НЕЙРОКОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ

1 лекция

Литература

Основная литература

1.Хайкин С. Нейронные сети: полный курс.2-е изд.: Пер. с англ. — М.: ООО «Изд. дом Вильямс», 2016 — 1104 с.

2. Барский А. Б. Введение в нейронные сети [Электронный ресурс] / Барский А. Б.— Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 358 c. — Режим доступа: http://www.iprbookshop.ru/52144.— ЭБС «IPRbooks», по паролю

Дополнительная литература

1. Галушкин А. И. Нейросетевые технологии в России (1982-2010) [Текст] / А. И. Галушкин, С. Н. Симоров. — М.: Горячая линия-Телеком, 2012. — 316 с.

2. Харитонова А. А. Нейрокомпьютерные технологии [Электронный ресурс]: метод. указания к лаб. работе № 1 "Исследование работы формального нейрона" / А. А. Харитонова, Ю. В. Алышев; ПГУТИ, Каф. ТОРС. — Самара : ИНУЛ ПГУТИ, 2015. — 19 с.

3. Нейрокомпьютеры: от программной к аппаратной реализации [Текст]. — М.: Горячая линия-Телеком, 2008. — 152 с.

4. Злобин В. К. Нейросети и нейрокомпьютеры [Текст] : учебное пособие для вузов / В. К. Злобин, В. Н. Ручкин. — СПб.: БХВ-Петербург, 2011. — 252 с.

5. Осовский С. Нейронные сети для обработки информации. — М.: Финансы и статистика, 2002. — 344 с.

Понятие о нейронных сетях

Мозг человека обладает способностью организовывать свои структурные компоненты, называемые *нейронами*. Нейроны выполняют конкретные задачи: распознавание образов, обработку сигналов органов чувств, моторные функции. Благодаря параллельной работе и большому количеству нейроны выполняют подобные задачи во много раз быстрее, чем современные быстродействующие компьютеры.

При рождении мозг имеет структуру, позволяющую строить собственные правила на основании опыта. Опыт накапливается с течением времени, особенно в первые два года жизни человека. Но развитие продолжается и до последних дней жизни человека.

Способность настройки нервной системы человека в соответствии с окружающими условиями, называют *пластичностью* мозга. Аналогично, в искусственных нейронных сетях работа производится искусственными нейронами. *Нейронная сеть* представляется собой вычислительную систему, моделирующую способ обработки данных при решении конкретной задачи.

Сеть реализуется с помощью электронных компонентов или моделируется программой на цифровом компьютере.

Предметом исследования в данной дисциплине являются нейронные сети, осуществляющие вычисления с помощью процесса *обучения*.

Нейронная сеть — это распределённый параллельный вычислитель, состоящий из элементарных единиц обработки информации, накапливающий экспериментальные данные и, представляя их для последующей обработки, тем самым, формируя и накапливая *знания*.

Схожесть нейронной сети с мозгом человека:

— данные поступают из окружающей среды и используются в процессе обучения и накопления знаний;

— для накопления знаний применяются связи между нейронами, называемые *синаптическими весами*.

Процедура, используемая в процессе обучения, называется *алгоритмом обучения*. Эта процедура выстраивает в определённом порядке синаптические веса нейронной сети для необходимой структуры взаимосвязей нейронов.

Изменение синаптических весов аналогичен изменению весовых коэффициентов в линейных адаптивных фильтрах. Отличие в том, что нейронные сети способны изменять свою топологию (как и в мозге человека, нейроны отмирают, а новые синаптические связи постоянно создаются).

В литературе нейронные сети называют нейрокомпьютерами, сетями связи, параллельными распределёнными процессами.

Нейронные сети способны самообучаться, то есть создавать обобщения. *Обобщение* — способность получать обоснованный результат новой скорректированной конфигурации топологии на основании данных, которые не встречались в процессе обучения.

Однако автономно нейронные сети не могут обеспечить готовые решения. Обычно они являются составной часть в сложных системах.

Использование нейронных сетей обеспечивает следующие полезные свойства систем:

1. Нелинейность

Нелинейность распределена по сети.

2. Отображение входной информации в выходную

Наиболее распространённая парадигма обучения: *обучение с учителем*. На основе набора маркированных *учебных примеров* изменяются синаптические веса. Каждый пример состоит из входного сигнала и соответствующего ему *желаемого отклика*. Из множества примеров случайным образом выбирается один из них, а нейронная сеть модифицирует синаптические веса для минимизации расхождений желаемого выходного сигнала и формируемого сетью согласно выбранному статистическому критерию, то есть при этом модифицируются *свободные параметры* сети. Ранее использованные примеры могут впоследствии применены снова в другом порядке. Обучение производится до тех пор, пока изменения синаптических весов станут незначительными. То есть нейронная сеть обучается на примерах, составляя таблицу соответствий вход-выход для конкретной задачи (*классификация образов*).

3. Адаптивность

В *нестационарной* среде нейронные сети могут изменять синаптические веса в реальном времени.
Однако при наличии помех адаптивность может стать отрицательным фактором (*дилемма стабильности-пластичности*)

4. Очевидность ответа

Кроме решения задач классификации, могут решаться задачи повышения достоверности принимаемого решения или для исключения сомнительных решений.

5. Контекстная информация

Знания представляются в самой структуре нейронной сети с помощью её состояния активации. Каждый нейрон сети потенциально может быть подвержен влиянию всех остальных её нейронов. Как следствие, существование нейронной сети связано с контекстной информацией.

6. Отказоустойчивость

В случае незначительных повреждений структуры нейронной сети катастрофические последствия не наступают из-за распределённого характера хранения информации.

7. Эффективная реализуемость на СБИС.

Параллельная структура обеспечивает масштабируемость нейронных сетей. Возможно представление сложного поведения с помощью иерархической структуры.

8. Единообразие анализа и проектирования

Одно и то же проектное решение нейронной сети может использоваться во многих предметных областях.

- Нейроны являются стандартными составными частями любой нейронной сети
- Можно применять одни и те же теории и алгоритмы обучения в различных нейросетевых приложениях
- Модульные сети могут быть построены на основе интеграции целых модулей.

9. Аналогия с нейробиологией

Нейробиологи рассматривают искусственные нейроны сети как средство моделирования физических явлений, а инженеры пытаются подчерпнуть у нейробиологов новые идеи, выходящие за рамки традиционных электросхем.

(*Вестибуло-окулярный рефлекс* (обеспечение стабильности визуального образа при поворотах головы за счёт движения глаз). Ранее механизм моделировался с помощью сосредоточенной линейной системы и теории управления. Модели были пригодны для описания свойств системы, но не давали чёткого представления о свойствах самих нейронов. На основе рекуррентных сетей (рекуррентное обучение в реальном времени) воспроизведены и описаны многие статические, динамические, нелинейные и распределённые аспекты обработки сигналов при реализации вестибуло-окулярного рефлекса.

Сетчатка — часть мозга, выполняющая функции взаимосвязи физического изображения, проецируемого на матрицу рецепторов, с первым *нейронным изображением*. Этот процесс рассматривается как проходящий три стадии.

1. Снятие фотокопии слоем нейронов-рецепторов
2. Передача сформированного сигнала химическими синапсами на слой *биполярных клеток*.
3. Передача этих сигналов (также с помощью химических синапсов) на выходные нейроны.

На двух последних стадиях в операции участвуют нейроны с латеральным торможением (*горизонтальные клетки*), задачей которых является преобразование сигнала между разными синаптическими слоями. Существуют и *центробежные элементы* обеспечивают передачу сигнала с внутреннего синаптического слоя на внешний. Были созданы специальные чипы (*нейроморфные контура*), которые могли адаптироваться к изменению освещённости, идентифицировать контуры и движение.

Модели нейронов

Нейрон представляет собой единицу обработки информации в нейронной сети. В этой модели можно выделить три основных элемента:

1. Набор *синапсов* (связей), каждый из которых характеризуется своим *весом* (силой).
2. Сумматор
3. Функция активации

В математическом представлении функционирование нейрона k описывается уравнениями:

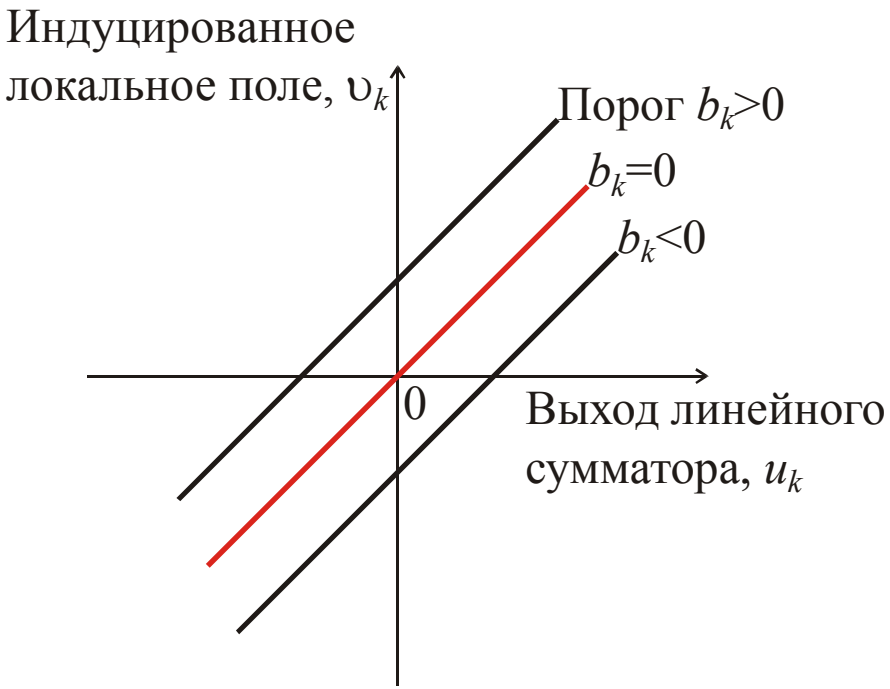
$$u_k = \sum_{j=1}^{m} w_{kj} x_j, \quad y_k = f\left(u_k + b_k\right),$$

здесь $x_1, x_2, ..., x_m$ — входные сигналы, $w_{k1}, w_{k2}, ..., w_{km}$ — синаптические веса нейрона k, u_k — линейная комбинация входных воздействий, b_k — порог, $f(\cdot)$ — функция активации, y_k — выходной сигнал нейрона.

Использование порога b_k обеспечивает эффект афинного преобразования (отображения плоскости или пространства в себя) выхода линейного сумматора u_k.

Постсинаптический потенциал: $\upsilon_k = u_k + b_k$

В зависимости от того, какое значение принимает порог b_k, положительное или отрицательное, *индуцированное локальное поле* (*потенциал активации*) υ_k нейрона k изменяется как показано на рисунке

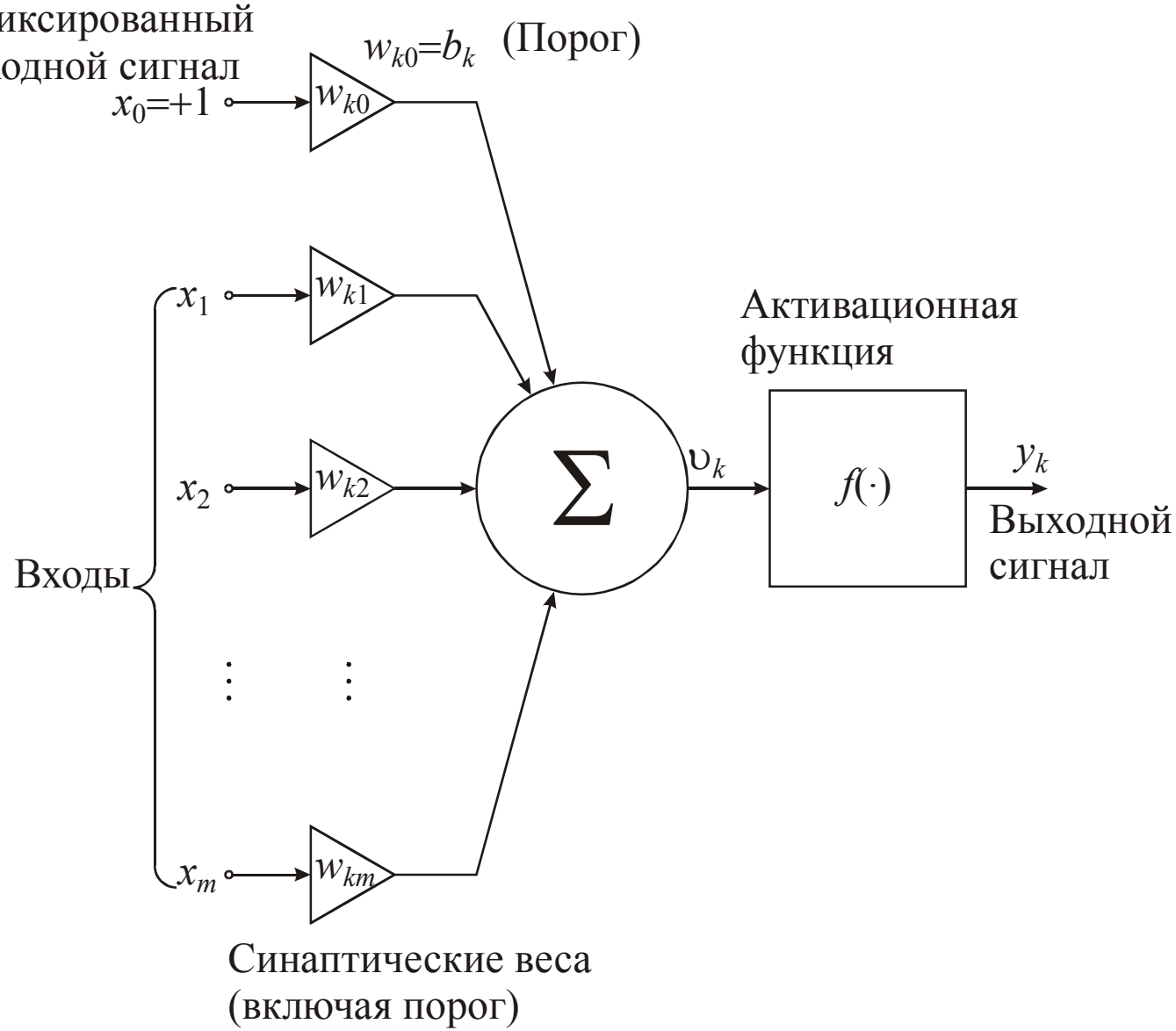


Порог b_k является внешним параметром искусственного нейрона k, но его можно представить как новый синапс:

$$\upsilon_k = \sum_{j=0}^{m} w_{kj} x_j$$

$$y_k = f(\upsilon_k)$$

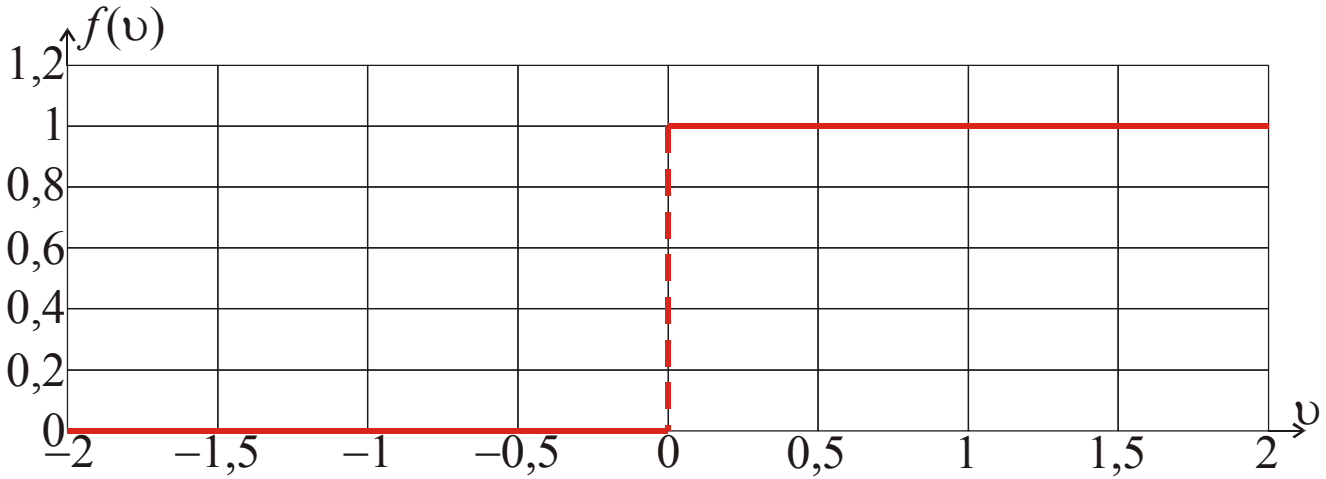
у которого входной сигнал $x_0 = +1$, а вес: $w_{k0} = b_k$. В итоге нелинейная модель нейрона:



Типы функций активации

1. Функция единичного скачка (модель Мак-Каллока–Питца)

$$f(\upsilon)=\begin{cases}1, & \upsilon \geqslant 0;\\ 0, & \upsilon < 0.\end{cases}$$



Выходной сигнал нейрона

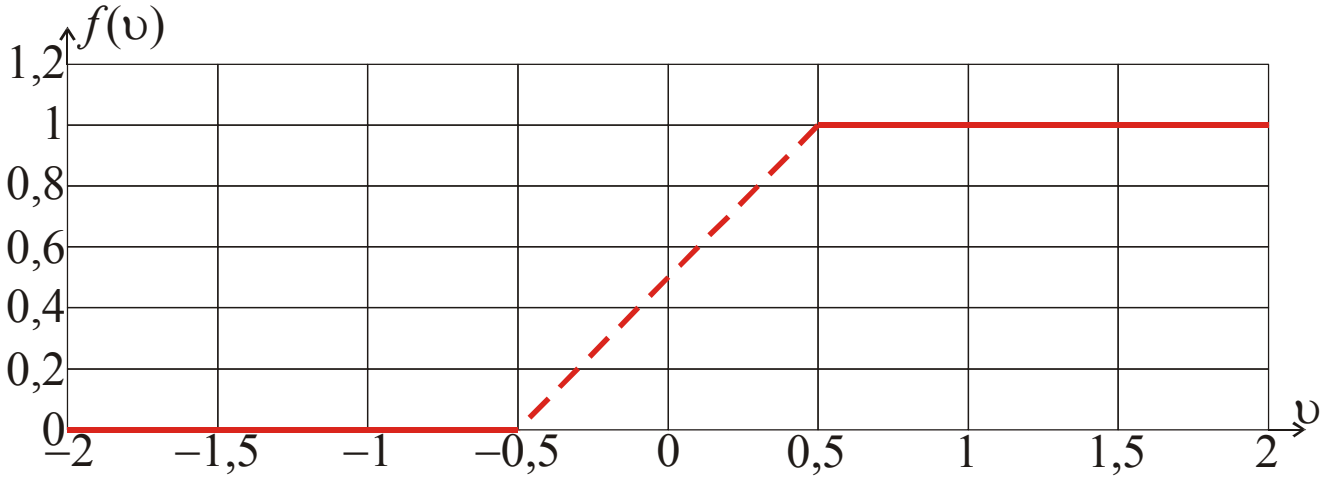
$$y_k=\begin{cases}1, & \upsilon_k \geqslant 0;\\ 0, & \upsilon_k < 0;\end{cases}$$

здесь υ_k — индуцированное локальное поле нейрона

$$\upsilon_k=\sum_{j=1}^{m} w_{kj}x_j + b_k\,.$$

2. Кусочно-линейная функция

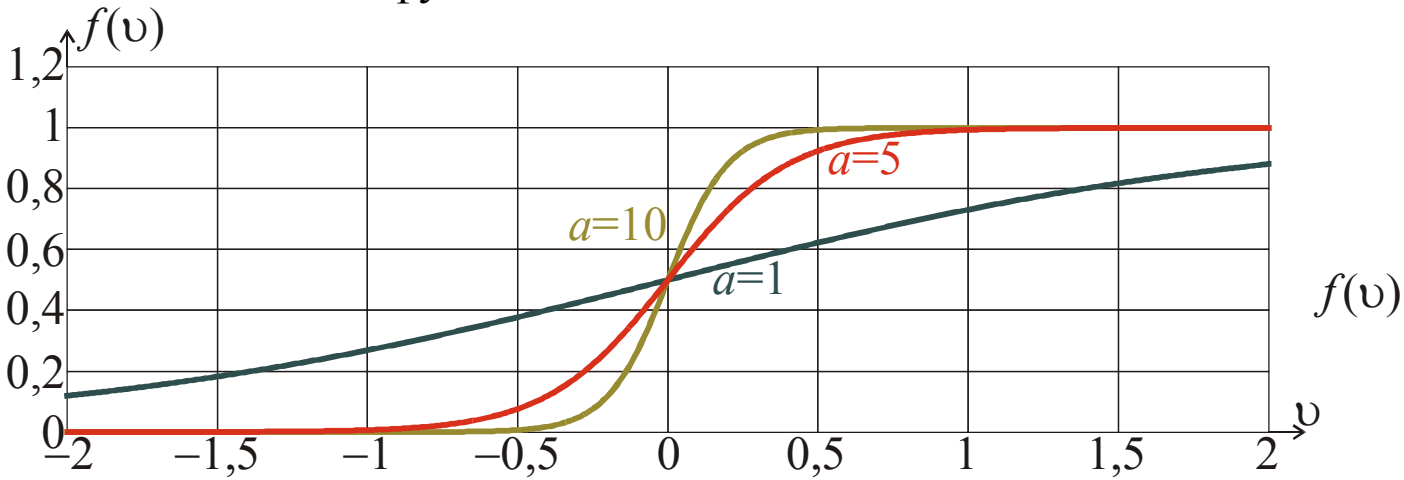
$$f(\upsilon)=\begin{cases}1, & \upsilon \geqslant +0{,}5;\\ |\upsilon|, & -0{,}5<\upsilon<+0{,}5;\\ 0, & \upsilon \leqslant -0{,}5,\end{cases}$$



3. Сигмоидальная функция (является дифференцируемой)

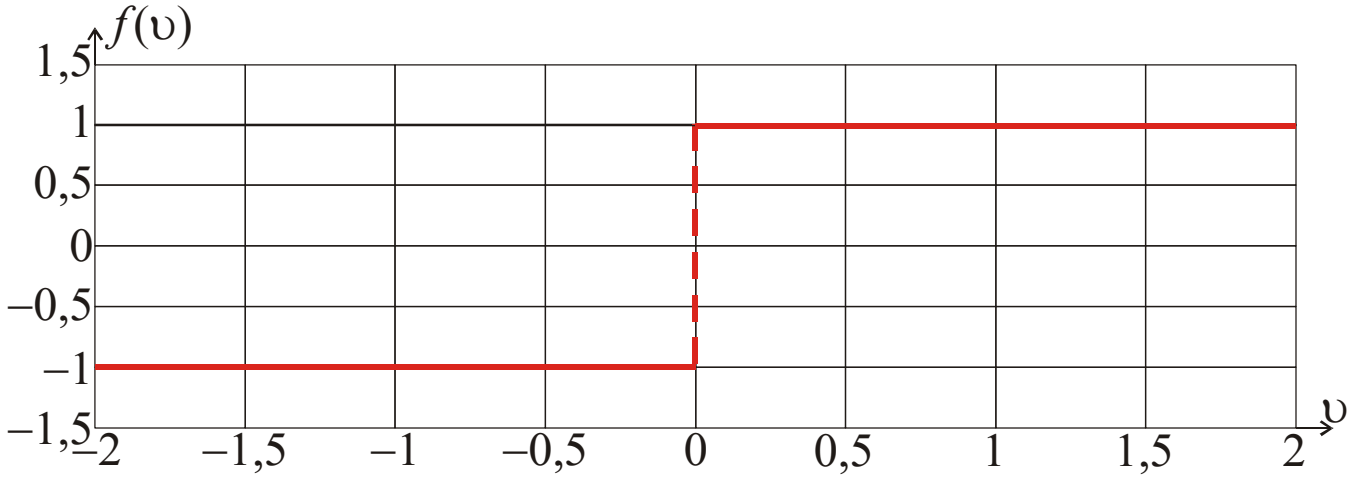
$$f(\upsilon)=\frac{1}{1+\exp(-a\upsilon)},$$

a — параметр наклона сигмоидальной функции.



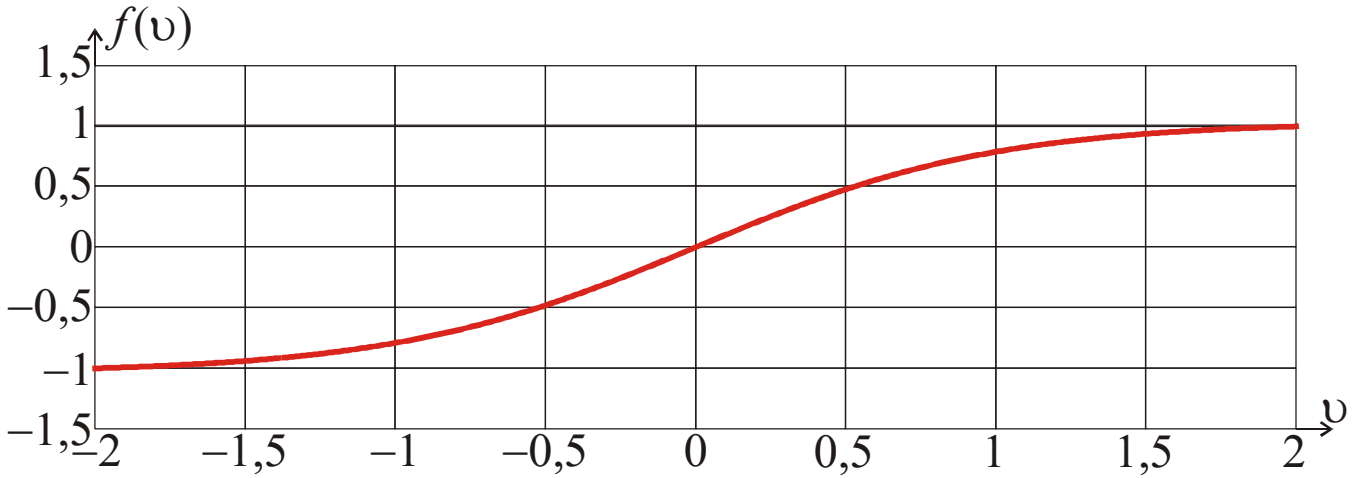
Биполярная пороговая функция

$$f(\upsilon)=\begin{cases}1, & \upsilon>0;\\ 0, & \upsilon=0;\\ -1, & \upsilon<0,\end{cases}$$



Биполярная сигмоидальная функция

$f(\upsilon)=\mathrm{th}(\upsilon)$.



Стохастическая модель нейрона

В подобных моделях нейрон может находиться в одном и двух состояний (+1 или −1). Решение о переключении принимается с учётом вероятности этого события.

$$x = \begin{cases} +1, & \text{с вероятностью } \mathrm{P}(\upsilon); \\ -1, & \text{с вероятностью } 1 - \mathrm{P}(\upsilon). \end{cases}$$

$$P(\upsilon) = \frac{1}{1 + \exp(-\upsilon / T)},$$

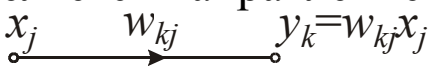
где T — аналог температуры, используемый для управления уровнем шума (степенью неопределённости переключения). При $T \to 0$ стохастический нейрон принимает детерминированную форму.

Представление сетей с помощью ориентированных графов

Граф передачи сигнала — это сеть направленных связей (*ветвей*), соединяющих отдельные точки (*узлы*). С каждым узлом j связан *сигнал* x_j. Обычная направленная сеть начинается в некотором узле j и заканчивается в другом узле k. С ней связана некоторая *передаточная функция*, определяющая зависимость сигнала y_k узла k от сигнала x_j узла j. Прохождение сигнала по различным частям графа подчиняется трём основным правилам.

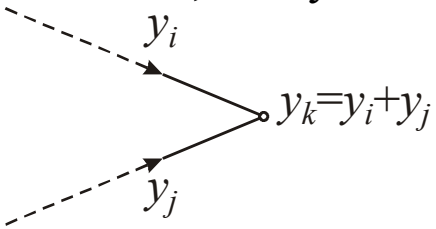
1. Направление прохождения сигнала вдоль каждой связи определяется направлением стрелки. Два вида

Синаптические связи — линейное соотношение вход-выход

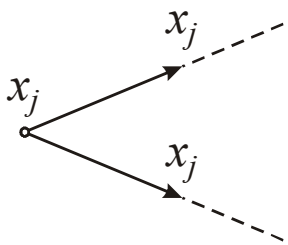


Активационные связи — нелинейное соотношение вход-выход

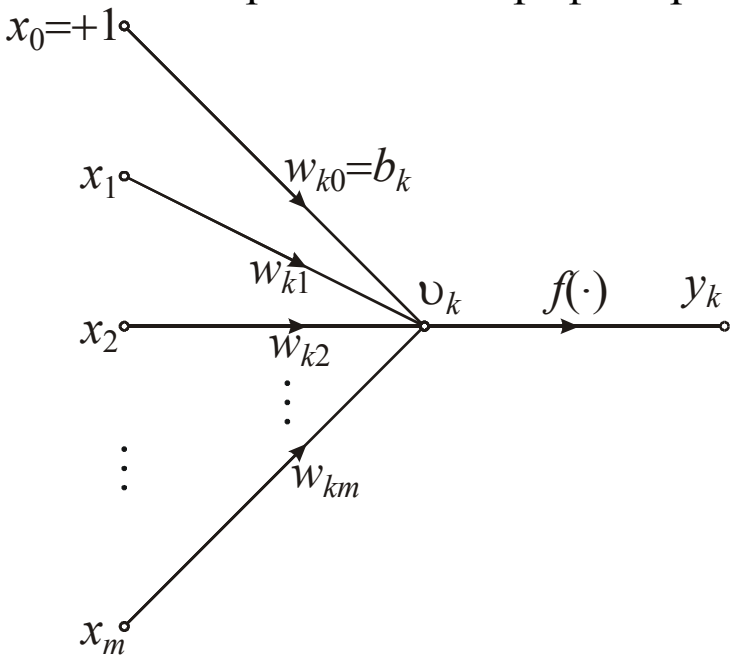
2. Сигнал узла равен алгебраической сумме сигналов, поступающих на его вход.



3. Сигнал данного узла передаётся по каждой исходящей связи без учёта передаточных функций исходящих связей.



Модель нейрона в виде графа передачи сигнала



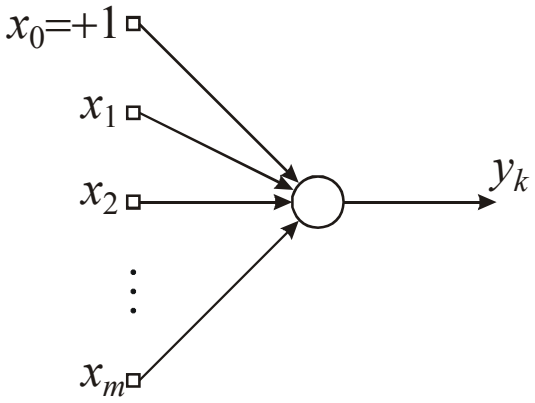
Нейронная сеть — это ориентированный граф (*полный граф*), состоящий из узлов, соединённых синаптическими и активационными связями, который характеризуется свойствами.

1. Каждый нейрон представляется множеством линейных синаптических связей, внешним порогом и, возможно, нелинейной связью активации. Порог, представляемый входной синаптической связью, считается равным +1.
2. Синаптические связи нейрона используются для взвешивания соответствующих входных сигналов.
3. Взвешенная сумма входных сигналов определяет индуцированное локальное поле каждого конкретного нейрона
4. Активационные связи модифицируют индуцированное локальное поле нейрона, создавая выходной сигнал.

При описании прохождения сигнала между нейронами используют *структурный граф* (*частично полный граф)*, который характеризуется следующими свойствами.

1. Входные сигналы графа формируются *узлами источника* (входными элементами)
2. Каждый нейрон представляется одним узлом, который называется *вычислительным*.
3. *Линии передачи сигнала*, соединяющие узлы источники и вычислительные узлы графа. Не имеют веса, а определяют направление прохождения сигнала на графе.

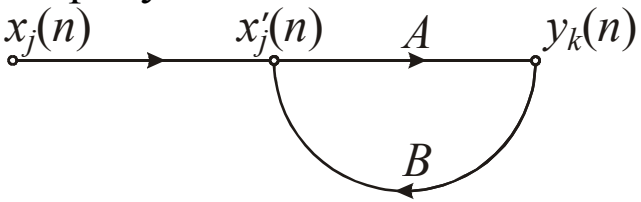
Простейший вариант графа, состоящего из одного нейрона с m входами и фиксированным порогом, равным +1



Обратная связь

Понятие *обратной связи* характерно для динамических систем, в которых выходной сигнал некоторого элемента системы оказывает влияние на входной сигнал этого элемента. Используется в рекуррентных сетях.

На рисунке



показан граф прохождения сигнала в *системе с одной обратной связью*. Входной сигнал $x_j(n)$, внутренний сигнал $x'_j(n)$ и выходной сигнал $y_j(n)$ являются функциями дискретной переменной n. Предполагается, что эта система линейна и содержит прямую и обратную связи, которые характеризуются операторами A и B соответственно. Выходной сигнал прямого канала частично определяет значение канала обратной связи.

$$y_k(n) = A\left[x_j{}'(n)\right]$$

$$x_j{}'(n) = x_j(n)B\left[y_k(n)\right]$$

квадратные скобки обозначают операторы A и B.

Исключая переменную $x'_j(n)$, получаем

$$y_k(n) = \frac{A}{1-AB}\left[x_j(n)\right]$$

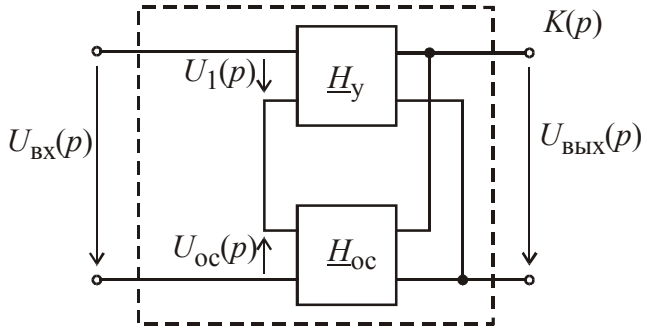
выражение $\dfrac{A}{1-AB}$ — *оператор замкнутого контура*.

выражение AB — *оператор разомкнутого контура*.

В общем случае оператор разомкнутого контура не обладает свойством коммутативности, то есть $AB \neq BA$.

Коэффициент передачи цепи с ОС (пояснение для приведённого оператора замкнутого контура)

Рассмотрим схему цепи с ОС с последовательным по напряжению ($U_{ОС}(p)$ зависит от $U_{ВЫХ}(p)$) способом включения и определим её коэффициент передачи $K(p)$.



Операторное напряжение на входе определяется уравнением вида: $U_{вх}(p)=U_1(p)-U_{ОС}(p)$,

где $U_1(p)=\frac{U_{вых}(p)}{H_у(p)}$, $U_{ОС}(p)=U_{ВЫХ}(p)\cdot H_{ОС}(p)$.

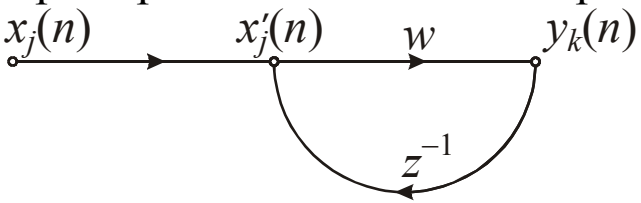
После подстановки получаем: $U_{вх}(p)=\frac{U_{вых}}{H_у(p)}-U_{вых}(p)\cdot H_{ОС}(p)$,

преобразовывая, получаем: $U_{вх}(p)=U_{вых}(p)\cdot\left(\frac{1}{H_у(p)}-H_{ОС}(p)\right)=U_{вых}(p)\cdot\frac{1-H_{ОС}(p)\cdot H_у(p)}{H_у(p)}$.

Операторный коэффициент передачи цепи с ОС определяется как:

$$H(p)=\frac{U_{вых}(p)}{U_{вх}(p)}=\frac{H_у(p)}{1-H_{ОС}(p)\cdot H_у(p)}$$

Пример системы с одной обратной связью.



A — фиксированный вес w

B — оператор единичной задержки z^{-1}

Оператор замкнутого контура: $\frac{A}{1-AB} = \frac{w}{1-wz^{-1}} = w(1-wz^{-1})^{-1}$

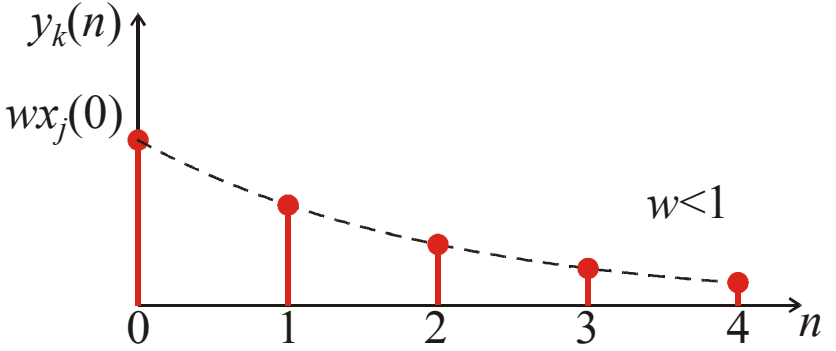
Используя биномиальное представление выражения $(1-wz^{-1})^{-1}$, оператор замкнутого контура можно записать в виде: $\frac{A}{1-AB} = w\sum_{l=0}^{\infty} w^l z^{-l}$. Подставляя в $y_k(n) = \frac{A}{1-AB}\left[x_j(n)\right]$, получим:

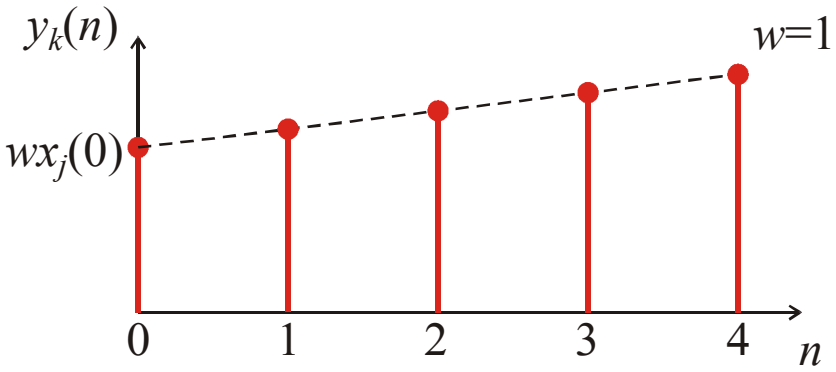
$$y_k(n) = w\sum_{l=0}^{\infty} w^l z^{-l}\left[x_j(n)\right]$$

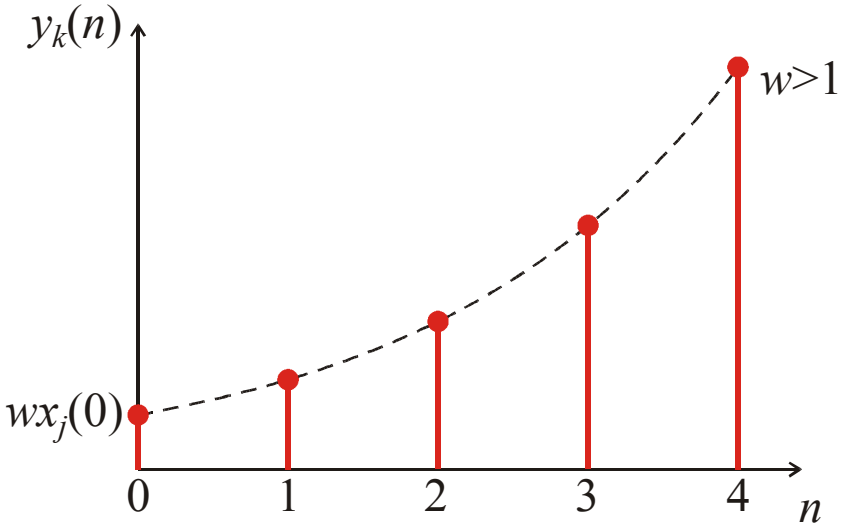
Квадратные скобки также обозначают, что z^{-1} является оператором.

Из определения этого оператора имеем $z^{-l}\left[x_j(n)\right] = x_j(n-l)$,

$x_j(n-l)$ — входной сигнал, задержанный на l тактов дискретизации.







Тогда выходной сигнал $y_k(n)$ можно представить как бесконечную взвешенную сумму текущих и предыдущих значений входного сигнала $x_j(n)$:

$$y_k(n) = \sum_{l=0}^{\infty} w^{l+1} x_j(n-l).$$

На рисунках показано как зависит динамика реакции системы от веса w.

1. $|w| < 1$. Выходной сигнал $y_k(n)$ экспоненциально сходится, то есть система устойчива.
2. $|w| \geqslant 1$. Выходной сигнал $y_k(n)$ расходится, то есть система неустойчива. Если $|w| = 1$, расходимость является линейной, если $|w| > 1$ — экспоненциальной.

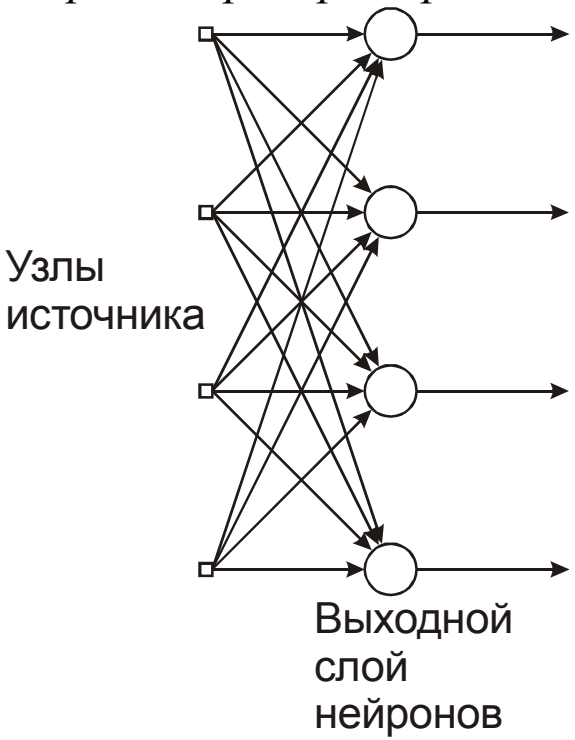
Случай $|w| < 1$ соответствует системам с бесконечной памятью. Влияние *вырождается* с течением времени.

2 лекция

Архитектура сетей. Однослойные сети прямого распространения

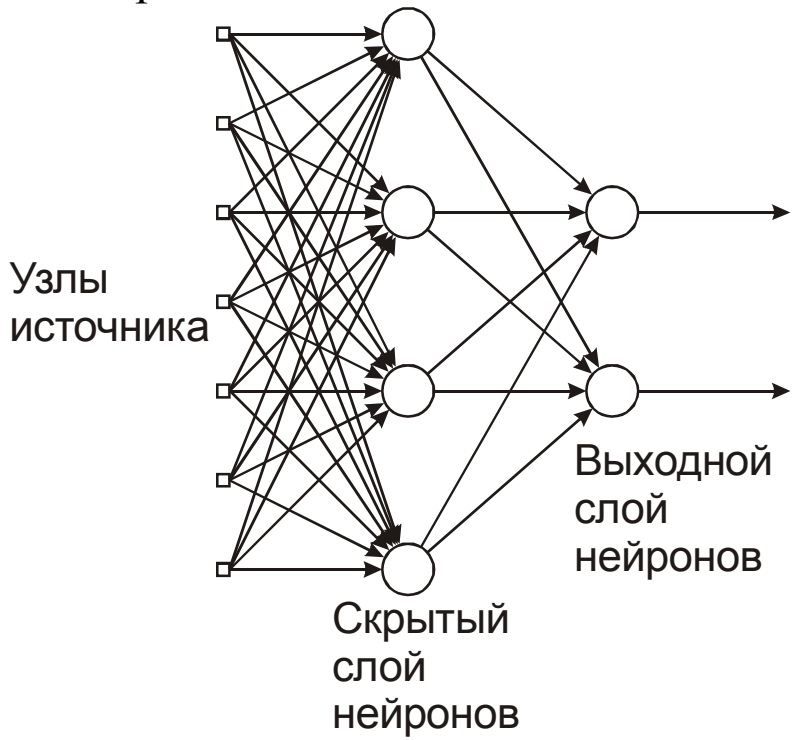
Структура нейронных сетей тесно связана с используемыми алгоритмами обучения.

Простейшая нейронная сеть, в которой информация передаётся от узлов источника на выходной слой нейронов, называется *однослойной сетью прямого распространения* (*ациклической сетью*).



Многослойные сети прямого распространения

В многослойных сетях прямого распространения между узлами источника и выходным слоем нейронов располагается как минимум один *скрытый слой*. Такая сеть позволяет выделять глобальные свойства данных с помощью локальных соединений за счёт наличия дополнительных синаптических связей и повышения уровня взаимодействия нейронов.

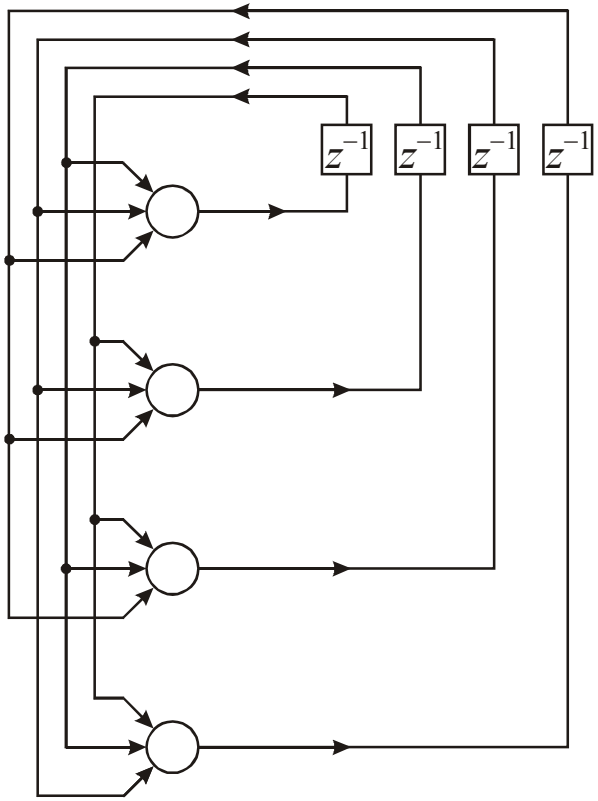


Нейронная сеть, показанная на рисунке, имеет 7 входов, 4 нейрона в скрытом слое и 2 нейрона в выходном слое. Сеть является *полносвязной* так как имеются все связи между всеми узлами соседних слоёв и между входными узлами и первым скрытым слоем. Если некоторые из синаптических связей отсутствуют, такая связь называется *неполносвязной*.

