МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение

высшего образования

«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ имени В. И. ВЕРНАДСКОГО»

(ФГАОУ ВО «КФУ им. В. И. Вернадского»)

Таврическая академия (структурное подразделение)

Факультет математики и информатики

Кафедра информатики

Мазинова Эльзара Эскандеровна

**Применение нейросетевых технологий в задачах прогнозирования землетрясений.**

Курсовой проект

обучающейся 3 курса

направления подготовки 01.03.02. Прикладная математика и информатика

форма обучения очная

Научный руководитель

доцент кафедры информатики,

кандидат физико-математических наук А.С.Анафиев

Оценка руководителя: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Симферополь, 2020

Оглавление

Введение

1. Что такое нейросеть.

1.1. Ниша нейросети.

1.2. История возникновения нейронных сетей.

1.3. Биологический нейрон и искусственный. Перцептрон.

2. Компоненты нейронной сети.

2.1. Как работает нейронная сеть. Понятие линейной регрессии.

2.2. Какими бывают искусственные нейроны и их функции активации.

2.3. Парадигмы обучения.

3. Прогнозирование землетрясений.

3.1. Причины возникновения землетрясений.

3.2. История прогнозирования землетрясений.

3.3. Создание нейронной сети.

Вывод

Список использованной литературы

Приложения

Введение

Нейронные сети применяются в разных областях деятельности человека: идентификация речи, голоса, изображения, в задачах распознавания и классификации, прогнозировании событий, а также контроль и автоматизация повседневных задач. Принцип действия нейросети, основанный на примере работы человеческого мозга, понятен: получаем информацию, анализируем данные и выдаем результат. Целью данной курсовой работы является рассмотрение основ нейронной сети, определить какие входные данные нужно использовать, оценить возможность применения нейросетевых технологий в задачах прогнозирования землетрясений и их работу, производительность. На основе полученных результатов сделать выводы.

1.Что такое нейросеть

Нейронная сеть – это набор алгоритмов, который находит соотношения между входными данными, основываясь на работе биологического нейрона. Главным преимуществом нейронной сети является способность к обучению и отказоустойчивость (fault tolerance), а итоговым результатом-сгенерированные и взаимосвязанные входные данные.

* 1. Ниша нейросети

Нейронная сеть является частью искусственного интеллекта, который ручается за их продвижение и создание. Существуют различные определения понятия «искусственный интеллект». В широком понимании, это обучение машин: учится, рассуждать, делать выводы, работать и анализировать, так же, как это делает человек. Данное обучение реализуется с помощью алгоритмов машинного обучения.

Основная идея алгоритма, основанного на машинном обучении – найти шаблоны в полученных данных, используя статистику, для того, чтобы предсказывать следующие (будущие) результаты. Сейчас описанный метод очень распространен в социальных сетях, поисковиках и других приложений, связанных с персонализацией данных. Получая данные об активности и действий пользователя (просмотренных видео, прочитанных статей), алгоритмы машинного обучения предугадывают ваше дальнейшее поведение (рекомендации, советы, предполагаемые ресурсы, которые возможно понравятся юзеру).

Далее следуют нейронные сети. Процесс работы нейросети описывается так: получаем на вход данные, умножаем на веса (более значимые данные имеют больший вес, а менее – меньший), суммируем полученные произведения и через функцию активации выводим ответ (расписан принцип работы модели искусственного нейрона). На основе примера данного алгоритма можем определять или предугадывать результаты. Например, определять на картинке кошку или собаку, предугадывать ценовые показатели на фондовом рынке.

Также неотъемлемой частью нейронных сетей является глубокое обучение. Оно позволяет находить немногочисленные шаблоны, благодаря своей технике обработки информации, состоящей из множества слоев. Итог: обрабатываем информацию и получаем спрогнозированный результат [2].

1.2 История возникновения нейронных сетей.

В 1943 году Уоррен Мак-Каллок и Уолтер Питтс представили модель нейронной сети, основанную на теории деятельности головного мозга и математических алгоритмов. Созданные в модели пороговые переключатели (для проводимости информации) показали, что такая сеть может высчитывать логические и арифметические функции.

Важным открытием, на заре создания нейронных сетей, было правило Хэбба. Оно означало, что связь между двумя нейронами может быть усилена, если они оба активны или возбуждены. Алгоритм, построенный на данном правиле, описывал базовую процедуру обучения нейросетей.

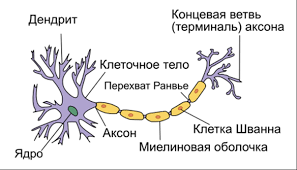
В попытках структурировать и закодировать работу мозга в 1958-1960 годах Фрэнк Розенблатт создает персептрон (математическая модель работы мозга), а в дальнейшем первый нейрокомпьютер «Марк-1», основанный на перцептроне. Данное электронное устройство могло распознавать цифры, используя графические сенсоры, напоминающие камеры(20 на 20 пикселей). Однако проделанная работа данной машины не всегда была точной. Если частично закрывать картинку или использовать другой размер изображения, то получим неверный ответ. Несмотря на это, работа Розенблатта получила восхищения и завышенные ожидания (в газете New York Times говорилось о том, что вооруженные силы США предполагали, что модель перцептрона поможет создать новый компьютер, который сможет мыслить, говорить, ходить и т.д.) [3].

Ажиотаж пошел на спад в 1969 году после публикации работы Марвина Минского и Сеймура Пейперта «Перцептроны». В ней говорилось, что перцепторон не способен решить все проблемы (операция XOR не выполнима, не способен выполнить вычислительный процесс с большим количеством данных). К тому же, Фрэнк Розенблатт погибает в кораблекрушении в 1971 году, следовательно не может отстоять свои идеи. Данные события приводят к спаду исследований и разработок .

С развитием производительности и мощности компьютеров стали появляться новые методы работы нейронных сетей: в 1975 Полом Вербосом был предложен метод обратного распространения ошибки (также данный метод был расписан Александром Галушниным независимо и одновременно с Полом Вербосом), который решил проблему операции XOR, в 1983 Кунихикой Фукусимой был разработан неокогнитон (многослойная нейронная сеть), который смог распознавать рукописные символы, в 1986 году Дэвид Румельхарт в своих научных трудах рассуждал о необходимости использовать алгоритм обратной распространении ошибки не только для второго слоя, но и для первого [4].

За последние 10 лет выявлен рост использования и применения нейронных сетей, разработка новых алгоритмов для получения точных результатов.

1.3. Биологический нейрон и перцептрон.

