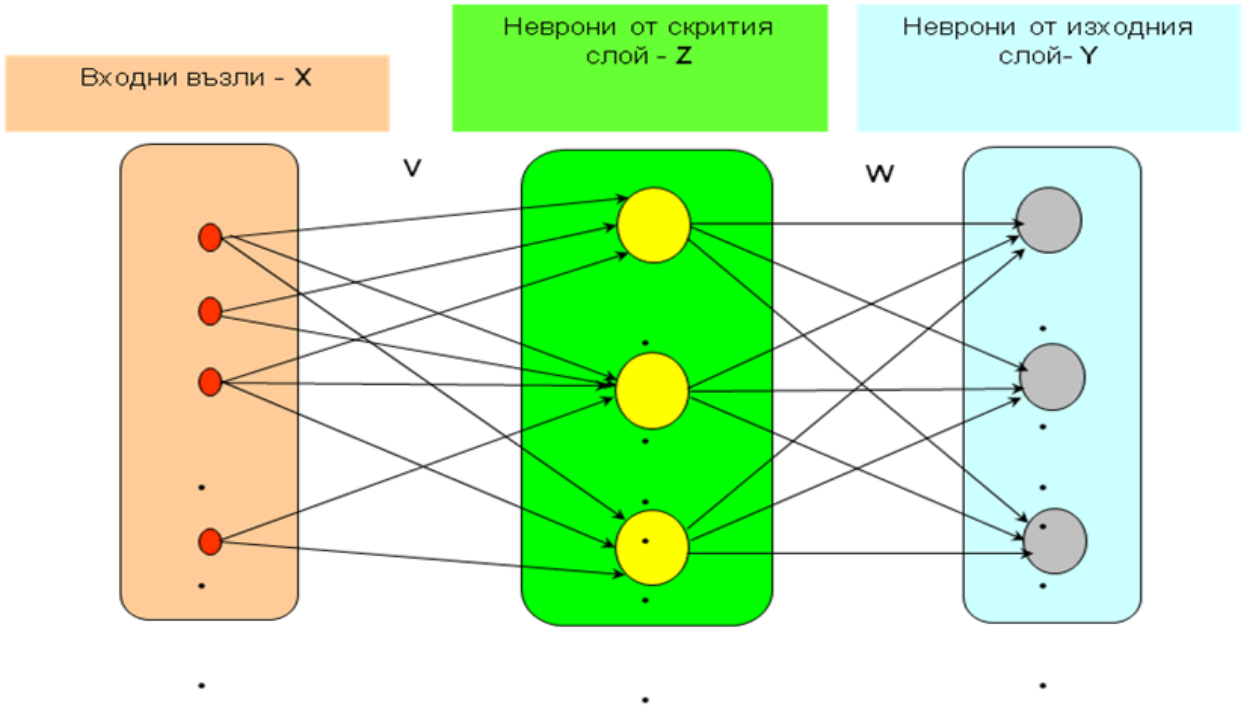
Невронни Мрежи

проф. В. Младенов

7. Многослойни мрежи с право предаване на сигнала и обратно разпространение на грешката. Подходи за подобряване на алгоритъма с обратното разпространение на грешката. Преимущества и недостатъци. Невронни мрежи с радиални базисни функции (Radial Basis Function Networks).

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

- Многослойните мрежи биват двуслойни, трислойни и т.н. На практика най-често използваните мрежи са двуслойни. Архитектурата на двуслойна невронна мрежа с еднопосочно предаване на сигнала е показана на фигурата:



Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Мрежата се състои от

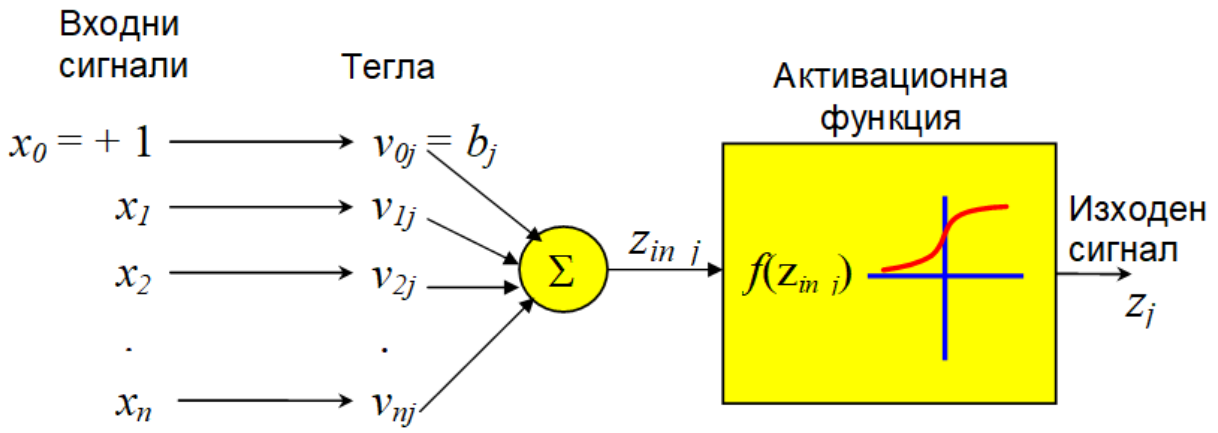
- един входен слой възли, посредством които се подават сигналите $X=[x_1\ x_2\ \dots\ x_n]^T$ към мрежата
- от един изходен слой неврони, векторът от изходните сигнали на които $Y=[y_1\ y_2\ \dots\ y_m]^T$ е изходният вектор на мрежата
- един скрит слой неврони, изходните сигнали на които са $Z=[z_1\ z_2\ \dots\ z_p]^T$.

Ако мрежата е трислойна, четирислойна и т.н., броят на скритите слоеве е съответно два, три и т.н. Без да се намалява общността на разглеждането, следващите резултати ще бъдат представяни за невронни мрежи с един скрит слой.

Входният сигнал на активационната функция на даден неврон (j-ти) от скрития слой е претеглената сума $z_{in\_j} = v_{0j} + \sum_{i=1}^{n} v_{ij} x_i$. Изходният сигнал на неврона е $z_j = f(z_{in\_j})$, където f е активационната функция.

Обикновено активационните функции на невроните от скрития слой са сигмоидални функции, а в зависимост от решаваните задачи, активационните функции на невроните от изходния слой могат да бъдат сигмоидални, линейни или др.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката



Обикновено теглата на връзките между входния и скрития слой и между скрития и изходния слой формират матриците V и W

$$\mathbf{V} = \begin{bmatrix} v_{01} & v_{02} & \cdots & v_{0p} \\ v_{11} & v_{12} & \cdots & v_{1p} \\ . & \cdots & \cdots & . \\ v_{n1} & v_{n2} & \cdots & v_{np} \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} w_{01} & w_{02} & \cdots & w_{0m} \\ w_{11} & w_{12} & \cdots & w_{1m} \\ . & \cdots & \cdots & . \\ w_{p1} & w_{p2} & \cdots & w_{pm} \end{bmatrix}.$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

При обучението на невронната мрежа е зададена обучаваща извадка от P на брой желани входно-изходни набори от данни за мрежата $\mathbf{x}(j):\mathbf{d}(j)$, където

$\mathbf{x}(j)=[x_1(j)\ x_2(j)\ \dots\ x_n(j)]^T$, $\mathbf{d}(j)=[d_1(j)\ d_2(j)\ \dots\ d_m(j)]^T$, $j=1,2,\dots,P$,

като числото в скобата показва номера на желаната входно-изходна двойка от обучаващата извадка, а с долен индекс се означават компонентите на съответния вектор.

В случая, на всеки вектор с номер j от входни сигнали $\mathbf{x}(j)$ съответства желан изходен вектор за мрежата $\mathbf{d}(j)$. Целта на обучението е да бъдат настроени теглата v_{ij}, $i=1,2,\dots,n$, $j=1,2,\dots,p$ между входния и скритията слой и w_{jk}, $j=1,2,\dots,p$, $k=1,2,\dots,m$ между скритията и изходния слой, така че за всеки от елементите на обучаващата извадка, изходният вектор на мрежата максимално да се доближава до желания изходен вектор.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

При изчисляването на изходните сигнали на мрежата y_k, $k=1,2,\ldots,m$ се използват следните зависимости:

$$y_{in\_k} = w_{0k} + \sum_{j=1}^{p} w_{jk} z_j ; \quad y_k = f\left(w_{0k} + \sum_{j=1}^{p} w_{jk} z_j \right); \quad k = 1,2,\ldots,m$$

$$z_{in\_j} = v_{0j} + \sum_{i=1}^{n} v_{ij} x_i ; \quad z_j = f\left(v_{0j} + \sum_{i=1}^{n} v_{ij} x_i \right); \quad j = 1,2,\ldots,p$$

Подходът, който ще бъде използван при настройката на теглата на мрежата, е градиентен подход за минимизиране на грешката между желаните и изчислените изходни сигнали на мрежата. Тъй като целевата функция (функцията на грешката), която ще бъде минимизирана, ще бъде формирана като сума от квадратите на грешките за всеки от изходните неврони, методът който се използва е от типа *методи на най-малките квадрати*.

Както и при адаптивния линеен елемент, методът на най-малките квадрати може да бъде приложен в последователен или пакетен режим.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Нека на стъпка t средноквадратичната моментна грешка (целевата функция) за всички изходни неврони е

$$E = E(t) = 0.5\sum_{k=1}^{m}\left(d_k(t) - y_k(t)\right)^2 .$$

Трябва да се подчертае, че тъй като $y_k(t)$, $k=1,2,\ldots,m$ зависят от теглата v_{ij}, $i=0,1,2,\ldots,n$, $j=1,2,\ldots,p$ между входния и скрития слой и теглата w_{jk}, $j=0,1,2,\ldots,p$, $k=1,2,\ldots,m$ *между* скрития и изходния слой, средноквадратичната моментна грешка е функция от всички тегла на мрежата $E = E(t) = E(\mathbf{V},\mathbf{W})$ и за намиране на нейния минимум трябва да се правят стъпки в посоката, обратна на градиента на функциата (т.к. градиентът определя посоката на най-бързото нарастване на функцията).

