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
Кривоносов М.И.
_____

Нижний Новгород

2017

Содержание

Постановка задачи

Необходимо изучить и реализовать метод обратного распространения ошибки для обучения глубоких нейронных сетей на примере двуслойной полносвязной сети (один скрытый слой). Тестирование и обучение нейронной сети провести на наборе изображений рукописных цифр MNIST.

Для выполнения лабораторной работы необходимо решить следующие задачи:

1. Изучить общую схему метода обратного распространения ошибки.
2. Вывод математических формул для вычисления градиентов функции ошибки по параметрам нейронной сети и формул коррекции весов.
3. Разработать программную реализацию метода, позволяющую работать с набором данных MNIST
4. Протестировать разработанную программную реализацию. Найти оптимальные параметры.

Метод обратного распространения ошибки

Постановка задачи оптимизации

Рассмотрим задачу обучения с учителем для полносвязной нейронной сети с одним скрытым слоем.

Пусть $X = \left\{x^k \middle| x^k = \left(x_i^k\right)_{i=\overline{1,N}}\right\}$ – множество входов обучающих примеров сети,

$Y = \left\{y^k \middle| y^k = \left(y_j^k\right)_{j=\overline{1,K}}\right\}$ – множество выходов обучающих примеров,

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

N – количество входных нейронов;

S – количество нейронов на скрытом слое;

K – количество выходных нейронов;

L – количество обучающих примеров.

Выберем в качестве функции ошибки кросс-энтропию:

$$E(w) = -\frac{1}{L}\sum_{k=1}^{L}\sum_{j=1}^{K} y_j^k \ln u_j^k, \qquad y_j^k = \begin{cases} 1, & \text{если } x^k \in j-\text{ому классу} \\ 0, & \text{иначе} \end{cases},$$

Оптимизационную задачу будем решать последовательно для каждого отдельного примера из выборки, обновляя синаптические веса.

Рассмотрим один пример из выборки, тогда функция ошибки:

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

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

$v_s = \varphi(f_s), s = \overline{1,S}$, где φ – функция активации на скрытом слое,

$f_s = \sum_{i=0}^{N} w_{is}^{(1)} x_i$ – взвешенная сумма входных сигналов.

где $x_0 = 1$.

Сигнал выходного нейрона:

$u_j = h(g_j)$, где h - функция активации на последнем слое,

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

В качестве функции активации на выходном слое рассмотрим функцию **softmax**:

$$u_j = \frac{e^{g_j}}{\sum_{p=1}^{K} e^{g_p}}$$

Таким образом,

$$E(\mathbf{w}) = -\sum_{j=1}^{K} y_j \ln \frac{e^{g_j}}{\sum_{p=1}^{K} e^{g_p}} = -\sum_{j=1}^{K} y_j \left(g_j - \ln \sum_{p=1}^{K} e^{g_p} \right),$$

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

Задача обучения нейронной сети сводится к задаче оптимизации функции ошибки по всем весам сети:

$$\min_{\mathbf{w}} E(\mathbf{w}).$$

Метод обратного распространения ошибки

Для решения задачи оптимизации используют метод обратного распространения ошибки. Используя градиентные методы оптимизации, получим формулы для пересчёта синаптических весов **w**. Вычислим производную целевой функции по параметрам последнего слоя $w_{sj}^{(2)}$:

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

$$\delta_j^{(2)} = \frac{\partial E}{\partial g_j} = -\frac{\partial}{\partial g_j} \sum_{j=1}^{K} y_j \left(g_j - \ln \sum_{p=1}^{K} e^{g_p} \right) =$$

$$= -\left(- \sum_{k=1,\ k \neq j}^{K} y_j \frac{e^{g_j}}{\sum_{p=1}^{K} e^{g_p}} + y_j \left(1 - \frac{e^{g_j}}{\sum_{p=1}^{K} e^{g_p}} \right) \right) =$$

$$= \left(\sum_{j=1}^{K} y_j \right) \frac{e^{g_j}}{\sum_{p=1}^{K} e^{g_p}} - y_j = \frac{e^{g_j}}{\sum_{p=1}^{K} e^{g_p}} - y_j = u_j - y_j$$