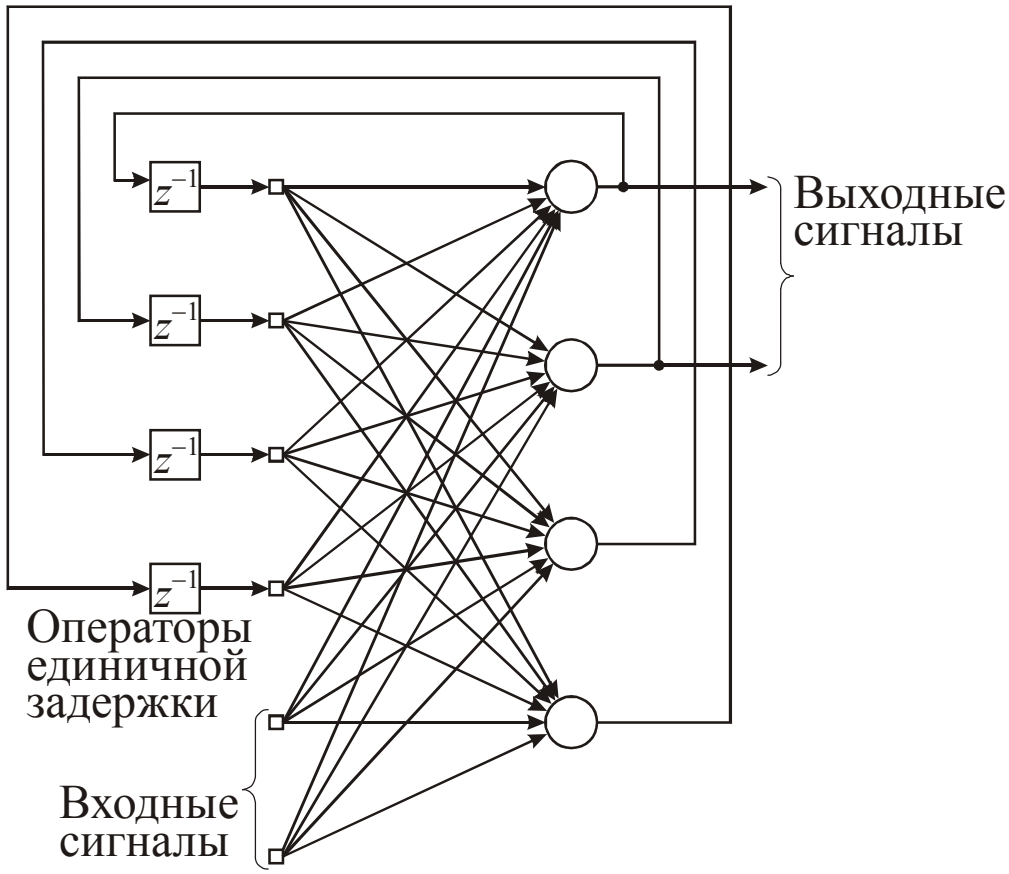
Рекуррентные сети

Рекуррентная нейронная сеть отличается от сети прямого распространения наличием, по крайней мере, одной обратной связи.

На рисунке представлена рекуррентная сеть без скрытых нейронов и обратных связей нейронов с самими собой



Рекуррентная сеть со скрытыми нейронами. Обратные связи исходят как из скрытых, так и из выходных нейронов



Представление знаний

Под знаниями понимается хранимая информация или модели, используемые человеком или машиной для интерпретации, предсказания и реакции на внешние события.

Вопросы представления знаний:

- какую информацию хранить;
- как информацию представить физически для её последующего использования.

Основная задача нейронной сети: наилучшее обучение модели окружающего мира для решения поставленной задачи. Знания о мире включают два типа информации.

1. *Априорная*
2. *Примеры*

Примеры: *маркированные* (входному сигналу соответствует *желаемый отклик*), *не маркированные* (несколько реализаций одного входного сигнала).

Множество пар сигналов вход-выход — *обучающие данные* (*обучающая выборка*).

Пример задачи *распознавания цифр*.

Входной сигнал: матрица из чёрных и белых точек. Каждое изображение: одна из десяти рукописных цифр на белом фоне. Желаемый отклик сети: конкретная цифра, изображение которой подаётся в качестве входного сигнала. Обучающая выборка состоит из большого числа рукописных цифр (ситуации, которые могут возникнуть в реальном мире).

Создание нейронной сети для данного примера.

1. (*режим обучения*) Выбор нейросетевой структуры, в которой размер входного слоя соответствует количеству пикселей на рисунке, а в выходном слое содержится десять нейронов, соответствующих цифрам. Далее выполняется настройка весовых коэффициентов сети на основе обучающего множества.
2. (*режим обобщения*) Эффективность обучения сети проверяется на множестве примеров, отличных от использованных при обучении. При этом на вход сети подаётся изображение, для которого известен целевой выход сети. Сравниваются результаты распознавания.

Набор данных, используемый для обучения сети, должен содержать как положительные, так и отрицательные примеры.

В нейронной сети заданной архитектуры знания об окружающей среде представляются множеством свободных параметров (синаптических весов и порогов) сети.

Правило 1. Сходные входные сигналы от схожих классов должны формировать единое представление в нейронной сети. Они должны быть классифицированы как принадлежащие к одной категории.

Обычно степень подобия определяется на основе *евклидова расстояния*.

Рассмотрим некоторый вектор $\mathbf{x}_i$ размерности m, элементы которого действительные числа

$$\mathbf{x}_i = \left[x_{i1}, x_{i2}, ..., x_{im}\right]^{\mathrm{T}}.$$

Вектор $\mathbf{x}_i$ определяет некоторую точку в m-мерном евклидовом пространстве $\Re^m$. *Евклидово расстояние* между парой m-мерных векторов $\mathbf{x}_i$ и $\mathbf{x}_j$ вычисляется как

$$d\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left\|\mathbf{x}_i - \mathbf{x}_j\right\| = \sqrt{\sum_{k=1}^{m}\left(x_{ik} - x_{jk}\right)^2},$$

где $\mathbf{x}_{ik}$ и $\mathbf{x}_{jk}$ — k-е элементы векторов $\mathbf{x}_i$ и $\mathbf{x}_j$ соответственно. То есть сходство между входными сигналами является величиной обратной евклидову расстоянию $d\left(\mathbf{x}_i, \mathbf{x}_j\right)$.

Другой подход определения сходства: вычисление *скалярного произведения*. Если векторы $\mathbf{x}_i$ и $\mathbf{x}_j$ имеют одинаковую размерность, их скалярное произведение $\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j$ определяется следующим образом:

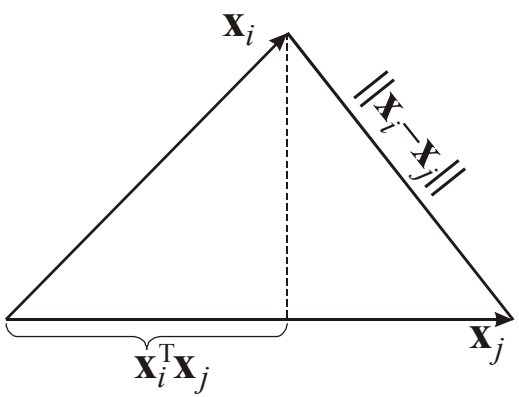
$$\left(\mathbf{x}_i, \mathbf{x}_j\right) = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j = \sum_{k=1}^{m} x_{ik}x_{jk}$$

Результат деления произведения $\left(\mathbf{x}_i, \mathbf{x}_j\right)$ на $\left\|\mathbf{x}_i\right\| \left\|\mathbf{x}_j\right\|$ равен косинусу внутреннего угла между векторами $\mathbf{x}_i$ и $\mathbf{x}_j$.

Два метода тесно связаны друг с другом.

Евклидово расстояние $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|$ между векторами $\mathbf{x}_i$ и $\mathbf{x}_j$ связано с проекцией вектора $\mathbf{x}_i$ на вектор $\mathbf{x}_j$.

Иллюстрация взаимосвязи между скалярным произведением и евклидовым расстоянием, выбираемым в качестве меры сходства образов. Из рисунка видно, что чем меньше евклидово расстояние $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|$, тем больше скалярное произведение $\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j$.



Чтобы формализовать это соотношение, нормируем векторы $\mathbf{x}_i$ и $\mathbf{x}_j$. При этом их длина $\left\|\mathbf{x}_i\right\| = \left\|\mathbf{x}_j\right\| = 1$.

$$d^2\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left(\mathbf{x}_i - \mathbf{x}_j\right)^{\mathrm{T}}\left(\mathbf{x}_i - \mathbf{x}_j\right) = 2 - 2\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j$$

Из выражения видно, что минимизация евклидова расстояния $\left\|\mathbf{x}_i - \mathbf{x}_j\right\|$ и максимизация скалярного произведения $\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j$ приводят к увеличению сходства между векторами $\mathbf{x}_i$ и $\mathbf{x}_j$.

Если случайные векторы $\mathbf{x}_i$ и $\mathbf{x}_j$ взяты из разных множеств данных, различие между двумя множествами данных выражается в различии между векторами их математического ожидания.

Пусть векторы $\boldsymbol{\mu}_i$ и $\boldsymbol{\mu}_j$ определяют средние значения векторов $\mathbf{x}_i$ и $\mathbf{x}_j$ соответственно, то есть

$$\boldsymbol{\mu}_i = E\left[\mathbf{x}_i\right],\ \boldsymbol{\mu}_j = E\left[\mathbf{x}_j\right],$$

где E — статистический оператор математического ожидания.

Для измерения расстояния между двумя множествами используют *расстояние Махаланобиса* d_{ij}.

$$d_{ij}^2 = \left(\mathbf{x}_i - \boldsymbol{\mu}_i\right)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \left(\mathbf{x}_j - \boldsymbol{\mu}_j\right)$$

где $\boldsymbol{\Sigma}^{-1}$ — обратная матрица для матрицы ковариации $\boldsymbol{\sigma}$. Предполагается, что матрица ковариации для обоих множеств одна и та же, то есть

$$\boldsymbol{\Sigma} = E\left[\left(\mathbf{x}_i - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_i - \boldsymbol{\mu}_i\right)^{\mathrm{T}}\right] = E\left[\left(\mathbf{x}_j - \boldsymbol{\mu}_j\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_j^{\mathrm{T}}\right)\right].$$

В теории вероятностей ковариационная матрица составляется из попарных ковариаций (линейной зависимости двух случайных величин) элементов одного или двух случайных векторов. Квадратная симметричная неотрицательно определённая, на диагонали которой располагаются дисперсии компонент вектора, а внедиагональные элементы — ковариации между компонентами.

В частном случае, когда $\mathbf{x}_i = \mathbf{x}_j, \boldsymbol{\mu}_i = \boldsymbol{\mu}_j = \boldsymbol{\mu}$ и $\boldsymbol{\Sigma} = \mathbf{I}$, где $\mathbf{I}$ — единичная матрица, расстояние Махаланобиса вырождается в Евклидово расстояние между вектором $\mathbf{x}_i$ и вектором математического ожидания $\boldsymbol{\mu}_i$.

Правило 2. Элементы, отнесённые к различным классам, должны иметь в сети как можно более отличные представления.

Правило 3. Если некоторое свойство имеет важное значение, то для его представления в сети необходимо использовать большое количество нейронов.

Например, в *критерии Неймана-Пирсона вероятность обнаружения* должна быть максимальной, а *вероятность ложной тревоги* не должна превосходить некоторой заданной величины. То есть, в процесс принятия решения о наличии цели должно быть вовлечено большое число нейронов.

Правило 4. В структуру нейронной сети должны быть встроены априорная информация и инварианты, что упрощает структуру сети и процесс её обучения.

Правильная конфигурация сети обеспечивает её *специализацию*.

Нейронная сеть со специализированной структурой включает значительно меньшее количество свободных параметров, которые нужно настраивать (в отличие от полносвязной сети). На обучение сети будет тратиться меньше времени, и сеть будет обладать лучшей обобщающей способностью. Сеть будет обладать большей пропускной способностью, и стоимость её создания будет также ниже.

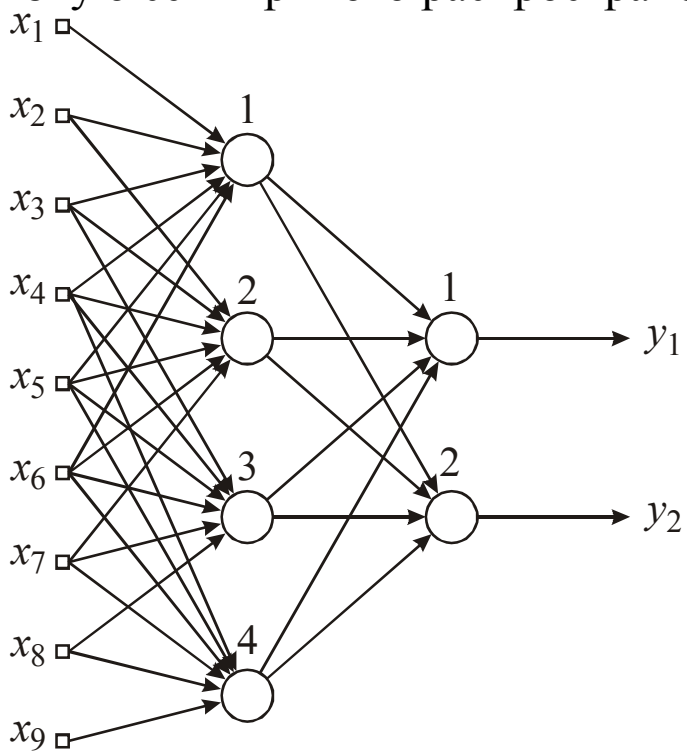
Встраивание априорной информации в структуру нейронной сети

Для выполнения правила 4 нет чёткого решения.

Есть некоторые предложения:

1. *Ограничение сетевой архитектуры* с помощью локальных связей, называемых *рецепторными полями.*
2. *Ограничение выбора синаптических весов* за счёт *совместного использования весов*.

Например, рассмотрим неполносвязную сеть прямого распространения.



Совместное использование весов состоит в применении одинакового набора синаптических весов для каждого из нейронов скрытого слоя сети.

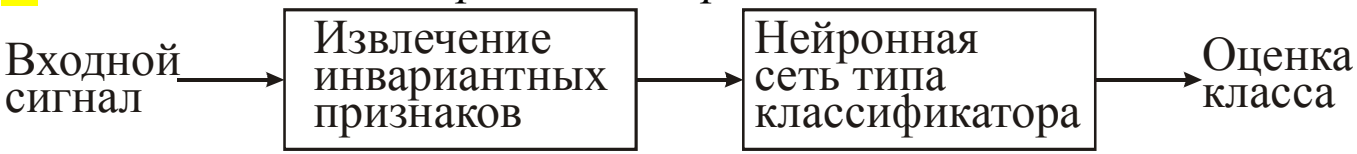
$$\upsilon_j = \sum_{i=1}^{6} w_i x_{i+j-1}, \quad j = 1, 2, 3, 4,$$

где $\{w_i\}_{i=1}^{6}$ определяет один и тот же набор весов, совместно используемых всеми четырьмя скрытыми нейронами, x_k — сигналы, получаемые из узла с источником с номером $k = i + j - 1$, а само выражение записано в виде свёрточной суммы, то есть это пример *свёрточной сети*.

Встраивание инвариантов в структуру нейронной сети

Вращение объекта, эффект Доплера, разная громкость звучания и т. п. — *трансформации* наблюдаемого сигнала. На результат классификации не должны влиять трансформации сигнала, должен быть учтён диапазон трансформаций.

1. *Структурная инвариантность*. Синаптические связи строятся так, чтобы трансформированные версии одного и того же сигнала вызывали один и тот же сигнал. (недостаток: количество синаптических связей слишком большое)
2. *Инвариантность по обучению*. Обучение на разных трансформированных вариантах (недостаток: сеть, наученная на одном классе, не обязательно будет распознавать трансформации других классов; ресурсоёмкая задача)
3. *Использование инвариантных признаков*.



из входного сигнала выделяют информативные признаки, описывающие самую существенную информацию, инвариантную к трансформациям входного сигнала. Здесь предъявляются требования хорошего знания специфики проблемы.

Искусственный интеллект и нейронные сети

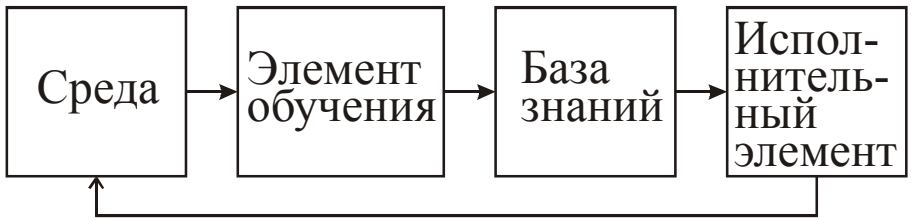
Основной задачей *искусственного интеллекта* (ИИ) является разработка парадигм или алгоритмов, обеспечивающих компьютерное решение когнитивных (познание, изучение, осознание — умственное восприятие и переработка внешней информации) задач, свойственных человеческому мозгу: накопление знаний, применение накопленных знаний для решения проблемы и извлечение знаний из опыта. Системы ИИ реализуют три ключевые функции.

1. *Представление*. Использование символьного языка для представления общих знаний о предметной области и конкретных знаний о способах решения задачи. Знания могут иметь процедурный (внедрение знаний в процедуры, функционирующие независимо от смысла самих знаний) и декларативный (статический набор фактов) характер.

2. *Рассуждение*. Способность решать задачи. Требования: описывать и решать широкий спектр задач. Понимать явную и неявную информацию. Иметь механизм управления, определяющий операции, используемые для решения отдельных задач.

Решение задач рассматривают как задачу *поиска*, в процессе которого используются *правила*, *данные* и *управляющие воздействия*. Если набор знаний является неполным или неточным, то используются *вероятностные рассуждения*.

3. *Обучение*. В простейшем случае информацию для обучения предоставляет сама среда



Обучаемый элемент использует информацию для коррекции базы знаний. Так как информация из внешней среды несовершенна, то обучаемый элемент выдвигает рабочие гипотезы. Для проверок рабочих гипотез используется механизм обратной связи.

Машинное обучение может включать 2 способа обработки информации: *индуктивный* (на основе практического опыта и потоков данных — подобия) и *дедуктивный* (для определения конкретных фактов используются общие правила —теоремы).

При использовании накопленного опыта (людьми) в создании методов и алгоритмов пополнения баз данных применяются *экспертные системы*.

Сравнение когнитивных моделей нейронных сетей с символьными системами искусственного интеллекта.

1. *Уровень объяснения*. С точки зрения познания ИИ предполагает существование ментального представления, в котором познание осуществляется как *последовательная обработка* символьной информации. В моделях *параллельной распределённой обработки*, обработка информации происходит за счёт взаимодействия большого количества нейронов, каждый из которых передаёт сигналы возбуждения и торможения другим нейронам сети

2. *Стиль обработки*. В классических системах ИИ обработка происходит последовательно (программы для процессора). Концепция обработки информации в нейронных сетях: принцип *параллелизма*.

3. *Структура представления*. В классических системах ИИ моделью является язык мышления (квази-лингвистическая структура), новые смысловые выражения строятся на основе композиции символьных выражений и аналогии между синтаксической структурой и семантикой. В нейронных сетях природа и структура представления являются проблемами.

То есть при решении когнитивных задач целесообразно создавать *структурированные модели на основе связей* или *гибридные системы*, объединяющие оба подхода. Интегрирование нейронных сетей с интеллектуальными машинами обеспечивает решение следующих задач.

▲ Верификация нейросетевых компонентов в программных системах. Внутреннее состояние нейронной сети переводится в форму, понятную пользователям.

▲ Улучшение обобщающей способности нейронной сети за счёт выявления областей входного пространства, не достаточно полно представленных в обучаемом множестве, а также определение условий, при которых обобщение невозможно.

▲ Выявление скрытых зависимостей на множестве входных данных.

▲ Интеграция символьного и коннекционистского (установления связей в сетях из связанных между собой из простых элементов) подходов при разработке интеллектуальных машин.

▲ Обеспечение безопасности систем, для которых эта безопасность критична

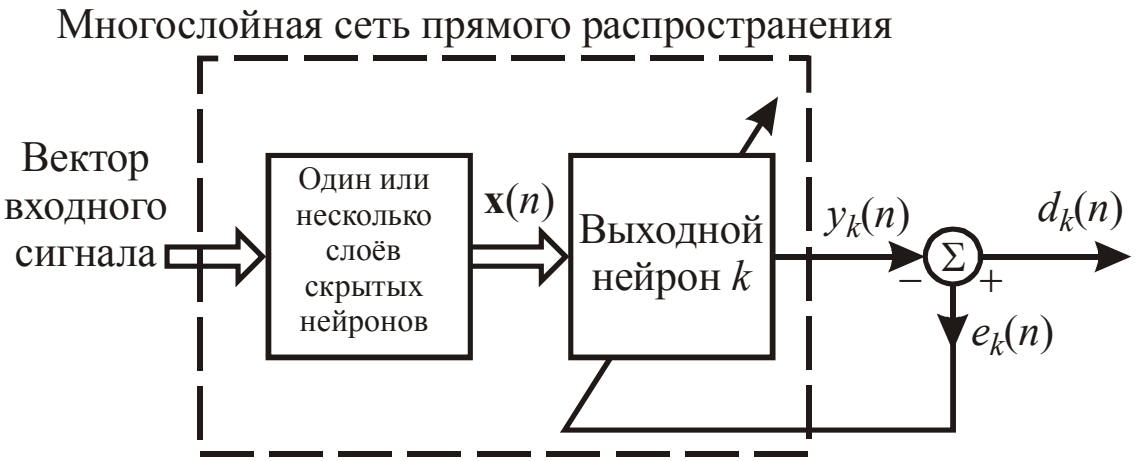
3 лекция
Процессы обучения

Обучение нейронной сети происходит посредством интерактивного процесса корректировки синаптических весов и порогов.

Обучение — процесс в котором свободные параметры нейронной сети настраиваются посредством моделирования среды, в которую эта сеть встроена. Тип обучения определяется способом подстройки этих параметров.

Обучение, основанное на коррекции ошибок

Дан пример нейрона k — единственного вычислительного узла выходного слоя нейронной сети прямого распространения



Нейрон k работает под управлением *вектора сигнала* $x(n)$, производимого одним или несколькими скрытыми слоями нейронов, которые, в свою очередь, получают информацию из входного вектора, передаваемого начальным узлам нейронной сети. Выходной сигнал $y_k(n)$ сравнивается с желаемым сигналом $d_k(n)$. Сигнал ошибки $e_k(n) = d_k(n) - y_k(n)$ инициализирует *механизм управления*, цель которого заключается в применении последовательности корректировок к синаптическим весам нейрона k. Эти изменения нацелены на пошаговое приближение выходного сигнала $y_k(n)$ к желаемому $d_k(n)$. Эта цель достигается за счёт минимизации *функции стоимости* (*индекса производительности*) $\mathbf{E}(n)$, определяемой в терминах сигнала ошибки: $\mathbf{E}(n) = \frac{1}{2} e_k^2(n)$ — *текущее значение энергии ошибки*.

Пошаговая корректировка синаптических весов нейрона k продолжается до тех пор, пока система не достигнет *устойчивого состояния*.

Минимизация функции стоимости $\mathbf{E}(n)$ выполняется по правилу Видроу-Хоффа (дельта-правилу).

В соответствии с этим правилом изменение текущего значения синаптического веса определяется:

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n)$$

здесь η — положительная константа, определяющая *скорость обучения*.

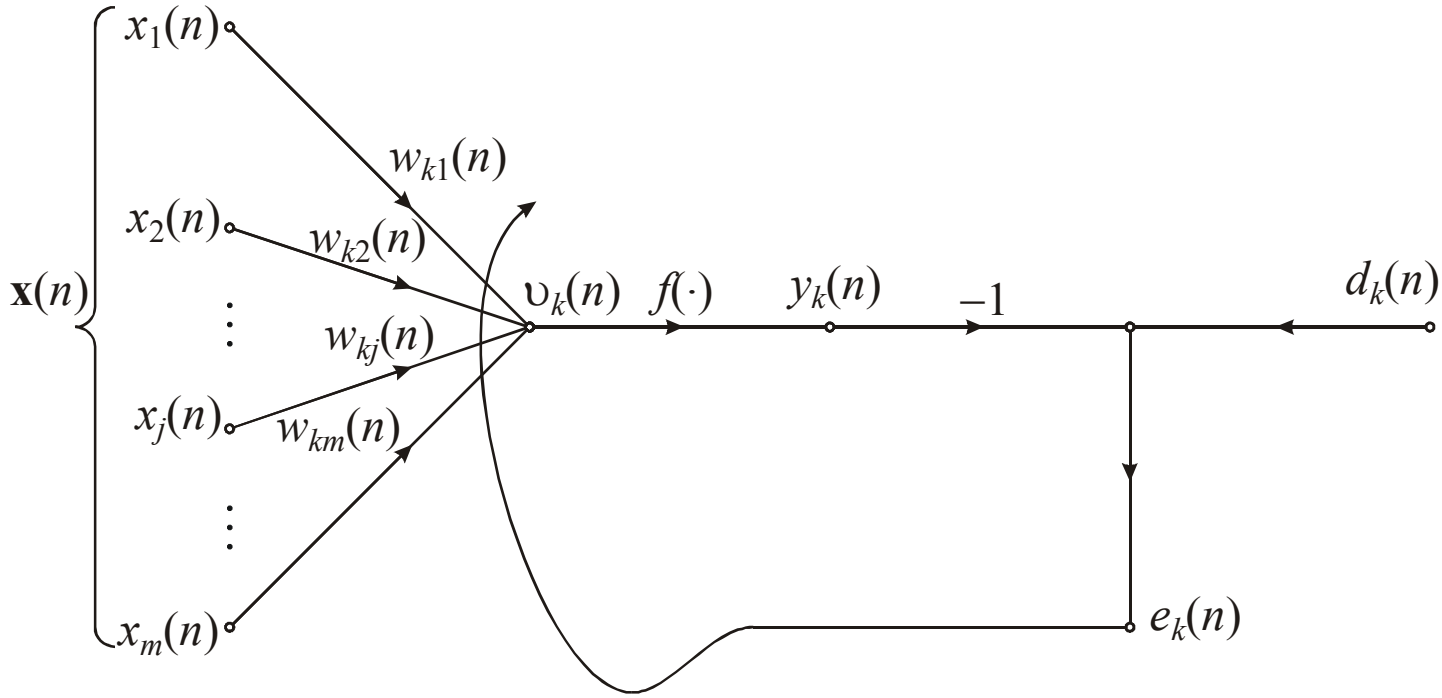
Корректировка, применяемая к синаптическому весу нейрона, пропорциональна произведению сигнала ошибки на входной сигнал, его вызвавший.

Новое значение синаптического веса для следующего шага дискретизации: $w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$.

Другая запись: $w_{kj}(n) = z^{-1}\left[w_{kj}(n+1)\right]$.

Здесь z^{-1} — оператор единичной задержки (элемент памяти).

Граф прохождения сигнала в процессе обучения, основанного на коррекции ошибок, для выделенного нейрона *k*.



Входной сигнал x_k и индуцированное локальное поле υ_k нейрона k представлены в виде *предсинаптического* и *постсинаптического* сигналов *j*-го синапса нейрона *k*.

То есть обучение на основе коррекции ошибок — пример замкнутой системы с обратной связью. Устойчивость такой системы определяется коэффициентом скорости обучения η.

Обучение на основе памяти

Весь прошлый опыт накапливается в большом хранилище правильно классифицированных примеров вида вход-выход: $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$, где $\mathbf{x}_i$ — входной вектор, а d_i — соответствующий ему желаемый сигнал.

Алгоритмы обучения на основе памяти включают 2 составляющие.

1. Критерий, используемый для определения окрестности вектора $\mathbf{x}_{\text{test}}$.
2. Правило обучения, применяемое к примеру из окрестности тестового вектора.

Различаются способы реализации этих составляющих.

Правило ближайшего соседа — в окрестность включается пример, ближайший к тестовому, например, вектор $\mathbf{x}'_N \in \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ считается ближайшим соседом вектора $\mathbf{x}_{\text{test}}$, если выполняется условие $\min_i d(\mathbf{x}_i, \mathbf{x}_{\text{test}}) = d(\mathbf{x}'_N, \mathbf{x}_{\text{test}})$, где $d(\mathbf{x}_i, \mathbf{x}_{\text{test}})$ — евклидово расстояние между векторами $\mathbf{x}_i$ и $\mathbf{x}_{\text{test}}$.

Исследование правила ближайшего соседа основывается на двух предположениях

1. Классифицируемые примеры $(\mathbf{x}_i, d_i)$ независимы и равномерно распределены в соответствии с совместным распределением $(\mathbf{x}, d)$.
2. Размерность обучающего множества N бесконечно велика.

Другим классификатором является *классификатор k-ближайших соседей*. Работает подобно устройству усреднения исключая единичные выбросы.

Обучение Хебба

Если аксон клетки А находится на достаточно близком расстоянии от клетки В и постоянно или периодически участвует в её возбуждении, наблюдается процесс метаболических изменений в одном или обоих нейронах, выражающийся в том, что эффективность нейрона А как одного из возбудителей нейрона В возрастает.

По-другому:

1. Если 2 нейрона по обе стороны синапса (соединения) активизируются одновременно (синхронно), то прочность этого соединения возрастает.
2. Если 2 нейрона по обе стороны синапса активизируются асинхронно, то такой синапс ослабляется или вообще отмирает.

Синапс Хебба использует зависящий от времени, в высшей степени локальный механизм взаимодействия, изменяющий эффективность синаптического соединения в зависимости от корреляции между предсинаптической и постсинаптической активностью.

Четыре свойства, характеризующие синапс Хебба.

1. *Зависимость от времени*. Зависит от точного времени возникновения предсинаптического и постсинаптического сигналов.
2. *Локальность*. Синапс Хебба является узлом передачи данных, в котором информационные сигналы находятся в пространственно-временной близости.
3. *Интерактивность*. Изменения в синапсе Хебба определяются сигналами на обоих его концах и имеют детерминированный или статистический характер.
4. *Корреляция*. Условием изменения эффективности синаптической связи является зависимость между предсинаптическим и постсинаптическим сигналами.

Усиление и ослабление синаптической связи

Синаптическая связь по модели Хебба усиливается при положительной корреляции предсинаптических и постсинаптических сигналов и ослабляется в противном случае — *хеббовская модель*.

Синаптическая связь ослабляется при положительной корреляции предсинаптических и постсинаптических сигналов и усиливается в противном случае — *антихеббовская модель.*

Нехеббовская модель не связана с механизмом модификации, предложенным Хеббом.

Математические модели предложенного Хеббом механизма модификации синаптической связи

Изменение синаптического веса w_{kj} в момент времени n с предсинаптическим и постсинаптическим сигналами x_j и y_k

$$\Delta w_{kj}(n) = F\left(y_k(n), x_j(n)\right).$$

Здесь $F(\cdot,\cdot)$ — некоторая функция, зависящая от предсинаптического и постсинаптического сигналов.

Одна из форм записи: гипотеза Хебба

Правило умножения активности

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n)$$

η — скорость обучения (положительная константа)

Другая форма записи: гипотеза ковариации

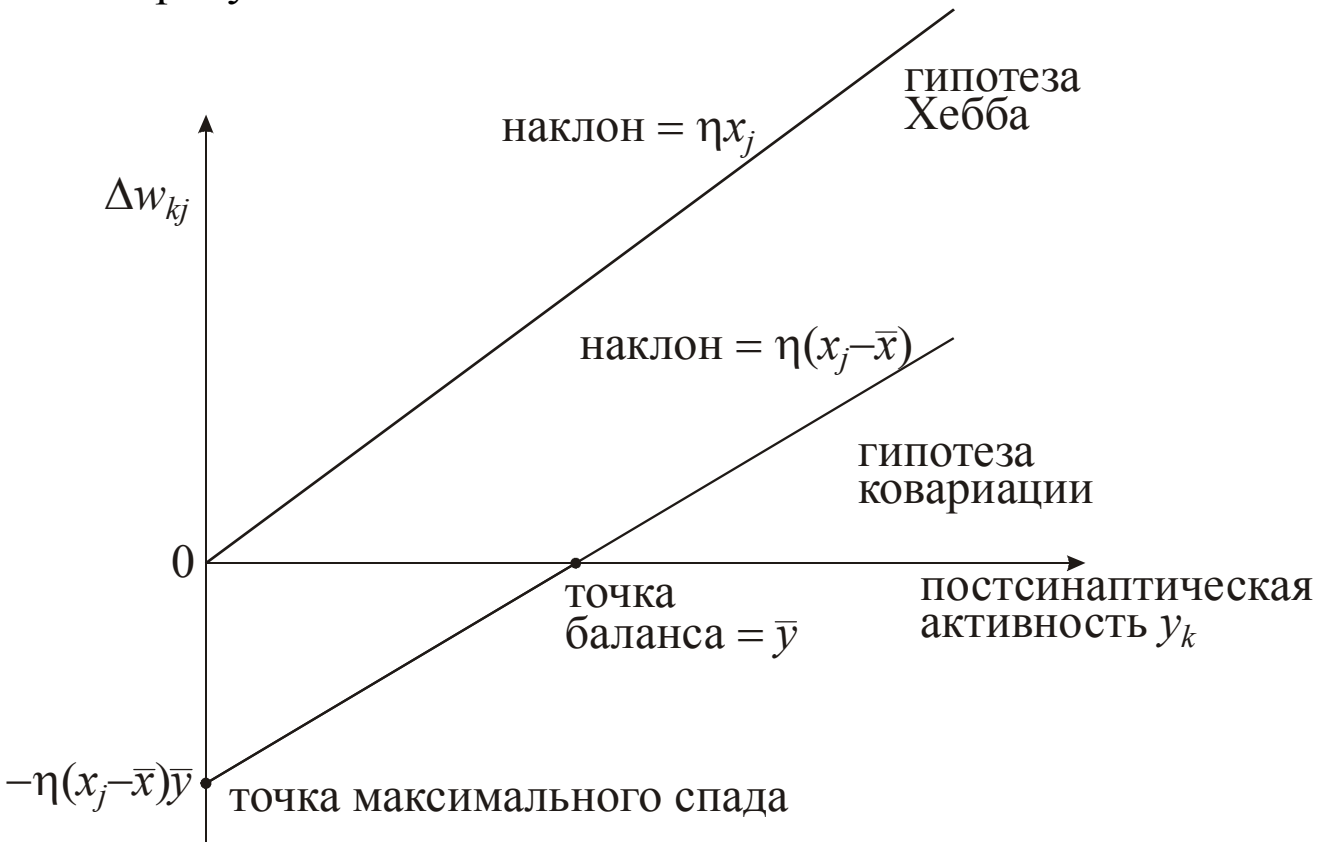
Предсинаптический и постсинаптический сигналы заменяются отклонениями этих сигналов от их средних значений на данном отрезке времени

$$\Delta w_{kj} = \eta\left(x_j - \overline{x}\right)\left(y_k - \overline{y}\right)$$

$\overline{x}$ и $\overline{y}$ — усреднённые по времени значения, содержащие предсинаптический и постсинаптический пороги, которые определяют знак синаптической модификации.

Гипотеза ковариации обеспечивает сходимость к нетривиальному состоянию при $x_k = \overline{x}$ и $y_j = \overline{y}$ и прогнозирование усиления (ослабления) синаптической связи.

Отличие гипотез показано на рисунке



Выводы:

1. Синаптический вес связи усиливается при высоком уровне предсинаптического и постсинаптического сигналов, то есть когда одновременно выполняются условия: $x_j > \overline{x}$ и $y_k > \overline{y}$.

2. Синаптический вес ослабляется когда предсинаптическая активность $x_j > \overline{x}$ не вызывает постсинаптической активности $y_k < \overline{y}$ или постсинаптическая активность $y_k > \overline{y}$ возникает при отсутствии существенной предсинаптической активности $x_j < \overline{x}$.

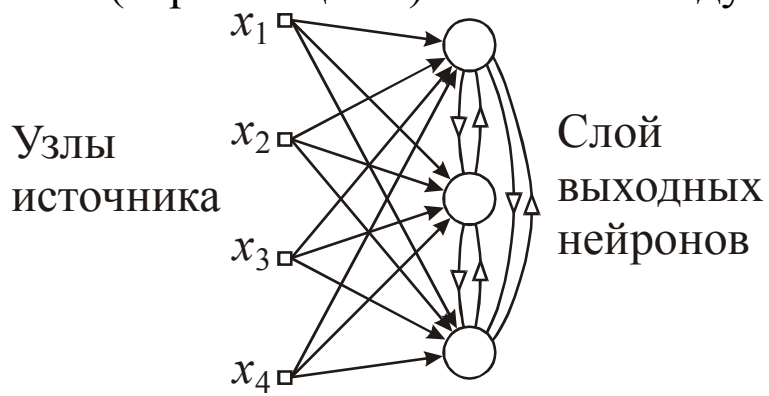
Такое поведение рассматривается как форма временно́й конкуренции между входными моделями.

Конкурентное обучение

В отличие от нейронной сети, основанной на обучении Хебба, в которой одновременно в возбуждённом состоянии может находиться несколько нейронов, в конкурентной сети в каждый момент времени может находиться только один нейрон.

Каждый отдельный нейрон сети соответствует группе близких образов (*детекторы признаков* различных классов входных образов).

Структурный граф простой сети конкурентного обучения с прямыми (возбуждающими) связями от входных узлов к нейронам и обратными (тормозящими) связями между нейронами.

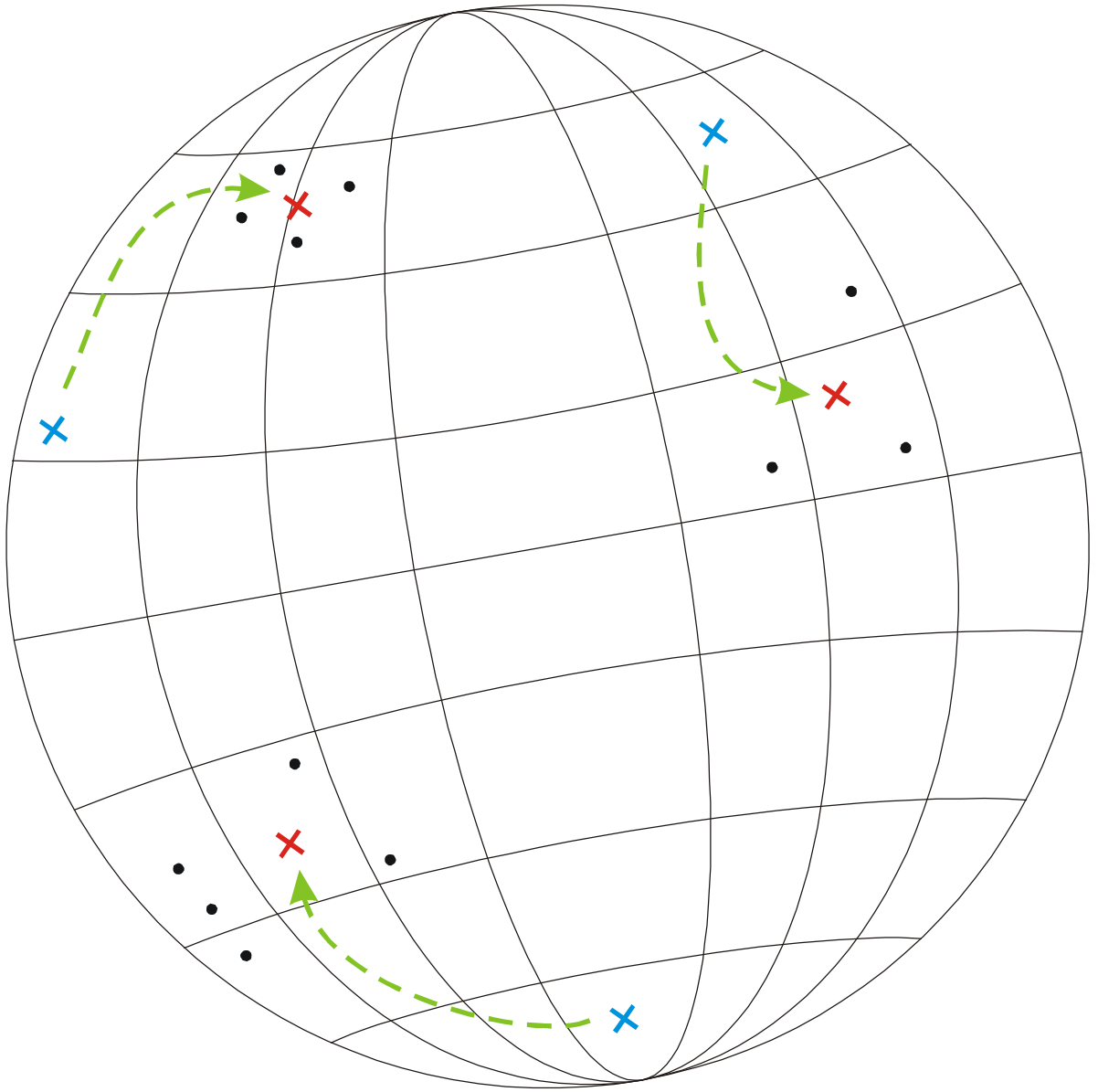


Условие победы нейрона k в конкурентной борьбе

$$y_k = \begin{cases} 1, & \upsilon_k > \upsilon_j,\ j \neq k, \\ 0 & \text{в остальных случаях} \end{cases}$$

индуцированное локальное поле υ_k представляет сводное возбуждение нейрона k от всех входных сигналов и сигналов обратной связи.

Геометрическая интерпретация процесса конкурентного обучения. Точки — входные векторы. Крестики — векторы синаптических весов трёх выходных нейронов в исходном (синим цветом) и конечном (красным цветом) состоянии сети. Все входные векторы и нейроны сети имеют евклидову норму $\sum_j w_{kj}^2 = 1, \forall k$



Продемонстрирована способность нейронной сети решать задачи *кластеризации* в процессе конкурентного обучения.

Обучение Больцмана

Представляет стохастический алгоритм обучения (машина Больцмана).

В машине Больцмана все нейроны представляются рекуррентными структурами, работающие с бинарными сигналами ($+1$ — включённое состояние и -1 выключенное состояние).

Функция энергии машины Больцмана

$$E = -\frac{1}{2}\sum_{j}\sum_{k(j\neq k)} w_{kj}x_k x_j$$

Её значение определяется конкурентными состояниями отдельных нейронов, составляющих эту машину. x_j — состояние нейрона j. w_{kj} — синаптический вес нейронов j и k. Условие $j \neq k$ означает, что в сети нейроны не имеют обратных связей с самими собой.

Работа этой машины заключается в свободном выборе некоторого нейрона на определённом шаге процесса обучения и переводе этого нейрона из состояния x_k в состояние $-x_k$ при некоторой температуре (псевдотемпературе) T с вероятностью $P(x_k \to -x_k) = \dfrac{1}{1+\exp(-\Delta E_k / T)}$,

где ΔE_k — изменение энергии машины, вызванное переходом состояний.

Рассматривают 2 режима функционирования такой сети

Скованное состояние — все видимые нейроны находятся в состояниях, предопределённых внешней средой.

Свободное состояние — все нейроны могут свободно функционировать.

Изменение синаптического веса w_{kj} связи между нейронами j и k

$\Delta w_{kj} = \eta(\rho_{kj}^{+} - \rho_{kj}^{-}),\ j \neq k,$

$\rho_{kj}^{+} \in (-1...+1)$ — корреляция между состояниями нейронов j и k когда сеть находится в скованном состоянии,

$\rho_{kj}^{-} \in (-1...+1)$ — корреляция между состояниями нейронов j и k когда сеть находится в свободном состоянии.

Задача присваивания коэффициентов доверия

Это задача присваивания коэффициентов доверия или недоверия всем результатам, полученным некоторой обучаемой машиной (*задача загрузки*).

Две подзадачи:

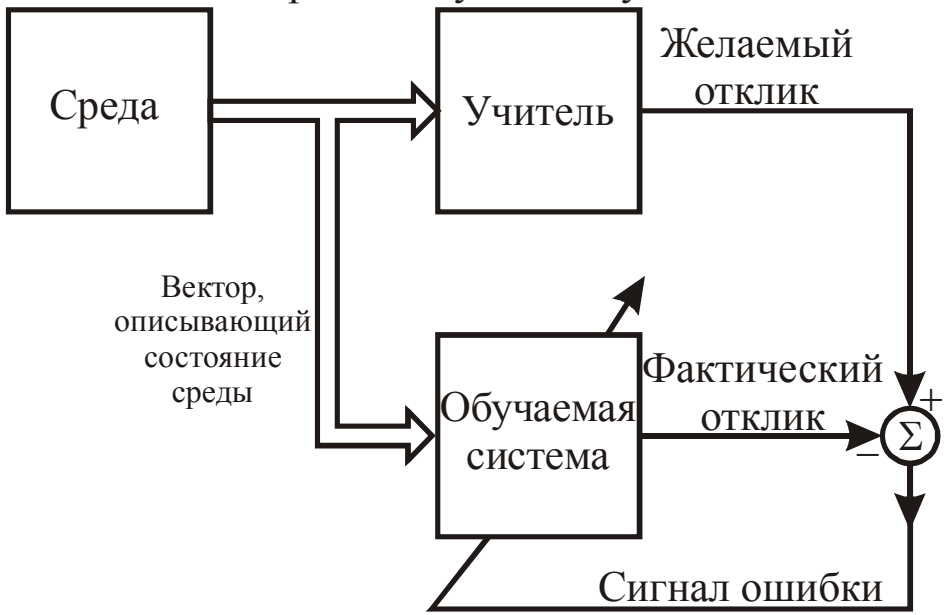
1. Присваивание коэффициентов доверия результатам действий (*временно́й задачей присваивания коэффициентов доверия*)
2. Присваивание коэффициентов доверия действий внутренним решениям (*структурная задача присваивания коэффициентов доверия*)

Структурная задача присваивания коэффициентов доверия имеет смысл в контексте многокомпонентных обучаемых машин, когда необходимо определить, поведение какого элемента системы нужно скорректировать и на какую величину, чтобы повысить общую производительность системы.

Временна́я задача присваивания коэффициентов доверия ставится в том случае, когда обучаемая машина выполняет достаточно много действий, приводящих к некоторому результату, требуется определить, какие из этих действий несут ответственность за результат.

