Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт по лабораторной работе №4 по дисциплине "Основы искусственного интеллекта"

Тема Основы ИНС

Студент Варламова Е. А.

Группа ИУ7-13М

Оценка (баллы) _____

Преподаватели Строганов Ю.В.

Москва — 2023 г.

СОДЕРЖАНИЕ

Введение

Нейронные сети – это вычислительные системы, которые способны обучаться на основе данных, распознавать сложные паттерны, и выполнять разнообразные задачи, включая классификацию, регрессию, обработку изображений, обработку естественного языка и другие задачи машинного обучения.

Целью данной лабораторной работы является создание программы, которая классифицирует данные из датасета MNIST [1] с использованием нейросетевого подхода с функциией активации ReLU и функцией потерь Cross Entropy.

Для этого необходимо решить следующие задачи:

— привести постановку задачи;

— описать нейронную сеть: функцию активации ReLU и функцию потерь Cross Entropy;

— рассчитать аналитически необходимый размер обучающей выборки по неравенству Чебышёва, необходимое для гарантированного успешного выполнения поставленной задачи;

— привести схему нейронной сети;

— привести особенности реализации ПО, решающего поставленную задачу;

— провести исследование, состоящее в определении состояний недообучения и переобучения для различного соотношения (% и %) обучающей и тестовой выборок, а также для различного количества скрытых слоёв нейронной сети.

1 Аналитическая часть

1.1 Постановка задачи

1. Разработать ПО, которое классифицирует данные из датасета MNIST [1] с использованием нейросетевого подхода с функциией активации ReLU и функцией потерь Cross Entropy.

2. Определить состояния переобучения и недообучения для

   (a) различного соотношения (% и %) обучающей и тестовой выборок:
   - 10 и 90;
   - 20 и 80;
   - 30 и 70;
   - 40 и 60;
   - 50 и 50;
   - 60 и 40;
   - 70 и 30;
   - 80 и 20;
   - 90 и 10;

   (b) различного количества скрытых слоёв нейронной сети:
   - 0;
   - 1;
   - 5;

3. рассчитать аналитически необходимый размер обучающей выборки по неравенству Чебышёва, необходимое для гарантированного успешного выполнения поставленной задачи.

1.2 Описание нейронной сети

Нейронные сети – это вычислительные системы, которые способны обучаться на основе данных, распознавать сложные паттерны, и выполнять разнообразные задачи, включая классификацию, регрессию, обработку изображений, обработку естественного языка и другие задачи машинного обучения. Нейронные сети состоят из множества связанных между собой элементов, называемых нейронами, которые имитируют функционирование нейронов в головном мозге.

1.2.1 Основные понятия нейронной сети

Перечислим основные понятия, исползуемые в нейросетях.

— Нейрон (или узел): Нейрон – это базовый элемент нейронной сети, имитирующий функцию биологического нейрона. Он принимает входные сигналы, обрабатывает их и генерирует выходной сигнал. Каждый нейрон обычно имеет несколько входов, каждый из которых соответствует весу (степени важности) исходного сигнала.

— Веса: Веса представляют собой параметры, которые определяют степень важности входной информации для каждого нейрона. Они отражают силу связей между нейронами и являются ключевыми для эффективного обучения нейронной сети.

— Функция активации: Функция активации определяет выходное значение нейрона на основе его взвешенного входа и возможно добавления смещения. Различные функции активации, такие как ReLU, сигмоида, tanh, используются для введения нелинейности в нейронные сети.

— Структура и связи: Нейронные сети могут быть организованы в различные архитектуры, такие как многослойные перцептроны, сверточные нейронные сети, рекуррентные нейронные сети и другие. Связи между нейронами формируют слои и определяют поток информации в сети.

— Функция потерь: Функция потерь измеряет разницу между предсказанными значениями модели и фактическими значениями. Во время обучения нейронная сеть минимизирует эту функцию, чтобы улучшить качество своих прогнозов.

— Оптимизатор: Оптимизатор используется для коррекции весов нейронов в ходе обучения, с целью минимизации функции потерь. Примером может служить алгоритм градиентного спуска.

— Слои: Нейронные сети состоят из различных слоев, таких как входной, скрытый и выходной слои. Каждый слой выполняет определенные вычисления и обработку данных.

