Лекция 1. Введение в искусственный интеллект и нейронные сети.

5 ноября 2019 г.

Содержание

Искусственный интеллект – направление исследования, набор технологий, которые позволяют решать задачи, которые решает человек, задействуя мозг.

Прикладная цель создания искусственного интеллекта – это получить помощника, который будет упрощать жизнь человека.

Фундаментальная цель – понять природу человека, приблизиться к пониманию интеллекта человека. А значит и к пониманию смысла человеческого существования.

1. Подходы к построению

Как можно решать задачу построения ИИ? Исторически сложилось два пути:

1. Посмотреть, как это работает у нас (подглядеть у природы и попытаться скопировать низкоуровневые процессы). Мы знаем, что мозг – это сеть нейронов. Построим математическую модель единичного нейрона, сети нейронов, затем придумаем, как она будет обучаться. И придём к модели, которая называется **искуственные нейронные сети**.

2. Попытаться проанализировать и смоделировать высокоуровневый процесс мышления, абстрагировавшись от того, какие конкретно процессы происходят внутри и скоцентрировавшись на формальных продукционных правилах (если A, то B – экспертные системы) и логике.

   Считалось, что создавая логический язык, можно создать интеллект, сравнимый с человеческим. Такой подход называется **символьный искусственный интеллект**. Впоследствие интерес исследователей сместился в сторону ИНС.

2. История

Норберт Винер – одна из ключевых фигур в первых исследованиях в области ИИ. Придумал термин кибернетика, к 10 годам окончил школу, к 14 года получил степень бакалавра, а к 17 годам защитил докторскую диссертацию. Пытался математически описать модель мышления.

В 1950-75х годах символьный искусственный интеллект получил широкое распространение. На тот момент были чат-боты (Элиза), программы, доказывающие теоремы.

В 1975-80х годах – бум ИИ, экспертные системы.

В 90х годах ИИ потеряли былую привлекательность, а в 2010+ получили новый виток развития. Глубокие нейросети уже сегодня показывают результаты, которые лучше, чем человек, справляются с задачами, например, классификации изображений (ImageNet).

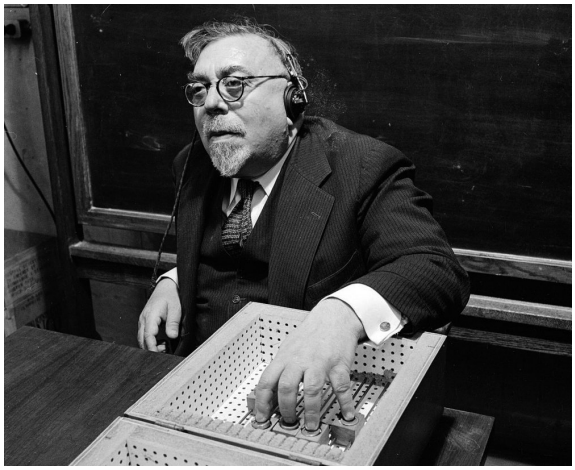

Рис. 1: Норберт Винер

3. Что такое ИНС и как они учатся?

Так что же такое Искусственные Нейронные Сети, Глубокое Обучение и с чем это всё едят?

Начать нужно с понимания, как работает мозг человека и отдельный нейрон, ведь мы подглядываем у природы, как работает нейрон и пытаемся повторить.