Обучение с учителем

Блочная диаграмма обучения с учителем



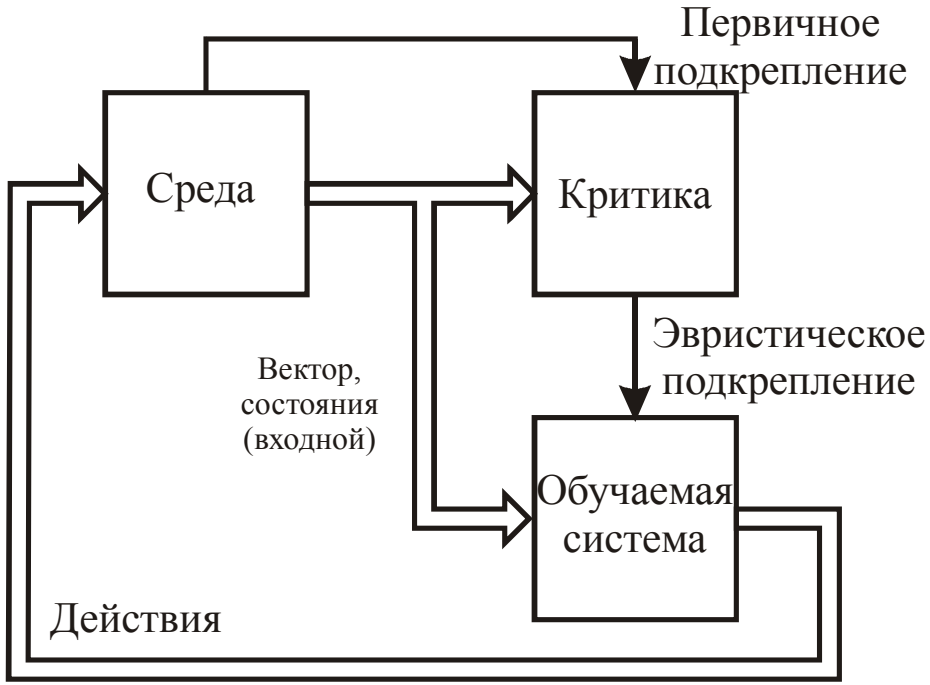
Концептуально участие учителя рассматривается как наличие знаний об окружающей среде, представленных в виде пар *вход-выход*. При этом сама среда неизвестна обучаемой нейронной сети. Параметры сети корректируются с учётом обучающего вектора и сигнала ошибки (разностью между желаемым сигналом и текущим откликом нейронной сети. Корректировка параметров выполняется пошагово с целью *имитации* нейронной сетью поведения учителя.

Производительность системы оценивается в терминах среднеквадратической ошибки, представленной в виде функции от свободных параметров. Для такой функции можно построить многомерную *поверхность ошибки* в координатах свободных параметров. При этом реальная поверхность ошибки усредняется по всем возможным примерам пар вход-выход. Значение сигнала ошибки смещается в сторону минимума на поверхности ошибок. Минимум может быть как локальным, так и глобальным. Система может обладать полезной информацией о *градиенте* поверхности ошибок. Градиент поверхности ошибок в любой точке — вектор, определяющий направление наискорейшего спуска по этой поверхности. При использовании соответствующего алгоритма минимизации функции стоимости, удачном наборе обучающих примеров (пар вход-выход) и достаточном времени на обучении системы обучения с учителем дают хорошие результаты.

Обучение без учителя, обучение с подкреплением (нейродинамическое программирование)

Формирование отображения входных сигналов в выходные выполняется в процессе взаимодействия с внешней средой с целью минимизации скалярного индекса производительности.

Блочная диаграмма обучения с подкреплением

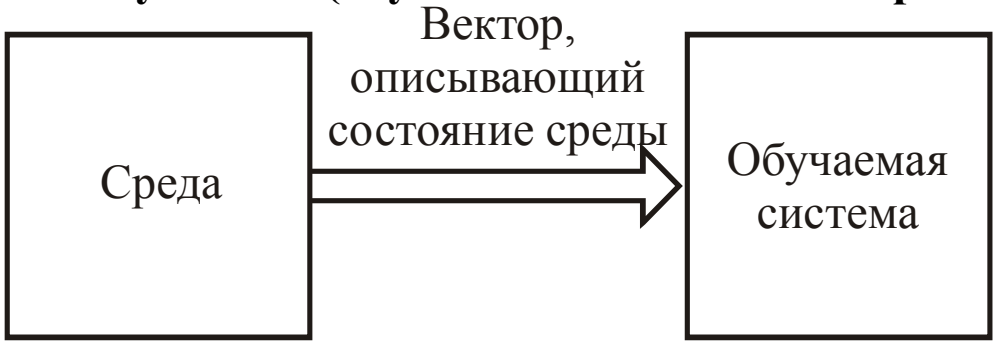


Такая система предполагает обучение с отложенным подкреплением, то есть система получает из внешней среды последовательность векторов состояния, которые приводят к генерации эвристического сигнала подкрепления. Целью обучения является минимизация функции стоимости перехода, определённой как математическое ожидание кумулятивной стоимости действий, предпринятых в течение нескольких шагов. Обучаемая машина определяет эти действия и формирует на их основе сигнал обратной связи во внешнюю среду.

Сложности. Не существует учителя, формирующего желаемый отклик на каждом шаге обучения.

Наличие задержки при формировании первичного сигнала подкрепления требует решения *вре́менной задачи присваивания коэффициентов доверия* на всех шагах, приводящих к конечному результату, в то время как первичный сигнал подкрепления формируется только на основе конечного результата. Такое обучение тесно связано с динамическим программированием, реализующего математический формализм последовательного принятия решений.

Обучение без учителя (обучение на основе самоорганизации)



Свободные параметры сети оптимизируются по отношению к мере качества представления, которому должна научиться нейронная сеть. После обучения на статистические закономерности входного сигнала нейронная сеть формирует внутреннее представление кодируемых признаков входных данных и автоматически создаёт новые классы.

Например, входной слой получает доступные данные, а выходной слой состоит из нейронов, конкурирующих за право отклика на признаки, содержащиеся во входных данных. Нейрон с наибольшим суммарным входным сигналом переходит в активное состояние, а остальные нейроны отключаются.

Задачи обучения

Выбор алгоритма обучения зависит от задач, решению которых обучают нейронные сети. Выделяются шесть основных задач.

1. Ассоциативная память
2. Распознавание образов
3. Аппроксимация функций
4. Управление
5. Фильтрация
6. Формирование диаграммы направленности

Ассоциативная память

Ассоциативная память — распределённая память, в которая запоминает образы ассоциаций.

2 типа ассоциативной памяти:

автоассоциативная — в нейронной сети запоминаются передаваемые ей образы (векторы) (используется обучение без учителя);

гетероассоциативная — произвольному набору входных образов ставится в соответствие другой произвольный набор входных данных (используется обучение с учителем).

Пример.

$\mathbf{x}_k$ — ключевой образ (вектор), применяемый для решения задач ассоциативной памяти;

$\mathbf{y}_k$ — запомненный образ (вектор).

Отношение ассоциации образов:

$\mathbf{x}_k \to \mathbf{y}_k, k = 1, 2, ..., q$,

где q — количество хранимых в сети образов (ёмкость памяти сети). Ключевой образ $\mathbf{x}_k$ выступает в роли стимула, определяет местоположение запомненного образа $\mathbf{y}_k$ и содержит ключ для его извлечения.

В ассоциативной памяти пространства входных и выходных данных имеют одинаковую размерность, а в гетероассоциативной она может быть неодинаковой.

В ассоциативной памяти имеется две фазы: фаза запоминания (процесс обучения $\mathbf{x}_k \to \mathbf{y}_k, k = 1, 2, ..., q$) и фаза восстановления (извлечение запомненного образа в ответ на представление в сеть зашумлённой или искажённой версии ключа).

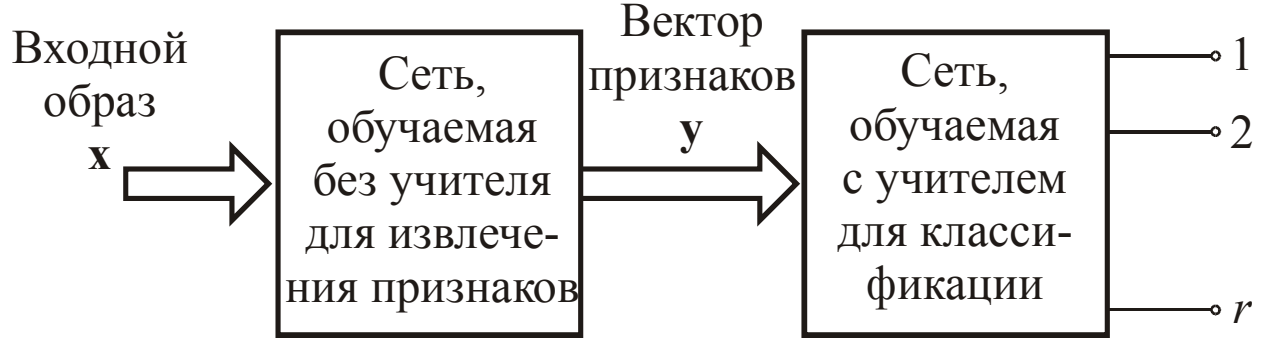
Распознавание образов

Распознавание образов — процесс, в котором получаемый образ/сигнал должен быть отнесён к одному из предопределённых классов (категорий).

Образы представляются отдельными точками в многомерном пространстве решений. Пространство решений разделяется на отдельные области, каждая из которых ассоциируется с определённым классом. Границы областей формируются в процессе обучения. Построение этих границ выполняется статистически на основе дисперсии, присущей данным отдельных классов.

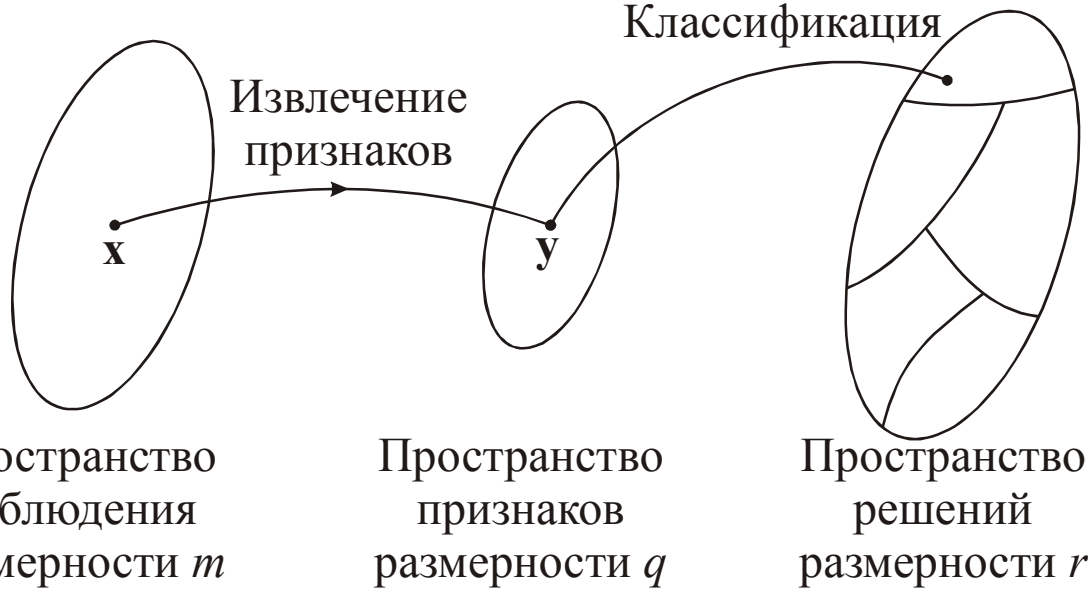
Машины распознавания образов, на основе нейронных сетей.

1. Система состоит из сети извлечения признаков и сети классификации.



Образ представляется как набор из m наблюдений, каждое из которых рассматривается как точка **x** в m-мерном пространстве наблюдений данных. Извлечение признаков — преобразование, которое переводит точку **x** в промежуточную точку **y** в q-мерном пространстве признаков ($q<m$, операция снижения размерности пространства, упрощающую задачу классификации). Классификация описывается как преобразование, отображающее точку **y** в один из классов r-мерного пространства решений (r — количество выделяемых классов)

2. Система проектируется как единая многослойная сеть прямого распространения, использующая один из алгоритмов обучения с учителем. Задача извлечения признаков выполняется вычислительными узлами скрытого слоя сети.



Аппроксимация функций

Является примером задачи для обучения с учителем.

Пусть нелинейное отображение «вход-выход», задано соотношением $\mathbf{d} = \mathbf{f}(\mathbf{x})$

Вектор $\mathbf{x}$ — вход, вектор $\mathbf{d}$ — выход (желаемый отклик), функция $\mathbf{f}(\cdot)$ неизвестна.

Для познания функции $\mathbf{f}(\cdot)$ предложено множество маркированных примеров $\mathbf{T} = \left\{\left(\mathbf{x}_i, \mathbf{d}_i\right)\right\}_{i=1}^{N}$.

Требование: функция $\mathbf{F}(\cdot)$, описывающая отображение сигнала в выходной, должна быть достаточно близка к функции $\mathbf{f}(\cdot)$ согласно евклидовой метрике на множестве всех входных векторов $\mathbf{x}$:

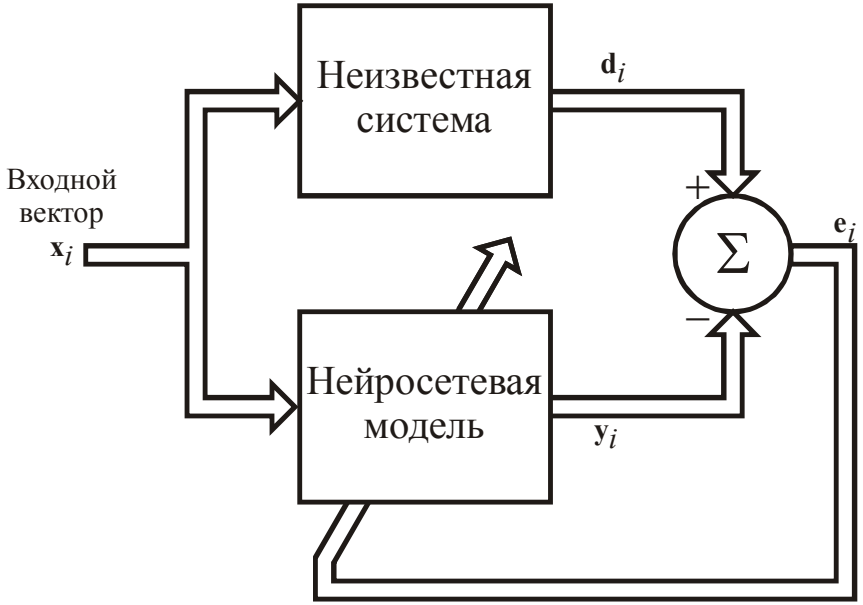
$\|\mathbf{F}(\mathbf{x}) - \mathbf{f}(\mathbf{x})\| < \varepsilon, \forall \mathbf{x}$,

ε — некоторое малое положительное число. Если N достаточно велико и в сети имеется достаточное количество свободных параметров, то ошибку аппроксимации ε можно сделать достаточно малой.

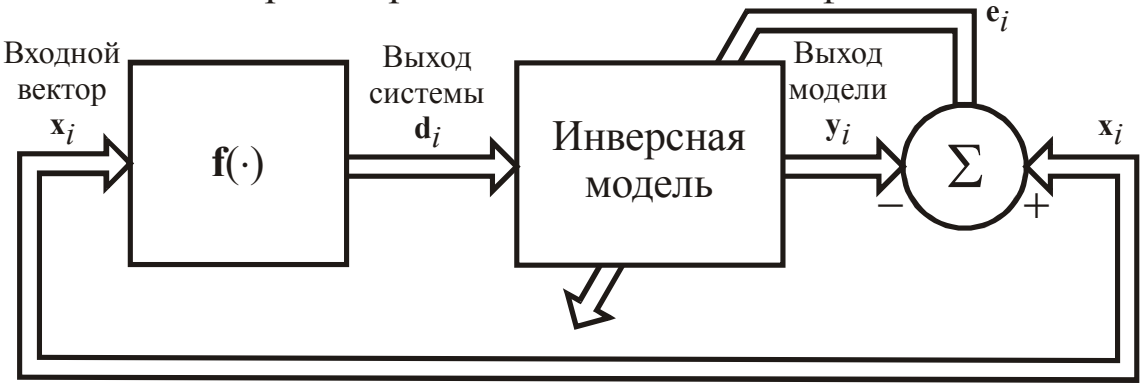
То есть задача обучения с учителем сводится к задаче аппроксимации.

Способность сети аппроксимировать неизвестное отображение входного пространства в выходное используется, например, при идентификации систем или для инверсных систем ($\mathbf{x} = \mathbf{f}^{-1}(\mathbf{d})$).

Блочная диаграмма решения задачи идентификации системы



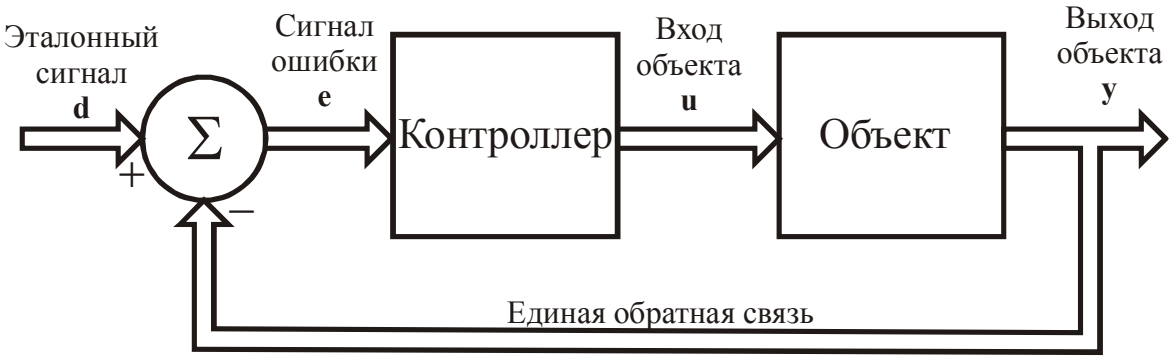
Блочная диаграмма решения задачи идентификации системы



Управление

Управление «предприятием» (процесс или критическая часть схемы, подлежащие управлению). Пример:

Блочная диаграмма системы управления с обратной связью



контроллер инвертирует отображение вход-выход объекта управления (настраивает свободные параметры), то есть его задача: поддержание такого входного вектора объекта, для которого выходной сигнал **y** соответствует эталонному значению **d**.

Для настройки свободных параметров объекта управления в соответствии с алгоритмом обучения на основе коррекции ошибок требуется знание матрицы Якоби:

$$\mathbf{J} = \left\{ \frac{\partial y_k}{\partial u_j} \right\}$$

y_k — элемент выходного сигнала **y** объекта управления; u_j — элемент вектора входов объекта **u**.

Так как частные производные для различных k и j зависят от рабочей точки объекта управления, то они неизвестны. Для их оценки используют следующие подходы.

Непрямое обучение. С использованием текущих измерений входа и выхода объекта управления строится модель нейронной сети, воспроизводящая эту зависимость (Оценивается матрица **J**).

Прямое обучение. Знаки частных производных известны и остаются постоянными в некотором динамическом диапазоне значений объекта управления, то есть частные производные можно аппроксимировать по их знакам. Абсолютные значения частных производных задаются распределённым представлением в свободных параметрах нейроконтроллера, то есть их можно настраивать непосредственно с помощью объекта управления.

Фильтрация

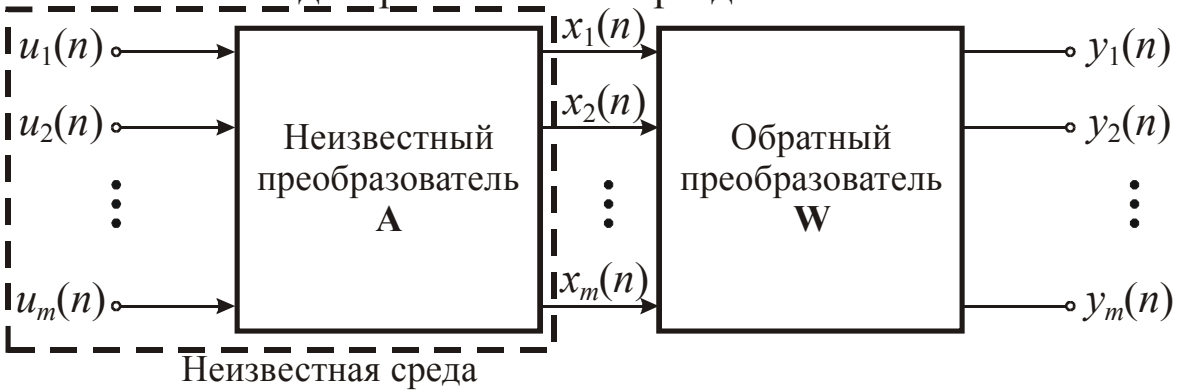
Задачи обработки информации

1. Фильтрация. Извлечение полезной информации в дискретные моменты времени из данных.
2. Сглаживание. Информация извлекается после некоторого запаздывания.
3. Прогнозирование. Прогноз относительно состояния объекта управления в некоторый последующий момент времени на основе данных полученных ранее.

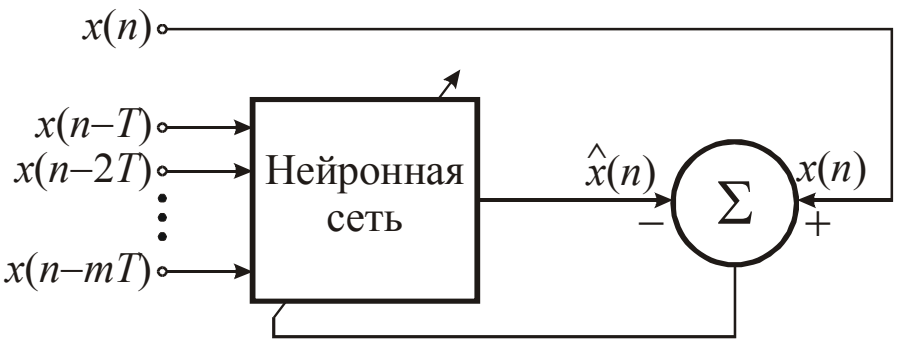
Пример фильтрации на основе *слепого разделения сигнала*.

Пусть $\{s_i(n)\}_{i=1}^{m}$ — неизвестный источник независимых сигналов. Эти сигналы линейно смешиваются. В результате вектор наблюдения имеет размерность $m \times 1$.

Блочная диаграмма слепого разделения источников



Блочная диаграмма нелинейного прогнозирования



$\mathbf{x}(n) = \mathbf{A}\mathbf{u}(n)$, $\mathbf{u}(n) = \left[u_1(n), u_2(n), ..., u_m(n)\right]^{\mathrm{T}}$, $\mathbf{x}(n) = \left[x_1(n), x_2(n), ..., x_m(n)\right]^{\mathrm{T}}$,

A — неизвестная несингулярная матрица смешивания размерности $m \times m$.

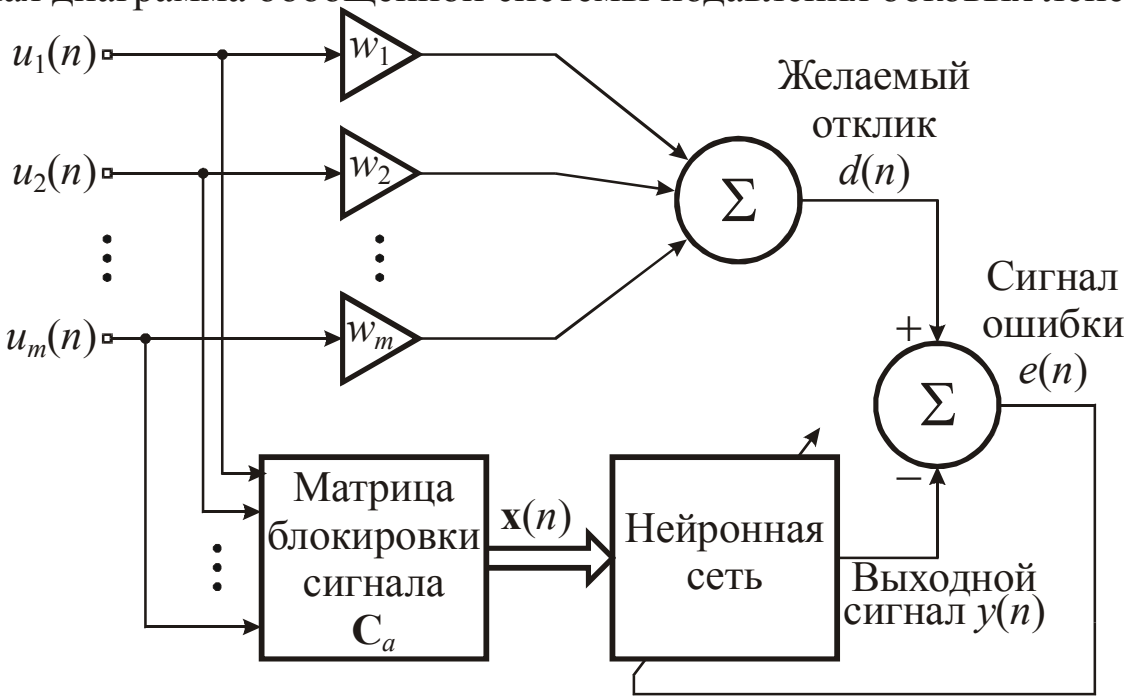
Вектор наблюдения $\mathbf{x}(n)$ известен. Требуется восстановить исходный сигнал $u_1(n), u_2(n), ..., u_m(n)$ методом обучения без учителя.

Можно воспользоваться задачей прогнозирования, которая может быть решена с помощью обучения на основе коррекции ошибок без учителя, так как обучающие примеры получаются из самого процесса. При этом $x(n)$ выступает в роли желаемого отклика. Обозначим $\hat{x}(n)$ результат прогнозирования на один шаг вперёд, сгенерированный нейронной сетью в момент времени n. Сигнал ошибки определяется как разность между $x(n)$ и $\hat{x}(n)$ и используется для настройки свободных параметров нейронной сети.

Формирование диаграммы направленности

Пространственная фильтрация (разделение пространственных признаков полезного сигнала и шума).
Отслеживание траектории цели в условиях помех и интерференции.
Сложности: полезный сигнал распространяется в неизвестном направлении; нет априорной информации о наложении сигналов при интерференции.

Блочная диаграмма обобщённой системы подавления боковых лепестков



Система состоит из следующих компонентов.

- Массив антенных элементов (снимет сигналы в дискретных точках пространства)
- Линейный сумматор (определён множеством фиксированных весов $\{w_i\}_{i=1}^{m}$, на выходе которого формируется желаемый отклик.
- Матрица блокировки сигнала $\mathbf{C}_a$ (блокировка сторонних сигналов, которые фиксируются боковыми лепестками диаграммы направленности).
- Нейронная сеть с настраиваемыми параметрами, предназначенная для адаптации к статистическим вариациям сторонних сигналов.

Настройка свободных параметров нейронной сети выполняется с помощью алгоритма обучения на основе коррекции ошибки (разность между выходом линейного сумматора $d(n)$ и фактическим выходным сигналом $y(n)$ нейронной сети. Линейный сумматор, находящийся вне контура обратной связи нейронной сети, выполняет роль учителя.

Память

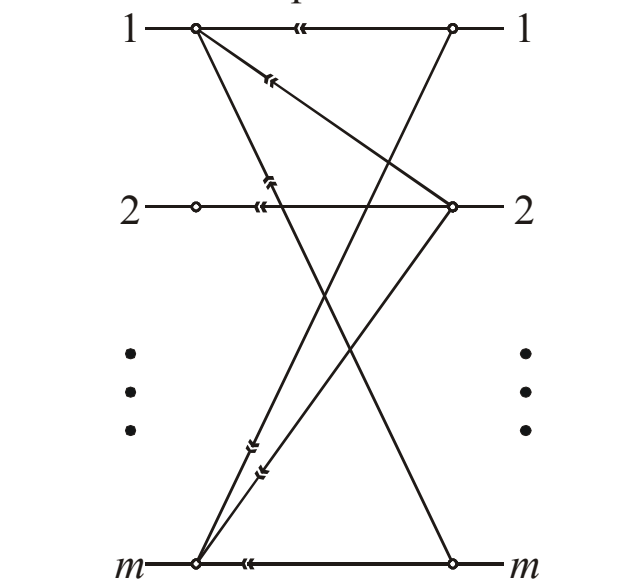
Особенности ассоциативной памяти

- Память является распределённой
- Стимулы (ключи) и отклики (хранимые образы) представляют собой векторы данных
- Информация запоминается с помощью формирования пространственных образов нейросетевой активности на большом количестве нейронов
- Информация, содержащаяся в стимул, определяет место её запоминания и адрес её извлечения
- Высокая устойчивость к помехам и искажению данных
- Между отдельными образами, хранимыми в памяти, могут существовать внутренние взаимосвязи. Есть некоторая вероятность ошибки при извлечении информации из памяти.

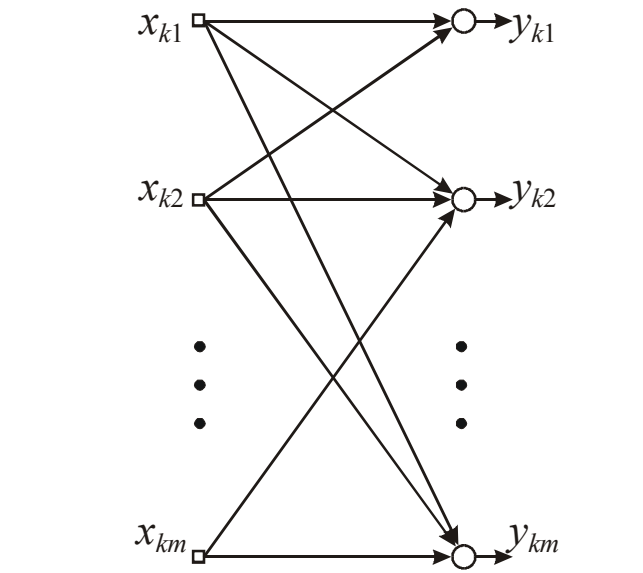
В распределённой памяти при обработке внутренних и внешних стимулов происходит почти параллельное функционирование множества различных нейронов. Память выполняет распределённое отображение образов в пространстве входных сигналов в другие образы выходного пространства. Свойства отображения распределённой памяти можно рассмотреть на примере идеализированной нейросети, состоящей из двух слоев нейронов, таких как сеть, которую можно рассматривать как *модельный компонент нервной системы*. В этой сети уровень активности нейрона входного слоя оказывает влияние на степень активности любого нейрона выходного слоя.

Другим примером, служит сеть, где узлы источника из входного слоя и нейроны выходного слоя работают как вычислительные элементы. Синаптические веса интегрированы в нейроны выходного слоя. Такие нейронные сети считаются линейными.

Компонент модели ассоциативной памяти нервной системы



Модель ассоциативной памяти с искусственными нейронами

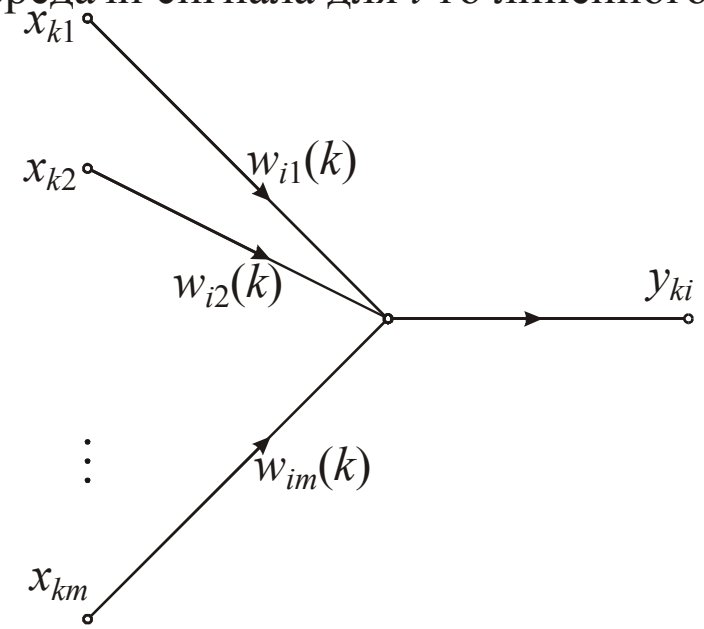


Все нейроны рассматриваются как линейные сумматоры, как показано на графе передачи сигнала. Пусть во входной слой передаётся образ $\mathbf{x}_k$, при этом в выходном слое возникает образ $\mathbf{y}_k$. Рассмотрим вопрос обучения на основе ассоциации между образами $\mathbf{x}_k$ и $\mathbf{y}_k$. Образы $\mathbf{x}_k$ и $\mathbf{y}_k$ представлены векторами

$$\mathbf{x}_k(n) = \left[x_{k1}(n), x_{k2}(n), ..., x_{km}(n)\right]^{\mathrm{T}},$$

$$\mathbf{y}_k(n) = \left[\mathbf{y}_{k1}(n), \mathbf{y}_{k2}(n), ..., \mathbf{y}_{km}(n)\right]^{\mathrm{T}}.$$

граф передачи сигнала для i-го линейного нейрона



Пусть размерности векторов равны m (*размерность сети*). Значение m равно количеству узлов источника во входном слое и числу вычислительных нейронов выходного слоя. Ассоциация между ключевым вектором $\mathbf{x}_k$ и запомненным вектором $\mathbf{y}_k$ можно представить в матричном виде $\mathbf{y}_k = \mathbf{W}(k)\mathbf{x}_k$, $k = 1, 2, ..., q$,

где $\mathbf{W}(k)$ — матрица весов, определяемая парами «вход-выход» $\left(\mathbf{x}_k, \mathbf{y}_k\right)$.

Выход нейрона y_{ki} вычисляется как взвешенная сумма элементов ключевого образа $\mathbf{x}_k$

$y_{ki} = \sum_{j=1}^{m} w_{ij}(k)\, x_{kj}$, $i = 1, 2, ..., m$, где $w_{ij}(k)$, $j = 1, 2, ..., m$ — синаптические веса нейрона i, соответствующие k-й паре ассоциированных образов. Подставляя элемент y_{ki}

$$y_{ki} = \left[w_{i1}(k), w_{i2}(k), ..., w_{im}(k)\right] \begin{bmatrix} x_{k1} \\ x_{k2} \\ ... \\ x_{km} \end{bmatrix}, i = 1, 2, ..., m.$$

в определение запоминаемого вектора $\mathbf{y}_k$ размерности $m \times 1$, имеем

$$\begin{bmatrix} y_{k1} \\ y_{k2} \\ \dots \\ y_{km} \end{bmatrix} = \begin{bmatrix} w_{11}(k) & w_{12}(k) & \dots & w_{1m}(k) \\ w_{21}(k) & w_{22}(k) & \dots & w_{2m}(k) \\ \dots & \dots & \dots & \dots \\ w_{m1}(k) & w_{m2}(k) & \dots & w_{mm}(k) \end{bmatrix} \begin{bmatrix} y_{k1} \\ y_{k2} \\ \dots \\ y_{km} \end{bmatrix}.$$

Это соотношение описывает матричное преобразование (отображение) в развёрнутом виде.

$$\mathbf{W}(k) = \begin{bmatrix} w_{11}(k) & w_{12}(k) & \dots & w_{1m}(k) \\ w_{21}(k) & w_{22}(k) & \dots & w_{2m}(k) \\ \dots & \dots & \dots & \dots \\ w_{m1}(k) & w_{m2}(k) & \dots & w_{mm}(k) \end{bmatrix}.$$

Отдельные представления q пар ассоциированных образов $\mathbf{x}_k \to \mathbf{y}_k$, $k = 1, 2, ..., q$ формируют значения элементов отдельных матриц $\mathbf{W}(1)$, $\mathbf{W}(2)$, …, $\mathbf{W}(q)$. Учитывая, что ассоциация образов представляется матрицей весов $\mathbf{W}(k)$, *матрицу памяти* размерности $m \times m$ можно определить как сумму матриц весовых коэффициентов всего набора ассоциаций

$$\mathbf{M} = \sum_{k=1}^{q} \mathbf{W}(k).$$

Матрица **M** определяет связи между входным и выходным слоями ассоциативной памяти. Она представляет *опыт*, накопленный в результате q образов, представленных в виде пар «вход-выход». То есть в матрице **M** содержатся данные обо всех парах «вход-выход», представленных для записи в память.

В рекурсивной форме описание матрицы памяти представлена как

$\mathbf{M}_k = \mathbf{M}_{k-1} + \mathbf{W}(k), k = 1, 2, ..., q$.

При добавлении к матрице $\mathbf{M}_{k-1}$ приращение $\mathbf{W}(k)$ теряет свою идентичность в сумме комбинаций, формирующих матрицу **M**. При увеличении числа q хранимых образов влияние каждого из них на состояние памяти ослабляется.

Память в виде матрицы корреляции

Пусть ассоциативная память обучена на парах входного и выходного сигналов $\mathbf{x}_k \to \mathbf{y}_k$, в результате чего вычислена матрица памяти $\mathbf{M}$. Оценка матрицы $\mathbf{M}$

$$\hat{\mathbf{M}} = \sum_{k=1}^{q} \mathbf{y}_k \mathbf{x}_k^{\mathrm{T}}$$

Здесь $\mathbf{y}_k \mathbf{x}_k^{\mathrm{T}}$ — внешнее матричное произведение ключевого $\mathbf{x}_k$ и запомненного $\mathbf{y}_k$ образов. Это произведение является оценкой матрицы весов $\mathbf{W}(k)$, которая отображает выходной вектор $\mathbf{y}_k$, на входной вектор $\mathbf{x}_k$.

Элемент внешнего произведения $\mathbf{y}_k \mathbf{x}_k^{\mathrm{T}}$ обозначается как $y_{ki} x_{kj}$, где x_{kj} — выходной сигнал узла j входного слоя, а y_{ki} — значение нейрона i выходного слоя. В контексте синаптического веса $w_{ij}(k)$ для k-й ассоциации узел источника j выступает в роли предсинаптического узла, а нейрон i в качестве постсинаптического узла. То есть рассматриваемый «локальный» процесс обучения рассматривается как *обобщение постулата обучения Хебба* (или *правилом внешнего матричного произведения*).

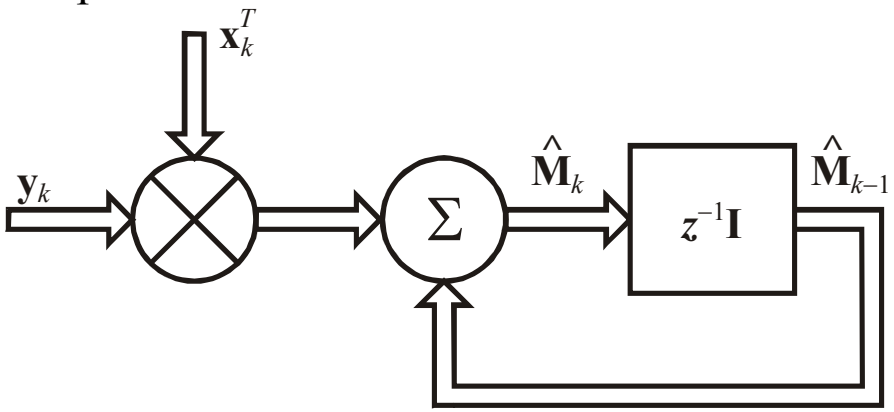
Построенная таким образом матрица памяти $\hat{\mathbf{M}}$ называется *памятью в виде матрицы корреляции*. Кстати, корреляция является основой процесса обучения, распознавания ассоциаций и образов, а также извлечения данных из памяти в нервной системе человека.

Оценку матрицы $\mathbf{M}$ можно записать в форме

$$\hat{\mathbf{M}} = \left[\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_q\right] \begin{bmatrix} \mathbf{x}_1^{\mathrm{T}} \\ \mathbf{x}_2^{\mathrm{T}} \\ \dots \\ \mathbf{x}_q^{\mathrm{T}} \end{bmatrix} = \mathbf{Y}\mathbf{X}^{\mathrm{T}}, \text{ где } \mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_q\right],\ \mathbf{Y} = \left[\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_q\right].$$

Матрица $\mathbf{X}$ (*матрица ключей*) имеет размерность $m \times q$ и состоит из множества ключевых образов, использованных в процессе обучения. Матрица $\mathbf{Y}$ (*матрица запоминания*) имеет размерность $q \times m$ и составлена из соответствующего множества запомненных образов.

В рекурсивной форме $\hat{\mathbf{M}}_k = \hat{\mathbf{M}}_{k-1} + \mathbf{y}_k \mathbf{x}_k^{\mathrm{T}}, k = 1, 2, ..., q$.
Граф передачи сигнала для этого выражения:



Матрица $\hat{\mathbf{M}}_{k-1}$ представляет собой существенную оценку матрицы памяти, а матрица $\hat{\mathbf{M}}_{k-1}$ — её обновлённую оценку в свете новой ассоциации между образами $\mathbf{x}_k$ и $\mathbf{y}_k$. Внешнее произведение $\mathbf{y}_k \mathbf{x}_k^{\mathrm{T}}$ представляет собой оценку матрицы весов $\mathbf{W}(k)$, соответствующую *k*-й ассоциации ключевого и запомненного образов $\mathbf{x}_k$ и $\mathbf{y}_k$.

Извлечение из памяти

Фундаментальные задачи при использовании ассоциативной памяти, являются её адресация и извлечение запомненных образов.

Пусть на вход системы ассоциативной памяти подаётся случайный вектор возбуждения $\mathbf{x}_j$, для которого требуется получить вектор *отклика* $\mathbf{y} = \hat{\mathbf{M}}\mathbf{x}_j$.

Так как $\mathbf{M} = \sum_{k=1}^{q} \mathbf{W}(k)$, то $\mathbf{y} = \sum_{k=1}^{q} \mathbf{y}_k \mathbf{x}_k^{\mathrm{T}} \mathbf{x}_j = \sum_{k=1}^{m} \left(\mathbf{x}_k^{\mathrm{T}} \mathbf{x}_j\right) \mathbf{y}_k = \left(\mathbf{x}_j^{\mathrm{T}} \mathbf{x}_j\right) \mathbf{y}_j + \sum_{k=1, k \neq j}^{q} \left(\mathbf{x}_k^{\mathrm{T}} \mathbf{x}_j\right) \mathbf{y}_k$,

где $\mathbf{x}_k^{\mathrm{T}} \mathbf{x}_j$ — скалярное произведение векторов $\mathbf{x}_k$ и $\mathbf{x}_j$.

Пусть ключевые образы нормированы

$$E_k = \sum_{l=1}^{q} x_{kl}^2 = \mathbf{x}_k^{\mathrm{T}} \mathbf{x}_k = 1, k = 1, 2, ..., q.$$

Тогда отклик памяти на возбудитель (ключевой образ) $\mathbf{x}_j$ можно упростить

$\mathbf{y} = \mathbf{y}_j + \mathbf{v}_j$, где $\mathbf{v}_j = \sum_{k=1, k \neq j}^{q} \left(\mathbf{x}_k^{\mathrm{T}} \mathbf{x}_j\right) \mathbf{y}_k$.

Первое слагаемое представляет собой ожидаемый отклик $\mathbf{y}_j$ (сигнальная составляющая фактического отклика **y**). Второе слагаемое $\mathbf{v}_j$ — шум, который возникает в результате смешивания ключевого вектора $\mathbf{x}_j$ со всеми остальными векторами, хранящимися в памяти. Этот вектор шума несёт ответственность за ошибки при извлечении из памяти.

Адаптация

В случае нестационарной среды требуется *непрерывное обучение*. Обучение с учителем не подходит.
Требуются *линейные адаптивные фильтры*, построенные для линейного сумматора.
Свойство *псевдостационарности* позволяет решать задачу переучивания, позволяющую учесть вариации входных данных.

1. Выбирается короткий интервал времени, на котором данные можно считать стационарными, и они используются для обучения сети
2. Отбрасывается старый вектор и добавить в выборку новый пример.
3. Для обучения сети используется новая выборка
4. Постоянное повторение процедур

В итоге реализуется принцип *непрерывного обучения на упорядоченных во времени примерах*. Реализация на компьютере требует высокого быстродействия

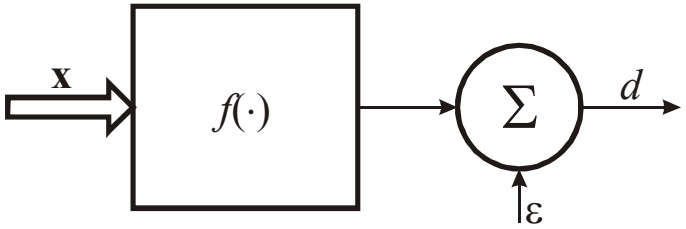
Статистическая природа процесса обучения

Рассмотрим пример стохастического явления, описываемого случайным вектором **X**, состоящим из набора *независимых примеров*, и случайный скаляр *D*, представляющий *зависимую переменную*. Пусть существуют *N* реализаций случайного вектора **X**, обозначенных $\{\mathbf{x}_i\}_{i=1}^{N}$, и соответствующее им множество реализаций случайного скаляра *D*, которое обозначим $\{d_i\}_{i=1}^{N}$. Эти реализации (измерения) в совокупности составляют обучающую выборку $\mathbf{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$. Обычно функциональная связь между **X** и *D* неизвестна.

Модель $D = f(\mathbf{X}) + \varepsilon$,

где $f(\cdot)$ — некоторая детерминированная функция векторного аргумента, ε — ожидаемая ошибка, представляющая неизвестную взаимосвязь между **X** и *D*, называется регрессионной. В этой модели вектор **X** используется для описания или предсказания зависимой переменной *D*.

Математическое представление нейронной сети



Ожидаемая ошибка ε в общем случае является случайной величиной с нормальным распределением и нулевым математическим ожиданием. Такая модель обладает следующими свойствами.

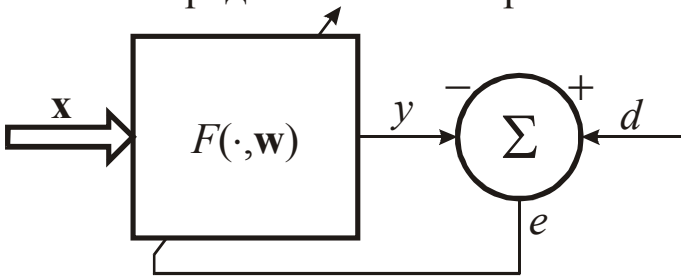
1. Среднее значение ожидаемой ошибки ε для любой реализации **x** равно нулю $E[\varepsilon|\mathbf{x}] = 0$,

E — статистический оператор математического ожидания. Следствие: регрессионная функция $f(\mathbf{x})$ является условным средним модели выхода *D* для входного сигнала $\mathbf{X} = \mathbf{x}$: $f(\mathbf{x}) = E[D|\mathbf{x}]$.

