МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки: «Прикладная математика и информатика»

Магистерская программа: «Вычислительные методы и суперкомпьютерные технологии»

Образовательный курс «Глубокое обучение»

**ОТЧЕТ**

по лабораторной работе №3

**Разработка сверточных нейронных сетей**

**Выполнил:**

студент группы 381703-3м

Жариков Алексей

Нижний Новгород

2018

Содержание

[Цели 3](#_Toc530326965)

[Задачи 4](#_Toc530326966)

[Решаемая задача 5](#_Toc530326967)

[Выбор библиотеки 6](#_Toc530326968)

[Метрика качества решения задачи 6](#_Toc530326969)

[Тренировочные и тестовые наборы данных 6](#_Toc530326970)

[Конфигурации нейронных сетей 7](#_Toc530326971)

[Описание содержимого директории src 8](#_Toc530326972)

[Результаты экспериментов 8](#_Toc530326973)

[Анализ результатов 9](#_Toc530326974)

[Выводы 10](#_Toc530326975)

[Литература 11](#_Toc530326976)

# Цели

1. ***Цель*** настоящей работы состоит в том, чтобы построить архитектуру сверточной нейронной сети, которая позволяет решать практическую задачу с высокими показателями качества.
2. Задачи
3. Выполнение практической работы предполагает решение ***следующих задач***:
4. 1. Разработка нескольких архитектур сверточных нейронных сетей (варьируются количество слоев и виды функций активации на каждом слое) в формате, который принимается выбранной библиотекой глубокого обучения.
5. 2. Обучение разработанных глубоких моделей.
6. 3. Тестирование обученных глубоких моделей.
7. 4. Публикация разработанных программ/скриптов в репозитории на GitHub.
8. 5. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

# Решаемая задача

В данной работе рассматривается набор данных Fashion MNIST, который содержит 70 000 изображений в оттенках серого в 10 категориях. На изображениях показаны отдельные предметы одежды с низким разрешением (28 на 28 пикселей):

Получить можно данный набор по ссылке

<https://www.kaggle.com/zalando-research/fashionmnist> ,

но в ходе работы использовался функционал используемой библиотеки глубокого обучения, непосредственно из нее, просто импортировав и загрузив данные

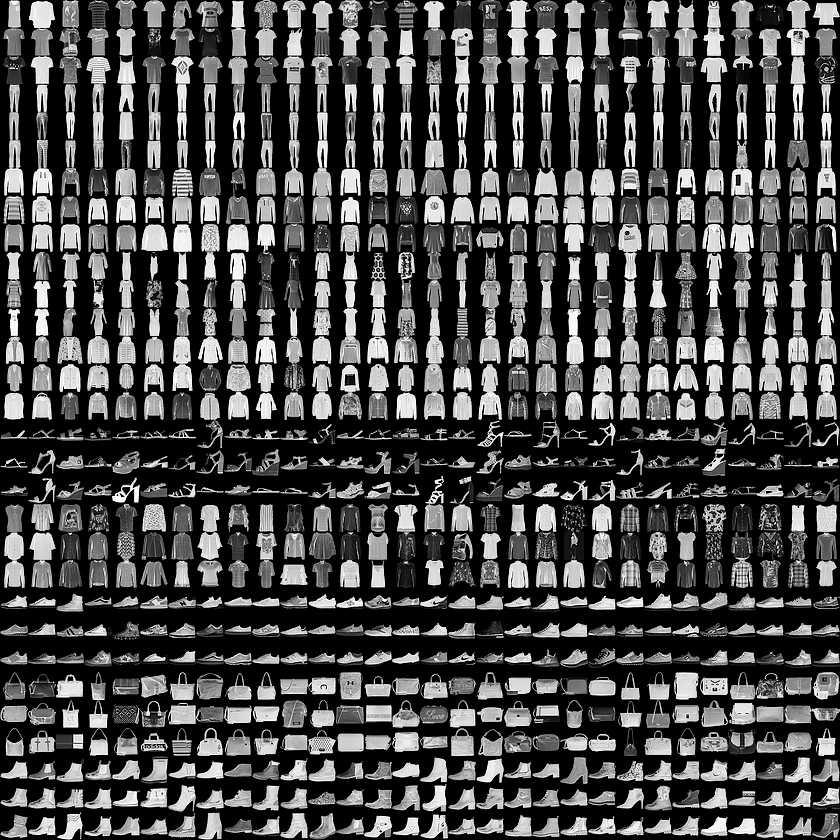


Рисунок Примеры набора данных

# Выбор библиотеки

Для выполнения лабораторных работ выбрана библиотека TensorFlow для языка программирования Python c надстройкой Keras .

