Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Нижегородский государственный университет им. Н.И. Лобачевского»
Институт информационных технологий, математики и механики

Отчет по лабораторной работе

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

Выполнила Бастракова Ксения
Группа 381603м4

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). В качестве данных для обучения и тестирования сети необходимо использовать набор MNIST.

В работе решаются следующие задачи:

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. Разработка программной реализации.
4. Тестирование разработанной программной реализации на наборе данных MNIST.

Вывод математических формул

Рассматривается двухслойная полносвязная нейронная сеть с N нейронами входного слоя, K нейронами скрытого слоя и M нейронами выходного слоя [1].
Обозначим w набор весов на всех уровнях.

В качестве функции ошибки используется кросс энтропия [1]

$$E(w) = -\frac{1}{L}\sum_{k=1}^{L}\sum_{j=1}^{M} y_j^k \ln\left(u_j^k\right),$$

где $y^k = \left(y_j^k\right)$, $j = \overline{1,M}, k = \overline{1,L}$ – k-й пример из обучающей выборки, $u^k = \left(u_j^k\right), j = \overline{1,M}, k = \overline{1,L}$ – выход нейронной сети на соответствующем входе.

Для вывода, а также в программной реализации будем использовать предположение, что используется последовательный режим обучения. В этом режиме корректировка весов выполняется после обработки каждого элемента обучающей выборки.

Рассмотрим некоторый обучающий пример:

$$x^k = \left(x_j^k\right),$$
$$y^k = \left(y_j^k\right),$$
$$u^k = \left(u_j^k\right),$$

где x – вектор входных параметров, y – вектор результатов сети, u – вектор правильных результатов.

В этом случае функция ошибки примет следующий вид:

$$E(w) = \sum_{j=1}^{M} y_i \ln\left(u_j\right).$$

Обозначим $w_{js}^{(1)}$ веса синаптических связей от входных нейронов к нейронам скрытого слоя, $w_{sj}^{(2)}$ – веса от нейронов скрытого слоя к выходным нейронам нашей сети.

В качестве функции активации на скрытом слое – гиперболический тангенс:

$$\varphi(f_s) = \mathrm{th}(f_s).$$

Выходной сигнал нейрона скрытого слоя вычисляется как $v_s = \varphi(f_s)$. При этом f_s определяется как

$$f_s = \sum_{i=1}^{N} w_{is}^{(1)} x_i.$$

На последнем слое используется **SoftMax**-функция активации:

$$h(u_j) = \frac{e^{u_j}}{\sum_{j=1}^{M} e^{u_j}}$$

Сигнал выходного нейрона определяется как $u_j = h(g_j)$.

Запишем соответствующее значение кросс – энтропии:

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

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

Опишем метод обратного распространения ошибки: с помощью него определяются параметры сети w.

Запишем частную производную по каждому из весов:

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

$$\delta_j^{(2)} = \frac{\partial E}{\partial e^{u_j}} = (\sum_{j=1}^{M} y_j) \frac{e^{u_j}}{\sum_{m=1}^{M} e^{u_m}} - y_j = \frac{e^{u_j}}{\sum_{m=1}^{M} e^{u_m}} - y_j$$

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

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

На шаге $r+1$ обучения сети производится коррекция весов согласно направлению антиградиента по следующим формулам:

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

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

где η – скорость обучения ($0 < \eta < 1$).

Для гиперболического тангенса выражение принимает вид:

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

Программная реализация

Были реализованы следующие структуры и алгоритмы:

- **Neuron** – данный класс хранит набор синапсов и значение смещения. Инициализируется начальными весами.

- **NeuralNetwork –** класс имплементирующий нейронную сеть. Хранит множество нейронов по слоям.
- **LearningAlgorthm–** предназначен для тренировки, а так же анализа точности нейронной сети.

Сборка проекта осуществляется с помощью Visual Studio 2015.

Запуск производится следующим образом:

NeuralNetwork.exe [pathlearnImg] [pathtestImg][pathLearnLabls][pathTestLabls] [numHiddenLayers][learnRate][learnError]

Описание аргументов командной строки приведено ниже:

- [pathlearnImg] – путь до обучающей изображений обучающей выборки;
- [pathtestImg] – путь до изображений тестовой выборки;
- [pathLearnLabls] – путь для меток обучающей выборки;
- [pathTestLabls] – путь для меток тестовой выборки;
- [numHiddenLayers] – количество скрытых слоев (по умолчанию один);
- [learnRate] – скорость обучения (по умолчанию 0,9);
- [learnError] – значение для критерия остановки обучения (по умолчанию 0,001).

Результаты экспериментов

Тестирование проводилось на наборе данных MNIST dataset http://yann.lecun.com/exdb/mnist – объёмной базе данных образцов рукописного написания цифр. База данных MNIST содержит 60 000 изображений для обучения и 10 000 изображений для тестирования. Половина образцов для обучения и тестирования были взяты из набора NIST для обучения, а другая половина — из набора NIST для тестирования.

Начальные веса w инициализируются случайными значениями от -1 до 1. На вход программы подается массив двумерных массивов интенсивности (градации серого) для каждого пикселя изображения, а также массив меток, определяющий, к какой цифре относится каждое изображение.

Результаты тестирования на сетке с разным количеством нейронов на скрытом слое приведены в таблице 1.

Таблица 1. Результаты экспериментов.

Количество нейронов на скрытом слое	Скорость обучения	Ошибка обучения	Количество итераций	Ошибка на тестовой выборке
100	0,9	0,001	38	0,017
200	0,9	0,001	41	0,015
300	0,9	0,001	42	0,016
1000	0,9	0,001	29	0,015

Результаты эксперимента показывают, что при увеличении количества нейронов на скрытом слое количество итераций, необходимое для обучения, возрастает, но ошибка на тестовой выборке уменьшается.

Заключение

Изучена общая схема метода обратного распространения ошибки. Выведены математические формулы для вычисления градиентов по параметрам нейронной сети и формул коррекции весов. Разработана программная реализация двуслойной послносвязной нейронной сети и метода обратного распространения ошибки. Программная реализация протестирована на наборе данных MNIST.

Список литературы

1. Кустикова В.Д. Образовательный курс «Методы глубокого обучения для решения задач компьютерного зрения» Лекция №4 Сверточные нейронные сети. – Нижний Новгород, 2017.
2. Хайкин С. Нейронные сети. Полный курс. – М.: Издательский дом «Вильямс». – 2006.
3. Осовский С. Нейронные сети для обработки информации. – М.: Финансы и статистика. – 2002.