При намирането на различните компоненти на градиента, за индекси на теглата ще бъдат използвани главни и малки букви. Главните букви ще бъдат използвани за индекси на теглата, които ще се настройват, а малките букви, ще бъдат използвани за индекси при сумиранията.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Частната производна на средноквадратичната грешка по отношение на всяко от теглата между скрития и изходния слой (съответният компонент на градиента) е

$$\frac{\partial E}{\partial w_{JK}} = \frac{\partial}{\partial w_{JK}}(0.5\sum_{k=1}^{m}(d_k - y_k)^2) = \frac{\partial}{\partial w_{JK}}(0.5(d_K - y_K)^2)$$

$$= (d_K - y_K)\frac{\partial}{\partial w_{JK}}(-y_K) = -(d_K - y_K)\frac{\partial}{\partial w_{JK}} f(y_{in\_K})$$

$$= -(d_K - y_K) f'(y_{in\_K})\frac{\partial}{\partial w_{JK}}(y_{in\_K})$$

$$= -(d_K - y_K) f'(y_{in\_K}) z_J \ .$$

Последното равенство следва от това, че $y_{in\_K} = w_{0K} + \sum_{j=1}^{p} w_{jK} z_j$ и само $w_{JK} z_J$ включва z_J.

Нека с δ_K е означена мащабираната с $f'(y_{in\_K})$ грешка на K-ти неврон от изходения слой

$$\delta_K = (d_K - y_K) f'(y_{in\_K}) \ .$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Тогава, съгласно градиентния подход, донастройката Δw_{JK} на теглото w_{JK} трябва да бъде

$$\Delta w_{JK} = \alpha.\left(-\frac{\partial E}{\partial w_{JK}}\right) = \alpha.\delta_K.z_J\,, \quad K = 1,2,...,m\,, J = 1,2,...,p\,,$$

където α е скоростта на обучението и определя големината на стъпката, която се прави в посока обратна на градиента на целевата функция.

Ако $w_{JK}(t)$ е стойността на теглото w_{JK} на стъпка t, то новата му стойност за стъпка $t+1$ се изчислява по формулата

$$w_{JK}(t+1) = w_{JK}(t) + \Delta w_{JK} = w_{JK}(t) + \alpha.\delta_K.z_J\,, \quad K = 1,2,...,m\,, J = 0,1,2,...,p\,,$$

където мащабираната грешка на K-тия изходен неврон се изчислява с формулата

$$\delta_K = (d_K - y_K) f'(y_{in\_K})$$

Векторът на градиента определя посоката на промяна на вектора с теглата, а α определя големината на стъпката при промяната. Ако α е с голяма стойност, процедурата по настройка на теглата може да стане разходяща, а ако α има малка стойност, процедурата ще е много бавна.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Частната производна на средноквадратичната грешка по отношение на теглата между входния и скрития (съответният компонент на градиента) е

$$\frac{\partial E}{\partial v_{IJ}} = \frac{\partial}{\partial v_{IJ}}(0.5\sum_{k=1}^{m}(d_k - y_k)^2)$$

$$= -\sum_{k=1}^{m}((d_k - y_k)\frac{\partial}{\partial v_{IJ}} y_k) \qquad \text{(т.к. } y_k \text{ включва } v_{IJ}\text{, понеже е свързан с } z_J\text{)}$$

$$= -\sum_{k=1}^{m}((d_k - y_k)\frac{\partial}{\partial v_{IJ}} f(y_{in\_k}))$$

$$= -\sum_{k=1}^{m}((d_k - y_k)f'(y_{in\_k})\frac{\partial}{\partial v_{IJ}}(y_{in\_k}))$$

$$= -\sum_{k=1}^{m}(\delta_k \frac{\partial}{\partial v_{IJ}}(y_{in\_k})) \qquad \text{(т.к. } \delta_k = (d_k - y_k)f'(y_{in\_k})\text{)}$$

$$= -\sum_{k=1}^{m}(\delta_k w_{Jk} \frac{\partial}{\partial v_{IJ}} z_J)\,.$$

Последното равенство от формула $\Delta w_{JK} = \alpha.\left(-\frac{\partial E}{\partial w_{JK}}\right) = \alpha.\delta_K.z_J\,, \quad K = 1, 2, \ldots, m\,, J = 1, 2, \ldots, p$ следва от това, че само един от компонентите в $y_{in\_k} = w_{0k} + \sum_{j=1}^{p} w_{jk} z_j$, а именно $w_{Jk}z_J$ включва v_{IJ} чрез z_J. Оттук

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$\frac{\partial E}{\partial v_{IJ}} = -\sum_{k=1}^{m}(\delta_k w_{Jk}\frac{\partial}{\partial v_{IJ}} z_J)$$

$$= -\sum_{k=1}^{m}(\delta_k w_{Jk} f'(z_{in\_J})\frac{\partial}{\partial v_{IJ}}(z_{in\_J}))$$

$$= -\sum_{k=1}^{m}(\delta_k w_{Jk} f'(z_{in\_J})\cdot x_I) \quad \text{(само } v_{IJ}x_I \text{ от } z_{in\_J} \text{ включва } v_{IJ}\text{)}$$

$$= -f'(z_{in\_J})\cdot x_I \cdot \sum_{k=1}^{m}\delta_k w_{Jk} \quad \text{(защото } x_I \text{ и } z_{in\_J} \text{ не зависят от } k\text{)}$$

$$= -f'(z_{in\_J}) x_I \cdot \delta_{in\_J} \ .$$

Нека с δ_J е означена мащабираната с $f'(z_{in\_J})$ грешка, свързана с изходния сигнал на J-ти неврон от скрития слой

$$\delta_J = \delta_{in\_J} . f'(z_{in\_J}) = \left(\sum_{k=1}^{m}\delta_k w_{Jk}\right) . f'(z_{in\_J}) \ .$$

Тогава, съгласно градиентния подход, донастройката Δv_{IJ} на теглото v_{IJ} трябва да бъде

$$\Delta v_{IJ} = \alpha \cdot (-\frac{\partial E}{\partial v_{IJ}}) = \alpha \cdot \delta_J \cdot x_I \ ,$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

където α е скоростта на обучението и тя определя големината на стъпката, която се прави в посока обратна на градиента на целевата функция.

Ако $v_{IJ}(t)$ е стойността на теглото v_{IJ} на стъпка t, то новата му стойност за стъпка $t+1$ се изчислява по формулата

$$v_{IJ}(t+1) = v_{IJ}(t) + \Delta v_{IJ} = v_{IJ}(t) + \alpha.\delta_J.x_I\,, \quad J = 1,2,...,p, \quad I = 0,1,2,...,n\,,$$

където мащабираната с $f'(z_{in\_J})$ грешка, свързана с изходния сигнал на J-ти неврон от скрития слой, се изчислява с формулата $\delta_J = \delta_{in\_J}.f'(z_{in\_J}) = \left(\sum_{k=1}^{m}\delta_k w_{Jk}\right).f'(z_{in\_J})$.

Настройката на теглата при разгледания метод на най-малките квадрати за многослойна невронна мрежа се нарича *обобщено делта правило*, т.к. както и при адаптивния линеен неврон, новата стойност на теглата на всяка стъпка е пропорционална на грешката (делта), свързана с неврона от съответния слой (изходен или скрит). Както и при адаптивния линеен неврон, грешката свързана с невроните от изходния слой директно се определя като разлика от желаната и изчислената стойност на изходния сигнал на неврона.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

В случая грешката $\delta_K = (d_K - y_K) f'(y_{in\_K})$ се мащабира със стойността на първата производна на активационната функция. За невроните от скрития слой обаче, грешката между желана и изчислена стойност не може да се изчисли директно, както при невроните от изходния слой, и това е една от причините правилото за обучение на многослойни мрежи, базирано на градиентен подход за редуциране стойността на целевата функция, да не бъде разработено паралелно с делта правилото при адаптивния линеен елемент. В резултат на направените разглеждания е получена формулата $\delta_J = \delta_{in\_J} . f'(z_{in\_J}) = \left(\sum_{k=1}^{m} \delta_k w_{Jk} \right) . f'(z_{in\_J})$ за определяне на мащабираната грешка. В тази формула участва компонентът $\sum_{k=1}^{m} \delta_k w_{Jk}$, който по същество представлява претеглена с теглата свързани с J-тия неврон от скрития слой, сума от грешките свързани с всички изходни неврони.

По този начин в стойностите на грешките, свързани с невроните от скрития слой, участват всички грешки свързани с невроните от изходния слой.