Рис.1. Строение нейрона

Нейрон – это клетка, а значит, в нем присутствуют все компоненты клетки, необходимые для ее жизнедеятельности(см. рис.1). Аксон – длинный отросток нейрона, способен передавать информацию от одного нейрона к другому. Дендриты – короткие отростки нейрона, получают информацию благодаря соединениям (синапсам).

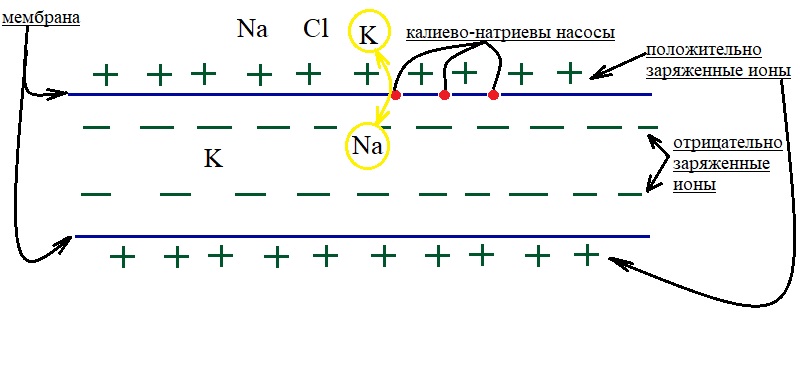
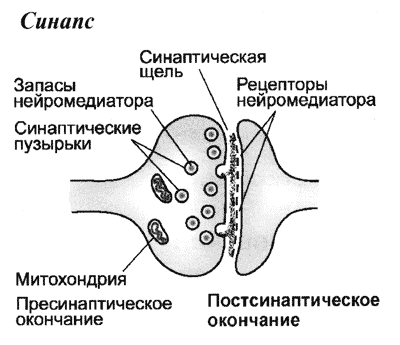


Рис.2 Передача сигналов в нейроне

Передача сигналов в биологическом нейроне в основном основана на химических реакциях (см. рис. 2). Вокруг мембраны возникает разность потенциалов. Внутри клетки в основном заряжено отрицательно (внутри присутствуют больше ионов калия), а снаружи – положительно (снаружи в основном присутствуют ионы натрия и хлора). Постоянно работает калиево-натриевый насос, который в состоянии покоя выводит из клетки ионы натрия и вводит ионы калия (первое условие), а в состоянии взаимодействия работает в обратном направлении. Вторым условием состоянием покоя нейрона является диффузия, которая старается выровнять дисбаланс: ионы калия пытаются выйти снаружи, а ионы натрия попасть внутрь. Третье условие – электростатические силы, которые выравнивают потенциал внутри и снаружи клетки. Если все три условия взаимодействуют, то возникает динамическое равновесие.

При изменении состояния клетки равновесие нарушается: поступают в клетку ионы натрия и выходят ионы калия; в дальнейшем приток натрия блокируется, а калий продолжает выходить; внутри клетка становится положительно заряженной, изменяя нижнюю часть мембраны и способствуя продвижению импульса. Поскольку все действия выполняются довольно медленно, то чтобы увеличить скорость в аксонах присутствует миелиновая оболочка, препятствующая оттоку и притоку калия и натрия [6].

Рис.3 Строение синапса

При передаче информации от одного нейрона к другому происходит по иному процессу. Два нейрона, для передачи импульсов, разделены щелью – синапсом. При воспроизведении сигнала в синаптическую щель выделяется нейромедиатор (класс веществ, который провоцирует каналы, находящиеся в следующей клетке и чувствительные к нейромедиатору, открыться) и происходит деполяризация (уменьшение разности потенциалов). Не всегда данная деполяризация может запустить действие для следующей клетки. Активация может произойти: если запущены несколько синапсов одновременно или только один синапс, работающий постоянно с малым промежутком времени. Существуют активирующие синапсы (нейромедиатор открывает натриевые каналы, клетка внутри становится положительно заряженной и происходит возбуждение) и тормозные (нейромедиатор выделяет хлорные каналы, клетка внутри становится отрицательно заряженной и вызвать дальнейшее действие клетки становится невозможным) [5].

Искусственный нейрон можно описать следующим образом.

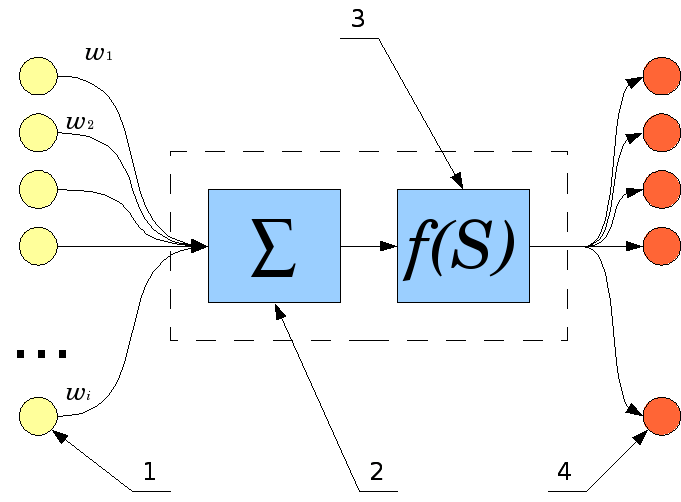


Рис.4. Математическая модель работы искусственного нейрона

Составляющая часть: жёлтые круги обозначают входы (1), а символы … – веса, сумматорная функция (2), функция активации (3), выходы (на изображении – оранжевые круги,4). Входы обозначают входящую информацию, веса – показатель «значимости» единицы входной информации, сумматорная функция соединяет поступающую информацию с учетом их веса (сумма произведений входов на их веса), далее результат передается в активационную функцию (нормализация итога, полученного сумматором), которая впоследствии выдает выходные сигналы.

Практической реализацией искусственного нейрона является перцептрон. Перцептрон описывается формулой :

Где – вектор весов, – вектор входных данных, – смещение, – выходная активация перцептрона. Процесс обучения перцептрона заключается в следующем: есть входные данные и соответствующие ответы, но первоначальные веса определены случайно, поэтому при первой попытке вероятность получения правильного ответа минимальна. Значит нужно изменить веса так, чтобы получить верный результат.

