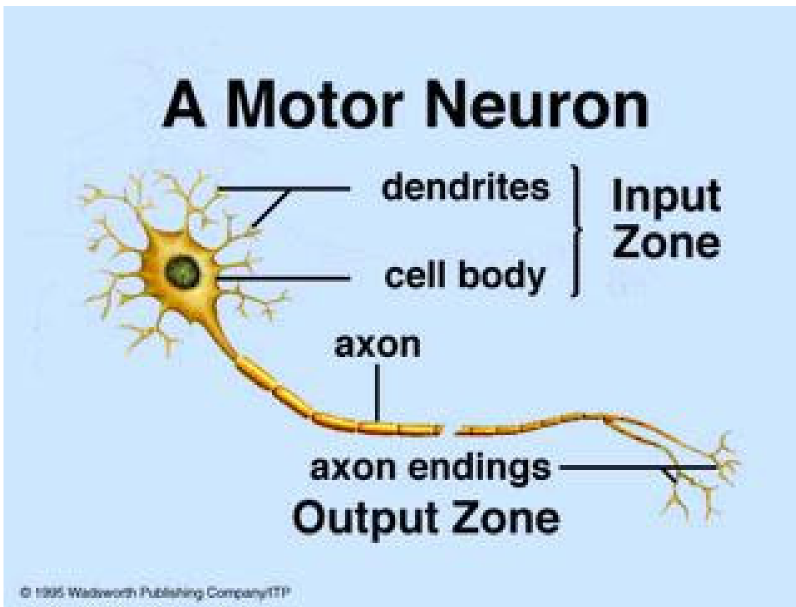
A neurális hálózatok alapjai

(A „Neurális hálózatok és műszaki alkalmazásaik” című könyv (ld. források) alapján)

1. Biológiai alapok

A biológiai alapok megismerése azért fontos, mert nagyon sok egyedi neurális struktúra, és maguk a neurális hálózatok is ezen alapulnak. Az agy az ember elsődleges információfeldolgozó egysége. Képes tanulni, nagyságrendekkel gyorsabb, mint a jelenlegi processzorok, hibatűrő képessége kiemelkedő. Az agy fő feldolgozó egységei a neuronok.

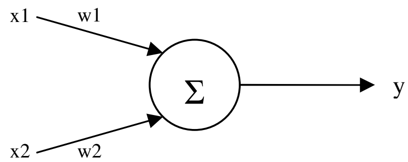


A neuron felépítése

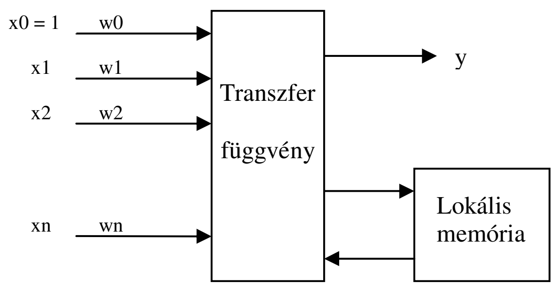
Az agyi neuronok bemenetei a dendritek, amelyek más neuronokhoz csatlakoznak. A tényleges feldolgozást a neuron tulajdonképpeni sejtmagja végzi, és az eredményét az axonon keresztül juttatja el a többi neuronhoz. Az axon végződései több másik neuron dendritjeihez csatlakozhatnak. Az idegrendszer további részei a neuronok mellett a gliasejtek, amelyek tápláló és védő funkciót töltenek be (az utóbbi idők kutatásai szerint egyéb a feldolgozást is segítő funkcióik is vannak).
A biológiai kutatások eredményeképpen egyre többet tudhatunk meg a neuronok és az agy felépítéséről, működéséről. Ezen kutatások ereményeit felhasználva műszaki analógiák sorát alkothatjuk meg, amelyek, mint számos példa bizonyítja, használhatóak is lehetnek.