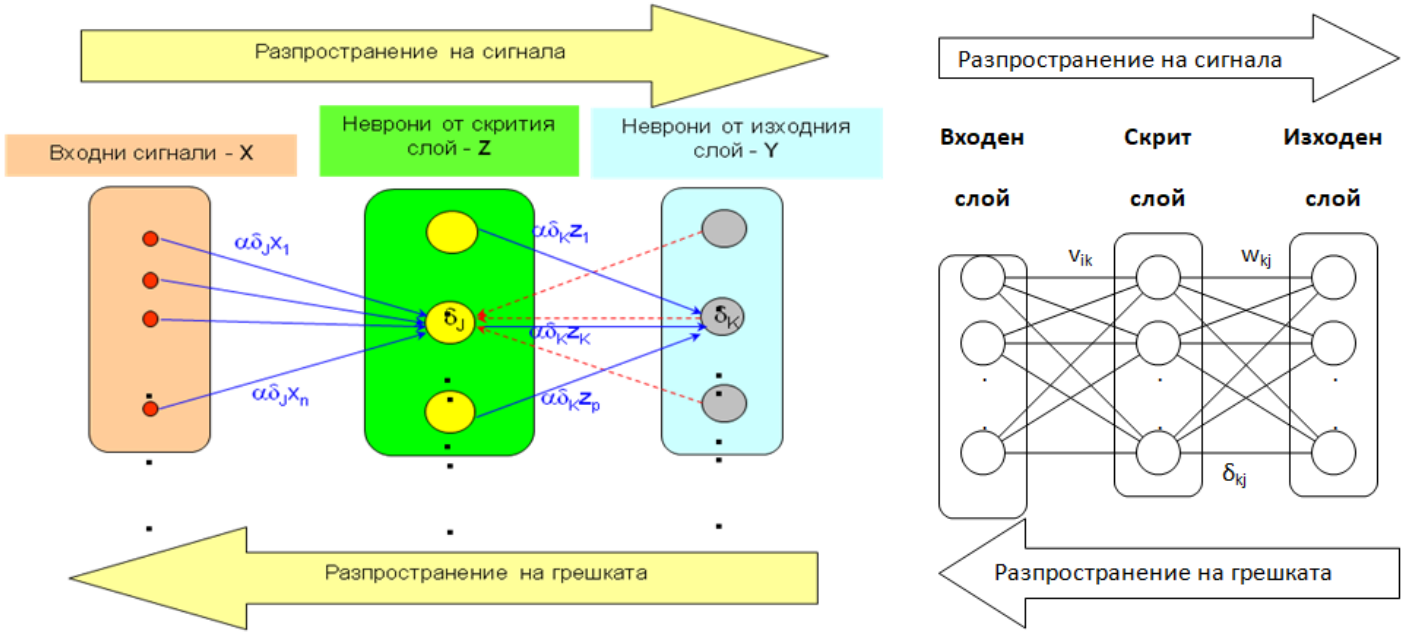
Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

В този смисъл грешките, свързани с невроните от изходния слой се *разпространяват в обратна посока* към скрития слой и на тяхна база се определят грешките, свързани с невроните от скритията слой. Този резултат е довел и до името, с което е популярен разглежданият метод за настройка на теглата – *метод с обратно разпространение на грешката*. Ако невронната мрежа има повече скрити слоеве за определяне на грешките, свързани с невроните от даден скрит слой, се използват грешките свързани с всички неврони от следващия слой.

Алгоритъмът за прилагане на разглеждания метод включва два етапа за всеки елемент от обучаващата извадка. При първия етап има т.нар. разпространение на сигнала от входа на мрежата към изхода и се изчисляват стойностите на изходите на невроните от скритията и изходния слой. При втория етап се изчисляват грешките. Първо се изчисляват грешките свързани с невроните от изходния слой, които се базират на разликата между желана и изчислена стойност на всеки от изходите.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

На база на тези грешки, се изчисляват грешките, свързани с невроните от предишния скрит слой и т.н. слой по слой, т.е. при този етап има обратно разпространение на грешките от изходния към предходните слоеве, както е показано на фигурата:



На база на тези грешки се изчисляват и донастройките на теглата, свързани със съответните неврони.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Алгоритъмът за прилагане на метода с обратно разпространение на грешката се състои от следните стъпки:

Стъпка 0: Инициализиране (задаване на начални стойности) на теглата w_{jk} и v_{ij}, избор на стойност на скоростта на обучение α, дефиниране на критерий за спиране (обикновено това е желана стойност на целевата функция, която трябва да бъде достигната при минимизацията) и инициализиране на брояча на стъпките $t=1$.

Стъпка 1: Докато условието за спиране не е удоволетворено увеличи брояча на стъпките и *прави стъпки 2 – 9.*

Стъпка 2: За всеки елемент от обуч. извадка $\mathbf{x}(t):\mathbf{d}(t)$ *прави стъпки 3 – 8.*

*/*разпространение на сигнала (изч. на изходните сигнали на невроните)*/*

Стъпка 3: Прилагане на входен вектор **x**.

Стъпка 4: Изчисляване изходните сигнали на невроните от скрития слой

$$y_{in\_k} = w_{0k} + \sum_{j=1}^{p} w_{jk} z_j ; \quad y_k = f\left(w_{0k} + \sum_{j=1}^{p} w_{jk} z_j \right); \quad k = 1,2,...,m .$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Стъпка 5: Изчисляване на изходните сигнали на невроните от изходния слой

$$z_{in\_j} = v_{0j} + \sum_{i=1}^{n} v_{ij} x_i; \quad z_j = f\left(v_{0j} + \sum_{i=1}^{n} v_{ij} x_i \right); \quad j = 1, 2, ..., p \ .$$

/*обратно разпространение на грешката*/

Стъпка 6: За всеки неврон K от изходния слой се изчислява

$$\delta_K = (d_K - y_K) f'(y_{in\_K}) \qquad /* \text{ грешка } */$$

$$\Delta w_{JK} = \alpha . \left(-\frac{\partial E}{\partial w_{JK}} \right) = \alpha . \delta_K . z_J \qquad /* \text{ донастройка на теглата } */$$

Стъпка 7: За всеки неврон J от скрития слой се изчислява

$$\delta_J = \delta_{in\_J} . f'(z_{in\_J}) = \left(\sum_{k=1}^{m} \delta_k w_{Jk} \right) . f'(z_{in\_J}) \ , \quad /* \text{ обр. разпр. на грешк. } */$$

$$\Delta v_{IJ} = \alpha \cdot (-\frac{\partial E}{\partial v_{IJ}}) = \alpha \cdot \delta_J \cdot x_I \ . \qquad /* \text{ донастройка на теглата } */$$

Стъпка 8: Изчисляване на новите стойности на теглата

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$w_{JK}(t+1) = w_{JK}(t) + \Delta w_{JK} = w_{JK}(t) + \alpha.\delta_K.z_J \quad \text{за всяко } J,\ K.$$

$$v_{IJ}(t+1) = v_{IJ}(t) + \Delta v_{IJ} = v_{IJ}(t) + \alpha.\delta_J.x_I \quad \text{за всяко } I,\ J.$$

Стъпка 9: Проверка на условието за спиране.

Трябва да се отбележи, че на всяка стъпка t, се взема различен елемент $\mathbf{x}(t):\mathbf{d}(t)$ от обучаващата извадка. След изчерпването на елементите, алгоритъмът продължава да бъде изпълняван със същите елементи отново и отново, докато критерият за спиране бъде удоволетворен. В този смисъл обучението на мрежата може да трае няколко *епоха*. Както беше отбелязано, като критерий за спиране на алгоритъма може да бъде използвана дадена желана стойност на целевата функция. В много случаи обаче, по различни причини, тази стойност не може да бъде достигната. Тогава се налага да бъде въведен и допълнителен критерий с цел осигуряване на спиране на работата на алгоритъма. Обкновенно това е максимален брой епохи за достигане на желаната стойност на целевата функция.

При пакетния подход за прилагане на *обобщеното делта правило* при настройката на теглата, тяхната промяна (донастройка) се извършва след

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

акумулирането на компонентите от грешките, дължащи се на всички елементи от обучаващата извадка. Както и при линейния адаптивен елемент, при този подход, донастройката на вектора с теглата се изчислява в края на всеки епох.

Както и при адаптивния линеен елемент, сходимостта на разгледания алгоритъм на най-малките квадрати може да бъде ускорена с адаптивна промяна на скоростта на обучение α.

След завършване на фазата на обучение, невронната мрежа е готова за използване, при решаване на задачата, за която е обучена. Във фазата на използване се извършва реалната обработка на сигналите като се задават необходимият брой входни вектори и на тяхна база се изчисляват съответните изходни вектори. За проверка на генерализиращите свойства на мрежата е необходимо да бъде тествана коректната и работа при използване на нови данни различни от тези, с които е обучена.

Доказано е, че с многослойни невронни мрежи, базирани на метода с обратно разпространение на грешката, могат да бъдат представени (апроксимирани)

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

всички L^2 функции. Същевременно при решаване на различни апроксимационни задачи, генерализиращите свойства на мрежите са добри, стига обучаващата извадка да е достатъчна, а данните за тестване, които са различни от тези за обучение, да са в диапазона, от който са подбирани данните за обучение. Многослойни невронни мрежи, базирани на метода с обратно разпространение на грешката са типичен представител на мрежите за обучение с учител. Те работят на принципа на черната кутия и при достатъчно данни за обучение, резултатите при използване са много добри. В случая не се изисква познаване на същността на проблема, който се апроксимира с невронната мрежа, а само достатъчно входно-изходни данни. Същевременно тези невронни мрежи са робастни по отношение на шум в данните и в стойностите на теглата на мрежата, при липсващи данни и др. Не на последно място трябва да се отчете, че алгоритъмът на метода с обратно разпространение на грешката се прилага много лесно.

Основен недостатък на многослойните невронни мрежи с обратно разпространение на грешката е дългото време за обучение до достигане на

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

желаните стойности на целевата функция. При по-сложни задачи обучението трае хиляди епохи. Тъй като невронната мрежа работи на принципа на черната кутия много е трудно да бъде въведена и използвана априорна информация. В случая невроните от скрития слой нямат семантично значение. Същевременно подходът с намаляване на градиента на целевата функция гарантира намаляване на грешката до локален минимум на целевата функция. При тези случаи трябва да се опита да бъде решен проблемът с невронна мрежа с друг брой неврони от скрития слой (промяна на целевата функция, поради промяна на броя на теглата, които се настройват), или да се приложи алгоритъмът с друг избор на началните стойности на теглата. Съществуват и редица модификации на разгледания алгоритъм с обратно разпространение на грешката, при които избягването от локалния минимум на целевата функция става чрез случайни смущения, използване на т.нар. моменти или други специфични техники от изчислителната математика. В някои специфични случаи например, въпреки че грешката при обучението е редуцирана до нула, генерализацията не е гарантирана. Това обикновенно са случаите при избор на относително много неврони от скрития слой по отношение на големината на обучаващата извадка.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Получава се т.нар ефект на *претрениране* (*over-fitting*, *over-training*), т.е. грешката при обучение намалява до нула, а при тестване на мрежата с други данни различни от тези, с които е обучена, грешките са относително големи.