3.1 Базовые знания об устройстве мозга

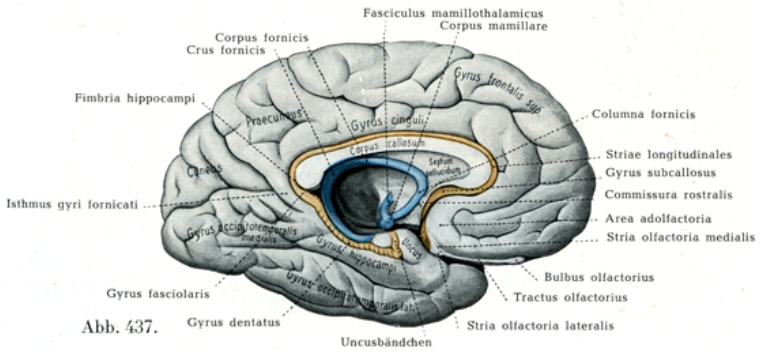

Рис. 2: Строение мозга

Мозг состоит из множества отделов. Давайте рассмотрим один из них – гиппокамп, а именно зубчатую его извилину, так называемую gyrus dentatus или fascia dentata hippocampi.

Гиппокамп – структура мозга, отвечающая за эпизодическую память, то есть память о конкретных событиях. А зубчатая фасция важна для того, чтобы мозг отличал два схожих события.

Зубчатая фасция состоит из слоёв, а каждый слой – из нейронов. Например, клетки (нейроны) гранулярного слоя зубчатой фасции лабораторных мышей типа C57BL/6 (mus musculus – домовая мышь).

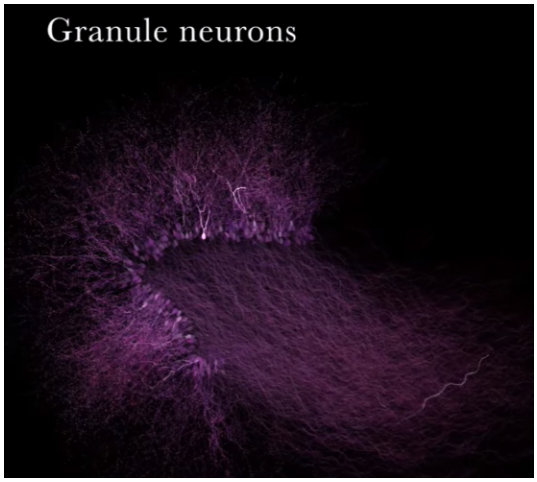

Рис. 3: Гранулярный слой зубчатой фасции гиппокампа мыши

Каждый нейрон в этом (и в любом другом) слое состоит из тела клетки (шарик), дендритного дерева (входы нейрона) и аксона (выхода).

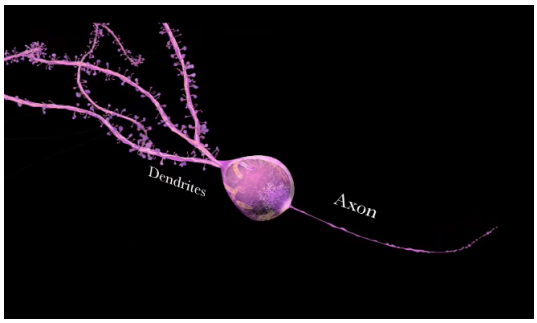

Рис. 4: Строение нейрона

Пусть по аксону другого слоя, который лежит рядом с дендритным деревом рассматриваемого нейрона, проходит сигнал. Нейромедиаторы – это химические вещества, которые позволяют понять одному нейрону, что активировался другой нейрон, то есть служат для передачи состояний между нейронами. Один нейрон "чувствует запах"другого нейрона и происходит формирование синапса – связи между двумя нейронами.

В зависимости от типа нейрона различается длина аксонов и дендритов, а значит и число нейронов, которые они могут активировать и от которых, наоборот, принимать сигнал активации.

3.2 Математическая модель

Искусственный нейрон (или перцептрон) была предложена Фрэнком Розенблаттом в 1957 году.

Теперь, после того, как мы примерно поняли, что происходит на уровне биохимии в мозге, нужно перевести это знание на язык математики.

