Глава 8.

Нейронные сети ассоциативной памяти

Е.М.Миркес

Вычислительный центр СО РАН в г. Красноярске[1]

Рассматриваются нейронные сети ассоциативной памяти, восстанавливающие по искаженному и/или зашумленному образу ближайший к нему эталонный. Исследована информационная емкость сетей и предложено несколько путей ее повышения, в том числе - ортогональные тензорные (многочастичные) сети. Построены способы предобработки, позволяющие конструировать нейронные сети ассоциативной памяти для обработки образов, инвариантной относительно групп преобразований. Описан численный эксперимент по использованию нейронных сетей для декодирования различных кодов.

1. Введение

Прежде чем заниматься конструированием сетей ассоциативной памяти необходимо ответить на следующие два вопроса: “Как устроена ассоциативная память?” и “Какие задачи она решает?”. Когда мы задаем эти вопросы, имеется в виду не устройство отделов мозга, отвечающих за ассоциативную память, а наше представление о макропроцессах, происходящих при проявлении ассоциативной памяти.

Принято говорить, что у человека возникла ассоциация, если при получении некоторой неполной информации он может подробно описать объект, к которому по его мнению относится эта информация. Достаточно хорошим примером может служить описание малознакомого человека. К примеру, при высказывании: “Слушай, а что за парень, с которым ты вчера разговаривал на вечеринке, такой высокий блондин?”- у собеседника возникает образ

[1] 660036, Красноярск-36, ВЦК СО РАН E-mail: amse@cckr.krasnoyarsk.su

вчерашнего собеседника, не ограничивающийся ростом и цветом волос. В ответ на заданный вопрос он может рассказать об этом человеке довольно много. При этом следует заметить, что содержащейся в вопросе информации явно недостаточно для точной идентификации собеседника. Более того, если вчерашний собеседник был случайным, то без дополнительной информации его и не вспомнят.

В качестве другого примера можно рассмотреть ситуацию, когда ваша однокурсница появляется в институте с совершенно новой прической и в незнакомой вам одежде. При этом вы, тем не менее, чаще всего ее узнаете и сможете определить чем ее новый образ отличается от привычного. Можно предположить, что это происходит следующим образом. При виде ее нового облика в вашей памяти возникает ассоциация с привычным для вас. А далее сравнивая эти два облика вы можете определить отличия.

Исходя из рассмотренных примеров можно сказать, что ассоциативная память позволяет по неполной и даже частично недостоверной информации восстановить достаточно полное описание *знакомого* объекта. Слово *знакомого* является очень важным, поскольку невозможно вызвать ассоциации с незнакомыми объектами. При этом объект должен быть знаком тому, у кого возникают ассоциации.

Одновременно рассмотренные примеры позволяют сформулировать решаемые ассоциативной памятью задачи:

1. Соотнести входную информацию со знакомыми объектами, и дополнить ее до точного описания объекта.
2. Отфильтровать из входной информации недостоверную, а на основании оставшейся решить первую задачу.

Очевидно, что под точным описанием объекта следует понимать всю информацию, которая доступна ассоциативной памяти. Вторая задача решается не поэтапно, а одновременно происходит соотнесение полученной информации с известными образцами и отсев недостоверной информации.

2. Постановка задачи

Пусть задан набор из m **эталонов** – n-мерных векторов $\left\{x^i\right\}$. Требуется построить сеть, которая при предъявлении на вход произвольного образа – вектора x – давала бы на выходе “наиболее похожий” эталон.

Всюду далее образы и, в том числе, эталоны - n-мерные векторы с координатами ± 1. Эталон, “наиболее похожий” на x - ближайший к x вектор x^i. Легко заметить, что это требование эквивалентно требованию максимальности скалярного произведения векторов x и x^i:

$$\left\|x - x^i\right\| = \|x\|^2 + \left\|x^i\right\|^2 - 2\left(x, x^i\right).$$

Первые два слагаемых в правой части совпадают для любых образов x и x^i, так как длины всех векторов-образов равны $\sqrt{n}$. Таким образом, задача поиска ближайшего образа сводится к поиску образа, скалярное произведение с которым максимально. Этот простой факт приводит к тому, что сравнивать придется линейные функции от образов, тогда как расстояние является квадратичной функцией.

3. Сети Хопфилда

Наиболее известной сетью ассоциативной памяти является сеть Хопфилда [1]. В основе сети Хопфилда лежит следующая идея – запишем систему дифференциальных уравнений для градиентной минимизации “энергии” H (функции Ляпунова). Точки равновесия такой системы находятся в точках минимума энергии. Функцию энергии будем строить из следующих соображений:

1. Каждый эталон должен быть точкой минимума.
2. В точке минимума все координаты образа должны иметь значения ± 1.