Предимства на мрежите с обратно разпространение на грешката:

- Могат да представят всяка интегрируема функция.
- Нуждаят се само от подходяща обучаваща извадка.
- Не изискват предварително познаване на проблема (т.е. способни са да решават лошо структурирани задачи).
- Устойчиви са на шум и липса на данни в обучаващата извадка.
- Лесно реализиране на алгоритъма за управление.

Недостатъци на мрежите с обратно разпространение на грешката:

- Обучението може да бъде продължително.
- При обучението на мрежата е възможно спиране на алгоритъма в локален минимум вместо достигането до глобалния оптимум.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

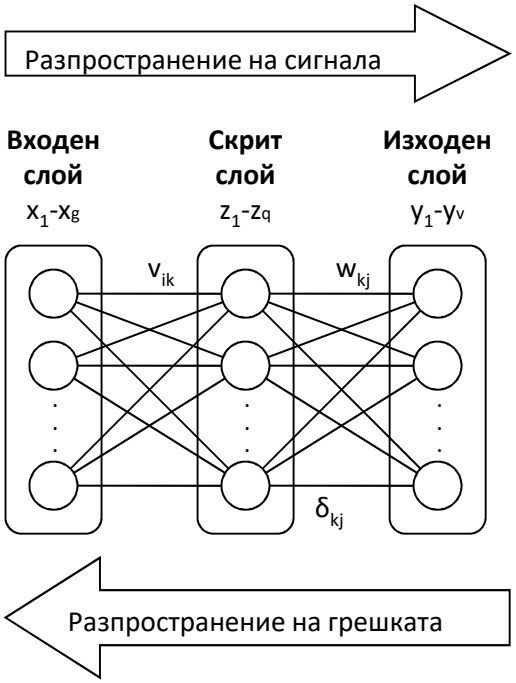
Броят на невроните от скрития слой е много важен при използване на многослойните невронни мрежи с обратно разпространение на грешката. Обикновенно броят на невроните се избира от емпирични съображения и ако с даден брой неврони мрежата не може да бъде обучена (не може да бъде достигната желана стойност на функцията на грешката) за обозрим брой епохи се правят проби с друг брой неврони от скрития слой, до успешното обучение на мрежата. При относително малък брой неврони от скрития слой, почти е невъзможно достигане на желани стойности на целевата функция, при сложни проблеми и голям обем на обучаваюата извадка. От друга страна, относително големият брой неврони от скрития слой може да доведе до претрениране на мрежата. Броят на невроните от скрития слой зависи от броя на входните сигнали на мрежата, броя на изходните неврони, обема на обучаващата извадка, наличието на шум в данните, сложността на решаваната задача, конкретния алгоритъм на приложение на метода с обратното разпространение на грешката и др.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Важен фактор в работата на невронната мрежа при това положение е броят на невроните в скрития слой. Ако той е прекалено малък, мрежата трудно ще се справя със сложни данни. Ако е прекалено голям времето за обучение ще стане твърде дълго и при малка обучаваща извадка е невъзможно минимизирането на грешката.В тази връзка се използват различни емпирични правила за избор на броя на неврони от скрития слой. Едно от най-популярните емпирични правила е правилото на Оja.

Едно от най-популярните общоприети правила за броя на невроните в скрития слой е правилото на Оя: $A = \frac{D}{5(B+C)}$, където A е броят на невроните в скрития слой, B - размер на входния слой, C - брой на невроните в изходния слой, D - размер на обучаващата извадка.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката



Алгоритъмът с обратно разпространение на грешката е известен още като обобщено делта-правило. В основата си той представлява градиентен метод, целящ минимизиране на грешката на изхода на мрежата.

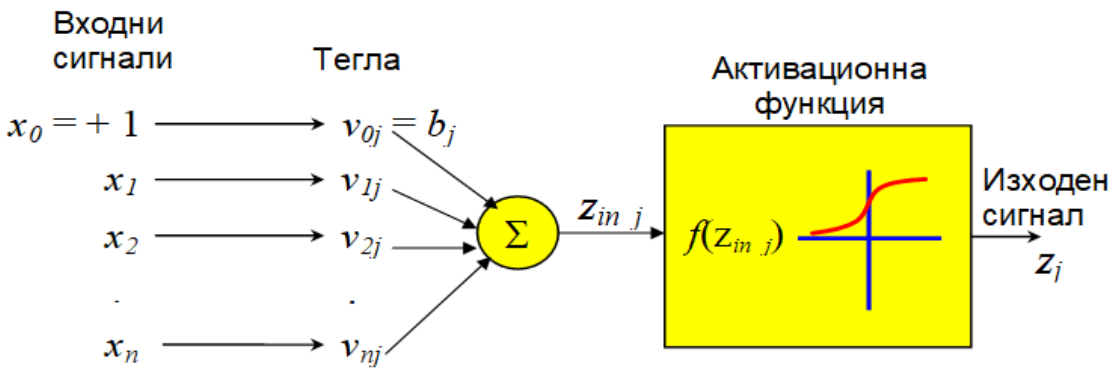
Обучението се състои от три основни етапа – право разпространение на входния сигнал през мрежата, изчисляване и обратно разпространение на грешката и настройка на теглата.

След стадия за обучение работата на мрежата включва само право разпространение на сигнала. Въпреки че обучението може да бъде много бавно, обучената по този метод мрежа реагира бързо и точно в работен режим. Разработени са редица подобрения на оригиналния алгоритъм, които водят до ускоряване на процеса на обучение на мрежата.

Обикновено това са многослойни невронни мрежи, чиито изходни и скрити слоеве имат отместване (bias). Отместването на изходния неврон y_k се означава с w_{0k} , а на неврон z_j от скрития слой - съответно с v_{0j}.

Тези отмествания играят ролята на тегла, приложени към връзки от фиктивни неврони, чиито изходи са винаги единица. Изкуственият неврон сумира претеглените входни сигнали и на тяхна база изработва свой изход, използвайки някаква математическа функция. Тази математическа функция се нарича активационна. Обикновено за невроните от един и същ слой активационната функция е еднаква. На фигурата е даден пример за такава функция за невроните от скрития слой в една многослойна невронна мрежа:

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката



При този метод, както и при методите, разглеждани досега, се използва обучение с учител. Това е най-често използваният метод. При него на мрежата се подават входни и съответстващи им изходни набори от данни, на базата на които се определят теглата в мрежата. Процесът на настройка на теглата (алгоритъма) се нарича обучение на невронната мрежа. Многослойните мрежи са по-подходящи за някои по-сложни задачи за класификация. Многослойните мрежи, например, могат да бъдат обучени да извършват нелинейни трансформации на вектори от едно пространство в друго, като двете пространства могат да имат различна размерност.

Алгоритъм за обучение

Обучението на невронната мрежа преминава през три етапа. По време на първия, наречен право разпространение, всеки входен неврон получава входен сигнал и го изпраща на всеки от невроните от скритият слой. Невроните в скритият

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

слой, от своя страна, изработват своя изходен сигнал на тази база и го изпращат на невроните от изходния слой. В края на процеса изходните неврони изчисляват своите изходни сигнали и по този начин формират реакцията на мрежата на подаденото входно въздействие.

На следващия етап от обучаващия процес реакцията на всеки изходен неврон се сравнява с желаната реакция с цел определяне на грешката. На нейна основа се определя коефициентът δ_k за всеки от невроните в изходния слой. Коефициентът δ_k се използва, за да се разпредели грешката от изходния слой към предходния. По-късно тя се използва за преизчисляване на теглата между скрития и изходния слой. По подобен начин се изчислява и δ_j за скрития слой. Не е необходимо разпространението на грешката назад към входните неврони, но тя се използва за обновяване на стойността на тегловните коефициенти между входа и скрития слой.

След като всички коефициенти δ бъдат определени, теглата на всички слоеве на мрежата се обновяват едновременно.

Активационната функция на мрежата с обратно разпространение на грешката трябва да има няколко важни свойства – да бъде непрекъсната, диференцируема

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

желателно производната й да бъде лесна за изчисление. Обикновено това са функции, чиито производни могат да бъдат намерени с помощта на стойностите на самата функция. Друго условие е функцията да има крайни максимум и минимум, които да се достигат асимптотически. Най-често използваната функция, отговаряща на горните условия, е сигмоидалната или биполярната сигмоидална. Коя функция ще се използва се определя от вида на обучаващите изходни данни.

Последователността от стъпки при обучение е следната

Стъпка 1.

Инициализиране на теглата – избират се произволни малки стойности.

Стъпка 2.

Изпълняват се стъпки от 3 до 8 за всяка обучаваща извадка.

Право разпространение на сигнала към скрития слой.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Право разпространение на сигнала към изходния слой.

Стъпка 3.

Входният слой получава набор от обучаващи данни от извадката и го предава към следващия слой.

Стъпка 4.

Всеки неврон от скрития слой сумира претеглените входни сигнали

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad ,$$

прилага активационната си функция за да изчисли изходния си сигнал

$$z_j = f(z\_in_j)$$

и го изпраща към всички неврони от следващия слой (изходния).

Стъпка 5.

Всеки от изходните неврони сумира претеглените входни сигнали от скрития слой преди него

$$y\_in_k = w_{0k} + \sum_{j=1}^{m} z_j w_{jk}$$

и, прилагайки своите активационни функции, формира изходния сигнал на мрежата

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$y_k = f(y\_in_k).$$

Обратно разпространение на грешката.

Стъпка 6.

Всички изходни неврони получават обучаващ изходен вектор, съответстващ на подадения на входа, и според него се намира коефициентът на грешката

$$\delta_k = (t_k - y_k) f'(y\_in_k),$$

изчислява се корекционния коефициент за обновяване на теглата по-късно

$$\Delta w_{jk} = \alpha \delta_k z_j,$$

изчислява се корекционния коефициент за обновяване на отместването

$$\Delta w_{0k} = \alpha \delta_k$$

и се изпраща δ_k към невроните в предходния слой.

Стъпка 7.