Будем характеризовать нейрон с индексом i некоторым количеством входов (синапсов) $x_j, j \in [1,N]$, у каждого из которых различная **концентрация нейромедиатора** (как хорошо расыпляет), то есть различное численное значение x_j. Также у каждой связи есть характеристика, которую мы будем называть весом и обозначать w_{ij}. Это характеристика – **чувствительность нейрона**, то, насколько хорошо он улавливает какой-либо нейромедиатор, реагирует на него.

Затем происходит суммирование результатов сигналов на всех синапсах (чем больше чувствительность w_{ij} и больше концентрация нейромедиаторов x_j, тем больший вклад в активацию даёт этот нейрон). Итак:

$$a_i = \sum_{j=1}^{N} w_{ij} \cdot x_j$$

Нейрон может быть либо активен (1), либо неактивен (0). Чтобы принять решение о том, передавать ли дальше сигнал или нет необходимо ввести **нелинейную функцию активации**, которая будет ставить в соответствие всякому $a_i \in (-\infty; +\infty)$ значение $y_i = \Phi(a_i - \theta_i) \in [0,1]$.

Функция Φ называется функцией активации и обычно представляет сигмоидальную функцию типа $\sigma(x) = \dfrac{1}{1+e^{-x}}$. Сдвиг в аргументе функции активации связан с наличием потенциала покоя нейрона и часто называется **bias** в англоязычном сообществе. Этот сдвиг не даёт нейрону сработать случайно и устанавливает порог активации.

3.3 Базовые понятия

Прелесть подхода к решению задач с использованием нейронных сетей заключается в том, он позволяет научиться решать задачи без явного указания алгоритмических шагов по достижению цели. Обратимся к основным понятиям нейронных сетей и машинного обучения.

Обучающая выборка – правильные примеры решения какой-либо задачи, на которых нейронная сеть учится. Например, размеченный набор картинок для задачи бинарной классификации.

Тестовая (валидационная) выборка – неиспользованные при обучении примеры решения задачи, на которых проверяется качество модели и то, выучила ли она какие-то отличительные характеристики примеров различных классов (если речь идёт о задачах классификации).

Обучение нейросети – изменение весов в соответствие с вычисленной разницей между предсказанным моделью и истинным результатом решения задачи (например, метки класса).

Но как конкретно происходит перенастройка весов в нейросети? Рассмотрим способ обратного распространения ошибки (backpropagation или просто backprop).

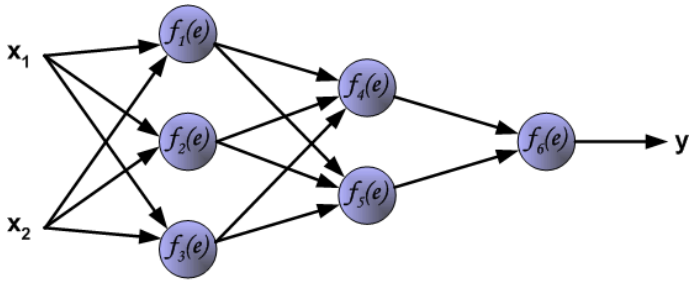

Рис. 5: Трёхслойная простейшая нейросеть

Итак, для начала отметим, что каждый нейрон может быть декомпозирован на два базовых элемента – элемент суммирования и элемент активации. Первый вычисляет сумму произведений входов и весов, а второй – применяет к результату нелинейную функцию типа сигмоиды.

Допустим, мы подаём на вход нейросети пример из обучающей выборки, а именно – значения входных сигналов x_1 и x_2 и правильное значение на выходе z. Обозначение $w_{(xi)j}$ означает вес синапса от входного сигнала с номером (xi) к нейрону с номером j.