Функция

$$H = -\frac{1}{2}\sum_{i=1}^{m}\left(x, x^i\right)^2 + \alpha\sum_{j=1}^{n}\left(x_j^2 - 1\right)^2$$

не удовлетворяет этим требованиям строго, но можно предполагать, что первое слагаемое обеспечит притяжение к эталонам (для вектора x фиксированной длины максимум квадрата скалярного произведения $\left(x, x^i\right)^2$ достигается при $x=x_i$), а второе слагаемое $\sum_{j=1}^{n}\left(x_j^2-1\right)^2$ – приблизит к единице абсолютные величины всех координат точки минимума. Величина α характеризует соотношение между этими двумя требованиями и может меняться со временем.

Используя выражение для энергии, можно записать систему уравнений, описывающих функционирование сети Хопфилда:

$$\dot{x}_j = -\partial H / \partial x_j = \sum_{i=1}^{m}\left(x, x^i\right)x_j^i - 4\alpha\left(x_j^2-1\right)x_j . \qquad (1)$$

Сеть Хопфилда в виде (1) является сетью с непрерывным временем. Это, быть может, и удобно для некоторых вариантов аналоговой реализации, но для цифровых компьютеров лучше воспользоваться сетями, функционирующими в дискретном времени - шаг за шагом.

Построим сеть Хопфилда с дискретным временем. Сеть должна осуществлять преобразование входного вектора x так, чтобы выходной вектор x' был ближе к тому эталону, который является правильным ответом. Преобразование сети будем искать в следующем виде:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} w_i x^i\right), \qquad (2)$$

где w_i – вес i-го эталона, характеризующий его близость к вектору x, Sign - нелинейный оператор, переводящий вектор с координатами y_i в вектор с координатами $\mathrm{sign}\, y_i$.

Функционирование сети. Сеть работает следующим образом:

1. На вход сети подается образ x, а на выходе снимается образ x'.
2. Если $x' \neq x$, то полагаем $x = x'$ и возвращаемся к шагу 1.
3. Полученный вектор x' является ответом.

Таким образом, ответ всегда является неподвижной точкой преобразования сети (2) и именно это условие (неизменность при обработке образа сетью) и является условием остановки.

Пусть i^* – номер эталона, ближайшего к образу x. Тогда, если выбрать веса пропорционально близости эталонов к исходному образу x, то следует ожидать, что образ x' будет ближе к эталону x^{i^*}, чем x, а после нескольких итераций он станет совпадать с эталоном x^{i^*}.

Наиболее простой сетью вида (2) является дискретный вариант сети Хопфилда с весами равными скалярному произведению эталонов на предъявляемый образ:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \left(x, x^i \right) x^i \right). \qquad (3)$$

О сетях Хопфилда известно, что они способны запомнить и точно воспроизвести "порядка $0.14n$ слабо скоррелированных образов". В этом высказывании содержится два ограничения:

- число эталонов не превосходит $0.14n$.
- эталоны слабо скоррелированны.

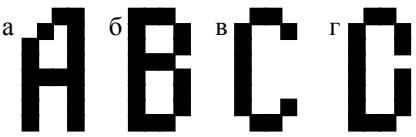

Рис. 1. а, б, с – эталоны, г – ответ сети на предъявление любого эталона

Наиболее существенным является второе ограничение, поскольку образы, которые сеть должна обрабатывать, часто очень похожи. Примером могут служить буквы латинского алфавита. При обучении сети Хопфилда распознаванию трех первых букв (см. рис. 1 а, б, в), при предъявлении на вход сети любого их эталонов в качестве ответа получается образ, приведенный на рис. 1 г.

В связи с такими примерами первый вопрос о качестве работы сети ассоциативной памяти звучит тривиально: будет ли сеть правильно обрабатывать сами эталонные образы (т.е. не искажает их)?

Зависимость работы сети Хопфилда от степени скоррелированности образов можно легко продемонстрировать на следующем примере. Пусть даны три эталона x^1, x^2, x^3 таких, что

$$\begin{aligned} &\left(x^1, x^2\right) + \left(x^1, x^3\right) > n, \\ &\left(x^1, x^2\right) + \left(x^2, x^3\right) > n, \\ &\left(x^1, x^3\right) + \left(x^2, x^3\right) > n, \\ &\left(x^i, x^j\right) > 0 \ (\forall i, j). \end{aligned} \tag{4}$$

Для любой координаты существует одна из четырех возможностей:

$$\begin{aligned} &1) \ \ x_l^i = x_l^j = 1 \ (\forall i, j), \\ &2) x_l^i = -x_l^j = x_l^k = 1, \\ &3) x_l^i = -x_l^j = x_l^k = -1, \\ &4) \ \ x_l^i = x_l^j = -1 \ \ (\forall i, j) \ . \end{aligned}$$

В первом случае при предъявлении сети q-го эталона в силу формулы (3) получаем $x_l' = Sign\left(\sum_{i=1}^{3}\left(x^q, x^i\right) \times 1\right) = 1$, так как все скалярные произведения положительны по условию (4). Аналогично получаем в четвертом случае $x_j' = -1$.

