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: «Вычислительные методы и суперкомпьютерные технологии»

Образовательный курс «Глубокое обучение»

ОТЧЕТ

по практической работе №1

Реализация метода обратного распространения ошибки для двухслойного персептрона

Выполнила:
студентка группы 381703-3м
Крюкова Полина

Нижний Новгород
2018

Содержание

Цели

Изучить метод обратного распространения ошибки для обучения глубоких нейронных сетей на примере двухслойной полностью связанной сети (один скрытый слой).

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.
5. Подготовка отчета

Метод обратного распространения ошибки

Метод обратного распространения ошибки определяет стратегию выбора параметров сети w с использованием градиентных методов оптимизации в предположении, что целевая функция $E(w)$ непрерывна. Градиентные методы на каждом шаге уточняют значения параметров, по которым проводится оптимизация, согласно формуле:

$$w(k+1) = w(k)+\Delta w,$$

где $\Delta w=\eta p(w)$ определяет сдвиг значений параметров, $\eta, 0<\eta<1$ – ***скорость обучения*** – параметр обучения, который определяет «скорость» движения в направлении минимального значения функции, $p(w)$ – направление в многомерном пространстве параметров нейронной сети. В классическом методе обратного распространения ошибки направление движения совпадает с направлением антиградиента $p(w)=-\nabla E(w)$.

Общая схема метода обратного распространения ошибки включает несколько основных этапов. Первоначально синаптические веса сети инициализируются определенным образом, например, нулевыми значениями или случайно из некоторого распределения. Далее метод работает для каждого примера обучающей выборки.

1. Прямой проход нейронной сети в направлении передачи информации от входного сигнала к скрытым слоям и выходному слою сети. На данном этапе вычисляются значения выходных сигналов нейронов скрытых слоев и выходного слоя, а также соответствующие значения производных функций активации на каждом слое сети.
2. Вычисление значения целевой функции и градиента этой функции.
3. Обратный проход нейронной сети в направлении от выходного слоя к входному слою, и корректировка синаптических весов.
4. Повторение этапов 1 – 3 до момента выполнения критериев остановки. В качестве критериев остановки используется число итераций метода (количество проходов), либо достигнутая точность.

Рассмотрим процедуру вычисления производных и значений функции ошибки на примере двуслойной нейронной сети (рис. 5). В данном случае целевая функция записывается следующим образом:

$$E(w)=\frac{1}{2}\sum_{j=1}^{M}(y_j-u_j)^2=\frac{1}{2}\sum_{j=1}^{M}\left(y_j-\varphi^{(2)}\left(\sum_{s=0}^{K}w_{js}^{(2)}\varphi^{(1)}\left(\sum_{i=0}^{N}w_{si}^{(1)}x_i\right)\right)\right)^2$$

Здесь $wij^{(k)}$ – вес, связывающий i нейрон $(k-1)$-го слоя с j-ым нейроном k-го слоя, $\varphi(k)$ – функция активации всех нейронов k-го слоя.

Производная целевой функции по параметрам последнего слоя:

$$\frac{\partial E}{\partial w_{js}^{(2)}}=(y_j-u_j)\frac{d\varphi^{(2)}(g_j)}{dg_j}v_s=\delta_j^2 v_s,\quad g_j=\sum_{s=0}^{K}w_{js}^{(2)}v_s,\quad \delta_j^2=\partial E\frac{(w)}{\partial g_j}$$

Производная целевой функции по параметрам скрытого слоя:

$$\frac{\partial E}{\partial w_{si}^{(1)}}=\sum_{j=1}^{M}(y_j-u_j)\frac{d\varphi^{(2)}(g_j)}{dg_j}\frac{dg_j(v_s)}{dv_s}\frac{d\varphi^{(1)}(f_s)}{df_s}x_i=$$

$$\sum_{j=1}^{M}(y_j-u_j)\frac{d\varphi^{(2)}(g_j)}{dg_j}w_{js}^{(2)}\frac{d\varphi^{(1)}(f_s)}{df_s}x_i=\delta_s^{(1)}x_i,\quad f_s=\sum_{i=0}^{N}w_{si}^{(1)}x_i,\quad \delta_s^{(1)}=\partial E\frac{(w)}{\partial f_s}$$

Если $\varphi^{(1)}(u)=\frac{1}{1+e^{-u}}$ (сигмоидальная функция), а $\varphi^{(2)}(u_i)=\frac{e^{u_i}}{\sum_{k=1}^{M}e^{u_k}}$ (softmax), то:

$$\frac{\partial E}{\partial w_{js}^{(2)}}=-\frac{y_j}{u_j}\frac{e^{g_j}\left(\sum_{k=1}^{M}e^{g_k}-e^{g_j}\right)}{\left(\sum_{k=1}^{M}e^{g_k}\right)^2}v_s=\delta_j^{(2)}v_s$$

$$\frac{\partial E}{\partial w_{si}^{(1)}}=-\sum_{j=1}^{M}\frac{y_i}{u_j}\frac{e^{g_j-f_s}\left(\sum_{k=1}^{M}e^{g_k-g_j}\right)}{\left(\sum_{k=1}^{M}e^{g_k}\right)^2\left(1+e^{-f_s}\right)^2}w_{js}^{(2)}x_i=\delta_s^{(1)}x_i$$

Разработанная программа

Разработанная программа содержит следующие файлы

- вспомогательный скрипт get_training_data.py - скачивает тренировочные и тестовые данные из базы MNIST в формате gzip
- основной скрипт neural_network.py

Результаты экспериментов

Требуемое значение кросс-энтропии - 0.05.

Скорость обучения - 0.01.

Количество эпох - 10.

Результаты представлены в таблице:

Число нейронов скрытого слоя	Точность на тренировочном наборе	Точность на тестовом наборе
50	0.86131	0.8593
100	0.92335	0.9193
200	**0.**95106	0.9442