1. Приведите пример задачи, решаемой обучением ИНС с подкреплением.

Формальная постановка задачи:

– На каждом шаге агент может находиться в состоянии $s \in S$.

– На каждом шаге агент выбирает из имеющегося набора действий некоторое действие $a \in A$.

– Окружающая среда сообщает агенту, какую награду r он за это получил и в каком состоянии s' он после этого оказался.

С несколькими состояниями ($s \in S$)

Крестики-нолики (задача — победить), где «награда» будет даваться только в конце игры – в зависимости от проигрыша или выигрыша, а чтобы ее распространить на промежуточные ходы, использовать временное разностное обучение (Temporal difference (TD) learning), а именно функцию для каждого состояния типа $V(s) := V(s) + \alpha[V(s') - V(s)]$, где агент попадал из состояния s в состояние s'. Суть многих методов обучения с подкреплением – в том, чтобы оценивать и оптимизировать такую функцию.

С одним состоянием ($|S| = 1$)

Это значит, что состояние агента не меняется. У него фиксированный набор действий и возможность выбора действия из него. Самый популярный пример такой задачи называется «Многорукие бандиты». Она заключается в том, что перед агентом есть несколько игровых автоматов, у каждого из которых своё математическое ожидание выигрыша. Нужно за ограниченное кол-во попыток выбрать лучший автомат. Эту задачу можно решить жадным алгоритмом, то есть выбирать стратегию, максимизирующую награду, но при этом награду (прибыль) ожидать оптимистично, и чтобы поменять стратегию, требовать бóльшего свидетельства. Это нужно для того, чтобы не ошибиться в конечном результате обучения при неудачных начальных данных (выборке).

2. Почему у CNN меньше параметров чем у FNN?

У CNN гораздо меньшее количество используемых весов по сравнению с любой полносвязной сетью (MLP). На лекции был дан наглядный пример. Для сети FNN, являющихся классикой MLP, у которой на входе картинка 100x100, три скрытых слоя по 100 нейронов каждый, и выходом на 10 классов, число параметров будет примерно 1M (10000*100 + 100*100 + 100*100 + 100*10).

В CNN каждая плоскость в свёрточном слое — это один нейрон, реализующий операцию свёртки (convolution) и являющийся матричным фильтром небольшого размера (например, 5x5). Затем идут слои субдискретизации (subsampling, spatial pooling), которые уменьшают размер изображения. Например, два subsampling. Если у нее тоже на входе картинка 100x100, тогда число параметров будет примерно 650K, что гораздо меньше 1M: (5*5*1*100+5*5*100*100+5*5*100*100+12*12*100*10).

3. Что такое полностью сверточная сеть?

В ней нет полносвязных слоев, тогда как в CHC они являются последними слоями перед выходным. Этот полносвязный слой преобразуется в свертку. Для этого надо пересчитать веса полносвязного слоя в матрицу (ядро) свертки. При этом, чтобы получился эквивалентный сверточный слой из полносвязного, остается установить размерность фильтра равной размерности карты признаков, а количество фильтров должно равняться количеству нейронов данного полносвязного слоя.

4. Для чего нужен слой Flatten?

Flattening трансформирует двумерную матрицу, характеризующую картинку, в вектор. В данной работе картинка 28x28, вектор будет размером 724. Если бы мы использовали CNN, это делать было бы необязательно (сверточные слои CNN Conv2D работают с входными изображениями, которые рассматриваются как двумерные матрицы), но в принципе в CHC

это иногда делают после сверточной части в слое выравнивания перед слоями обычного многослойного персептрона, использующимися для классификации. Но у нас обычная полносвязная сеть (обычный многослойный персептрон).

5. В чем разница между методами Adam и NAdam?

Если Adam по сути RMSprop с импульсом (momentum), то Nadam это RMSprop с импульсом Нестерова (Nesterov momentum). Nesterov momentum это простое изменение обычного momentum, они оба являются небольшой вариацией градиентного спуска и помогают ускорить процесс обучения. У Nesterov momentum градиент считается не из текущей позиции θ_t в параметрическом пространстве, а вместо этого $\theta_{intermediate} = \theta_t + \mu v_t$, где v (velocity) – скорость, а μ (momentum) – импульс, контролирующий, насколько быстро изменяется скорость и как локальный градиент влияет на «долгосрочное движение» (long term movement), то есть в какую сторону двигаться в долгосрочной перспективе, потому что это важно для решения задачи нахождения самого минимального локального минимума, т. е. решения задачи минимизации. Эта деталь в импульсе Нестерова помогает, потому что, грубо говоря, градиент всегда указывает в нужном направлении, то есть направлении, где потери минимальны, а импульс не всегда. И когда импульс показывает в неверном направлении или перебрасывает нас мимо искомой точки, градиент теперь помогает поправить результат прямо на данном шаге обновления.