АЛЬТЕРНАТИВНЫЙ СПОСОБ ОБУЧЕНИЯ МНОГОСЛОЙНЫХ НЕЙРОСЕТЕЙ ПРЯМОГО РАСПРОСТРАНЕНИЯ В ЗАДАЧЕ РАСПОЗНАВАНИЯ ОБРАЗОВ

Колентьев С.В.

Саратовский государственный технический университет

На сегодня наиболее часто используемый в целях обучения сети алгоритм обратного распространения страдает своей медлительностью и не гарантированным успехом распознавания искаженных образов при цифровой обработке сигналов. Кроме того, остается неочевидной зависимость между количеством нейронов в скрытых слоях модели и количеством распознаваемых сетью образов, а также точностью их распознавания. Также, сеть в результате обучения с помощью алгоритма обратного распространения (и многочисленных его модификаций), на этапе распознавания образов в сигнале начинает вести себя непредсказуемо. В результате на выходе сети при распознавании появляются образы, которым ее никто не обучал или не распознаются сигналы очень похожие на элементы обучающей выборки. При задании более строгих ограничений на качество обучения, сам процесс тренировки сети может затянуться на неопределенный срок.

Предлагается алгоритм, позволяющий избежать всех этих неприятностей, а кроме того, заранее, на этапе проектирования модели, знать сколько потребуется в ней нейронов для конкретного объема обучающих образов.

Алгоритм содержит элементы обучения, чем-то похожие на методы встречного распространения.

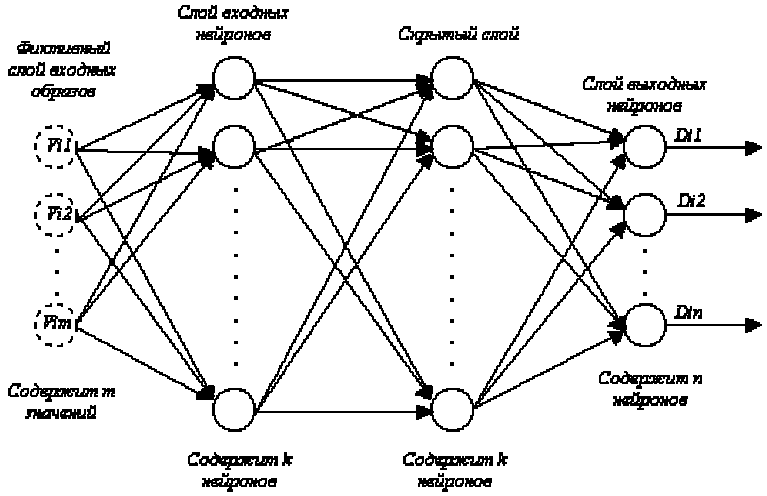

Рис. 1. Топология нейросети.

Метод обратного распространения ошибки [1] (back propagation) основан, фактически на многомерной антиградиентной оптимизации синаптических весов сети в соответствии с набором обучающей выборки – пар входных образов и соответствующих им желаемых выходных значений сети. Метод же встречного распространения [2] (counter propagation) основан на отыскании корреляции пар этих образов обучающей выборки (входных и соответствующих им «желаемых» выходных).

Рассмотрим ситуацию, когда у нас имеется k образов в обучающей выборке. Причем каждому «входному» образу Vi соответствует один выходной образ Di $(i = 1..k)$. В каждом входном образе содержится m элементов (Vij – j-тый элемент i-го входного образа), а в каждом выходном образе содержится n элементов (Dij – j-тый элемент i-го выходного образа).

Задача распознавания сети сводится к тому, чтобы сопоставить искаженному входному образу Pi, похожему на один из входных образов, соответствующий ему «желаемый» выходной образ, причем это будет не обязательно один из выходных образов обучающей выборки, а скорее комбинация из нескольких Di, в соответствии со степенью искажения Pi по отношению к Vi.

Рассмотрим, также топологию сети (см. рис. 1). Это обычный «персептрон» – многослойная сеть прямого распространения.

