

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

Направление подготовки «Прикладная математика и информатика»
Магистерская программа «Математическая кибернетика»

Отчет по лабораторной работе

Реализация метода обратного распространения ошибки для
двуслойной полностью связанной нейронной сети

Выполнила:
студентка 381603м3 группы
Садова Дарья Дмитриевна

Нижний Новгород, 2017г.

Содержание

1	Постановка задачи	3
2	Архитектура сети и математические выкладки	3
2.1	Прямой проход	3
2.2	Обратный проход	3
3	Программная реализация	4
4	Результаты	4

1 Постановка задачи

Необходимо изучить и реализовать метод обратного распространения ошибки для обучения глубокой нейронной сети на примере двуслойной полностью связанной сети. Нейронная сеть обучается с помощью MNIST-данных¹ для распознавания рукописных цифр. Тренировочный набор состоит из 60.000 экземпляров, тестовый - 10.000.

1. Изучить общую схему метода обратного распространения ошибки.
2. Вывести математические формулы для вычисления градиентов функции ошибки по параметрам нейронной сети и формулы коррекции весов.
3. Спроектировать и разработать программную реализацию.
4. Протестировать программу и найти оптимальные параметры.

2 Архитектура сети и математические выкладки

По условию задачи сеть должна состоять из 3 слоев (один скрытый слой). В случае нашей сети на вход подается 28*28 нейронов. На внутреннем слое количество нейронов может варьироваться в зависимости от входных параметров сети. На выходном слое столько нейронов, на сколько типов классифицируются изображения. Сеть является полносвязной, то есть каждый нейрон связан со всеми нейронами предыдущего слоя.

Способ обучения состоит из прямого прохождения сети и затем, на каждом шаге, методом обратного распространения ошибки корректируются веса каждого из слоев.

2.1 Прямой проход

Пусть x_1, \dots, x_n - множество входных нейронов, u_1, \dots, u_k - множество выходных нейронов.

Перед обучением сети инициализируются веса ω_{ij} и смещения b_j случайными малыми значениями. Обучение происходит с учителем, нам известны правильные классы для обучающей выборки - для каждого изображения дан вектор y_1, \dots, y_s . Чтобы узнать насколько ответ сети отличается от правильного, используем функцию ошибки кросс-энтропию:

$$E = - \sum_{j=1}^n y_j \log(u_j) \quad (1)$$

На прямой проход сети каждый входной нейрон отправляет сигнал всем нейронам в следующем скрытом слое

$$s_j = b_j + \sum_{i=1}^m x_i \omega_{ij} \quad (2)$$

и применяется функция активации - гиперболический тангенс. После чего для выходного слоя аналогично находится взвешенная сумма и применяется функция softmax:

$$f(s_j) = \frac{e^{s_j}}{\sum_{j=1}^n e^{s_j}} \quad (3)$$

2.2 Обратный проход

Для каждого нейрона выходного слоя рассчитывается ошибка:

$$\frac{\partial E}{\partial s_i} = \sum_{j=1}^n \frac{\partial E}{\partial u_j} \frac{\partial u_j}{\partial s_i}; \quad \frac{\partial E}{\partial u_j} = - \frac{\partial y_j \log(u_j)}{\partial u_j} = - \frac{y_j}{u_j} \quad (4)$$

¹<http://yann.lecun.com/exdb/mnist>

$$\frac{\partial u_j}{\partial s_i} = \begin{cases} \frac{\partial \frac{e^{s_j}}{\sum_k e^{s_k}}}{\partial s_j} = \frac{e^{s_j} \sum_k e^{s_k} - (e^{s_j})^2}{(\sum_k e^{s_k})^2} = \frac{e^{s_j}}{\sum_k e^{s_k}} - \frac{(e^{s_j})^2}{(\sum_k e^{s_k})^2} = u_j(1 - u_j), & \text{если } i = j \\ \frac{\partial \frac{e^{s_i}}{\sum_k e^{s_k}}}{\partial s_j} = \frac{-e^{s_i} e^{s_j}}{(\sum_k e^{s_k})^2} = -u_i u_j, & \text{если } i \neq j \end{cases} \quad (5)$$