Для подробного объяснения буду использовать пример. Допустим, хотим выбрать подушку, идеальный для нас вариант – квадратная (не особо значимый показатель) и мягкая внутри (важный показатель). Даны 3 подушки: две с подходящими коэффициентами и одной, которая не подходит. Нужно найти веса и предугадать подходит ли нам следующая подушка с другими коэффициентами [6]. Приведен код на языке Python (см. листинг1).

import numpy as np

import math

pr=np.array([[1,1,0.8], [1,0.3, 0.9], [1,0.2, 0.5]])

w=np.array([0.0, 0.0, 0.0])

def target(x):

if x[1]==1 and x[2]==0.8:

return 1

elif x[1]==0.3 and x[2]==0.9:

return 1

elif x[1]==0.2 and x[2]==0.5:

return 0

def predict(x):

sum=w.dot(x)

if sum>0:

return 1

else:

return 0

perfect=False

while not perfect:

perfect=True

for p in pr:

if predict(p)!=target(p):

perfect=False

if predict(p)==0:

w+=p

else:

w-=p

print(w)

q=np.array([1.0, 0.4, 0.9])

print(predict(q))

Листинг 1. Алгоритм работы искусственного нейрона на примере с подушками.

Объяснение принципа работы листинга 1:

pr=np.array([[1,1,0.8], [1,0.3, 0.9], [1,0.2, 0.5]]) - к каждому вектору примеру первым элементом добавили единицу, которая потом будет умножаться на первый элемент весов – смещение.

В функции target представлены набор данных и правильный ответ. В функции predict реализованы сумматорная и активационная функции. В основном коде представлен процесс обучения: пока не достигнем должного результата, будем сверять точные результаты с вычисленными. Если предсказанные значения не совпадают с определенными, рассматриваем подсчитанное значение и меняем веса.

Результат:

Рис.5. Результат работы

2. Компоненты нейронной сети.

Основными составляющими нейронной сети являются нейроны и их веса (взвешенные соединения между ними). В дальнейшем в своей работе буду рассматривать многослойный перцептрон.

Данные поступают в нейроны, взаимодействуют с весами и в итоге приводят к активному состоянию или пассивному. Для того чтобы полученные значения проходили через нейроны, находящихся во внутренних узлах, существует функция распространения (propagation function). Например, существуют внутренние узлы i и j,– веса между данными узлами, – выходные значения узла i. Функция распространения получает значения и превращает их с учетом весов (скалярное перемножение вектора весов на вектор выходных значений) во входные параметры узла j (входные параметры – network input – результат работы функции распространения), чтобы в дальнейшем использовать полученный результат в функции активации. Пороговое значение (threshold value) показывает максимум градиента функции активации (в биологическом смысле это порог при котором нейрон начнет работать или активизироваться), однако сложно получить активационную функцию для того, чтобы натренировать пороговые значения. Поэтому пороговые значения могут быть представлены как взаимодействующие веса действующего нейрона, поэтому смещение (bias) имеет всегда значение 1. Смещение позволяет сдвигать активационную функцию, тем самым улучшая результаты предсказания. Например, используем сигмоидальную активационную функцию без смещения (изображение на следующей странице). Если будем изменять шаг, то это изменит только угол наклона самой функции. Однако если захотим, чтобы данная функция выводила 0, когда входное значение равно 3, нужно применять смещение [7].

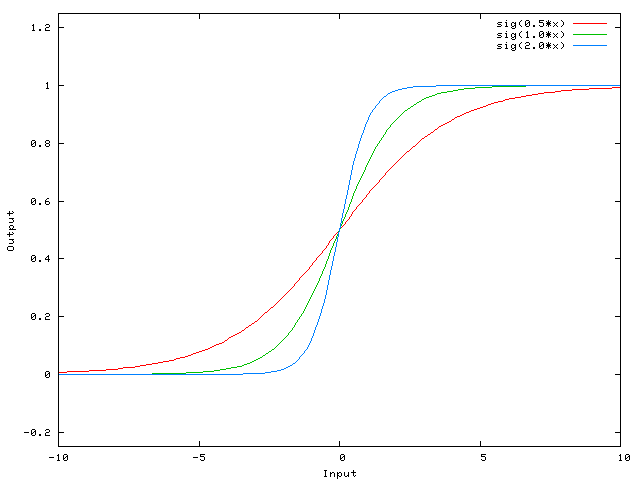


Рис.6. Активационная функция без смещения

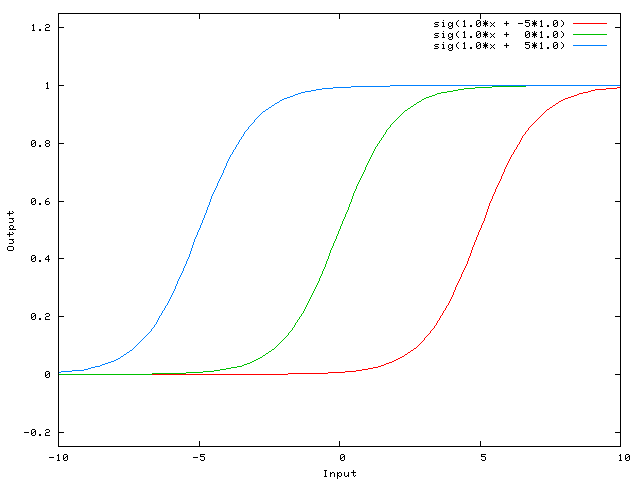


Рис.7. Активационная функция со смещением

Активация показывает степень активности нейрона. Функция активации (activation function, transfer function) «рассматривает» значение нейрона и определяет его как активный или пассивный. В итоге используя определенный обучающий алгоритм, получаем выходной сигнал.

2.1. Как работает нейронная сеть. Понятие линейной регрессии.

Работа нейронной сети заключается в следующем: входные данные обрабатываются нейронами, через нейроны проходят сигналы, суммируются произведения веса и выходного значения нейрона (каждое соединение между нейронами содержит вес), дальнейшие вычисления производятся в активационной функции для соответствующего выходного сигнала.

Однако анализ данных и прогнозирование могут осуществлять не только нейросети, но и некоторые статистические методы. Среди них есть линейная регрессия.

Линейная регрессия – это модель зависимости между двумя переменными в соответствии с применением линейного уравнения к полученным наблюдениям. Одна переменная должна быть представлена как целевая (обязательно должна быть количественной), а другая, как зависимая. Обе переменные связаны линейно. Математическая модель имеет вид:

вектор коэффициентов, y – вектор значений целевой переменной, Х – матрица наблюдений, е – случайная ошибка. Главная цель – получить более точный результат, значит найти такое (вектор оценок коэффициентов), что сумма квадратов

где – предсказанное значение, – реальное значение (учитываю данную функцию квадратов как целевую, так как он часто «штрафует» за отклонения наблюдений, тем самым приближая предсказанные значения к фактическим).

