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**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

Направление подготовки: «Фундаментальная информатика

и информационные технологии»

Магистерская программа: «Компьютерная графика»

**Отчет по лабораторной работе**

**«Реализация метода обратного распространения ошибки для двухслойной полносвязной нейронной сети»**

Выполнил:

студент группы 381606м2

Сахаров А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород

2017

Оглавление

[Постановка задачи 3](#_Toc501789720)

[Метод обратного распространения ошибки 4](#_Toc501789721)

[Программная реализация 7](#_Toc501789722)

[Результаты экспериментов 9](#_Toc501789723)

[Вывод 10](#_Toc501789724)

# Постановка задачи

**Цель работы**

На примере построения двухслойной полносвязной нейронной сети для решения задачи классификации рукописных цифр необходимо познакомиться с основными концепциями построения такого рода сетей.

**Задачи**

* Вывести математические формулы для вычисления градиентов функции ошибки по параметрам нейронной сети и формул коррекции весов.
* Разработать программную реализацию.
* Протестировать разработанную программную реализацию.
* Собрать результаты экспериментов.
* Подготовить отчет по проделанной работе.

# Метод обратного распространения ошибки

**Описание:**

Суть метода состоит в распространении сигналов ошибки от выходов ко входам сети, что потребуется для выделения оптимального направления движения метода оптимизации, минимизирующего некоторую функцию ошибки сети. Дано некоторое множество входов . Обозначим вес -го и -го ребра через , а выход -го нейрона обозначим как . Так же у нас есть ожидаемый результат – вектор . Чтобы узнать, насколькоответ нашей сети отличается от ожидаемого результата, будем использовать функцию ошибки *кросс-энтропию*:

Метод обратного распространения ошибки состоит из 2-х частей. Первый этап – прямой проход – позволяет получить итоговое предсказание сети для данного входа. Второй этап – обратный проход – распространяет ошибку в обратном направлении и на основании неё позволяет сделать шаг метода оптимизации в направлении антиградиента минимизируемой функции. Перед началом обучения необходимо проинициализировать веса и смещения некоторыми малыми значениями.

Далее будем рассматривать последовательный режим обучения. В таком режиме корректировка весов выполняется после предъявления каждого примера обучающей выборки. Рассмотрим сам алгоритм.

**Прямой проход:**

1. Подаем на вход вектор входных сигналов .
2. Каждый нейрон входного слоя отправляет полученный сигнал всем нейронам скрытого слоя.
3. Каждый скрытый нейрон суммирует взвешенные входящие сигналы

, а затем применяет активационную функцию:

1. Каждый выходной нейрон суммирует взвешенные входящие сигналы

, а затем применяет активационную функцию:

**Обратный проход:**

1. Каждый выходной нейрон получает целевые значения и вычисляет ошибку:

Используем получившиеся значения:

1. Каждый скрытый нейрон суммирует входящие ошибки (от нейронов в последующем слое) и вычисляет величину ошибки, умножая полученное значение на производную активационной функции:

где – поправка, вычисленная для к-го узла.

Таким образом, для нейронов скрытого слоя получаем следующую ошибку:

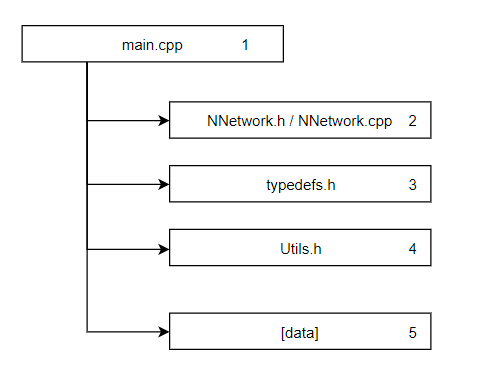
А так же ошибку для нейронов выходного слоя равную

1. Каждый скрытый и выходной нейроны будут изменять веса своих скрытых связей по следующей формуле:

где , – скорость обучения сети, – ошибка, – выход предыдущего слоя.

# Программная реализация

Программа состоит из следующих компонентов:

  
Рис.1 Простейшая диаграмма компонентов программы.

Точкой входа является файл main.cpp, содержащий непосредственную инициализацию параметров сети (из командной строки или значениями по умолчанию).

Класс NNetwork содержит основную логику двухслойной сети. Можно рассмотреть интерфейс сети (NNetwork.h) подробнее:

class NNetwork {

public:

NNetwork(int in\_count, int hidden\_count, int out\_count);

void setup(vec2d &\_data, vec2d& \_validata, int \_epoch\_num, float \_lrn\_rate, float \_epsilon, vec2d& \_validation\_set, ivec1d& \_validation\_labels);

Создание объекта класса производится в два этапа: вызов конструктора и метода setup. Конструктор принимает на вход кол-во нейронов на соответствующих слоях и главным образом занимается резервированием памяти и начальной инициализацией весов случайными числами из равномерного распределения на отрезке (0,1). Метод setup используется для передачи тренировочных и тестовых данных, а также для указания параметров обучения – кол-во эпох, скорость обучения и остановочная точность для кросс-энтропии.

Методы

void train();

void predict(vec1d &x);

int get\_class();

запускают процесс обучения (train), позволяют получить предсказание на выходе (predict) и, собственно, получить номер класса от 0 до 10 (get\_class).

Файл typedefs.h определяет используемые типы, а директория data содержит заранее выгруженные данные датасета MNIST в бинарном формате.

# Результаты экспериментов

Ниже в таблице представлены результаты обучения и тестирования сети на наборе данных MNIST. В зависимости от зависимости параметра скорости обучения изменялась точность выдаваемых результатов.

Таблица 1. Результаты экспериментов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кол-во нейронов на скрытом слое | Скорость обучения | Кол-во эпох | Точность | Энтропия |
| 2 | 0.1 | 10 | 0.3849 | 1.36036 |
| 50 | 0.1 | 10 | 0.9021 | 1.00412 |
| 100 | 0.01 | 10 | 0.9588 | 0.0841478 |
| 100 | 0.1 | 10 | 0.9571 | 0.227585 |
| 200 | 0.01 | 10 | 0.9631 | 0.0595547 |
| 300 | 0.05 | 10 | 0.9673 |  |

По данным проведенных экспериментов сеть дала лучший результат при скорости обучения равной 0.05 и числе нейронов на скрытом слое 300.

# Вывод

В данной работе была рассмотрена и построена двухслойная полносвязная нейронная сеть, решающая задачу распознавания рукописных цифр. На примере разработанной программы был изучен механизм обратного распространения ошибки.