Всеки неврон от междинния слой сумира сигналите на грешките, получени от следващия (изходния) слой

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk},$$

умножава ги по производната на активационната си функция, за да намери своя коефициент на грешката

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$\delta_j = \delta\_in_j f'(z\_in_j),$$

изчислява корекционния коефициент за теглата на връзките от входния слой

$$\Delta v_{ij} = \alpha \delta_j x_i,$$

както и коефициента за коригиране на отместването

$$\Delta v_{0j} = \alpha \delta_j .$$

Обновяване на тегловните коефициенти.

Стъпка 8.

Всеки от невроните в изходния слой обновява теглата и отместванията си

$$w_{jk\_нов} = w_{jk\_стар} + \Delta w_{jk},$$

а невроните в междинния слой обновяват своите

$$v_{ij\_нов} = v_{ij\_стар} + \Delta v_{ij} .$$

Стъпка 9.

Ако условието за прекратяване на алгоритъма (например грешката е спаднала под някакъв определен праг) не е изпълнено, стъпки от 2 до 9 се повтарят. В противен случай обучението завършва.

Един такъв цикъл (Стъпки 2 до 9) се нарича епоха (epoch) и обикновено трябва да бъде повторен многократно. Обновяването на теглата може да се извърши след подаване на всяка от обучаващите извадки, но обикновено това става след

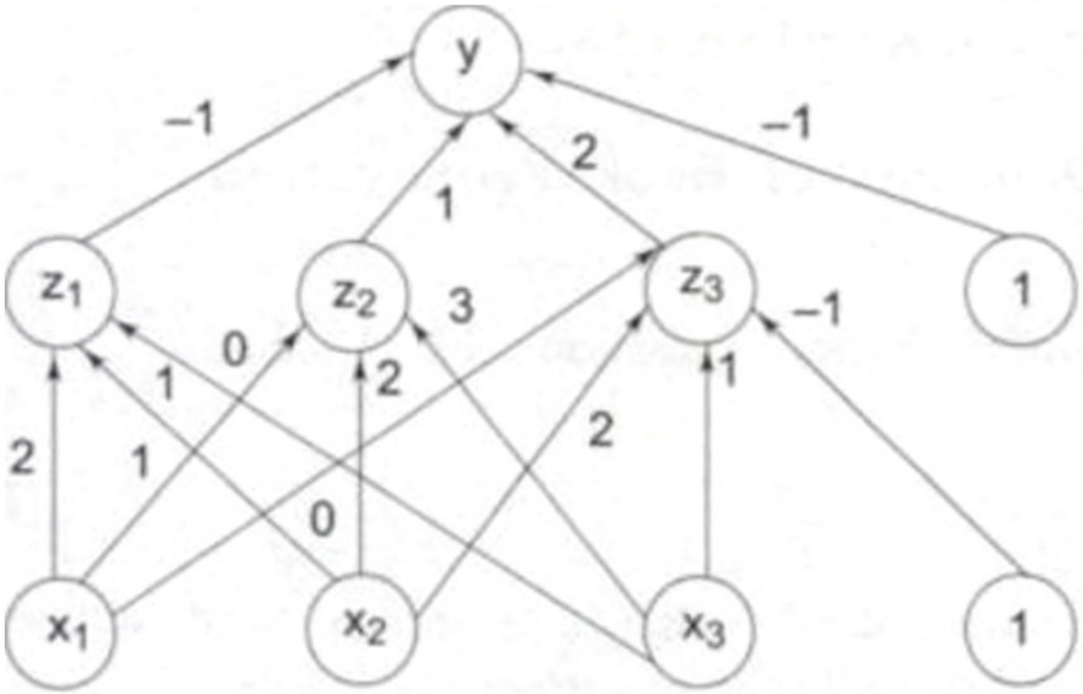
Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

завършването на една епоха, т.е. след подаването на целия обучаващ комплект от извадки.

В основата си този алгоритъм представлява оптимизационна техника, позната като градиентен метод. Градиентът на функцията (грешката) дава посоката, в която функцията нараства най-бързо. Обратната посока на градиента дава посоката, в която функцията намалява рязко.

Пример за работата на алгоритъма. Да се намерят новите тегла за показаната на фигурата невронна мрежа при входна послеователност [0.6 0.8 0] и желана изходна стойност 0.9. Нека скоростта на обучение да е α и да се използва бинарна сигмоидална активационна функция.

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката



Примерна многослойна мрежа

Решение.

Стъпка 1. Инициализираме теглата и отместванията.

$$\mathbf{W} = [-1 \quad 1 \quad 2], wo = [-1] \ ,$$

$$\mathbf{V} = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 2 \\ 0 & 3 & 1 \end{bmatrix} \ ,$$

$$\mathbf{Vo} = [0 \quad 0 \quad -1] \ .$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Стъпка 3. За всяка двойка от обучаващата извадка се задава

$$\mathbf{X} = [0.6 \quad 0.8 \quad 0] \ ,$$
$$t = [0.9] \ .$$

Право предаване на сигнала.

Стъпка 4.

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad ,$$

следователно,

$$z\_in_1 = v_{01} + \sum_{i=1}^{3} x_i v_{i1} = v_{01} + x_1 v_{11} + x_2 v_{21} + x_3 v_{31} = 0 + 0.6 \times 2 + 0.8 \times 1 + 0 \times 0 = 1.2 + 0.8 = 2,$$

$$z\_in_2 = v_{02} + \sum_{i=1}^{3} x_i v_{i2} = v_{02} + x_1 v_{12} + x_2 v_{22} + x_3 v_{32} = 0 + 0.6 + 0.8 \times 2 + 0 \times 3 = 2.2 \quad ,$$

И

$$z\_in_3 = v_{03} + \sum_{i=1}^{3} x_i v_{i3} = v_{03} + x_1 v_{13} + x_2 v_{23} + x_3 v_{33} = -1 + 0.6 \times 0 + 0.8 \times 2 + 0 \times 1 = 0.6$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$z1 = f(z\_in1) = \frac{1}{1+e^{-2}} = 0.8808 \quad ,$$

$$z2 = f(z\_in2) = \frac{1}{1+e^{-2.2}} = 0.9002 \quad ,$$

$$z3 = f(z\_in3) = \frac{1}{1+e^{-0.6}} = 0.646 \quad .$$

Стъпка 5. Изчисляваме

$$y\_inK = w_{0K} + \sum_{j=1}^{p} z_j w_{jk} \quad ,$$

$$y\_in1 = w_{01} + \sum_{j=1}^{3} z_j w_{j1} = w_{01} + z_1 w_{11} + z_2 w_{21} + z_3 w_{31} =$$

$$= -1 + 0.8808 \times (-1) + 0.9002 \times 1 + 0.646 \times 2 = -1 - 0.8808 + 0.9002 + 1.292 \quad ,$$

$$y\_in1 = [0.3114] \quad .$$

Изчисляваме изходния сигнал

$$y_1 = f(y_{in1}) = \frac{1}{1+e^{-0.3114}} = 0.5772 \quad .$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

Обратно разпространение на грешката.

Стъпка 6. Изчисляваме коефициента на грешката

$$\delta_k = (t_k - y_k) f'(y_{\_ink}) \ ,$$

$$\delta_1 = (t_1 - y_1) f'(y_{\_in1})$$

и тъй като знаем, че за двоични сигмоидални функции

$$f'(x) = f(x)(1 - f(x)) \ ,$$

$$f'(y_{\_in1}) = f(y_{\_in1})(1 - f(y_{\_in1})) = 0.5772(1 - 0.5772) = 0.2440 \ ,$$

$$\delta_1 = (t_1 - y_1) f'(y_{\_in1}) = (0.9 - 0.5772)(0.2440) = 0.0708 \ .$$

Стъпка 7 Обратно разпространение към първия скрит слой (i=1,2,3). Изчисляваме

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk}$$

$$\delta\_in1 \Rightarrow \delta_1 w_{11} \Rightarrow 0.0788 \times (-1) = -0.0788$$

$$\delta\_in2 \Rightarrow \delta_1 w_{21} \Rightarrow 0.0788 \times 1 = 0.0788$$

$$\delta\_in3 \Rightarrow \delta_1 w_{31} \Rightarrow 0.0788 \times 2 = 0.01576$$

За да се изчисли големината на грешката в скрития слой

$$\Delta = \delta_{\_inj} f'(z_{\_inj}) \ ,$$

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$f'(z\_in_j) = f(z\_in_j)(1 - f(z\_in_j)) = (0.8808)(1 - 0.8808) = 0.1049 \ ,$$
$$\Delta_1 = \delta\_in_1 f(z\_in_1)(1 - f(z\_in_1)) = (-0.0788)(0.1049) = -0.0083 \ ,$$
$$\Delta_2 = \delta\_in_2 f'(z\_in_2) \ ,$$
$$f'(z\_in_2) = (0.9002)(1 - 0.9002) = 0.09 \ ,$$
$$\Delta_2 = 0.0788 \times 0.09 = 0.0071 \ ,$$
$$\Delta_3 = \delta\_in_3 f'(z\_in_3) \ ,$$
$$f'(z\_in_3) = (0.646)(1 - 0.646) = 0.2286 \ ,$$
$$\Delta_2 = 0.1576 \times 0.2286 = 0.0361 \ .$$