— Обучающие данные и цели: Нейронные сети обучаются на основе обучающих данных и их соответствующих целей (или меток). Эти данные используются для корректировки весов нейронов в процессе обучения.

Для решения задачи классификации необходимо создать нейросеть. При этом должны быть решены следующие задачи:

1. определена структура нейросети: определено количество слоёв, определено количество нейронов в каждом слое, заданы функции активации;
2. проведено обучение нейросети на обучающей выборке;
3. проведена оценка нейросети на тестовой выборке (не пересекающейся с обучающей).

1.2.2 Обучение нейросети

Обучение нейросети состоит из нескольких эпох (итераций). В течение одной эпохи обучения нейронной сети происходит несколько этапов, которые повторяются для каждого обучающего примера:

1. прямое распространение:
   — входные данные подаются на входной слой нейронов;
   — данные передаются через скрытые слои, взвешиваются с использованием соответствующих весов и агрегируются;
   — агрегированные значения проходят через функции активации каждого нейрона в скрытых слоях и выходном слое, что приводит к формированию выходов сети;
2. оценка ошибки : вычисляется ошибка между выходами сети и ожидаемыми значениями (целевыми метками/метками классов);
3. обратное распространение ошибки:
   — ошибка распространяется обратно через сеть, начиная с последнего слоя и двигаясь к входному слою;
   — для каждого слоя вычисляется градиент функции потерь по весам и смещениям сети;

4. обновление весов, чтобы уменьшить ошибку модели: используя градиент ошибки, веса сети обновляются с использованием метода оптимизации, такого как стохастический градиентный спуск или его модификации;

Эти этапы повторяются для каждой эпохи обучения с тем, чтобы постепенно корректировать веса сети и уменьшать ошибку прогноза. Процесс обучения заключается в том, чтобы минимизировать ошибку модели и достичь желаемой производительности в решении конкретной задачи.

1.2.3 Функция активации ReLU

Функция ReLU – это популярная функция активации в нейронных сетях, которая обладает нелинейными свойствами. Ее формула определяется следующим образом:

$$f(x) = \max(0, x) \tag{1.1}$$

Таким образом, ReLU функция активации принимает следующие значения:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases} \tag{1.2}$$

ReLU функция подходит для использования в нейронных сетях по нескольким причинам, включая то, что она обеспечивает нелинейность (что важно для изучения сложных функций) и простоту вычисления.

1.2.4 Функция потерь Cross Entropy

Cross Entropy (кросс-энтропия) – это функция потерь, широко применяемая в задачах оптимизации нейронных сетей, особенно в задачах классификации.

Формула для бинарной кросс-энтропии (binary cross entropy) выглядит следующим образом:

$$H(p, q) = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \log(p(y_i)) + (1 - y_i) \log(1 - p(y_i)) \right] \tag{1.3}$$

где

— N – количество примеров в выборке

— y_i – фактическое значение для i-го примера (0 или 1 в задаче бинарной классификации)

— $p(y_i)$ – предсказанная вероятность для i-го примера

Для случая многоклассовой классификации формула кросс-энтропии выглядит немного иначе:

$$H(p,q) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

где

— N – количество примеров в выборке

— M – количество классов

— y_{ij} – бинарное индикаторное значение (1 или 0) для того, принадлежит ли пример i к классу j

— p_{ij} – предсказанная вероятность принадлежности примера i к классу j

При обучении нейронных сетей кросс-энтропия обычно минимизируется с помощью методов оптимизации, таких как градиентный спуск или его различные вариации.

1.3 Определение размера выборки аналитически

Определим необходимый для гарантированного обучения модели размер обучающей выборки аналитически, учитывая закон больших чисел, представленный в форме второго неравенства Чебышева.

$$P\left\{|X - MX| \geq \varepsilon\right\} \leq \frac{\sigma^2}{N\varepsilon^2}, \tag{1.4}$$

где

— N – размер выборки;

— X – случайная величина;

— σ^2 – дисперсия случайной величины; MX – математическое ожидание случайной величины;

— ε – требуемая точность.

Критерий качества выборки лежит в диапазоне от $[0; 1]$. Пусть:

— дисперсия оцениваемого показателя не превышает 0.1;

— достоверность $= 0.99$;

— точность = 0.01.