2. Ожидаемая ошибка ε не коррелирует с функцией регрессии $f(\mathbf{X})$, то есть $E[\varepsilon f(\mathbf{X})] = 0$. (Принцип ортогональности, вся информация о *D*, доступная через входной сигнал **X**, закодирована в функции регрессии $f(\mathbf{X})$).

Физическая модель данной среды

Физическое представление нейронной сети



основана на нейронной сети и позволяет закодировать эмпирические знания, заключённые в обучающей выборке **T**, с помощью соответствующего набора векторов синаптических весов **w**: $\mathbf{T} \to \mathbf{w}$.
Нейронная сеть обеспечивает аппроксимацию регрессионной модели.
Пусть фактический отклик нейронной сети на входной вектор **x** обозначается вероятностной переменной
$Y = F(\mathbf{X}, \mathbf{w})$,
где $F(\cdot, \mathbf{w})$ — функция отображения входных данных в выходные, реализуемая с помощью нейронной сети.
Для набора данных обучения **T**, представленного в виде множества, вектор синаптических весов **w** можно вычислить путём минимизации функции стоимости:

$$\mathbf{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(d_i - F\left(\mathbf{x}_i, \mathbf{w}\right)\right)^2$$

Она описывает сумму квадратов разностей между желаемым *d* и фактическим *y* откликами нейронной сети для всего набора примеров обучения **T**. Настройка синаптических весов нейронной сети выполняется для всего массива примеров обучения в целом, а не для каждого примера в отдельности.
Пусть E_{T} — оператор усреднения по всей обучающей выборке **T**. Тогда функция стоимости
$\mathbf{E}(\mathbf{w}) = \frac{1}{2}E_{\mathrm{T}}\left[\left(d_i - F\left(\mathbf{x}_i, \mathbf{T}\right)\right)^2\right]$, где функция $F\left(\mathbf{x}_i, \mathbf{T}\right)$ соответствует функции $F\left(\mathbf{x}_i, \mathbf{w}\right)$, то есть эти функции взаимозаменяемы.

Добавляя функцию $f(\mathbf{x})$ к аргументу $\left(d_i - F\left(\mathbf{x}_i, \mathbf{T}\right)\right)^2$ и вычитая её, получим

$$d - F(\mathbf{x}, \mathbf{T}) = \left(d - f(\mathbf{x})\right) + \left(f(\mathbf{x}) - F(\mathbf{x}, \mathbf{T})\right) = \varepsilon + \left(f(\mathbf{x}) - F(\mathbf{x}, \mathbf{T})\right)$$

В итоге функция стоимости

$$\mathbf{E}(\mathbf{w}) = \frac{1}{2} E_{\mathrm{T}}[\varepsilon^2] + \frac{1}{2} E_{\mathrm{T}}\left[\left(f(\mathbf{x}) - F(\mathbf{x},\mathbf{T})\right)^2\right] + E_{\mathrm{T}}\left[\varepsilon\left(f(\mathbf{x}) - F(\mathbf{x},\mathbf{T})\right)\right]$$

Здесь последнее слагаемое равно нулю так как ожидаемая ошибка ε не коррелирует с регрессионной функцией $f(\mathbf{x})$. Тогда

$$\mathbf{E}(\mathbf{w}) = \frac{1}{2} E_{\mathrm{T}}[\varepsilon^2] + \frac{1}{2} E_{\mathrm{T}}\left[\left(f(\mathbf{x}) - F(\mathbf{x},\mathbf{T})\right)^2\right]$$

Первое слагаемое описывает дисперсию ожидаемой ошибки ε, вычисленной на обучающей выборке **T**. Это *исходная* ошибка, независящая от весов **w** и её можно не учитывать, так как главной задачей является минимизация функции стоимости $\mathbf{E}(\mathbf{w})$ относительно вектора **w**.

Естественной мерой эффективности использования $F(\mathbf{x},\mathbf{w})$ для прогнозирования желаемого отклика *d* является функция $L_{\mathrm{av}}\left(f(\mathbf{x}), F(\mathbf{x},\mathbf{w})\right) = E_{\mathrm{T}}\left[\left(f(\mathbf{x}) - F(\mathbf{x},\mathbf{w})\right)^2\right]$.

Данный результат обеспечивает математическую основу для изучения зависимости между смещением и дисперсией, полученными в результате использования $F(\mathbf{x},\mathbf{w})$ в качестве аппроксимации функции $f(\mathbf{x})$.

Дилемма смещения и дисперсии

Среднее значение ошибки оценивания регрессионной функции $f(\mathbf{x}) = E[D|\mathbf{X} = \mathbf{x}]$ функцией аппроксимации $F(\mathbf{x}, \mathbf{w})$, вычисленной на всём обучающем множестве **T**

$$L_{\text{av}}\left(f(\mathbf{x}), F(\mathbf{x}, \mathbf{w})\right) = E_{\text{T}}\left[\left(E[D|\mathbf{X} = \mathbf{x}] - F(\mathbf{x}, \mathbf{w})\right)^2\right].$$

Условное среднее $E[D|\mathbf{X} = \mathbf{x}]$ имеет постоянное математическое ожидание на обучающем множестве **T**. Данное выражение можно представить в виде суммы двух слагаемых

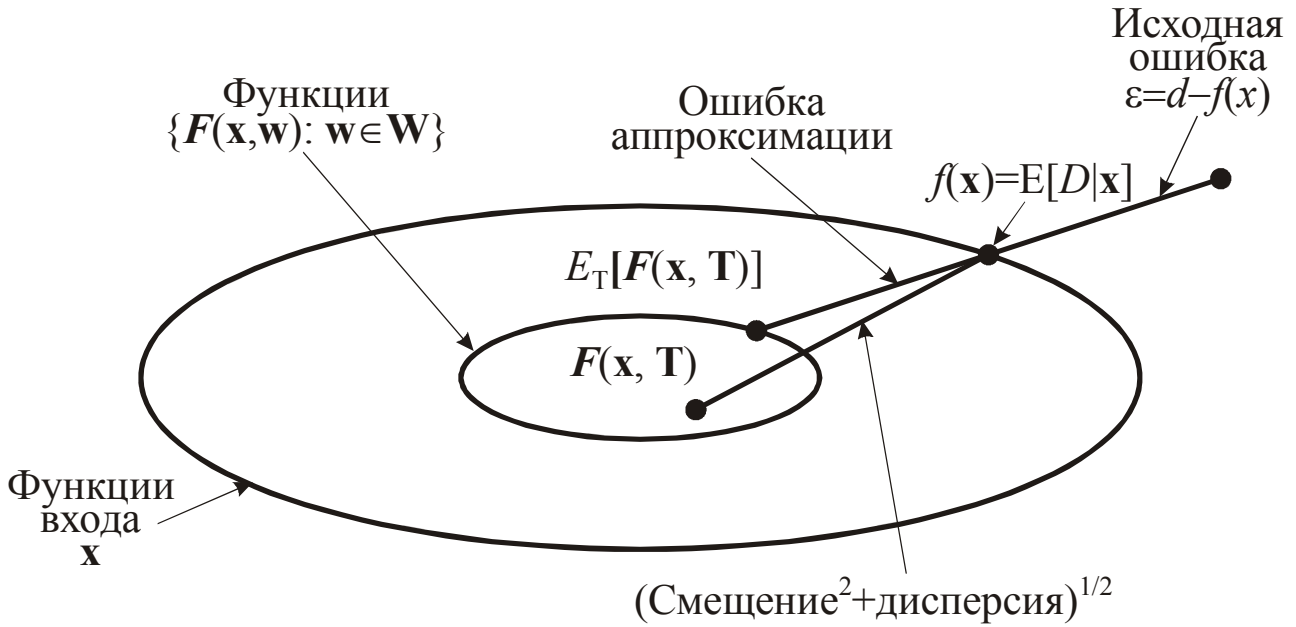
$$L_{\text{av}}\left(f(\mathbf{x}), F(\mathbf{x}, \mathbf{w})\right) = B^2(\mathbf{w}) + V(\mathbf{w}),$$

где $B(\mathbf{w}) = E_{\text{T}}\left[F(\mathbf{x}, \mathbf{T})\right] - E[D|\mathbf{X} = \mathbf{x}]$, $V(\mathbf{w}) = E_{\text{T}}\left[\left(F(\mathbf{x}, \mathbf{T}) - E_{\text{T}}\left[F(\mathbf{x}, \mathbf{T})\right]\right)^2\right]$.

1. Элемент $B(\mathbf{w})$ описывает *смещение* среднего значения аппроксимирующей функции $F(\mathbf{x}, \mathbf{T})$ относительно регрессионной функции $f(\mathbf{x}) = E[D|\mathbf{X} = \mathbf{x}]$. Этот элемент выражает неспособность нейронной сети, представленной функцией $F(\mathbf{x}, \mathbf{T})$, точно аппроксимировать регрессионную функцию $f(\mathbf{x}) = E[D|\mathbf{X} = \mathbf{x}]$, то есть элемент $B(\mathbf{w})$ можно считать *ошибкой аппроксимации*.
2. Элемент $V(\mathbf{w})$ представляет *дисперсию* аппроксимирующей функции $F(\mathbf{x}, \mathbf{w})$ на всём обучающем множестве **T**. Это слагаемое отражает неадекватность информации о регрессионной функции $f(\mathbf{x})$, содержащейся в обучающем множестве **T**. Таким образом, элемент $V(\mathbf{w})$ можно считать *ошибкой оценивания*.

На рисунке изображены взаимосвязи между целевой и аппроксимирующей функциями, а также как накапливаются ошибки оценивания — смещение и дисперсия.

Различные источники ошибки при решении задачи регрессии



Достижение приемлемой производительности возможно только при условии, что смещение $B(\mathbf{w})$ и дисперсия $V(\mathbf{w})$ функции аппроксимации $F(\mathbf{x},\mathbf{w}) = F(\mathbf{x},\mathbf{T})$ небольшие по значению.

Однако в нейронных сетях, обучаемых на данных выборки фиксированного размера, малое смещение достигается за счёт большой дисперсии. Одновременно уменьшить дисперсию и смещение можно, если размер обучающего множества бесконечно велик (*дилемма смещения/дисперсии*). Следствием этой проблемы является медленная сходимость процесса обучения.

Вариант обхода этой проблемы: преднамеренно ввести смещение, которое сводит дисперсию к нулю или существенно уменьшает. При этом требуется обоснование приемлемости встроенного в нейронную сеть смещения. Для достижения этой цели вводят ограниченную сетевую архитектуру. Ограничения могут принимать форму априорных знаний, встроенных в архитектуру нейронной сети путём *совместного использования весов* и/или создания *локальных рецепторных полей*, связанных с отдельными нейронами сети.

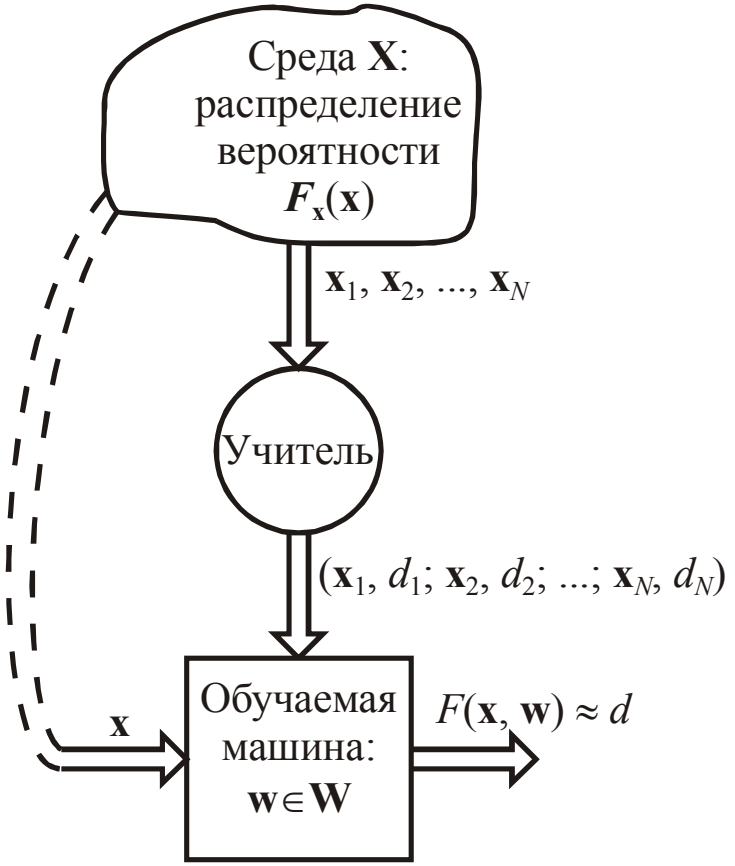
4 лекция
Теория статистического обучения

Решается задача управления обобщающей способностью нейронных сетей в контексте обучения с учителем

Модель обучения с учителем состоит из трёх взаимосвязанных компонентов.

1. *Среда* (Стационарная. Представлена векторами **x** с фиксированной, но неизвестной функцией распределения вероятностей $F_{\mathrm{X}}(\mathbf{x})$.
2. *Учитель* (Генерирует желательный отклик d для каждого из входных векторов **x**, полученных из внешней среды, в соответствии с условной функцией распределения $F_{\mathrm{X}}(\mathbf{x}|d)$, которая также фиксирована, но неизвестна). Связь желаемого отклика d и входного вектора **x**: $d = f(\mathbf{x}, v)$, где v — шум.
3. *Обучаемая машина*. (Способна реализовать множество функций отображения вход-выход), описываемых соотношением $y = F(\mathbf{x}, \mathbf{w})$, где y — фактический отклик, сгенерированный обучаемой машиной в ответ на входной сигнал **x**; **w** — набор свободных параметров, выбранных из пространства параметров **W**.

Модель процесса обучения с учителем



Задача обучения с учителем состоит в выборе конкретной функции $F(\mathbf{x}, \mathbf{w})$, которая оптимально аппроксимирует ожидаемый отклик d.

Выбор основывается на множестве N независимых, равномерно распределённых примеров обучения $\mathbf{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$. Каждая пара выбирается обучаемой машиной из множества **T** с некоторой обобщённой функцией распределения вероятности $F_{\mathrm{X,D}}(\mathbf{x}, d)$.

Рассмотрим задачу обучения с учителем как задачу аппроксимации, состоящую в нахождении функции $F(\mathbf{x}, \mathbf{w})$, которая наилучшим образом приближает желаемую функцию $f(\mathbf{x})$.

Пусть $L\left(d, F\left(\mathbf{x}, \mathbf{w}\right)\right)$ — мера потерь (несходства) между желаемым откликом d, соответствующим входному вектору $\mathbf{x}$, и откликом $F\left(\mathbf{x}, \mathbf{w}\right)$, сгенерированным обучающей машиной. В качестве меры $L\left(d, F\left(\mathbf{x}, \mathbf{w}\right)\right)$ рассматривают квадратичную функцию потерь, определённую как $L\left(d, F\left(\mathbf{x}, \mathbf{w}\right)\right) = \left(d - F\left(\mathbf{x}, \mathbf{w}\right)\right)^2$.
Квадратичное расстояние в формуле — усреднённое по множеству всех пар примеров $\left(\mathbf{x}, d\right)$ расширение меры $L\left(d, F\left(\mathbf{x}, \mathbf{w}\right)\right)$.
Ожидаемая величина потерь определяется функционалом риска

$$R(w) = \int L\left(d, F\left(\mathbf{x}, \mathbf{w}\right)\right) dF_{\mathrm{X,D}}\left(\mathbf{x}, d\right)$$

Интеграл берётся по всем возможным значениям $\left(\mathbf{x}, d\right)$. Целью обучения с учителем является минимизация функционала риска $R(\mathbf{w})$ в классе функций аппроксимации $\left\{F\left(\mathbf{x}, \mathbf{w}\right), \mathbf{w} \in \mathbf{W}\right\}$.
Оценка функционала риска усложняется тем, что обобщённая функция распределения $F_{\mathrm{X,D}}\left(\mathbf{x}, d\right)$ обычно неизвестна. При обучении вся доступная информация содержится в множестве данных обучения **T**. С помощью индуктивного принципа минимизации эмпирического риска можно обойти эту математическую сложность. Этот принцип основан на доступности обучающего множества **T**.

Некоторые основные определения

Сходимость по вероятности. Пусть имеется последовательность случайных переменных $a_1, a_2, ..., a_N$. Эта последовательность считается *сходящейся по вероятности* к случайной переменной a_0, если для любого $\sigma > 0$ выполняется следующее вероятностное отношение:

$$P\left(\left|a_N - a_0\right| > \sigma\right) \xrightarrow{P} \text{при } N \to \infty$$

Нижний и верхний пределы (supremum & infimum)**.** Верхним пределом непустого множества скалярных величин $\mathbf{A}$ ($\sup \mathbf{A}$) называется наименьший из скаляров *x*, для которых истинно неравенство $x \geqslant y$ для всех $y \in \mathbf{A}$. Если его не существует, то верхний предел ∞. Нижним пределом непустого множества скалярных величин $\mathbf{A}$ ($\inf \mathbf{A}$) называется наибольший из скаляров *x*, для которых истинно неравенство $x \leqslant y$ для всех $y \in \mathbf{A}$. Если его не существует, то верхний предел $-\infty$.

Функционал эмпирического риска. Для обучающего множества $\mathbf{T} = \left\{\left(\mathbf{x}_i, d_i\right)\right\}_{i=1}^{N}$ функционал эмпирического риска определяется в терминах функции потерь $L\left(d_i, F\left(\mathbf{x}_i, \mathbf{w}_i\right)\right)$:

$$R_{\text{emp}}\left(\mathbf{w}\right) = \frac{1}{N}\sum_{i=1}^{N} L\left(d_i, F\left(\mathbf{x}_i, \mathbf{w}\right)\right)$$

Строгая состоятельность (strict consistency). Пусть $\mathbf{W}\left(c\right)$ — непустое подмножество множества $\mathbf{W}$ функций $L\left(d_i, F\left(\mathbf{x}_i, \mathbf{w}\right)\right)$, распределение которых определяется интегральной функцией распределения $F_{X,D}\left(\mathbf{x}, d\right)$, такое, что $\mathbf{W}\left(c\right) = \left\{\mathbf{w} : \int L\left(d, F\left(\mathbf{x}, \mathbf{w}\right)\right) \geqslant c\right\}$, где $c \in \left(-\infty, +\infty\right)$. Функционал эмпирического риска считается *строго состоятельным*, если для любого подмножества $\mathbf{W}\left(c\right)$ обеспечивается сходимость по вероятности

$$\inf_{\mathbf{w} \in \mathbf{W}(c)} R_{\text{emp}}\left(\mathbf{w}\right) \xrightarrow{P} \inf_{\mathbf{w} \in \mathbf{W}(c)} R\left(\mathbf{w}\right) \text{при } N \to \infty$$

Принцип минимизации эмпирического риска

Этот принцип состоит в использовании функционала эмпирического риска $R_{\text{emp}}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L\left(d_i, F(\mathbf{x}_i, \mathbf{w})\right)$.

В отличии от $R(\mathbf{w}) = \int L\left(d, F(\mathbf{x}, \mathbf{w})\right) dF_{\mathbf{X},D}(\mathbf{x}, d)$ он явно не зависит от неизвестной функции распределения $F_{X,D}(\mathbf{x}, d)$ и теоретически его можно минимизировать по вектору весовых коэффициентов **w**.

Пусть $\mathbf{w}_{\text{emp}}$ и $F(\mathbf{x}, \mathbf{w}_{\text{emp}})$ — вектор весов и соответствующее ему отображение, которые минимизируют функционал эмпирического риска $R_{\text{emp}}(\mathbf{w})$. Пусть $\mathbf{w}_o$ и $F(\mathbf{x}, \mathbf{w}_o)$ — вектор весовых коэффициентов и отображение, минимизирующее фактический функционал риска $R(\mathbf{w})$. Векторы $\mathbf{w}_{\text{emp}}$ и $\mathbf{w}_o$ принадлежат пространству весов **W**. Требуется найти условия, при которых аппроксимирующее отображение $F(\mathbf{x}, \mathbf{w}_{\text{emp}})$ достаточно «близко» к фактическому отображению $F(\mathbf{x}, \mathbf{w}_o)$.

Для некоторого фиксированного $\mathbf{w} = \mathbf{w}^*$ функционал риска $R(\mathbf{w}^*)$ определяет *математическое ожидание* случайной переменной, определяемое соотношением

$$Z_{\mathbf{w}^*} = L\left(d, F(\mathbf{x}, \mathbf{w}^*)\right).$$

В отличие от него функционал эмпирического риска $R_{\text{emp}}(\mathbf{w}^*)$ обеспечивает *эмпирическое (арифметическое) среднее значение* случайной переменной Z_{w^*}. Согласно *закону больших чисел* для обучающего множества **T** бесконечно большого размера *N* эмпирическое среднее случайной переменной Z_w в общем случае сходится к её ожидаемому значению. При этом, требование, что вектор весовых коэффициентов $\mathbf{w}_{\text{emp}}$, минимизирующий функционал эмпирического риска $R_{\text{emp}}(\mathbf{w})$, будет также минимизировать и функционал риска $R_{\text{emp}}(\mathbf{w})$, можно приближённо удовлетворить, применив следующий подход.

Если функционал эмпирического риска $R_{\text{emp}}(\mathbf{w})$ аппроксимирует исходный функционал риска $R(\mathbf{w})$ равномерно по w с некоторой точностью ε, тот минимум $R_{\text{emp}}(\mathbf{w})$ отстоит от минимума $R(\mathbf{w})$ не более чем на величину 2ε. Это означает необходимость обязательного выполнения условия, что для любого $\mathrm{w} \in \mathrm{W}$ и $\varepsilon > 0$ должно выполняться вероятностное соотношение

$$P\left(\sup_{\mathbf{w}} \left| R(\mathbf{w}) - R_{\text{emp}}(\mathbf{w}) \right| > \varepsilon \right) \to 0 \text{ при } N \to \infty.$$

Если это условие выполняется, то можно утверждать, что вектор весов **w** среднего эмпирического риска равномерно сходится к своему ожидаемому значению. То есть для любой точности ε и некоторого положительного α выполняются неравенства

$$P\left(\sup_{\mathbf{w}} \left| R(\mathbf{w}) - R_{\text{emp}}(\mathbf{w}) \right| > \varepsilon \right) < \alpha,$$

$$P\left(R_{\text{emp}}(\mathbf{w}) - R(\mathbf{w}_o) > 2\varepsilon \right) < \alpha.$$

То есть, если выполняется 1 условие (из последних двух), то с вероятностью $1-\alpha$ решение $F(\mathbf{x}, \mathbf{W}_{\text{emp}})$, минимизирующее функционал эмпирического риска $R_{\text{emp}}(\mathbf{w})$, обеспечивает отличие фактического риска $R(\mathbf{w}_{\text{emp}})$ от минимально возможного фактического риска на величину не превышающую 2ε. То есть одновременно выполняются 2 неравенства

$$R(\mathbf{w}_{\text{emp}}) - R_{\text{emp}}(\mathbf{w}_{\text{emp}}) < \varepsilon,$$

$$R_{\text{emp}}(\mathbf{w}_o) - R(\mathbf{w}_o) < \varepsilon.$$

Эти два соотношения определяют различие между функционалами истинного и эмпирического рисков в точках $\mathbf{w} = \mathbf{w}_{\text{emp}}$ и $\mathbf{w} = \mathbf{w}_o$. Учитывая, что $\mathbf{w}_{\text{emp}}$ и $\mathbf{w}_o$ являются точками минимума функционалов $R_{\text{emp}}(\mathbf{w})$ и $R(\mathbf{w})$, можно сделать вывод, о том, что $R_{\text{emp}}(\mathbf{w}_{\text{emp}}) \leqslant R_{\text{emp}}(\mathbf{w}_o)$ или $R(\mathbf{w}_{\text{emp}}) - R(\mathbf{w}_o) < 2\varepsilon$ которое выполняется с той же вероятностью $1-\alpha$.

В итоге формулировка принципа минимизации эмпирического риска состоит из трёх взаимосвязанных частей.

1. Вместо функционала риска $R(\mathbf{w})$ строится функционал эмпирического риска

$$R_{\text{emp}}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L\left(d_i, F(\mathbf{x}_i, \mathbf{w})\right)$$

на базе множества примеров обучения $(\mathbf{x}_i, d_i)$, $i = 1, 2, ..., N$.

2. Пусть $\mathbf{w}_{\text{emp}}$ — вектор весовых коэффициентов, минимизирующий функционал эмпирического риска $R_{\text{emp}}(\mathbf{w})$ в пространстве весов **W**. При этом при увеличении количества *N* примеров обучения до бесконечности функционал эмпирического риска $R_{\text{emp}}(\mathbf{w})$ равномерно сходится к функционалу фактического риска $R(\mathbf{w})$.

3. Равномерная сходимость, определяемая как

$$P\left(\sup_{\mathbf{w}} \left|R(\mathbf{w}) - R_{\text{emp}}(\mathbf{w})\right| > \varepsilon\right) \to 0 \text{ при } N \to \infty$$

является необходимым и достаточным условием непротеворечивости принципа минимизации эмпирического риска.

VC-измерение

Теория равномерной сходимости функционала эмпирического риска $R_{\text{emp}}(\mathbf{w})$ к функционалу фактического риска $R(\mathbf{w})$ включает ограничения на скорость сходимости, которые основаны на параметре, называемом *измерение Вапника-Червоненкиса* (VC-измерение). Это измерение является мерой ёмкости или вычислительной мощности семейства функций классификации, реализованных обучаемыми машинами.

Для описания концепции VC-измерения рассматривается задача двоичной классификации образов (множество ожидаемых откликов состоит из двух значений $d \in \{0, 1\}$.

Для обозначения правила принятия решения или функции двоичной классификации используется термин *дихотомия*.

Пусть **F** — множество дихотомий, реализованных обучаемой машиной, то есть

$$\mathbf{F} = \left\{ F(\mathbf{x}, \mathbf{w}) : \mathbf{w} \in \mathbf{W},\ F : \Re^m \mathbf{W} \to \{0, 1\} \right\}.$$

Пусть **L** — множество, содержащее *N* точек *m*-мерного пространства **X** входных векторов:

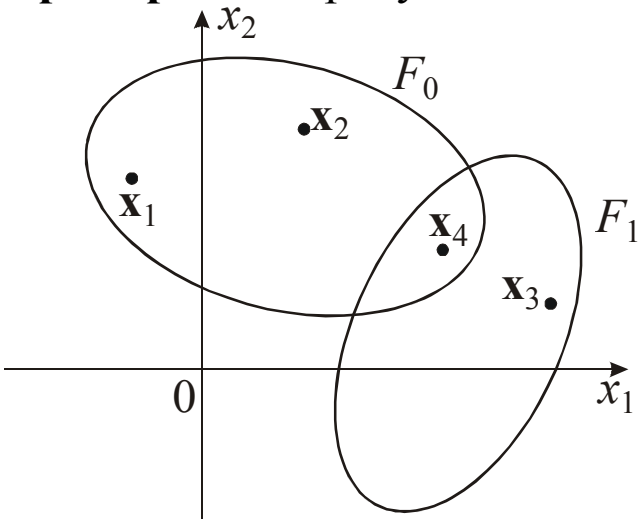
$$\mathbf{L} = \left\{ \mathbf{x}_i \in \mathbf{X};\ i = 1, 2, ..., N \right\}.$$

Дихотомия, реализованная обучаемой машиной, разбивает множество **L** на два непересекающихся подмножества $\mathbf{L}_0$ и $\mathbf{L}_1$, таких, что

$$F(\mathbf{x}, \mathbf{w}) = \begin{cases} 0 \text{ для } \mathbf{x} \in \mathbf{L}_0, \\ 1 \text{ для } \mathbf{x} \in \mathbf{L}_1. \end{cases}$$

Пусть $\Delta_F(\mathbf{L})$ — количество различных дихотомий, реализованных обучаемой машиной; $\Delta_F(l)$ — максимум $\Delta_F(\mathbf{L})$ на множестве всех **L**, для которых $|\mathbf{L}| = l$, где $|\mathbf{L}|$ — количество элементов в **L**. Говорят, что ансамбль дихотомий **F** является *разбиением* множества **L**, если $\Delta_F(\mathbf{L}) = 2^{|L|}$, то есть если все возможные дихотомии в **L** могут быть реализованы функциями **F**. При этом $\Delta_F(l)$ называется *функцией роста*.

Пример 2.1 На рисунке



показано двумерное входное пространство $\mathbf{X}$, состоящее из четырёх точек $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, $\mathbf{x}_4$. Показанные на рисунке границы решений функций F_0 и F_1 соответствует классам (гипотезам) 0 и 1. По рисунку видно, что функция F_0 индуцирует дихотомию $\mathbf{D}_0 = \left\{\mathbf{G}_0 = \left\{x_1, x_2, x_4\right\}, \mathbf{G}_1 = \left\{x_3\right\}\right\}$.

С другой стороны функция F_1 описывает дихотомию $\mathbf{D}_1 = \left\{\mathbf{G}_0 = \left\{x_1, x_2\right\}, \mathbf{G}_1 = \left\{x_3, x_4\right\}\right\}$.

Так как множество $\mathbf{G}$ состоит из четырёх точек, мощность $\left|\mathbf{G}\right| = 4$. Следовательно, $\Delta_F\left(\mathbf{G}\right) = 2^4 = 16$.

VC-*измерением* $\mathbf{F}$ *называется мощность наибольшего множества* $\mathbf{L}$, *разбиением которого является* $\mathbf{F}$.

VC-*измерением* $\mathbf{F}$ (обозначается $\mathrm{VC}\dim\left(\mathbf{F}\right)$) является самое большое значение N, для которого $\Delta_F\left(N\right) = 2^N$.

VC-измерение множества функций классификации $\left\{F\left(\mathbf{x}, \mathbf{w}\right) : \mathbf{w} \in \mathbf{W}\right\}$ — это максимальное число образов, на которых машина может быть обучена без ошибок для всех возможных бинарных маркировок функций классификации.

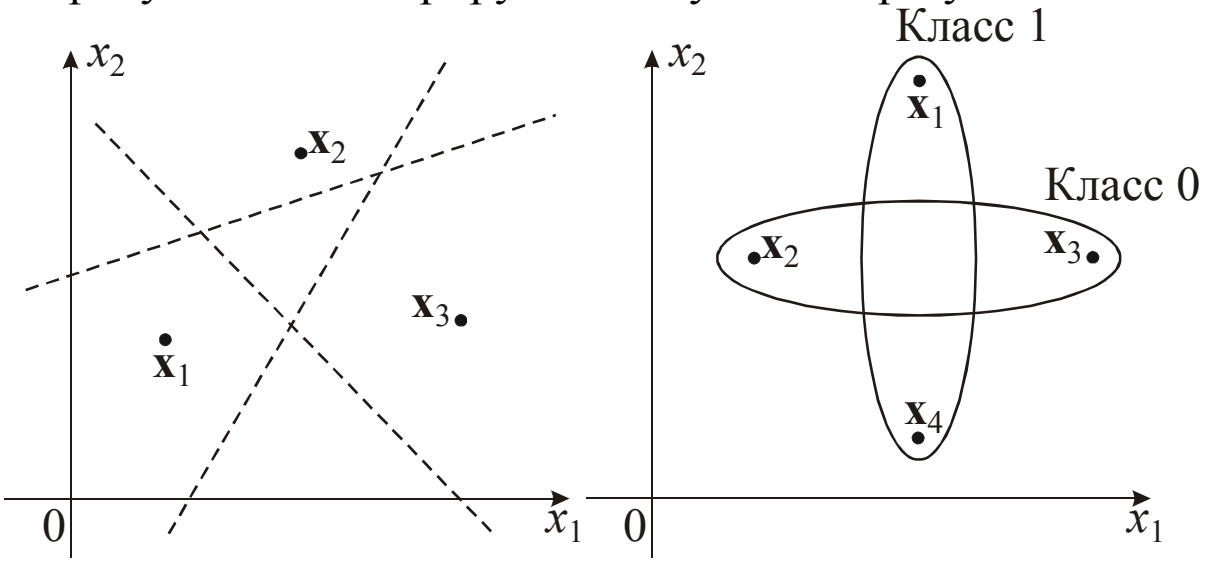
Пример 2.2 Рассматривается простое решающее правило в m-мерном пространстве $\mathbf{X}$ входных векторов:
$\mathbf{F}: y = \varphi\left(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b\right)$,

где $\mathbf{x}$ — m-мерный вектор весов; b — порог. Функция активации φ является пороговой, то есть

$$\varphi(\upsilon) = \begin{cases} 1, \upsilon \geqslant 0, \\ 0, \upsilon < 0. \end{cases}$$

VC-измерение приведённого решающего правила определяется соотношением
$\mathrm{VC}\dim(\mathbf{F}) = m + 1$.

На рисунках иллюстрируется полученный результат.



Рассматривается двумерное входное пространство (то есть $m = 2$). На левом рисунке показаны три точки: $\mathbf{x}_1$, $\mathbf{x}_2$ и $\mathbf{x}_3$, а также три возможных варианта разделения этих точек. Эти точки могут быть разделены тремя линиями. На рисунке справа изображены четыре точки: $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ и $\mathbf{x}_4$. Точки $\mathbf{x}_2$ и $\mathbf{x}_3$ относятся к классу 0, а точки $\mathbf{x}_1$ и $\mathbf{x}_4$ — к классу 1. Точки $\mathbf{x}_2$ и $\mathbf{x}_3$ нельзя отделить от точек $\mathbf{x}_1$ и $\mathbf{x}_4$ одной линией. Таким образом, VC-измерение решающего правила $\mathbf{F}: y = \varphi\left(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b\right)$ при $m = 2$ равно 3, что соответствует

$\mathrm{VC}\dim(\mathbf{F}) = m + 1$.

Пример 2.3

Поскольку VC-измерение является мерой ёмкости множества функций классификации (индикаторов), то можно ожидать, что обучаемая машина с большим числом свободных параметров будет иметь большое VC-измерение, и наоборот.

Контрпример, опровергающий данное утверждение.

Рассматривается однопараметрическое семейство функций-индикаторов

$f(x,a) = \operatorname{sgn}(\sin(ax)), a \in \Re$.

Пусть задано некоторое число N, для которого нужно найти N точек, для которых необходимо построить разбиение. Этому требованию удовлетворяет набор функций $f(x,a)$, где

$x_i = 10^{-i}, i = 1, 2, ..., N$.

Чтобы разделить эти точки данных на два класса, определяемых последовательностью

$d_1, d_2, ..., d_N, d_i \in \{-1, 1\}$,

достаточно выбрать параметр a согласно формуле

$$a = \pi\left(1 + \sum_{i=1}^{N} \frac{\left(1 - d_i 10^i\right)}{2}\right).$$

Отсюда можно заключить, что VC-измерение семейства функций-индикаторов $f(x,a)$ с единственным свободным параметром a равно бесконечности.

Важность VC-измерения и его оценка

VC-измерение является сугубо *комбинаторным понятием* и никак не связано с геометрическим понятием измерения.

С конструкторской точки зрения количество примеров, необходимых для обучения системы данным некоторого класса, строго пропорционально VC-измерению этого класса.

В некоторых случаях VC-измерение определяется свободными параметрами нейронной сети. На практике VC-измерение сложно получить аналитическими методами, тем не менее границы VC-измерения устанавливаются достаточно несложно.

Два примера для пояснения определения VC-измерения.

1. Пусть **N** — произвольная нейронная сеть прямого распространения, состоящая из нейронов с функцией активации Хевисайда

$$\varphi(\upsilon)=\begin{cases}1, \upsilon \geqslant 0,\\ 0, \upsilon < 0.\end{cases}$$

VC-измерение сети **N** составляет $O(W\log W)$, где W — общее количество свободных параметров сети.

2. Пусть **N** — произвольная многослойная нейронная сеть прямого распространения, состоящая из нейронов с сигмоидальной функцией активации

$$\varphi(\upsilon)=\frac{1}{1+\exp(-\upsilon)}$$

VC-измерение сети **N** составляет $O(W^2)$, где W — общее количество свободных параметров сети.

Конструктивные, независимые от распределения пределы обобщающей способности обучаемых машин

В задачах двоичной классификации образов, в которых ожидаемый отклик определяется множеством $d = \{0, 1\}$, функция потерь может принимать одно из двух значений

$$L\left(d, F\left(\mathbf{x}, w\right)\right) = \begin{cases} 0, \textit{если } F\left(\mathbf{x}, \mathbf{w}\right) = d, \\ 1, \textit{если } F\left(\mathbf{x}, \mathbf{w}\right) \neq d. \end{cases}$$

При этих условиях функционалы риска $R\left(\mathbf{w}\right)$ и эмпирического риска $R_{\text{emp}}\left(\mathbf{w}\right)$ могут иметь следующую интерпретацию.

- Функционал риска $R\left(\mathbf{w}\right)$ — это вероятность ошибки классификации, обозначаемой $P\left(\mathbf{w}\right)$.
- Функционал эмпирического риска $R_{\text{emp}}\left(\mathbf{w}\right)$ — это ошибка обучения, обозначаемая $\upsilon\left(\mathbf{w}\right)$.

Согласно закону больших чисел, эмпирическая частота возникновения каких-либо событий сходится к фактической вероятности этих же событий при количестве попыток, стремящихся к бесконечности при предположении, что эти попытки независимы и одинаково распределены. То есть для любого вектора **w**, не зависящего от обучающего множества, и для любой точности $\varepsilon > 0$ выполняется условие

$$P\left(\left|P\left(\mathbf{w}\right) - \upsilon\left(\mathbf{w}\right)\right| > \varepsilon\right) \to 0 \text{ при } N \to \infty,$$

где N — размер множества обучения. Выполнение этого условия не означает, что минимизация ошибки обучения $\upsilon\left(\mathbf{w}\right)$ при использовании некоторого правила классификации влечёт минимизацию вероятности ошибки классификации $P\left(\mathbf{w}\right)$. Для существенно большого размера N обучающего множества близость между $\upsilon\left(\mathbf{w}\right)$ и $P\left(\mathbf{w}\right)$ следует из более строгого условия

$$P\left(\sup_{\mathbf{w}}\left|P\left(\mathbf{w}\right) - \upsilon\left(\mathbf{w}\right)\right| > \varepsilon\right) \to 0 \text{ при } N \to \infty$$

В данном случае речь идёт о равномерной сходимости частоты ошибок обучения к вероятности того, что $\upsilon\left(\mathbf{w}\right) = P\left(\mathbf{w}\right)$.

Понятие VC-измерения накладывает ограничения на скорость равномерной сходимости. В частности, для множества функций классификации с VC-измерением равным h, выполняется неравенство

$$P\left(\sup_{\mathbf{w}}\left|P(\mathbf{w})-\upsilon(\mathbf{w})\right|>\varepsilon\right)<\left(\frac{2eN}{h}\right)^{h}\exp\left(-\varepsilon^{2}N\right),$$

где N — размер обучающего множества; e — основание натурального логарифма. Для того чтобы достичь равномерной сходимости, требуется обеспечить малое значение правой части неравенства для больших значений N.

Множитель $\left(\frac{2eN}{h}\right)^{h}$ представляет собой предел роста функции $\Delta_{F}(l)$ для семейства функций $\mathbf{F}=\left\{F(\mathbf{x},\mathbf{w});\mathbf{w}\in\mathbf{W}\right\}$ при $l\geqslant h\geqslant 1$. Ограничив быстрый рост этой функции можно обеспечить сходимость правой части неравенства к нулю при $N\to\infty$. Требование будет удовлетворено, если VC-измерение h не является бесконечно большим. То есть, конечность VC-измерения является необходимым и достаточным условием равномерной сходимости принципа минимизации эмпирического риска.

Если входное пространство **X** обладает конечной мощностью, то семейство дихотомий **F** будет иметь конечное VC-измерение по **X** (обратное утверждение не всегда верно).

Пусть α — вероятность события $\sup_{\mathbf{w}}\left|P(\mathbf{w})-\upsilon(\mathbf{w})\right|\geqslant\varepsilon$.

Тогда с вероятностью $1-\alpha$ можно утверждать, что все векторы весовых коэффициентов $\mathbf{w}\in\mathbf{W}$ удовлетворяют следующему неравенству:

$$P(\mathbf{w})<\upsilon(\mathbf{w})+\varepsilon.$$

Используя неравенство $P\left(\sup_{\mathbf{w}}\left|P(\mathbf{w})-\upsilon(\mathbf{w})\right|>\varepsilon\right)<\left(\frac{2eN}{h}\right)^{h}\exp\left(-\varepsilon^{2}N\right)$ и определение вероятности α, можно записать:

$$\alpha=\left(\frac{2eN}{h}\right)^{h}\exp\left(-\varepsilon^{2}N\right).$$

Пусть $\varepsilon_0(N, h, \alpha)$ — некоторое значение ε, удовлетворяющее соотношению для α.
Тогда можно получить результат:

$$\varepsilon_0(N, h, \alpha) = \sqrt{\frac{h}{N}\left(\log\left(\frac{2N}{h}\right)+1\right)-\frac{1}{N}\log\alpha}.$$

Величина $\varepsilon_0(N, h, \alpha)$ называется доверительным интервалом. Его значение зависит от размера обучающей выборки N, VC-измерения h и вероятности α.

Предел, описываемый выражением $P\left(\sup_{\mathbf{w}}\left|P(\mathbf{w})-\upsilon(\mathbf{w})\right|>\varepsilon\right)<\left(\frac{2eN}{h}\right)^h \exp\left(-\varepsilon^2 N\right)$ при $\varepsilon=\varepsilon_0(N, h, \alpha)$ достигается в худшем случае с вероятностью $P(\mathbf{w})=0{,}5$. Для малых значений $P(\mathbf{w})$ более полезное ограничение получается после модификации неравенства:

$$P\left(\sup_{\mathbf{w}}\frac{\left|P(\mathbf{w})-\upsilon(\mathbf{w})\right|}{\sqrt{p(\mathbf{w})}}>\varepsilon\right)<\left(\frac{2eN}{h}\right)^h \exp\left(\frac{-\varepsilon^2 N}{4}\right).$$

Из этого неравенства следует, что с вероятностью $1-\alpha$ одновременно для всех $\mathbf{w}\in\mathbf{W}$ выцполняется соотношение

$P(\mathbf{w})\leqslant\upsilon(\mathbf{w})+\varepsilon_1(N, h, \alpha, \upsilon)$,

где $\varepsilon_1(N, h, \alpha, \upsilon)$ — новый доверительный интервал, определяемый в терминах ранее рассмотренного доверительного интервала $\varepsilon_0(N, h, \alpha)$ следующим образом:

$$\varepsilon_1(N, h, \alpha, \upsilon)=2\varepsilon_0^2(N, h, \alpha)\left(1+\sqrt{1+\frac{\upsilon(\mathbf{w})}{\varepsilon_0^2(N, h, \alpha)}}\right).$$

Этот доверительный интервал зависит от ошибки обучения $\upsilon(\mathbf{w})$. При $\upsilon(\mathbf{w})=0$

$\varepsilon_1(N, h, \alpha, \upsilon)=4\varepsilon_0^2(N, h, \alpha)$.

В итоге определяются два ограничения на скорость равномерной сходимости.
В общем случае скорость равномерной сходимости удовлетворяет ограничению

$$P(\mathbf{w}) \leqslant \upsilon(\mathbf{w}) + \varepsilon_1(N, h, \alpha, \upsilon),$$

где $\varepsilon_1(N, h, \alpha, \upsilon) = 2\varepsilon_0^2(N, h, \alpha)\left(1 + \sqrt{1 + \frac{\upsilon(\mathbf{w})}{\varepsilon_0^2(N, h, \alpha)}}\right)$.

При малых значениях ошибки обучения $\left(\upsilon(\mathbf{w}) \to 0\right)$

$$P(\mathbf{w}) \leqslant \upsilon(\mathbf{w}) + 4\varepsilon_0^2(N, h, \alpha).$$

При больших значениях ошибки обучения $\upsilon(\mathbf{w})$

$$P(\mathbf{w}) < \upsilon(\mathbf{w}) + \varepsilon_0(N, h, \alpha).$$

Минимизация структурного риска

Под *ошибкой обучения* $\upsilon_{\text{train}}(\mathbf{w})$ понимается частота сделанных машиной ошибок в течение сеанса обучения для определённого вектора весов **w**.

Под *ошибкой общения* $\upsilon_{\text{gene}}(\mathbf{w})$ понимается частота сделанных машиной ошибок приеё тестировании на не встречавшихся ранее примерах (предполагается, что тестовые данные принадлежат тому же семейству, что и данные обучения).

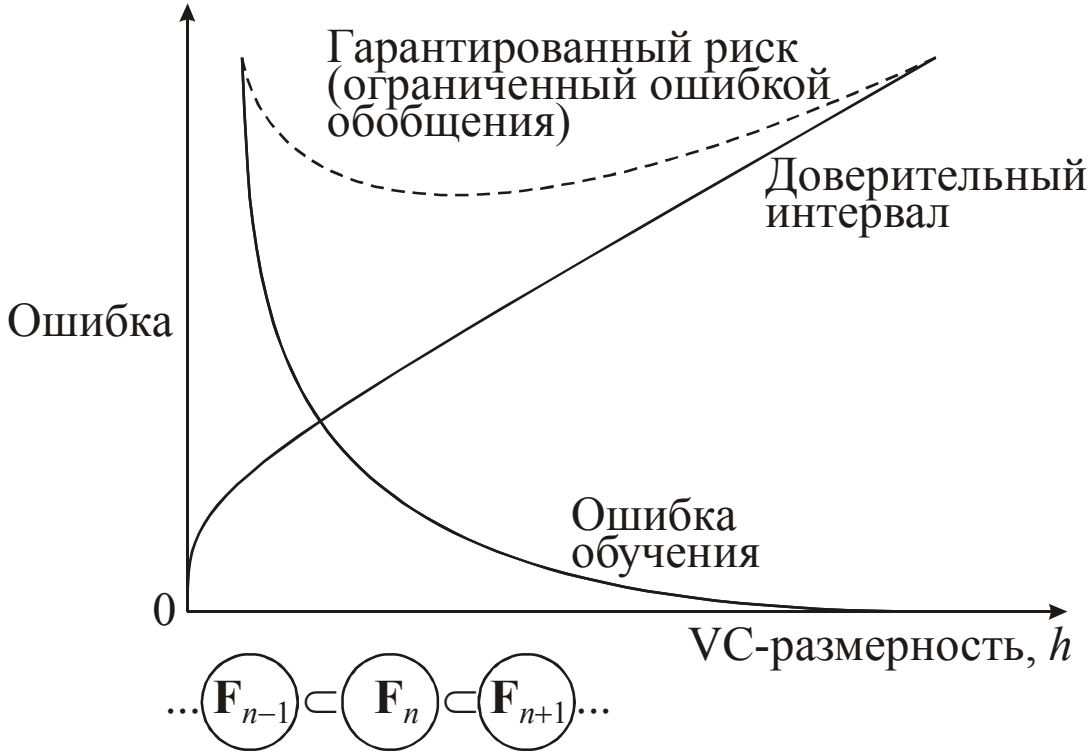
Обозначим символом h VC-измерение семейства функций классификации $\{F(\mathbf{x},\mathbf{w});\mathbf{w}\in\mathbf{W}\}$ по отношению для пространства входных сигналов **X**. Тогда согласно теории скорости равномерной сходимостиможно утверждать, сто с вероятностью $1-\alpha$ для количества примеров обучения $N>h$, одновременнодля всех функций классификации $F(\mathbf{x},\mathbf{w})$ ошибка обобщения $\upsilon_{\text{gene}}(\mathbf{w})$ имеет меньшее значение, чем *гарантированный риск*, определяемый как сумма пары конкурирующих величин

$$\upsilon_{\text{guarant}}(\mathbf{w})=\upsilon_{\text{train}}(\mathbf{w})+\varepsilon_1(N,h,\alpha,\upsilon_{\text{train}}),$$

где доверительный интервал $\varepsilon_1(N,h,\alpha,\upsilon_{\text{train}})=2\varepsilon_0^2(N,h,\alpha)\left(1+\sqrt{1+\dfrac{\upsilon(\mathbf{w})}{\varepsilon_0^2(N,h,\alpha)}}\right)$.