Во втором случае рассмотрим отдельно три варианта

$$\begin{aligned} x = x^i, \ \ x_j' &= Sign\left(-\left(x^i, x^i\right) + \left(x^i, x^j\right) + \left(x^i, x^k\right)\right) = 1 \\ x = x^j, \ \ x_l' &= Sign\left(-\left(x^j, x^i\right) + \left(x^j, x^j\right) + \left(x^j, x^k\right)\right) = 1 \\ x = x^k, \ \ x_l' &= Sign\left(-\left(x^k, x^i\right) + \left(x^k, x^j\right) + \left(x^k, x^k\right)\right) = 1 \end{aligned}$$

так как скалярный квадрат любого образа равен n, а сумма двух любых скалярных произведений эталонов больше n, по условию (4). Таким образом, независимо от предъявленного эталона получаем $x_j' = 1$. Аналогично в третьем случае получаем $x_j' = -1$.

Окончательный вывод таков: если эталоны удовлетворяют условиям (4), то при предъявлении любого эталона на выходе всегда будет один образ. Этот образ может быть эталоном или “химерой”, составленной, чаще всего, из узнаваемых фрагментов различных эталонов (примером “химеры” может служить образ, приведенный на рис. 1 г). Рассмотренный ранее пример с буквами детально иллюстрирует такую ситуацию.

Приведенные выше соображения позволяют сформулировать требование, детализирующие понятие “слабо скоррелированных образов”. Для правильного распознавания всех эталонов достаточно (но не необходимо) потребовать, чтобы выполнялось следующее неравенство $\sum_{\substack{i=1 \\ i\neq j}}^{m}\left|\left(x^i,x^j\right)\right| < n, \forall j$. Более простое и наглядное, хотя и более сильное условие можно записать в виде $\left|\left(x^i,x^j\right)\right| < \frac{n}{m}, \forall i \neq j$. Из этих условий видно, что чем больше задано эталонов, тем более жесткие требования предъявляются к степени их скоррелированности, тем ближе они должны быть к ортогональным.

Рассмотрим преобразование (3) как суперпозицию двух преобразований:

$$Px = \sum_{i=1}^{m}\left(x,x^i\right)x^i, \qquad x' = \mathrm{Sign}\left(Px\right). \tag{5}$$

Обозначим через $L\left(\left\{x^i\right\}\right) = \left\{x \mid x = \sum_{i=1}^{m}\alpha_i x^i;\ \alpha_i \in \mathrm{R}\right\}$ – линейное пространство, натянутое на множество эталонов. Тогда первое преобразование в (5) переводит векторы из R^n в $L\left(\left\{x^i\right\}\right)$. Второе преобразование в (5) переводит результат первого преобразования Px в одну из вершин гиперкуба образов. Легко показать, что второе преобразование в (5) переводит точку Px в ближайшую вершину гиперкуба. Действительно, пусть a и b две различные вершины гиперкуба такие, что a – ближайшая к Px, а $b = x'$. Из того, что a и b различны следует, что существует множество индексов, в которых

координаты векторов a и b различны. Обозначим это множество через $I = \{i : a_i = -b_i\}$. Из второго преобразования в (5) и того, что $b = x'$, следует, что знаки координат вектора Px всегда совпадают со знаками соответствующих координат вектора b. Учитывая различие знаков i-х координат векторов a и Px при $i \in I$ можно записать $\left|a_i - (Px)_i\right| = |a_i| + \left|(Px)_i\right| = 1 + \left|(Px)_i\right|$. Совпадение знаков i-х координат векторов b и Px при $i \in I$ позволяет записать следующее неравенство $\left|b_i - (Px)_i\right| = \left||a_i| - \left|(Px)_i\right|\right| < 1 + \left|(Px)_i\right|$. Сравним расстояния от вершин a и b до точки Px

$$\|b - Px\| = \sum_{i=1}^{n}\left(b_i - (Px)_i\right)^2 = \sum_{i \in I}\left(b_i - (Px)_i\right)^2 + \sum_{i \notin I}\left(b_i - (Px)_i\right)^2 <$$

$$\sum_{i \in I}\left(1 + \left|(Px)_i\right|\right)^2 + \sum_{i \notin I}\left(a_i - (Px)_i\right)^2 = \sum_{i=1}^{n}\left(a_i - (Px)_i\right)^2 = \|a - Px\|.$$

Полученное неравенство $\|b - Px\| < \|a - Px\|$ противоречит тому, что a – ближайшая к Px. Таким образом доказано, что второе преобразование в (5) переводит точку Px в ближайшую вершину гиперкуба образов.

