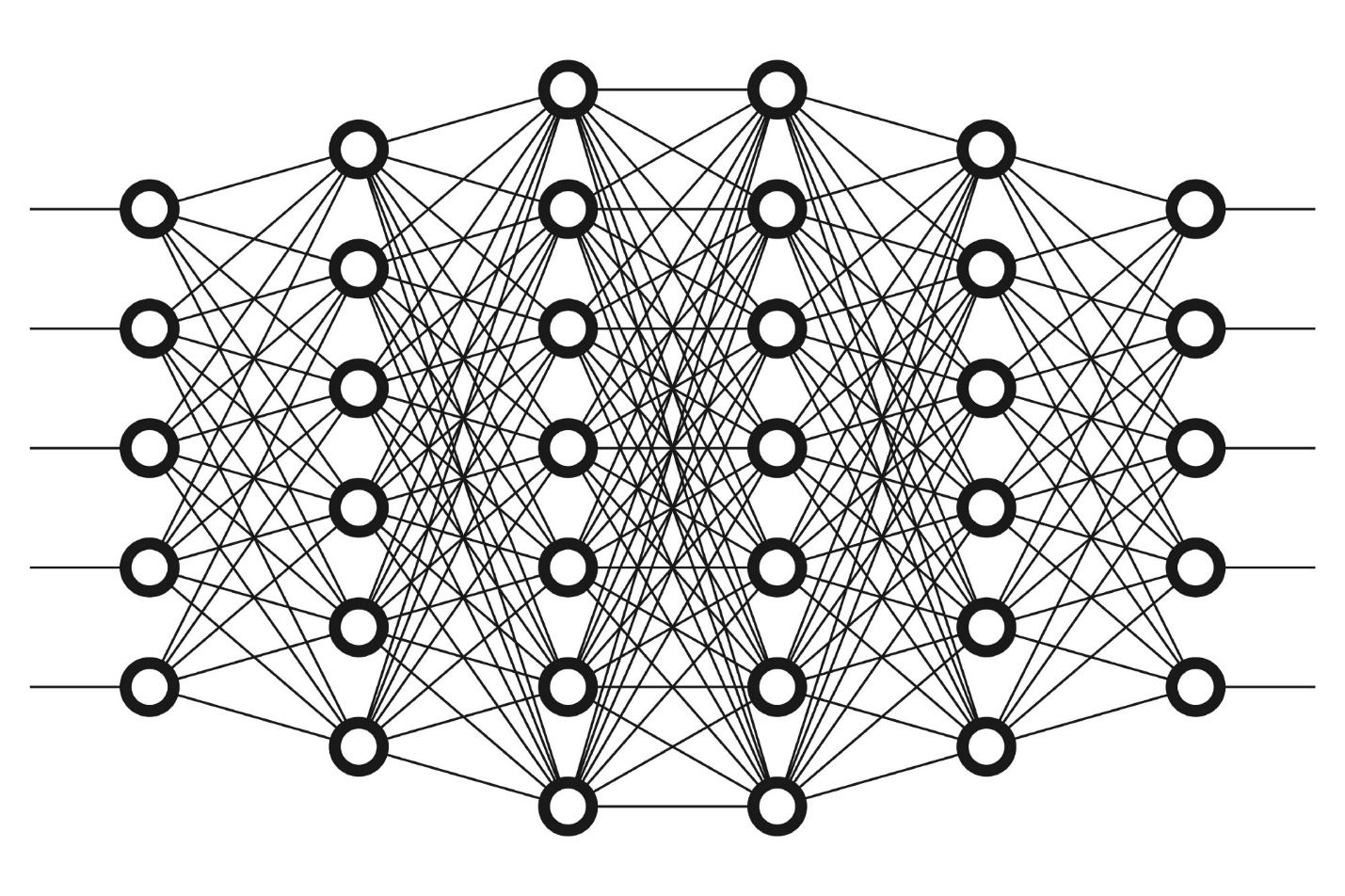
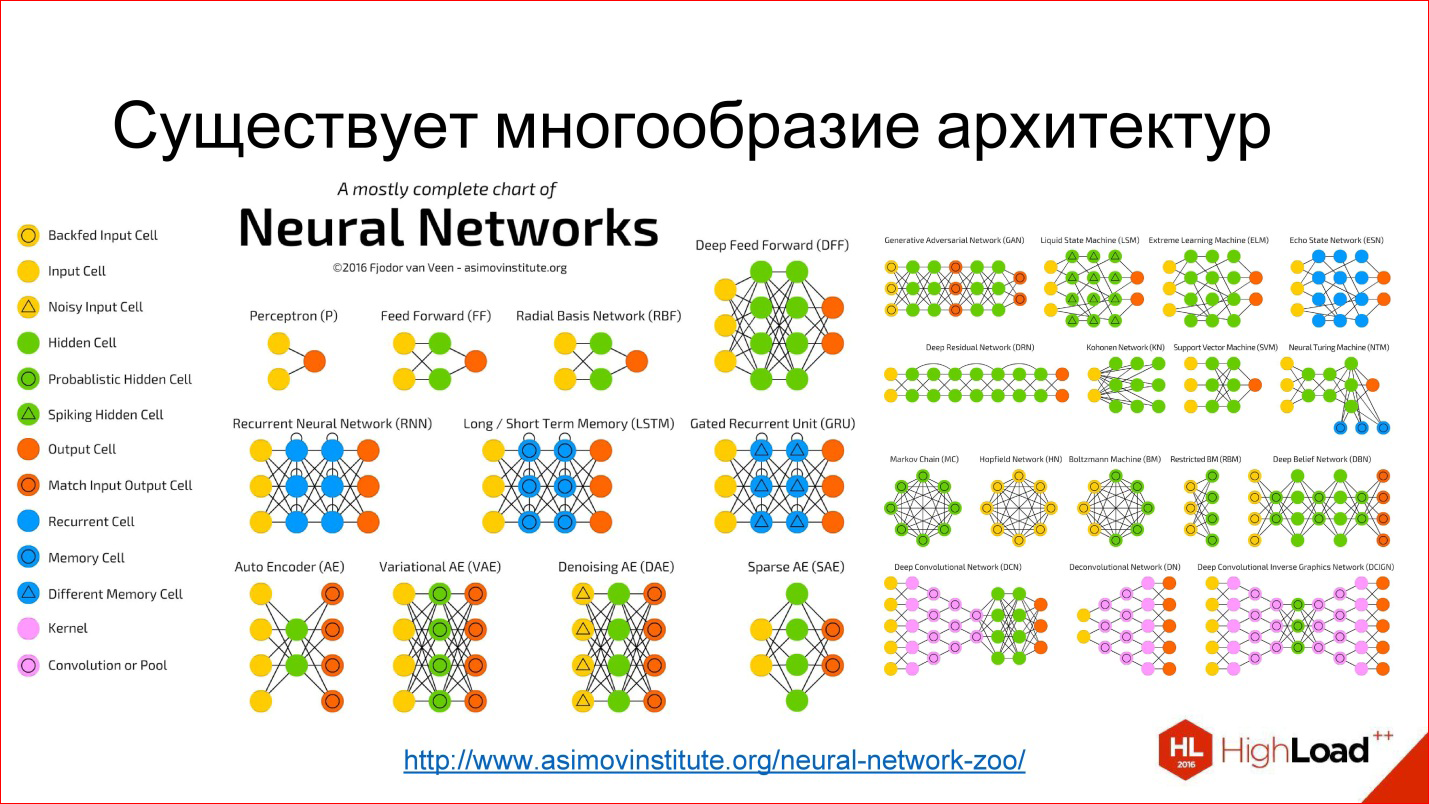
**Что такое нейросети? Для чего нужны и как работают?**

