Введение в обучение с подкреплением

Тема 11: DQN для Atari

Лектор: Кривошеин А.В.

DQN для Atari игр

Метод DQN был успешно применён в статье 2013 года
“**Playing Atari with Deep Reinforcement Learning”** (Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller),

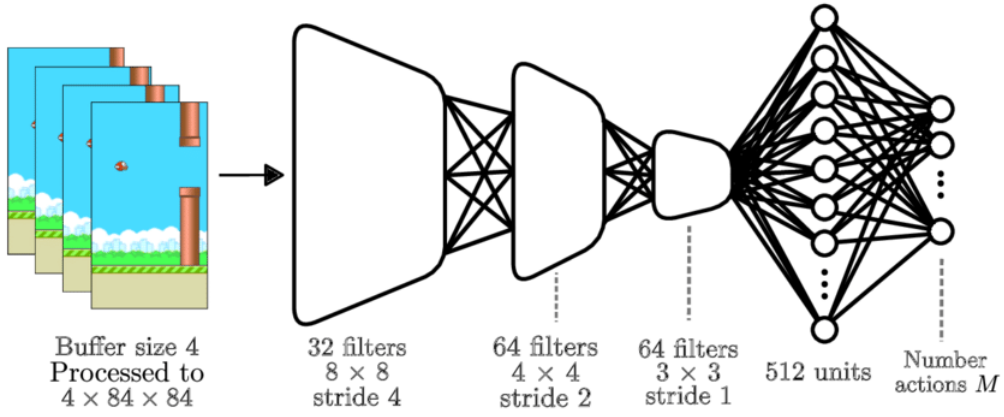
где агент обучался играть в Atari игры.

Информация о состояниях среды, то есть информация о текущем игровом моменте, передавалась агенту в виде картинки текущего игрового состояния. Но статичной картинки не достаточно, чтобы понять направление и скорость движения игровых объектов. Поэтому формировался блок из 4-х последовательно идущих фреймов, он и передавался агенту в качестве состояния.

Для обработки изображений ИНС должна включать **свёрточные слои.** Свёрточный слой представляет собой обработку изображения с помощью набора свёрточных фильтров. При применении фильтра вычисляется дискретная свёртки с изображением.

Результатом дискретной свёртки также будет некоторое изображение, которое далее подаётся на следующие свёрточные слои. Параметрами свёрточных слоёв являются коэффициенты свёрточных фильтров.

В целом свёрточные слои занимаются выделением **особенностей изображения**, которые в последующих обычных полносвязных слоях используются для формирования оценок Q-функции.



Дискретная свёртка

Обсудим подробнее дискретную свёртку. Сначала рассмотрим одномерное определение для свёртки двух последовательностей. Элементы последовательности x будем обозначать $x[n]$, где индекс n пробегает по множеству целых чисел $\mathbb{Z}$, $x = \{x[n]\}_{n\in\mathbb{Z}}$.

Рассмотрим две последовательности x, h, можно считать, что они имеют лишь конечное число ненулевых элементов. Их **(дискретной) свёрткой** называют последовательность y, элементы которой определяются равенством

$$y[n] = \sum_{k=-\infty}^{+\infty} x[k]\, h[n-k], \quad \forall\ n \in \mathbb{Z}\,.$$

Результат свёртки последовательностей x, h также обозначают $x * h$.

