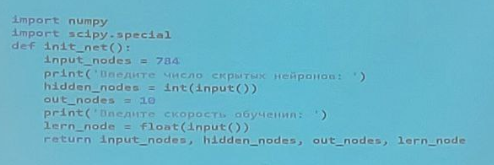
**Пример сети**



Теперь создадим саму сеть.

Мы создадим все связи между слоями, если точнее - силу этих связей.

Для этого мы будем использовать матрицы которые будут хранить эти связи.

Каждый нейрон связан с каждым нейроном следующего слоя (например если в скрытом слое 100 нейронов, то для каждого нейрона входного слоя будет иметься 1x100 связей и т.д.)

Нейроны входного слоя просто передают информацию поступившую на них (для них не используется функция активации) нейрона следующего слоя по связям.

Далее мы эти данные **умножаем на вес связи** и они поступают в нейрон следующего слоя.

Нейрон используя функцию активации обрабатывает эти данные.

Мы будем использовать **Сигмоиду**. И передает дальше.

def fuc\_active(x):

return scipy.special.expit(x)

У нас есть каркас.

Теперь нам надо ее обучить **распознавать рукописный текст.**

Первым делом для этого нужны входные данные( много ) а так же проверочные выходные данные с которыми мы будем сверяться в ходе обучения.

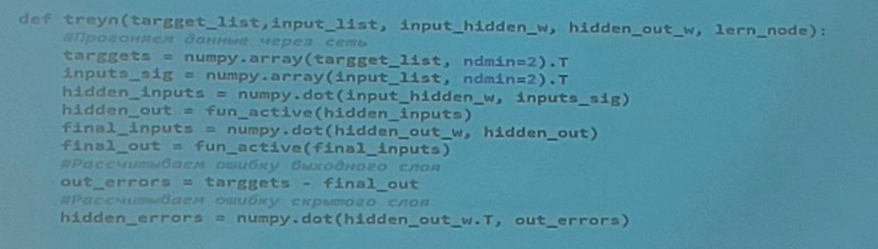
Мы возьмем готовую базу **MINIST** - объёмная база данных образцов рукописного написания цифр.

База данных является стандартом, предложенным Национальным институтом стандартов и технологий США с целью сопоставления методов распознавания изображений с помощью машинного обучения в первую очередь на основе нейронных сетей.

Смысл обучения нейронной сети сводится к коррекции **W** весов связей между нейронами.

Первым делом нужно вычислить ошибку выходных данных.

Чтобы посчитать ошибку для нейронов скрытого слоя применяется метод **обратного распространения ошибки.**

****

Для изменения значений весов используется **метод Градиентного спуска.**

Обучающие данные.

**mnist\_train.csv**

**mnist\_test.csv**

Данные представлены в удобной текстовой форме, разделённые запятой.

Тренировочный набор содержит около **60.000** промаркированных образцов, тестовый поменьше

Около **10.000.**

Первое значение это маркер от **0-9,** какая цифра изображена.

Далее следует пиксельный массив **28х28.**

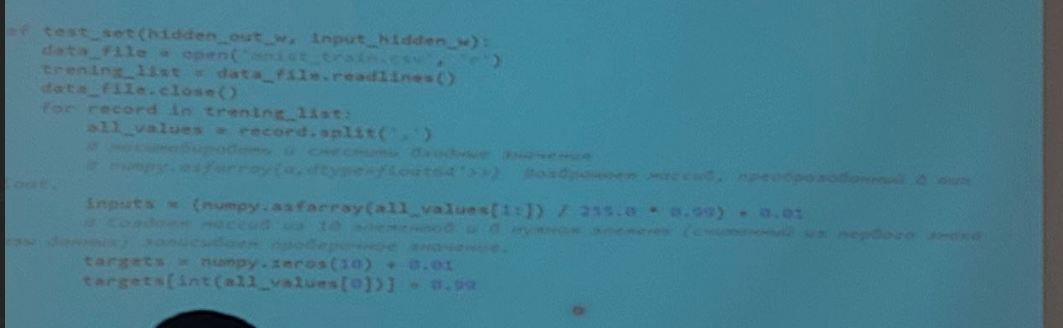
Всего **784** значения от **0 до 255.**

Прежде чем использовать эти данные для обучения и проверки нейронной сети, необходимо подготовить их для работы с функцией активации.

Чтобы данные оставались в оптимальном диапазоне для неё.

Для нас будет оптимально значение **от 0,01 до 1.00.**

Почему не 0 ? Нулевое значение может помешать нам обновлять весовые коэффициенты .



Наша нейросеть будет классифицировать изображения по 10 признакам, цифра от 0 до 9 .

Выходные значения опять же будут в рамках нашей функции активации, фактически сигмоида не может достичь 1 , по этому мы записываем 0.99.

Используя эту базу данных мы будем подавать на каждый входной нейрон одно значение пикселя картинки. То есть входной слой будет у нас матрица **784 нейрона.**

А выходной слой будет содержать **10 нейронов**- **классификаторов**.

Самый важный опрос : сколько нейронов должно быть в скрытом слое?

**Библиотеки глубокого изучения**

**TensorFlow**

Базовый язык **C++,** но имеет API для **Python**.

Разработан **Google**.

Вычисления с использованием графов потоков данных.

Предлагает мощные средства мониторинга процесса обучения моделей и визуализации.

Поддерживает **распределенное обучение**.

Имеет достаточно высокий входной порог

**theano**

Базовый язык **Python**.

Разработан университетом **Монреаля**.

Вычисления с использованием графов потоков данных Эффективная обработка тензоров.

Вычисления выражаются **NumPy** – подобным синтаксисом.

**K**

Библиотека верхнего уровня.

Поддерживает Python.

В качестве вычислительного backend используется TensorFlow или Theano.

Позволяет создавать и обучать нс на очень высоком уровне.

**Microsoft CNTK**

Базовый язык **C++.**

Поддерживает API для **C++, C#, Python, Java**.

Разработана **Microsoft**.

Обеспечивает скорость обучения, сравнимую с **TensorFlow**, а на рекуррентных сетях превосходит его.

Имеет API разных уровней абстракции.

**Caffe**

Имеет небольшую скорост в сравнении с другими библиотеками

**Torch**

**НЕЙРОСЕТИ ДЛЯ РАСПОЗНАВАНИЯ ЭЛЕМЕНТОВ ОДЕЖДЫ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕК ГЛУБОКОГО ОБУЧЕНИЯ**

**Постановка задачи**

**Дано**: ряд различных изображений с предметами одежды

**Должны** **получить**: к какому классу относится данная картинка

Нужно разработать нейронную сеть которая принимает на вход изображение и выдает на выходе информацио к какому классу относится данное изображение

**Исходные данные**

Для обучения будем использовать набор данных Fashion MINIST

Это набор достаточно прост, поэтому с ним легко работать

Это задача классификации

На входе нейронной сети изображение, на выходе класс

Открытый набор данных

* Github.com/zalandoresearch/fashion-mnist
* **60 000** изображений предметов одежды

2 файла

* Файл с изображениями
* Файл с метками классов

Формат изображений

* Размер **28х28**
* Оттенки серого**[0;255px]**
* Изображения в бинарном виде записаны в один файл

Входные значения сети

* Интенсивность пикселей
* Количество значений: **784**

Архитектура сети

* https://en.wikipedia.org/wiki/MNIST database

Формат изображений

* **800 нейронов**

Выходной слой

* **10 нейронов**
* Вероятность того, что на изображении данный предмет одежды

