Домашнее задание № 4.

1. **Сравнительный анализ современных методов оптимизации (SGD, NAG, Adagrad, ADAM) на примере многослойного персептрона.**
2. **Использование генетического алгоритма для оптимизации гиперпараметров (число слоев и число нейронов) многослойного персептрона.**

1. Сравнительный анализ современных методов оптимизации (SGD, NAG, Adagrad, ADAM) на примере многослойного персептрона.

1.1. Stochastic gradient descent (SGD)

$$\boldsymbol{\theta = \theta - \eta \nabla_{\theta} J(\theta; x^{(i)}; y^{(i)})}$$

В отличии от традиционного градиентного метода, метод стохастического градиента позволяет обрабатывать за одну итерацию относительно небольшой набор обучающих данных из всего набора данных (выборки - batch).

Применение этого метода позволяет избежать недостатки традиционного градиентного метода:

1) Требуется значительно меньше необходимого объема памяти.
2) Повышается скорость сходимости.
3) Увеличивается робастность к невыпуклости целевых функций.
4) Возможна оптимизация многоэкстремальных целевых функций.

1.1.1. Mini-batch gradient descent

$$\theta = \theta - \eta \nabla\theta J(\theta; x\ i{:}i+n\ ;\ y\ (i{:}i+n)\ )$$

За каждую эпоху отбирается $\boldsymbol{n}$ данных ($\boldsymbol{n = 50 - 250}$) небольшой набор обучающих данных из всего набора данных (выборки - batch). Применение этого метода дает следующие преимущества:

1) Уменьшает дисперсию разброса параметров что ведет к более устойчивой сходимости.
2) Позволяет использовать матричные и тензорные операции, входящие в состав современных библиотек методов глубокого обучения Все это делает использование SGD на минибатчах очень эффективными для решения задач по глубокому обучению.

1.2. NAG. (Nesterov accelerated gradient)

$$vt = vt{-}1 + \eta \nabla\theta J(\theta - \gamma vt{-}1)$$
$$\theta = \theta - v$$

1.3. Adagrad. (Adapted gradient)

Оптимизатор первого порядка. Скорость обучения не постоянна, а зависит от целевой функции. На каждой итерации глобальная скорость обучения делится на $\boldsymbol{l_2}$ −норму прошлых градиентов вплоть до текущей.

Adagrad. (Adapted gradient)

$$g_{t,i} = \nabla_{\theta_t} J(\theta_{t,i})$$

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \varepsilon}} g_{t,i}$$

$$G_{t,ii} = G_{t-1,ii} + g_{t,i}^2$$

1.4. Adam. (Adaptive moment estimation)

Описание методов оптимизации можно найти в INTERNETe и в моей лекции № 4.

Adam. (Adaptive moment estimation)

$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t$.

$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\, g_t.$

$v_t = \beta_2 v_{t-1} + (1 - \beta_2)\, g_t^2.$

$\widehat{m_t} = \frac{m_t}{1 - \beta_1^t}.$

$\widehat{v_t} = \frac{v_t}{1 - \beta_2^t}.$

$\beta_1 = \beta_2 = 0.98.$

2. Генетический алгоритм.

Постановка задачи

Дана целевая функция $f(x) = f(x_1, x_2, ..., x_n)$, определенная на множестве допустимых решений $D \subseteq R^n$. Требуется найти глобальные минимумы заданных функций $f(x)$ на допустимом множестве D. То есть такую точку

$$x^e = Arg \min_{x \in D} f(x), \text{ где } x = (x_1, ..., x_n)^T, D = \{x | x_i \in [\alpha, \beta], i = 1, ..., n.\}. \qquad \textbf{(1)}$$

Задача поиска максимума целевой функции $f(X)$ сводится к задаче поиска минимума путем замены знака перед функцией на противоположный:

$$f(x^e) = \max_{x \in D} f(x) = -\min_{x \in D} [-f(x)].$$

2.1. Стратегия поиска

Генетические алгоритмы имитируют природные способы оптимизации, присущие процессам эволюции живых систем. А именно:

- генетическое наследование;
- изменчивость;
- естественный отбор.

