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: прикладной математики

Направление подготовки: «Прикладная математика и информатика»

Магистерская программа: «Системное программирование»

Отчет по лабораторной работе
«Реализация метода обратного распространения ошибки для двуслойной полностью связанной нейронной сети»

Выполнил:
студент группы 381603м4
Беспалов М.А.
_____

Нижний Новгород

2017

1. ПОСТАНОВКА ЗАДАЧИ

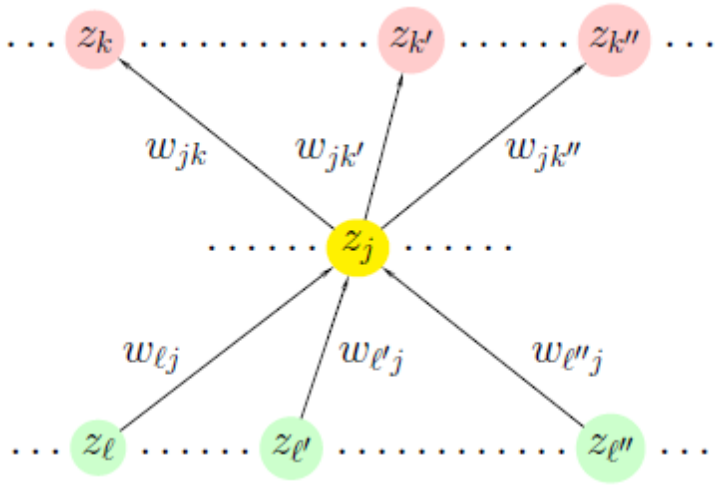
Требуется реализовать метод обратного распространения ошибки на примере 2-хслойной полно-связной сети прямого распространения. Для тестирования метода должен использоваться датасет MNIST.

Порядок выполнения:

1. Изучение общей схемы метода обратного распространения ошибки.
2. Вывод математических формул для вычисления градиентов функции ошибки по параметрам нейронной сети и формул коррекции весов.
3. Проектирование и разработка программной реализации.
4. Тестирование разработанной программной реализации.
5. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

2. ТЕОРИЯ

Будем рассматривать произвольный слой нейронной сети:



$$z_j = \sigma(s_j) = \sigma\Big(\sum w_{\ell j} z_\ell\Big)$$

Требуется минимизировать ошибку:

$$\delta_j = \frac{\partial R^{(i)}}{\partial s_j}$$

Для выходных узлов:

$$R^{(i)}(w) = \frac{1}{2}\sum\Big(y_j^{(i)} - f_j(x^{(i)})\Big)^2$$

$$\delta_j = \frac{\partial R^{(i)}}{\partial s_j} = z_j - y_j^{(i)}$$

Для всех остальных:

$$\delta_j = \frac{\partial R^{(i)}}{\partial s_j} = \sum_{k\in\text{Children}(j)} \frac{\partial R^{(i)}}{\partial s_k}\cdot\frac{\partial s_k}{\partial z_j}\cdot\frac{\partial z_j}{\partial s_j} =$$

$$\sum_{k\in\text{Children}(j)} \delta_k\, w_{jk}\,\sigma'(s_j) = \sigma'(s_j)\sum_{k\in\text{Children}(j)} \delta_k\, w_{jk}$$

Для все нейронов справедливо, что

$$\frac{\partial R^{(i)}}{\partial w_{\ell j}} = \frac{\partial R^{(i)}}{\partial s_j} \cdot \frac{\partial s_j}{\partial w_{\ell j}} = \delta_j z_\ell$$

Оптимизация весов достигается за счет градиентного спуска:

$$w_{\ell j}^{(r+1)} = w_{\ell j}^{(r)} - \rho \frac{\partial R^{(i)}}{\partial w_{\ell j}}$$

Общий алгоритм действий:

1. Случайная инициализация весов
2. Цикл по элементам датасета:

   2.1 Прямой ход – вычисляем значения нейронов для входа

   2.2 Обратный ход – начиная с верхнего слоя пересчитываем ошибки

   2.3 Вычитаем градиент

В качестве функции активации используется супер-тангенс. Его производная выглядит следующим образом:

$$\sigma'(s_j) = (1 - s_j)(1 + s_j)$$

2. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

Файлы проекта:

- Lab1\src\main.cpp – исходник исполняемого файла
- Lab1\src\MnistReader.h, Lab1\src\MnistReader.cpp – исходные файлы для чтения датасета MNIST из файла.
- Lab1\src\TwoLayerNN.h, Lab1\src\TwoLayerNN.cpp – исходные файлы двухслойной нейронной сети
- Lab1\src\LayerNN.h, Lab1\src\LayerNN.cpp – исходные файлы для класса Layer
- Lab1\CMakeLists.txt – файл проекта CMake
- Lab1\resource\x_test.idx3-ubyte, Lab1\resource\x_train.idx3-ubyte, Lab1\resource\y_test.idx1-ubyte, Lab1\resource\y_train.idx1-ubyte – файлы датасета MNIST.

3. РЕЗУЛЬТАТЫ

Наилучший результат:

- Количество нейронов на скрытом слое – 325
- Learning rate – 0.006
- Количество эпох – 50
- Веса заданы с помощью равномерного распределения в отрезке [-0.5, 0.5]
- Accuracy на тестовой выборке – 0.9818