Дискретная свёртка

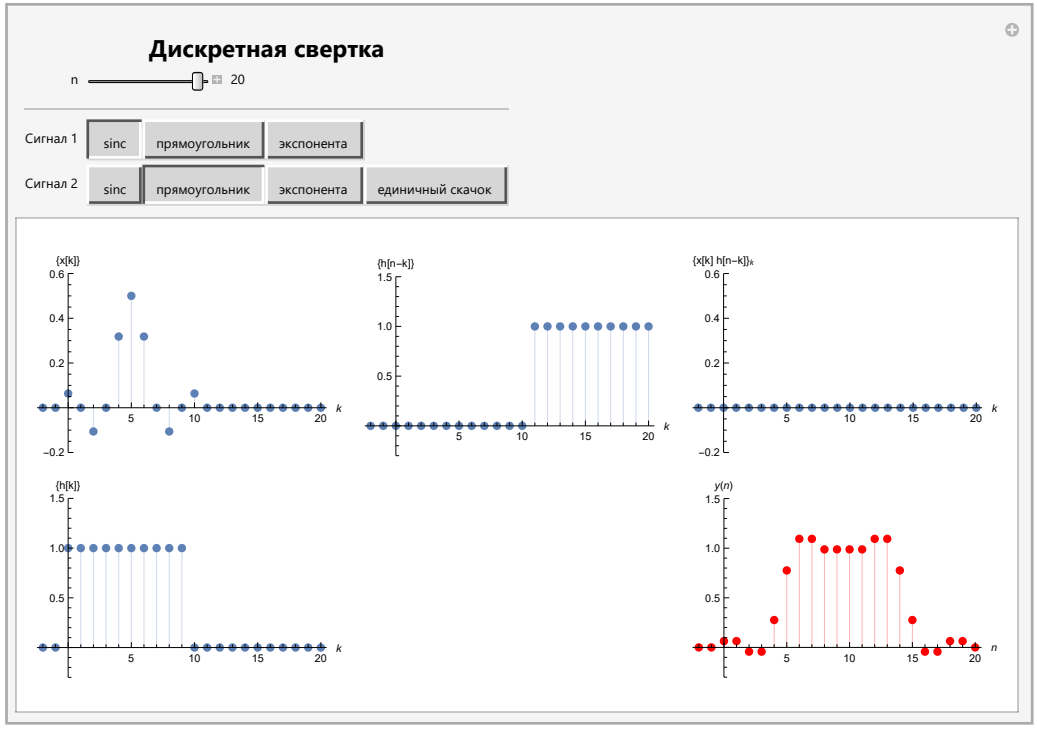
Формула свётки работает так: элемент $y[n]$ получается как результат:

1. поэлементного умножения последовательностей $\{x[k]\}_k$ и $\{h[n-k]\}_k$;
2. последующего суммирования произведений вида $x[k]\,h[n-k]$ по индексу k.

Последовательность $\{h[n-k]\}_k$ получена из $\{h[k]\}_k$ как результат:

1. отражения относительно нуля: $\{h[k]\}_k \rightarrow \{h[-k]\}_k$,
2. сдвига на n отсчетов $\{h[-k]\}_k \rightarrow \{h[n-k]\}_k$

Отсюда следует, что для подсчёта $y[n]$ используются ВСЕ элементы двух последовательностей.



Дискретная свёртка

Пусть последовательность x такова, что $x[n] \neq 0$ только если $N_1 \leq n \leq N_2$, где $N_1,\ N_2 \in \mathbb{Z}, N_1 \leq N_2$.

Пусть последовательность h такова, что $h[n] \neq 0$ только если $M_1 \leq n \leq M_2$, где $M_1,\ M_2 \in \mathbb{Z}, M_1 \leq M_2$.

Тогда $\{y[n]\} = \{x[n]\} * \{h[n]\}$ такова, что $y[n]$ отлично от нуля при $N_1 + M_1 \leq n \leq N_2 + M_2$.
Для индексов вне этого интервала $y[n] = 0$. Действительно, поскольку

$$y[n] = \sum_{k=N_1}^{N_2} x[k]\, h[n-k], \quad M_1 + k \leq n \leq M_2 + k.$$

Фильтрация изображений

Свёртка 2D последовательностей определяется аналогично одномерному случаю. Рассмотрим две последовательности x, h, индексы которых пробегают по множеству $\mathbb{Z}^2$. Их **дискретной свёрткой** называют последовательность $y = x * h$, элементы которой определяются в виде:

$$(x * h)[n_1, n_2] = y[n_1, n_2] = \sum_{(i,j)\in\mathbb{Z}^2} x[i, j]\, h[n_1 - i, n_2 - j], \quad n = (n_1, n_2) \in \mathbb{Z}^2.$$

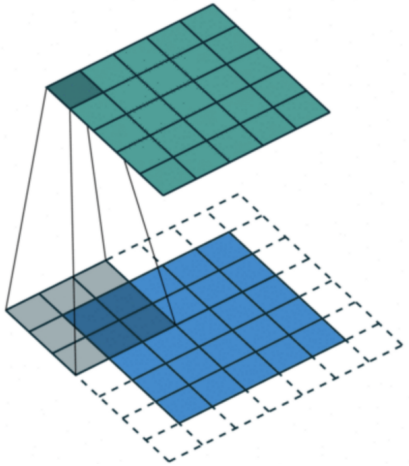
Под x можно понимать изображение (то есть последовательность с конечным числом ненулевых элементов). Последовательность h называют фильтром, а дискретную свёртку — фильтрацией.