Допустим, есть линейная модель

хотим получить хорошие предсказания на основе данного уравнения

следовательно, найти и . Значит,

где Е – любая функция содержащая в качестве переменных векторы оценок коэффициентов. Так как нужно найти экстремум (минимум), то найдем производные и приравняем к нулю:

При дальнейших вычислениях получаем, что

(4)

где и – средние значения. Данные формулы получены только для регрессии с одной зависимой переменой. Наиболее простой способ нахождения вектора – матричный:

(6)

Данная формула основывается на:

-разности квадратов, в матричном виде представима в виде формулы , где

-свойствах матрицы

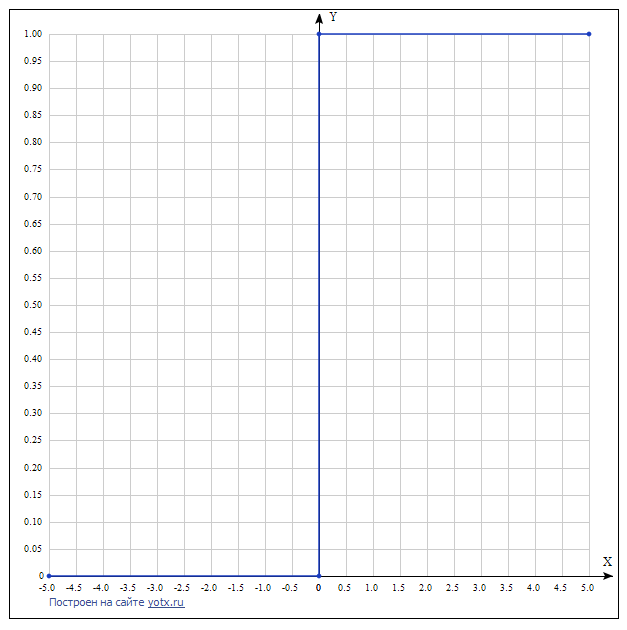
-выражении у через матрицу Н:

Рассмотренный статистический метод понятен и прост, но он не используется как модификация к нейронным сетям, потому что он основывается только на линейных входных данных (нейросеть в основном использует нелинейные данные). Несмотря на это, линейная регрессия, в общем, характеризует работу нейронной сети. Основное различие между нейросетью и линейной регрессией состоит в функции активации [6].

2.2. Какими бывают искусственные нейроны и их функции активации.

«Базовым» понятием искусственного нейрона является перцептрон. Математическое понятие перцептрона можно представить в виде:

График активационной функции:

Рис. 8. График перцептрона

Из графика видно, что функция имеет разрыв в нуле. Однако важна дифференцируемость функции и нелинейность (для нахождения оптимальных значений), а значит ее непрерывность. В виду этого, нельзя использовать активационную функцию перцептрона в задачах, в которых важна дифференцируемость функции. Помимо перцептрона существуют и другие искусственные нейроны, которые отличаются между собой активационными функциями.

Определение линейного нейрона можно записать следующим образом:

(8)

где – выходная активация нейрона, – вектор входных данных, – вектор весов, – смещение. У линейного нейрона несложная активационная функция - (всегда возвращает значение аргумента).

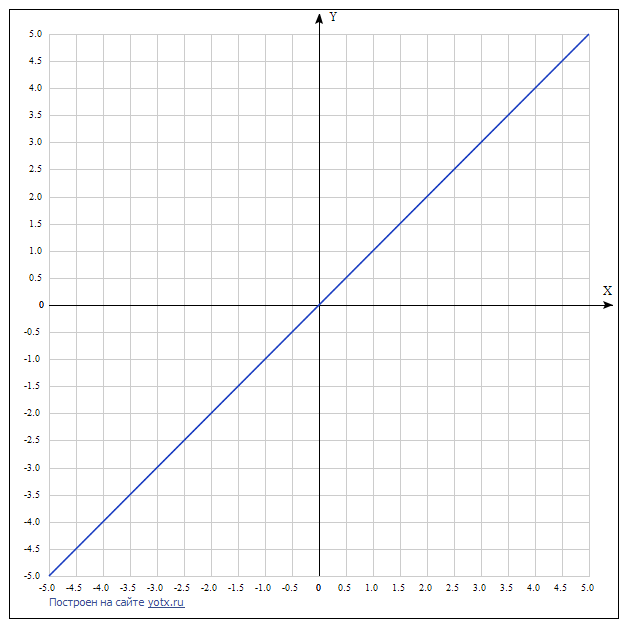


Рис. 9. График линейного нейрона

Определение нейрона можно упростить. Так как является скалярным произведением, то можно упростить:

(9)

Или представить другим способом:

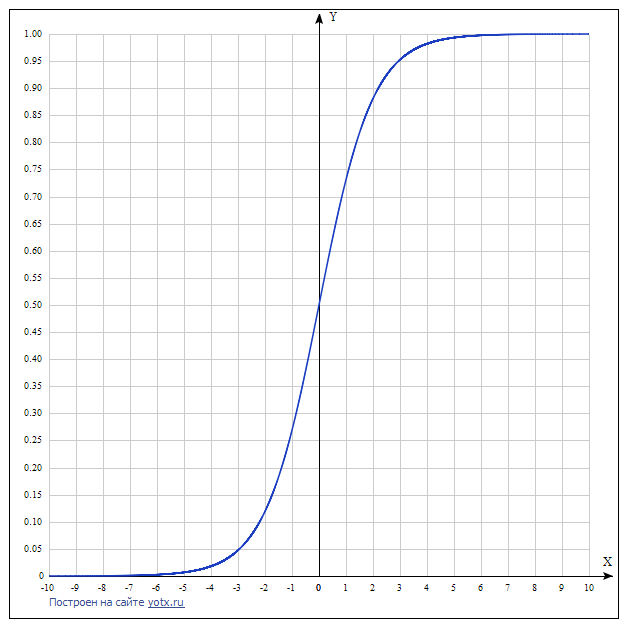
(10)

где – вектор входных данных с фиктивной колонкой (колонкой, где все значения равны 1), а – вектор весов со смещением. Своим алгоритмом действий линейный нейрон представляет работу линейной регрессии. Использование линейного нейрона в основном применяется для нахождения вещественных значений (например, определить вес человека по росту).

Одним из самых распространенных нейроном является сигмоид, который представим в виде:

где – логистическая функция, которая описывается следующим образом:

График функции:

Рис.10. График сигмоиды

Одним из свойств сигмоидальной функции является то, что мы не выбираем конкретно1 или 0, а получаем вероятность получения искомого значения.

