Вики

Алгоритм обратного распространения ошибки (Back propagation algorithm)

> Синонимы: Алгоритм BackProp, Алгоритм Back Propagation, BackProp
> Разделы: Алгоритмы
> Loginom: Нейросеть (классификация) (обработчик), Нейросеть (регрессия) (обработчик)

Алгоритм обратного распространения ошибки — популярный алгоритм обучения плоскослоистых нейронных сетей прямого распространения (многослойных персептронов). Относится к методам обучения с учителем, поэтому требует, чтобы в обучающих примерах были заданы целевые значения. Также является одним из наиболее известных алгоритмов машинного обучения.

В основе идеи алгоритма лежит использование выходной ошибки нейронной сети

$$E = \frac{1}{2}\sum_{i=1}^{k}(y - y')^2$$

для вычисления величин коррекции весов нейронов в ее скрытых слоях, где k — число выходных нейронов сети, y — целевое значение, y' — фактическое выходное значение. Алгоритм является итеративным и использует принцип обучения «по шагам» (обучение в режиме on-line), когда веса нейронов сети корректируются после подачи на ее вход одного обучающего примера.

На каждой итерации происходит два прохода сети — прямой и обратный. На прямом входной вектор распространяется от входов сети к ее выходам и формирует некоторый выходной вектор, соответствующий текущему (фактическому) состоянию весов. Затем вычисляется ошибка нейронной сети как разность между фактическим и целевым значениями. На обратном проходе эта ошибка распространяется от выхода сети к ее входам, и производится коррекция весов нейронов в соответствии с правилом:

$$\Delta w_{j,i}(n) = -\eta\frac{\partial E_{av}}{\partial w_{ij}}, (1)$$

где $w_{j,i}$ — вес i-й связи j-го нейрона, η — параметр скорости обучения, который позволяет дополнительно управлять величиной шага коррекции $\Delta w_{j,i}$ с целью более точной настройки на минимум ошибки и подбирается экспериментально в процессе обучения (изменяется в интервале от 0 до 1).

Учитывая, что выходная сумма j-го нейрона равна

$$S_j = \sum_{i=1}^{n} w_{ij}x_i,$$

можно показать, что

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial S_j}\frac{\partial S_j}{\partial w_{ij}} = x_i\frac{\partial E}{\partial S_j}.$$

Из последнего выражения следует, что дифференциал ∂S_j активационной функции нейронов сети $f(s)$ должен существовать и не быть равным нулю в любой точке, т.е. активационная функция должна быть дефференцируема на всей числовой оси. Поэтому для применения метода обратного распространения используют сигмоидальные активационные функции, например, логистическую или гиперболический тангенс.

Таким образом, алгоритм использует так называемый стохастический градиентный спуск, «продвигаясь» в многомерном пространстве весов в направлении антиградиента с целью достичь минимума функции ошибки.

На практике обучение продолжают не до точной настройки сети на минимум функции ошибки, а до тех пор, пока не будет достигнуто достаточно точное его приближение. Это позволит, с одной стороны, уменьшить число итераций обучения, а с другой — избежать переобучения сети.

В настоящее время разработано множество модификаций алгоритма обратного распространения. Например, используется обучение не «по шагам» (когда выходная ошибка вычисляется, а веса корректируются на каждом примере), а «по эпохам» в режиме off-line (когда изменение весов производится после подачи на вход сети всех примеров обучающего множества, а ошибка усредняется по всем примерам).

Обучение «по эпохам» является более устойчивым к выбросам и аномальным значениям целевой переменной за счет усреднения ошибки по многим примерам. Но при этом повышается вероятность «застревания» алгоритма в локальных минимумах. Вероятность этого для обучения «по шагам» меньше, поскольку использование отдельных примеров создает «шум», который «выталкивает» алгоритм из ям градиентного рельефа.

К преимуществам алгоритма обратного распространения ошибки относятся простота реализации и устойчивость к аномалиям и выбросам в данных. К недостаткам можно отнести:

- неопределенно долгий процесс обучения;
- возможность «паралича сети», когда при больших значениях рабочая точка активационной функции оказывается в области насыщения сигмоиды и производная в выражении (1) становится близкой к 0, а коррекции весов практически не происходит и процесс обучения «замирает»;
- уязвимость алгоритма к попаданию в локальные минимумы функции ошибки.

Впервые алгоритм был описан в 1974 г. А.И. Галушкиным, а также, независимо и одновременно, Полом Дж. Вербосом. Далее существенно развит в 1986 г. Дэвидом И. Румельхартом.