4. Ортогональные сети

Для обеспечения правильного воспроизведения эталонов достаточно потребовать, чтобы первое преобразование в (5) было таким, что $x^i = Px^i$. Очевидно, что если проектор является ортогональным, то это требование выполняется, поскольку $x = Px$ при $x \in L\left(\left\{x^i\right\}\right)$, а $x^j \in L\left(\left\{x^i\right\}\right)$ по определению множества $L\left(\left\{x^i\right\}\right)$.

Для обеспечения ортогональности проектора воспользуемся дуальным множеством векторов. Множество векторов $V\left(\left\{x^i\right\}\right)$ называется дуальным к

множеству векторов $\left\{x^i\right\}$ если все вектора этого множества v^j удовлетворяют следующим требованиям:

1. $\left(x^i, v^j\right) = \delta_{ij};\ \delta_{ij} = 0,$ при $i \neq j;\ \delta_{ij} = 1$ при $i = j;$

2. $v^j \in L\left(\left\{x^i\right\}\right)$.

Преобразование $Px = \sum_{i=1}^{m}\left(x, v^i\right)x^i, \quad v^i \in V\left(\left\{x^i\right\}\right)$ является ортогональным проектором на линейное пространство $L\left(\left\{x^i\right\}\right)$.

Ортогональная сеть ассоциативной памяти преобразует образы по формуле

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x, v^i\right)x^i\right). \qquad (6)$$

Дуальное множество векторов существует тогда и только тогда, когда множество векторов $\left\{x^i\right\}$ линейно независимо. Если множество эталонов $\left\{x^i\right\}$ линейно зависимо, то исключим из него линейно зависимые образы и будем рассматривать полученное усеченное множество эталонов как основу для построения дуального множества и преобразования (6). Образы, исключенные из исходного множества эталонов, будут по-прежнему сохраняться сетью в исходном виде (преобразовываться в самих себя). Действительно, пусть эталон x является линейно зависимым от остальных m эталонов. Тогда его можно представить в виде $x = \sum_{i=1}^{m}\alpha_i x^i$. Подставив полученное выражение в преобразование (6) и учитывая свойства дуального множества получим:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x, v^i\right)x^i\right) = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(\sum_{j=1}^{m}\alpha_j x^j, v^i\right)x^i\right) =$$
$$= \mathrm{Sign}\left(\sum_{i,j=1}^{m}\alpha_j\left(x^j, v^i\right)x^i\right) = \mathrm{Sign}\left(\sum_{j=1}^{m}\alpha_j x^j\right) = \mathrm{Sign}\left(x\right) = x \qquad (7)$$

Рассмотрим свойства сети (6) [2]. Во-первых, количество запоминаемых и точно воспроизводимых эталонов не зависит от степени их скоррелированности. Во-вторых, формально сеть способна работать без искажений при любом возможном числе эталонов (всего их может быть до 2^n). Однако, если число линейно независимых эталонов (т.е. ранг множества эталонов) равно n, сеть становится прозрачной – какой бы образ не предъявили на ее вход, на выходе окажется тот же образ. Действительно, как было показано в (7), все образы, линейно зависимые от эталонов, преобразуются проективной частью преобразования (6) сами в себя. Значит, если в множестве эталонов есть n линейно независимых, то любой образ можно представить в виде линейной комбинации эталонов (точнее n линейно независимых эталонов), а проективная часть преобразования (6) в силу формулы (7) переводит любую линейную комбинацию эталонов в саму себя.

Если число линейно независимых эталонов меньше *n*, то сеть преобразует поступающий образ, отфильтровывая помехи, ортогональные всем эталонам.

Отметим, что результаты работы сетей (3) и (6) эквивалентны, если все эталоны попарно ортогональны.

