ФИЛЬТРАЦИЯ ЦИФРОВЫХ ИЗОБРАЖЕНИЙ НА ОСНОВЕ АВТОЭНКОДЕРА

Яна Акмаева
студент группы М80-105М

1 Введение в понятие автоэнкодера

Автоэнкодер (автоассоциатор) представляет собой специальную нейронную сеть прямого распространения, которая обучается без учителя, то есть, не используя размеченную базу данных. В этом случае нейронная сеть подбирает значения выходов таким образом, чтобы они были как можно ближе к значениям на входе сети. На рис. 1 представлена модель трехслойного автоэнкодера (один входной слой, один скрытый и один выходной). Входной слой содержит шесть входных единиц (нейронов) и одну единицу смещения, скрытый слой содержит три скрытые единицы и одну единицу смещения, выходной слой содержит такое же число единиц, что и входной.

2 Принцип работы автоэнкодера

Автоэнкодер пытается получить функцию (гипотезу) $h_{w,b}(x) \approx x$, которая аппроксимирует вход x оценкой $\hat{x}$ на выходе, как показано на рис.1. Для того, чтобы преобразование не было тривиальным, на скрытый слой автоэнкодера накладывают ограничение. Число единиц в скрытом слое должно быть меньше, чем во входном.

Ограничение числа нейронов в скрытом слое позволяет получить сжатое представление входных данных, в смысле понижения их размерности. Это означает, что автоэнкодер обучается восстанавливать входные данные из сжатого представления, что может быть полезно для реализации алгоритма шумоподавления.

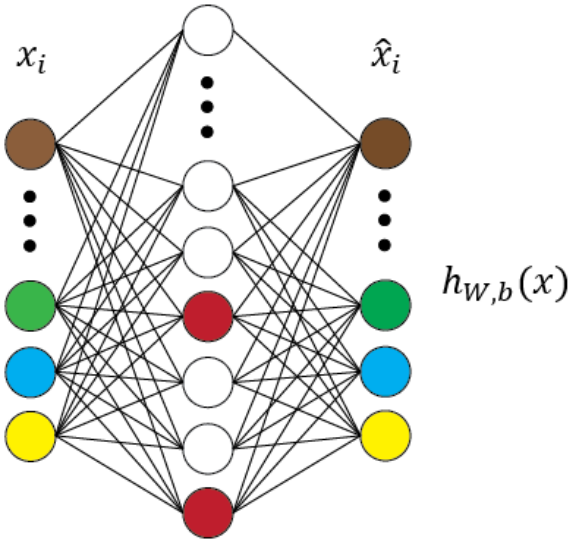

Рис. 1: Автоэнкодер

Обучение такой нейронной сети выполняется с использованием метода обратного распространения хошибки путем минимизации стоимостной функции, например, с помощью градиентного спуска, позволяющего настроить веса (параметры модели) автоэнкодера и получить гипотезу $h_{w,b}(x) \approx x$.

3 Алгоритм

Рассмотрим алгоритм шумоподавления на основе модели трехслойного автоэнкодера, пример которого приведен выше. На вход алгоритма поступает зашумленное аддитивным белым гауссовским шумом n с нулевым математическим ожиданием и известной дисперсией σ^2 изображение $y = x + n$.

Обозначим через $m(i)$ число единиц в конкретном слое автоэнкодера, где индекс $i = 1, 2, 3$ обозначает номер слоя сети.

Алгоритм.

1. Выделяем на зашумленном изображении y совокупность перекрывающихся блоков, каждый из которых состоит из области обучения, области фильтрации и области наложения блоков.

2. Для обучения автоэнкодера необходимы данные, взятые из зашумленного изображения, поэтому требуется сформировать матрицу данных S_y, столбцами которой являются всевозможные векторы обучения,

набранные внутри области обучения. При формировании векторов обучения внутри области обучения выбираются всевозможные блоки размером $|x|$, которые затем представляются в виде векторов-столбцов размером l^2. Матрица данных S_{y} имеет размер $l^2 \times n$, где l^2 – размер векторов обучения, то есть размерность данных, n – количество набранных векторов обучения.