2. A mesterséges neuron

A mesterséges neuron (processing element) az agyi neuron másolata. A legegyszerűbb formájában a következőképpen néz ki:



A két bemenet súlyozva egy összegző bemenetére kerül, és ez adja a kimenetet (ld. Oja hálózat).
A neuron jóval általánosabb megfogalmazása is lehetséges. Az n súlyozott bemenetet és egy konstansra választott bemenetet (bias) egy összegzés és valamilyen általában nemlineáris függvény követ. Ez utóbbi kettőt szokás transzfer függvénynek is nevezni. A neuron rendelkezhet lokális memóriával is, amelyet például késleltetéseken, visszacsatolásokon, lineáris szűrőkön keresztül érhetünk el.
Egy ábrán illusztrálva a fentieket:



A transzfer függvény tehát tulajdonképpen egy összegző, amely a bemenetek súlyozott összegét adja, és egy ezt követő általában nemlinerális függvény. A következőkben néhány ilyen függvényt adunk meg:

$$y(s) = \begin{cases} +1, s > 0 \\ -1, s \le 0 \end{cases}$$

lépcsőfüggvény

$$y(s) = \begin{cases} +1, s > 1 \\ s, -1 \le s \le 1 \\ -1, s < -1 \end{cases}$$

telítéses lineáris függvény

$$y(s) = \frac{1 - e^{-Ks}}{1 + e^{-Ks}}; K > 0$$

tangens hiperbolikus függvény (K=2)

$$y(s) = \frac{1}{1+e^{-Ks}}; K > 0$$

logisztikus függvény

Természetesen más típusú függvények is elképzelhetőek.
A neuron által megvalósított leképezés tehát így írható:

$$y = f(s) = f(\sum_{i=0}^{N} w_i x_i) = \mathbf{w}^T \mathbf{x},$$

ahol **x** a bemeneti vektor, **w** a súlyvektor, míg f egy függvény, például valamelyik a fentiek közül.

3. Neurális hálózatok

A neuronokból álló hálózatokat nevezzük neurális hálózatoknak. De nézzük meg, hogy milyen jellemzők is írjnak le egy ilyen hálózatot. A neuronok ugyanolyan vagy hasonló típusú műveleteket végeznek. Egy hálózatban ezeket a műveleteket a többiektől függetlenül, lokálisan végzik. Egy neuron nagyon sok másik neuronnal lehet összekapcsolva, és mint sejthetjük a biológiai analógiából, ezek a hálózatok tanulni is képesek. Tehát a neurális hálózatok olyan információfeldolgozó eszközök, amelyek párhuzamos, elosztott működésre képesek, lokális feldolgozást végző neuronokból állnak, képesek tanulni, és a megtanult információt felhasználni. Megvalósításuk szerint lehetnek hardver, szoftver, vagy akár a kettő kombinációja is.
A neurális hálózatok tehát a neuronok különböző összekapcsolásaiból jönnek létre. Általában egy irányított gráffal reprezentáljuk őket, amelyben a csomópontok az egyes neuronok, míg az irányok a kimenetektől a bemenetek felé mutatnak. Bonyolultabb esetekben ez a jelölésmód áttekinthetetlen lehet.
A neuron egy hálózaton belül általában csak meghatározott számú neuronnal vannak összekötve, és ez a kapcsolat általában egyirányú, ezért a hálózatokat különböző rétegekre szoktuk bontani az összekötések szerint. Az egyes rétegekhez tartozó neuronok az előző réteg neuronjainak kimenetével, vagy a bemenettel, illetve a következő réteg bemenetével vannak összekötve.

a. Bemeneti réteg: azok a neuronok találhatók itt, amelyek a bemeneti jel továbbítását végzik a hálózat felé. A legtöbb esetben nem jelöljük őket külön.
b. Rejtett réteg: a tulajdonképpeni feldolgozást végző neuronok tartoznak ide. Egy hálózaton belül több rejtett réteg is lehet.
c. Kimeneti réteg: az itt található neuronok a külvilág felé továbbítják az információt. A feladatuk ugyanaz, mint a rejtett rétegbeli neuronoké.

