СЛАЙД 1

Всем привет! На сегодняшнем уроке мы наконец закончим с нейронными сетями)

СЛАЙД 2

На прошлом уроке мы остановились на решении задачи обновления весов связей в нейронной сети и на том, что данная задача оказалось очень сложной и ее удалось решить лишь в 60-70-х годах прошлого века.

СЛАЙД 3

Итак, как же нам разрешить эту явно трудную проблему?
Давайте немного отвлечемся и представим себе некий ландшафт с очень сложным рельефом, имеющим возвышения и впадины, а также холмы и ямы. Вокруг так темно, что ничего не видно. Но, например, вы знаете, что находитесь на склоне холма, и вам нужно добраться до его подножия. Точной карты местности у вас нет. Но у вас есть фонарь. Что вы будете делать? Пожалуй, вы воспользуетесь фонарем и осмотритесь вокруг себя. Света фонаря не хватит для дальнего обзора, и вы наверняка не сможете осмотреть весь ландшафт целиком. Но вы сможете увидеть, по какому участку проще всего начать спуск к подножию холма, и сделаете несколько небольших шагов в этом направлении. Действуя подобным образом, вы будете медленно, шаг за шагом, продвигаться вниз. Математическая версия этого подхода называется методом градиентного спуска. После того как вы сделали шаг в выбранном направлении, вы вновь осматриваетесь, чтобы увидеть, какой путь ведет вас к цели, и делаете очередной шаг в этом направлении. Вы продолжаете действовать точно так же до тех пор, пока благополучно не спуститесь к подножию холма. А теперь представьте, что этим сложным ландшафтом является математическая функция. Метод градиентного спуска позволяет находить минимум, даже не располагая знаниями свойств этой функции

СЛАЙД 4

Итак, если функция вычисления весов для минимизации ошибки настолько сложна, что простого способа нахождения минимума ошибки алгебраическими методами не существует, то мы можем вместо этого применить метод градиентного спуска. Ясное дело, он может не дать нам точный ответ, поскольку мы приближаемся к ответу шаг за шагом, постепенно улучшая нашу позицию. Но это лучше, чем вообще не иметь никакого ответа. Во всяком случае, мы можем продолжить уточнение ответа еще более мелкими шагами по направлению к минимуму, пока не достигнем желаемой точности.
Рассмотрим использование метода градиентного спуска на простейшем примере.
На схеме приведен график простой функции $y = (x - 1)^2 + 1$. Если бы это была функция, описывающая ошибку, то мы должны были бы найти значение x , которое минимизирует эту функцию.
Итак, метод градиентного спуска — это действительно хороший способ нахождения минимума функции, и он прекрасно работает, когда функция настолько сложна, что ее обработка алгебраическими методами сопряжена с большими трудностями.

Снова подведем итоги:
• Ошибка нейронной сети является функцией весов внутренних связей.
• Улучшение нейронной сети означает уменьшение этой ошибки посредством изменения указанных весов.
• Простой подбор подходящих весов наталкивается на значительные трудности. Альтернативный подход заключается в постепенном улучшении весовых коэффициентов путем уменьшения функции

ошибки небольшими шагами. Каждый шаг совершается в направлении скорейшего спуска из текущей позиции. Этот подход называется градиентным спуском.

СЛАЙД 5

Рассмотрим теперь полный алгоритм обучения нейросети:

- подать на входной слой один из тренировочных наборов данных и определить значения выходов нейросети
- зная правильные значения выходов вычислить ошибку
- применив метод обратного распространения ошибки распределить ошибку между всеми узлами пропорционально их весам
- с помощью метода градиентного спуска скорректировать все веса нейронной сети и снова определить значения выходов нейросети
- если ошибка по прежнему существенна, то снова перейти к первому шагу.

Один такой цикл называется эпохой обучения.

СЛАЙД 6

А теперь обобщим информацию по всем трем урокам:
Любая нейросеть — это набор нейронов и связей между ними. Нейрон лучше всего представлять просто как функцию с множеством входов и одним выходом. Задача нейрона — взять числа со своих входов, выполнить над ними некую функцию и отдать результат на выход. Простой пример полезного нейрона: просуммировать все цифры со входов, и если их сумма больше некоторого порога — выдать на выход единицу, иначе — ноль.

Связи — это каналы, через которые нейроны шлют друг другу сигналы (фактически числа). У каждой связи есть свой вес — её единственный параметр, который можно условно представить как прочность связи. Когда через связь с весом 0.5 проходит число 10, оно превращается в 5. Сам нейрон не разбирается, что к нему пришло и суммирует всё подряд — вот веса и нужны, чтобы управлять на какие входы нейрон должен реагировать, а на какие нет.

СЛАЙД 7

Чтобы сеть не превратилась в анархию, нейроны решили связывать не как захочется, а по слоям. Внутри одного слоя нейроны никак не связаны, но соединены с нейронами следующего и предыдущего слоя. Данные в такой сети идут строго в одном направлении — от входов первого слоя к выходам последнего.

В реальном программировании, естественно, никаких нейронов и связей не пишут, всё представляют матрицами и считают матричными произведениями.

Когда мы построили сеть, наша задача правильно расставить веса, чтобы нейроны реагировали на нужные сигналы. Тут нужно вспомнить, что у нас же есть тренировочные данные — примеры «входов» и правильных «выходов». Например, наша нейросеть распознает рукописные цифры. Будем показывать нейросети рисунок цифры 9 и говорить «подстрой свои веса так, чтобы на твоём выходе при таком входе всегда загоралась девятка».

СЛАЙД 8
Сначала все веса просто расставлены случайно, мы показываем сети цифру, она выдаёт какой-то случайный ответ (весов-то нет), а мы сравниваем, насколько результат отличается от нужного нам. Затем идём по сети в обратном направлении от выходов ко входам (обратное распространение ошибки) и говорим каждому нейрону — так, ты вот тут зачем-то активировался, из-за тебя всё пошло не так, давай ты будешь чуть меньше реагировать на вот эту связь и чуть больше на вон ту, ок?

Через тысяч сто таких циклов «прогнали-проверили-подогнали» есть надежда, что веса в сети откорректируются так, как мы хотели.

На этом мы заканчиваем с вводными уроками по нейросетям и на следующем уроке попробуем начать разбираться с компьютерным зрением...