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: «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
Новак А.Е.

____

Нижний Новгород
2017

СОДЕРЖАНИЕ

.

1 Постановка задачи

1.1 Постановка задачи

Необходимо изучить и реализовать метод обратного распространения ошибки для обучения глубоких нейронных сетей на примере двуслойной полностью связанной сети (один скрытый слой). В качестве данных для обучения и тестирования сети необходимо использовать набор MNIST.

1.2 Цели работы

Для выполнения лабораторной работы требуется решить следующие задачи:

- Изучить общую схему метода обратного распространения ошибки.
- Вывести необходимые математические формулы для вычисления градиентов, правильной коррекции весов и вычисления ошибки.
- Разработать программную реализацию метода, позволяющую работать с набором данных MNIST
- Протестировать разработанную программную реализацию. Найти оптимальные параметры.

2 Описание метода обратного распространения ошибки. Вывод математических формул

2.1 Математическое объяснение метода. Постановка задачи оптимизации

Для дальнейших рассуждений введем следующие обозначения:

- N – количество входных нейронов;
- M – количество выходных нейронов;
- K – количество нейронов на скрытом слое;
- L – количество обучающих примеров.

В качестве функции ошибки рассматривается кросс-энтропия:

$$E(w) = -\frac{1}{L}\sum_{k=1}^{L}\sum_{j=1}^{M} y_j^k ln(u_j^k)\,, y_j^k = 1 \leftrightarrow x^k \in j \text{ классу},$$

где

$$y^k = (y_j^k)_{j=\overline{1,M}} \in Y \text{ – множество обучающих примеров,}$$

$$u^k = (u_j^k)_{j=\overline{1,M}} \text{ – выход нейронной сети, полученный для входного примера}$$

$$x^k = (x_i^k)_{i=\overline{1,N}} \in X.$$

Для вывода, а также в программной реализации будем использовать предположение, что у нас последовательный режим обучения. В этом режиме корректировка весов выполняется после прохода каждого примера обучающей выборки. Возьмём конкретный обучающий пример

$$x = (x_1, x_2, \dots, x_N),$$

$$y = (y_1, y_2, \dots, y_M),$$

$$u = (u_1, u_2, \dots, u_M).$$

Тогда функция ошибки примет вид

$$E(w) = -\sum_{j=1}^{M} y_j \ln(u_j).$$

Обозначим $w_{is}^{(1)}$ – веса синаптических связей от входных нейронов к нейронам скрытого слоя, $w_{sj}^{(2)}$ – от нейронов скрытого слоя к выходным нейронам нашей сети. Выходной сигнал нейрона скрытого слоя вычисляется как

$$v_s = \varphi(f_s), \text{ где } \varphi \text{ – функция активации на скрытом слое,}$$

$$f_s = \sum_{i=0}^{N} w_{is}^{(1)} x_i\,,\ \ s = \overline{0,K} \text{- взвешенная сумма входных сигналов.}$$

Сигнал выходного нейрона определяется как

$$u_j = h\big(g_j\big), \text{ где } h \text{ - функция активации на последнем слое,}$$

$g_j = \sum_{s=0}^{K} w_{sj}^{(2)} v_s$, $j = \overline{1, M}$ – взвешенная сумма сигналов со скрытого слоя.

В качестве функции активации на выходном слое рассмотрим функцию $\boldsymbol{softmax}$:

$$u_j = \frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}}.$$

Таким образом,

$$E(w) = -\sum_{j=1}^{M} y_j \ln\left(\frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}}\right) = -\sum_{j=1}^{M} y_j\left(g_j - \ln\sum_{m=1}^{M} e^{g_j}\right),$$

$$g_j = \sum_{s=0}^{K} w_{sj}^{(2)} \varphi\left(\sum_{i=0}^{N} w_{is}^{(1)} x_i\right).$$

Глядя на полученную функцию ошибки, можно сказать, что задача обучения нейронной сети сводится к задаче оптимизации функции ошибки по всем весам сети

$$E(w) \to min_w.$$

2.2 Обратное распространение

Метод обратного распространения ошибки определяет то, каким образом мы будем производить изменение параметров сети w . Для этого, например, используются градиентные методы оптимизации. Производная целевой функции по параметрам последнего слоя $w_{sj}^{(2)}$ вычисляется по следующей формуле:

$$\frac{\partial E(w)}{\partial w_{sj}^{(2)}} = \frac{\partial E}{\partial g_j}\frac{\partial g_j}{\partial w_{sj}^{(2)}},$$

$$\frac{\partial g_j}{\partial w_{sj}^{(2)}} = v_s,$$

В нашей задаче:

$$\sum_{j=1}^{M} y_j = 1$$

$$\delta_j^{(2)} = \frac{\partial E}{\partial g_j} = -\frac{\partial}{\partial g_j}\left[\sum_{j=1}^{M} y_j\left(g_j - \ln\sum_{m=1}^{M} e^{g_j}\right)\right] = -\left(-y_1 \frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}} - \cdots - y_{j-1}\frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}}\right.$$

$$\left. + \; y_j\left(1 - \frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}}\right) - \; y_{j+1}\frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}} - \cdots - y_M \frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}}\right) =$$

$$\left(\sum_{j=1}^{M} y_j\right) \cdot \frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}} - \; y_j = \frac{e^{g_j}}{\sum_{m=1}^{M} e^{g_m}} - \; y_j = u_j - \; y_j.$$

В итоге получаем:

$$\frac{\partial E(w)}{\partial w_{sj}^{(2)}} = \left(u_j - y_j\right) \cdot v_s = \delta_j^{(2)} \cdot v_s.$$

