Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Нижегородский государственный университет им. Н.И. Лобачевского»
Исследовательский университет

Институт информационных технологий, математики и механики

Отчет по лабораторной работе

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

Выполнил: студент группы 381603м4

Умников Е.Д.

Нижний Новгород, 2017

Оглавление

Постановка задачи 3

Вывод математических формул 3

Алгоритм метода обратного распространения ошибки 4

Программная реализация 5

Результаты тестирования 5

Постановка 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. Проектирование и разработка программной реализации.
4. Тестирование разработанной программной реализации.

Вывод математических формул

Функции ошибки кросс энтропия:

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

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

$u^k = (u_j^k)_{j=\overline{1,M}}$ - выход нейронной сети.

SoftMax – функция активации на последнем слое:

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

Функция активации на скрытом слое – гиперболический тангенс.

$$\varphi(f_s) = th(f_s)$$

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

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

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

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

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

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

Для гиперболического тангенса:

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

Алгоритм метода обратного распространения ошибки

Шаг 1. Инициализация весов W (случайными небольшими значениями).

Шаг 2. До тех пор пока условие прекращения работы алгоритма неверно, выполняются Шаги 3 – 4.

Шаг 3. Прямой проход нейронной сети. На вход x_i. Выход последнего слоя u_j $j = \overline{1, M}$, M – количество классов.

Шаг 4. Обратный проход нейронной сети. Каждый выходной нейрон получает целевое значение, и вычисляются значения градиентов целевой функции. В обратном направлении происходит корректировка весов.

Критерий остановки:

-количество итераций метода (number of epochs) превосходит допустимое значение.

-достигнута необходимая точность E(w) < cross erros.

Программная реализация

Были реализованны следующие структуры и алгоритмы:

Neuron - данный класс хранит набор синапсов и значение смещения.

NeuralNetwork – класс имплементирующий двухслойную нейронную сеть. Содержит в себе два набора нейронов (std::vector<Neuron>).

BackPropagationMethod – предназначен для тренировки, а так же анализа точности нейронной сети. Для данного класса Neuron и NeuralNetwork являются friend с целью изменения параметров сети.

Результаты тестирования

Тестирование проводилось на наборе данных MNIST dataset http://yann.lecun.com/exdb/mnist. На Рис.1. показан пример работы программы.

```
D:\Study\Deap Learning\BackPropagation\x64\Release>BackPropagation.exe d://train/ d://test/
training...
Count of samples: 60000
Error: 0.173003
Continue training..
Error: 0.104108
Continue training..
Error: 0.0794636
Continue training..
Error: 0.0647102
Continue training..
Error: 0.0531042
Continue training..
Error: 0.0443408
Continue training..
Error: 0.0356792
Continue training..
Error: 0.0281902
Continue training..
Error: 0.0222211
Continue training..
Error: 0.0173254
Continue training..
Error: 0.0143856
Continue training..
Error: 0.0120832
Continue training..
Error: 0.00968448
Continue training..
Error: 0.0066102
Continue training..
Error: 0.00482141

Train result: 0.999383
Test result:0.9803
Для продолжения нажмите любую клавишу . . .
```

Рис.1. Пример работы программы.

После ряда экспериментов достигнута наилучшая точность при следующих параметрах работы:

-number of epochs = 30.
-cross error = 0.005.
-learn rate = 0.01
-number of neurons in hidden layer = 300

ошибка для тренировочных данных составила 0.999383, для тестовых данных 0.9803.