Таким образом производная по весам второго слоя:

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

Вычислим производную целевой функции по параметрам скрытого слоя $w_{is}^{(1)}$:

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

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

В итоге имеем

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

В качестве функции активации на скрытом слое возьмём гиперболический тангенс:

$\varphi(f_s) = \tanh f_s$, то

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

Градиент целевой функции:

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

В соответствии с градиентным методом на $t+1$ шаге обучения сети необходимо произвести коррекцию синаптических весов:

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

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

где η – скорость обучения.

Алгоритм метода обратного распространения ошибки

1. Инициализация весов w некоторыми значениями
2. Цикл по эпохам

   Для каждого элемента в перемешанной выборке:

   1) Прямой проход нейронной сети
   2) Обратный проход

Прямой проход:

1. подать на вход x_i;
2. вычислить значения выходных сигналов нейронов скрытого слоя v_s и значение производной функции активации на скрытом слое $\frac{\partial \varphi}{\partial f_s}, s = \overline{1,S}$;
3. вычислить выходные сигналы нейронов последнего слоя $u_j, j = \overline{1,K}$.

Обратный проход:

1. Вычислить значения градиентов целевой функции:

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

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

2. Обновить значения синаптических весов

$$w_{is}^{(1)} \leftarrow w_{is}^{(1)} - \eta \frac{\partial E(w)}{\partial w_{is}^{(1)}}$$

$$w_{sj}^{(2)} \leftarrow w_{sj}^{(2)} - \eta \frac{\partial E(w)}{\partial w_{sj}^{(2)}}$$

Описание программной реализации

Структура проекта

Разработан проект, который содержит следующие файлы:

- code\ann-library\src\ANN.h – (Artificial Neural Network) описание класса нейронной сети
- code\ann-library\src\ANN.cpp – реализация методов для работы с нейросетью
- code\ann-library\tests\ANN_test.cpp – тесты методов класса ANN
- code\lab1-mnist\src\main.cpp – тестовое приложение для класса ANN на наборе MNIST

Сборка проекта

Создание проекта Visual Studio и загрузка файлов MNIST:

```
md ..\ann-course-practice-build
cd ..\ann-course-practice-build
cmake ..\ann-course-practice
```

Проект можно собрать с помощью Visual Studio, либо через командную строку с помощью msbuild. Для сборки командной строки необходимо указать вместо `%VS140COMNTOOLS%` путь до `VsDevCmd.bat`. Обычно достаточно изменить версию `140` на необходимую.

```
call "%VS140COMNTOOLS%\VsDevCmd.bat"

msbuild ..\ann-course-practice-build\ann-course-practice.sln
/property:Configuration=Release
```

Запуск тестов

Для запуска тестов класса ANN необходимо запустить файл:

```
ann-course-practice-build\bin\test-ann-library.exe
```

Запуск приложения

Для запуска приложения для тестирования класса ANN на данных MNIST необходимо запустить файл:

`ann-course-practice-build\bin\lab1-mnist.exe`

Доступные аргументы командной строки:

