Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

«Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»

Институт Информационных Технологий, Математики и Механики

Направление: Фундаментальная информатика и информационные технологии

Магистерская программа: Когнитивные системы

**ОТЧЕТ**

по лабораторной работе №2

Тема:

**«Разработка полносвязной нейронной сети»**

Выполнили:

Краснов Александр (гр. 381806м4)

Разин Вячеслав (гр. 381806м4)

Ахмедов Саркар (гр. 381806м4)

Проверила:

доцент, к.т.н. Кустикова В.Д.

Нижний Новгород

2019

Оглавление

[Введение 3](#_Toc32823563)

[Постановка задачи 4](#_Toc32823564)

[Описание набора данных 5](#_Toc32823565)

[Метрика качества решения задачи 6](#_Toc32823566)

[Подготовка данных для нейронной сети 7](#_Toc32823567)

[Тестовые конфигурации сетей 8](#_Toc32823568)

[Результаты экспериментов 8](#_Toc32823569)

[Анализ результатов 11](#_Toc32823570)

# Введение

Пусть - множество изображений, – множество номеров (или наименований) классов. Существует неизвестная *целевая зависимость* – отображение , значения которой известны только на объектах конечной обучающей выборки . Требуется построить алгоритм , способный классифицировать произвольный объект .

В рамках данной лабораторной работы ставится задача классификации изображений из [американского языка жестов (ASL Alphabet).](https://www.kaggle.com/grassknoted/asl-alphabet)

# Постановка задачи

* **Цель работы**

Цель настоящей работы состоит в том, чтобы получить базовые навыки работы с одной из библиотек глубокого обучения (Caffe, Torch, TensorFlow, MXNet или какая-либо другая библиотека на выбор студента) на примере полностью связанных нейронных сетей.

* **Задачи**

Выполнение практической работы предполагает решение следующих задач:

1. Выбор библиотеки для выполнения практических работ курса.

2. Установка выбранной библиотеки на кластере (параметры аутентификации и инструкция по работе с кластером выложена в отдельной задаче в системе redmine).

3. Проверка корректности установки библиотеки. Разработка и запуск тестового примера сети, соответствующей логистической регрессии, для решения задачи классификации рукописных цифр набора данных MNIST (пример разобран в лекционных материалах).

4. Выбор практической задачи компьютерного зрения для выполнения практических работ.

5. Разработка программ/скриптов для подготовки тренировочных и тестовых данных в формате, который обрабатывается выбранной библиотекой.

6. Разработка нескольких архитектур полностью связанных нейронных сетей (варьируются количество слоев и виды функций активации на каждом слое) в формате, который принимается выбранной библиотекой.

7. Обучение разработанных глубоких моделей.

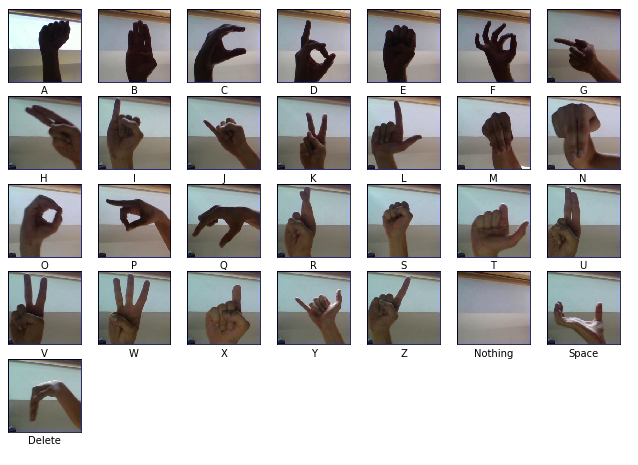
8. Тестирование обученных глубоких моделей.

9. Публикация разработанных программ/скриптов в репозитории на GitHub.

10. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

# Описание набора данных

Для лабораторных работ был выбран один из представленных на Kaggle набор данных <https://www.kaggle.com/grassknoted/asl-alphabet> (ASL Alphabet).



*Рис.1. Представители всех классов набора данных ASL alphabet*

Набор данных ASL Alphabet представляет собой набор изображений из американского языка жестов. Набор включает в себя 87 000 изображений размером 200х200 пикселей. Есть 29 классов, из которых 26-для букв A-Z и 3 класса SPACE, DELETE и NOTHING. В каждом классе 3000 изображений. Для тестирования использовано 10% изображений.

Для реализации различных архитектур нейронных сетей была выбрана открытая библиотека глубокого обучения Keras, написанная на языке программирования Python.

# Метрика качества решения задачи

Для оценки качества задачи классификации выбрана метрика "" (точность), показывающая отношение числа правильных предсказаний к числу предсказаний класса для всех изображений:  
где – количество изображений, – значения, выданные сетью при классификации изображений, – истинные значения классов элементов одежды с изображения.

# Подготовка данных для нейронной сети

В силу большого количества входных изображений было решено сжать каждое из них до размера 32 x 32 пикселей. Сжатие происходит с сохранением соотношения сторон, так как все исходные изображения имеют разрешение 200 x 200. После этого изображения добавляются в массив, нормализуются делением на 255 и разделяются на тренировочную и тестовую выборку в соотношении 9 к 1. Также создаётся отдельный массив для меток, отвечающих за принадлежность изображения к одному из 29 классов. Используется метод one-hot encoding.