Для фиксированного числа N примеров обучения ошибка обучения монотонно уменьшается при увеличении VC-измерения h, а доверительный интервал монотонно увеличивается. Следовательно, как гарантированный риск, так и ошибка обобщения имеют точку минимума.

Взаимосвязь между ошибкой обучения, доверительным интервалом и гарантированным риском



До момента достижения точки минимума задача является переопределённой в том смысле, что ёмкость машины h слишком мала, чтобы вместить весь объём деталей обучения. После прохождения точки минимума задача обучения является недоопределённой, то есть ёмкость машины слишком велика для такого объёма данных обучения.

Таким образом, при решении задачи обучения с учителем необходимо обеспечить максимальную эффективность обобщения за счёт приведения в соответствие ёмкости машины с доступным количеством данных обучения.

Метод минимизации структурного риска обеспечивает индуктивную процедуру достижения этой цели, в которой VC-измерение обучаемой машины рассматривается как управляющая переменная. Для большей конкретизации рассмотрим ансамбль классификаторов образов $\left\{F\left(\mathbf{x},\mathbf{w}\right);\mathbf{w}\in\mathbf{W}\right\}$ и определим вложенную структуру, состоящую из n подобных машин:

$\mathbf{F}_k=\left\{F\left(\mathbf{x},\mathbf{w}\right);\mathbf{w}\in\mathbf{W}\right\}, k=1,2,...,n,$

таких, что

$\mathbf{F}_1\subset\mathbf{F}_1\subset...\subset\mathbf{F}_n,$

где символ $\subset$ означает «содержится в». Соответственно VC-измерения отдельных классификаторов образов удовлетворяют следующему условию:

$h_1\leqslant h_2\leqslant...\leqslant h_n.$

• Это говорит о том, что VC-измерение каждого из классификаторов конечно. Тогда метод минимизации структурного риска излагается следующим образом.

• Минимизируется эмпирический риск (ошибка обучения) для каждого из классификаторов

Определяется классификатор $\mathbf{F}^*$, который имеет наименьший гарантированный риск. Эта конкретная машина обеспечивает наилучший компромисс между ошибкой обучения (качеством аппроксимации данных обучения) и доверительным интервалом (сложностью функции аппроксимации), которые конкурируют друг с другом.

Цель. Поиск такой нейросетевой структуры, в которой уменьшение VC-измерения достигается за счёт минимально возможного увеличения ошибки обучения.

Принцип минимизации структурного риска может быть реализован множеством различных способов. Например, VC-измерение h можно изменять за счёт изменения количества скрытых нейронов. В качестве примера рассмотрим ансамбль полносвязных многослойных сетей прямого распространения, в которых количество нейронов в одном из скрытых слоёв монотонно возрастает. В соответствии с принципом минимизации структурного риска наилучшей сетью в этом множестве будет та, для которой гарантированный риск будет минимальным.

Вероятностно-корректная в смысле аппроксимации модель обучения (probably-approximately correct — PAC)

Эта модель представляет собой вероятностный «каркас» (или среду) для изучения процессов обучения и обобщения в системах двоичной классификации. Она тесно связана с принципом обучения с учителем.

Терминология.

Множество элементов X называется *понятием*, а любой набор его подмножества — *классом понятий*.

Примером понятия называется любой объект из предметной области, вместе с меткой своего класса. Если пример относится к данному понятию, он называется *положительным примером*. Иначе — *отрицательным примером*.

Понятие, для которого приводятся примеры, называется *целевым*.

Последовательность данных обучения длины N для понятия c можно определить как

$$\mathbf{T} = \left\{\left(\mathbf{x}_i, c\left(\mathbf{x}_i\right)\right)\right\}_{i=1}^{N}.$$

В этой последовательности могут содержаться и повторяющиеся примеры. Примеры $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ выбираются из среды $\mathbf{X}$ случайным образом, в соответствии с некоторым фиксированным, но неизвестным распределением вероятности.

- Целевое понятие $c\left(\mathbf{x}_i\right)$ рассматривается как функция, отображающая $\mathbf{X}$ в множество $\left\{0, 1\right\}$. При этом предполагается, что функция $c\left(\mathbf{x}_i\right)$ неизвестна.
- Предполагается, что все примеры статистически независимы. Это значит, что функция плотности совместной вероятности двух различных примеров $\mathbf{x}_i$ и $\mathbf{x}_j$ равна произведению соответствующих функций плотности вероятности.

Среда **X** соответствует пространству входных сигналов нейронной сети, а целевое понятие — ожидаемому отклику сети.

Набор понятий, порождаемых средой **X**, называется *пространством понятий* **B**. Каждое из этих понятий может быть закодировано различными способами при формировании множеств положительных и отрицательных примеров.

При обучении с учителем обучаемая машина представляет собой множество функций, каждая из которых соответствует определённому состоянию.

Множество всех функций (понятий), определяемых обучаемой машиной, называется *пространством гипотез* **G**. Это пространство может совпадать или не совпадать с пространством понятий **B**.

Пространства понятий и гипотез являются аналогами функции $f\left(x\right)$ и аппроксимирующей функции $F\left(\mathbf{x},\mathbf{w}\right)$.

Пусть существует некоторое целевое понятие $c(\mathbf{x}) \in \mathbf{B}$, принимающее значения 0 и 1. Требуется обучить этому понятию нейронную сеть при помощи её настройки на множестве данных **T**, определённым выражением $\mathbf{T} = \left\{\left(\mathbf{x}_i, c(\mathbf{x}_i)\right)\right\}_{i=1}^{N}$.

Пусть $g(\mathbf{x}) \in \mathbf{G}$ — гипотеза, соответствующая отображению. Входа на выход, сформированному в результате проведённого обучения. Одним из способов достижения успеха в обучении является измерение степени близости гипотезы $g(\mathbf{x})$ к целевой концепции $c(\mathbf{x})$. Всегда существуют ошибки, обеспечивающие различие этих величин. Эти ошибки являются следствием того, что мы пытаемся обучить нейронную сеть некоторой функции на основе ограниченной информации о ней. Вероятность ошибки обучения определяется выражением

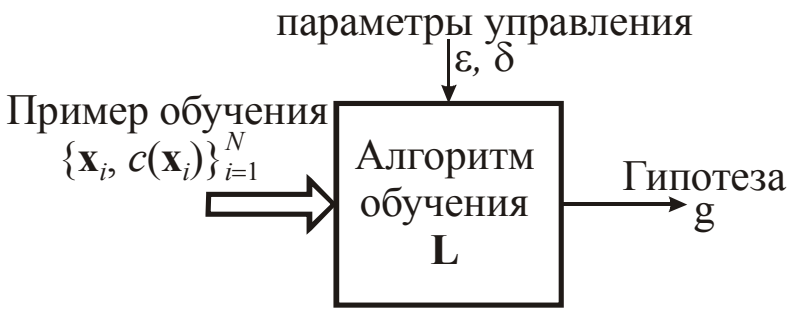
$\upsilon_{\text{train}} = P\left(\mathbf{x} \in \mathbf{X} : g(\mathbf{x}) \neq c(\mathbf{x})\right)$.

Распределение вероятности в этом примере должно быть таким же, как и при формировании примеров. Целью обучения PAC является минимизация значения υ_{train}.

Предметная область, доступная алгоритму обучения, определяется размером *N* обучающего множества **T**. Кроме того, алгоритм обучения имеет два следующих параметра управления.

- *Параметр ошибки* $\varepsilon \in (0, 1]$. Этот параметр задаёт величину ошибки, при которой аппроксимация целевого понятия $c(\mathbf{x})$ гипотезой $g(\mathbf{x})$ считается удовлетворительной.
- *Параметр доверия* $\delta \in (0, 1]$. Этот параметр задаёт степень правдоподобия при построении «хорошей» аппроксимации.

Блочная диаграмма, иллюстрирующая модель обучения PAC



Пусть **B** — класс понятий для среды **X**. Считается, что класс **B** является PAC-обучаемым, если существует алгоритм **L**, обладающий следующим свойством. Для любого целевого понятия $c \in \mathbf{B}$, для любого распределения вероятности на **X** и для всех $0 < \varepsilon < 0{,}5$ и $0 < \delta < 0{,}5$ при использовании алгоритма **L** для множества примеров обучения $\mathbf{T} = \left\{\left(\mathbf{x}_i, c\left(\mathbf{x}_i\right)\right)\right\}_{i=1}^{N}$ с вероятностью не хуже $1 - \delta$ результатом алгоритма обучения **L** будет гипотеза *g* с ошибкой обучения $\upsilon_{\text{train}} < \varepsilon$. Эта вероятность получается на любом случайном подмножестве множества **T** и при любой внутренней рандомизации, которая может существовать в алгоритме обучения **L**. При этом размер обучающего множества *N* должен превышать значение некоторой функции от δ и ε.

То есть, если размер *N* обучающего множества **T** достаточно велик, то существует вероятность, что в результате обучения сети на этом наборе примеров отображения входа на выход, реализуемое сетью, будет «приблизительно корректным».

Не смотря на зависимость от δ и ε, количество примеров *N* не обязательно зависит от целевого понятия *c* и распределения вероятности в **X**.

Сложность обучающего множества

Вопрос *сложности обучающего множества*: сколько случайных примеров нужно предоставить алгоритму обучения, чтобы обеспечить его информацией, достаточной для «изучения» неизвестного понятия *c*, выбранного из класса понятий **B**, или насколько большим должно быть обучающее множество **T**?

Понятие согласованности: Пусть $\mathbf{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$ — некоторое множество маркированных примеров, в котором все $\mathbf{x}_i \in \mathbf{X}$ и все $d_i \in (0,1)$. Тогда понятие *c* называется согласованным с набором примеров **T** (и наоборот), если для любого $1 \leqslant i \leqslant N$ выполняется равенство $c(\mathbf{x}_i) = d_i$.

В концепции PAC-обучения является не размер множества вычисляемых нейронной сетью функций отображения входа на выход, а VC-измерение сети.

1. Любой состоятельный алгоритм обучения для нейронной сети с конечным VC-измерением $h \geqslant 1$ является алгоритмом обучения PAC.
2. Существует такая константа *K*, что достаточным размером обучающего множества **T** для любого алгоритма является

$$N = \frac{K}{\varepsilon}\left(h\log\left(\frac{1}{\varepsilon}\right) + \log\left(\frac{1}{\delta}\right)\right)$$

где ε — параметр ошибки; δ — параметр доверия.

Общность этого результата можно применить к обучению с учителем, независимо от типа алгоритма обучения и распределения вероятности маркированных примеров.

Вычислительная сложность

Понятие *вычислительной сложности* связано с пессимистической оценкой времени, необходимого для обучения нейронной сети на множестве маркированных примеров мощности N.

Алгоритм считается вычислительно *эффективным*, если время его работы пропорционально $O\left(m^r\right)$, где $r \geqslant 1$, то есть полиномиально зависит от m, а сам алгоритм называется *алгоритмом с полиномиальным временем выполнения*.

Если речь идёт о сложности обучающего множества, то рассматривается также параметр ошибки ε, который является фиксированным, но произвольным. При оценке вычислительной сложности алгоритма обучения необходимо знать, как она зависит от этого параметра. Со временем должно быть достигнуто некоторое состояние, которое обеспечит вероятностно-корректный в смысле аппроксимации выход. Для обеспечение эффективности вычислений соответствующее состояние должно достигаться за полиномиальное время по $1/\varepsilon$.

Итоговое формальное утверждение: *Алгоритм обучения является вычислительно эффективным по параметру ошибки ε, размерности примеров обучения m и размеру обучающего множества N, если время его выполнения является полиномиальным по N, и существует такое значение* $N_0\left(\delta, \varepsilon\right)$*, достаточное для PAC-обучения, при котором алгоритм является полиномиальным по m и* ε^{-1}.

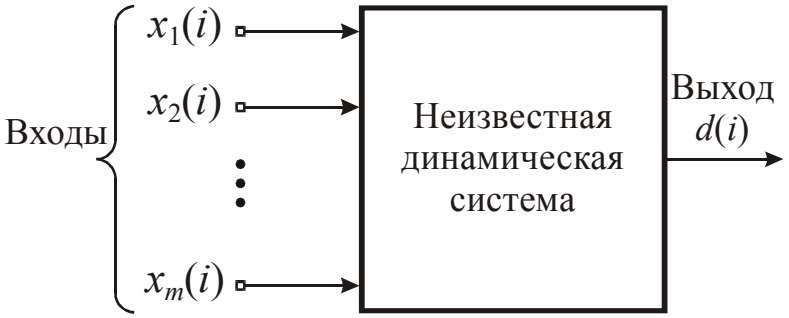
5 лекция
Однослойный персептрон

В 1958 году введено понятие персептрона как первой модели обучения с учителем.
Персептрон представляет собой простейшую форму нейронной сети, предназначенную для классификации линейно-разделимых сигналов (образы, которые можно разделить некоторой *гиперплоскостью*).
Персестрон состоит из одного нейрона с настраиваемыми синаптическими весами и порогом.
Если образы (векторы), используемые для обучения персептрона, выбраны из двух линейноразделимых классов, то алгоритм персептрона сходится и формирует поверхность решений в форме гиперплоскости, разделяющей эти два класса.
Персептрон, построенный на одном нейроне, ограничен выполнением задачи разделения только двух классов (гипотез). Увеличивая размерность выходного (вычислительного) слоя персептрона и включая в него несколько нейронов, можно решать задачи классификации на большее число классовю
Единичный нейрон также составляет основу адаптивного фильтра. При адаптивной фильтрации используется алгоритм минимизации среднеквадратической ошибки (LMS — least-mean-square algorithm), известный также под названием *дельта-правила*.

Задача адаптивной фильтрации

Рассматривается динамическая система, математические характеристики, которой неизвестны. Входные и выходные сигналы генерируются системой в равномерные дискретные моменты времени. В частности, если *m* входных узлов генерируют *m*-мерное входное воздействие $\mathbf{x}(i)$, в ответ система формирует скалярный выходной сигнал $d(i)$, где $i = 1, 2, ..., n$.

Неизвестная динамическая система



Внешнее поведение системы описывается множеством данных:

$\mathbf{T}: \{\mathbf{x}(i), d(i); i = 1, 2, ..., n, ...\}$,

где $\mathbf{x}(i) = [x_1(i), x_2(i), ..., x_m(i)]^{\mathrm{T}}$.

Множество **T** содержит примеры, одинаково распределённые согласно некоторому вероятностному закону. Размерность входного вектора $\mathbf{x}(i)$ называют *размерностью входного пространства*.

Представление сигнала $\mathbf{x}(i)$.

• Если m элементов сигнала $\mathbf{x}(i)$ зарождаются в различных точках пространства, то входной сигнал можно считать *моментальным снимком данных*.

• Если m элементов сигнала $\mathbf{x}(i)$ представляют собой текущее и $m-1$ предыдущие значения возбуждения, снятые через равномерные промежутки времени, означает, что входной сигнал снимается во временной области.

Для строящейся нейронной модели, работающей под управлением некоторого алгоритма, обеспечивающего настройку синаптических весов, принимаются следующие соглашения.

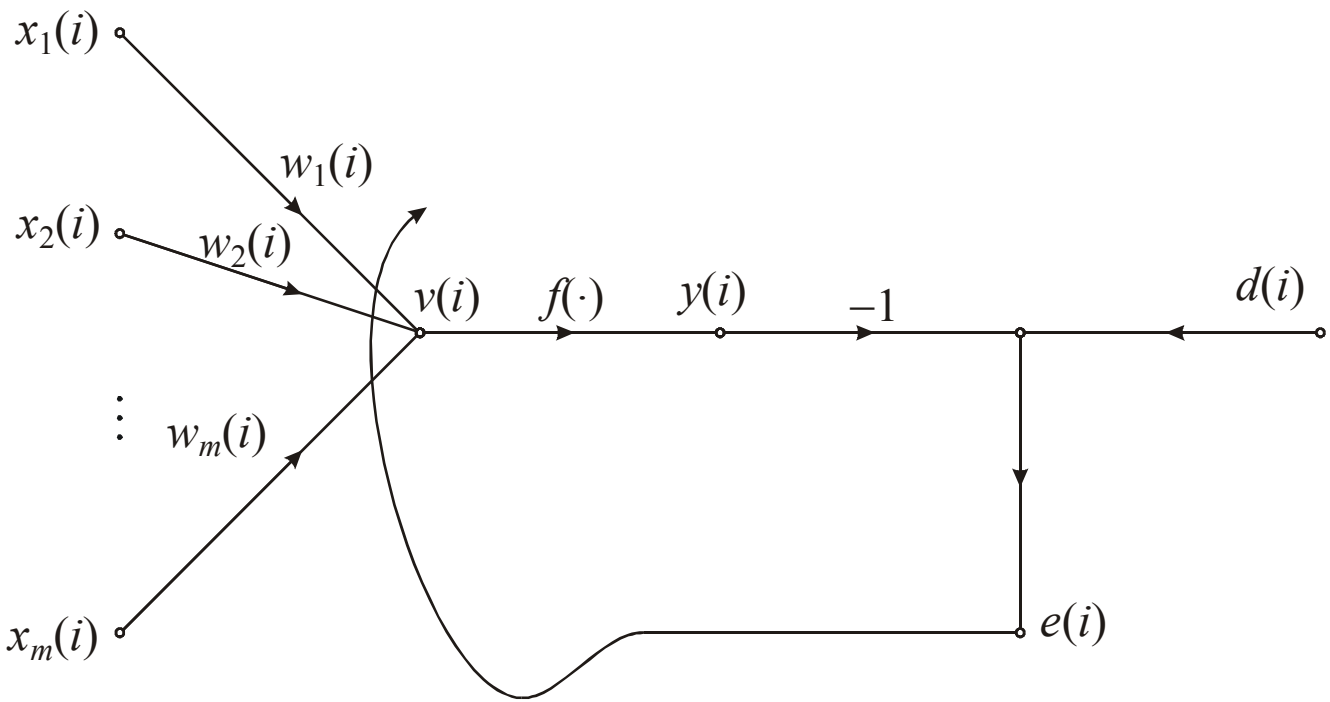
• Алгоритм начинает работу с произвольных значений синаптических весов

• Настройка синаптических весов в ответ на статические вариации поведения системы выполняется непрерывно

• Вычисление корректирующих значений синаптических весов выполняется через равномерные промежутки времени.

Модель, описываемая таким образом, называется *адаптивным фильтром*.

Граф прохождения сигнала в адаптивной модели системы



Работа адаптивного фильтра включает в себя два последовательных процесса.

1. Процесс фильтрации, предполагающий вычисление двух сигналов: выход-ного сигнала $y(i)$, генерируемого в ответ на вектор входного воздействия $\mathbf{x}(i) = x_1(i), x_2(i), ..., x_m(i)$ и сигнал ошибки $e(i)$ — отклонения выходного сигнала $y(i)$ от выходного сигнала реальной системы $d(i)$.

2. Процесс адаптации, включающий автоматическую подстройку синаптических весов нейрона на основе сигнала ошибки $e(i)$.

Комбинация этих двух процессов — *контур с обратной связью* нейрона.

Выходной сигнал $y(i)$ совпадает с индуцированным локальным полем $v(i)$:

$y(i) = v(i) = \sum_{k=1}^{m} w_k(i) x_k(i)$, где $w_1(i), w_2(i), ..., w_m(i)$ — m синаптических весов нейрона, измеренных в момент времени i.

В матричной форме $y(i) = \mathbf{x}^{\mathrm{T}}(i)\mathbf{w}(i)$, где $\mathbf{w}(i) = \left[w_1(i), w_2(i), ..., w_m(i) \right]^{\mathrm{T}}$.

Алгоритм применения сигнала ошибки $e(i) = y(i) - d(i)$ для коррекции синаптических весов нейрона является функцией стоимости, используемой конкретным методом адаптивной фильтрации, что тесно связано с задачей оптимизации.

Методы безусловной оптимизации

Рассматривается непрерывно дифференцируемая функция стоимости $\mathbf{E}(\mathbf{w})$, зависящую от некоторого неизвестного вектора **w**. Эта функция отображает элементы вектора **w** в пространство действительных чисел и является мерой оптимальности выбранного для алгоритма адаптивной фильтрации вектора **w**.

Требуется отыскать такое решение $\mathbf{w}^*$, что $\mathbf{E}(\mathbf{w}^*) \leqslant \mathbf{E}(\mathbf{w})$, то есть необходимо решить задачу безусловной оптимизации.

Формулировка задачи.

Минимизировать функцию стоимости $\mathbf{E}(\mathbf{w})$ по отношению к вектору весов **w**

$\mathbf{E}(\mathbf{w}) \to \min$.

Необходимым условием оптимальности является:

$\nabla \mathbf{E}(\mathbf{w}^*) = 0$,

где ∇ — оператор градиента $\nabla = \left[\frac{\partial}{\partial w_1}, \frac{\partial}{\partial w_2}, ..., \frac{\partial}{\partial w_m} \right]^{\mathrm{T}}$,

$\nabla \mathbf{E}(\mathbf{w})$ — вектор градиента функции стоимости

$\nabla \mathbf{E}(\mathbf{w}) = \left[\frac{\partial \mathbf{E}}{\partial w_1}, \frac{\partial \mathbf{E}}{\partial w_2}, ..., \frac{\partial \mathbf{E}}{\partial w_m} \right]^{\mathrm{T}}$.

Для создания адаптивных фильтров используют алгоритмы последовательного спуска.

Начиная с исходного значения $\mathbf{w}(0)$ генерируется последовательность векторов весовых весовых коэффициентов $\mathbf{w}(1), \mathbf{w}(2), ...$, таких, что с каждой итерацией алгоритма значение функции стоимости уменьшается: $\mathbf{E}(\mathbf{w}(n+1)) < \mathbf{E}(\mathbf{w}(n))$, где $\mathbf{w}(n)$ — предыдущее значение вектора весов; $\mathbf{w}(n+1)$ — последующее.

Но есть вероятность, что алгоритм станет неустойчивым (будет расходиться). Далее рассматриваются три метода безусловной оптимизации.

Метод наискорейшего спуска

Корректировка вектора весов выполняется в направлении максимального уменьшения функции стоимости, то есть в направлении, противоположном вектору градиента $\mathbf{g} = \nabla \mathbf{E}(\mathbf{w})$.

Формальная запись алгоритма:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta \mathbf{g}(n),$$

где η — положительная константа — *параметр скорости обучения*; $\mathbf{g}(n)$ — вектор градиента, вычисленный в точке $\mathbf{w}(n)$.

Коррекция весовых коэффициентов:

$$\Delta \mathbf{w}(n) = \mathbf{w}(n+1) - \mathbf{w}(n) = -\eta \mathbf{g}(n).$$

Далее показывается, что алгоритм наискорейшего спуска удовлетворяет условию $\mathbf{E}(\mathbf{w}(n+1)) < \mathbf{E}(\mathbf{w}(n))$.

Выражение $\mathbf{E}(\mathbf{w}(n+1))$ раскладывается в ряд Тейлора относительно $\mathbf{w}(n)$ с точностью до членов первого порядка

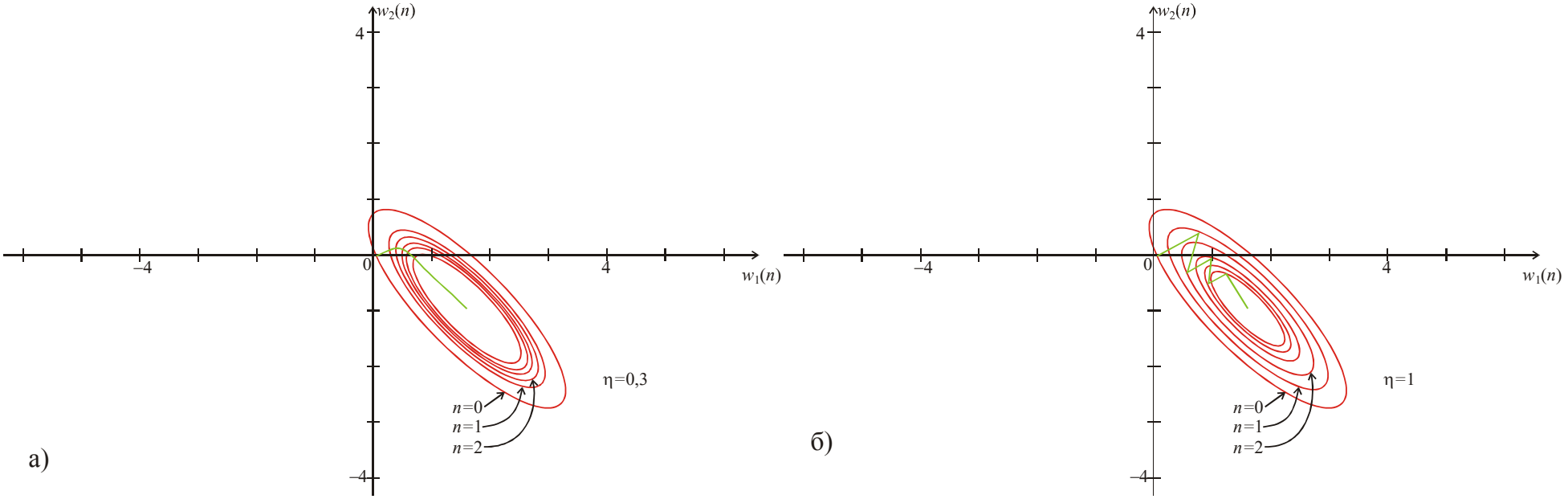
$$\mathbf{E}(\mathbf{w}(n+1)) = \mathbf{E}(\mathbf{w}(n)) + \mathbf{g}^{\mathrm{T}}(n) \Delta \mathbf{w}(n),$$

которое допустимо при малых значениях параметра η. Подставляя в эту формулу выражение для коррекции весовых коэффициентов $\Delta \mathbf{w}(n) = -\eta \mathbf{g}(n)$, получим:

$$\mathbf{E}(\mathbf{w}(n+1)) = \mathbf{E}(\mathbf{w}(n)) - \eta \mathbf{g}^{\mathrm{T}}(n) \mathbf{g}(n) = \mathbf{E}(\mathbf{w}(n)) + \eta \left\| \mathbf{g}(n) \right\|^2.$$

Для малых значений параметра скорости обучения η значение функции стоимости уменьшается на каждой итерации. Метод наискорейшего спуска сходится к оптимальному значению $\mathbf{w}^*$ достаточно медленно. На скорость сходимости также влияет параметр η.

Траектория изменения вектора весовых коэффициентов по методу наискорейшего спуска в двумерном пространстве для двух различных значений параметра скорости обучения $\eta = 0{,}3$ (а) и $\eta = 1{,}0$ (б).
Координаты w_1 и w_2 являются элементами вектора весов **w**.



Если параметр η мал, алгоритм замедляется (траектория изменения — гладкая кривая)
Если параметр η велик, алгоритм ускоряется (траектория изменения имеет зигзагообразный характер)
Если параметр η превосходит некоторое критическое значение, то алгоритм становится неустойчивым.

Метод Ньютона

Минимизируется квадратичная аппроксимация функции стоимости $\mathbf{E}(\mathbf{w})$ в точке $\mathbf{w}(n)$. Минимизация выполняется на каждом шаге алгоритма.

Функция стоимости раскладывается в ряд Тейлора относительно $\mathbf{w}(n)$ с точностью до членов второго порядка

$$\Delta\mathbf{E}(\mathbf{w}(n)) = \mathbf{E}(\mathbf{w}(n+1)) - \mathbf{E}(\mathbf{w}(n)) = \mathbf{g}^{\mathrm{T}}(n)\Delta\mathbf{w}(n) + \frac{1}{2}\Delta\mathbf{w}^{\mathrm{T}}(n)\mathbf{H}(n)\Delta\mathbf{w}(n). \qquad (*)$$

Здесь $\mathbf{g}(n)$ — вектор градиента функции $\mathbf{E}(\mathbf{w})$ размерности $m \times 1$, вычисленный в точке $\mathbf{w}(n)$. Матрица $\mathbf{H}(n)$ — *матрица Гессе*, или Гессиан, также вычисленный в точке $\mathbf{w}(n)$ по следующей формуле:

$$\mathbf{H} = \nabla^2\mathbf{E}(\mathbf{w}) = \begin{bmatrix} \frac{\partial^2\mathbf{E}}{\partial w_1^2} & \frac{\partial^2\mathbf{E}}{\partial w_1 \partial w_2} & \cdots & \frac{\partial^2\mathbf{E}}{\partial w_1 \partial w_m} \\ \frac{\partial^2\mathbf{E}}{\partial w_2 \partial w_1} & \frac{\partial^2\mathbf{E}}{\partial w_2^2} & \cdots & \frac{\partial^2\mathbf{E}}{\partial w_2 \partial w_m} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2\mathbf{E}}{\partial w_m \partial w_1} & \frac{\partial^2\mathbf{E}}{\partial w_m \partial w_2} & \cdots & \frac{\partial^2\mathbf{E}}{\partial w_m^2} \end{bmatrix}.$$

Для существования Гессиана функция стоимости должна быть дважды непрерывно дифференцируемой по элементам вектора $\mathbf{w}$.

Дифференцируя (*) по $\Delta\mathbf{w}$, получим, что инкремент $\nabla\mathbf{E}(\mathbf{w})$ достигает минимума при условии

$$\mathbf{g}(n) + \mathbf{H}(n)\Delta\mathbf{w}(n) = 0.$$

Разрешая это уравнение относительно $\Delta\mathbf{w}(n)$, получим $\Delta\mathbf{w}(n) = -\mathbf{H}^{-1}(n)\mathbf{g}(n)$.

Таким образом, $\mathbf{w}(n+1) = \mathbf{w}(n) + \Delta\mathbf{w}(n) = \mathbf{w}(n) - \mathbf{H}^{-1}(n)\mathbf{g}(n)$, где $\mathbf{H}^{-1}(n)$ — матрица, обратная Гессиану.

Метод Ньютона быстро асимптотически сходится и не приводит к появлению зигзагообразных траекторий, как метод наискорейшего спуска. Однако для обеспечения работоспособности метода Ньютона матрица Гессе $\mathbf{H}(n)$ должна быть *положительно определённой* для всех n. Для всех итераций алгоритма это не гарантируется. Если Гессиан не является положительно определённой матрицей, метод Ньютона требует некоторой коррекции.

Метод Гаусса-Ньютона

Применяется для минимизации функции стоимости, представленной в виде суммы квадратов ошибок.

Пусть $\mathbf{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n} e^2(i)$,

где коэффициент $\frac{1}{2}$ введён для упрощения последующего анализа. Все слагаемые ошибок в этой формуле вычисляются на основании вектора весов **w**, фиксированного на всём интервале наблюдения $1 \leqslant i \leqslant n$.

Сигнал ошибки $e(i)$ является функцией от настраиваемого вектора весов **w**. Для текущего значения $\mathbf{w}(n)$ зависимость $e(i)$ от **w** можно лианезировать следующим образом:

$$e'(i, \mathbf{w}) = e(i) + \left[\frac{\partial e(i)}{\partial \mathbf{w}}\right]^{\mathrm{T}}_{\mathbf{w}=\mathbf{w}(n)} \left(\mathbf{w} - \mathbf{w}(n)\right), i = 1, 2, ..., n.$$

Или в матричном виде:

$\mathbf{e}'(n, \mathbf{w}) = \mathbf{e}(n) + \mathbf{J}(n)\left(\mathbf{w} - \mathbf{w}(n)\right)$, где $\mathbf{e}(n)$ — вектор ошибки

$\mathbf{e}(n) = \left[e(1), e(2), ..., e(m)\right]^{\mathrm{T}}$,

$\mathbf{J}(n)$ — матрица Якоби ошибки

$$\mathbf{J}(n) = \begin{bmatrix} \frac{\partial e(1)}{\partial w_1} & \frac{\partial e(1)}{\partial w_2} & \cdots & \frac{\partial e(1)}{\partial w_m} \\ \frac{\partial e(2)}{\partial w_1} & \frac{\partial e(2)}{\partial w_2} & \cdots & \frac{\partial e(2)}{\partial w_m} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial e(n)}{\partial w_1} & \frac{\partial e(n)}{\partial w_2} & \cdots & \frac{\partial e(n)}{\partial w_m} \end{bmatrix}_{\mathbf{w}=\mathbf{w}(n)}.$$

Якобиан — это транспонированная матрица градиента $\nabla \mathbf{e}(n)$

$$\nabla \mathbf{e}(n) = \left[\nabla e(1), \nabla e(2), ..., \nabla e(n) \right].$$

Обновлённый вектор $\mathbf{w}(n+1)$ можно записать в следующем виде

$$\mathbf{w}(n+1) = \arg\min_{\mathbf{w}} \left\{ \frac{1}{2} \left\| \mathbf{e}'(n, \mathbf{w}) \right\|^2 \right\}$$

Используя формулу $\mathbf{e}'(n, \mathbf{w}) = \mathbf{e}(n) + \mathbf{J}(n)\left(\mathbf{w} - \mathbf{w}(n)\right)$ для оценки квадратичной Евклидовой нормы $\left\| \mathbf{e}'(n, \mathbf{w}) \right\|^2$, получим:

$$\frac{1}{2} \left\| \mathbf{e}'(n, \mathbf{w}) \right\|^2 = \frac{1}{2} \left\| \mathbf{e}(n) \right\|^2 + \mathbf{e}^{\mathbf{T}}(n) \mathbf{J}(n) \left(\mathbf{w} - \mathbf{w}(n)\right) + \frac{1}{2} \left(\mathbf{w} - \mathbf{w}(n)\right)^{\mathbf{T}} \mathbf{J}^{\mathbf{T}}(n) \mathbf{J}(n) \left(\mathbf{w} - \mathbf{w}(n)\right).$$

Дифференцируя это выражение по **w** и приравнивая результат к нулю, получим

$$\mathbf{J}^{\mathbf{T}}(n) \mathbf{e}(n) + \mathbf{J}^{\mathbf{T}}(n) \mathbf{J}(n) \left(\mathbf{w} - \mathbf{w}(n)\right) = 0.$$

Разрешая это уравнение относительно **w** и учитывая, что $\mathbf{w}(n+1) = \arg\min_{\mathbf{w}} \left\{ \frac{1}{2} \left\| \mathbf{e}'(n, \mathbf{w}) \right\|^2 \right\}$, можно записать

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \left(\mathbf{J}^{\mathbf{T}}(n) \mathbf{J}(n) \right)^{-1} \mathbf{J}^{\mathbf{T}}(n) \mathbf{e}(n).$$

Полученная формула полностью описывает метод Гаусса-Ньютона.

В отличие от метода Ньютона, требующего знания матрицы Гессе для функции стоимости $\mathbf{E}(\mathbf{w})$, метод Гаусса-Ньютона требует только знания матрицы Якоби вектора ошибки $e(n)$. Тем не менее для реализации итеративного метода Гаусса-Ньютона матрица произведения $\mathbf{J}^{\mathbf{T}}(n)\mathbf{J}(n)$ должна быть не сингулярной.
Для обеспечения несингулярности Якобиан $\mathbf{J}(n)$ должен иметь ранг *n*, то есть *n* строк матрицы $\mathbf{J}(n)$ должны быть линейно независимы.
Это условие выполняется не всегда. Для обеспечения необходимого ранга матрицы $\mathbf{J}(n)$, общей к произведению $\mathbf{J}^{\mathbf{T}}(n)\mathbf{J}(n)$, добавляют диагональную матрицу $\delta\mathbf{I}$, где $\mathbf{I}$ — единичная матрица. Параметр δ является малой положительной константой, обеспечивающей положительную определённость матрицы $\mathbf{J}^{\mathbf{T}}(n)\mathbf{J}(n)+\delta\mathbf{I}$ для всех *n*.
В итоге уравнение метода Гаусса-Ньютона записывается в другом виде

$$\mathbf{w}(n+1)=\mathbf{w}(n)-\left(\mathbf{J}^{\mathbf{T}}(n)\mathbf{J}(n)+\delta\mathbf{I}\right)^{-1}\mathbf{J}^{\mathbf{T}}(n)\mathbf{e}(n)$$

Влияние этой модификации постепенно ослабляется с увеличением количества итераций *n*.
Полученное рекурсивное выражение является решением задачи минимизации модифицированной функции стоимости:

$$\mathbf{E}(\mathbf{w})=\frac{1}{2}\left(\delta\left\|\mathbf{w}-\mathbf{w}(n)\right\|^2+\sum_{i=1}^{n}e^2(i)\right),$$

где $\mathbf{w}(n)$ — текущее значение вектора весовых коэффициентов $\mathbf{w}(i)$.

Линейный фильтр, построенный по методу наименьших квадратов

Этот фильтр строится для отдельного линейного нейрона

Функция стоимости $\mathbf{E}(\mathbf{w})$, используемая для создания этого фильтра, представляет собой сумму квадратов ошибок $\mathbf{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n} e^2(i)$.

На основе $y(i) = \mathbf{x}^{\mathrm{T}}(i)\mathbf{w}(i)$ и $e(i) = y(i) - d(i)$, для вектора ошибки $\mathbf{e}(n)$

$$\mathbf{e}(n) = \mathbf{d}(n) - \left[\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(n)\right]^{\mathrm{T}} \mathbf{w}(n) = \mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}(n),$$

где $\mathbf{d}(n)$ — вектор желаемого *отклика* размерности n

$$\mathbf{d}(n) = \left[\mathbf{d}(1), \mathbf{d}(2), ..., \mathbf{d}(n)\right]^{\mathrm{T}},$$

$\mathbf{X}(n)$ — матрица данных размерности $n \times m$

$$\mathbf{X}(n) = \left[\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(n)\right]^{\mathrm{T}}.$$

Дифференцируя выражения для $\mathbf{e}(n)$ по $\mathbf{w}(n)$, получим матрицу градиента

$$\nabla\mathbf{e}(n) = -\mathbf{X}^{\mathrm{T}}(n).$$

Следовательно, Якобиан $\mathbf{e}(n)$ можно записать в виде:

$$\mathbf{J}(n) = -\mathbf{X}(n).$$

Так как уравнение ошибки $\mathbf{e}'(n, \mathbf{w}) = \mathbf{e}(n) + \mathbf{J}(n)\left(\mathbf{w} - \mathbf{w}(n)\right)$ является линейным относительно вектора весовых коэффициентов $\mathbf{w}(n)$, метод Гаусса-Ньютона сходится за одну итерацию.

Подставляя $\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}(n)$ и $\mathbf{J}(n) = -\mathbf{X}(n)$ в выражение

$\mathbf{w}(n+1) = \mathbf{w}(n) - \left(\mathbf{J}^{\mathbf{T}}(n)\mathbf{J}(n) + \delta\mathbf{I}\right)^{-1}\mathbf{J}^{\mathbf{T}}(n)\mathbf{e}(n)$, получим

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\mathbf{X}^{\mathrm{T}}(n)\left(\mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}(n)\right) = \left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\mathbf{X}^{\mathrm{T}}(n)\mathbf{d}(n)$$

Выражение $\left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\mathbf{X}^{\mathrm{T}}(n)$ называют псевдообратной матрицей для матрицы данных $\mathbf{X}(n)$ и обозначают

$$\mathbf{X}^{+}(n)=\left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\mathbf{X}^{\mathrm{T}}(n)$$

Исходя из этого, последнее выражение для $\mathbf{w}(n+1)$ можно переписать в компактном виде

$$\mathbf{w}(n+1)=\mathbf{X}^{+}(n)\mathbf{d}(n)$$

То есть вектор весовых коэффициентов $\mathbf{w}(n+1)$ является решением линейной задачи фильтрации, решаемой по методу наименьших квадратов на интервале наблюдения длительности n.

Фильтр Винера как ограниченная форма линейного фильтра, построенного по методу наименьших квадратов, для эргодической среды

Для случая получения вектора входного сигнала $\mathbf{x}(i)$ и желаемого отклика $\mathbf{d}(i)$ из эргодической стационарной среды, вместо усреднения по времени можно использовать математическое ожидание по множеству. Такая среда описывается статистическими характеристиками второго порядка.

- Матрица корреляции $\mathbf{R}_{\mathbf{x}}$ вектора входного сигнала $\mathbf{x}(i)$.
- Вектор взаимной корреляции $\mathbf{r}_{\mathbf{xd}}$ между вектором входного сигнала $\mathbf{x}(i)$ и ожидаемого отклика $d(i)$.

Эти величины определяются выражениями:

$$\mathbf{R}_{\mathbf{x}} = E\left(\mathbf{x}(i)\mathbf{x}^{\mathrm{T}}(i)\right) = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}(i)\mathbf{x}^{\mathrm{T}}(i) = \lim_{n\to\infty}\frac{1}{n}\mathbf{X}^{\mathrm{T}}(i)\mathbf{X}(i),$$

$$\mathbf{r}_{\mathbf{xd}} = E\left(\mathbf{x}(i)d(i)\right) = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}(i)d(i) = \lim_{n\to\infty}\frac{1}{n}\mathbf{X}^{\mathrm{T}}(i)\mathbf{d}(i).$$

Тогда решение линейной задачи фильтрации по методу наименьших квадратов $\mathbf{w}(n+1) = \left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\mathbf{X}^{\mathrm{T}}(n)\mathbf{d}(n)$ можно переписать в виде

$$\mathbf{w}_o = \lim_{n\to\infty}\mathbf{w}(n+1) = \lim_{n\to\infty}\left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\mathbf{X}^{\mathrm{T}}(n)\mathbf{d}(n) = \lim_{n\to\infty}\frac{1}{n}\left(\mathbf{X}^{\mathrm{T}}(n)\mathbf{X}(n)\right)^{-1}\lim_{n\to\infty}\frac{1}{n}\mathbf{X}^{\mathrm{T}}(n)\mathbf{d}(n) = \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{r}_{\mathbf{xd}},$$

где $\mathbf{R}_{\mathbf{x}}^{-1}$ — обратная матрица для матрицы корреляции $\mathbf{R}_{\mathbf{x}}$. Вектор весовых коэффициентов $\mathbf{w}_o$ называют *Винеровским решением* линейной задачи оптимальной фильтрации. Исходя из этого, можно сформулировать утверждение.

Для эргодического процесса линейный фильтр, построенный по методу наименьших квадратов, асимптотически сходится к фильтру Винера по мере приближения количества наблюдений к бесконечности.

Для построения фильтра Винера необходимо знать статистические характеристики второго порядка: матрицу корреляции $\mathbf{R}_{\mathbf{x}}$ для вектора входного сигнала $\mathbf{x}(n)$ и вектора взаимной корреляции $\mathbf{r}_{\mathbf{xd}}$ между $\mathbf{x}(n)$ желаемым откликом $\mathbf{d}(n)$. Однако на практике эта информация обычно недоступна. В неизвестной среде можно использовать линейный адаптивный (возможность настраивать свободные параметры фильтра в соответствии со статистическими вариациями среды) фильтр. Алгоритм такой настройки на непрерывной основе является алгоритм минимизации среднеквадратической ошибки, который тесно связан с фильтром Винера.

Алгоритм минимизации среднеквадратической ошибки

Основан на использовании дискретных значений функции стоимости:

$$\mathbf{E}(\mathbf{w}) = \frac{1}{2} e^2(n),$$

где $e(n)$ — сигнал ошибки, измеренный в момент времени n. Дифференцируя $\mathbf{E}(\mathbf{w})$ по вектору весов $\mathbf{w}$, получим

$$\frac{\partial \mathbf{E}(\mathbf{w})}{\partial \mathbf{w}} = e(n) \frac{\partial e(n)}{\partial \mathbf{w}}.$$

Алгоритм минимизации среднеквадратической ошибки работает с линейным нейроном. Сигнал ошибки можно записать в виде:

$$e(n) = d(n) - \mathbf{x}^{\mathrm{T}}(n)\mathbf{w}(n).$$

Следовательно,

$$\frac{\partial e(n)}{\partial \mathbf{w}(n)} = -\mathbf{x}(n), \quad \frac{\partial \mathbf{E}(\mathbf{w})}{\partial \mathbf{w}(n)} = -\mathbf{x}(n)e(n).$$

Используя полученный результат, можно *оценить* вектор градиента

$$\hat{\mathbf{g}}(n) = -\mathbf{x}(n)e(n).$$

Используя эту формулу в методе наискорейшего спуска $\mathbf{w}(n+1) = \mathbf{w}(n) - \eta \mathbf{g}(n)$, можно сформулировать алгоритм минимизации среднеквадратической ошибки в виде:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta \mathbf{x}(n)e(n),$$

где η — параметр скорости обучения. Контур обратной связи для вектора весов $\hat{\mathbf{w}}(n)$ в алгоритм LMS ведёт себя как низкочастотный фильтр. Усреднённая временная константа этой фильтрации обратно пропорциональна параметру скорости обучения η, является мерой памяти алгоритма LMS.

Использование $\hat{\mathbf{w}}(n)$ вместо $\mathbf{w}(n)$ говорит, что алгоритм LMS только *оценивает* вектор весовых коэффициентов, который в алгоритме наискорейшего спуска изменяется по детерминированной траектории в пространстве весов при выбранном η.

В алгоритме LMS вектор $\hat{\mathbf{w}}(n)$ перемещается по случайной траектории (*стохастический градиентный алгоритм*). При бесконечном количестве итераций вектор $\hat{\mathbf{w}}(n)$ выписывает хаотическую траекторию вокруг Винеровского решения $\mathbf{w}_o$.