3. Подготавливаем обрабатываемые данные, состоящие из яркостей пикселей изображения y, для обучения автоэнкодера, а именно, производим масштабирование данных в S_{y} путем деления их на максимальное значение яркости пикселя в зашумленном изображении. Полученную матрицу обозначим через $\overline{S_y}$.

4. Используем модель автоэнкодера, у которого число входных единиц $m^{(1)}$ в первом слое будет равно размеру вектора обучения l^2, то есть число нейронов во входном слое зависит от размерности данных в матрице $\overline{S_y}$. Количество нейронов в скрытом слое должно быть больше количества нейронов во входном слое ($m^{(1)} < m^{(2)}$).

5. Выполняем алгоритм прямого распространения сигнала, который в данном случае описывается следующими выражениями, представленными в матричной форме:

$$a_j^{(2)} = f(z_j^{(2)}) = f(W^{(1)}(\overline{S_y})_j + b^{(1)}), \tag{1}$$

$$a_j^{(3)} = h_{w,b}((\overline{S_y})_j) = z_j^{(3)} = W^{(2)}a_j^{(2)} + b^{(2)}. \tag{2}$$

Активации единиц скрытого слоя автоэнкодера получаются при использовании нелинейного преобразования f, основанного на сигмоидной функции, а активации нейронов на выходе сети – при помощи линейного преобразования, определяющего линейный декодер. В выражениях (1) и (2) $(\overline{S_y})_j$ – j-й вектор-столбец матрицы $\overline{S_y}$, $a_j^{(2)}$ – вектор-столбец размерностью $m^{(2)}$ рассчитанных активаций нейронов скрытого слоя автоэнкодера, $a_j^{(3)}$ – вектор-столбец размерностью $m^{(1)}$ рассчитанных активаций нейронов выходного слоя, $W^{(1)}$ – матрица размерностью $m^{(2)} \times m^{(1)}$, содержащая параметры модели и обеспечивающая связь между нейронами входного и скрытого слоев автоэнкодера, $W^{(2)}$ – матрица размерностью $m^{(1)} \times m^{(2)}$, содержащая параметры модели и обеспечивающая связь между нейронами скрытого и выходного слоев автоэнкодера, $b^{(1)}$ и $b^{(2)}$ – векторы-столбцы размером $m^{(2)}$ и $m^{(1)}$ соответственно, содержащие элементы смещения для нейронов скрытого и выходного слоев.

Необходимо отметить, что исходно веса нейронной сети инициализируются случайным образом – для того чтобы избежать проблемы симметричности весов.

6. Получив начальную гипотезу $h_{w,b}((\overline{S_y})_j)$, определяем следующую стоимостную функцию, посчитанную по всей выборке векторов обучения в матрице $\overline{S_y}$:

$$J(W,b) = [\frac{1}{2n}\sum_{\text{o}=1}^{n} ||h_{w,b}((\overline{S_y})_j) - (\overline{S_y})_j||_2^2] + \\ + \frac{\lambda}{2}[\sum_{j=1}^{m^{(2)}}\sum_{k=1}^{m^{(1)}}(W_j k^{(1)})^2 + \sum_{j=1}^{m^{(1)}}\sum_{k=1}^{m^{(2)}}(W_j k^{(2)})^2 + \\ + \sum_{j=1}^{m^{(2)}}(b_j^{(1)})^2 + \sum_{j=1}^{m^{(1)}}(b_j^{(2)})^2], \qquad (3)$$

где $||*|| - L^2$-норма, а λ – параметр регуляризации, контролирующий сложность модели.

7. Производим поиск частных производных функции (3) по параметрам $W^{(1)}, W^{(2)}, b^{(1)}, b^{(2)}$ с использованием алгоритма обратного распространения ошибки.

8. Получив необходимые частные производные функции (3), осуществляем ее минимизацию по параметрам сети $W^{(1)}, W^{(2)}, b^{(1)}, b^{(2)}$ при помощи алгоритма численной оптимизации Бройдена–Флетчера–Гольдфарба–Шанно с ограниченным использованием памяти.