Тогда можно выразить размер выборки, требуемый для обучения модели:

$$N \geq \frac{0.1}{0.01^2 \cdot (1 - 0,99)}, \tag{1.5}$$

$$N \geq 100000. \tag{1.6}$$

Таким образом, можно заключить, что для обеспечения ошибки, не превышающей 1% с вероятностью 0.99, размер обучающей выборки должен быть не менее 100000 примеров.

Вывод

В данном разделе была приведена постановка задачи, описаны основные компоненты нейронных сетей, в том числе были описаны функция потерь Cross Entropy и функция активации ReLU. Кроме того, был рассчитан аналитически необходимый размер обучающей выборки по неравенству Чебышёва, необходимый для гарантированного успешного выполнения поставленной задачи.

2 | Конструкторская часть

2.1 Схема нейронной сети

В соотвествии с заданием, необходимо построить сети, состоящие из 0, 1 и 5 скрытых слоёев. В качестве функции активациии использовать функцию ReLU. Соответствующие схемы представлены на рисунках 2.2, 2.3 и 2.4.

Схема одного нейрона нейронной сети представлена на рисунке 2.1.

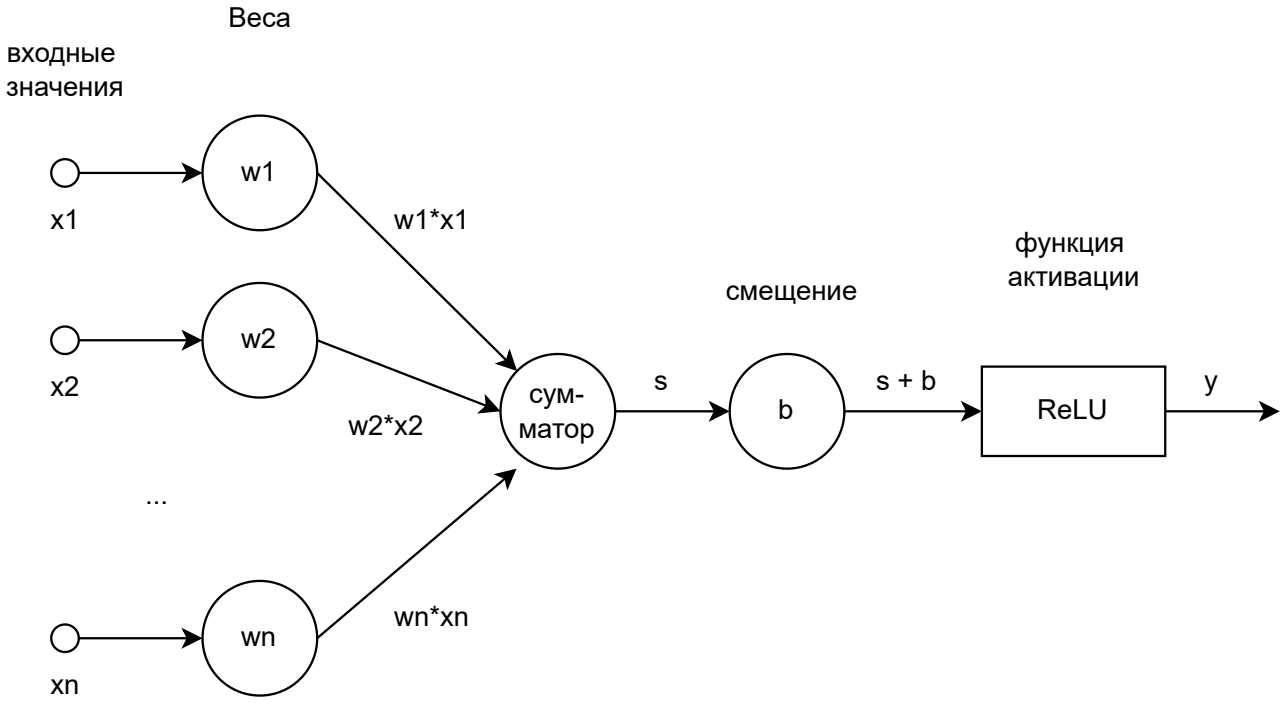

Рис. 2.1: Схема одного нейрона

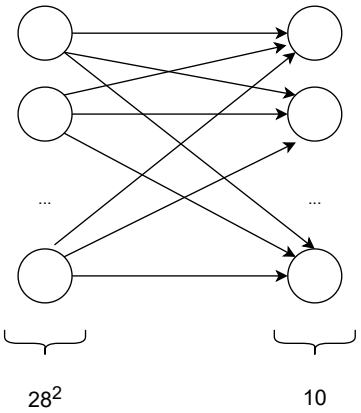

Рис. 2.2: Схема сети с 0 скрытых слоев

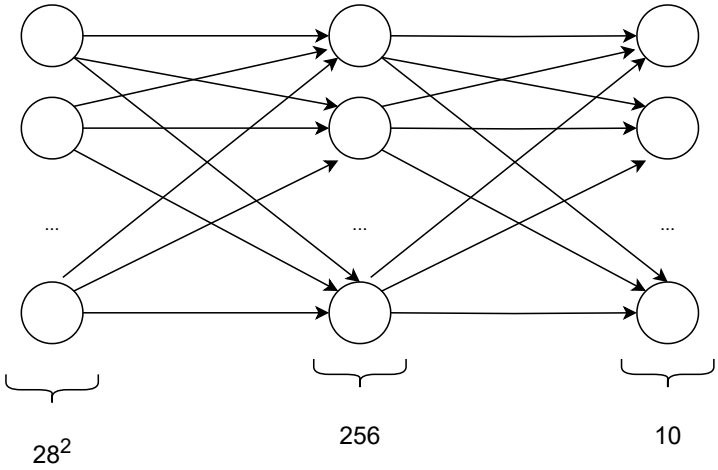

Рис. 2.3: Схема сети с 1 скрытым слоем

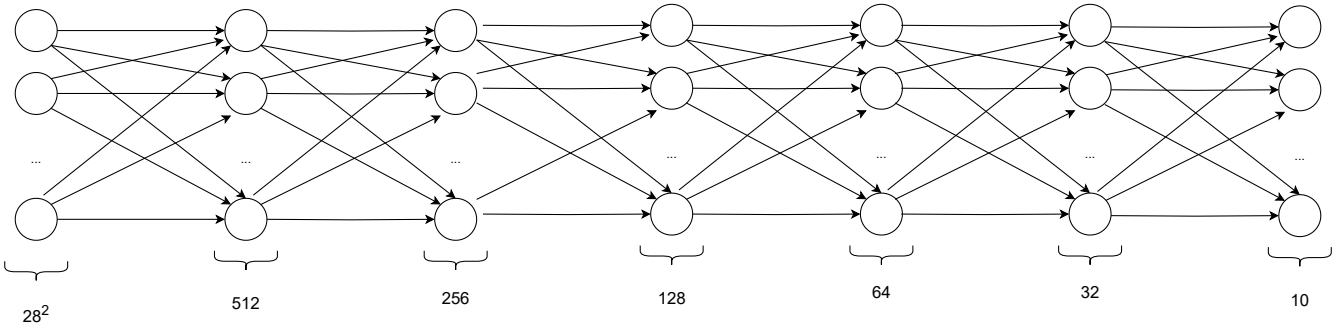

Рис. 2.4: Схема сети с 5 скрытыми слоями

Вывод

В данном разделе была приведена схема нейронной сети.

3 | Технологическая часть

3.1 Выбор средств разработки

В качестве языка программирования был использован язык Python, поскольку этот язык кроссплатформенный и для него разработано огромное количество библиотек и модулей, решающих разнообразные задачи.

В частности, имеются библиотеки, включающие в себя инструменты для создания нейронных сетй. В данной работе была использована библиотека «pytorch» [2] для разработки нейронных сетей.

3.2 Листинги ПО

В листинге 3.1 представлено определение структуры сетей с 0, 1 и 5 скрытыми слоями соответственно.