В отличие от метода наискорейшего спуска алгоритм LMS такжене требует знния статистических характеристик окружающей среды.

Краткое описание алгоритма LMS:

Обучающий пример: Вектор входного сигнала = $\mathbf{x}(n)$, желаемый отклик = $\mathbf{d}(n)$.

Выбираемый пользователем параметр: η.

Инициализация весов: $\hat{\mathbf{w}}(n) = 0$.

Вычислительна схема: $n = 1, 2, \ldots$

$$e(n) = d(n) - \mathbf{w}^{\mathrm{T}}(n)\hat{\mathbf{x}}(n)$$

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) - \eta\mathbf{x}(n)e(n)$$

Граф передачи сигнала для алгоритма минимизации среднеквадратической ошибки

Объединяя $e(n)=d(n)-\mathbf{x}^{\mathrm{T}}(n)\mathbf{w}(n)$ и $\hat{\mathbf{w}}(n+1)=\hat{\mathbf{w}}(n)+\eta\mathbf{x}(n)e(n)$, эволюцию вектора весов в алгоритме LMSможно представить в виде:

$$\hat{\mathbf{w}}(n+1)=\hat{\mathbf{w}}(n)-\eta\mathbf{x}(n)\left(d(n)-\mathbf{x}^{\mathrm{T}}(n)\hat{\mathbf{w}}(n)\right)=\left(\mathbf{I}-\eta\mathbf{x}(n)\mathbf{x}^{\mathrm{T}}(n)\right)\hat{\mathbf{w}}(n)+\eta\mathbf{x}(n)d(n),$$

где **I** — единичная матрица.

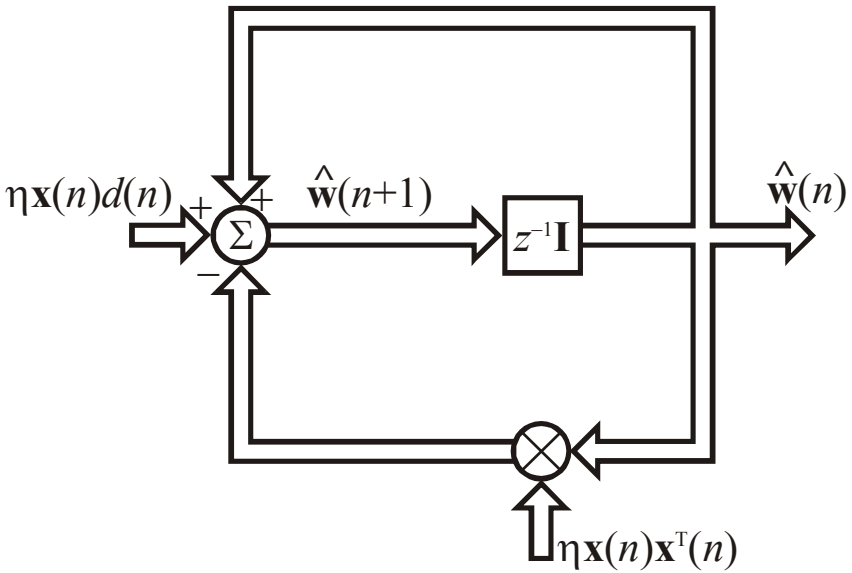
При использовании алгоритма LMS

$$\hat{\mathbf{w}}(n)=z^{-1}\left(\hat{\mathbf{w}}(n+1)\right),$$

где z^{-1} — оператор единичной задержки, реализующей память алгоритма.

Использовав последние два выражения, алгоритм LMS можно представить в виде графа передачи сигнала,

Граф передачи сигнала для алгоритма минимизации среднеквадратической ошибки



из чего видно, что алгоритм LMS является частным случаем стохастической системы с обратной связью. Наличие обратной связи оказывает влияние на сходимость алгоритма LMS.

Условия сходимости алгоритма LMS

Нижний контур (см. рис.) обратной связи обеспечивает изменение поведения алгоритма LMS.

Пропускная способность контура обратной связи определяется:

коэффициентом скорости обучения η;

вектором входного сигнала $\mathbf{x}(n)$.

То есть, сходимость алгоритма LMS зависит от статистических характеристик входного вектора $\mathbf{x}(n)$ и значения параметра η, который обеспечит сходимость алгоритма LMS.

Первый критерий сходимости алгоритма LMS: *сходимость в среднем*:

$E(\hat{\mathbf{w}}(n)) \to \mathbf{w}_o$ при $n \to \infty$,

где $\mathbf{w}_o$ — решение Винера. (Не практичен, так как последовательность произвольных векторов, имеющих среднее значение 0, также будет удовлетворять этому условию.

Практический критерий: *среднеквадратическая сходимость*:

$E(e^2(n)) \to \text{const}$ при $n \to \infty$.

Детальный анализ сходимости в смысле *среднеквадратической сходимости* сложен. Для упрощения делаются предположения.

1. Векторы входных сигналов $\mathbf{x}(1), \mathbf{x}(2), \ldots$ являются статистическими независимыми друг от друга.

2. В момент времени вектор n входного сигнала $\mathbf{x}(n)$ является статистически независимым от всех предыдущих желаемых откликов $d(1), d(2), \ldots, d(n-1)$.

3. В момент времени n желаемый отклик $d(n)$ зависит от вектора $\mathbf{x}(n)$, но статистически не зависит от всех предыдущих значений желаемого отклика.

4. Вектор входного сигнала $\mathbf{x}(n)$ и желаемый отклик $d(n)$ выбираются из множества, подчиняющегося распределению Гаусса.

Статистический анализ алгоритма LMS при наличии вышеописанных допущений называется *теорией независимости*.

При малом значении η алгоритм LMS сходится в смысле среднеквадратического значения, если η удовлетворяет условию

$$0 < \eta < \frac{2}{\lambda_{max}},$$

где λ_{max} — наибольшее собственное значение матрицы корреляции $\mathbf{R}_x$.

Значение λ_{max} как правило неизвестно. В качестве оценки сверху значения λ_{max} можно использовать *след* матрицы $\mathbf{R}_x$. Тогда условие будет выглядеть:

$$0 < \eta < \frac{2}{\mathrm{tr}[\mathbf{R}_x]}$$

где $\mathrm{tr}[\mathbf{R}_x]$ — след матрицы $\mathbf{R}_x$.

След матрицы определяется как сумма её диагональных элементов. Так как каждый из диагональных элементов матрицы корреляции $\mathbf{R}_x$ является среднеквадратическим значением соответствующего входного сигнала, условие сходимости алгоритма минимизации среднеквадратической ошибки формулируется как

$$0 < \eta < \frac{2}{\text{сумма среднеквадратических значений входных сигналов}}.$$

Если параметр η удовлетворяет этому условию, то алгоритм LMS сходится также и в смысле среднего значения (достаточное условие сходимости в среднем). Обратное утверждение верно не всегда.

Преимущества и недостатки алгоритма LMS

Преимущество: простота, независим от модели (является робастным). При малой неопределённости в модели и небольших возмущениях сигнал ошибки невелик.

Алгоритм LMS является оптимальным согласно минимаксному критерию H^{∞}.

Идея оптимальности в смысле H^{∞} — обеспечить наилучшее выполнение пессимистического сценария: *если вы не знаете с чем столкнулись, предположите наихудшее и оптимизируйте решение*.

Если среда нестационарна (статистические характеристики среды меняются во времени), то в этой среде оптимальное решение Винера зависит от времени, а алгоритм LMS выполняет дополнительную задачу отслеживания изменения параметров фильтра Винера.

Ограничение алгоритма: медленная скорость сходимости (в 10 раз больше итераций, чем размерность пространства входных сигналов) и чувствительность к изменению собственных значений матрицы входных сигналов (наибольшая чувствительность проявляется к изменению числа обусловленности или разброса собственных чисел матрицы корреляции $\mathbf{R}_{\mathbf{x}}$ входного вектора $\mathbf{x}$. Число обусловленности $\chi(\mathbf{R}_{\mathbf{x}})$ определяется как

$$\chi(\mathbf{R}_{\mathbf{x}}) = \frac{\lambda_{\max}}{\lambda_{\min}},$$

где $\lambda_{\min}$ и $\lambda_{\max}$ — минимальное и максимальное собственные числа матрицы $\mathbf{R}_{\mathbf{x}}$. Чувствительность алгоритма LMS к вариациям числа обусловленности $\chi(\mathbf{R}_{\mathbf{x}})$ становится проблемой, когда обучающая выборка, которой принадлежит вектор $\mathbf{x}(n)$, является плохо обусловленной, то есть число обусловленности $\chi(\mathbf{R}_{\mathbf{x}})$ достаточно велико.

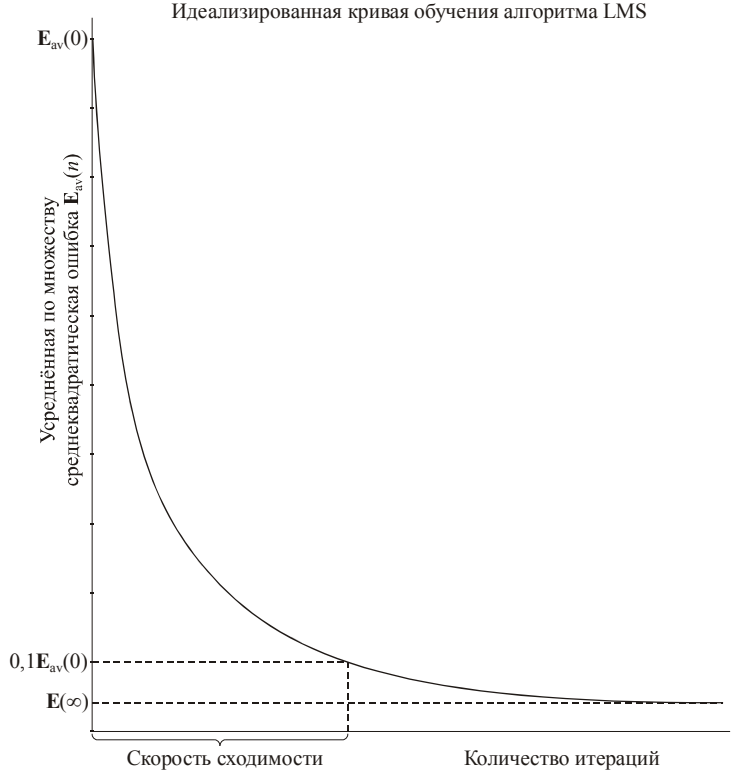
В алгоритме LMS матрица Гессе, определяемая как вторая производная от функции стоимости $\mathbf{E}(\mathbf{w})$ по вектору $\mathbf{w}$, эквивалентна матрице корреляции $\mathbf{R}_{\mathbf{x}}$, поэтому оба термина могут использоваться в одинаковом значении.

Графики процесса обучения

Информативный способ проверки сходимости алгоритма построение графиков процесса обучения (*кривые обучения*) для различных условий внешней среды.

Кривая обучения является *графиком среднеквадратического значения ошибки оценивания* $\mathbf{E}_{\text{av}}(\mathbf{w})$ в зависимости от *количества итераций n*.

В предположении устойчивости адаптивного фильтра усреднённая по множеству кривая обучения начинается с большого значения $\mathbf{E}_{\text{av}}(0)$, затем значения уменьшаются со скоростью, зависящей от типа фильтра, а в пределе сходятся к некоторому устойчивому значению $\mathbf{E}_{\text{av}}(\infty)$.



Основываясь на кривой можно определить скорость сходимости адаптивного фильтра (число итераций, необходимых для того, чтобы для любого произвольного начального значения $\mathbf{E}_{\text{av}}(0)$ величина $\mathbf{E}_{\text{av}}(n)$ уменьшилась в 10 раз.

Адаптивный фильтр обладает также характеристикой *рассогласования* **M**, получаемой из усреднения кривой обучения.

Пусть $\mathbf{E}_{\min}$ — минимальная среднеквадратическая ошибка, обеспечиваемая фильтром Винера, построенным на основе известных значений матрицы корреляции $\mathbf{R}_{\mathbf{x}}$ и вектора взаимной корреляции $\mathbf{r}_{\mathbf{x}d}$.

Рассогласование можно определить как

$$\mathbf{M} = \frac{\mathbf{E}(\infty) - \mathbf{E}_{\min}}{\mathbf{E}_{\min}} = \frac{\mathbf{E}(\infty)}{\mathbf{E}_{\min}} - 1.$$

Это безразмерная величина — мера измерения близости адаптивного фильтра к оптимальному в смысле среднеквадратической ошибки. Чем ближе **M** к единице, тем точнее адаптивная фильтрация алгоритма.

Характеристикой алгоритма LMS является также *время установки*.

Например, кривую обучения можно аппроксимировать экспоненциальной функцией с *усреднённой временной константой* τ_{av}. Чем меньше τ_{av}, тем короче время установки (алгоритм LMS быстрее сходится к установившемуся состоянию).

Параметр η прямо пропорционален M, а параметр τ_{av} обратно пропорционален η.

Противоречие: если уменьшать η для уменьшения M, увеличивается время установки алгоритма LMS, следовательно, для ускорения процесса обучения нужно увеличивать η.

Поэтому основное внимание уделяется выбору параметра η.

Изменение параметра скорости обучения по модели отжига

Сложность работы с алгоритмом LMS из-за эмпирического параметра η. Этот параметр можно не изменять $\eta(n) = \eta_0$ для всех n.

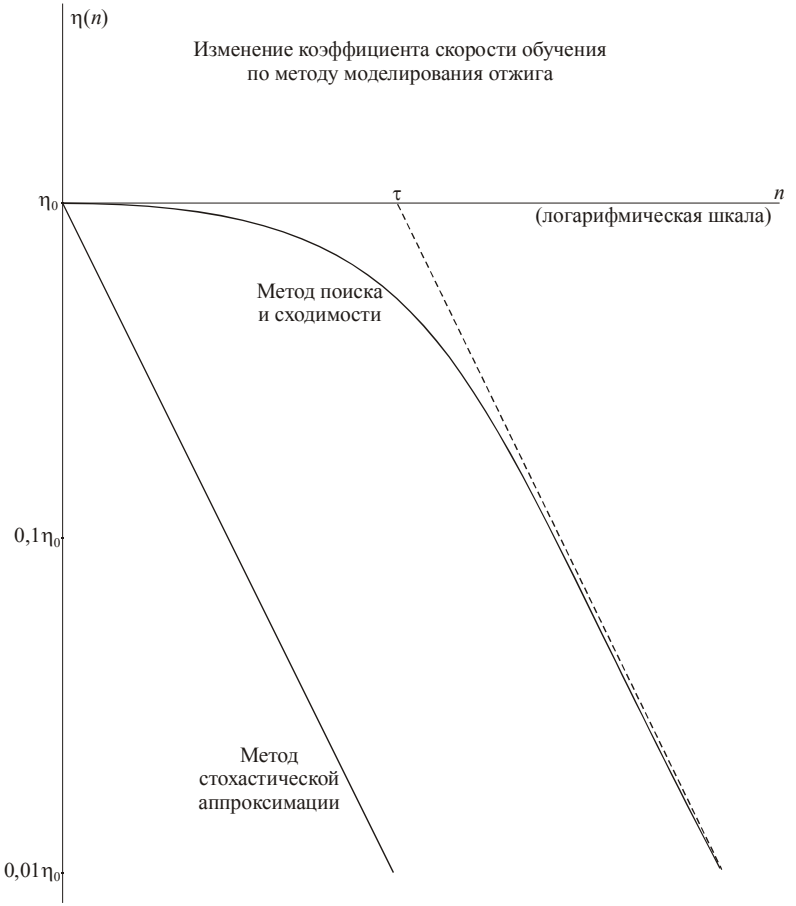
В стохастической аппроксимации параметр η изменяется со временем. Одна из форм изменения $\eta(n) = \frac{c}{n}$, где c — константа.

Такого выбора достаточно, чтобы гарантировать сходимость алгоритма стохастической аппроксимации. Если константа велика, есть риск выхода алгоритма из под контроля.

Альтернативой является подход на основе поиска и сходимости $\eta(n) = \frac{\eta_0}{1 + \frac{n}{\tau}}$, где η_0 и τ — заданные пользователем константы.

На первых шагах адаптации число n является малым по сравнению с константой времени поиска τ. Поэтому параметр скорости $\eta(n)$ практически равен константе η_0, и алгоритм ведёт себя как стандартный алгоритм LMS.

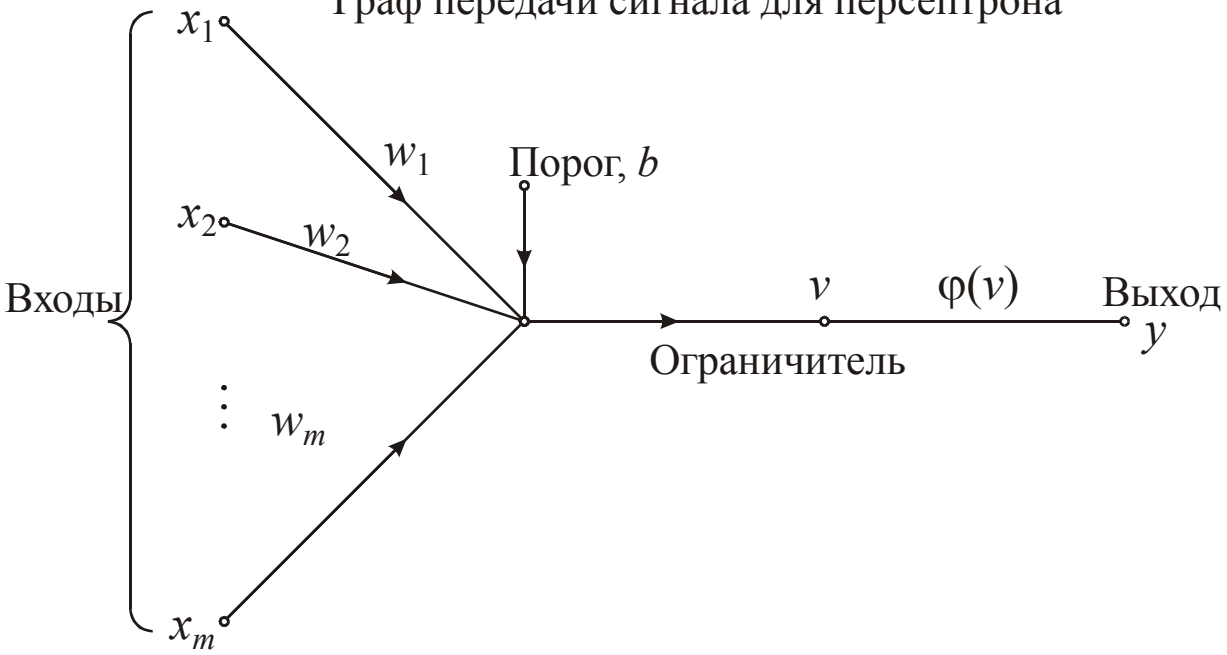
При количестве операций n, значительно превосходящих константу времени поиска τ, параметр интенсивности скорости $\eta(n)$ будет сходиться к функции $\frac{c}{n}$, где $c = \tau\eta_0$. Алгоритм при этом будет вести себя как обычный алгоритм стохастической аппроксимации, а веса будут сходиться к своим оптимальным значениям.



Персептрон

Персептрон строится для нелинейного нейрона (для модели Мак-Каллок-Питца). Ограничитель — пороговая функция вычисления знака.

Граф передачи сигнала для персептрона



Суммирующий узел этой нейронной модели вычисляет линейную комбинацию входных сигналов, поступающих на синапсы с учётом внешнего возмущения (порога). Полученная сумма (индуцированное локальное поле) передаётся на узел ограничителя. То есть, сигнал на выходе нейрона принимает $+1$, если синал на выходе сумматора положителен и -1 для отрицательного значения.

На диаграмме передачи сигнала синаптические веса $w_1, w_2, ..., w_m$ входные сигналы $x_1, x_2, ..., x_m$, пороговое значение — b. Индуцированное локальное поле:

$$v = \sum_{i=1}^{m} w_i x_i + b.$$

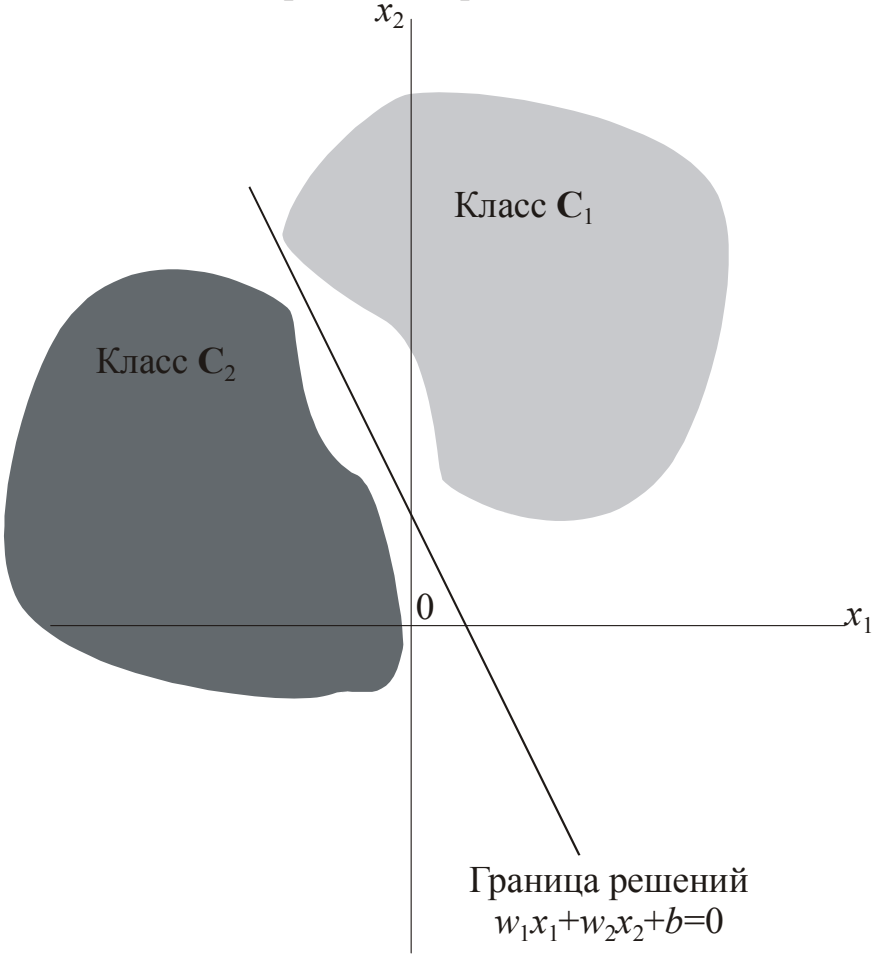
Целью персептрона является конкретное отнесение множества внешних стимулов $x_1, x_2, ..., x_m$ к одному из двух классов $\mathbf{C}_1$ или $\mathbf{C}_2$. Решающее правило такой классификации:

Входной сигнал относится к классу $\mathbf{C}_1$, если $y = +1$ и к классу $\mathbf{C}_2$, если $y = -1$.

Для изучения поведения классификатора строится карта областей решения в m-мерном пространстве сигналов $x_1, x_2, ..., x_m$. В простейшем случае будет две области решения, разделённые гиперплоскостьюЮ определяемой плоскостью

$$\sum_{i=1}^{m} w_i x_i + b = 0 .$$

Разделяющая поверхность в виде гиперплоскости (в данном случае - прямой) для двумерной задачи классификации образов на два класса



На рисунке для случая двух переменных, x_1 и x_2, когда разделяющая гиперплоскость вырождается в прямую. Точки (x_1, x_2), лежащие выше этой прямой, относятся к классу $\mathbf{C}_1$, точки ниже прямой — к классу $\mathbf{C}_2$.

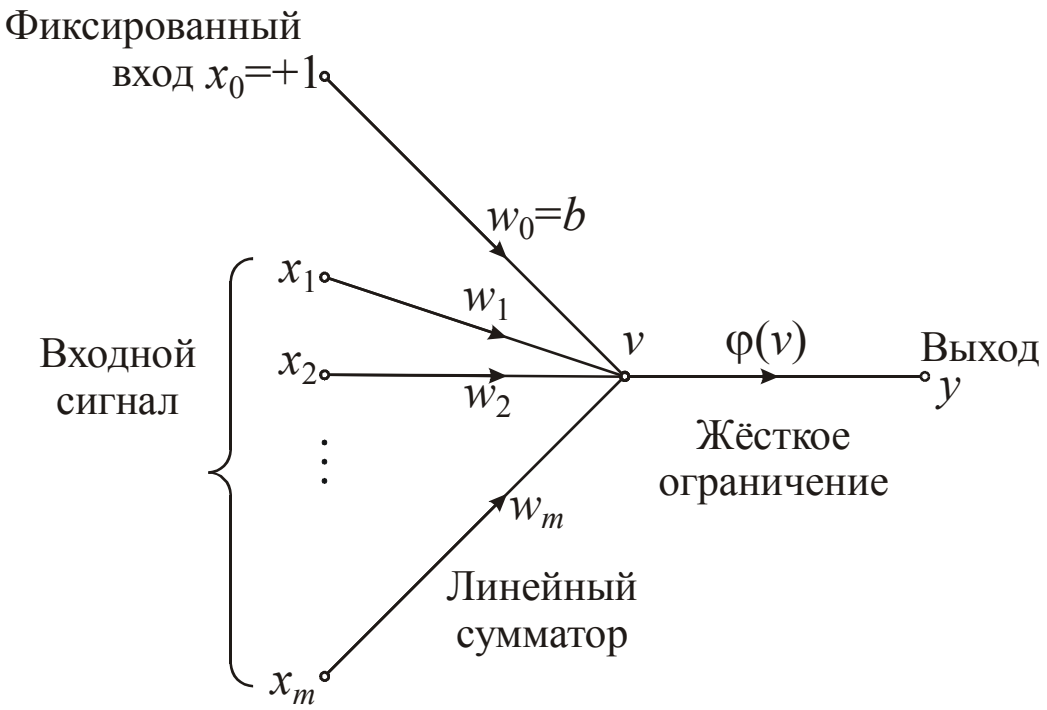
Пороговое значение смещает разделяющую поверхность по отношению к началу координат.

Синаптические веса персептрона $w_1, w_2, ..., w_m$ можно адаптировать итеративным методом с помощью алгоритма, основанного на коррекции ошибок (*алгоритм сходимости персептрона*).

Теорема о сходимости персептрона

Для вывода алгоритма обучения персептрона, основанного на коррекции ошибок, строится модифицированный граф передачи сигнала.

Эквивалентный граф передачи сигнала для персептрона



Порог $b(n)$ рассматривается как синаптический вес связи с фиксированным входным сигналом +1.

Входной вектор

$$\mathbf{x}(n)=\left[+1, x_1(n), x_2(n), ..., x_m(n)\right]^{\mathrm{T}}$$

имеет размерность $m+1$. Здесь n — номер итерация алгоритма.

$m+1$-мерный вектор весовых коэффициентов:

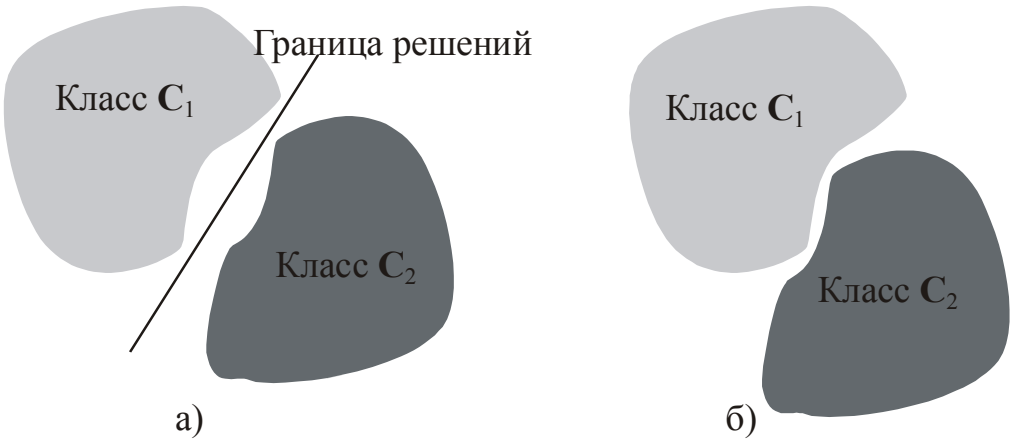
$$\mathbf{w}(n)=\left[b(n), w_1(n), w_2(n), ..., w_m(n)\right]^{\mathrm{T}}.$$

Выход линейного сумматора:

$$v(n)=\sum_{i=0}^{m} w_i(n)x_i(n)=w^{\mathrm{T}}(n)x(n),$$

где $w_0(n)$ — пороговое значение $b(n)$.

Пара линейно разделимых (а) и нелинейно разделимых образов (б)



При фиксированном значении n уравнение $\mathbf{w}^{\mathrm{T}}\mathbf{x}=0$ в m-мерном пространстве с координатами $x_1, x_2, ..., x_m$ определяет гиперплоскость, которая является поверхностью решений для двух различных классов входных сигналов. Для корректного функционирования персептрона два класса $\mathbf{C}_1$ и $\mathbf{C}_2$ должны быть линейно-разделимыми. Нелинейно-разделимый случай выходит за рамки вычислительных возможностей персептрона.

Пусть входные переменные персептрона принадлежат двум линейно-разделимым классам.

$\mathbf{X}_1$ — подмножество векторов обучения $x_1(1), x_1(2), ...,$ принадлежащее классу $\mathbf{C}_1$

$\mathbf{X}_2$ — подмножество векторов обучения $x_2(1), x_2(2), ...,$ принадлежащее классу $\mathbf{C}_2$

Объединение подмножеств $\mathbf{X}_1$ и $\mathbf{X}_2$ составляет всё обучающее множество **X**.

Существует такой вектор весовых коэффициентов **w**, для которого истинно утверждение:

$\mathbf{w}^T\mathbf{x} > 0$ для любого вектора x, принадлежащего классу $\mathbf{C}_1$,

$\mathbf{w}^T\mathbf{x} \leqslant 0$ для любого вектора x, принадлежащего классу $\mathbf{C}_2$.

Равенство $\mathbf{w}^T\mathbf{x} = 0$ произвольно назначено для класса $\mathbf{C}_2$.

При определённых таким образом подмножествах $\mathbf{X}_1$ и $\mathbf{X}_2$ задача обучения элементарного персептрона сводится к нахождению такого вектора **w**, для которого выполняются оба неравенства.

Формулировка алгоритма адаптации вектора весовых коэффициентов элементарного персептрона.

Если *n*-й элемент $x(n)$ обучающего множества корректно классифицирован с помощью весовых коэффициентов $w(n)$, вычисленных на *n*-м шаге алгоритма, то вектор весов не корректируется, то есть действует правило:

$\mathbf{w}(n+1) = \mathbf{w}(n)$, если $\mathbf{w}^T\mathbf{x}(n) > 0$ и $\mathbf{x}(n) \in \mathbf{C}_1$,

$\mathbf{w}(n+1) = \mathbf{w}(n)$, если $\mathbf{w}^T\mathbf{x}(n) \leqslant 0$ и $\mathbf{x}(n) \in \mathbf{C}_2$.

В противном случае вектор весов персептрона подвергается коррекции в соответствии правила:

$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\mathbf{x}(n)$, если $\mathbf{w}^T\mathbf{x}(n) > 0$ и $\mathbf{x}(n) \in \mathbf{C}_2$,

$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\mathbf{x}(n)$, если $\mathbf{w}^T\mathbf{x}(n) \leqslant 0$ и $\mathbf{x}(n) \in \mathbf{C}_1$,

где интенсивность настройки вектора весов на шаге n определяется параметром скорости обучения $\eta(n)$.

Если $\eta(n) = \eta > 0$, где η — константа, не зависящая от номера итерации *n*.

Вышеописанный алгоритм называется правилом адаптации с фиксированным приращением.

Сходимость правила адаптации с фиксированным приращением $\eta = 1$.

$\eta(n)$

В начале процесса обучения $\mathbf{w}(0) = 0$. Пусть для $n = 1, 2, ...$

$\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n) < 0,$

а входной вектор $x(n)$ принадлежит подмножеству $\mathbf{X}_1$. Это значит, что персептрон некорректно классифицировал векторы $\mathbf{x}(1), \mathbf{x}(2), ...,$ то есть условие

$\mathbf{w}^{\mathrm{T}}\mathbf{x} > 0$ для любого вектора x, принадлежащего классу $\mathbf{C}_1$,

$\mathbf{w}^{\mathrm{T}}\mathbf{x} \leqslant 0$ для любого вектора x, принадлежащего классу $\mathbf{C}_2$.

не выполнено. Следовательно, для $\eta(n) = 1$ можно использовать вторую строку правила

$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\mathbf{x}(n)$, если $\mathbf{w}^{\mathrm{T}}\mathbf{x}(n) \leqslant 0$ и $\mathbf{x}(n) \in \mathbf{C}_1$:

$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{x}(n)$, для $\mathbf{x}(n) \in \mathbf{C}_1$.

Так как начальное состояние $\mathbf{w}(0) = 0$, то это уравнение для $\mathbf{w}(n+1)$ можно решить итеративно и получить результат:

$\mathbf{w}(n+1) = \mathbf{x}(1) + \mathbf{x}(2) + ... + \mathbf{x}(n).$

Так как по предположению $\mathbf{C}_1$ и $\mathbf{C}_2$ линейно-разделимы, то существует такое решение $\mathbf{w}_0$, при котором будет выполняться условие $\mathbf{w}^{\mathrm{T}}\mathbf{x}(n) > 0$ для векторов $\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(n)$, принадлежащих подмножеству $\mathbf{X}_1$.

Для фиксированного решения $\mathbf{w}_0$ определяется положительное число α

$\alpha = \min_{\mathbf{x}(n) \in \mathbf{X}_1} \mathbf{w}_0^{\mathrm{T}}\mathbf{x}(n)$

Умножая обе части уравнения $\mathbf{w}(n+1) = \mathbf{x}(1) + \mathbf{x}(2) + ... + \mathbf{x}(n)$ на вектор строку $\mathbf{w}_0^{\mathrm{T}}$, получим

$\mathbf{w}_0^{\mathrm{T}}\mathbf{w}(n+1) = \mathbf{w}_0^{\mathrm{T}}\mathbf{x}(1) + \mathbf{w}_0^{\mathrm{T}}\mathbf{x}(1) + ... + \mathbf{w}_0^{\mathrm{T}}\mathbf{x}(n).$

В свете определения для α имеем

$$\mathbf{w}_0^{\mathrm{T}}\mathbf{w}(n+1) \geqslant n\alpha$$

Используем неравенство Коши-Шварца для двух векторов $\mathbf{w}_0$ и $\mathbf{w}(n+1)$ запишем

$$\|\mathbf{w}_0\|^2 \|\mathbf{w}(n+1)\|^2 \geqslant \left(\mathbf{w}_0^{\mathrm{T}}\mathbf{w}(n+1)\right)^2$$

где $\|\cdot\|$ — Евклидова норма векторного аргумента; $\mathbf{w}_0^T\mathbf{w}(n+1)$ —скалярное произведение векторов.

Согласно условию $\mathbf{w}_0^{\mathrm{T}}\mathbf{w}(n+1) \geqslant n\alpha$, учтём это в последнем неравенстве

$$\left(\mathbf{w}_0^T\mathbf{w}(n+1)\right)^2 \geq n^2\alpha^2$$

или

$$\|\mathbf{w}(n+1)\|^2 \geqslant \frac{n^2\alpha^2}{\|\mathbf{w}_0\|^2}$$

Уравнение $\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{x}(n)$, для $\mathbf{x}(n) \in \mathbf{C}_1$ можно решить итеративно, получив результат

Рассмотрим проблему с другой стороны, пусть

$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{x}(k)$, для $k = 1, \ldots, n$ и $\mathbf{x}(k) \in \mathbf{X}_1$

Вычисляя Евклидову норму векторов в обеих частях этого уравнения, получим:

$$\|\mathbf{w}(k+1)\|^2 = \|\mathbf{w}(k)\|^2 + \|x(k)\|^2 + 2\mathbf{w}^{\mathrm{T}}(k)\mathbf{x}(k).$$

Если персептрон некорректно классифицировал входной вектор $\mathbf{x}(k)$, принадлежащий подмножеству $\mathbf{X}_1$, то $\mathbf{w}^{\mathrm{T}}(k)\mathbf{x}(k) < 0$. Следовательно

$$\|\mathbf{w}(k+1)\|^2 \leqslant \|\mathbf{w}(k)\|^2 + \|\mathbf{x}(k)\|^2$$

или

$$\|\mathbf{w}(k+1)\|^2 - \|\mathbf{w}(k)\|^2 \leqslant \|\mathbf{x}(k)\|^2 \text{ для } k = 1, \ldots, n.$$

Применяя эти неравенства последовательно для $k = 1, ..., n$ и учитывая начальное допущение, что $\mathbf{w}(0) = 0$, получим

$\|\mathbf{w}(n+1)\|^2 \leqslant \sum_{k=1}^{n} \|\mathbf{x}(k)\|^2 \leqslant n\beta$, где β — положительное число $\beta = \max_{\mathbf{x}(k) \in \mathbf{X}_1} \|\mathbf{x}(k)\|^2$.

Это уравнение означает, что Евклидова норма вектора весов $\mathbf{w}(n+1)$ линейно возрастает с увеличением номера итерации *n*. При больших *n* номер итерации не может превышать некоторого значения $n_{\max}$, при котором неравенства $\|\mathbf{w}(n+1)\|^2 \geqslant \frac{n^2\alpha^2}{\|\mathbf{w}_0\|^2}$ и $\|\mathbf{w}(n+1)\|^2 \leqslant \sum_{k=1}^{n} \|\mathbf{x}(k)\|^2 \leqslant n\beta$ удовлетворяются со знаком равенства. Значит число $n_{\max}$ должно быть решением уравнения

$$\frac{n_{\max}^2 \alpha^2}{\|\mathbf{w}_0\|^2} = n_{\max}\beta.$$

Разрешая это уравнение для $n_{\max}$ относительно $\mathbf{w}_0$, получим

$$n_{\max} = \frac{\beta \|\mathbf{w}_0\|^2}{\alpha^2}.$$

Таким образом доказано, что для $\eta(n) = 1$ и $\mathbf{w}(0) = 0$ в предположении существования вектора решения $\mathbf{w}_0$ процесс адаптации синаптических весов персептрона должен прекратиться не позднее итерации $n_{\max}$.

Решение для $\mathbf{w}_0$ и $n_{\max}$ не единственное.

Формулировка теоремы сходимости для алгоритма обучения персептрона с фиксированным приращением.

Пусть подмножества векторов $\mathbf{X}_1$ *и* $\mathbf{X}_2$ *линейно разделимы. Пусть входные сигнала поступают персептрону только из этих подмножеств. Тогда алгоритм обучения персептрона сходится после некоторого числа* n_0 *итераций в том смысле, что* $\mathbf{w}(n_0) = \mathbf{w}(n_0+1) = \mathbf{w}(n_0+2) = ...$

является вектором решения для $n_0 \leqslant n_{\max}$.

Абсолютная процедура адаптации однослойного персептрона на основе коррекции ошибок, в которой $\eta(n)$ — переменная величина. В частности, пусть $\eta(n)$ такое, у которого есть наименьшее целое число, для котоого выполняется соотношение

$$\eta(n)\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n) > \left|\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n)\right|.$$

Согласно этой процедуре, если скалярное произведение $\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n)$ на шаге *n* имеет неверный знак, то $\mathbf{w}^{\mathrm{T}}(n+1)\mathbf{x}(n)$ на итерации $n+1$ будет иметь правильный знак. Таким образом, предполагается, что если знак произведения $\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n)$ некорректен, то можно изменить последовательность обучения для иетрации $n+1$, приняв $\mathbf{x}(n+1)=\mathbf{x}(n)$. То есть каждый из образов представляется персептрону до тех пор, пока он не будет классифицирован корректно.

Использование $\mathbf{w}(0)$ отличного от нуля приводит к увеличению или уменьшению количества итераций, необходимых для сходимости, в зависимости от того, насколько близким окажется исходное состояние $\mathbf{w}(0)$ к решению $\mathbf{w}_0$. Но в любом случае сходимость будет обеспечена.

Алгоритм сходимости персептрона в краткой форме

Переменные и параметры

$\mathbf{x}(n) = \left[+1, x_1(n), ..., x_m(n)\right]^{\mathrm{T}}$ — вектор-строка размерности $m+1$;

$\mathbf{w}(n) = \left[b(n), w_1(n), ..., w_m(n)\right]^{\mathrm{T}}$ — вектор-строка размерности $m+1$;

$b(n)$ — порог;

$y(n)$ — фактический отклик (дискретизированный);

$d(n)$ — желаемый отклик;

$0 < \eta \leqslant 1$ — параметр скорости обучения;

1. Инициализация

Пусть $\mathbf{w}(0) = 0$. Последующие вычисления для шагов $n = 1, 2, ...$

2. Активация

На шаге *n* активируем персептрон, используя вектор $x(n)$ с вещественными компонентами и желаемый отклик $d(n)$.

3. Вычисление фактического ответа

Вычисляем фактический отклик персептрона: $y(n) = \mathrm{sgn}\left(\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n)\right)$.

4. Адаптация вектора весов

Изменяем вектор весов персептрона:

$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta\left(d(n) - y(n)\right)\mathbf{x}(n)$, где

$$d(n) = \begin{cases} +1, \text{ если } \mathbf{x}(n) \in \mathbf{C}_1, \\ -1, \text{ если } \mathbf{x}(n) \in \mathbf{C}_2. \end{cases}$$

5. Продолжение

Увеличиваем номер итерации *n* на единицу и возвращаемся ко 2 пункту.

Таким образом, дискретный отклик $y(n)$ персептрона можно выразить в компактной форме:

$$y(n) = \operatorname{sgn}\left(\mathbf{w}^{\mathrm{T}}(n)\mathbf{x}(n)\right).$$

Входной сигнал $x(n)$ является вектором-строкой размерности $m+1$, первый элемент которого — фиксированная величина (равная +1) на всём протяжении алгоритма. Соответственно первым элементом вектора-строки весовых коэффициентов $\mathbf{w}(n)$ размерности $m+1$ является порог $b(n)$.

Дискретный желаемый отклик $d(n)$ определяется выражением

$$d(n) = \begin{cases} +1, \text{ если } \mathbf{x}(n) \in \mathbf{C}_1, \\ -1, \text{ если } \mathbf{x}(n) \in \mathbf{C}_2. \end{cases}$$

Таким образом, алгоритм адаптации вектора весовых коэффициентов $\mathbf{w}(n)$ соответствует правилу обучения на основе коррекции ошибок

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta\left(d(n) - y(n)\right)\mathbf{x}(n),$$

где η —параметр скорости обучения, а разность $d(n) - y(n)$ выступает в роли сигнала ошибки.

Выбирается параметр η из диапазона $0 < \eta \leqslant 1$ исходя из взаимоисключающих требований:

усреднение предыдущих входных сигналов, обеспечивающее устойчивость оценки вектора весов, требует малых значений η;

быстрая адаптация к реальным изменения распределения процесса, отвечающего за формирование векторов входного сигнала **x**, требует больших значений η.

Взаимосвязь персептрона и байесовского классификатора в гауссовской среде (204)

Классическая система классификации образов называется байесовским классификатором.

Байесовский классификатор

В байесовской процедуре проверки гипотез минимизируется средний риск **R**. Для задачи двух классов ($\mathbf{C}_1$ и $\mathbf{C}_2$) средний риск определяется:

$$\mathbf{R} = c_{11}p_1\int_{\mathbf{X}_1} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)d\mathbf{x} + c_{22}p_2\int_{\mathbf{X}_2} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)d\mathbf{x} + c_{21}p_1\int_{\mathbf{X}_2} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)d\mathbf{x} + c_{12}p_2\int_{\mathbf{X}_1} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)d\mathbf{x},$$

где, p_i — априорная вероятность того, что вектор наблюдения **x** (реализация случайного вектора **X**) принадлежит подпространству $\mathbf{X}_i$ при $i = 1, 2$, и $p_1 + p_2 = 1$; c_{ij} — стоимость решения в пользу класса $\mathbf{C}_i$, представленного подпространством $\mathbf{X}_i$, когда истинным является класс $\mathbf{C}_j$ (то есть вектор принадлежит подпространству $\mathbf{X}_j$) при $(i, j) = 1, 2$; $f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_i\right)$ — функция плотности условной вероятности случайного вектора **X** при условии, что вектор наблюдения x принадлежит подпространству $\mathbf{X}_i$ для $i = 1, 2$.

Первые два слагаемых в правой части уравнения для среднего риска представляют корректные решения (корректные классификации), а вторая пара слагаемых — некорректные (ошибки классификации). Каждое решение взвешивается произведением двух двух факторов: стоимости принятия решения решения и относительной частоты его принятия (то есть априорной вероятности).

Целью является нахождение стратегии *минимизации среднего риска*. В процессе принятия решения каждому вектору наблюдения **x** из пространства **X** должно быть сопоставлено какое-либо из подпространств — $\mathbf{X}_1$ или $\mathbf{X}_2$. Таким образом, $\mathbf{X} = \mathbf{X}_1 + \mathbf{X}_2$.

Тогда выражение для среднего риска можно переписать в эквивалентной форме

$$\mathbf{R} = c_{11}p_1\int_{\mathbf{X}_1} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)d\mathbf{x} + c_{22}p_2\int_{\mathbf{X}_2} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)d\mathbf{x} + c_{21}p_1\int_{\mathbf{X}_2} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)d\mathbf{x} + c_{12}p_2\int_{\mathbf{X}_1} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)d\mathbf{x},$$

где $c_{11} < c_{21}$ и $c_{22} < c_{12}$. Принимая во внимание, что

$$\int_{\mathbf{X}} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)d\mathbf{x} = \int_{\mathbf{X}} f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)d\mathbf{x} = 1$$

выражение для среднего риска сводится к следующему выражению

$$\mathbf{R} = c_{11}p_1 + c_{22}p_2 + \int_{\mathbf{X}_1} p_2\left(c_{12} - c_{22}\right)f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right) - p_1\left(c_{21} - c_{11}\right)f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)d\mathbf{x}$$

Первые два слагаемых в правой части представляют собой фиксированную стоимость. Теперь для минимизации среднего риска **R** можно вывести следующую стратегию оптимальной классификации.

1. Чтобы интеграл вносил отрицательный вклад в назначение риска **R**, все значения вектора наблюдения **x**, для которых подынтегральное выражение (выражение в квадратных скобках) является отрицательным, должны быть отнесены к подпространству $\mathbf{X}_1$ (то есть к классу$\mathbf{C}_1$).
2. Чтобы интеграл вносил положительный вклад в значение риска **R**, все значения вектора наблюдения **x**, для которых подынтегральное выражение является положительным, должны быть исключены из подпространства $\mathbf{X}_1$ (то есть отнесены к классу $\mathbf{C}_2$).
3. Значения **x**, при которых подынтегральное выражение равно нулю, не влияют на риск **R**. Их можно отнести к любому классу произвольным образом. В данном случае будем относить их к подпространству $\mathbf{X}_2$ (то есть к классу$\mathbf{C}_2$).