На этапе проверки корректности установки библиотеки выполнена разработка и запуск тестового примера сети для решения задачи классификации рукописных цифр набора данных MNIST. Достигнута точность 0.9225.

# Метрика качества решения задачи

В качестве метрики точности решения используется отношение угаданных животных ко всем в тестовой выборке:

# Тренировочные и тестовые наборы данных

# Обучающее множество 60.000 примеров и тестовый набор 10000 примеров. Всего имеется 10 классов различной одежды и обуви.

# Каждый пример обучения и теста назначается одному из следующих классов:

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер** | **Описание** | **Кол.**  **трен** | **Кол.**  **тест** |
| 0 | Футболки / сверху | 6000 | 1000 |
| 1 | Брюки | 6000 | 1000 |
| 2 | Пуловер | 6000 | 1000 |
| 3 | Платье | 6000 | 1000 |
| 4 | Пальто | 6000 | 1000 |
| 5 | Сандали | 6000 | 1000 |
| 6 | Рубашка | 6000 | 1000 |
| 7 | Кроссовки | 6000 | 1000 |
| 8 | Мешок | 6000 | 1000 |
| 9 | Ботильоны | 6000 | 1000 |

Таблица Классы и количество примеров в наборе данных

# Конфигурации автокодировщиков

# В работе рассмотрены по одной конфигурации автокодировщиков сверточный и полносвязанные. Данные схемы получены в результате нескольких экспериментов и будут приведены в отчете как давшие результаты точности.

# Автокодировщики состоят из кодировщика и декодировщика.

# Для данной задачи в качестве функции потерь будет использоваться квадратичное отклонение, это обусловлено тем, что изображение в наборе данных 8-мибитное и по факту нам надо будет предсказать вероятность в каждом из 256 пикселей, и просчет такой нейронной сети является весьма затратным мероприятием, поэтом нам проще и дешевле использовать регрессию. Для бинарного изображения сгодилась бы и кроссэнтропия, но пробы сделать изображение черно-белым для данной задачи давали неудовлетворительные результаты при классификации, и в целом получается решение другой задачи.

# Полносвязанный автокодировщик:

Рисунок Схема полносвязанного автокодировщика

# Сверточный автокодировщик:

# Приведена более подробная схема, так как сверточный автокодировщик имеет более сложную структуру и принципиальны размеры выходных данных после сверток и изменений размерности.

# 

Рисунок Декодировщик CNN - автокодировщика

Рисунок Кодировщик CNN - автокодировщика

**Результаты работы автокодировщиков**



Рисунок Оригиналы изображений



Рисунок Результат работы CNN - автокодировщика



Рисунок Результат работы FC - автокодировщика

Видно, сравнивая результаты автокодировщиков, что сверточная версия справилась куда лучше с задачей отображения, чем полносвязанная конфигурация.

# Конфигурации нейронных сетей

# Результаты экспериментов

Тестовая инфраструктура:

ОП: Python Jupiter cloud colab.research.google.com

# Эксперименты проводились для обоих автокодировщиков с предобученным классификатором и с непредобученным. Суть эксперимента заключалась в сравнении качества обучения и времени обучения нейронной сети из кодировщика+выходной слой в виде функции softmax. С результатами можно ознакомится в таблице.

* Кол-во эпох -3
* Скорость обучения – 0.01

|  |  |  |  |
| --- | --- | --- | --- |
| Автокодировщик | Конфигурация | Точность на тестовом множестве | Время, с |
| Сверточный | Непредобученная | 0.9300 | 65,69 |
| Предобученная | 0.9032 | 46.56 |
| Полносвязанный | Непредобученная | 0.8943 | 654.83 |
| Предобученная | 0.8874 | 231.99 |

Предобучение при помощи автокодировщика дало прирост скорости обучения, но по итогу дало небольшое уменьшение точности.