Стъпка 8. Обновление на теглата

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$\Delta v_{ij} = \alpha \Delta_j x_i \ ,$$
$$x = [0.6 \quad 0.8 \quad 0] \ ,$$
$$\Delta = [-0.0083 \quad 0.0071 \quad 0.0361] \ ,$$
$$\alpha = 0.3 \ ,$$
$$\Delta v_{11} = \alpha \Delta_1 x_1 = 0.3 \times -0.0083 \times 0.6 = -0.0015 \ ,$$
$$\Delta v_{12} = \alpha \Delta_2 x_1 = 0.3 \times 0.0071 \times 0.6 = 0.0013 \ ,$$
$$\Delta v_{21} = \alpha \Delta_1 x_2 = 0.3 \times -0.0083 \times 0.8 = -0.002 \ ,$$
$$\Delta v_{22} = \alpha \Delta_2 x_2 = 0.3 \times 0.0071 \times 0.8 = 0.0017 \ ,$$
$$\Delta v_{13} = \alpha \Delta_3 x_1 = 0.3 \times 0.0361 \times 0.6 = 0.0065 \ ,$$
$$\Delta v_{23} = \alpha \Delta_3 x_2 = 0.0087 \ ,$$
$$\Delta v_{31} = \Delta v_{32} = \Delta v_{33} = 0 \ ,$$
$$\Delta v_{01} = \alpha \Delta_1; \Delta v_{02} = \alpha \Delta_2; \Delta v_{03} = \alpha \Delta_3 \ ,$$
$$\Delta v_{01} = 0.3 \times -0.0023 \Rightarrow -0.0025 \ ,$$
$$\Delta v_{02} = 0.3 \times 0.0071 \Rightarrow 0.0021 \ ,$$
$$\Delta v_{03} = 0.3 \times 0.0361 \Rightarrow 0.0108 \ ,$$

$$v_{new} = v_{old} + \Delta v_i \ ,$$
$$v_{11}(new) = v_{11}(old) + \Delta v_{11} = 2 - 0.0015 = 1.9985 \ ,$$
$$v_{12}(new) = v_{12}(old) + \Delta v_{13} = 9 + 0.0013 = 1.0013 \ ,$$
$$v_{13}(new) = v_{13}(old) + \Delta v_{13} = 0 + 0.065 = 0.065 \ ,$$
$$v_{21}(new) = v_{21}(old) + \Delta v_{21} = 1 - 0.002 = 0.998 \ ,$$
$$v_{22}(new) = v_{22}(old) + \Delta v_{22} = 2 + 0.0017 = 2.0017 \ ,$$
$$v_{23}(new) = v_{23}(old) + \Delta v_{23} = 2 + 0.0067 = 2.0087 \ ,$$
$$v_{31}(new) = v_{31}(old) + \Delta v_{31} = 0 + 0 = 0 \ ,$$
$$v_{32}(new) = v_{32}(old) + \Delta v_{32} = 3 + 0 = 3 \ ,$$
$$v_{33}(new) = v_{33}(old) + \Delta v_{33} = 1 + 0 = 1$$

и по този начин получаваме, че

Невронни мрежи с право предаване на сигнала и обратно разпространение на грешката

$$\mathbf{V} = \begin{bmatrix} 1.9985 & 1.0013 & 0.085 \\ 0.998 & 2.0017 & 2.0087 \\ 0 & 3 & 1 \end{bmatrix}.$$

След това изчисляваме

$$\Delta w_{OK} = \alpha \delta_K z_j \ ,$$

$$\Delta w_{11} = \alpha \delta_1 z_1 = 0.3 \times 0.0788 \times 0.8808 = 0.0208 \ ,$$

$$\Delta w_{12} = \alpha \delta_1 z_2 = 0.3 \times 0.0788 \times 0.9002 = 0.0212 \ ,$$

$$\Delta w_{13} = \alpha \delta_1 z_3 = 0.3 \times 0.0788 \times 0.646 = 0.0153 \ ,$$

$$\Delta w_{11}(new) = w_{11}(old) + \Delta w_{11} = -1 + 0.0208 = 0.9792 \ ,$$

$$\Delta w_{12}(new) = w_{12}(old) + \Delta w_{12} = 1 + 0.0212 = 1.0212 \ ,$$

$$\Delta w_{13}(new) = w_{13}(old) + \Delta w_{13} = 2 + 0.0.153 = 2.0153$$

и получаваме, че

$$\Delta w_O = 0.3 \times 0.0788 = 0.02364 \ ,$$

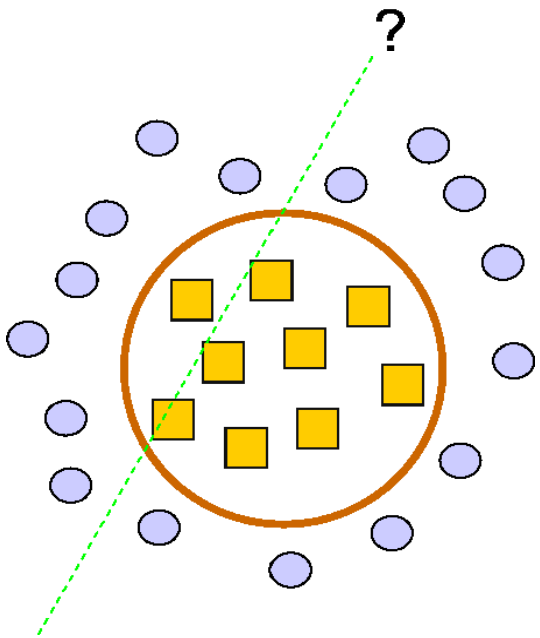
$$\mathbf{W} = \begin{bmatrix} 0.9792 & 1.0212 & 2.0153 \end{bmatrix} .$$

По този начин теглата биват изчислени и процедурата може да бъде продължена до удовлетворение на някое от правилата за спиране на итерационната процедура.

Невронни мрежи с радиални базисни функции (RBFN)

Проблемът с линейната разделимост

- Използването на линейни разделящи правила е лесен за реализация и използване подход.
- На практика, обаче, съществуват множество класификационни задачи, които не могат да бъдат решени с линейни класификатори



В многократно разглежданото до този момент 2-D пространство на данните, групата на квадратчетата и кръгчетата няма как да бъдат разделени от който и да е линеен класификатор.

Решение? – нелинеен класификатор?

Теорема на Cover

- Теоремата на Cover за разделимост на образи гласи, че:

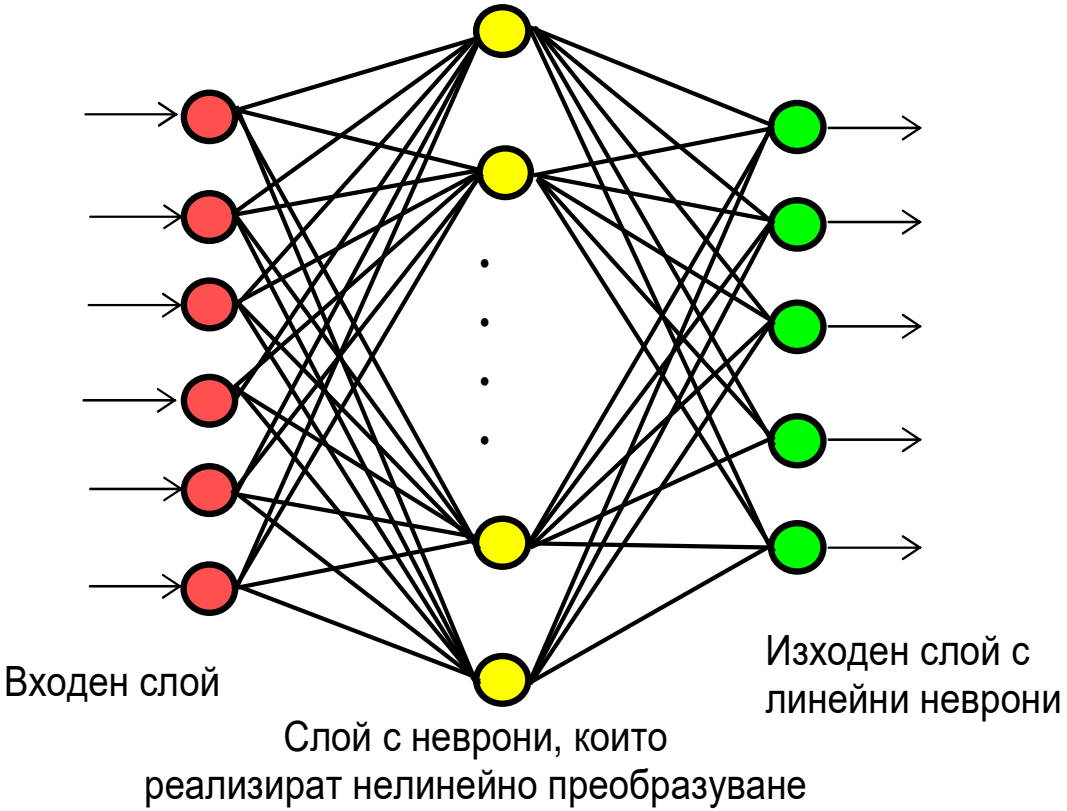
  Образите, които са нелинейно разделими в пространство с ниска размерност, е възможно да бъдат линейно разделими в пространство с по-висока размерност.

  →търси преобразуване на пространството, в което са описани зададените образи, в пространство с по-голяма размерност, така че в новото пространство образите да бъдат линейно разделими, използвайки само един слой от неврони с линейни активационни функции.