Получаем:

$$\delta_i = \frac{\partial E}{\partial s_i} = \sum_j u_i y_j - y_i(1 - u_i) = -y_i + y_i u_i + u_i \sum_j y_j = -y_i + u_i(y_i + \sum_j y_j) = u_i - y_i \quad (6)$$

Каждый нейрон скрытого слоя суммирует входящие ошибки и умножает величину полученной ошибки на производную активационной функции. Если активационная функция - гиперболический тангенс, то получаем:

$$\phi(s_i) = th(s_i); \quad \frac{\partial \phi}{\partial s_i} = (1 - \phi) * (1 + \phi); \quad (7)$$

Таким образом получаем для нейронов выходного слоя ошибку

$\delta_j = u_j - y_j$, а для нейронов скрытого слоя $\delta_j = (1 - \phi) * (1 + \phi)$

На каждом шаге обучения производится коррекция весов:

$$\omega_{ij} = \omega_{ij} - \eta \delta_j u_i \quad (8)$$

Данная процедура повторяется для каждого тренировочного набора. Обучение останавливается после обучения всего тестового набора заданное количество эпох или по достижении заданной ошибки между ожидаемым и реальным выходами по формуле (1).

3 Программная реализация

Программа написана на языке программирования C++ и содержит следующие структуры:

1. *Класс NNetwork*: реализована архитектура полносвязной нейронной сети.
2. *Класс ReadData*: происходит чтение из файла тренировочных и тестовых данных.
3. *Функция main*: отвечает за создание классов и имеет возможность интерактивного ввода параметров сети.

4 Результаты

Для тестирования обучения сети использовались различные параметры. Точность правильных ответов тренировочной и тестовой выборки в зависимости от параметров показаны в таблице:

Кол-во эпох	Нейронов в скрытом слое	Скорость обучения	Макс. ошибка	Точность тренировочного набора	Точность тестового набора
20	200	0.05	0.05	0.9432	0.9368
15	200	0.01	0.005	0.9992	0.9792
20	150	0.03	0.005	0.9847	0.9689
20	300	0.006	0.0005	0.999	0.9784
20	200	0.009	0.003	0.9999	0.9797
15	300	0.01	0.005	0.9981	0.9794

Нейронная сеть с наибольшим процентом правильных ответов показала точности:

0.9999 для тренировочного набора;

0.9797 для тестового набора.

```
epoch = 0
Accuracy of training set: 0.944317 Accuracy of test set: 0.943
epoch = 1
Accuracy of training set: 0.966333 Accuracy of test set: 0.9625
epoch = 2
Accuracy of training set: 0.975983 Accuracy of test set: 0.9698
epoch = 3
Accuracy of training set: 0.980983 Accuracy of test set: 0.9721
epoch = 4
Accuracy of training set: 0.984967 Accuracy of test set: 0.9734
epoch = 5
Accuracy of training set: 0.987667 Accuracy of test set: 0.9745
epoch = 6
Accuracy of training set: 0.9899 Accuracy of test set: 0.9756
epoch = 7
Accuracy of training set: 0.992017 Accuracy of test set: 0.9767
epoch = 8
Accuracy of training set: 0.993333 Accuracy of test set: 0.9764
epoch = 9
Accuracy of training set: 0.9945 Accuracy of test set: 0.9765
epoch = 10
Accuracy of training set: 0.995817 Accuracy of test set: 0.9774
epoch = 11
Accuracy of training set: 0.997033 Accuracy of test set: 0.9775
epoch = 12
Accuracy of training set: 0.997583 Accuracy of test set: 0.9788
epoch = 13
Accuracy of training set: 0.998067 Accuracy of test set: 0.9794
```

Рис. 1: Пример работы программы