Aprendizaje Reforzado

Maestría en Ciencia de Datos, DC - UBA

Julián Martínez
Javier Kreiner

Deep Q-Learning para Juegos de Atari. Paper original:

- Human-level control through deep reinforcement learning: https://storage.googleapis.com/deepmind-media/dqn/DQNNaturePaper.pdf

Fuentes para el código:

- https://github.com/rohitgirdhar/Deep-Q-Networks/
- https://github.com/keon/deep-q-learning/blob/master/dqn.py
- https://github.com/AdamStelmaszczyk/dqn/

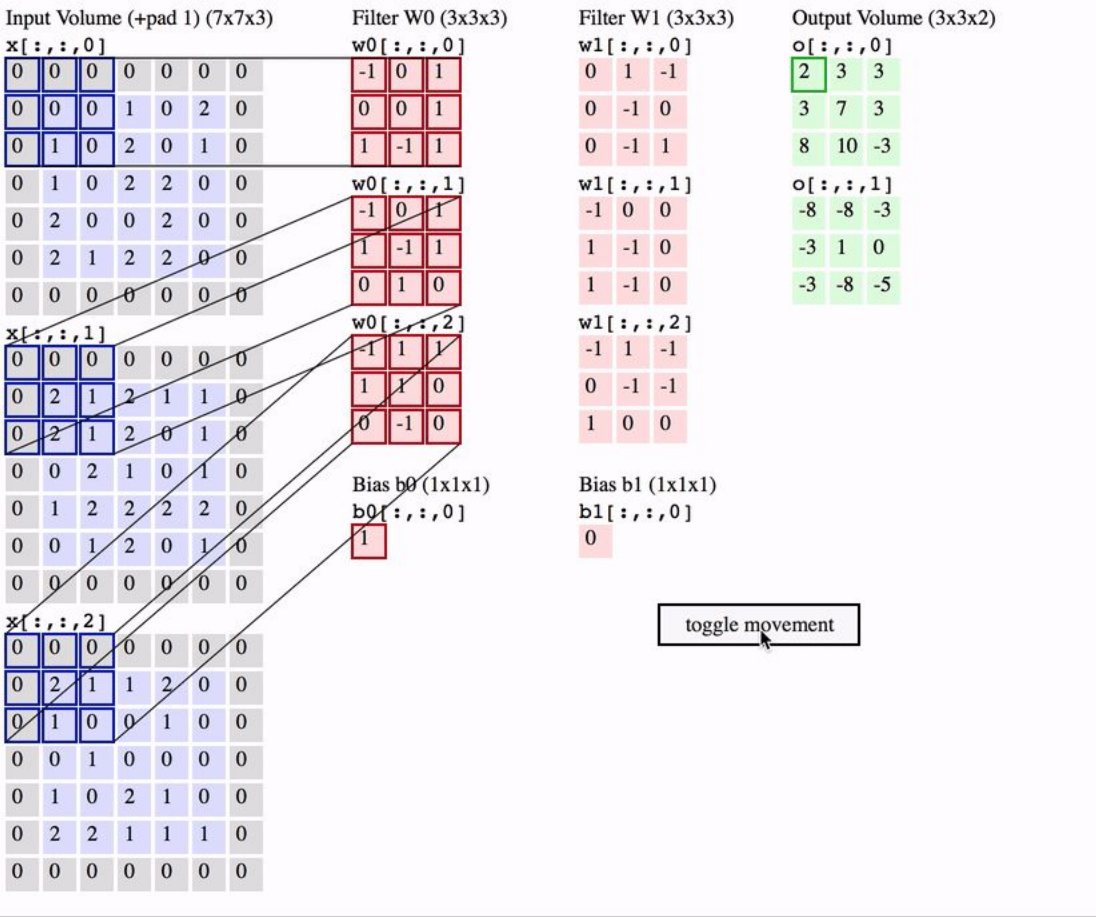
Preprocesamiento

- imagen blanco y negro en vez de canales de color
- reducir el tamaño de la imagen a 84x84
- combinar 4 frames consecutivos