Для каждого синаптического веса w_{ij} будем полагать, что i – это индекс нейрона из предыдущего слоя, а j – это индекс нейрона, которому принадлежит вес w_{ij}.

В таком случае этап обучения состоит из шагов:

1. Синаптические веса нейронов входного слоя принимают значения в соответствии с выражением $w_{ij} = Vji$. Это означает, что веса содержат в себе «следы» входных образов обучающей выборки.
2. Синаптические веса нейронов выходного слоя принимают значения в соответствии с выражением $w_{ij} = Dij$. Это означает, что веса содержат в себе «следы» выходных образов обучающей выборки.
3. Синаптические веса нейронов скрытого слоя принимают значения, по суть, коэффициентов корреляции элементов входных и выходных образов. Выражение для них имеет вид:

$$\begin{cases} w_{jj} = \dfrac{1.0}{\sum\limits_{k=1}^{m} V_{kj}^2} \\ w_{ij} = -\dfrac{1.0}{\sum\limits_{k=1}^{m} \left(V_{ki} \cdot V_{kj}\right)} \end{cases}, \text{ где } i, j = 1..k$$

Первая из этих формул, по сути, определяет коэффициент присутствия *j*–го образа во входных значениях сети, тогда как вторая определяет коэффициент «похожести» входного образа на *i*–тый образ.

Выходное значение каждого нейрона рассчитывается в соответствии с выражением: $Out_j = F\left(\sum\limits_{i=1}^{p}\left(In_i \cdot w_{ij}\right) - u\right)$, где F – функция ограничения, u – порог, In_i – входные значения для нейрона, w_{ij} – его синаптические веса, p – количество входов нейрона, а i – номер нейрона в слое.

Как правило, выбор функции ограничения остается на усмотрение разработчика, так что все приведенные здесь выражения для подсчета начальных значений весов взяты без ее учета. Порог чаще всего в практических моделях тоже не используется. Собственно говоря, данная модель может использоваться как прослойка в «более многослойной» сети произвольной топологии. Если использовать порог или ограничительную функцию в этой модели, то необходимо подкорректировать начальные значения весов w_{ij} в вышеуказанных трех пунктах.

Итак, первый слой нейронов вычисляет коэффициенты корреляции заданного в процессе распознавания входного образа *Pi* с каждым входным образом *Vi* из «эталонной» обучающей выборки.

Далее по этим коэффициентам скрытый слой вычисляет коэффициенты присутствия во входных значениях *Pi* каждого из *k* входных образов *Vi* обучающей выборки (коэффициент «присутствия» – это, по сути, выход каждого из нейронов скрытого слоя).

И уже по этим данным выходной слой подсчитывает реальный выход сети, скомбинированный из всех выходных образов *Di* обучающей выборки, в соответствии с коэффициентом присутствия каждого из них.

Между тем, гарантировано, что при появлении образов *Pi*, максимально численно «похожих» на *Vi*, выходы сети будут максимально похожи на *Di*.

Как видно из рис. 1, на всю модель требуется $2 \cdot k + n$ нейронов с $k \cdot (k + m + n)$ синаптическими связями. Это, конечно, требует немалых затрат оперативной памяти, однако взамен мы получаем возможность обучить ее всем нужным образам выборки за один раз, не прибегая к итерационным процессам.

Самое удобное, что после этого данную модель можно дообучать по своему усмотрению любым доступным методом, в том числе и методом обратного распространения ошибки.