Листинг 3.1: Определение структуры сетей

```
class Net0(nn.Module):
    def __init__(self):
        super(Net0, self).__init__()
        self.fc1 = nn.Linear(28 * 28, 10, bias=True)
    def forward(self, x):
        x = F.relu(self.fc1(x))
        return x
class Net1(nn.Module):
    def __init__(self):
        super(Net1, self).__init__()
        self.fc1 = nn.Linear(28 * 28, 256, bias=True)
        self.fc2 = nn.Linear(256, 10, bias=True)
    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        return x
class Net5(nn.Module):
    def __init__(self, hl = 0):
        super(Net5, self).__init__()
        self.fc1 = nn.Linear(28 * 28, 512, bias=True)
        self.fc2 = nn.Linear(512, 256, bias=True)
```

```
        self.fc3 = nn.Linear(256, 128, bias=True)
        self.fc4 = nn.Linear(128, 64, bias=True)
        self.fc5 = nn.Linear(64, 32, bias=True)
        self.fc6 = nn.Linear(32, 10, bias=True)
    def forward(self, x):
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = F.relu(self.fc3(x))
        x = F.relu(self.fc4(x))
        x = F.relu(self.fc5(x))
        x = F.relu(self.fc6(x))
        return x
```

В листинге 3.2 представлен код обучения сетей и их оценки.

Листинг 3.2: Обучение сетей и их оценка

```
def CreateNN(net, train_size=0.1, batch_size=64, epochs=40):
    ## train
    train_loader, test_loader = get_data_loaders(train_size, batch_size)
    optimizer = optim.SGD(net.parameters(), lr=1e-5, momentum=0.9)
    criterion = nn.CrossEntropyLoss()
    for epoch in range(epochs):
        for batch_idx, (data, target) in enumerate(train_loader):
            data, target = Variable(data), Variable(target)
            data = data.view(-1, 28*28)
            optimizer.zero_grad()
            net_out = net(data)
            loss = criterion(net_out, target)
            loss.backward()
            optimizer.step()
    # test
    res = []
    res.append(train_size)
    for loader in [train_loader, test_loader]:
        test_loss = 0
        correct = 0
        for data, target in loader:
            data, target = Variable(data), Variable(target)
            data = data.view(-1, 28 * 28)
            net_out = net(data)
            test_loss += criterion(net_out, target).data.item()
            pred = net_out.data.max(1)[1]
            correct += pred.eq(target.data).sum()

        test_loss /= len(loader.dataset)
        res.append(test_loss)
        res.append(100. * correct / len(loader.dataset))
        res.append(len(loader.dataset))

    return res
```

Вывод

В данном разделе были обоснованы средства реализации ПО и реализовано ПО, решающее поставленную задачу.

4 | Исследовательская часть

4.1 Определение состояний недообучения

Цель: провести исследование, состоящее в определении состояний недообучения и переобучения для различного соотношения (% и %) обучающей и тестовой выборок, а также для различного количества скрытых слоёв нейронной сети.

Примем, что модель обучена при точности $> 70\%$. В таблицах 4.1 - 4.3 представлены результаты обучения сетей с 0, 1 и 5 скрытыми слоями с соответствующими состояниями обучения.

Для проведения исследования использовался датасет, состоящий из 60000 картинок, который делился на тестовую и обучающую выборки. Количество эпох обучения – 40.

Таблица 4.1: Результаты обучения для 0 скрытых слоев

Доля обучающей выборки	Значение функции потерь на обучающей выборке	Точность на обучающей выборке(%)	Размер обучающей выборки	Значение функции потерь на тестовой выборке	Точность на тестовой выборке (%)	Размер тестовой выборки	Состояние обучения
0.1	0.0247	57.90	6000	0.0253	56.54	54000	недообучена
0.2	0.0132	80.69	12000	0.0137	80.03	48000	обучена
0.3	0.0096	85.40	18000	0.0096	85.57	42000	обучена
0.4	0.0078	87.38	24000	0.0080	87.11	36000	обучена
0.5	0.0070	88.32	30000	0.0071	87.94	30000	обучена
0.6	0.0065	88.89	36000	0.0064	88.77	24000	обучена
0.7	0.0061	89.30	42000	0.0060	89.36	18000	обучена
0.8	0.0058	89.60	48000	0.0055	90.31	12000	обучена
0.9	0.0056	89.88	54000	0.0047	91.88	6000	обучена

Видим, что в большинстве случаев точность растет с увеличением размера обучающей выборки, однако для первых двух сетей (с 0 и 1 скрытыми слоями) можно заметить, что с определённой доли обучающей выборки (0.7 и 0.5 соответственно) с увеличением размера выборки точность остается почти постоянной. Это означает, что с еще большим увеличением обучающей выборки сети начнут переобучаться.