RBFN Архитектура

Имаме проста трислойна структура

- Входен слой
- Скрит слой с нелинейна активационна функция
- Изходен слой с линейна активационна функция

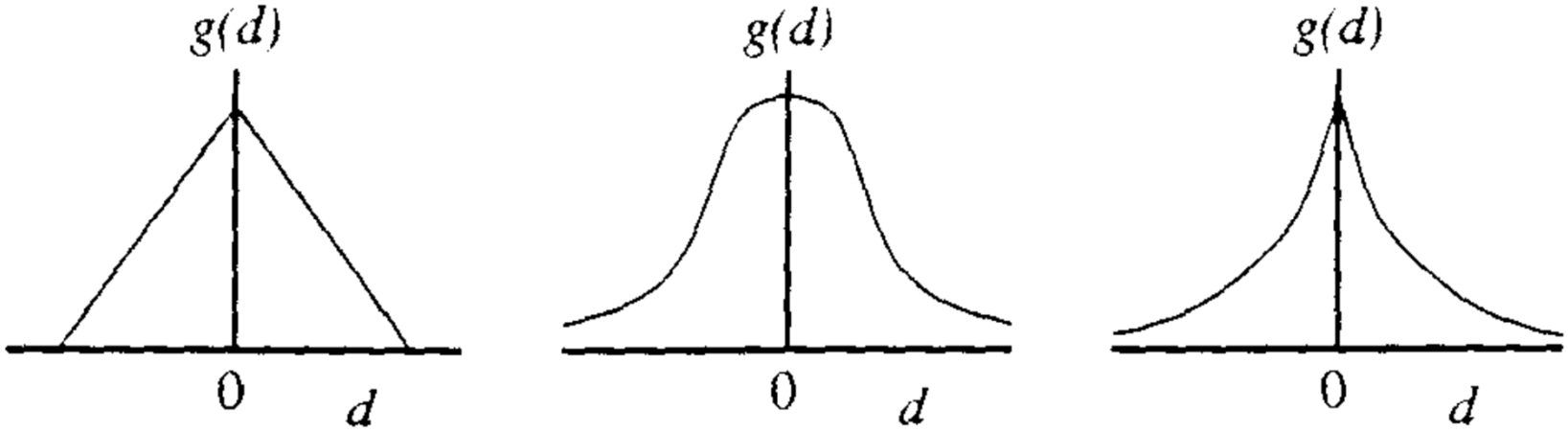


Радиални базисни функции

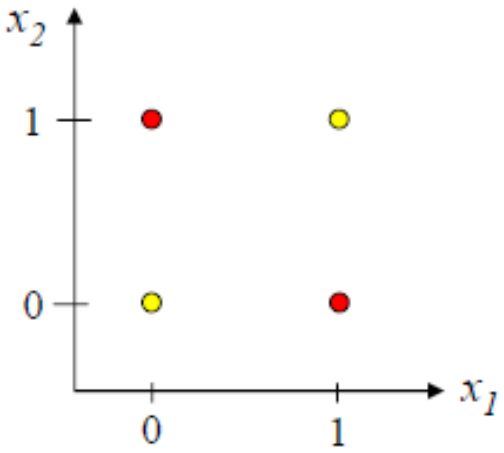
- Показани са някои типични форми на такива ф-ции (по разстояние, Гаусови и реципрочни)
- Най-често се използва Гаусовата радиално-базисна функция:

$$g(\mathbf{x}) = \exp[-(\|\mathbf{x} - \mathbf{m}\|/\sigma)^2]$$

d е разстоянието между запомнената последователност и новата последователност



Задачата за решаване на XOR



Нека имаме нелинейни функции, които правят преобразуване на водния вектор x в φ_1- φ_2 пространството

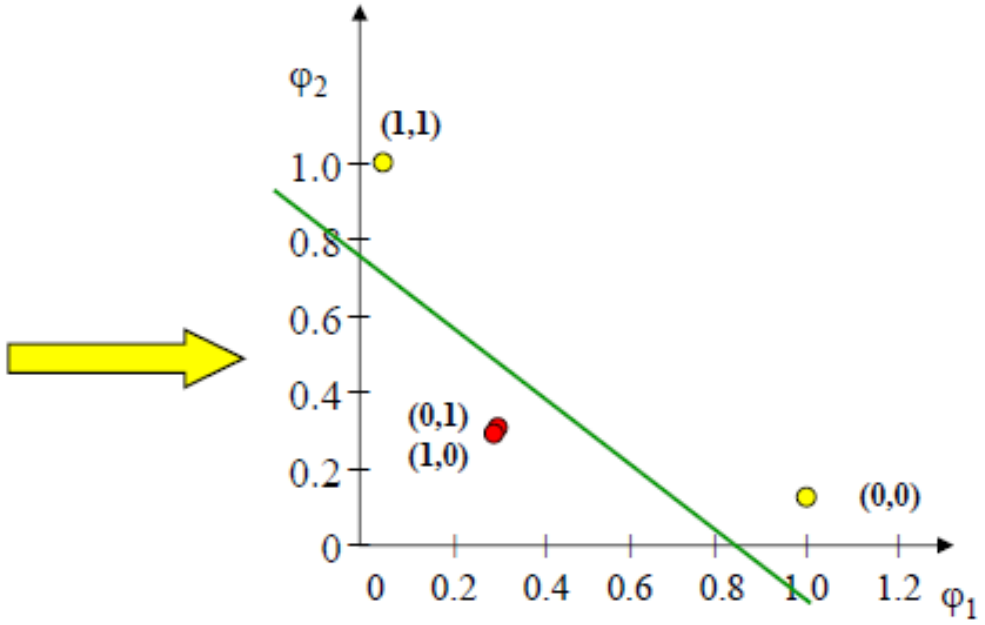
$\mathbf{x}=[x_1\ x_2]$

$$\varphi_1(\mathbf{x}) = e^{-\|\mathbf{x}-u_1\|^2} \qquad u_1 = [u_{11}\ \ u_{12}]^T = [1\ \ 1]^T$$

$$\varphi_2(\mathbf{x}) = e^{-\|\mathbf{x}-u_2\|^2} \qquad u_2 = [u_{21}\ \ u_{22}]^T = [0\ \ 0]^T$$

Нелинейната функция φ преобразува една нелинейно разделима задача в линейноразделима!!!

Вход x	$\varphi_1(x)$	$\varphi_2(x)$
(1,1)	1	0.1353
(0,1)	0.3678	0.3678
(1,0)	0.3678	0.3678
(0,0)	0.1353	1



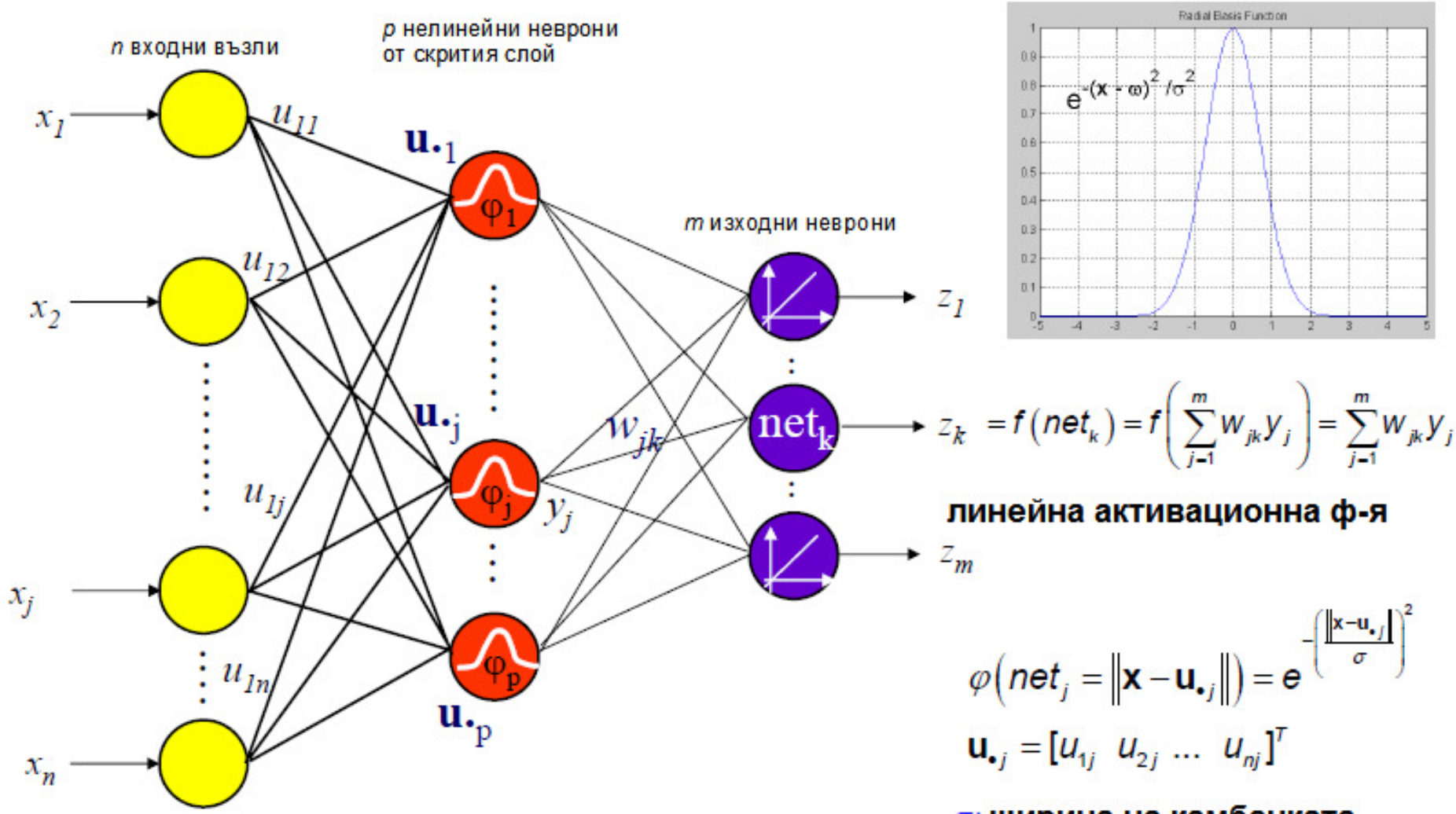
Оценка на метода

- Използвайки нелинейни функции ние можем да преобразуваме една задача за нелинейна класификация в задача за линейна класиф.
- От гледна точка на апроксимацията на функции, това е еквивалентно на имплементация на комплексна функция (съответстваща на нелинейната разделяща хиперравнина) използвайки прости функции (съответстващи на линейната разделяща хиперравнина както при Перцептрона)
- Имплементацията на тази процедура (идеология) в мрежова структура, води до RBF мрежи, ако нелинейните преобразуващи функции са радиални базисни.