# Тестовые конфигурации сетей

В ходе лабораторной работы были изучены четыре различные конфигурации сетей. Варьировались количество скрытых слоев и функции активации. Схемы конфигураций представлены ниже:

|  |  |  |
| --- | --- | --- |
| Input: Input layer | ReLu: Dense | Softmax: Dense |
| (32,32) -> (None, 3072) | (None, 3072)->(None, 1024) | (None, 512)->(None, 29) |

*Рис 2.* Модель 1

|  |  |  |
| --- | --- | --- |
| Input: Input layer | Sigmoid: Dense | Softmax: Dense |
| (32,32) -> (None, 3072) | (None, 3072)->(None, 1024) | (None, 512)->(None, 29) |

*Рис 3.* Модель 2

|  |  |  |
| --- | --- | --- |
| Input: Input layer | Tanh: Dense | Softmax: Dense |
| (32,32) -> (None, 3072) | (None, 3072)->(None, 1024) | (None, 512)->(None, 29) |

*Рис 4.* Модель 3

|  |  |  |  |
| --- | --- | --- | --- |
| Input: Input layer | ReLu: Dense | ReLu: Dense | Softmax: Dense |
| (32,32)->(None, 7500) | (None, 3072)->(None, 1024) | (None, 1024)->(None, 512) | (None, 512)->(None, 29) |

*Рис 5.* Модель 4

|  |  |  |  |
| --- | --- | --- | --- |
| Input: Input layer | Sigmoid: Dense | Sigmoid: Dense | Softmax: Dense |
| (32,32)->(None, 7500) | (None, 3072)->(None, 1024) | (None, 1024)->(None, 512) | (None, 512)->(None, 29) |

*Рис 6.* Модель 5

|  |  |  |  |
| --- | --- | --- | --- |
| Input: Input layer | Tanh: Dense | Tanh: Dense | Softmax: Dense |
| (32,32)->(None, 7500) | (None, 3072)->(None, 1024) | (None, 1024)->(None, 512) | (None, 512)->(None, 29) |

*Рис 7.* Модель 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input: Input layer | ReLu: Dense | ReLu: Dense | ReLu: Dense | Softmax: Dense |
| (32,32)->(None, 7500) | (None, 3072)->(None, 1024) | (None, 1024)->(None, 512) | (None, 512)->(None, 256) | (None, 256)->(None, 29) |

*Рис 8.* Модель 7.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input: Input layer | Sigmoid: Dense | Sigmoid: Dense | Sigmoid: Dense | Softmax: Dense |
| (32,32)->(None, 7500) | (None, 3072)->(None, 1024) | (None, 1024)->(None, 512) | (None, 512)->(None, 256) | (None, 256)->(None, 29) |

*Рис 9.* Модель 8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input: Input layer | Tanh: Dense | Tanh: Dense | Tanh: Dense | Softmax: Dense |
| (32,32)->(None, 7500) | (None, 3072)->(None, 1024) | (None, 1024)->(None, 512) | (None, 512)->(None, 256) | (None, 256)->(None, 29) |

*Рис 10.* Модель 9.

# Результаты экспериментов

Все вычислительные эксперименты выполнялись с помощью Google Colaboratory режиме GPU.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Модель сети | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Количество скрытых нейронов | 1024  29 | 1024  29 | 1024  29 | 1024  512  29 | 1024  512  29 | 1024  512  29 | 1024  512  256  29 | 1024  512  256  29 | 1024  512  256  29 |
| Количество скрытых слоёв | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 |
| Функция активации на свёрточных слоях | relu | sigm | tanh | relu | sigm | tanh | relu | sigm | tanh |
| Батч | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 |
| Количество эпох | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| Скорость обучения | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 |
| Оптимизатор | adam | adam | adam | adam | adam | adam | adam | adam | adam |
| Время обучения, сек | 54.66 | 52.69 | 52.74 | 58.51 | 61.34 | 55.4 | 73.53 | 63.44 | 60.64 |
| Точность (Accuracy) на тренировочном наборе, % | 80.4 | 89.24 | 74.20 | 92.17 | 90.65 | 58.21 | 91.63 | 86.87 | 51.6 |
| Ошибка на тренировочном наборе | 0.61 | 0.4 | 0.79 | 0.22 | 0.28 | 1.16 | 0.23 | 0.36 | 1.31 |
| Точность (Accuracy) на тестовом наборе, % | 78.48 | 89.31 | 72.95 | **93.03** | 91.66 | 61 | 91.83 | 86.79 | 50.45 |
| Ошибка на тестовом наборе | 0.65 | 0.4 | 0.8 | 0.19 | 25.38 | 1.06 | 0.22 | 0.36 | 1.29 |

# Анализ результатов

Варьируемые параметры: количество скрытых слоёв от 1 до 3 и функции активации (relu, sigmoid, tanh).

Итоговые результаты:

* Конфигурации 1-3: лучшую точность показывает сеть с sigmoid – 89.31%.
* Конфигурации 4-6: лучшую точность показывает сеть с relu – 93.03%.
* Конфигурации 7-9: лучшую точность показывает сеть с relu – 91.83%.

Итак, максимальный результат показала четвёртая модель с двумя скрытыми слоями с relu на каждом. Использование трёх скрытых слоёв не оправдало себя. Функция активации tanh показала себя хуже других, причём с увеличением скрытых слоёв точность монотонно снижалась. Sigmoid хороша на одном скрытом слое, но relu лучше на нескольких.

Полносвязные нейронные сети неплохо себя показали в данной задаче, но при использовании сверточных сетей мы можем рассчитывать на лучший результат за более короткое время.