Остановимся несколько подробнее на алгоритме вычисления дуального множества векторов. Обозначим через $\Gamma\left(\left\{x^i\right\}\right)$ матрицу Грама множества векторов $\left\{x^i\right\}$. Элементы матрицы Грама имеют вид $\gamma_{ij} = \left(x^i, x^j\right)$ (ij-ый элемент матрицы Грама равен скалярному произведению i-го эталона на j-ый). Известно, что векторы дуального множества можно записать в следующем виде:

$$v^i = \sum_{j=1}^{m} \gamma_{ij}^{-1} x^j \,, \tag{8}$$

где γ_{ij}^{-1} – элемент матрицы $\Gamma^{-1}\left(\left\{x^i\right\}\right)$. Поскольку определитель матрицы Грама равен нулю, если множество векторов линейно зависимо, то матрица, обратная к матрице Грама, а следовательно и дуальное множество векторов существует только тогда, когда множество эталонов линейно независимо.

Для работ сети (6) необходимо хранить эталоны и матрицу $\Gamma^{-1}\left(\left\{x^i\right\}\right)$.

Рассмотрим процедуру добавления нового эталона к сети (6). Эта операция часто называется дообучением сети. Важным критерием оценки алгоритма формирования сети является соотношение вычислительных затрат на обучение и дообучение. Затраты на дообучение не должны зависеть от числа освоенных ранее эталонов.

Для сетей Хопфилда это, очевидно, выполняется - добавление еще одного эталона сводится к прибавлению к функции *H* одного слагаемого $\left(x, x^{m+1}\right)^2$, а модификация связей в сети - состоит в прибавлении к весу *ij*-й связи числа $x_i^{m+1} x_j^{m+1}$ - всего n^2 операций.

Для рассматриваемых сетей с ортогональным проектированием также возможно простое дообучение. На первый взгляд, это может показаться странным - если добавляемый эталон линейно независим от старых эталонов, то вообще говоря необходимо пересчитать матрицу Грама и обратить ее. Однако симметричность матрицы Грама позволяет не производить заново процедуру обращения всей матрицы. Действительно, обозначим через G_m – матрицу Грама для множества из m векторов x^i; через E_m – единичную матрицу размерности $m \times m$. При обращении матриц методом Гаусса используется следующая процедура:

1. Запишем матрицу размерности $m \times 2m$ следующего вида: $\left(\mathrm{G}_m \middle| \mathrm{E}_m\right)$.

2. Используя операции сложения строк и умножения строки на ненулевое число преобразуем левую квадратную подматрицу к единичной. В результате получим $\left(\mathrm{E}_m \middle| \mathrm{G}_m^{-1}\right)$.

Пусть известна G_m^{-1} – обратная к матрице Грама для множества из m векторов x^i. Добавим к этому множеству вектор x^{m+1}. Тогда матрица для обращения матрицы G_{m+1} методом Гаусса будет иметь вид:

$$\left(\begin{array}{ccc|c} & & \left(x^1, x^{m+1}\right) & \\ & \mathrm{G}_m & \vdots & \\ & & \left(x^m, x^{m+1}\right) & \mathrm{E}_{m+1} \\ \left(x^1, x^{m+1}\right) & \cdots \left(x^m, x^{m+1}\right) & \left(x^{m+1}, x^{m+1}\right) & \end{array}\right).$$

После приведения к единичной матрице главного минора ранга m получится следующая матрица:

$$\left(\begin{array}{ccc|ccc} & & b_1 & & & 0 \\ & \mathrm{E}_m & \vdots & & \mathrm{G}_m^{-1} & \vdots \\ & & b_m & & & 0 \\ \left(x^1, x^{m+1}\right) & \cdots \left(x^m, x^{m+1}\right) & \left(x^{m+1}, x^{m+1}\right) & 0 & \cdots 0 & 1 \end{array}\right),$$

где b_i – неизвестные величины, полученные в ходе приведения главного минора к единичной матрице. Для завершения обращения матрицы G_{m+1} необходимо привести к нулевому виду первые m элементов последней строки и $(m+1)$-о столбца. Для обращения в ноль i-о элемента последней строки необходимо умножить i-ю строку на $\left(x^i, x^{m+1}\right)$ и вычесть из последней строки. После проведения этого преобразования получим

$$\left(\begin{array}{ccc|ccc} & & b_1 & & & 0 \\ & \mathrm{E}_m & \vdots & & \mathrm{G}_m^{-1} & \vdots \\ & & b_m & & & 0 \\ 0 & \cdots 0 & b_0 & c_1 & \cdots c_m & 1 \end{array}\right),$$