В этом докладе я попытаюсь объяснить основы нейронных сетей. Прежде всего стоит отметить, что нейронные сети – это просто нейроны связанные между собой с помощью графов. Структура нейронной сети пришла в мир программирования прямиком из биологии. Благодаря такой структуре, машина обретает способность анализировать, изменять и запоминать различную информацию.



Хорошо, а что же тогда такое нейрон?

По своей сути это ячейка, хранящая в себе значение от нуля до единицы (в более продвинутых нейросетях это значение может варьироваться от -1 до 1). Нейрон получает на вход информацию, проводит над ней математические вычисления и передаёт её дальше.

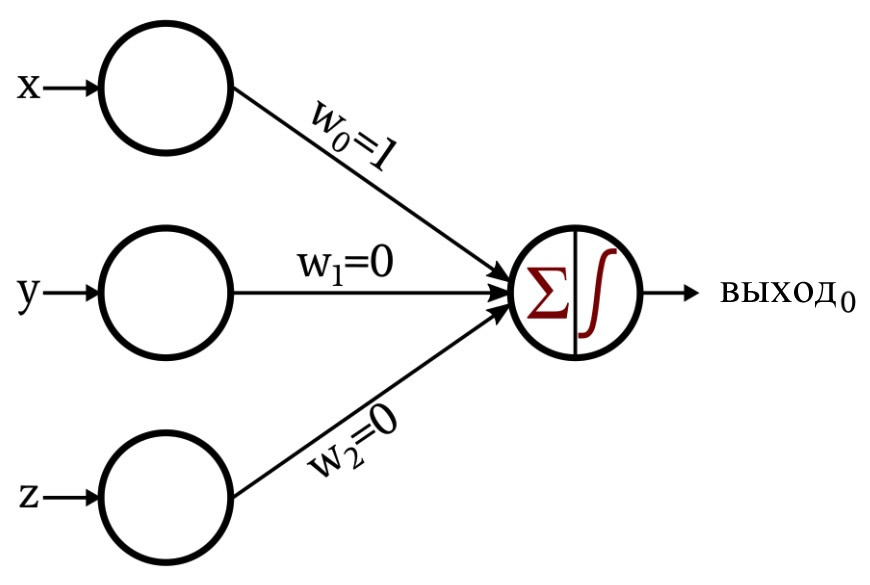


Нейронные сети используются для решения сложных задач, которые требуют аналитических вычислений подобных тем, что делает человеческий мозг. По сути, задача любой нейронной сети – как-либо обработать информацию. Для различных задач существует огромное множество архитектур построения. Выше показана малая доля из всех возможных архитектур (самые популярные из них).

Остановимся на самой простой архитектуре - **перцептрон**.

Перцептрон – это нейронная сеть, которая представляет собой алгоритм для выполнения двоичной классификации. Он определяет, относится ли объект к определенной категории (например, является ли животное на рисунке кошкой или нет).

На картинке ниже показан перцептрон. У него три входных нейрона (input – x, y, z), веса (w) и нейрон с функцией активации. К этому всему чуть позже.

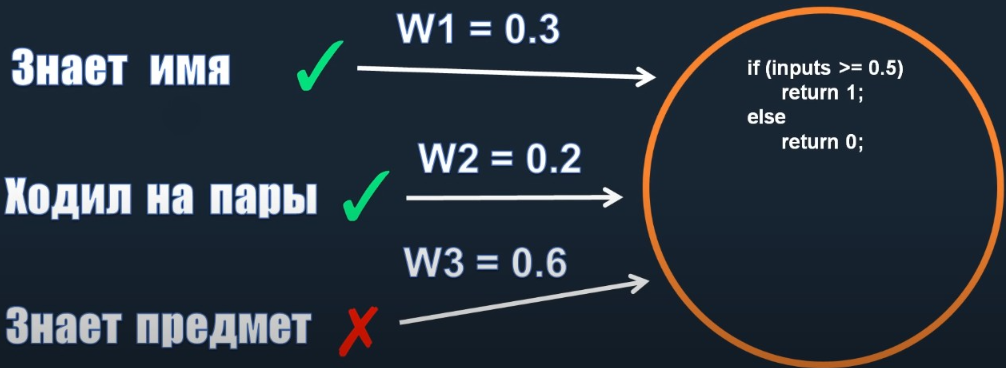


Рассмотрим пример:

Представим, что с помощью нейрона мы описываем работу преподавателя в университете. Задача – определить, поставит ли преподаватель зачёт студенту или нет.

Нейросеть получает сигналы из инпутов (input – вход), обрабатывает их и даёт на выход (output - выход) значение один или ноль. Один – студент получит зачёт, ноль – нет.

Первый вход отвечает за то, знает ли студент имя преподавателя. Второй – ходит ли он на пары. Третий – знает ли он предмет. Если студент знает имя, на вход придёт единица иначе пришёл бы ноль.



Чтобы обработать пришедшую информацию, нейрон использует так называемую **функцию активации**. Здесь представлена самая простая из них – линейная функция, в крупных нейросетях обычно используют функции сложнее данной.

Так же можно заметить, что все графы имеют какую-то переменную. Это переменная – **вес**. Значение каждого инпута для конкретного решения может носить разный характер, разный вес. Взять пример из реальной жизни, преподаватель, думая поставить ли зачёт студенту, скорее всего в первую очередь будет смотреть на знание самого предмета, а потом уже на прочие факторы. Каждый инпут перед вычислениями в функции активации получает множитель, этот самый вес. Обычно веса изначально ставят случайным образом (от 0 до 1).

Вернёмся к примеру. Давайте выясним, получит ли студент зачёт, исходя из изображения выше.

**Output = 1 \* 0.3 + 1 \* 0.2 + 0 \* 0.6 = 0.5**

В данном случаи студент знает имя и ходит на пары, тем самым на выходе нейрон выдаст единицу, студент получит зачёт.

Представим, что студент знает имя преподавателя, но не ходит на занятия и не знает предмет.

**Output = 1 \* 0.3 + 0 \* 0.2 + 0 \* 0.6 = 0.3**

Меньше 0.5, значит функция активации вернёт ноль, студент не получит зачёт.

Наверняка вы слышали, что нейронные сети можно тренировать, обучать. Но что это значит? Ключевой момент нейронных сетей – это то, что **значения весов могут изменяться**, благодаря этому и происходит обучение. Ведь если бы эти множители были бы другие, то и результат работы сети мог бы быть иным. Например, у нас бы мог быть преподаватель, который ставит зачёты всем, кто просто ходит на пары.

Как тренировать нейроны?

Обучение заключается в том, чтобы найти **правильные веса**. Обучение построено на простой идее, что если мы на выходе нейрона знаем, какой должен быть ответ, и знаем, какой он получился, нам становится известна эта разница, ошибка. Эту ошибку можно отправить обратно ко всем входам нейрона и понять, какой вход насколько сильно повлиял на эту ошибку, и соответственно, подкорректировать вес на этом входе так, чтобы ошибку уменьшить. Грубо говоря, если нам известно, что из 5 долларов должно получиться 410 рублей, то мы можем подогнать веса к этому результату.

Теперь рассмотрим, как работает обучение нейрона на реальном примере:

Вы можете перейти на сайт и рассмотреть код, чтобы понять, как работает тренировка поподробнее.

(ссылка на github) <https://petrxleb.github.io/neuralnetwork/>

В данном примере показано обучение одного нейрона архитектуры перцептрон. **Задача программы** – научится конвертировать килограммы в фунты. Стоит отметить, что задача выполняется намного проще путём обычного деления на коэффициент.

let kg = 100;

let funt = kg / 0.45359237;

Но в качестве понимания тренировки весов, стоит попробовать выполнить эту задачу с помощью нейросетей.