Целевая функция $f(x)$ соответствует природному понятию ***приспособленности*** живого организма. Вектор переменных $x = (x_1, ..., x_n)^T$ целевой функции называется ***фенотипом***, а отдельные его параметры – ***признаками*** $i = 1, 2, ..., n$.

Любой живой организм может быть представлен своим генотипом и фенотипом.

Генотип – это совокупность наследственных признаков, информация о которых заключена в хромосомном наборе генов.

Фенотип – совокупность всех признаков и свойств организма, формирующихся в процессе взаимодействия его генотипа и внешней среды. Каждый ген имеет своё отражение в фенотипе.

Генетические алгоритмы ведут поиск решения на уровне генотипа. Каждую координату x_i вектора $x = (x_1,..., x_n)^T \in D$ представляют в некоторой форме s_i, удобной для использования в генетическом алгоритме и называется *геном*. Для этого необходимо выполнить преобразование, в общем случае не взаимно однозначное, вектора переменных $x = (x_1,..., x_n)^T \in D$ в некоторую структуру $s = (s_1, s_2,..., s_n)^T \in S$, называемую ***хромосомой (генотипом, особью):*** $D \xrightarrow{e} S$, где e функция кодирования, S - пространство представлений (как правило $D \neq S$).

Для того, чтобы восстанавливать по хромосоме решение, необходимо задать обратное преобразование $S \xrightarrow{e^{-1}} D$, где e^{-1} - функция декодирования.

В пространстве представлений S вводится так называемая ***функция приспособленности (функция фитнеса)*** $\mu(s) : S \xrightarrow{\mu} R$, где R - множество вещественных чисел, аналогичная целевой функции $f(x)$ на множестве D. Функция $\mu(s)$ может быть любая функция, удовлетворяющая условию:

$$\forall x^1, x^2 \in D : s^1 = e(x^1), s^2 = e(x^2), s^1 \neq s^2, \text{если } f(x^1) > f(x^2), \text{то } \mu(s^1) > \mu(s^2) .$$

При решении используются конечные наборы:

$$I = \left\{ s^k = (s_1^k, s_2^k,..., s_n^k)^T, k = 1,2,..., m \right\} \subset S$$

Возможных решений, называемых ***популяциями,*** где s^k - хромосомы с номером k, m - размер популяции, s_i^k - ген с номером i k - той популяции.

Затем осуществляется обратное преобразование:

$$x^e = e^{-1}(s^*).$$

Различают два способа кодирования:

3. Бинарное кодирование.
4. Вещественное кодирование.

Будем использовать второй вариант кодирования. В этом случае целевая функция может использоваться непосредственно как функция фитнеса. Тогда в качестве функции фитнеса $\mu(x)$ получается как преобразование целевой функции, т.е. *функция фитнеса* $\mu(x) : D \xrightarrow{\mu} R,$ где R - множество вещественных чисел, аналогична целевой функции $f(x)$.

Функцией $\mu(x)$ может быть любая функция, удовлетворяющая следующему условию:

$$\forall x^1, x^2 \in D : \; x^1 \neq x^2, \text{если } f(x^1) > f(x^2), \text{то } \mu(x^1) > \mu(x^2) .$$

Решение исходной оптимизационной задачи $f(x^*) = \min_{x \in D} f(x)$ сводится к поиску решения x_μ^* другой оптимизационной задачи:

$$\mu(x_\mu^*) = \max_{x \in D} \mu(x). \qquad (2)$$

В силу выбора функции $\mu(x)$, решение задач (1) и (2) (хромосома) совпадают:

$$x^e = x^* = Arg \min_{x \in D} f(x) = Arg \max_{x \in D} \mu(x) = x_\mu^*. \qquad (3)$$

При решении задачи (2) используются конечные наборы $I = \left\{x^k = \left(x_1^k,..., x_n^k\right)^T, k = 1,2,..., Mp\right\} \subset D$ возможных решений, называемых *популяциями*, где x^k - хромосома с номером k, M - размер популяции, x_i^k - ген с номером i.

