Нижегородский государственный университет им. Н. И. Лобачевского

Институт информационных технологий, математики и механики

Направление подготовки Прикладная математика и информатика

Магистерская программа Вычислительные методы и суперкомпьютерные технологии

Образовательный курс «Методы глубокого обучения для решения задач компьютерного зрения»

**Отчёт**

по лабораторной работе № 1

**«Реализация метода обратного распространения ошибки для двуслойной полностью связанной нейронной сети»**

***Выполнил:***

студент гр. 381603м4

Семеренко А. И.

Нижний Новгород

2018

Оглавление

[Оглавление 2](#_Toc506196658)

[1. Постановка задачи 3](#_Toc506196659)

[2. Вывод математических формул метода обратного распространения ошибки для задачи классификации рукописных символов. 4](#_Toc506196660)

[3. Описание программной реализации 6](#_Toc506196661)

[4. Результаты экспериментов 7](#_Toc506196662)

1. Постановка задачи

В ходе лабораторной работы предполагается решение ***следующих задач***:

* Изучение общей схемы метода обратного распространения ошибки.
* Вывод математических формул для вычисления градиентов функции ошибки по параметрам нейронной сети и формул коррекции весов.
* Проектирование и разработка программной реализации сети, решающей задачу классификации рукописных символов.
* Тестирование разработанной программной реализации.
* Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

Для обучения и тестирования сети предполагается использовать набор данных MNIST.

Метод обратного распространения ошибки разрабатывается, исходя из следующих предположений:

* На входе сети имеется 𝑤 × ℎ нейронов, что соответствует разрешению изображения.
* На выходе сети имеется 𝑘 нейронов, что соответствует количеству классов изображений.
* Скрытый слой содержит 𝑠 нейронов.
* В качестве функции активации на втором слое используется функция softmax.
* В качестве функции ошибки используется кросс-энтропия.

1. Вывод математических формул метода обратного распространения ошибки для задачи классификации рукописных символов.

Нейронная сеть имеет: N - входных нейронов, K - нейронов на скрытом слое и M - выходных.

Вход:

N нейронов

Скрытый слой:

K нейронов

Выход:

M

классов

все связи

все связи

В целом метод обратного распространения ошибки сводится к задаче минимизации функции ошибки относительно синоптических весов. В качестве целевой функции будем использовать кросс энтропию.

где – -й пример из обучающей выборки, – выход нейронной сети на входе . Также учтем, что – новый нейрон.

Для вывода формул и реализации будем использовать предположение, что режим обучения последовательный. Корректировка весов выполняется для каждого элемента обучающей выборки, а также для задач классификации.

Выведем сначала формулы для одного конкретного элемента выборки. Пусть: , ,, тогда целевая функция для конкретного примера имеет вид:

Обозначим синоптические веса от входного слоя к скрытому - , от скрытого к выходному - . Также обозначим – функция активации на выходном слое (softmax), – функция активации на скрытом слое (гиперболический тангенс). В нашем случае все формулы для прямого прохода от входа к выходу выглядят следующим образом:

Итак, с учетом всех вышеуказанных формул и предположений:

Поскольку уже говорилось, что задача сводится к минимизации функции ошибки, в частности градиентными методами, то корректировка весов должна выполнятся по следующим формулам:

где – скорость обучения ().

Получим градиент для корректировки весов.

Частная производная целевой функции по весам от скрытого к выходному слою.

Частная производная целевой функции по весам от входного к скрытому слою.

1. Описание программной реализации

Реализован класс DoubleLayerFCNN.

Конструктор принимает на вход размеры слоев: входного, скрытого и выходного.

Класс имеет 3 публичных метода:

* train(double \*\*data, double \*label, int sampleSize, int numberEpochs, double learningRate, double errorCrossEntropy)
* precision(double \*\*data, double \*label, int sampleSize)
* cleanNetWeights() – сброс значений весов

где data – набор данных (количество элементов \* размер входного слоя), label – разметка (количество элементов), sampleSize – количество элементов выборки, learningRate – скорость обучения, numberEpochs и errorCrossEntropy – количество эпох обучения и требуемая ошибка, по совместительству являются критериями останова.

Общий алгоритм обучения сети:

1. Инициализация весов случайными значениями.
2. Для каждого элемента выборки.
3. Прямой проход: вычисляем значения на скрытом слое, далее вычисляем значения на выходном слое.
4. Обратный проход: вычисляем градиенты и корректируем веса
5. Подсчитываем кросс-энтропию
6. Повторяем пункты 2-5 пока кросс-энтропия не достигнет требуемого значения или не закончится указанное число эпох.

Точность в нашем случае считается как отношение верно определенных изображений к их общему числу.

Для запуска программы необходимо клонировать репозиторий, и запустить один из .bat скриптов в зависимости от версии, установленной на вашем компьютере Visual Studio. Если у вас установлена другая версия, то в корне репозитория из командной строки вызвать соответствующую команду cmake. Открыть собранное решение и запустить проект DL\_Lab1\_Backpropagation.

1. Результаты экспериментов

Требуемое значение кросс-энтропии для всех экспериментов было указано 0.005. Результаты представлены в таблице ниже.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Число нейронов скрытого слоя | Скорость обучения | Количество эпох | Точность классификации на тестовом наборе | Точность классификации на обучающем наборе |
| 50 | 0.005 | 34 | 0.9731 | 0.9997 |
| 100 | 0.005 | 17 | 0.9789 | 0.9997 |
| 200 | 0.01 | 15 | 0.9818 | 0.9997 |
| 200 | 0.005 | 20 | **0.9836** | 0.9997 |