Радиални Базисни Функции(RBF):

- Радиални: Симетрични са около центъра им
- Базисни функции: набор от функции, с чиито линейни комбинации могат да генерират произволни функции в даденото пространство на функцията

Мрежи с радиални базисни функции



$$z_k = f(net_k) = f\left(\sum_{j-1}^{m} w_{jk} y_j\right) = \sum_{j-1}^{m} w_{jk} y_j$$

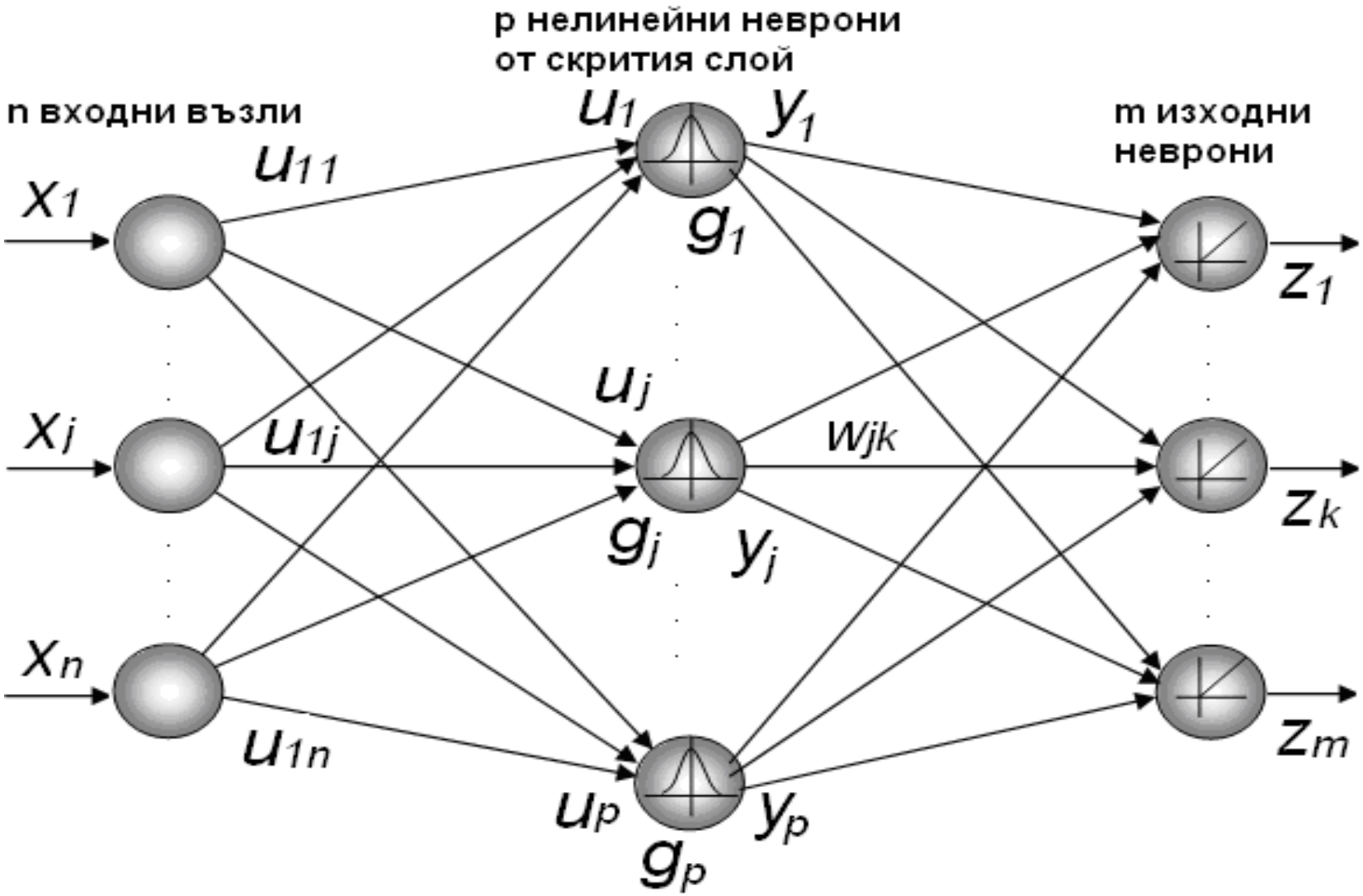
линейна активационна ф-я

$$\varphi\left(net_j = \left\|\mathbf{x} - \mathbf{u}_{\bullet j}\right\|\right) = e^{-\left(\frac{\left\|\mathbf{x}-\mathbf{u}_{\bullet j}\right\|}{\sigma}\right)^2}$$

$$\mathbf{u}_{\bullet j} = [u_{1j} \;\; u_{2j} \;\; \ldots \;\; u_{nj}]^T$$

σ: **ширина на камбанката**

Мрежи с радиални базисни функции



Параметри за настройка на RBFN

Три различни типа величини определят ‘производителността’ на RBFN мрежи с Гаусови функции:

- Центърът $\mathbf{u}_{\bullet j}$ на всяка радиално-базисна функция
- Ширината σ на всяка радиално-базисна активационна функция (Най-често ширината на камбанката за всички радиални функции се избира еднаква $\sigma_1=\sigma_2=...=\sigma_p=\sigma$, въпреки че в общия случай ширината на камбанките за отделните радиални функции са различни).
- Теглата w_{kj} на изходния слой

$$z_k = f\left(z_{in\_k}\right) = f\left(\sum_{j=1}^{p} w_{jk}\, y_j\right) = \sum_{j=1}^{p} w_{jk}\, y_j\,, \quad k = 1,2,...,m$$

$$y_j = g_j(y_{in\_j}) = g(y_{in\_j}) = g\left(\left\|\mathbf{x}-\mathbf{u}_{\bullet j}\right\|\right) = e^{-\left(\frac{\left\|\mathbf{x}-\mathbf{u}_{\bullet j}\right\|}{\sigma}\right)^2}$$

Алгоритми за обучение

Нормално при настройката на параметрите имаме два стадия:

- Моделиране на радиалните базисни функции : центрове и ширини на камбанките (почти винаги е ‘без учител’, т.н. unsupervised). Обикновено радиалните функции се избират предварително, т.е. параметрите $\mathbf{u}_{\bullet j}$ и σ се избират без обучение.
- Настройка (обучение) на теглата (‘с учител’, т.н. supervised). В този случай теглата между скрития и изходния слой се настройват с използване на обучение, базирно на градиентен подход за минимизиране на средноквадратичната грешка от желаните и изчислените изходи на мрежата за елементите на обучаващата извадка.

Алгоритми за обучение1

- Най-често се използват три подхода при обучението на невронните мрежи с радиални базисни функции.
- При първия подход центровете $\mathbf{u}_{\bullet j}$, $j=1,2,\ldots,p$ на радиалните функции се избират случайно в пространството на входните сигнали $[x_1\ x_2\ \ldots\ x_n]^T$. Ширината на камбанките на радиалните функции се изчислява с формулата $\sigma=\rho_{max}/(\sqrt{2p})$, където p е броя на невроните от скрития слой, а ρ_{max} е максималното разстояние между всеки два неврона. Този избор гарантира, че радиалните функции няма да бъдат много стръмни или широки. След това, за всеки от елементите на обучаващата извадка се изчисляват изходните сигнали на невроните от скрития слой. Тъй като за всеки от елементите на обучаващата извадка са известни желаните изходни сигнали, обучението за намирането на теглата между скрития и изходния слой се свежда до прилагането на метода на най-малките квадрати, както при адаптивния линеен елемент

  $\mathbf{w}_{\bullet k}(t+1) = \mathbf{w}_{\bullet k}(t) + \eta\ \mathbf{y}(t)e_k(t)$,

  където $\mathbf{w}_{\bullet k}(t)$ е векторът с теглата свързващи скрития слой и k-ти изходен неврон на стъпка t от прилагане на алгоритъма, $e_k(t) = d_k(t) - \mathbf{y}(t)\ \mathbf{w}_{\bullet k}(t)$ е грешката свързана с k-ти изходен неврон $k=1,2,..,m$, $\mathbf{y}(t)$ е векторът с изходните сигнали на невроните от скрития слой на стъпка t, $d_k(t)$ е желаният изходен сигнал на k-ти изходен неврон на стъпка t, а η е скоростта на обучение.

Алгоритми за обучение2

- При втория подход центровете $\mathbf{u}_{\bullet j}$, $j=1,2,\ldots,p$ на радиалните функции се избират по подходящ начин в пространството на входните сигнали $[x_1\ x_2\ \ldots\ x_n]^T$.
- Съществуват редица подходи за избор на центровете на радиалните функции, като използване на най-близки съседи, самоорганизация (както е разгледано в следващата глава) и др.
- За ширината на камбанките на радиалните функции може да се използва разгледаната вече формула $\sigma=\rho_{max}/(\sqrt{2p})$. Тогава за всеки от елементите на обучаващата извадка се изчисляват изходните сигнали на невроните от скрития слой.
- Теглата между скрития и изходния слой на мрежата се намират с обучение, като се използва методът на най-малките квадрати

Алгоритми за обучение3

- При третия подход всички параметри - центровете на всяка радиална функция $\mathbf{u}_{\bullet j}$, теглата на връзките между скрития и изходния слой и ширината на камбанките на всяка от радиалните функции σ се настройват с обучение.
- Използва се принципът за намаляване на средноквадратичната грешка между желаните и изчислените стойности на изходните сигнали на невроните.
- Алгоритмите на най-малките квадрати за мрежи с радиални базисни функции са обобщение на оригиналния подход с най-малки квадрати.

RBF мрежи

- Невронните мрежи с радиални базисни функции се използват като универсален апарат за апроксимация при известни входно-изходни данни за обекта, който се моделира (както и многослойните невронни мрежи с обратно разпространение на грешката).
- В много случаи, сходимостта на алгоритмите за обучение при тези мрежи е много по-бърза в сравнение с мрежите с обратно разпространение на грешката.
- Резултатите от използването на невронните мрежи с радиални базисни функции са лесни за интерпретация.
- Тези невронни мрежи са основа за въвеждане на експертна информация, за невроните в скрития слой и по този начин могат да бъдат получени различни модификации на т.нар. размито-невронни системи.
- **Като недостатък може да се изтъкне, че тези мрежи много често изискват по-голям брой неврони в скрития слой.**