Значения (y_i) на выходах нейронов первого (входного) слоя:

$$y_1 = f_1(w_{(x1)1} \cdot x_1 + w_{(x2)1} \cdot x_2)$$

$$y_2 = f_2(w_{(x1)2} \cdot x_1 + w_{(x2)2} \cdot x_2)$$

$$y_3 = f_3(w_{(x1)3} \cdot x_1 + w_{(x2)3} \cdot x_2)$$

Теперь рассморим, что происходит во втором (скрытом) слое. При этом входами для нейронов скрытого слоя уже будут выступать сигналы от нейронов входного слоя, а именно те самые y_i, которые мы вычислили выше. Для краткости будем обозначать их в сокращённой форме, то есть y_i, не расписывая вычисленные выражения целиком. Веса w_{ij} ниже означают вес связи между i-м нейроном входного слоя и j-м нейроном скрытого слоя.

Значения функций активации y_i на нейронах скрытого слоя:

$$y_4 = f_4(w_{14} \cdot y_1 + w_{24} \cdot y_2 + w_{34} \cdot y_3)$$
$$y_5 = f_5(w_{15} \cdot y_1 + w_{25} \cdot y_2 + w_{35} \cdot y_3)$$

И сигнал на выходном нейроне в итоге:

$$y = y_6 = f_6(w_{46} \cdot y_4 + w_{56} \cdot y_5)$$

Затем вычисляется разница между истинной и предсказанной меткой (то есть между $y = y_6$ и z): $\delta = z - y$.

Теперь ошибка распространяется назад с теми же весами между нейронами, что и при прямом проходе через нейросеть. Это работает, потому что логично предположить, что нейрон с большим вкладом в активацию какого-либо другого нейрона внесёт и большую ошибку.

Тогда ошибка для нейронов скрытого слоя:

$$\delta_4 = w_{46} \cdot \delta$$
$$\delta_5 = w_{56} \cdot \delta$$

И для нейронов входного слоя:

$$\delta_1 = w_{14} \cdot \delta_4 + w_{15} \cdot \delta_5$$
$$\delta_2 = w_{24} \cdot \delta_4 + w_{25} \cdot \delta_5$$
$$\delta_3 = w_{34} \cdot \delta_4 + w_{35} \cdot \delta_5$$

Далее, после вычисления ошибок для каждого нейрона, можно заняться корректировкой весов с учётом направления, величины ошибки, значению входного сигнала и скорости обучения:

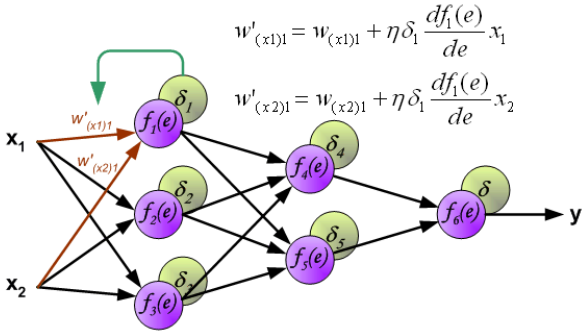

Рис. 6: Пересчёт весов для первого нейрона

Это базовый способ обучения нейросети. Сейчас делают немного хитрее и берут не один пример из обучающей выборки, а несколько, так называемый **batch** примеров и для обновления весов используем усреднённый по батчу градиент. Это снижает вычислительную сложность. Такой метод называется стохастическим градиентным спуском (stochastic gradient descent).

4. Основные направления исследований

4.1 Свёрточные нейронные сети. Машинное зрение

Свёртка – математическое понятие, численно отражающее перекрытие двух функций. Формальное определение для непрерывных функций f, g:

$$(f * g)(t) \stackrel{def}{=} \int_{-\infty}^{+\infty} f(\tau) \cdot g(t - \tau) d\tau$$

В случае дискретных функций:

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n - m]$$

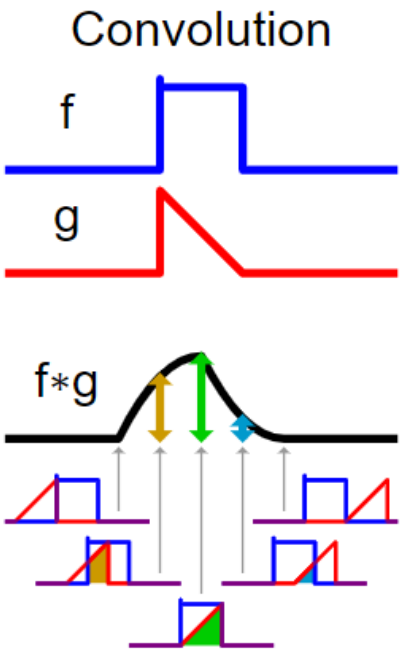

Рис. 7: К определению понятия свёртки двух функций

Это понятие удобно для обработки изображений и решения задач машинного зрения. Мы двигаем шаблон по картинке и смотрим на предмет наилучших совпадений сигнала с функцией шаблона.

Рассмотрим двумерную свёртку.

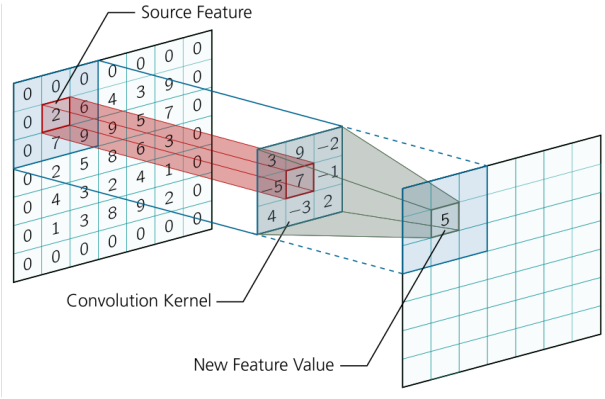

Рис. 8: Двумерная свёртка

Вспомним определение свёртки дискретных функций и поймём, что это просто сумма перемноженных компонент из матрицы исходных признаков (source features) s_i и k_i из матрицы-шаблона – ядра свёртки (convolution kernel). Посчитаем новое значение признака f для примера с рисунка № 8 на странице 9:

$$f = \sum_{i=0}^{8} s_i \cdot k_i \quad = (0 \cdot 3) + (0 \cdot 9) + (0 \cdot (-2)) + (0 \cdot (-5)) + (2 \cdot 7) +$$
$$+(6 \cdot (-1)) + (0 \cdot 4) + (7 \cdot (-3)) + (9 \cdot 2) =$$
$$= 14 - 6 - 21 + 18 = 5$$

Таким образом сканируется вся исходная матрица (вся картинка с шагом в один или несколько пикселей) и на выходе получаем карту признаков. Создавая такую сеть помимо самих свёрток используют ещё операцию сжатию карты признаков, уменьшая число значений в ней в какое-то число раз (обычно в 2 по каждому измерению).

Таким типом нейросетей можно решать задачи нахождения объектов на изображении, предсказания пола и возраста на фото, эмоций людей.

Но можно делать ещё более хитрые вещи, а именно – переносить стиль одного изображения на другой. В процессе свёртки размерность матрицы признаков уменьшается, однако вся информация о изображении остаётся и мы можем, грубо говоря, раскладывать её по разному базису – по разным стилям оформления. То есть получить представление произвольного содержания теми признаками, которые удалось выудить из операций свёртки над изображением данного стиля.

А ещё можно научить свёрточную нейронную сеть читать по губам, синтезировать изображение говорящей головы любого человека и он будет говорить что угодно. Скрытые в недрах нейросети представления лиц и предметов в виде векторов позволяют производить простейшие арифметические операции (см. рисунок № 9.