Кроме того, дообучение можно тоже оптимизировать. Как видно из вышеописанного, модель требует значительных затрат памяти (в основном на скрытый слой). Вследствие этого, при дообучении, когда появился новый образ, которому нужно натренировать сеть, мы можем пойти двумя путями:

1. Если нехватки оперативной памяти у нас нет или сравнительно мало образов в обучающей выборке, то просто добавляем новый образ как самостоятельный. При этом в скрытый слой добавляется еще один нейрон; добавляются синаптические связи с ним в входной, скрытый и выходной слои, веса в которых устанавливаются в соответствии с выражениями для инициализации соответствующих w_{ij}, описанных выше.
2. Если же оперативная память в дефиците или количество образов в обучающей выборке число немалое, то можно поступить по аналогии с обучением карт Кохонена, а именно: никаких нейронов не добавлять, а сместить «акцент распознавания» в пользу того (или тех) образов *Vi*, на которые похож новый (добавляемый) образ *Pi*, пропорционально его «похожести». Достигается это путем умножения диагональных весов синаптической матрицы скрытого слоя на

величину этой самой «похожести». Эту «похожесть» можно рассчитывать по разному. Кроме того, можно не умножать все диагональные значения весов w_{jj} , а посчитать их сумму постоянной величиной и перераспределить эти значения весов в соответствии с коэффициентом схожести.

Предложенный алгоритм позволяет снизить временные затраты на обучение нейросети и, кроме того, может использоваться в задачах распознавания образов и обработки цифровых сигналов, в частности их экстраполяции или восстановления искаженных («зашумленных») участков.

Литература

1. Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика. – М.:Мир, 1992. – 301 с.
2. Hecht-Nielsen R. Counterpropagation Networks: Proceedings of the IEEE International Conference on Neural Networks, II, IEEE Press, New York, NY, pp. 19-32, 1987

ALTERNATIVE WAY OF TRAINING MULTILAYER FEEDFORWARD NEURAL NETS IN A TASKS OF PATTERN RECOGNITION

Kolentev S.

The algorithm of back propagation that used very often in last days has some serious disadvantages. Such as lower speed of training iteration process, not guaranteed successful recognition of patterns, that net was already trained, during the digital signal processing. Besides, the dependence between a number of neurons in a net and number of training set patterns is not obvious. Quality of recognition is also uncontrollable.

Net after a training process by algorithm of back propagation (and it's variants) behaves itself unpredictable during a recognition. For example it can converge results to a patterns which it has never trained before, or in another way – it can't recognize a pattern that it was trained. And if we will set up a more restriction for quality of training process, the time of iterations will be unpredictable.

We presenting the algorithm, that allow us to avoid all these troubles, and besides, beforehand, on stage of designing models, we'll know how much neurons in it's structure must be, depending on a number of training patterns. This algorithm contains elements, that look like counter propagation methods.

The algorithm of back propagation [1] based on multidimensional antigradient optimization of neural net synaptic weights according to a set of training patterns (pairs of input and corresponding desired outputs). Method of counter propagation [2] based on calculation of correlation of those pairs of patterns (pairs of input and corresponding desired outputs).

Let us consider that we have training set with k pairs of patterns and each input pattern Vi contains m elements, each output desired pattern Di consist of n elements. Each one input pattern Vi corresponds to one of output patterns Di. Thus the task of recognition is follow: to match noised («damaged») signal pattern Pi with all trained set input patterns Vi depending on their similarity to Pi and combine the output signal with several Di, corresponding these similarities.

The main idea is that: the first layer outputs contains a correlation coefficients of processing pattern Pi and each input pattern Vi of training set. After that by these coefficients the hidden layer calculates a coefficient of influence of each Vi on output result. And finally we get on outputs of output layer a combination of all output training patterns Di, depending on these influence coefficients.

Thus the resulting model of network will be contain $2 \cdot k + n$ neurons and $k \cdot (k + m + n)$ synaptic connections. And the training process will not include any iteration processes.

Another advantage of this algorithm – the net after that can be re-trained even by method of back propagation, in case of need, to add a new patterns in net's knowledge base.

Described algorithm allow us to lower time of neural network training and, besides, can be used in a tasks of pattern recognition and digital signal processing, in such tasks that prediction of time series or restoring of noised signals.

REFERENCES

1. Uosserman F. Neurocomputer techniques: Theory and practice. – M.:Mir, 1992. – p. 301.
2. Hecht-Nielsen R. Counterpropagation Networks: Proceedings of the IEEE International Conference on Neural Networks, II, IEEE Press, New York, NY, pp. 19-32, 1987