9. После минимизации стоимостной функции получаем параметры обученной сети $W^{(1)}, W^{(2)}, b^{(1)}, b^{(2)}$. Пропускаем через автоэнкодер зашумленные данные из матрицы $\overline{S_y}$, выполняя алгоритм прямого распространения (1) и (2) с использованием обновленных параметров сети. В итоге получаем оценку $\hat{S}_x$ матрицы незашумленных данных S_x. Далее преобразовываем векторы обучения из столбцов матрицы $\hat{S}_x$ обратно в блоки размерами $l \times l$, вкладывая их в соответствующие пространственные позиции области обучения. Область наложения блоков усредняется арифметически. Получаем восстановленную область обучения, внутри которой выделяем область фильтрации. Обучая автоэнкодер и обрабатывая им все оставшиеся области фильтрации на изображении y, а затем вкладывая их в соответствующие пространственные позиции, получим оценку $\hat{x}$ неискаженного изображения x. Области фильтрации вкладываются с наложением, поэтому область наложения необходимо арифметически усреднить, чтобы устранить артефакты блочности.

4 Алгоритм Бройдена — Флетчера — Гольдфарба — Шанно

Пусть задана некоторая функция и мы решаем задачу оптимизации: $minf(s,y)$. Где в общем случае $f(x,y)$ является не выпуклой функцией, которая имеет непрерывные вторые производные.

Шаг №1.

Инициализируем начальную точку x_0; Задаем точность поиска $\epsilon > 0$; Определяем начальное приближение $H_0 = B_0^{-1}$, где B_0^{-1} — обратный гессиан функции;

Каким нужно выбрать начальное приближение H_0? К сожалению не существует общей формулы, которая хорошо бы работала во всех случаях. В качестве начального приближения можно взять гессиан функции, вычисленный в начальной точке x_0. Иначе можно использовать хорошо обусловленную, невырожденную матрицу, на практике часто берут единичную матрицу.

Шаг №2.

Находим точку, в направлении которой будем производить поиск, она определяется следующим образом:

$$p_k = -H_k \times \nabla f_k$$

Шаг №3.

Вычисляем x_{k+1} через рекуррентное соотношение:

$$x_{k+1} = x_k + \alpha_k \times p_k$$

Коэффициент α_k находим, используя линейный поиск, где α_k удовлетворяет условиям Вольфе (Wolfe conditions):

$$f(x_k + \alpha_k \times p_k) \leqslant f(x_k) + c_1 \times \alpha_k \times \nabla f_k^T \times p_k$$

$$\nabla f(x_k + \alpha_k \times p_k)^T \geqslant c_2 \times \nabla f_k^T \times p_k$$

Константы c_1 и c_2 выбирают следующим образом: $0 \leqslant c_1 \leqslant c_2 \leqslant 1$. В большинстве реализаций: $c_1 = 0.0001$ и $c_2 = 0.9$.

Фактически мы находим такое α_k, при котором значение функции $f(x_k + \alpha_k \times p_k)$ минимально.

Шаг №4.

Определяем векторы:

$$s_k = x_{k+1} - x_k$$

$$y_k = \nabla f_{k+1} - \nabla f_k$$

s_k — шаг алгоритма на итерации, y_k — изменение градиента на итерации.

Шаг №5.

Обновляем гессиан функции, согласно следующей формуле:

$$H_{k+1} = (I - p_k \times s_k \times y_k^T) H_k (I - p_k \times y_k \times s_k^T) + p \times s_k \times s_k^T$$

где p_k

$$p_k = \frac{1}{y_k^T s_k}$$

I — единичная матрица.

Шаг №6.

Алгоритм продолжает выполнятся до тех пор пока истинно неравенство: $|\nabla f_k| > \epsilon$.

5 Выводы

Основным недостатком представленного алгоритма является высокая вычислительная сложность, обусловленная тем, что необходимо проводить обучение автоэнкодера для обработки каждой области фильтрации на зашумленном изображении. Ускорить работу алгоритма можно при помощи распараллеливания кода и его выполнения на графическом процессоре видеокарты.

Основным достоинством алгоритма является возможность качественного восстановления высокотекстурированных областей изображения из зашумленных данных.