Существует также искусственный нейрон основанный на гиперболическом тангенсе:

где – гиперболический тангенс от значения , представим в виде:

а также

Основываясь на последней формуле, можем предположить, что функция гиперболического тангенса является сдвинутой сигмоидальной функцией и имеет такие же свойства(дифференцируема, нелинейна). Одним из преимуществ является диапазон значений функции (-1; 1), что позволяет работать с данными, которые являются положительными, отрицательными или нулевыми. График активационной функции:

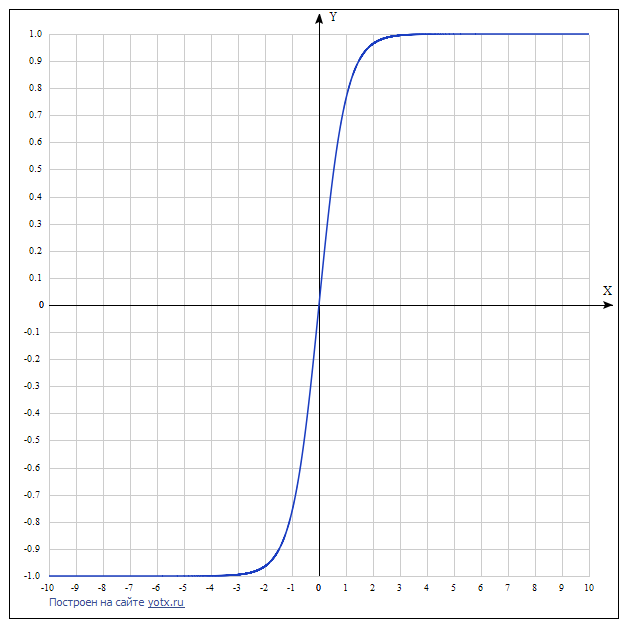
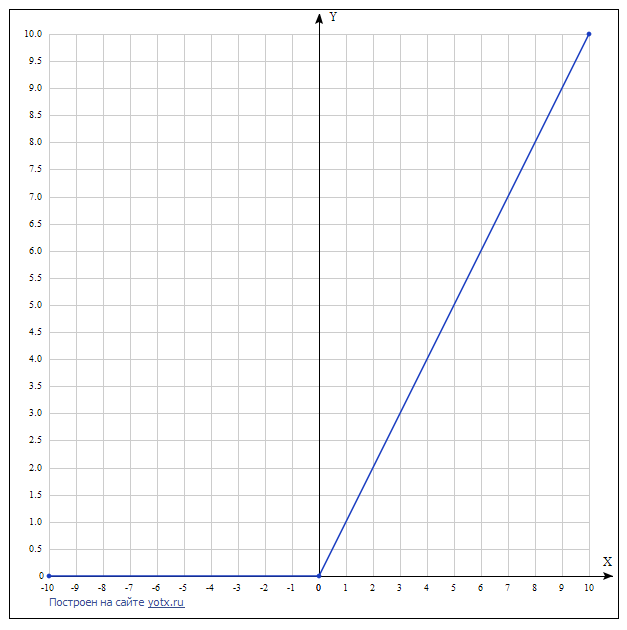


Рис.11.График гиперболического тангенса

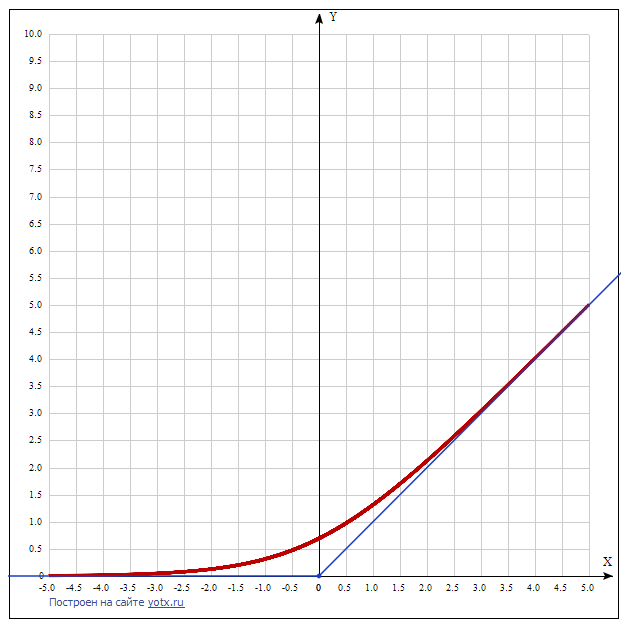
ReLU является одной из часто используемых функций. Расшифровывается как rectified linear unit. Можно представить в виде:

где – максимум из двух элементов. График функции:

Рис.12. График ReLU

Преимущества данной функции: активирует не все нейроны, а только положительные, быстрее находит значения (потому что в основе есть простая математическая функция ). Однако ReLU не дифференцируем в нуле (дифференцируемость функции полезна для обучения). Допустим, что есть функция , которая не дифференцируема, но хотим ее использовать, поэтому придумаем функцию , которая дифференцируема и по форме напоминает . Такой функцией для ReLU является softplus:

– softplus функция по определению. График (для наглядности – функция, обозначенная красным цветом – softplus, а синим – ReLU):

Рис.13. График softplus

Производная softplus равна сигмоидальной функции.

В основание всего написанного выше, получаем вывод, что данная активационная функция позволяет эффективней вычислять ReLU [8].

2.3. Парадигмы обучения.

Обучение нейронной сети состоит в том, чтобы получить нужное или правильно подсчитанное выходное значение, меняя значение весов. Основным компонентом обучения является обучающая выборка (training set). Существуют три подхода, которые соответствуют определенной обучающей задаче:

1) Обучение без учителя (unsupervised learning)

Даны только входные данные. Нейросеть пытается распознать схожие параметры и классифицировать их в подобные категории.

Ярким примером применения нейросети с обучением без учителя является самоорганизующаяся карта Кохонена. Самоорганизующаяся карта Кохонена – это нейронная сеть, которая обучается с использованием соревнующего обучения (competitive learning), сам состязательный процесс происходит до обучения. Вес элемента, который выиграл, изменяется в соответствии используемым законом обучения. Основа метода состоит в преобразовании многомерных измерений в одномерное или двумерное [9].

2) Обучение с учителем (supervised learning)

Дана обучающая выборка, которая состоит из входных данных и соответствующих правильных результатов. Таким образом, входные данные, поступающие в сеть, на выходе могут сравниться с правильным ответом. Веса будут меняться в зависимости от полученной разности. Такое обучение не часто встречается в реальных условиях, однако оно является эффективным и часто применимым.

3)Обучение с подкреплением (reinforcement learning)

В данном обучении сеть получает логическое или реальное значение. После окончания последовательных действий определяет полученный результат как хороший или плохой. Таким образом, мы даем оценку найденного нейросетью ответа (позволяем нейронной сети самой определить алгоритм для достижения нужного выходного значения) [3].