Таблица 4.2: Результаты обучения для 1 скрытого слоя

Доля обучающей выборки	Значение функции потерь на обучающей выборке	Точность на обучающей выборке(%)	Размер обучающей выборки	Значение функции потерь на тестовой выборке	Точность на тестовой выборке (%)	Размер тестовой выборки	Состояние обучения
0.1	0.0341	30.08	6000	0.0340	30.67	54000	недообучена
0.2	0.0280	52.91	12000	0.0282	51.81	48000	недообучена
0.3	0.0230	59.29	18000	0.0230	59.58	42000	недообучена
0.4	0.0198	62.88	24000	0.0199	62.70	36000	недообучена
0.5	0.0160	71.56	30000	0.0162	71.09	30000	обучена
0.6	0.0142	72.64	36000	0.0142	72.38	24000	обучена
0.7	0.0132	73.19	42000	0.0132	73.22	18000	обучена
0.8	0.0126	73.61	48000	0.0124	73.87	12000	обучена
0.9	0.0121	73.93	54000	0.0114	75.15	6000	обучена

Таблица 4.3: Результаты обучения для 5 скрытых слоев

Доля обучающей выборки	Значение функции потерь на обучающей выборке	Точность на обучающей выборке(%)	Размер обучающей выборки	Значение функции потерь на тестовой выборке	Точность на тестовой выборке (%)	Размер тестовой выборки	Состояние обучения
0.1	0.0361	10.13	6000	0.0360	9.83	54000	недообучена
0.2	0.0361	10.07	12000	0.0360	9.81	48000	недообучена
0.3	0.0360	11.47	18000	0.0360	11.41	42000	недообучена
0.4	0.0358	17.52	24000	0.0359	17.05	36000	недообучена
0.5	0.0357	16.86	30000	0.0357	16.01	30000	недообучена
0.6	0.0355	22.67	36000	0.0354	22.36	24000	недообучена
0.7	0.0347	28.18	42000	0.0348	28.17	18000	недообучена
0.8	0.0316	33.52	48000	0.0316	34.10	12000	недообучена
0.9	0.0207	59.98	54000	0.0199	62.13	6000	недообучена

Кроме того, по результатам для сети с 5 скрытыми слоями можно сделать вывод, что размер выборки слишком мал для обучения такой сети (даже при доле обучающей выборки 0.9 точность менее 60%).

Вывод

В данном разделе было проведено исследование, состоящее в определении состояний недообучения и переобучения для различного соотношения (% и %) обучающей и тестовой выборок, а также для различного количества скрытых слоёв нейронной сети. Были сделаны соответствующие выводы.

ЗАКЛЮЧЕНИЕ

Целью данной лабораторной работы являлось создание программы, которая классифицирует данные из датасета MNIST [1] с использованием нейросетевого подхода с функциией активации ReLU и функцией потерь Cross Entropy. Цель работы была достигнута. Для этого были решены следующие задачи:

— приведена постановка задачи;

— описана нейронная сеть: функция активации ReLU и функция потерь Cross Entropy;

— рассчитан аналитически необходимый размер обучающей выборки по неравенству Чебышёва, необходимый для гарантированного успешного выполнения поставленной задачи;

— приведена схема нейронной сети;

— приведены особенности реализации ПО, решающего поставленную задачу;

— проведено исследование, состоящее в определении состояний недообучения и переобучения для различного соотношения (% и %) обучающей и тестовой выборок, а также для различного количества скрытых слоёв нейронной сети.

В результате исследования было выяснено, что в большинстве случаев точность растет с увеличением размера обучающей выборки, однако для первых двух сетей (с 0 и 1 скрытым слоем) можно заметить, что с определённой доли обучающей выборки (0.7 и 0.5 соответственно) с увеличением размера выборки точность остается почти постоянной. Это означает, что с еще большим увеличением обучающей выборки сети начнут переобучаться.

Кроме того, по результам для сети с 5 скрытыми слоями можно сделать вывод, что размер выборки слишком мал для обучения такой сети (даже при доле обучающей выборки 0.9 точность менее 60%).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. База данных рукописных цифр MNIST [Электронный ресурс]. — Режим доступа: URL: http://yann.lecun.com/exdb/mnist/ (дата обращения: 15.12.2023).
2. Фреймворк машинного обучения Pytorch. — Режим доступа: URL: https://pytorch.org (дата обращения: 15.12.2023).