2.3. Генетический алгоритм.

Ш.1. Формирование исходной популяции.

Ш.1.1. Задается номер популяции $t = 0$, максимальное количество популяций Np, номер итерации цикла $k = 1$, размер популяции Mp.

Ш.1.2. Случайным образом выбирается начальная точка x^0 - исходная хромосома. Она может быть выбрана как внутренняя точка гиперкуба области допустимых значений D. Из этой точки формируется исходная популяция. Для этого с помощью равномерного распределения на единичном отрезке $[0,1]$ Mp раз генерируется последовательность из n случайных точек $\{P_i^{0k}\}_{i=1,n}^{k-1,Mp}$, $i = 1,..., n; k = 1,..., Mp$. Используя линейное преобразование, каждая точка отображается на соответствующий ей промежуток $[\alpha, \beta]: P_i^k = (\beta_i - \alpha_i)P_i^{0k} + \alpha_i$. Составляя векторы из точек последовательности $\left\{P_i^k\right\}$ при фиксированных k, получаем Mp начальных векторов $x^k = \left(x_1^k,..., x_n^k\right)^T$, $x_i^k = P_i^k$, $i = 1,..., n$, координаты которых x_i имеют равномерное распределение на отрезках $[\alpha_i, \beta_i]$, $i = 1,..., n$. Таким образом может быть сформирована начальная популяция $I_0 = \left\{x^k, k = 1,..., Mp \,\middle|\, x^k = \left(x_1^k, x_2^k,..., x_n^k\right) \in D\right\}$.

Ш.1.3. Вычисляется значение функции фитнеса для каждой особи $x^k \in I_0 : \mu_k = \mu(x^k), k = 1,..., Mp$ и популяции I_0 в целом $\mu = \sum_{k=1}^{Mp} \mu_i$.

Ш.2. Отбор (селекция).

Селекция – это операция, которая осуществляет отбор особь (хромосом) x^k в соответствии со значениями функции фитнеса $\mu(x^k)$ для последующего их скрещивания.

Ш.2.1. Вычислить кумулятивную вероятность $q_i = \sum_{j=1}^{i} \mu_j(x^j), i = 1,2,..., Mp$.

Ш.2.2. Сформировать случайное действительное число r в интервале $(0, Mp]$.

Ш.2.3. Выбрать i - ю хромосому x^i $(1 \le i \le Mp)$ так, чтобы $q_{i-1} < r \le q_i$.

Ш.2.4. Перейти на Ш.2.2. до тех пор, пока не будет сформирована новая популяция ($while(i \le Mp)$).

Ш.3. Кроссинговер (скрещивание).

Скрещивание – это операция, при которой из нескольких, обычно двух хромосом (особей), называемых родителями, порождается одна или несколько новых, называемых потомками.

Ш.3.1. Определяется параметр $Pc \in (0,1]$ как вероятность кроссинговера. Эта вероятность дает ожидаемое число $Pc \cdot Mp$ хромосом, подвергаемых операции кроссинговера.

Ш.3.2. Для операции кроссинговера выполняется процесс, повторяющийся от $i = 1$ до $Pc \cdot Mp$: формируется случайное действительное число r из сегмента $[0,1]$, при этом, если $r < Pc$, то хромосома x^i выбирается как родительская.