3. Прогнозирование землетрясений.

Землетрясения одни из больших бедствий на планете, которые происходят внезапно, наносят увечья людям и разрушают здания. За последние 10 лет исследователи пытались предугадать появление землетрясений с помощью выбросов в атмосферу радона, действий электромагнитных полей, изменение уровня подземных вод, поведение животных. Главная цель прогнозирования землетрясения найти где, когда и какого масштаба произойдет стихийное бедствие.

3.1. Причины возникновения землетрясений.

-вулканическая деятельность

Есть 2 типа вулканических землятресений: вулкано-тектонические землетрясения и длительные вулкано-тектонические землетрясения. Вулкано-тектонические землетрясения происходят из-за распространении разломов возле вулкана (вулканы часто возникают там, где существует слабая земная кора) и давлении в тех местах,где была магма (место освобождается и его нужно заполнить). В общем случае, они не связаны с магматической системой вулкана, а с деформацией поверхности в области слабых воздействий. Обычно, данные землетрясения имеют малую магнитуду и приводят только к деформации почвы. Длительные вулкано-тектонические землетрясения возникают благодаря вибрациям, полученных от движения магмы. Давление возрастает и горные породы начинают разрушаться, создавая небольшие землетряения.

-движение тектонических плит

Внешняя оболочка Земли (литосфера) разбита на океанические и континентальные плиты, которые движутся (скользят) по астеносфере (верхний слой мантии). Плиты постоянно в движении и в местах их взаимодействия происходят геологические процессы. Причиной движения тектонических плит является спрединг: в срединно-океанических хребтах есть рифтовая долина, куда постоянно поступает магма, образуя новое океаническое дно и раздвигая в стороны старые породы. Также объяснение движения плит связано с изменением полярности Земли ( за последние 80 миллионов лет Земля меняла свою полярность 170 раз). Базальтовые породы вблизи рифта намагничиваются в соответствии с полярностью магнитного поля Земли. Если полярность меняется, то и базальтовые породы намагничиваются в другом направлении. Существуют 4 типа сейсмических зон, которые возникают при тектонике плит. Первый тип землетрясений происходит в серединно-океанических хребтах. Литосфера в этом месторасположении тонкая и слабая, поэтому деформации не накапливаются, тем самым не провоцируя сильные землетрясения. Второй тип –малоглубинное событие (разломы). В данных разломах происходит трение между 2 плитами, в следствии чего, накапливается деформация, которая ослабевает после землетрясения. Третий тип – столкновение океанических и континентальных плит. Одна плита надвигается или подталкивается под другую и образуется впадина. Данный тип землетрясений может быть или слабым или сильным в зависимости от расположения «нижней» (на которую идет наслоение) плиты. Последний тип связан с горными хребтами, где происходит сжатие. Встречается вдоль границ континентальных плит (чаще землетрясения такого типа происходят в полосе горных хребтов от Бирмы до Средиземноморья).

-работы, проводимые в шахтах

Землетрясения такого типа называют землетрясениями краха (collapse earthquakes) или землетрясениями в шахтах. Происходят из-за возникновения давления, которое создается внутри горных пород. Это приводит к обрушению кровли и создание дальнейшим толчкам.

-ядерные взрывы

Землетрясения вызваны искусственно при воздействии взрыва большой мощности. Например, в 1930-х годах при испытании ядерных установок были спровоцированы землетрясения [10].

3.2. История прогнозирования землетрясений.

Первоначально предсказать землетрясение можно по скорости движения Земли, уровню грунтовых вод, давление и по поведению животных. Однако для прогноза на нейронной сети важны использования сейсмических значений.

Было совершено множество попыток предсказания землетрясений. Все они вели к той модели, которая сравнивает реальные землетрясения и предсказанную магнитуду на основании разных приближений (использования разных данных).

Одним из самых обсуждаемых попыток был метод «VAN», названный по инициалам ученных, которые работали над этим методом: Varotsos, Alexopoulos, Nomicos. Ученные пришли к выводу, что до землетрясения, происходят кратковременные изменения электрического поля Земли, известные как сейсмические электрические сигналы (Seismic Electric Signals, SES). Данные сигналы могут быть использованы для предсказаний магнитуды и места расположения сейсмического действия. Предсказание считалось успешным, если землетрясения произошло с разницей от несколько дней до несколько недель, 100 км от предсказанного эпицентра, и около 0.7 единиц измерения магнитуды по шкале Рихтера. Так были предсказаны более 60% землетрясений в Греции. Несмотря на это, многие исследователи считают, что успешные результаты получились случайно (нет определенной статистической проверки, не всегда можно получить SES за несколько километров от станции или при небольшом землетрясении, шум при получении сигнала и т.д.).

Нейросеть, входными данными которой были компоненты SES, была создана Lakkos S., Hadjiprocopis A., Comley R., Smith P. в 1994 году. Сеть использовала [Xerion](http://www.cs.toronto.edu/~xerion/) (симулятор нейронной сети) и обучающий алгоритм Delta-Bar-Delta (эвристический алгоритм, который изменяет степень обучения как тренировочный прогресс). Должна была предсказывать место (ширину и долготу) и магнитуду наближающегося землетрясения. На станции в Иоаннине (северо-западный регион Греции) были собраны данные, но их не было достаточно, поэтому их увеличили в пять раз добавлением небольшого количества гауссовского шума. Результаты работы данной нейронной сети были удовлетворительными. Ошибка предсказанного эпицентра землетрясения была меньше 0.3°, магнитуды – меньше 0.5 по шкале Рихтера.

Bodri B. в 2001 году пытался создать нейронную сеть на основе данных, полученных в Карпато-Паннонском регионе Венгрии и Пелопоннеско-Эгейский регион Греции. Создал трехслойный (входной, скрытый, выходной слои) многослойный персептрон с прямым распространением (three-layer feed-forward multilayer perceptron), используя метод обратного распространения ошибки (backpropagation error) как обучающий алгоритм, чтобы предсказывать время землетрясения с магнитудой больше 6.0 по шкале Рихтера. Входными значениями были количество землетрясений за единицу времени (били выбраны временные интервалы) с магнитудой больше 6.0 по шкале Рихтера. Несмотря на то, что были показаны хорошие результаты, такой набор данных был не приемлем из-за редких случаев землетрясения с большой магнитудой.