1. `num_hidden_layers` – число нейронов скрытого слоя (по умолчанию 300)
2. `num_epoch` – число эпох для расчета (по умолчанию 25)
3. `learning_rate` – скорость обучения (по умолчанию 0.008)
4. `sigma` – стандартное отклонение для инициализации весов с помощью нормального распределения (по умолчанию 0.01)
5. `images_relative_dir` – относительный путь к директории с изображениями (относительно каталога с приложением)
6. `images_dir` – абсолютный путь к директории с изображениями

```
F:\PC\UNN\ANN\ann-course-practice>(..\ann-course-practice-build\bin\lab1-mnist.exe sigma=0.01 lear=0.008 )
MNIST.
Train data set:
Path to labels: ..\ann-course-practice-build\bin/../download/train-labels.idx1-ubyte
Path to images: ..\ann-course-practice-build\bin/../download/train-images.idx3-ubyte
Size: 60000 Width: 28 Height: 28
Test data set:
Path to labels: ..\ann-course-practice-build\bin/../download/t10k-labels.idx1-ubyte
Path to images: ..\ann-course-practice-build\bin/../download/t10k-images.idx3-ubyte
Size: 10000 Width: 28 Height: 28
Epoch: 0 Test accuracy: 0.9481 Train accuracy: 0.9503 Calculation epoch time: 123 sec. All calculation time: 123 sec.
Epoch: 1 Test accuracy: 0.9605 Train accuracy: 0.964433 Calculation epoch time: 126 sec. All calculation time: 249 sec.
Epoch: 2 Test accuracy: 0.9713 Train accuracy: 0.978783 Calculation epoch time: 121 sec. All calculation time: 371 sec.
Epoch: 3 Test accuracy: 0.9717 Train accuracy: 0.98385 Calculation epoch time: 120 sec. All calculation time: 491 sec.
Epoch: 4 Test accuracy: 0.9726 Train accuracy: 0.985583 Calculation epoch time: 124 sec. All calculation time: 616 sec.
Epoch: 5 Test accuracy: 0.9778 Train accuracy: 0.990117 Calculation epoch time: 122 sec. All calculation time: 739 sec.
Epoch: 6 Test accuracy: 0.977 Train accuracy: 0.992 Calculation epoch time: 123 sec. All calculation time: 862 sec.
Epoch: 7 Test accuracy: 0.977 Train accuracy: 0.992183 Calculation epoch time: 131 sec. All calculation time: 993 sec.
Epoch: 8 Test accuracy: 0.977 Train accuracy: 0.99385 Calculation epoch time: 129 sec. All calculation time: 1122 sec.
Epoch: 9 Test accuracy: 0.9793 Train accuracy: 0.996617 Calculation epoch time: 117 sec. All calculation time: 1240 sec.
Epoch: 10 Test accuracy: 0.9794 Train accuracy: 0.99855 Calculation epoch time: 118 sec. All calculation time: 1359 sec.
Epoch: 11 Test accuracy: 0.9797 Train accuracy: 0.9987 Calculation epoch time: 118 sec. All calculation time: 1478 sec.
Epoch: 12 Test accuracy: 0.9763 Train accuracy: 0.997083 Calculation epoch time: 117 sec. All calculation time: 1595 sec.
Epoch: 13 Test accuracy: 0.9799 Train accuracy: 0.999667 Calculation epoch time: 118 sec. All calculation time: 1714 sec.
Epoch: 14 Test accuracy: 0.9798 Train accuracy: 0.999817 Calculation epoch time: 118 sec. All calculation time: 1832 sec.
Epoch: 15 Test accuracy: 0.9808 Train accuracy: 0.999917 Calculation epoch time: 118 sec. All calculation time: 1950 sec.
Epoch: 16 Test accuracy: 0.9804 Train accuracy: 0.999933 Calculation epoch time: 117 sec. All calculation time: 2068 sec.
Epoch: 17 Test accuracy: 0.98 Train accuracy: 0.999983 Calculation epoch time: 120 sec. All calculation time: 2189 sec.
Epoch: 18 Test accuracy: 0.9802 Train accuracy: 0.9997 Calculation epoch time: 121 sec. All calculation time: 2311 sec.
Epoch: 19 Test accuracy: 0.9801 Train accuracy: 0.999983 Calculation epoch time: 119 sec. All calculation time: 2430 sec.
```

Рис. 1. Пример запуска и вывод результатов

Результаты

Разработано приложение, позволяющее обучать и тестировать двухслойную нейронную сеть с использование набора данных MNIST.

Наилучшие результаты достигнуты с использование следующих параметров:

1. Число нейронов скрытого слоя – 300 нейронов
2. Число эпох – 25
3. Скорость обучения - 0.008
4. Начальные веса заданы с помощью равномерного распределения от 0 до 0.01
5. Точность на тестовой выборке – 0.9808
6. Точность на тренировочной выборке – 0.999917