A neuronokat a hálózatokon belüli elhelyezkedésük alapján általában három csoportba szokták sorolni.

a. bementi neuronok (nem minden irodalom jelöli őket): ezek a neuronok csak jeltovábbítást, a neuronok meghajtását végzik
b. rejtett neuronok: mind a bemeneteik, mind a kimenetük valamely más neuronhoz csatlakozik
c. kimeneti neuronok: a külvilág felé továbbítják az információt, jellegükben és feladatukban nem különböznek a többi neurontól

A legtöbb hálózat esetében az egyes rétegek teljesen össze vannak kötve, vagyis egy réteg egy neuronjának kimenete a következő réteg összes bementével össze van kötve. A hálózatoknál előforduló számítások elvégzéséhez célszerűbb a hálózatokat különböző mátrixokkal jellemezni. A bemeneteket elegendő egy darab vektorba rendezve ábrázolni, ez N bemenet esetén egy N-es vektort jelent. Az egyes rétegek közötti súlyokat hasonlóan a bemenetekhez mátrixokba szoktuk rendezni. Ez a következőképpen néz ki: ha az l. rétegben n darab neuron van az $l+1$. rétegben pedig m, akkor a köztük lévő súlyok száma n x m. Ez mátrixba rendezve egy m x n-es mátrixot jelent, amelynek oszlopai az l. réteg neuronjait, sorai az $l+1$. réteg neuronjait reprezentálják. Vagyis a **W** súlymátrix w_{ij}-dik eleme az l. réteg j. neuronjának kimenete és az $l+1$. réteg i. neuronjának bemenete közti súly értékét adja meg. Ebben az esetben minden két réteg között egy **W** mátrix jelöli a súlyokat, vagyis L rejtett réteg esetén L db **W** mátrixunk lesz. Ennek egy általánosítása, amikor minden neuront külön megszámozunk, a mátrix elemeinek jelentése ugyanaz, mint az előző esetben és azok az elemeket, amelyek között nincs összeköttetés 0-val jelöljük (ez az előző esetben is érvényes). Ilyenkor csak egyetlen mátrixunk van.
A legtöbb esetben az egyes rétegek között nincs visszacsatolás, ilyenkor előrecsatolt hálózatokról beszélünk, visszacsatolás esetén visszacsatolt hálózatokról. A visszacsatolásnak több esete is lehetséges, aszerint, hogy a visszacsatolás honnan történt. Eszerint lehetséges egy rejtett réteg valamely neuronjának kimenetét a saját bementére (elemi visszacsatolás), egy előző réteg neuronjának bemenetére (rétegek közötti visszacsatolás), vagy egy ugyanabban a rétegben található neuron bementére visszacsatolni (laterális visszacsatolás).
Egy neurális hálózat tehát sokféleképpen épülhet fel, és mint látni fogjuk sokféle tanítási eljárással tanítható. Ezek a tulajdonság teszik lehetővé, hogy a neurális hálózatokat sok helyen alkalmazhassuk, hiszen a felépítéstől és tanítási eljárástól függően dinamikusan tudjuk változtatni a hálózatok tulajdonságait.

4. A neurális hálózatok felhasználási területei

A neurális hálózatok felhasználása sok területen ígéretes lehet. Néhány terület ezek közül.
<u>Adattal címezhető memória</u>
A hálózatot úgy tanítjuk, hogy azonos címhez rendelje az összetartozó adatokat. Előhíváskor a bementre adott adatra valamely megtanult címet adja vissza a hálózat.
<u>Asszociatív memória</u>
A hálózatnak ebben az esetben is valamilyen adatot tanítunk, de itt cím helyett a kimeneten is adatot várunk. Aszerint, hogy ugyanazt, vagy más adatot várunk a kimeneten auto-, vagy heteroasszociációról beszélünk. Ez a funkció különböző jelfeldolgozási feladatok során lehet hasznos.
<u>Osztályozási feladatok</u>
Adott egy mintahalmaz, amelynek elemei különböző osztályokba sorolhatóak. Gondoljunk például egy olyan feladatra, amelyben írott szövegből kell az egyes karaktereket felismerni. Célunk a rendelkezésre álló minták alapján a hálózatot megtanítani arra, hogy az egyes írott karaktereket helyesen ismerje fel. Sok esetben azonban nem ismeretes, hogy melyik mintát hova kell sorolni, ilyenkor a hálózatnak magának kell összefüggéseket keresnie az egyes minták között, és megtanulni őket.
<u>Optimalizálási feladatok</u>

A feladat valamilyen költségfüggvény optimalizálása. A neurális hálózatot úgy alakítjuk ki, hogy a megoldandó probléma paramétereit tartalmazza. A ilyen célra létrehozott hálózatok általában visszacsatolt struktúrák.