где $b_0 = \left(x^{m+1}, x^{m+1}\right) - \sum_{i=1}^{m} \left(x^i, x^{m+1}\right) b_i$, $c_i = -\sum_{j=1}^{m} \left(x^j, x^{m+1}\right) \mathrm{G}^{-1}_{m,ji}$. $b_0 = 0$ только если новый эталон является линейной комбинацией первых m эталонов. Следовательно $b_0 \neq 0$. Для завершения обращения необходимо разделить последнюю строку на b_0 и затем вычесть из всех предыдущих строк последнюю, умноженную на соответствующее номеру строки b_i. В результате получим следующую матрицу

$$\left(\begin{array}{ccc|ccc} & & 0 & & & -b_1/b_0 \\ & \mathrm{E}_m & \vdots & & \mathrm{F} & \vdots \\ & & 0 & & & -b_m/b_0 \\ 0 & \cdots & 0 \quad 1 & c_1/b_0 & \cdots & c_m/b_0 \quad 1/b_0 \end{array}\right),$$

где $\mathrm{F}_{ij} = \mathrm{G}^{-1}_{m,ij} - b_i c_j / b_0$. Поскольку матрица, обратная к симметричной, всегда симметрична получаем $c_i/b_0 = -b_i/b_0$ при всех i. Так как $b_0 \neq 0$ следовательно $b_i = -c_i$.

Обозначим через d вектор $\left(\left(x^1, x^{m+1}\right), \ldots, \left(x^m, x^{m+1}\right)\right)$, через b – вектор $\left(b_1, \ldots, b_m\right)$. Используя эти обозначения можно записать $\mathrm{b} = \mathrm{G}^{-1}_m \mathrm{d}$, $b_0 = \left(x^{m+1}, x^{m+1}\right) - (\mathrm{d}, \mathrm{b})$. Матрица G^{-1}_{m+1} записывается в виде

$$\mathrm{G}^{-1}_{m+1} = \frac{1}{b_0} \begin{pmatrix} b_0 \mathrm{G}^{-1}_m + \mathrm{b} \otimes \mathrm{b} & -\mathrm{b} \\ -\mathrm{b} & 1 \end{pmatrix}.$$

Таким образом, при добавлении нового эталона требуется произвести следующие операции:

1. Вычислить вектор d (m скалярных произведений – mn операций, $mn \leq n^2$).
2. Вычислить вектор b (умножение вектора на матрицу – m^2 операций).
3. Вычислить b_0 (два скалярных произведения – $m + n$ операций).

4. Умножить матрицу на число и добавить тензорное произведение вектора b на себя ($2m^2$ операций).
5. Записать G_{m+1}^{-1}.

Таким образом, эта процедура требует $m + n + mn + 3m^2$ операций. Тогда как стандартная схема полного пересчета потребует:

1. Вычислить всю матрицу Грама ($nm(m+1)/2$ операций).
2. Методом Гаусса привести левую квадратную матрицу к единичному виду ($2m^3$ операций).
3. Записать G_{m+1}^{-1}.

Всего $2m^3 + nm(m+1)/2$ операций, что в m раз больше.

Используя ортогональную сеть (6), удалось добиться независимости способности сети к запоминанию и точному воспроизведению эталонов от степени скоррелированности эталонов. Так, например, ортогональная сеть смогла правильно воспроизвести все буквы латинского алфавита в написании, приведенном на рис. 1.

У сети (6) можно выделить два основных недостатка:

1. Число линейно независимых эталонов должно быть меньше размерности системы n.

2. Неинвариантностью - если два визуальных образа отличаются только своим положением в рамке, то в большинстве задач желательно объединять их в один эталон.

Оба этих недостатка можно устранить, изменив выбор весовых коэффициентов в (2).

5. Тензорные сети

Для увеличения числа линейно независимых эталонов, не приводящих к прозрачности сети, используется прием перехода к тензорным или многочастичным сетям [3-7].

Тензорным произведением k n-мерных векторов $y^1,\cdots,y^k$ называется k-индексная величина $b_{i_1\cdots i_k}$, у которой все индексы независимо пробегают весь набор значений от единицы до n, а $b_{i_1\cdots i_k}=y^1_{i_1}y^2_{i_2}\cdots y^k_{i_k}$. k-ой тензорной степенью вектора x будем называть вектор $x^{\otimes k}$, полученный как тензорное произведение k векторов x. Вектор $x^{\otimes k}$ является n^k-мерным вектором. Однако пространство $L\left(\left\{x^{i\otimes k}\right\}\right)$ имеет размерность, не превышающую величину $r_{n,k}=\sum_{i=0}^{k}\mathrm{C}^i_{n-1}$, где $\mathrm{C}^q_p=\frac{p!}{q!(p-q)!}$ – число сочетаний из p по q.