Простейший фильтр — это скользящее среднее. Фильтрация тогда означает, что каждый пиксель отфильтрованного изображения является локальным усреднением значений яркости исходного изображения. “Локально” означает, например, что усреднение производится в квадрате 3 на 3 пикселя:

$$h[i, j] = \begin{cases} \frac{1}{9}, & i, j = -1, 0, 1, \\ 0, & \text{иначе.} \end{cases} \quad \text{или} \quad h: \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

Фильтрация изображений

Двумерную свёртку можно проиллюстрировать следующим образом:



Конечно, при фильтрации изображений, надо особым образом обрабатывать границы изображения. Простейший способ, считать, что вне границ изображения все пиксели равны нулю. Также, можно считать, что изображение зеркально продолжено за границы или отражено от границы.

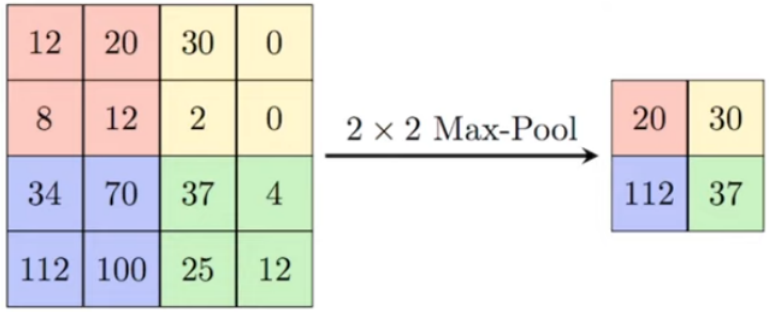
Также, результат фильтрации изображения будет чуть больше по размеру, чем исходное изображение. Как правило, в этом случае, изображение урезается до исходного размера.

Свёрточный слой

Свёрточный слой ИНС принимает на вход изображение и производит его свёртку с некоторым набором фильтров.

Для снижения размера изображений на выходе свёрточного слоя используется возможность сдвигать фильтр не на один пиксель, а на несколько сразу. В фреймворках для ИНС эта возможность задаётся параметром **stride**.

Также можно вводить слой подвыборки. Стандартный выбор — это **максимальная подвыборка** (англ. Max Pooling). Изображение разбивается на непересекающиеся блоки размера, например, 2 на 2 и в каждом блоке выбираем максимальное значение. Из этих значений составляем новое изображение. Также используют усредняющую подвыборку или суммирующую подвыборку.



Предобработка изображений

Прежде чем отправлять изображение с кадром игры агенту, производится его предобработка. Основные методы:

1. Пропуск избыточного числа кадров.
2. Объединение последовательных кадров в один некоторым образом (среднее, максимальное).
3. Перевод цветных кадров к изображению в серых тонах.
4. Вырезание из кадра только релевантной информации.
5. Снижение разрешения при низко детализированных кадрах.
6. Создание стака из последовательно идущих кадров (важно для динамических игр).
7. Масштабирование в диапазон от 0 до 1.

Модификация DQN

Выборка опыта из буфера памяти происходит случайным образом, это позволяет уменьшить корреляцию данных, на которых происходит обучение.

Однако, не весь опыт одинаково полезен. Те переходы, на которых получается большая TD ошибка, имеет смысл выдавать чаще других. TD ошибкой является величина:

$$\delta := \left| R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right|.$$

Этим соображением обосновывается модификация DQN **с приоритетным воспроизведением опыта** (англ. Prioritized Experience Replay).

Каждой записи буфера памяти мы будем назначать приоритет и извлекать записи из буфера будем с учётом приоритета. Есть две схемы назначение приоритетов: ранговая и пропорциональная.

Модификация DQN

При **пропорциональном** назначении приоритет определяется по формуле

$$p_i = (\delta_i + \varepsilon)^\alpha,$$

где p_i называют приоритетом i-ой записи,

δ_i — это TD-погрешность перехода i,

ε некоторая константа (нужна, чтобы гарантировать ненулевой приоритет). Степень α позволяет сделать подход более гибким.

При $\alpha = 0$ получаем случай равновероятной выборки.

При **ранговой** схеме приоритеты назначаются по формуле

$$p_i = \left(\frac{1}{\text{rank}(i)}\right)^\alpha,$$

где rank(i) значит позицию записи в списке всех записей, отсортированных по убыванию TD погрешности.

Приоритет легко преобразовать в вероятность выбора той или иной записи по формуле

$$P_i = \frac{p_i}{\sum_i p_i}.$$