<u>Approximálási feladatok</u>

Sokszor feladat egy problémánál a megfelelő leképezés létrehozása. Rendelkezünk adatokkal, például mérési adatokkal valamilyen probléma esetében, de nem tudjuk az adatok által leírt függvény jellemzőit, és azok analitikus úton nehezen, vagy egyáltalán nem határozhatóak meg. Ilyenkor segíthetnek a neurális hálózatok.

<u>Nemlineáris dinamikus rendszer</u>

A legtöbb folyamat változik az időben, tehát egy adott pillanatbeli értéke, nem csak az adott pillanatban jelen lévő bemeneti jeltől, hanem az előzményektől is függ. Ide sorolhatók például az a feladatok, amikor különböző idősorokat kell elemezni, és előre jelezni a következő értékeket. Dinamikus rendszerek létrehozása neurális hálózatokkal sokféleképpen történhet. Lehetséges dinamikus neurális hálózatok létrehozása, visszacsatolásokkal, késleltetésekkel, lehetséges egy lineáris dinamikus rendszer paramétereinek állítása neurális hálózatokkal. A terület legtöbb problémája, hogy a dinamikus struktúrák a legtöbb esetben lassan tanulnak, és nem mindig biztosított a rendszer stabilitása.

<u>Konkrét alkalmazások</u>

Több alkalmazás is született már, ahol neurális hálózatokat alkalmaztak. A felhasználási területek főként a felismerési feladatok. Ilyenek az alakzatfelismerés, és osztályozás, előrejelzési, szabályozási feladatok. Alkalmazási területek a robotika (szabályozás), üzleti alkalmazások (előrejelzések), orvosi alkalmazások (alakzatfelismerés, osztályozás).

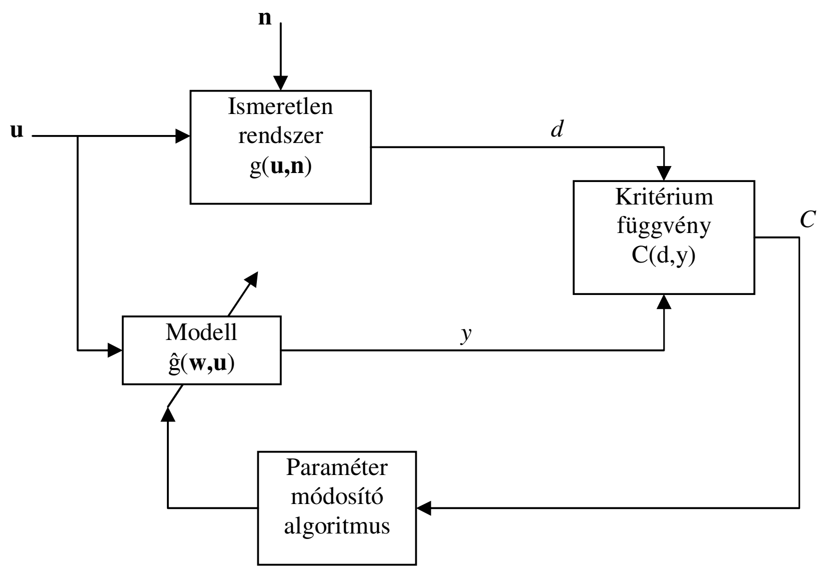
5. A neurális hálózatok tanulása

<u>Ellenőrzött tanulás</u>

Összetartozó be- és kimeneti adatpárok állnak rendelkezésre. Tanításkor tehát egy adott bemenet esetén tudjuk, hogy mit várunk a kimeneten, vagyis a hálózat válaszát össze tudjuk hasonlítani a kívánt válasszal. A keletkező hiba, a két válasz különbsége felhasználható a hálózat tanítására. Némely esetben csak azt tudjuk egy adott válaszról, hogy helyes-e vagy sem, de a pontos értéket nem ismerjük. Ezt a tanítási formát megerősítő tanulásnak nevezzük.