Существует общая формула для тренировки подобных сетей, выглядит она следующим образом:

**W(новый) = W(текущий) + (Smoothing \* (Result(ожидаемый) – Result(рассчитанный)), где**

**Smoothing** – это сглаживание, нужное для того, чтобы не было ситуация, когда шаг настолько большой, что программа не может переступить его. Чем ближе сглаживание к нулю, тем лучше будет итоговый вес. В данном программе сглаживание равняется 0.0001

**Result(ожидаемый)** – это то значение, результат которого мы точно знаем (например то, что 100кг = 220.46226 фунтам

**Result(рассчитанный)** – это то значение, что мы получили в последней итерации

Стоит отметить, что в больших нейронных сетях используют методы обучения, в фундаменте которых лежат различные математические функции, куда больше и сложнее, чем то, что показано здесь, но для понимания базы хватит и этого.

Сама программа без вёрстки выглядит так:

<script>

let inputVal = document.getElementById('input\_txt');

let outputVal = document.getElementById('output\_txt');

    //только для чтения

    outputVal.readOnly = true;

    //один нейрон

    let neuroset = {

        //вес

        weight: 0.5,

        //ожидаемый результат из 100 кг

        expectResult: 220.46226

    }

       function doCalc() {

        outputVal.value = (inputVal.value\*neuroset.weight).toFixed(7);

        console.log("---------------------");

        console.log(inputVal.value \* neuroset.weight);

        console.log("---------------------");

       }

       function Train(){

        console.log("Начало тренировки");

        console.log("---------------------");

        let actualRes;

        let lastErr;

        let correction;

        let smoothing = 0.0001;

        let i = 0;

            do {

                //получаем текущий результат обычным переумножением

                 actualRes = inputVal.value \* neuroset.weight;

                 lastErr = neuroset.expectResult - actualRes;

                //корректируем весы последними значениями

                 correction = (lastErr / actualRes) \* smoothing;

                 neuroset.weight += correction;

              //иттерируем ход

                i++;

                if (i%10000==0) {

                    //выводим каждую 10-ти тысячную операцию

                console.log(`Иттерация:\t${i}\nОшибка:\t${lastErr}`);

                }

                //

            } while ((lastErr > smoothing)||(lastErr < -smoothing));

            console.log("---------------------");

            console.log("Конец тренировки");

       }

</script>

Тренировка сети будет продолжаться до тех пор, пока не выполниться условия сглаживания:

      } while ((lastErr > smoothing)||(lastErr < -smoothing));

Изначальный вес равен 0.5 и конвертация без тренировки будет неверна. Попробуйте задать 100кг, сеть выдаст в ответе 50.

Теперь попробуем натренировать сеть, для этого в программе задан точный ответ конвертации, то что 100 кг точно равняются 220.46226 фунтам. После нажатия кнопки, мы попадаем в цикл, в котором вычисляется ранее написанная формула (в разбитом виде для понятности). Для лучшего понимания можно открыть консоль в браузере.

Так как сглаживание равно 0.0001, то программа проходит довольно много циклов (290.000) прежде чем результат будет удовлетворять условиям.

Так, после прохождения цикла, нейросеть научилась конвертировать килограммы в фунты. Стоит отметить, что сеть не выдаёт 100% верные результаты, но чаще всего устраивает и 99.9%. Чем меньше сглаживание, тем точнее результат. На сайте есть точные результаты конвертации, вы можете проверить получившиеся результаты.

**Итого:**

Нейронные сети – это нейроны, связанные между собой графами с весами. Нейрон хранит в себе значение от нуля до единицы. Нейроны можно обучать, изменяю веса с помощью специальных математических функций. Чем сложнее сеть, тем сложнее её обучение. Думаю, излишним будет говорить, что за нейронными сетями большое будущее.

*Чувствин П.П.*

*ИСиТ 2 курс*