Lakshmi S. и Tiwari R. изучали временное развитие сейсмичности в центральных, западных и северо-восточных Гималаях. Модель многослойной нейронной сети с прямым распространением предсказывала месячные показатели частоты появления землетрясений. Обучающим алгоритмом был метод обратного распространения ошибки с использованием градиентного спуска и скользящим контролем (или кросс-валидацией, cross-validation) (объяснение данного метода: существует контрольная выборка и обучающаяся, использование кросс-валидации означает поиск средней ошибки на контрольных подвыборках). Входными данными были землетрясения, происходившие в данных регионах с 1960 по 2003 годах и магнитудой больше 4.0 по шкале Рихтера, выходные значения – частота появления землетрясений в следующем месяце. Согласно среднеквадратичной ошибке между полученными данными и реальными, полученные результаты были хорошими. Хорошие результаты были получены в западном и северо-восточном регионе, так как прогнозируемый коэффициент корреляции был выше чем у центрального региона. Вероятностная нейросеть(probabilistic neural network) Adeli H. и Panakkat A. предсказывала магнитуду наближающегося землетрясения в определенный временной промежуток (15 дней) в Южной Калифорнии. Входными данными были восемь математических параметров землетрясений (индикаторы землетрясений). Наиболее важную математическую составляющую данных параметров является закон Гутенберга-Рихтера :

– число землетрясений, – магнитуда, – параметр, который соответствует сейсмической активности, – угол наклона линейной части частично-магнитудного распределения, указывающий на соотношение больших и малых магнитуд. Индикаторы: время, в котором происходили землетрясения, наклон кривой обратного степенного закона Гутенберга-Рихтера, среднее квадратное отклонение линии регрессии основанного на обратном степенном законе Гутенберга-Рихтера, средняя магнитуда, дефицит магнитуды (magnitude deficit) – показатель разницы между наблюдаемой максимальной магнитудой и ожидаемой, которая основана на законе Гутенберга-Рихтера, корень квадратный сейсмической энергии, среднее время или период между данными событиями и коэффициент вариации среднего времени. Оценка предсказания проводилась по 3 параметрам: показатель ложного срабатывания, вероятность обнаружения и умение получить правильный результат. Согласно этим параметрам, нейросеть хорошо срабатывала для землетрясений с магнитудой от 4.5 до 6.0 по шкале Рихтера, однако не срабатывала для землетрясений с магнитудой больше 6.0 по шкале Рихтера [11].

3.3 Создание нейронной сети.

При создании нейронной сети ориентировалась на статью К.А.Пупкова и Као Динь Чонг «Приминение нейросетевых технологий в задачах прогнозирования землетрясения». Использованный язык программирования – Python. Входные данные взяты из произошедших землетрясений в северо-западном регионе Вьетнама и их значения состоят из матрицы, где строки обозначают интервалы времени (5 лет), а столбцы – значения сейсмических характеристик (, , , , .

Значение обозначает частоту повторения землетрясения. Считается по формуле:

где – момент происхождения первого землетрясения, – момент происхождения n-го землетрясения, n – количество землетрясений.

Скорость выделения энергии считается по формуле:

где – энергия землетрясения, высчитываемая по формуле Рихтера и измеримая в эргсах (единица работы и энергии):

Средняя величина мощности n землетрясений определяется формулой:

– мощность i-го землетрясения по магнитудам.

Значения a и b, взятые из уравнения Гуттенберга-Рихтера (20), можно вычислить:

где – число землетрясений, имеющих магнитуды .

Среднее отклонение η функции Гуттенберга-Рихтера вычисляется по формуле:

Значение ΔМ – это разность между значением , максимальной магнитуды наблюдения по функции Гуттенберга-Рихтера, и , полученной магнитудой в реальности:

В результате нейронная сеть имеет входной слой (6 нейронов), скрытый (5 нейронов), выходной ( 1 нейрон). На выходе получается предсказанная магнитуда землетрясения. Чтобы подробно разобраться в принципах работы нейросети, в своем коде создала класс NN (имитация работы нейронной сети). Применяемая активационная функция – сигмоида [1]. Обучение основывается на ошибке обратного распространения (использование градиентного спуска), так как хотим получить приближенные результаты. Основа данного метода заключается в нахождении минимума функции потерь, которую мы используем. Тренируем сеть – получаем результаты(см. листинг 2).

Ответ представлен на рисунке 14

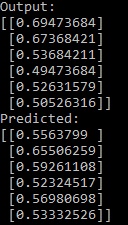


Рис. 14. Результат работы листинга 2

Опираясь на полученный результат, сделан вывод, что данный алгоритм не эффективен и необходима оптимизация. Есть примеры использования алгоритма Левенберга-Марквардта и создание нейронной сети на программе Matlab [1]. Критерий оптимизации алгоритма – среднеквадратичная ошибка, представлена в формуле (2.2). Основа алгоритма заключается в том, что дан набор данных (x,y), где x – свободная переменная, а y – зависимая, задана функция , нужно найти такой чтобы среднеквадратичная ошибка была минимальной. Изначально задается произвольный вектор , потом на каждом шаге итерации меняется на , где находится как:

где – матрица Якоби для функции , – единичная матрица, – параметр регуляризации [12]. При данных подсчетах, становится понятно, чтобы использовать данный алгоритм, нужно подключать библиотеки, такие как TensorFlow (матрица Якоби составлена из частных производных функций по всем переменным; входные данные представлены в виде набора значений, а не функций).

На основе предыдущих исследованиях, рассмотренных в разделе 3.2, можно оптимизировать листинг 2 использованием кросс-валидации или увеличить количество входных данных сейсмическими значениями до 8.

Вывод

В настоящее время нейронные сети имеют большую популярность, так как главным преимуществом является способность к обучению. История нейросети начинается с 60-х годов XIX века, за это время она прошла тернистый путь от расцвета до застоя и снова к всемирному применению. На данный момент, нейросети является разделом искусственного интеллекта, поэтому поиск шаблона, прогнозирование на основе полученных данных, анализ данных присуще не только нейронным сетям, но и машинному обучению, глубокому обучению. Принцип нейронной сети базируется на работе биологического нейрона. Одной из основных функций нейрона является передача информации. Однако в биологическом нейроне передача может ослабляться или усиливаться. Для этого в искусственном нейроне используется различные методы оптимизации ( в работе рассмотрены: поиск среднеквадратичной ошибки (2.2), метод градиентного спуска, кросс-валидация). Входными данными в искусственном нейроне являются числа, а не сигналы (как в биологическом), поэтому используются функции активации, которое определяет выходное значение нейрона. Так как выборка и цель использования нейронной сети различна, существуют три парадигмы обучения: с учителем, без учителя и обучение с подкреплением. Имеются статистические методы, выполняющие функции нейронных сетей, однако они базируются на линейных входных данных, а зачастую входные данные являются нелинейными.