Принимая это, байесовский классификатор можно определить следующим образом.

Если выполняется условие $p_1\left(c_{21} - c_{11}\right)f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right) > p_2\left(c_{12} - c_{22}\right)f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)$,то вектор наблюдения x следует относить к подпространству $\mathbf{X}_1$ (то есть к классу $\mathbf{C}_1$), в противном случае — к подпространству $\mathbf{X}_2$ (то есть к классу $\mathbf{C}_2$).

Для упрощения изложения вводятся обозначения

$$\Lambda(\mathbf{x}) = \frac{f_{\mathbf{X}}(\mathbf{x}|\mathbf{C}_1)}{f_{\mathbf{X}}(\mathbf{x}|\mathbf{C}_2)},\ \xi = \frac{p_2(c_{12} - c_{22})}{p_1(c_{21} - c_{11})}.$$

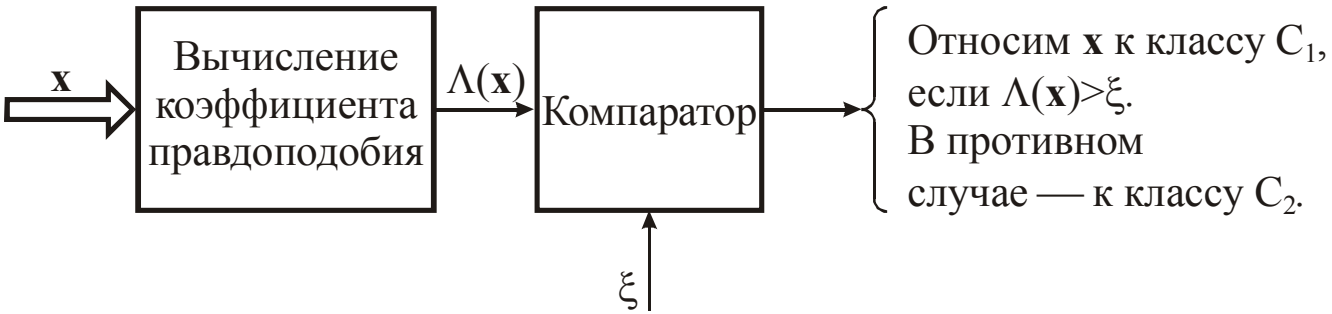
Величина $\Lambda(\mathbf{x})$, являющаяся частным двух функций плотности условной вероятности, называется *отношением правдоподобия*. Величина ξ называется *пороговым значением* процедуры проверки. Обе величины $\Lambda(\mathbf{x})$ и ξ — всегда положительны.

Тогда байесовский классификатор переопределяется следующим образом.

Если для вектора наблюдения **x** *отношение правдоподобия* $\Lambda(\mathbf{x})$ *превышает пороговый уровень* ξ*, то вектор* **x** *принадлежит к классу* $\mathbf{C}_1$*, в противном случае — к классу* $\mathbf{C}_2$.

На рисунке приведена блочная диаграмма байесовского классификатора

Реализация байесовского классификатора на основе отношения правдоподобия



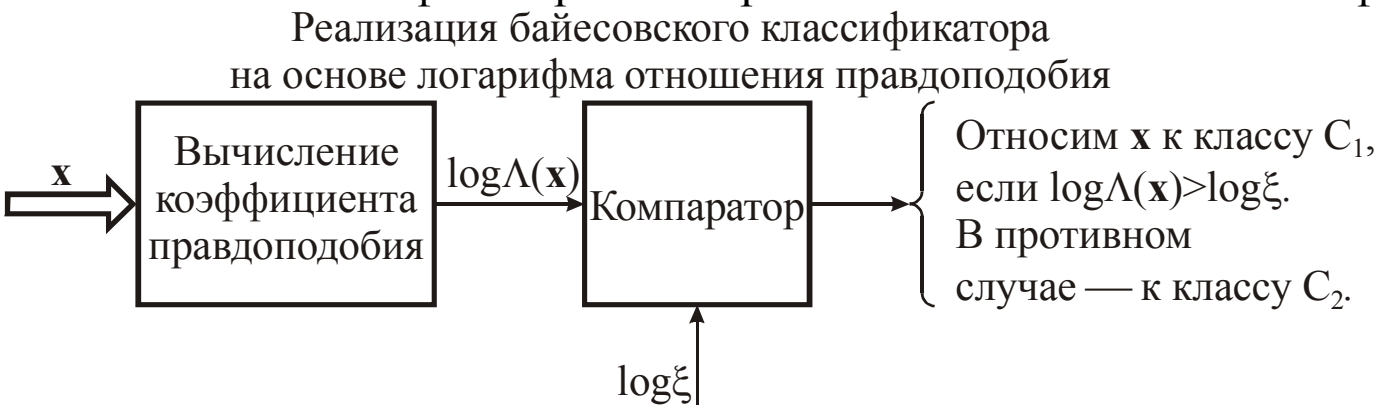
Свойства байесовского классификатора.

1. Обработка данных в байесовском классификаторе ограничена исключительно вычисление отношения правдоподобия $\Lambda(\mathbf{x})$.

2. Эти вычисления полностью инвариантны по отношению к значениям априорной вероятности и стоимости в процессе принятия решения. Эти значения влияют на величину порога ξ.

С вычислительной точки зрения удобнее работать с логарифмом отношения правдоподобия.

Байесовский классификатор можно реализовать в эквивалентной форме



Такой подход — *логарифмический критерий отношения правдоподобия*.

Байесовский классификатор и распределение Гаусса

Пусть случайная величина имеет распределение Гаусса. Среднее значение вектора X зависит от того, какому классу принадлежат его реализации — $\mathbf{C}_1$ или $\mathbf{C}_2$, однако матрица ковариации $\mathbf{X}$ остаётся одной и той же для обоих классов.

Класс $\mathbf{C}_1$: $E[\mathbf{X}] = \mu_1$, $E\left[(\mathbf{X}-\boldsymbol{\mu}_1)(\mathbf{X}-\boldsymbol{\mu}_1)^{\mathrm{T}}\right] = \mathbf{C}$.

Класс $\mathbf{C}_2$: $E[\mathbf{X}] = \mu_2$, $E\left[(\mathbf{X}-\boldsymbol{\mu}_2)(\mathbf{X}-\boldsymbol{\mu}_2)^{\mathrm{T}}\right] = \mathbf{C}$.

Матрица ковариации $\mathbf{C}$ не является диагональной, это означает, что образы классов $\mathbf{C}_1$ и $\mathbf{C}_2$ коррелированы. Предполагается, сто матрица $\mathbf{C}$ является несингулярной, поэтому существует обратная матрица $\mathbf{C}^{-1}$.

На основании этих соглашений, функция плотности условной вероятности $\mathbf{X}$ представляется как

$$f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_i\right) = \frac{1}{(2\pi)^{\frac{m}{2}}\left(\det(\mathbf{C})\right)^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^{\mathbf{T}}\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right), i = 1,2,$$

где m — размерность вектора наблюдения $\mathbf{x}$.

Введём предположения.

1. Вероятность принадлежности образа обоим классам — $\mathbf{C}_1$ и $\mathbf{C}_2$ — одинакова, то есть

$$p_1 = p_2 = \frac{1}{2}.$$

2. Ошибка классификации имеет постоянную стоимость, а корректная классификация стоимости не имеет: $c_{12} = c_{21}$ и $c_{11} = c_{22} = 0$.

Подставив выражение для $f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_i\right)$ в $\Lambda(\mathbf{x}) = \dfrac{f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_1\right)}{f_{\mathbf{X}}\left(\mathbf{x}\middle|\mathbf{C}_2\right)}$ и вычисляя натуральный логарифм, получим

$$\log\Lambda(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_1)^{\mathbf{T}}\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_1) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_2)^{\mathbf{T}}\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_2) = (\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^{\mathbf{T}}\mathbf{C}^{-1}\mathbf{x} + \frac{1}{2}\left(\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{\mu}_1\right).$$

Подставляя $p_1 = p_2 = \frac{1}{2}$, $c_{12} = c_{21}$ и $c_{11} = c_{22} = 0$ в $\xi = \frac{p_2(c_{12} - c_{22})}{p_1(c_{21} - c_{11})}$ и тоже находя натуральный лгарифм, получим

$\log \xi = 0$.

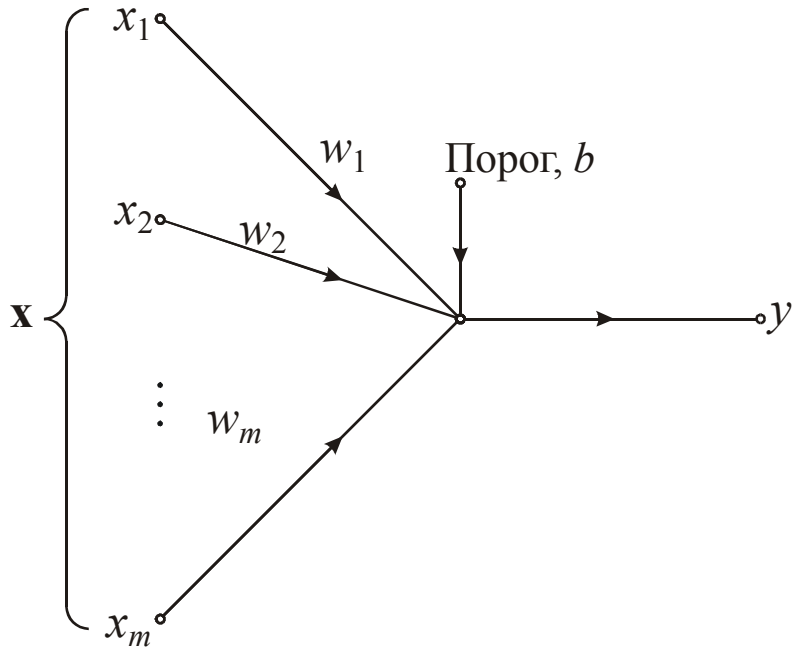
Последнее выражение и выражение для $\log \Lambda(\mathbf{x})$ свидетельствуют о том, что байесовский классификатор для данной задачи является *линейным классификатором*, описываемым соотношением

$y = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$,

где $y = \log \Lambda(\mathbf{x})$, $\mathbf{w} = \mathbf{C}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$, $b = \frac{1}{2}\left(\boldsymbol{\mu}_2^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{\mu}_1\right)$.

Боле точно, классификатор представляет собой линейный сумматор с вектором весов **w** и порогом *b*.

Граф передачи сигнала байесовского классификатора в гауссовой среде

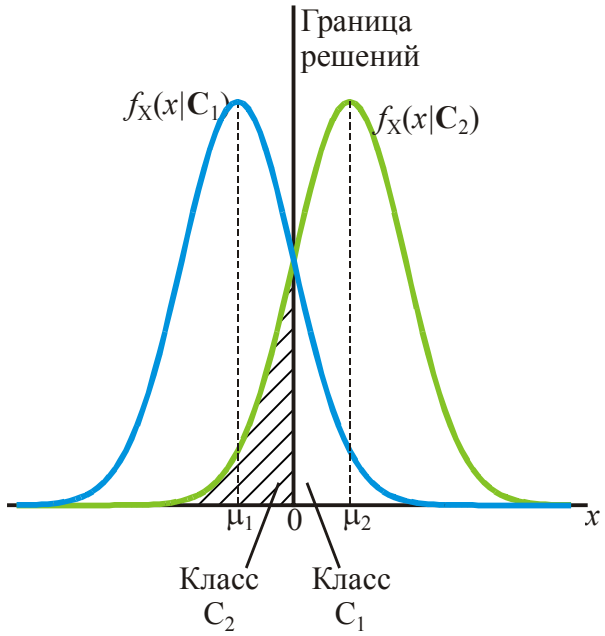


Логарифмический критерий отношения правдоподобия для задачи классификации на два класса.

Если выходной сигнал y линейного сумматора (содержащего порог b) положителен, вектор наблюдения ***x*** *относится к классу* $\mathbf{C}_1$*, в противном случае — к классу* $\mathbf{C}_2$.

Байесовский класификатор в гауссовой среде аналогичен персептрону в смысле линейности. Но есть небольшие различия.

Две пересекающиеся одномерные функции распределения Гаусса



• Персептрон работает при условии, что классифицируемые образы линейно разделимы. Распределение Гаусса в байесовском классификаторе предполагает пересечение образов. Границы этого пересечения определяются посредством векторов $\boldsymbol{\mu}_1$ и $\boldsymbol{\mu}_2$ и матрицы ковариации $\mathbf{C}$. Природа пересечения показана на рисунке для частного случая скалярной случайной переменной (то есть размерность $m = 1$). Если входные сигналы неразделимы и их функции распределения пересекаются, как показано на рисунке, алгоритм обучения персептрона не сходится, так как границы областей решения могут постоянно смещаться.

• Байесовский классификатор минимизирует вероятность ошибки классификации. Эта минимизация не зависит от пересечения между распределениями Гаусса двух классов. В частном случае, показанном на рисунке, байесовский классификатор всегда будет помещать границу областей решения в точку пересечения функций распределения Гаусса классов $\mathbf{C}_1$ и $\mathbf{C}_2$.

• Алгоритм работы персептрона является *непараметрическим*, то есть относительно формы рассматриваемых распределений никаких предварительных предположений не делается. Работа алгоритма базируется на коррекции ошибок, возникающих в точках пересечения функций распределения. Таким образом, персептрон хорошо работает с входными сигналами, генерируемыми нелинейными физическими процессами, даже если их распределения не симметричны и не являются гауссовыми, а это может ограничить область применения классификатора.

• Алгоритм сходимости персептрона является адаптивным и простым для реализации. Его требования к хранению информации ограничивается множеством синаптических весов и порогов. Архитектура байесовского классификатора является фиксированной; её можно сделать адаптивной только за счёт усиления требований к хранению информации и усложнения вычислений.

6 лекция

Однонаправленные многослойные сети сигмоидального типа

Объединённые в единую систему нейроны образуют искусственную нейронную сеть (ИНС).
Способы объединения:
Однонаправленные и рекуррентные (с обратной связью).

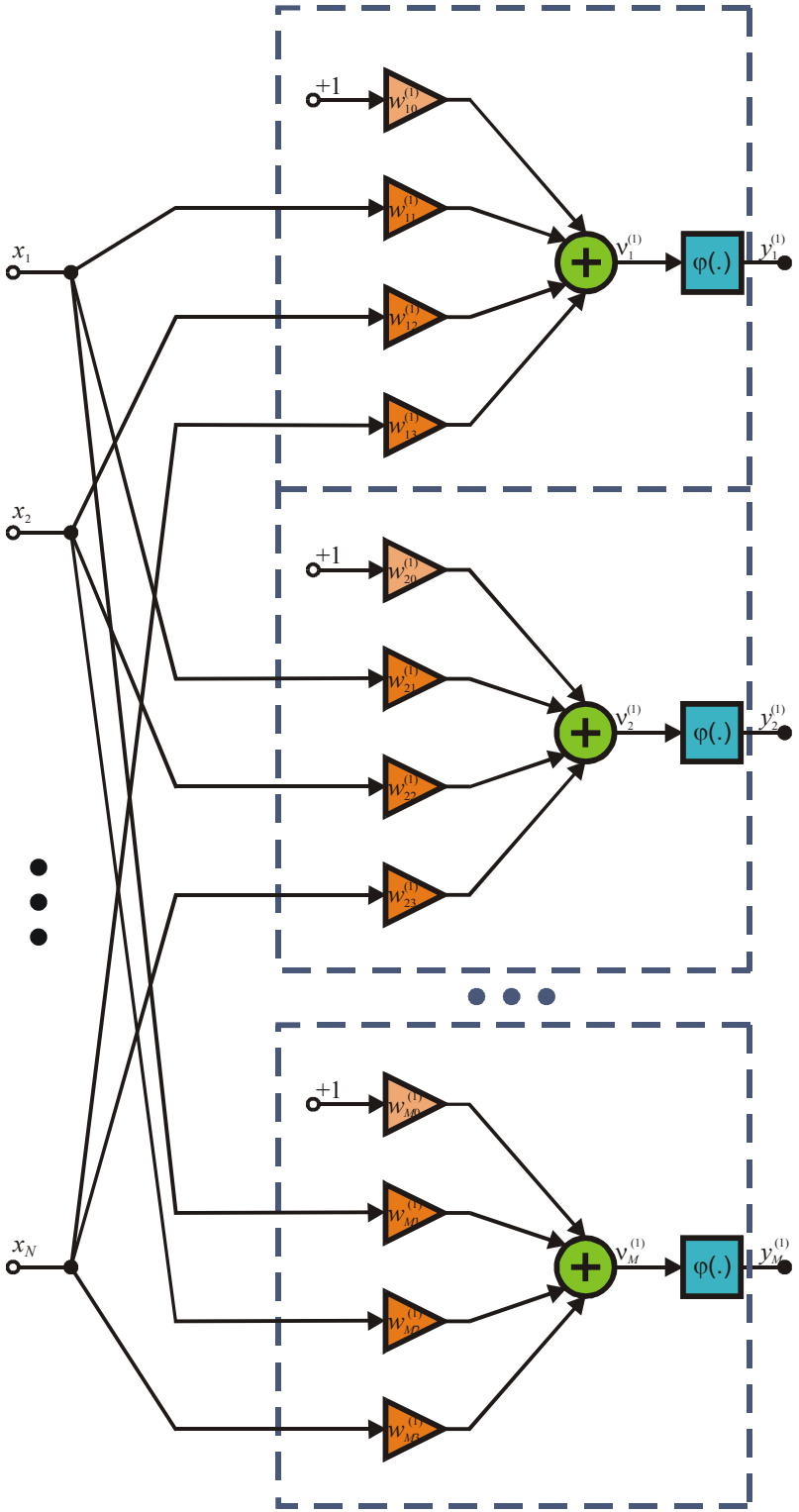
Однонаправленная многослойная сеть (MLP – MultiLayerPerceptron – многослойный персептрон)

Обучение такой сети, как правило, с учителем. Требуется подбор кортежей $\langle \mathbf{x}, \mathbf{d} \rangle$

x – входной вектор, **d** – выходной вектор сети.

Однако так как исторически первыми созданы однослойные сети и методы их обучения, и только позже предложена эффективная методика обучения многослойной сети, то сначала рассмотрим однослойную сеть.

Однослойную сеть образуют нейроны, расположенные в одной плоскости.



Каждый *i*-й нейрон имеет поляризацию (связь с весом w_{i0}, по которой поступает единичный сигнал), а также множество связей с весами w_{ij}, по которым поступают входные сигналы x_j. Значения весов подбираются в процессе обучения сети, состоящем в приближении выходных сигналов y_i к ожидаемым значениям d_i. Мерой близости считается значение целевой функции (стоимостной функции). При использовании p обучающих векторов $\langle \mathbf{x}, \mathbf{d} \rangle$ для обучения сети, включающей M выходных нейронов, целевую функцию можно определить евклидовой метрикой вида

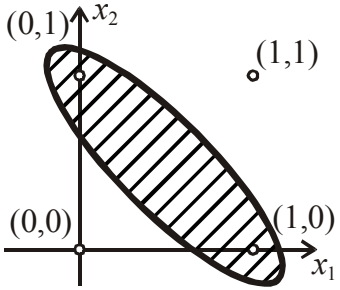
$$E = \frac{1}{2}\sum_{k=1}^{p} \left\| \mathbf{y}^{(k)} - \mathbf{d}^{(k)} \right\|^2 = \frac{1}{2}\sum_{k=1}^{p}\sum_{i=1}^{M} \left(\mathbf{y}_i^{(k)} - d_i^{(k)} \right)^2 \quad (1)$$

Выходные сигналы нейрона y_i являются функциями весов сети w_{ij}, значения которых уточняются в процессе обучения по критерию минимизации целевой функции.

Нейроны одного уровня функционирует независимо друг от друга, поэтому возможности такой сети ограничиваются свойствами отдельных нейронов. Веса нейронов образуют определённое пространство решений.

Выходной сигнал нейрона (1 или 0) будет зависеть от знака выражения $\sum_{j=0}^{N} w_{ij} x_j$, при фиксированных значениях весов зависит от расположения входного вектора **x**, который определяет гиперплоскость, разделяющую многомерное пространство на два подпространства. Поэтому задача классификации может быть решена с помощью единственного нейрона, если она относится к классу задач линейной сепарации (например, с применением логических функций AND или OR).

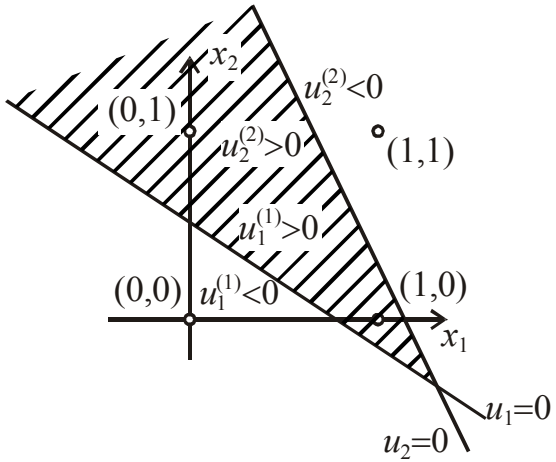
В случае применения функции XOR невозможно провести единственную линию, разделяющую пространство данных на 2 подпространства.



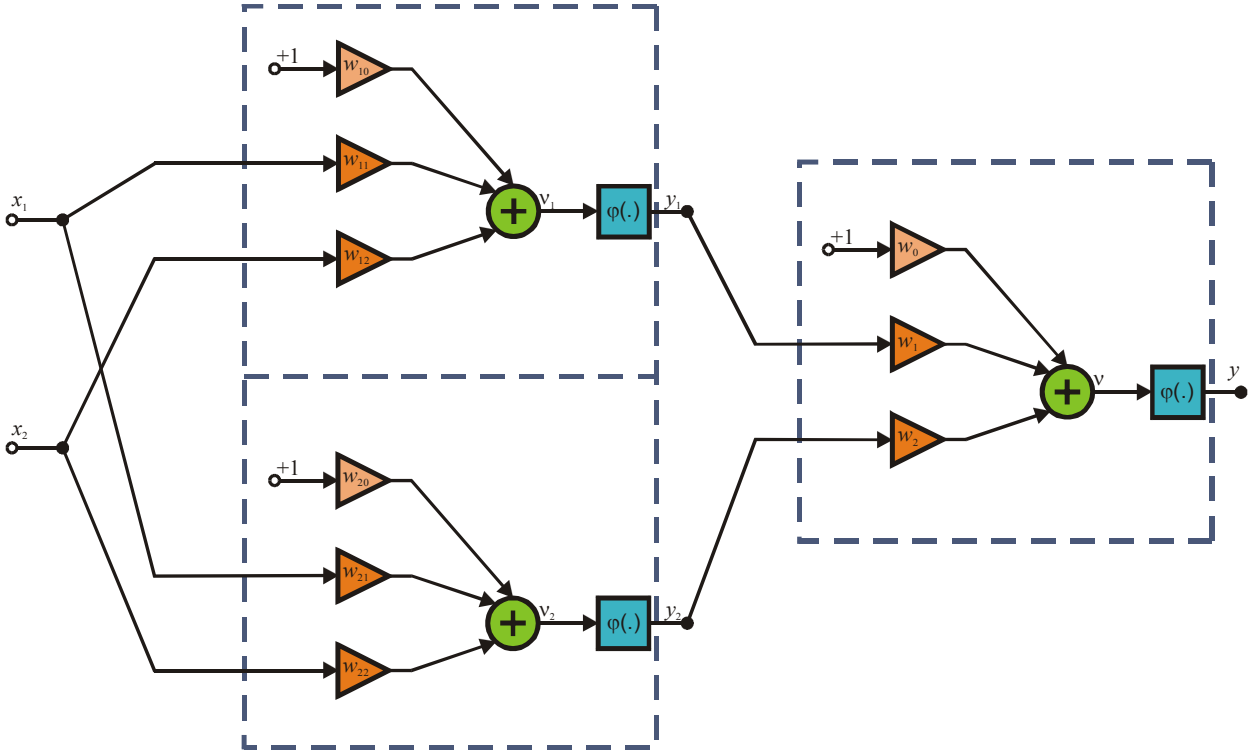
Добавив ещё один нейрон подберём веса так, чтобы они разделяли пространство на 2 части в зависимости от входного вектора $x = [x_1, x_2]^T$:

$u_1 = w_{11}x_1 + w_{12}x_2 + w_{10} > 0$ и $u_1 = w_{11}x_1 + w_{12}x_2 + w_{10} < 0$ первый нейрон

$u_2 = w_{21}x_1 + w_{22}x_2 + w_{20} > 0$ и $u_2 = w_{21}x_2 + w_{22}x_2 + w_{20} < 0$ второй нейрон



Добавлением на выходе ещё одного слоя из единственного нейрона, можно реализовать функцию логического суммирования, выделяющую общую часть подмножеств $u_1 > 0$, $u_2 > 0$.



Добавление в сеть дополнительного слоя позволило разрешить проблему невозможности линейного разделения данных.

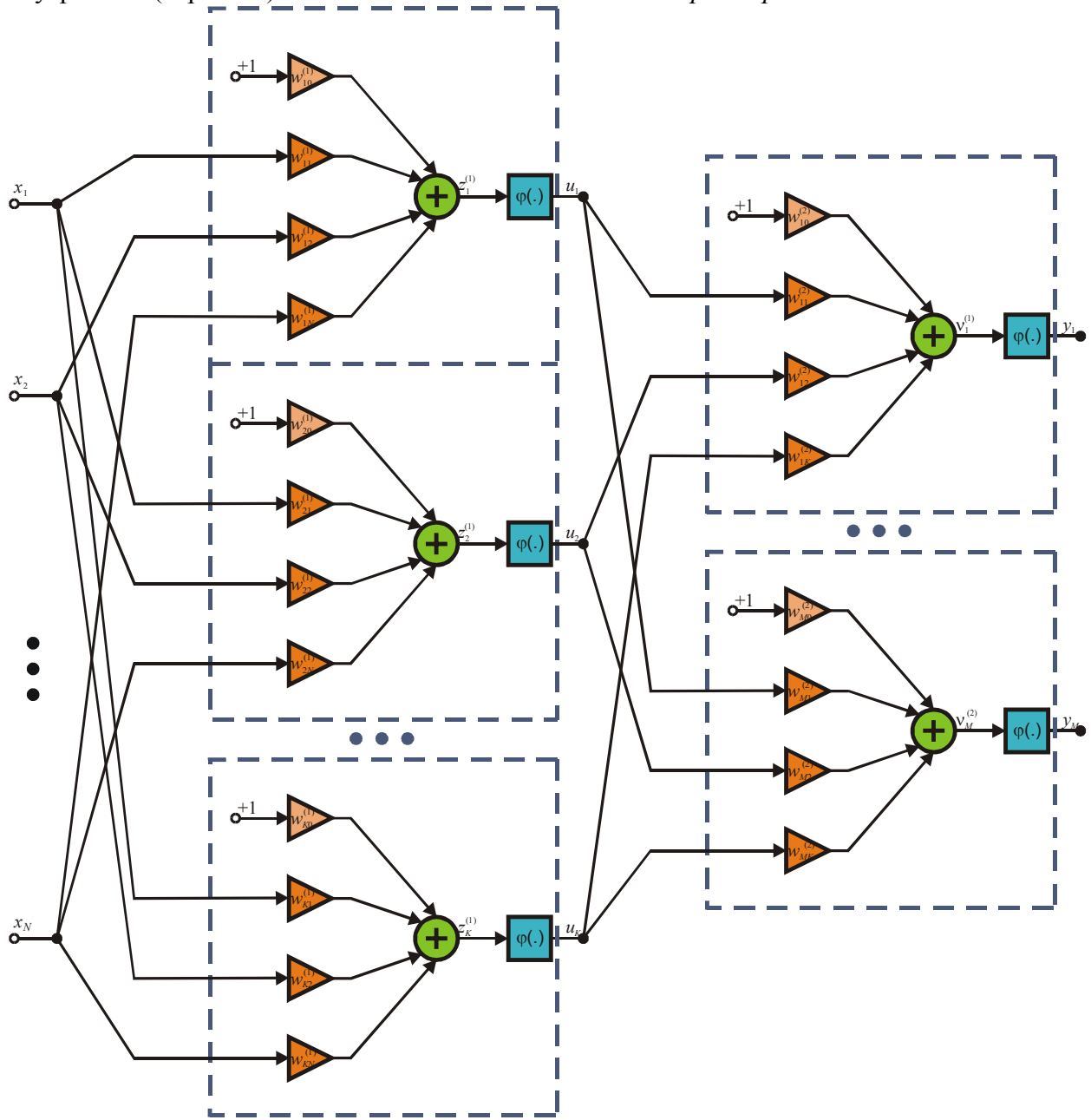
Несмотря на то, что однослойная сеть имеет небольшое практическое значение, её продолжают использовать там, где для решения поставленной задачи достаточно и одного слоя нейронов.

Выбор такой сети:

Количество входных нейронов определяется размерностью входного вектора **x**, а количество выходных нейронов определяется размерностью вектора **d**. Обучение сети производится, как правило, с учителем и является точной копией обучения одиночного нейрона.

Многослойный персептрон. Структура персептронной сети

Многослойная сеть состоящая из нейронов, расположенных на разных уровнях, причём, помимо входного и выходного слоёв, имеется ещё, как минимум, один внутренний (скрытый) слой называется *многослойным персептроном*.



На рисунке представлена сеть с одним скрытым слоем.

Веса нейронов скрытого слоя обозначены верхним индексом (1), а выходного слоя – верхним индексом (2). Выходные сигналы нейронов скрытого слоя обозначены $u_j\ (j=1,2,...,K)$, а выходного слоя – $y_j\ (j=1,2,...,M)$. Функция активации нейронов задана в сигмоидальной униполярной или биполярной форме.

Входной вектор $\mathbf{x}=[x_0,x_1,...,x_N]^T$, где $x_0=1$ соответствует единичному сигналу поляризации. С вектором $\mathbf{x}$ связаны два входных вектора сети: вектор фактических выходных сигналов $\mathbf{y}=[y_0,y_1,...,y_M]^T$ и вектор ожидаемых выходных сигналов $\mathbf{d}=[d_0,d_1,...,d_M]^T$.

Цель обучения состоит в подборе таких значений весов $w_{ij}^{(1)}$ и $w_{ij}^{(2)}$ для всех слоёв сети, чтобы при заданном входном векторе $\mathbf{x}$ получить на выходе значения сигналов y_i, которые с требуемой точностью будут совпадать с ожидаемыми значениями d_i для $i=1,2,...,M$.

Если рассматривать единичный поляризационный сигнал как один из компонентов входного вектора $\mathbf{x}$, то веса поляризации можно добавить в векторы весов соответствующих нейронов обоих слоёв. При таком подходе выходной сигнал *i*-го нейрона скрытого слоя

$$u_j=f\left(\sum_{j=0}^{N}w_{ij}^{(1)}x_j\right) \quad (2)$$

здесь индекс 0 соответствует сигналу и весам поляризации, причём $u_0\equiv 1$, $x_0\equiv 1$. В выходном слое *k*-й нейрон вырабатывает выходной сигнал

$$y_k=f\left(\sum_{i=0}^{K}w_{ki}^{(2)}u_i\right)=f\left(\sum_{i=0}^{K}w_{ki}^{(2)}f\left(\sum_{j=0}^{N}w_{ij}^{(1)}u_j\right)\right). \quad (3)$$

На значение выходного сигнала влияют веса обоих слоёв, тогда как сигналы, вырабатываемые в скрытом слое, не зависят от весов выходного слоя.

Алгоритм обратного распространения ошибки

Алгоритм обратного распространения ошибки определяет стратегию подбора весов многослойной сети с применением градиентных методов оптимизации. Основу составляет целевая функция, формулируемая, как правило, в виде квадратичной суммы разностей между фактическими и ожидаемыми значениями выходных сигналов. В случае единичной обучающей выборки $(\mathbf{x}, \mathbf{d})$ целевая функция определяется в виде

$$E(\mathbf{w}) = \frac{1}{2}\sum_{k=1}^{M}\left(y_k - d_k\right)^2 \quad (4)$$

При большом количестве обучающих выборок j $(j = 1, 2, ..., p)$ целевая функция превращается в сумму по всем выборкам

$$E(\mathbf{w}) = \frac{1}{2}\sum_{j=1}^{p}\sum_{k=1}^{M}\left(y_k^{(j)} - d_k^{(j)}\right)^2 \quad (5)$$

Уточнение весов может проводиться после предъявления каждой обучающей выборки, либо однократно после предъявления всех выборок, составляющих цикл обучения.

Цель обучения состоит в таком определении значений весов нейронов каждого слоя сети, чтобы при заданном входном векторе получить на выходе значения сигналов y_i, совпадающие с требуемой точностью с ожидаемыми значениями d_i при $i = 1, 2, ..., M$.

Обучение сети с использованием алгоритма обратного распространения ошибки проводится в несколько этапов. На первом из них предъявляется обучающая выборка **x** и рассчитываются значения сигналов соответствующих нейронов сети. При заданном векторе **x** определяются вначале значения выходных сигналов u_i скрытого слоя, а затем значения y_i нейронов выходного слоя. Для расчёта применяются формулы (2) и (3).

После получения значений выходных сигналов y_i становится возможным рассчитать фактическое значение целевой функции $E(\mathbf{w})$, заданной выражением (4).

На втором этапе минимизируется значение этой функции.

Если принять, что целевая функция непрерывна, то наиболее эффективными способами обучения оказываются градиентные методы оптимизации, согласно которым уточнение вектора весов (обучение) производится по формуле

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \Delta\mathbf{w} \quad (6)$$

где

$$\Delta\mathbf{w} = \eta\mathbf{p}(\mathbf{w}) \quad (7)$$

η – коэффициент обучения, а $\mathbf{p}(\mathbf{w})$ – направление в многомерном пространстве **w**.

Обучение многослойной сети с применением градиентных методов требует определения вектора градиента относительно весов всех слоёв сети, что необходимо для правильного выбора направления $\mathbf{p}(\mathbf{w})$. Эта задача имеет очевидное решение только для весов выходного слоя. Для других слоёв создана специальная стратегия, которая в теории ИНС называется алгоритмом обратного распространения ошибки (error backpropagation), отождествляемым, как правило, с процедурой обучения сети. В соответствии с этим алгоритмом в каждом цикле обучения выделяются следующие этапы.

1. Анализ нейронной сети в прямом направлении передачи информации при генерации входных сигналов, составляющих очередной вектор **x.** В результате такого анализа рассчитываются значения выходных сигналов нейронов скрытых слоёв и выходного слоя, а также соответствующие производные $\frac{df\left(u_i^{(1)}\right)}{du_i^{(1)}}, \frac{df\left(u_i^{(2)}\right)}{du_i^{(2)}}, \ldots, \frac{df\left(u_i^{(m)}\right)}{du_i^{(m)}}$ функций активации каждого слоя (*m* – количество слоёв сети).

2. Создание сети обратного распространения ошибок путём изменения направлений передачи сигналов, замена функций активации их производными и подача на бывший выход (а в настоящий момент – вход) сети возбуждения в виде разности между фактическим и ожидаемым значением. Для определённой таким образом сети необходимо рассчитать значения требуемых обратных разностей.

3. Уточнение весов (обучение сети) производится по предложенным выше формулам на основе результатов, полученных в п. 1 и 2, для оригинальной сети и для сети обратного распространения ошибки.

4. Описанный в п. 1, 2 и 3 процесс следует повторить для всех обучающих выборок, продолжая его вплоть до выполнения условия остановки алгоритма.

Действие алгоритма завершается в момент, когда норма градиента упадёт ниже априори заданного значения ε, характеризующего точность процесса обучения.

Рассмотрим этот процесс на примере сети приведённой на рисунке (выше).

Количество входных узлов — *N*.

Количество нейронов в скрытом слое — *K*.

Количество нейронов в выходном слое — *M*.

Будем использовать сигмоидальную функцию активации этих нейронов.

Основу алгоритма составляет расчёт значения целевой функции как квадратичной суммы разностей между фактическими и ожидаемыми значениями выходных сигналов сети. В случае единичной обучающей выборки $(\mathbf{x}, \mathbf{d})$ целевая функция задаётся формулой (4), а для множества обучающих выборок j $(j = 1, 2, ..., p)$ – формулой (5). Для упрощения излагаемого материала будем использовать целевую функцию вида (4), которая позволяет уточнять веса после предъявления каждой обучающей выборки.

$$E = \frac{1}{2}\sum_{k=1}^{M}\left(f\left(\sum_{i=0}^{K} w_{ki}^{(2)} u_i \right) - d_k \right)^2 = \frac{1}{2}\sum_{k=1}^{M}\left(f\left(\sum_{i=0}^{K} w_{ki}^{(2)} f\left(\sum_{j=0}^{N} w_{ij}^{(1)} x_i \right) \right) - d_k \right)^2 \quad (8)$$

Конкретные компоненты градиента рассчитываются дифференцированием зависимости (8). В первую очередь подбираются веса нейронов выходного слоя. Для выходных весов получаем:

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = \left(y_i - d_i\right)\frac{\mathrm{d}f\left(v_i^{(2)}\right)}{\mathrm{d}v_i^{(2)}} u_j \quad (9)$$

где

$v_i^{(2)} = \sum_{j=0}^{K} w_{ij}^{(2)} u_j$. Если ввести обозначение $\delta_i^{(2)} = \left(y_i - d_i\right)\frac{df\left(v_i^{(2)}\right)}{v_i^{(2)}}$, то соответствующий компонент градиента относительно весов нейронов выходного слоя можно представить в виде

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = \partial_i^{(2)} u_j \quad (10)$$

Компоненты градиента относительно нейронов скрытого слоя определяются по тому же принципу, однако они описываются другой, более сложной зависимостью, следующей из существования функции, заданной в виде

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \sum_{k=1}^{M}\left(y_k - d_k\right)\frac{dy_k}{du_i}\cdot\frac{du_i}{dw_{ij}^{(1)}} \quad (11)$$

После конкретизации отдельных составляющих этого выражения получаем:

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \sum_{k=1}^{M}\left(y_k - d_k\right)\frac{df\left(u_k^{(2)}\right)}{du_k^{(2)}} w_{ki}^{(2)} \frac{df\left(u_k^{(1)}\right)}{du_i^{(1)}} x_j \quad (12)$$

Если ввести обозначение

$$\partial_i^{(1)} = \sum_{k=1}^{M}\left(y_k - d_k\right)\frac{df\left(u_k^{(2)}\right)}{du_k^{(2)}} w_{ki}^{(2)} \frac{df\left(u_k^{(1)}\right)}{du_i^{(1)}} \quad (13)$$

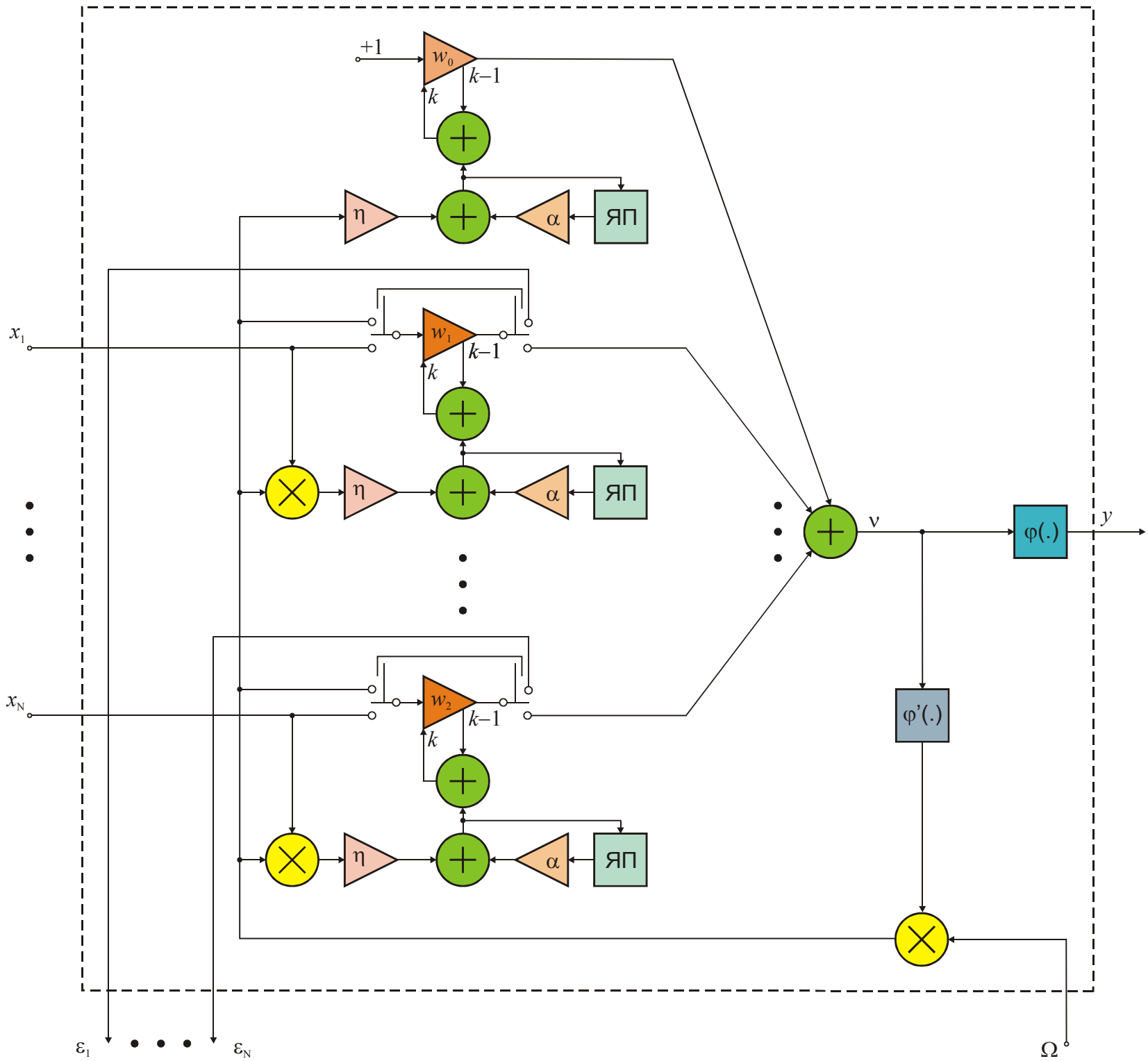
то получим выражение, определяющее компоненты градиента относительно весов нейронов скрытого слоя в виде

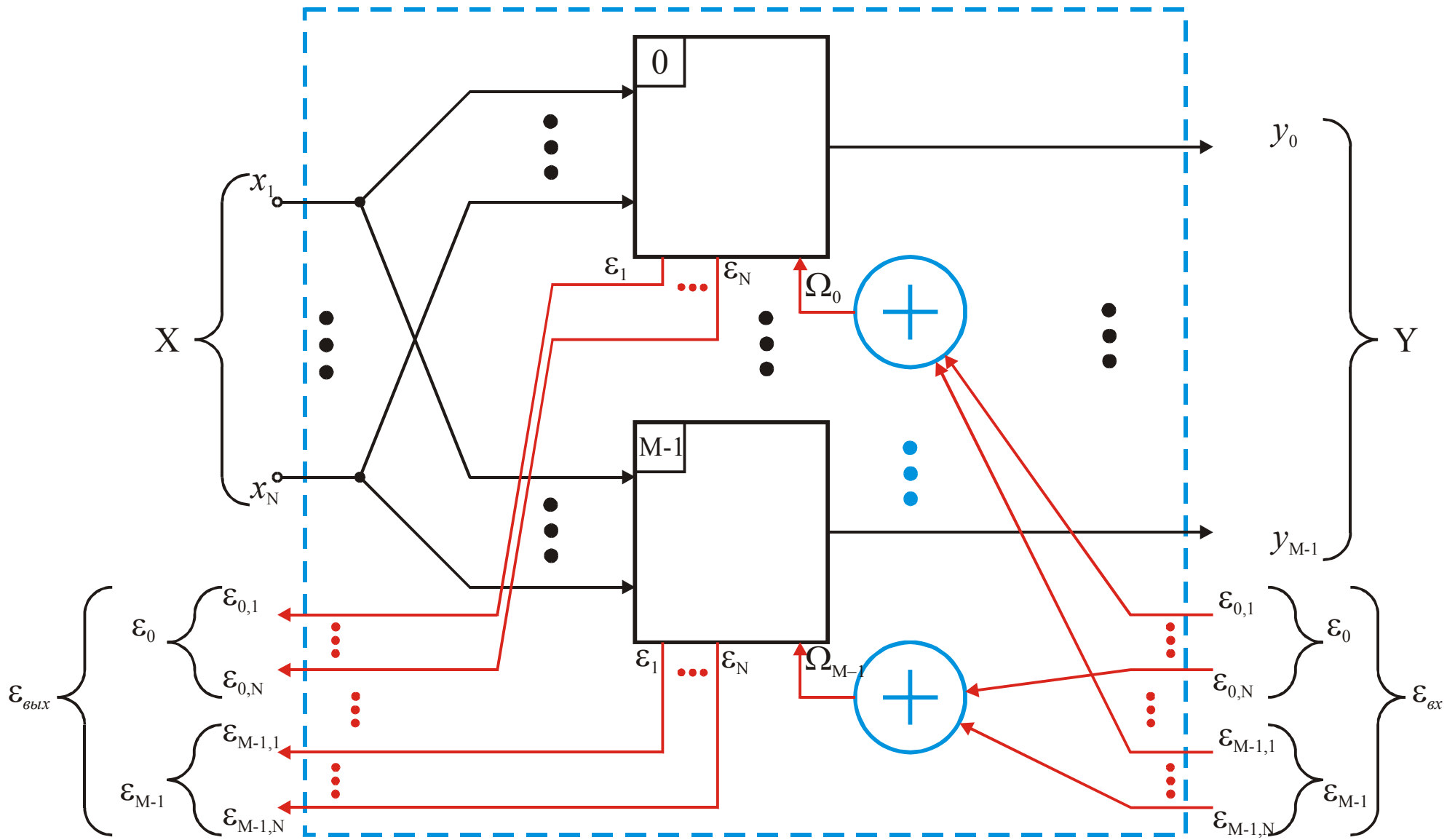
$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \partial_i^{(1)} x_j \quad (14)$$

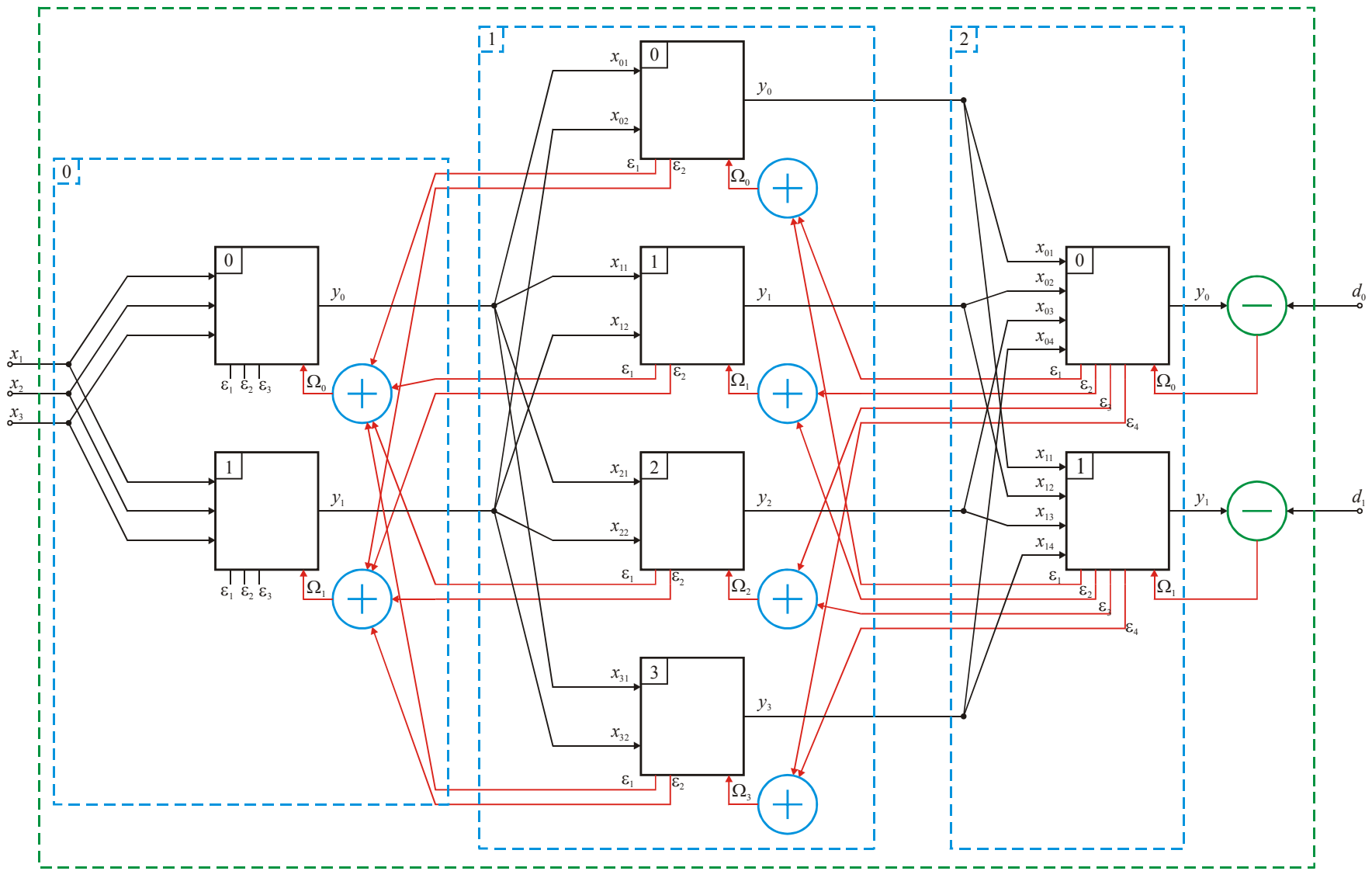
В обоих случаях ((10) и (14)) описание градиента имеет аналогичную структуру и представляется произведением двух сигналов: первый соответствует начальному узлу данной взвешенной связи, а второй – величине погрешности, перенесённой на узел, с которым эта связь установлена.