Az ellenőrzött tanulás esetében a célunk az, hogy a tanítandó hálózat működése minél jobban közelítse a vizsgált rendszer működését. Ez a megfelelő hálózat kiválasztásával és paramétereinek állításával történik.

A modell-illesztés általános alakja a következő ábrán látható:



A feladatunk tehát a

$$d = g(\mathbf{u},\mathbf{n})$$

függvénykapcsolathoz egy

$$y = \hat{g}(\mathbf{w},\mathbf{u})$$

változtatható paraméterű modell illesztését jelenti. Az illesztési feladat ebben az esetben általában egy minimalizálási, szélsőérték-keresési feladatot jelent, amelyben egy $C(d,y)$ kritériumfüggvény minimumát keressük **w** függvényében.
Neurális hálózatoknál a tanítás célja a megfelelő súlytényezők kialakítása, így a használt kritériumfüggvény az optimális ($\mathbf{w}^*$), és a tényleges súlyvektor ($\mathbf{w}$) függvénye lesz: $C(\mathbf{w}^*,\mathbf{w})$. A kritériumfüggvény megválasztása függ a feladat jellegétől, az esetleges a priori ismeretektől, és bizonyos mértékig szubjektív tényezőktől is.
A leggyakrabban használt kritériumfüggvény a négyzetes hibakritérium, amelyet a zaj jelenléte miatt várható értékként értelmezünk:

$$C(\mathbf{d},\mathbf{y}) = \mathrm{E}\left\{(\mathbf{d}-\mathbf{y})^T(\mathbf{d}-\mathbf{y})\right\} = \mathrm{E}\left\{\sum_j (d_j - y_j)^2\right\}$$

A négyzetes kritériumfüggvény előnyei, hogy egyszerű kezelni, másrészt a négyzetes hibának a hibateljesítmény révén fizikai értelmezés is adható. További hibafüggvények lehetnek még az abszolútérték függvény, különböző abszolútérték-hatvány függvények, az „igen-nem” függvény. Speciális esetekben használnak logaritmikus, trigonometrikus, hiperbolikus függvényekből alkalmasan előállított hibafüggvényeket is.

Gradiens alapú szélsőértékkeresési eljárások

Szélsőérték-keresés paramétereiben linearizálható modellek esetén

Newton-módszer

„Legmeredekebb lejtő” módszer

Konjugált gradiensek módszere

LMS algoritmus

Megerősítéses tanulás

Sztochasztikus szélsőérték-kereső eljárások

Véletlen keresés

Genetikus algoritmusok

Nemellenőrzött tanulás

Nem állnak rendelkezésünkre összetartozó be- és kimeneti adatpárok. A hálózatnak a bemenetek alapján kell valamilyen viselkedést kialakítani. Általában a hálózatoknak valamilyen összefüggéseket, hasonlóságokat kell felderíteniük a bemeneti adatokban. Ezeket a hálózatokat hívják önszerveződő hálózatoknak is.

A neurális hálózatok esetében leggyakrabban alkalmazott ilyen eljárások a Hebb és a versengő tanulás.

Hebb tanulás

Az eljárás biológiai analógián alapszik, és a következőképpen néz ki:

$$w_{ij}(k+1) = w_{ij}(k) + \mu y_i y_j ,$$

Vagyis két processzáló elem közötti súlytényező értéke a processzáló elemek kiemenetének szorzatával arányosan növekszik. A probléma az eljárással, hogy a súlyok a végtelenségig nőhetnének, ezért általában valamilyen normalizált változatát használják a gyakorlatban.