Одной из областей применения нейронных сетей является прогнозирование землетрясений. Причина возникновения землетрясений различны, поэтому сложно предугадать землетрясения не используя сейсмических значений (например, предугадывать по поведению животных). В моем алгоритме (см. листинг 2) использованы 6 сейсмических значений: , , , , , которые вычисляются по формулам (21.1)-(21.7). Для обучения используется метод градиентного спуска. Полученные данные не были корректными, поэтому на основе проводимых исследований по прогнозированию землетрясений, следует улучшать обучения алгоритма ( например, алгоритм Левенберга-Марквардта) и создавать нейросеть с помощью библиотек на Python (например, TensorFlow) или увеличивать количество входных параметров. Основываясь на всей проведенной работе, можно сделать вывод, что создать нейросеть для прогнозирования землетрясений можно. Однако наиболее точные предсказания получить не возможно.

Список использованной литературы

1. Пупков К.А., Као Динь Чонг Применение нейросетевых технологий в задачах прогнозирования землетрясений // Вестник МГТУ им. Н.Э. Баумана. Серия «Приборостроение». – 2012 . – №2 (87). – С. 70 –78.

2. What is machine learning? // MIT Technology Review URL: https://www.technologyreview.com/2018/11/17/103781/what-is-machine-learning-we-drew-you-another-flowchart/ (дата обращения: 20.04.2020).

3. David Kriesel A Brief Introduction to Neural Networks. [Электронный ресурс]. – Электрон.книга. 2007. - 244 с. – Режим доступа: http://www.dkriesel.com, свободный.

4. История возникновения нейронных сетей // Интерфейс Ltd. URL: http://www.interface.ru/home.asp?artId=37476 (дата обращения: 20.04.2020).

5. Распространение нервных импульсов // Элементы большой науки URL: https://elementy.ru/trefil/21176/Rasprostranenie\_nervnykh\_impulsov (дата обращения: 22.04.2020).

6. ﻿Арсений Москвичев, Анастасия Миллер. Курс нейронные сети.URL: https://stepik.org/course/401?auth=registration (дата обращения:05.05.2020).

7. Нейронные сети для начинающих. Часть 2 // Хабр URL: https://habr.com/ru/post/313216/ (дата обращения: 04.05.2020).

8. Secret Sauce behind the beauty of Deep Learning: Beginners guide to Activation Functions // Towards Data Science URL: https://towardsdatascience.com/secret-sauce-behind-the-beauty-of-deep-learning-beginners-guide-to-activation-functions-a8e23a57d046 (дата обращения: 28.04.2020).

9. Kohonen Self- Organizing Feature Map // Javatpoint URL: https://www.javatpoint.com/artificial-neural-network-kohonen-self-rganizing-feature-map (дата обращения: 01.05.2020).

10. Causes of earthquakes // Academia URL: https://www.academia.edu/5589095/Causes\_of\_earthquakes (дата обращения: 02.05.2020).

11. Maria Moustra, Marios Avraamides, Chris Christodoulou Artificial neural networks for earthquake prediction using time series magnitude data or Seismic Electric Signals // Expert Systems with Applications. - 2011. - №38(12). - С. 15032-15039.

12. Алгоритм Левенберга-Марквардта // MachineLearning.ru URL: http://www.machinelearning.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC\_%D0%9B%D0%B5%D0%B2%D0%B5%D0%BD%D0%B1%D0%B5%D1%80%D0%B3%D0%B0-%D0%9C%D0%B0%D1%80%D0%BA%D0%B2%D0%B0%D1%80%D0%B4%D1%82%D0%B0 (дата обращения: 07.05.2020).

Приложения

Применение нейросети для прогнозирования землетрясения

import numpy as np

#X-входные параметры, Y-выходной сигнал

X = np.array(([1716,12158100,0.72819,0.00676,4.01111,0.34626],

[1358,121972200,0.79910,0.00779,3.93152,0.78071],

[1484,22481790,0.69068,0.00135,4.03929,0.74227],

[1674,18781300,0.78829,0.01469,4.16528,0.42182],

[1555,11511200,0.97625,0.00336,3.92381,0.12019],

[1725,30112300,0.76267,0.01445,4.15556,0.41732]), dtype=float)

Y = np.array(([6.6], [6.4], [5.1], [4.7], [5.0], [4.8]), dtype=float)

#скаляризация параметров, так как числовые значения очень разняться

X = X/np.amax(X, axis=0) #находим максимум массива по столбцам и делим входные значения на полученный максимум

Y = Y/9.5 #максимальная магнитуда = 9.5

class NN():

# конструктор

def \_\_init\_\_(self):

# нейросеть состоит из входного слоя (6 нейронов, входных значений)

# скрытого слоя (5 нейронов)

# выходного слоя (1 нейрон, полученный результат)

# между 3 слоями создам 2 матрицы весов: от входа до скрытого слоя (w\_ih)

# от скрытого слоя до выходного значения (w\_ho)

self.w\_ih = np.random.randn(6, 5)

self.w\_ho = np.random.randn(5, 1)

# активационная функция, сигмоида

def sig(self,x):

return 1/(1+np.exp(-x))

# производная активационной функции

def deriv\_sig(self,x):

return x\*(1-x)

# feedforward, прямое распространение

def feedforward(self,X):

# np.dot(X,self.w\_ih) - перемножаю матрицу входных значений и матрицу весов

# self.sig(np.dot(X,self.w\_ih)) - для полученного результата применяю активационную функцию

# далее произвожу аналогичные действия для второй матрицы весов (w\_ho)

self.one=self.sig(np.dot(X,self.w\_ih))

self.two=self.sig(np.dot(self.one,self.w\_ho))

return self.two

# обратное распространение ошибки на основе приминения метода градиентного спуска

def backprop(self,X,Y,pred):

# (y-pred) - разница между предсказанным значением и реальным, полученная ошибка

# (y-pred)\*self.deriv\_sig(pred) - производная сигмоиды, умноженная на полученную ошибку

# так поступаем, так как хотим понять как веса влияют на данные и определить скорость изменения подсчитанных данных

self.a1=(Y-pred)\*self.deriv\_sig(pred)

# хотим узнать как скрытый слой повлиял на полученную ошибку

self.hidlay1=self.a1.dot(self.w\_ho.T)

self.hidlay2=self.hidlay1\*self.deriv\_sig(self.one)

# изменяем веса

self.w\_ih+=X.T.dot(self.hidlay2)

self.w\_ho+=self.one.T.dot(self.a1)

nn=NN()

for i in range(1000):

print ("Output:")

print(Y)

print ("Predicted:")

print(nn.feedforward(X))

pred=nn.feedforward(X)

nn.backprop(X,Y,pred)

Листинг 2. Создание нейросети для прогнозирования землетрясения