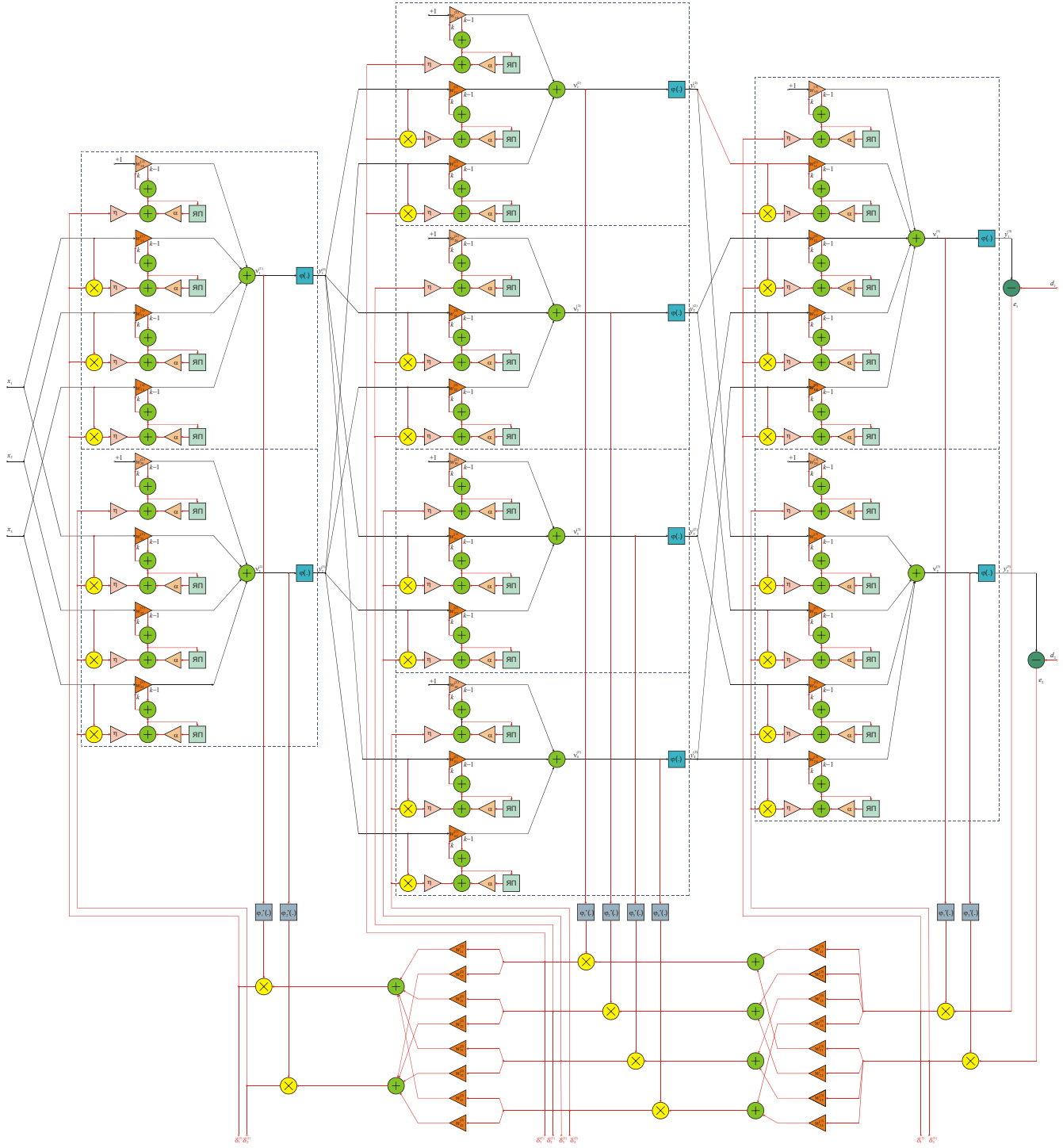
В классическом алгоритме обратного распространения ошибки фактор $\mathbf{p}(\mathbf{w})$, учитываемый в (6), задаёт направление отрицательного градиента, поэтому

$$\Delta\mathbf{w} = -\eta\nabla E(\mathbf{w}) \quad (15)$$


+1
w_0
k–1
k
η
α
ЯП
x_1
w_1
k–1
k
η
α
ЯП
x_N
w_2
k–1
k
η
α
ЯП
ν
φ(.)
y
φ'(.)
ε_1
ε_N
Ω


0
M-1
X
x_1
x_N
y_0
y_{M-1}
Y
ε_1
ε_N
Ω_0
Ω_{M-1}
$\varepsilon_{вых}$
ε_0
$\varepsilon_{0,1}$
$\varepsilon_{0,N}$
ε_{M-1}
$\varepsilon_{M-1,1}$
$\varepsilon_{M-1,N}$
$\varepsilon_{вх}$


0
1
2
0
1
2
3
0
1
0
1
x_1
x_2
x_3
x_{01}
x_{02}
x_{11}
x_{12}
x_{21}
x_{22}
x_{31}
x_{32}
x_{01}
x_{02}
x_{03}
x_{04}
x_{11}
x_{12}
x_{13}
x_{14}
y_0
y_1
y_2
y_3
y_0
y_1
d_0
d_1
ε_1
ε_2
ε_3
ε_4
Ω_0
Ω_1
Ω_2
Ω_3



p.txt

```
0.1     η
0.1     α
1       a
100000
2
4
[A]
0 0
[1]
1 1 1 1 0 1 1 1 1
[1]
0 1 0 1 0 1 0 1 0
[B]
1 0
[0]
1 1 1 1 1 0 1 0 1
[1]
0 1 0 1 1 1 1 1 0
[C]
1 1
[2]
0 1 0 0 0 0 0 1 0
[2]
0 1 0 0 0 0 0 0 0
```

d

```
0 1 0 1 1 1 0 1 0
```

```
neuron.exe file par.txt
neuron.exe file a1
neuron.exe file a2
neuron.exe file b1
neuron.exe file b2
neuron.exe file c1
neuron.exe file c2
```

```
neuron.exe file d
```

Элементы глобальной оптимизации

Все представленные ранее методы обучения нейронных сетей являются локальными – ведут к одному из локальных минимумов целевой функции, лежащему в окрестности точки начала обучения.

Если локальное решение признано неудовлетворительным, то процесс обучения повторяется при других начальных значениях весов (стохастический алгоритм – встряхивание весов).

Есть задачи, когда приблизительное местоположение минимума неизвестно. В этих случаях требуется глобальная оптимизация.

Алгоритм имитации отжига

Метод имитации отжига представляет собой алгоритмический аналог физического процесса управляемого охлаждения.

При отвердевании расплавленного материала его температура должна уменьшаться постепенно, вплоть до момента полной кристаллизации, чтобы не образовывались нерегулярности структуры материала, которые вызывают внутренние напряжения, из-за чего материал становится хрупким.

1. Процесс начинается из начальной точки w при заданной начальной «температуре» $T = T_{\max}$.

2. Пока $T > 0$, повторить L раз следующие действия:

- выбрать новое решение w' из окрестности w;
- рассчитать изменение целевой функции $\Delta = E(w') - E(w)$;
- если $\Delta \le 0$, принять $w = w'$; в противном случае (при $\Delta > 0$) принять, что $w = w'$ с вероятностью $e^{-\frac{\Delta}{T}}$ путём генерации случайного числа R из интервала (0…1) с последующим сравнением его со значением $e^{-\frac{\Delta}{T}}$; если $e^{-\frac{\Delta}{T}} > R$, принять новое решение $w = w'$; в противном случае проигнорировать его.

3. Уменьшить «температуру» $(T \leftarrow rT)$ с использованием коэффициента уменьшения r,выбираемого из интервала (0…1), и вернуться к п. 2.

4. После снижения температуры до нулевого значения провести обучение сети любым детерминированным методом, вплоть до достижения минимума целевой функции.

Проблемы: для повышения вероятности достижения глобального минимума длительность «отжига» (количество циклов L, повторяемых при одном и том же значении «температуры») должна быть достаточно большой, а коэффициент уменьшения «температуры» r — низким — это увеличивает продолжительность процесса моделирования и возникает проблема практической целесообразности.

Проблема конкурентоспособности: при многократном возобновлении процесса из различных точек. Грамотная статистическая обработка позволяет с высокой вероятностью достичь глобального минимума быстрее при помощи детерминированной оптимизации.

Выбор начальной «температуры» $T_{\max}$, коэффициента уменьшения «температуры» r, количество циклов L также влияет на эффективность метода.

Определение длительности моделирования процесса отжига затруднено.

Генетические алгоритмы

Имитируют процессы наследования свойств живыми организмами и генерируют последовательности новых векторов **w**, содержащие оптимизированные переменные: $\mathbf{w} = \left[w_1, w_2, ..., w_n \right]^T$. При этом выполняются операции трёх видов: селекция, скрещивание и мутация.

Каждому вектору **w** сопоставляется определённое значение целевой функции. Генетические операции выполняются для подбора таких значений переменных w_i вектора **w**, при которых максимизируется величина функции «приспособленности» (fitness function) $F(\mathbf{w})$. Она определяется через целевую функцию $E(\mathbf{w})$ путём её инвертирования.

На начальной стадии выполнения генетического алгоритма инициализируется определённая популяция хромосом (векторов **w**), формирующаяся случайным образом. Размер популяции пропорционален количеству оптимизируемых параметров.

Малая популяция хромосом приводит к замыканию в неглубоких локальных минимумах.

Большая — чрезмерно удлиняет вычислительную процедуру и также может не привести к точке глобального минимума.

Случайный выбор векторов **w**, составляющих исходную популяцию хромосом, делает их статистически независимыми и представляют собой начальное погружение в пространство параметров.

Часть этих хромосом лучше «приспособлена к существованию» (бóльшие значения функции соответствия и меньшие — целевой функции), другая — хуже.

Упорядочение хромосом в популяции производится от лучших к худшим.

Хромосомы отбираются (селекция) для формирования очередного поколения по значениям функции соответствия.

Селекция хромосом для спаривания может основываться, например, на **принципе элитарности**:

наиболее приспособленные хромосомы сохраняются, а наихудшие отбраковываются и заменяются вновь созданным потомством, полученным в результате скрещивания пар родителей; подбираются такие пары хромосом, потомство которых может быть включено в популяцию путём селекции;

спаривание:

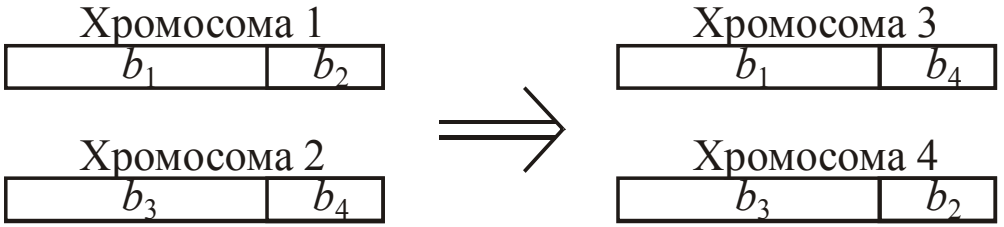
случайное (среди наиболее приспособленных хромосом)

взвешенно-случайное (в процессе отбора учитывается информация о текущем значении функции приспособленности — случайно, но вероятность отбираемых пропорционален величине её относительной функции приспособленности)

турнирная система (случайным образом отбираются несколько хромосом, среди которых определяются наиболее приспособленные — с наименьшим значением целевой функции)

Процесс скрещивания

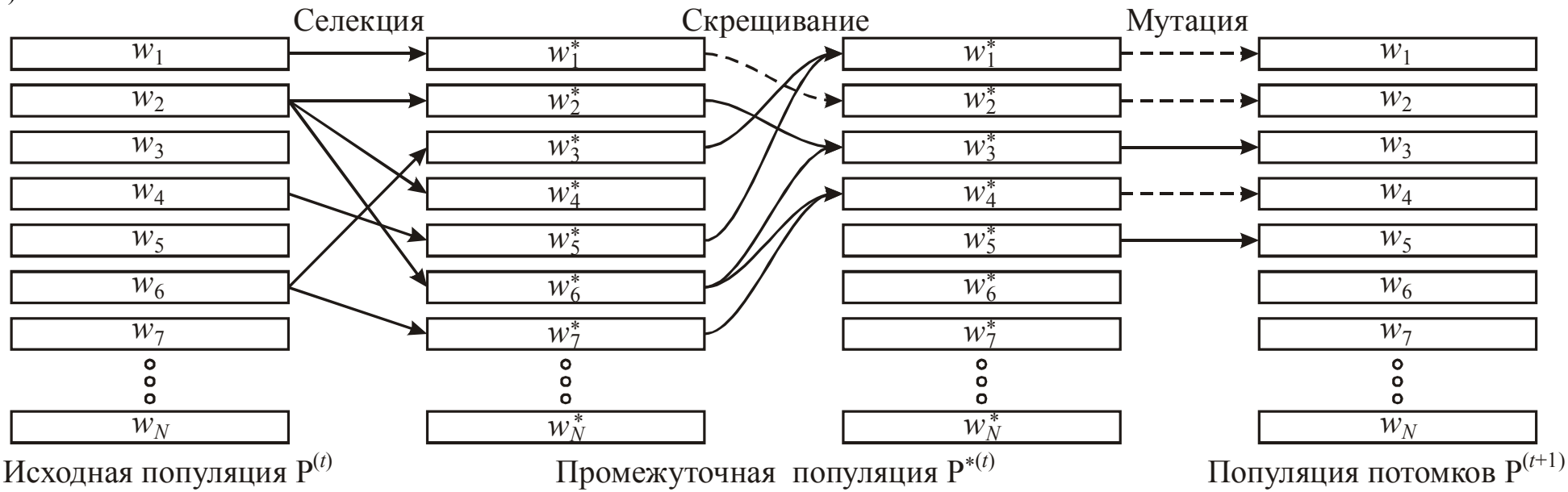
Основан на рассечении пары хомосом на 2 части с последующим обменом этих частей в хромосомах родителей. Место рассечения выбирается случайным образом.



Количество новых потомков равно количеству отбракованных в результате селекции. После их добавления к оставшимся хромосомам размер популяции остаётся неизменным. При вероятности успеха скрещивания на уровне 0,6…1 допускается перенос хромосом в очередное поколение без скрещивания.

Мутация. Замена значения какого-либо элемента вектора другим, случайно выбранным значением — защита от слишком раннего завершения алгоритма и от представления в какой-либо конкретной позиции всех хромосом одного и того же значения.

Хорошие результаты обучения приносит объединение алгоритмов глобальной оптимизации с детерминированными методами. На первом этапе обучения сети применяется выбранный алгоритм глобальной оптимизации, а после достижения целевой функцией определённого уровня включается детерминированная оптимизация с использованием какого-либо локального алгоритма (наискорейшего спуска, переменной метрики, Левенберга-Марквардта или сопряжённых градиентов)



РАДИАЛЬНЫЕ НЕЙРОННЫЕ СЕТИ

Радиальные сети представляют собой естественное дополнение сигмоидальных сетей.

Сигмоидальный нейрон представляется в многомерном пространстве гиперплоскостью, которая разделяет это пространство на две категории (2 класса), в которых выполняется одно из двух условий: либо $\sum_j w_{ij} x_j > 0$, либо $\sum_j w_{ij} x_j < 0$.

Радиальный нейрон представляет собой гиперсферу, которая осуществляет шаровое разделение вокруг центральной точки.

Круговая симметрия данных позволяет уменьшить количество нейронов, необходимых для разделения различных классов.

Структура типичной радиальной сети включает входной слой, на который подаются сигналы **x**, скрытый слой с нейронами радиального типа и выходной слой.

Функция выходного нейрона сводится исключительно к взвешенному суммированию сигналов, генерируемых скрытыми нейронами.

Математические основы

Теорема Т. Ковера о распознаваемости образов: нелинейные проекции образов в некоторое многомерное пространство могут быть линейно разделены с большей вероятностью, чем при их проекции в пространство с меньшей размерностью.

Если вектор радиальных функций $\boldsymbol{\varphi}(\mathbf{x}) = \left[\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), ..., \varphi_K(\mathbf{x})\right]^T$ в *N*-мерном входном пространстве обозначить $\boldsymbol{\varphi}(\mathbf{x})$, то это пространство является нелинейно φ-разделяемым на два пространственных класса X^+ и X^- тогда, когда существует такой вектор весов **w**, что

$\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) > 0, \mathbf{x} \in X^+$

$\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) < 0, \mathbf{x} \in X^-$

Граница между этими классами определяется уравнением $\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) = 0$.

Доказано, что каждое множество образов, случайным образом размещённых в многомерном пространстве, является φ-разделяемым с вероятностью 1 при условии соответственно большой размерности *K* этого пространства.

Практически это означает, что применение достаточно большого количества скрытых нейронов, реализующих радиальные функции $\varphi_i(\mathbf{x})$, гарантирует решение задачи классификации при построении всего лишь двухслойной сети: скрытый слой должен реализовать вектор $\boldsymbol{\varphi}(\mathbf{x})$, а выходной слой может состоять из единственного линейного нейрона, выполняющего суммирование выходных сигналов от скрытых нейронов с весовыми коэффициентами, заданными вектором **w**.

Простейшая нейронная сеть радиального типа функционирует по принципу многомерной интерполяции, состоящей в отображении *p* различных входных векторов $\mathbf{x}_i \left(i = 1, 2, ..., p\right)$ из входного *N*-мерного пространства во множество из *p* рациональных чисел $d_i \left(i = 1, 2, ..., p\right)$.

Для реализации этого процесса необходимо использовать *p* скрытых нейронов радиального типа и задать такую функцию отображения $F(\mathbf{x})$, для которой выполняется условие интерполяции $F\left(\mathbf{x}_i\right) = d_i$.

Использование *p* скрытых нейронов, соединяемых связями с весами w_i с выходными линейными нейронами, означает формирование выходных сигналов сети путём суммирования взвешенных значений соответствующих базисных функций.

Пример радиальной сети

Рассмотрим радиальную сеть с одним выходом и *p* обучающими парами $\left(\mathbf{x}_i, \mathbf{d}_i\right)$. Примем, что координаты каждого из *p* центров узлов сети определяется одним из векторов $\mathbf{x}_i$, то есть $\mathbf{c}_i = \mathbf{x}_i$. В этом случае взаимосвязь между входными данными и выходными сигналами сети может быть определена системой уравнений, линейных относительно весов $\mathbf{w}_i$, которая в матричной форме имеет вид:

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \dots & \varphi_{1p} \\ \varphi_{21} & \varphi_{22} & \dots & \varphi_{2p} \\ \dots & \dots & \dots & \dots \\ \varphi_{p1} & \varphi_{p2} & \dots & \varphi_{pp} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_p \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \dots \\ d_p \end{bmatrix}$$

где $\varphi_{ji} = \left(\left\| \mathbf{x}_j - \mathbf{x}_i \right\| \right)$ определяет радиальную функцию с центром в точке $\mathbf{x}_i$ с вынужденным вектором $\mathbf{x}_j$.

В обозначениях $\varphi_{ij} = \mathbf{\Phi}$, $\mathbf{w} = \left[w_1, w_2, ..., w_p^T \right]$, $\mathbf{d} = \left[d_1, d_2, ..., d_p \right]^T$

$$\mathbf{\Phi w} = \mathbf{d}.$$

Доказано также, что для ряда радиальных функций в случае $x_1 \neq x_1 \neq \dots \neq x_p$ квадратная интерполяционная матрица является несобственной и при этом неотрицательно определённой. Поэтому существует решение уравнения в виде

$$\mathbf{w} = \mathbf{\Phi}^{-1}\mathbf{d}$$

что позволяет получить вектор весов выходного нейрона сети.

Проблемы, возникающие при таком подходе:

При очень большом количестве обучающих выборок и равном ему количестве радиальных функций количество уравнений начинает превышать число степеней свободы физического процесса, что приводит к тому, что модель начинает адаптироваться под шумы и нерегулярности. В результате обобщающая гиперплоскость не будет гладкой, а обучающие возможности будут слабыми.

Радиальная нейронная сеть

Решение системы уравнений при больших значений *p* (большое число выборок) затруднительно.

В пространстве меньшей размерности ищется субоптимальное решение, которое с достаточной точностью аппроксимирует точное решение.

При ограничении *K* базисными функциями, аппроксимирующее выражение:

$$F(\mathbf{x}) = \sum_{i=1}^{K} \varphi\left(\left\| \mathbf{x} - \mathbf{c}_i \right\| \right)$$

$K < p$, $\mathbf{c}_i$ — множество центров, которые необходимо определить.

Задача аппроксимации заключается в подборе соответствующего количества радиальных функций $\varphi\left(\left\| \mathbf{x} - \mathbf{c}_i \right\| \right)$ и их параметров, а также в подборе весов $w_i \left(i = 1, 2, ..., K \right)$, чтобы решение уравнения было наиболее близким к точному.

Эту задачу можно свести к минимизации целевой функции

$$E = \sum_{i=1}^{p} \left(\sum_{j=1}^{K} w_j \varphi\left(\left\| \mathbf{x}_i - \mathbf{c}_i \right\| \right) - d_i \right)^2$$

K — количество радиальных нейронов, *p* — количество обучающих пар $\left(\mathbf{x}_i, d_i \right)$, $\mathbf{x}_i$ — входной вектор, d_i — соответствующая ему ожидаемая величина.

Введём обозначения

$\mathbf{d} = \left(d_1, d_2, ..., d_p \right)^T$ — вектор ожидаемых значений,

$\mathbf{w} = \left(w_1, w_2, ..., w_K \right)^T$ — вектор весов сети,

Радиальная матрица Грина **G**

$$\mathbf{G} = \begin{bmatrix} \varphi\left(\|\mathbf{x}_1 - \mathbf{c}_1\|\right) & \varphi\left(\|\mathbf{x}_1 - \mathbf{c}_2\|\right) & \dots & \varphi\left(\|\mathbf{x}_1 - \mathbf{c}_K\|\right) \\ \varphi\left(\|\mathbf{x}_2 - \mathbf{c}_1\|\right) & \varphi\left(\|\mathbf{x}_2 - \mathbf{c}_2\|\right) & \dots & \varphi\left(\|\mathbf{x}_2 - \mathbf{c}_K\|\right) \\ \dots & \dots & \dots & \dots \\ \varphi\left(\|\mathbf{x}_p - \mathbf{c}_1\|\right) & \varphi\left(\|\mathbf{x}_p - \mathbf{c}_2\|\right) & \dots & \varphi\left(\|\mathbf{x}_p - \mathbf{c}_K\|\right) \end{bmatrix}$$

Обычно $p \gg K$.

Если параметры радиальных функций известны, то оптимизационная задача сводится к решению системы уравнений, линейных относительно весов **w**

$$\mathbf{G}(\mathbf{w}) = \mathbf{d}$$

С помощью операции псевдоинверсии (из-за прямоугольности) матрицы **G** можно определить вектор весов **w**

$$\mathbf{w} = \mathbf{G}^{+}\mathbf{d}$$

здесь $\mathbf{G}^{+} = \left(\mathbf{G}^T\mathbf{G}\right)^{-1}\mathbf{G}^T$ — псевдоинверсия прямоугольной матрицы **G**.

Если многомерна функция имеет различный масштаб по каждой оси, то на практике уточняют норму масштабирования путём ввода в определение евклидовой метрики весовых коэффициентов в виде матрицы **Q**

$$\|\mathbf{x}\|_Q^2 = (\mathbf{Q}\mathbf{x})^T(\mathbf{Q}\mathbf{x}) = \mathbf{x}^T\mathbf{Q}^T\mathbf{Q}\mathbf{x}$$

Масштабируемая матрица при *N*-мерном векторе **x**

$$\mathbf{Q} = \begin{bmatrix} Q_{11} & Q_{12} & \dots & Q_{1N} \\ Q_{21} & Q_{22} & \dots & Q_{2N} \\ \dots & \dots & \dots & \dots \\ Q_{N1} & Q_{N2} & \dots & Q_{NN} \end{bmatrix}$$

В общем случае, учитывая матрицу корреляции **C**

$$\|\mathbf{x}\|_Q^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}x_i x_j$$

При диагональной масштабируемой матрице

$$\|\mathbf{x}\|_Q^2 = \sum_{i=1}^{N} C_{ii}x_i^2$$

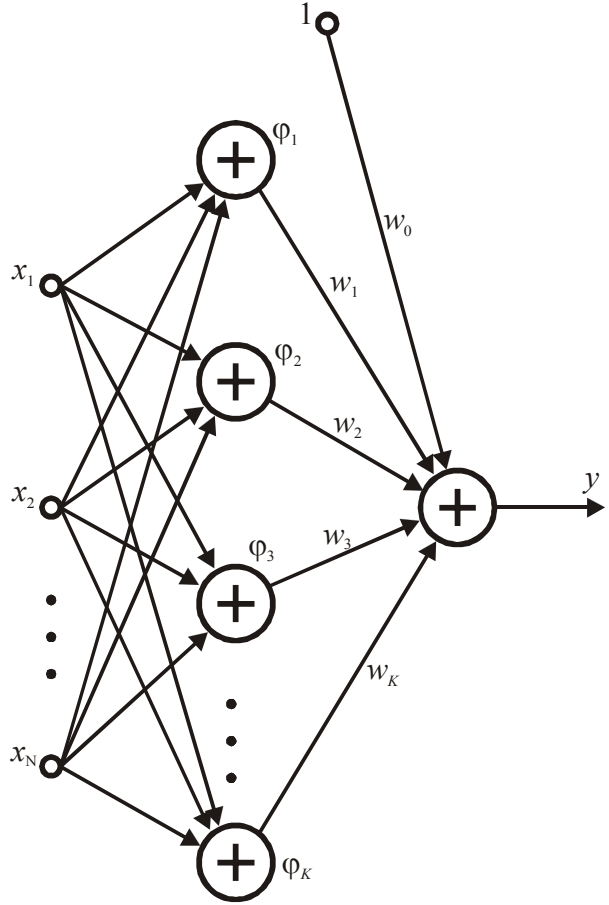
В качестве радиальной функции часто используют функцию Гаусса.

$$\varphi(\mathbf{x}) = \varphi\left(\|\mathbf{x} - \mathbf{c}_i\|\right) = \mathrm{e}^{-\frac{\|\mathbf{x}-\mathbf{c}_i\|^2}{2\sigma_i^2}}$$

Для радиальной функции гауссовской формы с центром в точке c_i и масштабирующей взвешенной матрицы $\mathbf{Q}_i$, связанной с *i*-й базисной функцией, обобщённая форма функции Гаусса

$$\varphi(\mathbf{x}) = \varphi\left(\|\mathbf{x} - \mathbf{c}_i\|_{Q_i}\right) = \mathrm{e}^{-(\mathbf{x}-\mathbf{c}_i)^T\mathbf{Q}_i^T\mathbf{Q}_i(\mathbf{x}-\mathbf{c}_i)} = \mathrm{e}^{-\frac{(\mathbf{x}-\mathbf{c}_i)^T\mathbf{C}_i(\mathbf{x}-\mathbf{c}_i)}{2}}$$

Обобщённая структура радиальной сети RBF



Решение, представляющее аппроксимирующую функцию в многомерном пространстве в виде взвешенной суммы локальных базисных радиальных функций может быть интерпретировано радиальной нейронной сетью (см. рисунок)

На рисунке для упрощения показан один выход.

Это сеть с двухслойной структурой, в которой только скрытый слой выполняет нелинейное отображение, реализуемое нейронами с базисными радиальными функциями.

Выходной нейрон — линеен — делает взвешенное суммирование сигналов, поступающих от скрытого слоя.

Вес w_0 представляет поляризацию, вводящую показатель постоянного смещения функции.

Отличие от сигмоидальной сети:

Сигмоидальная сеть

может иметь различное число слоёв, а выходные нейроны могут быть и нелинейными.

Имеет стандартные функции активации с одним и тем же параметром β

Аргумент сигмоидальных функций $\mathbf{w}^T\mathbf{x}$

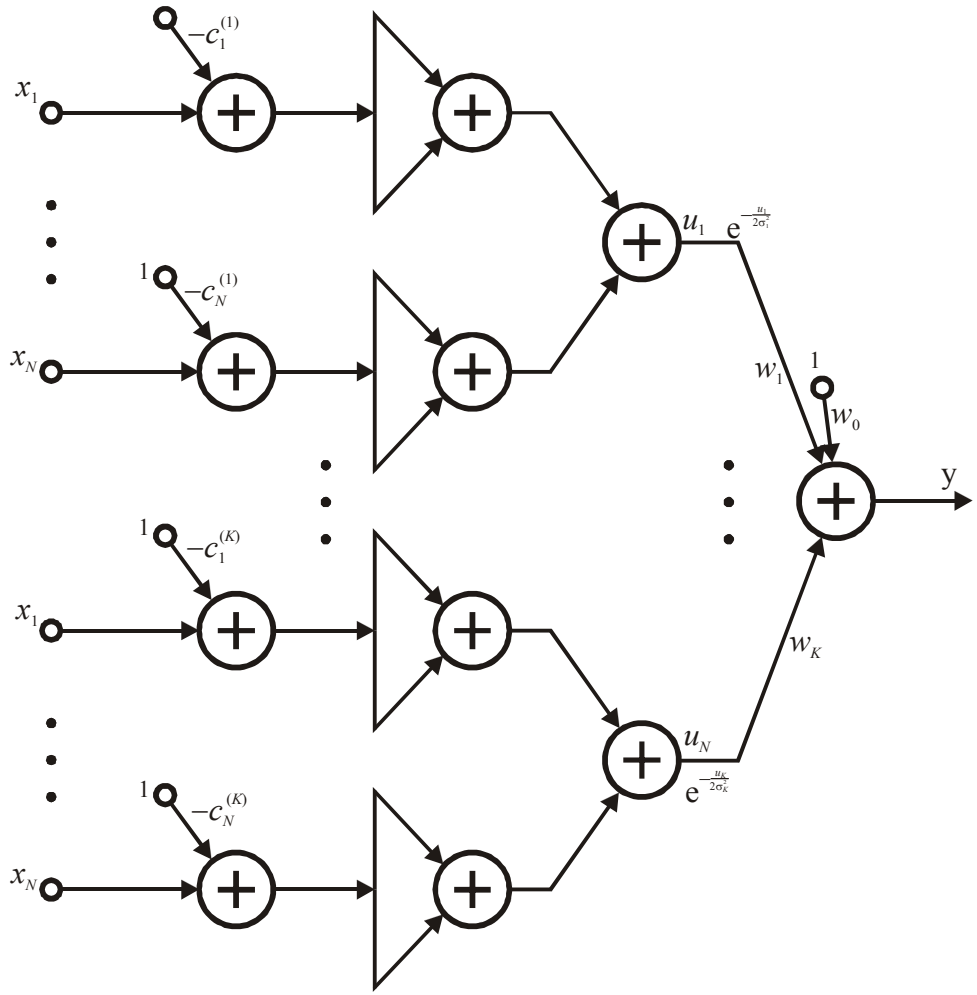
Радиальная сеть

имеет фиксированную структуру с одним скрытым слоем и линейными выходными нейронами

имеет свои значения параметров $\mathbf{c}_i$ и $\mathbf{s}_i$

Аргумент радиальных функций — евклидово расстояние образца $\mathbf{x}$ от центра $\mathbf{c}_i$

Детальное сравнение структур тоже поясняет различия

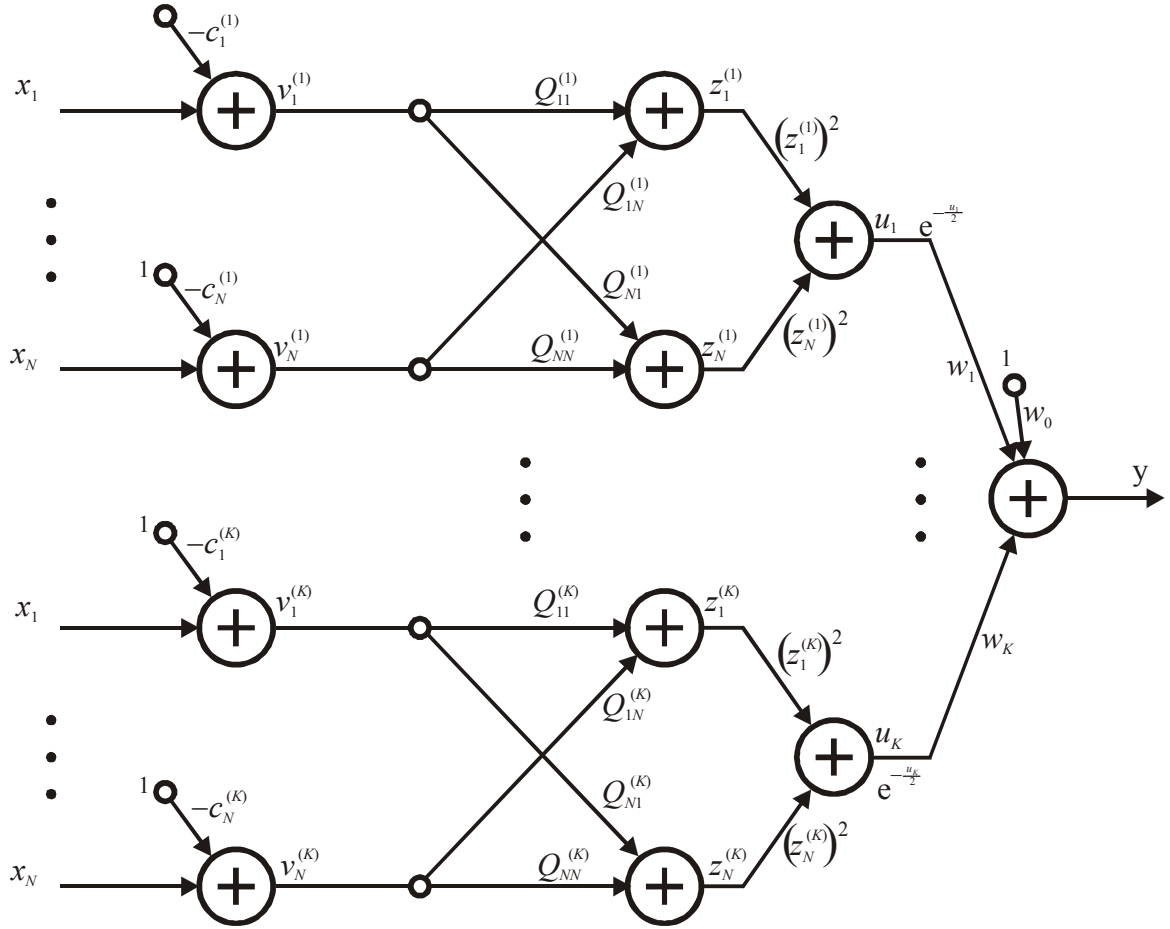


Детальная схема структуры радиальной сети RBF с радиальной функцией

$$\varphi(\mathbf{x}) = \varphi\left(\left\|\mathbf{x}-\mathbf{c}_i\right\|\right) = e^{-\frac{\left\|\mathbf{x}-\mathbf{c}_i\right\|^2}{2\sigma_i^2}}$$

Первый слой составляют нелинейные радиальные функции, параметры которых (центры $\mathbf{c}_i$ и коэффициенты $\mathbf{s}_i$) уточняются в процессе обучения.

Первый слой не содержит линейных весов.



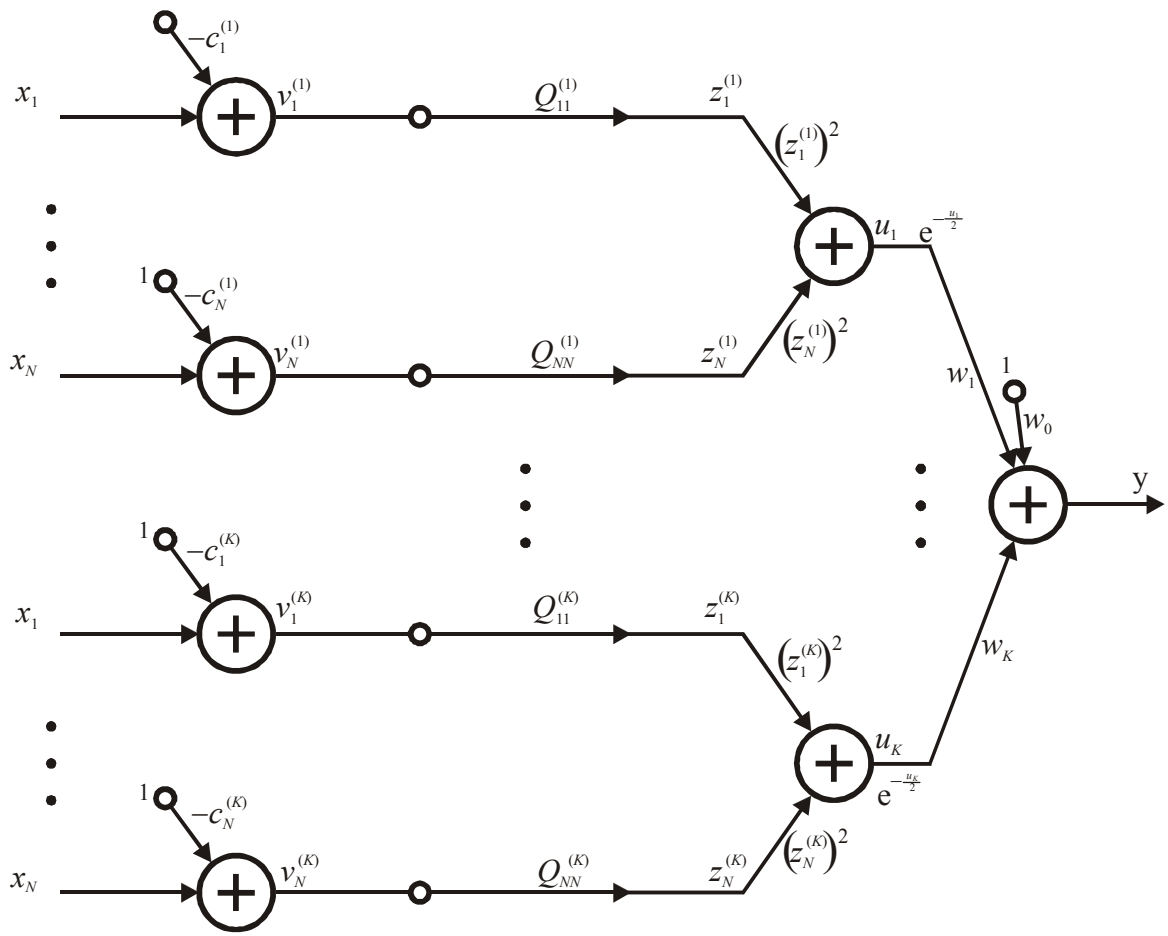
Детальная схема структуры радиальной сети HRBF (Hyper Radial Basis Function) с масштабируемой матрицей **Q** произвольного вида
Реализует масштабируемую радиальную функцию

$$\varphi(\mathbf{x}) = \varphi\left(\left\|\mathbf{x}-\mathbf{c}_i\right\|_{Q_i}\right) = e^{-(\mathbf{x}-\mathbf{c}_i)^T \mathbf{Q}_i^T \mathbf{Q}_i (\mathbf{x}-\mathbf{c}_i)} = e^{-\frac{(\mathbf{x}-\mathbf{c}_i)^T \mathbf{C}_i (\mathbf{x}-\mathbf{c}_i)}{2}}$$

Радиальный нейрон в своей структуре содержит сумматоры сигналов, аналогичные применяемым в сигмоидальной сети и показательные функции активации с параметрами, подлежащими уточнению в процессе обучения.

Веса $Q_{ij}^{(k)}$ k-го радиального нейрона скрытого слоя — элементы матрицы $\mathbf{Q}^{(k)}$

Они являются масштабирующей системой, вводят дополнительные степени свободы, что позволяет лучше приблизить выходной сигнал сети $y = f(\mathbf{x})$ к ожидаемой функции $d(\mathrm{x})$.



Часто масштабирующая матрица $\mathbf{Q}^{(k)}$ имеет диагональную форму, в которой только элементы $Q_{ii}^{(k)}$ принимают ненулевые значения.
В такой системе отсутствует круговое перемешивание сигналов, соответствующих различным компонентам вектора **x**, а элемент $Q_{ii}^{(k)}$ рассматривается как индивидуальный масштабирующий коэффициент для *i*-го компонента вектора **x** *k*-го нейрона.
На рисунке представлена детальная схема структуры радиальной сети HRBF с диагональными масштабируемыми матрицами $\mathbf{Q}^{(k)}$.
Роль коэффициентов σ_i^2 выполняют элементы матрицы **Q**, которые уточняются в процессе обучения.

Методы обучения радиальных нейронных сетей

В большинстве случаев $p \gg K$, поэтому процесс обучения сети RBF с учётом выбранного типа радиальной базисной функции сводится:

1. к подбору весов c_i и параметров σ_i формы базисных функций
2. у подбору весов нейронов выходного слоя

При этом вектор весов **w** может быть определён за 1 шаг псевдоинверсией матрицы **G**

$$\mathbf{w} = \mathbf{G}^{+}\mathbf{d}$$

Матрица **G,** имеющая *p* строк и *K* столбцов, представляет реакции нейронов скрытого слоя на очередные возбуждения векторами $\mathbf{x}_i \left(i = 1, 2, ..., p\right)$

Псевдоинверсия матрицы **G** рассчитывается с использованием разложения по собственным значениям

$$\mathbf{G} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}$$

U $\left(p \times p\right)$ и **V** $\left(K \times K\right)$ ортогональны. **S** $\left(p \times K\right)$ — псевдодиагональная матрица и $s_1 \geqslant s_2 \geqslant ... \geqslant s_K \geqslant 0$.

Если *r* первых элементов s_i имеют значимую величину (пренебрегая остальными), то получим редуцированные матрицы

$$\mathbf{U}_r = \left[\mathbf{u}_1\mathbf{u}_2...\mathbf{u}_r\right]$$

$$\mathbf{V}_r = \left[\mathbf{v}_1\mathbf{v}_2...\mathbf{v}_r\right]$$

А матрица $\mathbf{S}_r = \mathrm{diag}\left[\mathbf{s}_1\mathbf{s}_2...\mathbf{s}_r\right]$ становится полностью диагональной (квадратной)

То есть $\mathbf{G} \cong \mathbf{U}_r\mathbf{S}_r\mathbf{V}_r^{\mathrm{T}}$

Псевдообратная к **G** матрица

$$\mathbf{G}^{+} \cong \mathbf{V}_r\mathbf{S}_r^{-1}\mathbf{U}_r^{\mathrm{T}}$$

здесь $\mathbf{S}_r^{-1} = \left[\frac{1}{s_1}, \frac{1}{s_2}, ..., \frac{1}{s_r}\right]$, а вектор весов сети $\mathbf{w} = \mathbf{V}_r\mathbf{S}^{-1}\mathbf{U}_r^{T}\mathbf{d}$.

Выходные веса сети подбираются за 1 шаг.

Главная проблема обучения радиальных сетей: подбор параметров нелинейных радиальных функций, особенно центров c_i.

Можно подбирать на основе равномерного или гауссовского распределения, всё зависит от специфики задачи.