Производная целевой функции по параметрам скрытого слоя $w_{is}^{(1)}$ вычисляется по формуле:

$$\frac{\partial E(w)}{\partial w_{is}^{(1)}} = \frac{\partial E}{\partial f_s} \frac{\partial f_s}{\partial w_{is}^{(1)}} = \delta_s^{(1)} x_i.$$

$$\frac{\partial E}{\partial f_s} = \sum_{j=1}^{M} \frac{\partial E}{\partial g_j} \frac{\partial g_j}{\partial v_s} \frac{\partial \varphi}{\partial f_s} = \frac{\partial \varphi}{\partial f_s} \sum_{j=1}^{M} \frac{\partial E}{\partial g_j} \frac{\partial g_j}{\partial v_s} = \frac{\partial \varphi}{\partial f_s} \sum_{j=1}^{M} \delta_j^{(2)} w_{sj}^2$$

В итоге имеем $\frac{\partial E(w)}{\partial w_{is}^{(1)}} = \frac{\partial \varphi}{\partial f_s} \left[\sum_{j=1}^{M} \delta_j^{(2)} w_{sj}^2\right] \cdot x_i$.

Если функция активации на скрытом слое гиперболический тангенс:

$\varphi(f_s) = th(f_s)$, то

$$\frac{\partial \varphi}{\partial f_s} = (1 - \varphi) * (1 + \varphi) = (1 - v_s) * (1 + v_s).$$

Градиент будет выражаться:

$$\frac{\partial E(w)}{\partial w_{sj}^{(2)}} = \delta_j^{(2)} \cdot v_s, \quad \frac{\partial E(w)}{\partial w_{is}^{(1)}} = \delta_s^{(1)} x_i.$$

Согласно градиентным методам на каждом шаге $r + 1$ обучения сети производится коррекция весов по следующим формулам:

$$w_{is}^{(1)(r+1)} = w_{is}^{(1)(r)} - \eta \frac{\partial E(w)}{\partial w_{is}^{(1)}},$$

$$w_{sj}^{(2)(r+1)} = w_{sj}^{(2)(r)} - \eta \frac{\partial E(w)}{\partial w_{sj}^{(2)}}.$$

где η – скорость обучения.

3 Алгоритм метода обратного распространения ошибки

1.) Инициализация весов w некоторыми значениями
2.) for epoch = $\overline{1,\ \text{maxEpochs}}$
3.) for i = $\overline{0, \mathrm{W} * \mathrm{H}}$
4.) Прямой проход нейронной сети
5.) Обратный проход
6.) Шаги 3-5 повторяются до тех пока, пока не выполнится критерий остановки. Обычно это или максимальное число эпох или достигнутая точность обучения.

Прямой проход:

Подать на вход x_i. Вычислить значения выходных сигналов нейронов скрытого слоя

$v_s, s = \overline{0, K}, K$ – количество нейронов на скрытом слое

и значение производной функции активации на скрытом слое

$$\frac{\partial \varphi}{\partial f_s}.$$

Вычислить выходные сигналы нейронов последнего слоя

$u_j, j = \overline{1, M},\ M$ – количество классов изображений.

Коротко, его можно изобразить как:

$$x_i \to v_s, \frac{\partial \varphi}{\partial f_s} \to u_j$$

Обратный проход:

Вычисляются значения градиентов целевой функции:

Начинаем с конца:

- for $\mathbf{j = \overline{1, M}}$

$$\boldsymbol{\delta_j^{(2)} = u_j - y_j, \frac{\partial E(w)}{\partial w_{sj}^{(2)}} = \delta_j^{(2)} \cdot v_s}$$

Скрытый слой**:**

- **for** $\boldsymbol{s = \overline{0, K}}$

$$\boldsymbol{\delta_s^{(1)} = -\frac{\partial \varphi}{\partial f_s} \sum_{j=1}^{M} \delta_j^{(2)} w_{sj}^{2}\ , \frac{\partial E(w)}{\partial w_{is}^{(1)}} = \delta_s^{(1)} x_i}$$

- **for по дугам**

$$\mathbf{w}_{\mathbf{is}}^{(\mathbf{1})} = \mathbf{w}_{\mathbf{is}}^{(\mathbf{1})} - \boldsymbol{\eta}\frac{\partial \mathbf{E}(\mathbf{w})}{\partial \mathbf{w}_{\mathbf{is}}^{(\mathbf{1})}}$$

$$\mathbf{w}_{\mathbf{sj}}^{(\mathbf{2})} = \mathbf{w}_{\mathbf{sj}}^{(\mathbf{2})} - \boldsymbol{\eta}\frac{\partial \mathbf{E}(\mathbf{w})}{\partial \mathbf{w}_{\mathbf{sj}}^{(\mathbf{2})}}$$

4 Описание программной реализации

Разработан проект в MS VS2015 который содержит следующие файлы:

- ReadMnist.h – функции для работы с набором данных MNIST
- NeuralNetwork.h – описание класса нейронной сети
- NeuralNetwork.cpp – реализация методов для работы с нейросетью
- Main.cpp – тестовое приложение

5 Результаты

Было разработано приложение позволяющее обучать и тестировать двухслойную нейронную сеть с использование набора данных MNIST.

Наилучшее результаты были достигнуты с использование следующих параметров:

Число нейронов скрытого слоя – **300** нейронов

Число эпох – **25**

Скорость обучения - **0.008**

Точность на тестовой выборке – **0.9805**

Точность на тренировочной выборке - **0.999517**