Теорема. При $k<n$ в ранг $r_{n,k}$ множества $\left\{x^{\otimes k}\right\}$ равен:

$$r_{n,k}=\sum_{i=0}^{k}\mathrm{C}^i_{n-1}.$$

Небольшая модернизация треугольника Паскаля, позволяет легко вычислять эту величину. На рис. 1 приведен “тензорный” треугольник Паскаля. При его построении использованы следующие правила:

2									
3	4								
4	7	8							
5	11	15	16						
6	16	26	31	32					
7	22	42	57	63	64				
8	29	64	99	120	127	128			
9	37	93	163	219	247	255	256		
10	46	130	256	382	466	502	511	512	
$r_{1,1}$	$r_{1,2}$	$r_{1,3}$	$r_{1,4}$	$r_{1,5}$	$r_{1,6}$	$r_{1,7}$	$r_{1,8}$	$r_{1,9}$	$r_{1,10}$

Рис. 1. “Тензорный” треугольник Паскаля

1. Первая строка содержит двойку, поскольку при n=2 в множестве X всего два неколлинеарных вектора.

2. При переходе к новой строке, первый элемент получается добавлением единицы к первому элементу предыдущей строки, второй – как сумма первого и второго элементов предыдущей строки, третий – как сумма второго и третьего элементов и т.д. Последний элемент получается удвоением последнего элемента предыдущей строки.

Таблица 1.

n	k	n^k	C_{n+k-1}^{k-1}	$r_{n,k}$
5	2	25	15	11
	3	125	35	15
10	3	1 000	220	130
	6	1 000 000	5005	466
	8	100 000 000	24310	511

В табл. 1 приведено сравнение трех оценок информационной емкости тензорных сетей для некоторых значений n и k. Первая оценка - n^k - заведомо завышена, вторая - C_{n+k-1}^{k-1} - дается формулой Эйлера для размерности пространства симметричных тензоров и третья - точное значение $r_{n,k}$

Как легко видеть из таблицы, уточнение при переходе к оценке r_{nk} является весьма существенным. С другой стороны, предельная информационная емкость тензорной сети (число правильно воспроизводимых образов) может существенно превышать число нейронов, например, для 10 нейронов тензорная сеть валентности 8 имеет предельную информационную емкость 511.

Легко показать, что если множество векторов $\left\{x^i\right\}$ не содержит взаимно обратных, то размерность пространства $L\left(\left\{x^{i\otimes n}\right\}\right)$ равна числу векторов в множестве $\left\{x^i\right\}$. Сеть (2) для случая тензорных сетей имеет вид

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x^{\otimes k}, x^{i\otimes k}\right)x^i\right) = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x, x^i\right)^k x^i\right), \qquad (9)$$

а ортогональная тензорная сеть

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \left(x^{\otimes k}, v^i \right) x^i \right) = \mathrm{Sign}\left(\sum_{i=1}^{m} \sum_{j=1}^{m} \gamma_{ij}^{-1} \left(x, x^j \right)^k x^i \right), \qquad (10)$$

где γ_{ij}^{-1} – элемент матрицы $\Gamma^{-1}\left(\left\{ x^{i\otimes k} \right\} \right)$. Сеть (9) хорошо работает на слабо скоррелированных эталонах, а сеть (10) не чувствительна к степени скоррелированности эталонов.

6. Сети для инвариантной обработки изображений

Для того, чтобы при обработке переводить визуальные образов, отличающиеся только положением в рамке изображения, в один эталон, применяется следующий прием [7]. Преобразуем исходное изображение в некоторый вектор величин, не изменяющихся при сдвиге (вектор инвариантов). Простейший набор инвариантов дают автокорреляторы - скалярные произведения образа на сдвинутый образ, рассматриваемые как функции вектора сдвига.

В качестве примера рассмотрим вычисление сдвигового автокоррелятора для черно-белых изображений. Пусть дан двумерный образ S размером $p \times q = n$. Обозначим точки образа как s_{ij}. Элементами автокоррелятора $Ac(S)$ будут величины $a_{kl} = \sum_{i=1}^{p} \sum_{j=1}^{q} s_{ij} s_{i+k, j+l}$, где $s_{ij=0}$ при выполнении любого из неравенств $i < 1, i > p, j < 1, j > q$. Легко проверить, что автокорреляторы любых двух образов, отличающихся только расположением в рамке, совпадают. Отметим, что $a_{ij} = a_{-i,-j}$ при всех i, j, и $a_{ij} = 0$ при выполнении любого из неравенств $i < 1 - p, i > p - 1, j < 1 - q, j > q - 1$. Таким образом, можно считать, что размер автокоррелятора равен $p \times (2q + 1)$.