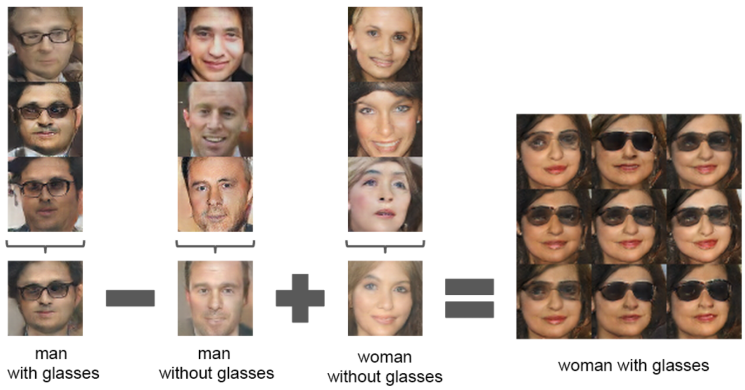

Рис. 9: Арифметика нейросетей

4.2 Рекуррентные нейронные сети

Обычно используются в задачах предсказания следующего члена какой-либо последовательности. Например, следующего слова в тексте.

Классические нейронные сети (feedforward) не обладают памятью и вход на следующем примере из обучающей выборки (наприер, на следующем слове из текста) никак не зависит от того, что происходило на всех предыдущих шагах.

Чтобы решить эту проблему, люди попытались добавить в нейросеть память. И логично было идти по пути представления полученной нейросети в виде обычной нейросети только с глубиной равной числу шагов. И при этом на каждом шаге сеть занимает определённое состояние. Чем плох такой подход? Во-первых, сигнал затухает, во-вторых, сеть не учится.

Тогда Юрген Шмидхубер придумал вместо обычного нейрона делать ячейку памяти на нейронах и назвал её LSTM (long short-term memory).

Ячейка долговременной краткосрочной памяти (LSTM). У ячейки есть вход старого значения памяти, старого значения активности (выход на предыдущем такте), текущий вход (иксы), текущий выход и выход нового значения активности.

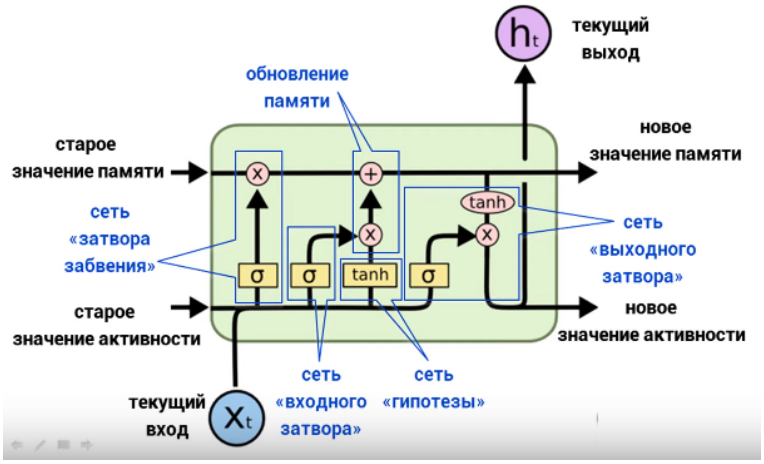

Рис. 10: Строение ячейки LSTM

Внутренний состав:

- **Тракт памяти**, в который приходит старое значение памяти и из которого выходит новое значение памяти, которое может измениться при проходе по этому тракту, а может остатья нетронутым
- **Сеть затвора забвения**. Позволяет стирать из памяти значения на основании предыдущих и текущих тактов. То есть мы можем так выучить веса, чтобы сеть научилась в нужное время стирать нерелевантную информацию из памяти.
- **Сеть входного затвора**. Позволяет подготовить данные для последующей работы с ними. Просто учитывает выход предыдущего такта и вход текущего.
- **Сеть гипотезы**. Отвечает за знак добавления в память. Ведь новое знание это не только про добавление в память со знаком $+$, но и со знаком $-$. То есть гипотеза может быть либо на какую-то долю верна, тогда мы подкрепляем память об этом в положительную сторону, либо неверна, тогда в отрицательную. Если в этой сети была бы тоже сигмоида, мы могли бы только добавлять в память.
- **Сеть выходного затвора**. Просто формирует выход на текущем такте с учётом предыдущего и значения из памяти (ведь мы можем как всё помнить про предыдущие состояния, так и всё забыть).