Red convolucional

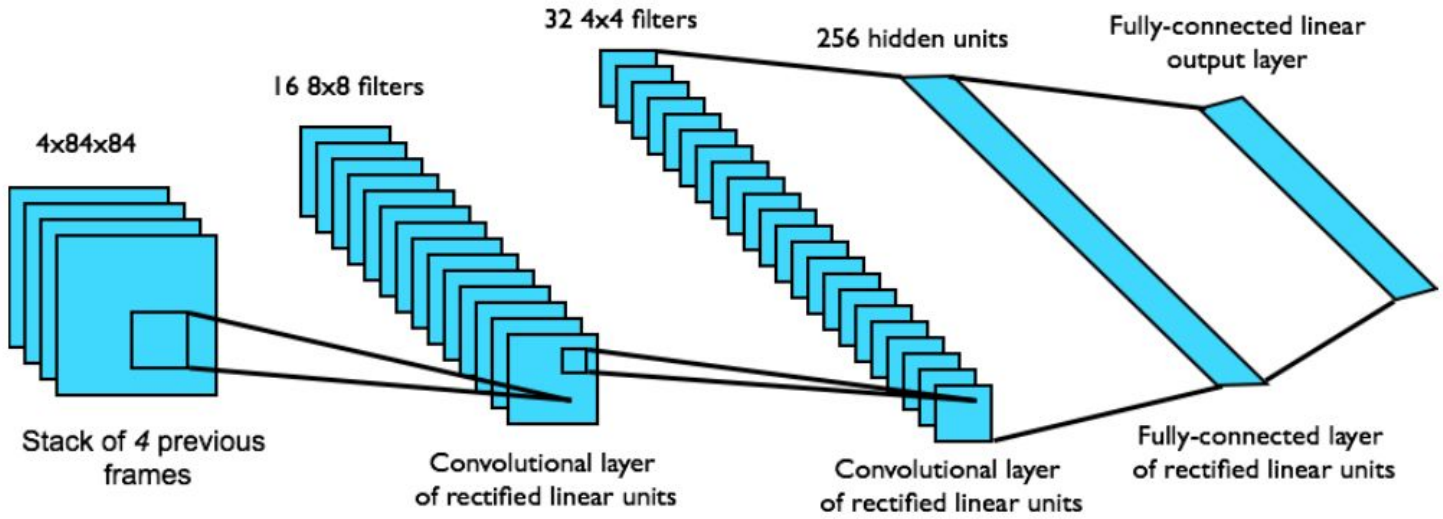
Parámetros:

- tamaño de los filtros: (w,h)
- tamaño del stride: (s_w, s_h)
- cantidad de filtros
- en keras:
  - ```
    keras.layers.Conv2D(filters,
    kernel_size, strides=(1, 1),
    padding='valid', data_format=None,
    dilation_rate=(1, 1),
    activation=None, use_bias=True,
    kernel_initializer='glorot_uniform
    ', bias_initializer='zeros',
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None)
    ```


    ```

Red convolucional

- El input son los últimos 4 frames ‘apilados’



Recordemos Q-learning

Dada $Q^k(s,a)$:

$$\pi_{k+1}(s) = \arg\max_{a'} Q^k(S_t, a'), \qquad \mu_{k+1}(a|s) = \pi^{\varepsilon}_{k+1}.$$

$$Q^{k+1}(S,A) = Q^k(S,A) + \alpha(R^+ + \gamma \max_{a'} Q^k(S^+, a') - Q^k(S,A))$$

Experience Replay

Tomo una muestra al azar de la observada con anterioridad

$$\langle s, v^{\pi} \rangle \sim \mathcal{D}$$

Actualizo con SGD

$$\Delta \mathbf{w} = \alpha(v^{\pi} - \hat{v}(s, \mathbf{w}))\nabla_{\mathbf{w}} \hat{v}(s, \mathbf{w})$$

Converge a

$$\mathbf{w}^{\pi} = \underset{\mathbf{w}}{\operatorname{argmin}}\ LS(\mathbf{w})$$

Red adicional para que los targets sean más estables

- Para que los targets sean más estables se mantiene una red con parámetros w^-_i que cambia más lentamente que w_i, o sea, cada cierta cantidad de pasos se copian los pesos de w a w^-.

- $$\mathcal{L}_i(w_i) = \mathbb{E}_{s,a,r,s' \sim \mathcal{D}_i}\left[\left(r + \gamma \max_{a'} Q(s', a'; w_i^-) - Q(s, a; w_i)\right)^2\right]$$

Pseudocódigo del algoritmo DQN:

- tomar acción a_t con política ϵ-greedy
- guardar la transición $(s_t, a_t, r_{t+1}, s_{t+1})$ en la memoria de replay D
- samplear un mini-batch aleatorio de transiciones (s, a, r, s') de D
- computar los targets de Q-Learning con respecto a los parámetros 'fijos' w^-
- Optimizar el error cuadrático medio entre la Q-network y los tardes de Q-Learning usando SGD:

$$\mathcal{L}_i(w_i) = \mathbb{E}_{s,a,r,s' \sim \mathcal{D}_i} \left[\left(r + \gamma \max_{a'} Q(s', a'; w_i^-) - Q(s, a; w_i) \right)^2 \right]$$