Автокорреляторная сеть имеет вид

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \left(Ac(x), Ac\left(x^i \right) \right) x^i \right). \qquad (11)$$

Сеть (11) позволяет обрабатывать различные визуальные образы, отличающиеся только положением в рамке, как один образ.

Подводя итоги, можно сказать, что все сети ассоциативной памяти типа (2) можно получить, комбинируя следующие преобразования:

1. Произвольное преобразование. Например, переход к автокорреляторам, позволяющий объединять в один выходной образ все образы, отличающиеся только положением в рамке.
2. Тензорное преобразование, позволяющее сильно увеличить способность сети запоминать и точно воспроизводить эталоны.
3. Переход к ортогональному проектору, снимающий зависимость надежности работы сети от степени скоррелированности образов.

Наиболее сложная сеть будет иметь вид:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \sum_{j=1}^{m} \gamma_{ij}^{-1} \left(F(x), F\left(x^j\right) \right)^k x^i \right), \qquad (12)$$

где γ_{ij}^{-1} – элементы матрицы, обратной матрице Грама системы векторов $\Gamma^{-1}\left(\left\{ F\left(x^{i\otimes k} \right) \right\} \right)$, $F(x)$ – произвольное преобразование.

7. Численный эксперимент

Работа ортогональных тензорных сетей при наличии помех сравнивалась с возможностями линейных кодов, исправляющих ошибки. Линейным кодом, исправляющим k ошибок, называется линейное подпространство в n-мерном пространстве над GF_2, все вектора которого удалены друг от друга не менее чем на $2k+1$ (см., например, [8]). Линейный код называется совершенным, если для любого вектора n-мерного пространства существует кодовый вектор, удаленный от данного не более, чем на k. Тензорной сети в качестве эталонов подавались все кодовые векторы избранного для сравнения кода. Численные эксперименты с совершенными кодами показали, что тензорная сеть минимально необходимой валентности правильно декодирует все векторы. Для несовершенных кодов

картина оказалась хуже – среди устойчивых образов тензорной сети появились “химеры” – векторы, не принадлежащие множеству эталонов.

В случае n=10, k=1 (см. табл. 2 и 3, строка 1) при валентностях 3 и 5 тензорная сеть работала как единичный оператор – все входные вектора передавались на выход сети без изменений. Однако уже при валентности 7 число химер резко сократилось и сеть правильно декодировала более 60% сигналов. При этом были правильно декодированы все векторы, удаленные от ближайшего эталона на расстояние 2, а часть векторов, удаленных от ближайшего эталона на расстояние 1, остались химерами. В случае n=10, k=2 (см. табл. 2 и 3, строки 3, 4, 5) наблюдалось уменьшение числа химер с ростом валентности, однако часть химер, удаленных от ближайшего эталона на расстояние 2 сохранялась. Сеть правильно декодировала более 50% сигналов. Таким образом при малых размерностях и кодах, далеких от совершенных, тензорная сеть работает довольно плохо. Однако, уже при n=15, k=3 и валентности, большей 3 (см. табл. 2 и 3, строки 6, 7), сеть правильно декодировала все сигналы с тремя ошибками. В большинстве экспериментов число эталонов было больше числа нейронов.

Подводя итог, можно сказать, что качество работы сети возрастает с ростом размерности пространства и валентности и по эффективности устранения ошибок сеть приближается к коду, гарантированно исправляющему ошибки.

Работа выполнена при поддержке Красноярского краевого фонда науки, грант 6F0124.

Литература

1. Hopfield J.J. Neural networks and physical systems with emergent collective computational abilities // Proc. Nat. Acad. Sci. USA. 1982. Vol. 79. P.2554-2558.
2. Горбань А.Н. Обучение нейронных сетей. М.: изд-во СССР-США СП “ParaGraph”, 1990. 160 с.
3. Коноплев В.А., Синицын Е.В. “Моделирование нейронных сетей с максимально высокой информационной емкостью” // Тез. докл. III Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: изд. КГТУ, 1995 г., с. 66.

4. Golub D.N. Gorban A.N. Multi-Particle Networks for Associative Memory // Proc. of the World Congress on Neural Networks, Sept. 15-18, 1996, San Diego, CA, Lawrence Erlbaum Associates, 1996. P. 772-775.
5. Горбань А.Н., Миркес Е.М. Информационная емкость тензорных сетей // Тез. докл. IV Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: изд. КГТУ, 1996 г., с. 22-23.
6. Горбань А.Н., Миркес Е.М. Помехоустойчивость тензорных сетей // Тез. докл. IV Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: изд. КГТУ, 1996 г., с. 24-25.
7. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука, 1996.
8. Блейхут Р. Теория и практика кодов, контролирующих ошибки. М.: Мир. 1986. 576 с.