Как теперь это применить? Будем предсказывать следующий символ и обучать сеть на субтитрах к диалогам из фильмов. И рекуррентная нейронная сеть научится осмысленно продолжать фразы, правильно писать слова (выучит грамматические правила языка), правильно будет расставлять знаки препинания (выучит синтаксис языка) и даже будет генерировать сентимент-близкие ответы на фразы.

Пример сложной диалоговой системы: **XiaoIce** от Microsoft Research Asia. Помимо осмысленного ведения диалога, выражения эмпатии к собеседнику, комментириев и шуток про изображения в диалоге, XiaoIce может ещё и управлять эмоциями пользователя, предсказывая, в какую сторону следует склонить беседу для максимизации вероятности положительной реакции пользователя на происходящее в чате. А ещё сочиняет стихи и сейчас даже опубликован сборник таких творений.

4.3 Обучение с подкреплением

Имеем некоторого агента, который взаимодействует с внешним миром. Обучающей выборки нет, но есть награждающие действия. Задача агента – выбирать действия, максимизирующие награду.

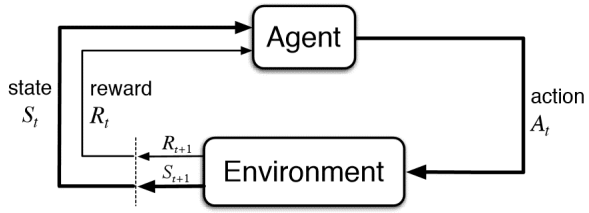

Рис. 11: Общая схема обучения с подкреплением

При этом **агент**: получает состояние s_i (state), получает скалярное значение награды r_t (reward) и выполняет действие a_t (action).

А среда: получает действие a_t, генерирует состояние s_t, генерирует скалярное значение награды r_t.

Стратегия π определяет выбор действия для данного состояния среды

$$a = \pi(s)$$

Функция полезности – полная ожидаемая награда при выборе действия a в состоянии s при использовании политики (policy, strategy) π:

$$Q^{\pi}(s,a) = \mathbb{E}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...|s,a]$$

Затем решается уравнение Беллмана итеративным расчётом функции полезности:

$$Q_{t+1}(s_t,a_t) = Q_t(s_t,a_t) + \alpha_t(s_t,a_t) \cdot (R_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q_t(s_t,a_t))$$

Модели обычно проверяют на игровом пространстве. А затем переносят в реальный мир.

В 2015 году DeepMind попадает на обложку журнала Nature со статьёй, в которой показано, что глубокое обучение с подкреплением позволяет обучиться играть в компьютерные игры не зная правил заранее лучше человека. Google впоследствие покупает DeepMind за $600M. А этот результат открывает качественно новые области для применения глубокого обучения в задачах управления, ИИ и робототехники.

В 1997 году нейросеть DeepBlue обыграла в шахматы Гарри Каспарова (чемпиона мира на тот момент), а 9 марта 2016 нейросеть AlphaGo (детище DeepMind) обыграла Ли Седоля (чемпиона мира по игре в го). Важно отметить, что у го и шахмат очень разнятся коэффициенты ветвления (число разрешённых позиций, в которые можно попасть из текущей). Для шахмат это ≈ 30, а для го ≈ 250. И снова на обложке Nature!

Удивительно, что AlphaGo нашёл методы игры, которых не знали профессиональные игроки и последние позаимствовали у алгоритма некоторые интуиции для удачного исхода игр.

Такие архитектуры можно использовать ещё и для уменьшения потребления электроэнергии (Google так и сделали) и для, например, поиска новых лекарств.