Ш.3.3. Отбираются пары родительских хромосом (x^i, x^j), где $i \neq j$. Действие оператора кроссинговера осуществляется следующим образом:

Ш.3.4. Формируется случайное число $c \in (0,1)$, затем оператор кроссинговера, действующий на исходные пары (x^i, x^j), производит две хромосомы потомки X и Y:

$$X = c \cdot x^i + (1-c) \cdot x^j, \quad Y = (1-c) \cdot x^i + c \cdot x^j.$$

Ш.3.5. Если допустимое множество является выпуклым, то кроссинговер обеспечивает допустимость обоих потомков, в случае если допустимы оба родителя. Следует проверить допустимость каждого потомка перед тем, как он будет включен в новую популяцию. Если оба потомка являются допустимыми, тогда родители заменяются этими потомками. Если это не так, сохраняется допустимый потомок, если он существует, а затем вновь выполняется оператор кроссинговера с новым значением случайного числа c до тех пор, пока не будут получены два новых допустимых потомка или не будет превышено заданное число циклов. В этом случае осуществляется замена родителей только теми (сохраненными ранее) потомками, которые оказались допустимыми.

Ш.4. Мутация.

Мутация – это преобразование хромосомы , случайно изменяющее один или несколько из её генов. Оператор мутации предназначен для того, чтобы поддерживать разнообразие особей в популяции.

Ш.4.1. Определим параметр $Pm \in (0,1]$ как вероятность мутации. Эта вероятность дает ожидаемое число $Pm \cdot Mp$ хромосом, подвергаемых операции мутации.

Ш.4.2. Для операции мутации выполняется процесс, повторяющийся от $i = 1$ до $Pm \cdot Mp$: формируется случайное действительное число r из сегмента $[0,1]$, при этом, если $r < Pm$, то хромосома x^i выбирается как родительская для операции мутации Для каждой выбранной родительской хромосомы x^i, обозначенной как $Z = (x_1, x_2, ..., x_n)$, производится мутация.

Ш.4.3. Поочередно рассматривается каждый потомок из ожидаемого числа $Pm \cdot Mp$ хромосом. Среди генов выбранной родительской хромосомы $Z = (x_1, x_2, ..., x_n)$ случайно (с вероятностью $1/n$) выбирается один с номером $p \in (1,2,..., n)$ подлежащий замене. Его новое значение x_p^M случайным образом выбирается из промежутка $[\alpha_p, \beta_p]$ изменения выбранной координаты x_p.

Ш.5. Формирование новой популяции.

Ш.5.1. С равной вероятностью из потомков мутантов предыдущего шага выбирается один $x^M = (x_1, x_2, ..., x_p^M, ..., x_n)$.

Ш.5.2. Выбранный потомок добавляется в популяцию вместо хромосомы, которой соответствует наименьшее значение функции фитнеса (наихудшее из допустимых значений).

Ш.5.3 Вычисляется значение функции фитнеса для мутантного потомка $\mu_M = \mu(x^M)$.

Ш.5.4. Проверка условий:

Ш.5.4.1. Если $k < Mp$, то $k = k + 1$ и переход на Ш.2.

Ш.5.4.2. Если $k = Mp$, то $t = t + 1$ и переход на Ш.6.

Ш.6. Проверка условия останова генетического алгоритма.

Условием окончания работы генетического алгоритма является формирование заданного количества популяций $t = Np$.

Ш.6.1. Если условие не выполнено, то полагаем $k = 1$ и переход на Ш.2.

Ш.6.2. Если условие окончания работы выполнено, то в качестве решения (приближенного) задачи $\mu(x_\mu^*) = \max\limits_{x \in D} \mu(x)$ выбирается особь с лучшим значением функции фитнеса из текущей популяции: $x_\mu^* \cong x_\mu^e = Arg \max \mu(x^k)$, а по нему определяется приближенное решение поставленной задачи $f(x^*) = \min f(x) : x^* = x_\mu^*$.

Замечание 1. В качестве *функции фитнеса* можно использовать обратную целевую функцию $\mu(x) = \dfrac{1}{f(x)}$.

Замечание 2. Обычно размер популяции выбирают в пределах 30-60 особей.

Замечание 3. Вероятность кроссинговера принимается равной $Pc = 0.3 - 0.5$, вероятность мутации $Pm = 0.05 - 0.2$.