    epsilon = 1.0  # Epsilon greedy
    epsilon_min = 0.1
    epsilon_max = 1.0
    epsilon_interval = (
        epsilon_max - epsilon_min
    )  # Rate at which to reduce chance of random action being taken
    batch_size = 32  # Size of batch taken from replay memory (buffer)
    max_steps_per_episode = (
        10000  # 10000 for testing/showcase, 10000000 for actual training
    )
    max_episodes = 20000  # 10 for testing/showcase, 10000 more for actual training
    epsilon_random_frames = 50000  # how many frames for taking random actions
    epsilon_greedy_frames = 1000000.0  # how many frames for exploration
    max_memory_length = 40000  # replay memory length
    update_after_actions = 4  # train the model after this much actions
    update_target_network = 10000
```

# TRENIRANJE

- Optimizer
  - ADAM
- Loss funkcija
  - U početku: CrossEntropyLoss
  - Sada: HuberLoss
  - https://ioannisanif.medium.com/deep-rl-dqn-regression-or-classification-95778dc6e68e
    (klasifikacija vs regresija)

$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for } |y - f(x)| \le \delta, \\ \delta \cdot \left(|y - f(x)| - \frac{1}{2}\delta\right), & \text{otherwise.} \end{cases}$$

# EVALUACIJA

- Evaluacija protiv Atari bota (razlika u broju osvojenih poena)
  - Prilikom play-a najviše -22
  - Prilikom treniranja

# EVALUACIJA

- Evaluacija protiv Atari bota (razlika u broju osvojenih poena)
  - Prilikom play-a najviše -22
  - Prilikom treniranja

# ZAKLJUČAK

- Za treniranje potrebno bar 10 miliona frejmova (DeepMind koristio 50 miliona)
  - Za 2 miliona frejmova potrebno bar 24h
- Prekompleksna igra za DQN metod
  - 18 mogućih akcija (sve moguće akcije sa konzole se koriste), dakle $2^{1024}$ mogućih različitih stanja ako gledamo RAM Atari 2600 kozole koja je imala
  - "*On the other hand, we were able to identify some games that seem to be harder than others for both algorithms. Both algorithms fail to make much progress on games such as Asteroids, Pitfall, and Tennis. These games generally pose hard exploration tasks to the agent; or have complex dynamics, demanding better representations capable of accurately encoding value function approximations.*" - Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., & Bowling, M. (2017). Revisiting the Arcade Learning Environment: Evaluation protocols and open problems for general agents. arXiv preprint arXiv:1709.06009.
  - Nezgodno, jer exploration ne donosi toliko dobre rezultate kao u ostalim igrama
  - Slični rezultati kao u gorenavedenom članku

# HVALA NA PAŽNJI