Versengő tanulás

A versengő tanulás esetén a processzáló elemek közül egyet valamilyen szempont szerint kiválasztunk győztesnek, például a győztes kimenete 1 értéket vesz fel, míg a többié 0-t. A versengő tanulás célja a bemeneti mintatér tartományokra osztása úgy, hogy minden egyes tartományhoz tartozó bemenet esetén egy processzáló elem aktiválódjon. A kiválasztás történhet automatikusan is, ilyenkor a processzáló elemek között oldalirányú kapcsolatok vannak.

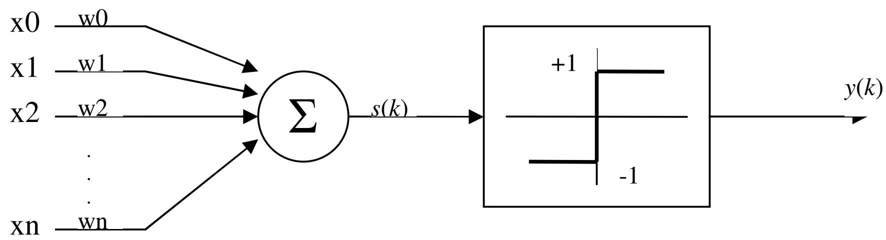
Analitikus tanulás

A megfelelő viselkedésű hálózatok kialakítása elméleti úton, a feladatból meghatározható, tehát valójában nem is tanulás történik ebben az esetben. Ilyenkor a tanító pontokból közvetlenül, analitikus összefüggéseken keresztül határozzuk meg a hálózat súlyait. Némely esetben valamilyen energiafüggvény képezi a súlyok meghatározásának alapját.

6. A kezdetek: Perceptron és Adaline

Az első próbálkozások 1943-ban történtek, amikor McCulloch és Pitts megalkották az első mesterséges neuront, amely a Threshold Logic Unit (TLU) nevet kapta. Ez a bemenetek súlyozott összegét egy küszöbfüggvény nemlinearitáson keresztül vezetve adott eredményt, amely a küszöbfüggvény eltolásától függött.

Az első nagyobb eredmény az 1959-ben Rosenblatt által alkotott perceptron volt, amelyet 1960-ban követett a Widrow és Hoff által javasolt Adaline (adaptive linear element). A perceptron mai formája hasonlatos a TLU-hoz, meg kell jegyeznünk azonban, hogy a Rosenblatt által javasolt forma sokkal bonyolultabb volt.
A perceptron felépítése az alábbiakban látható:



A működést megadó egyenlet:

$$y(k) = f(\sum_i w_i x_i) = \mathrm{sgn}(\sum_i w_i x_i)$$

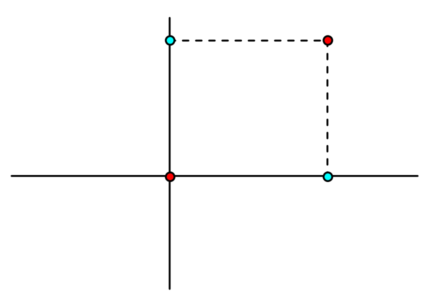
A perceptron tanítása pedig a következőképpen történik:

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \mu\varepsilon(k)\mathbf{x}(k)$$

Nézzük meg egy kicsit részletesebben a perceptron működését. Az ábrából láthatóan, ha az összegző kimenete nagyobb, mint 0, a kimenet +1, ha kisebb, akkor -1 értéket vesz fel, vagyis a perceptron a bemenetére kerülő adatokat két csoportba tudja osztani, vagyis egy kétosztályos lineáris szeparálási probléma megoldására alkalmas. A hiba kifejezésében $\varepsilon(k)$ értéke tehát a várt és tényleges kimenet különbségét adja:

$$\varepsilon = d - y,$$

Ez a kifejezés a [+2,-2] intervallumban vehet fel egész értékeket. A tanulási tényező értéke kisebb vagy egyenlő 1-el. Bizonyítható (a bizonyítást egyszerűsége miatt érdemes elolvasni a hivatkozott irodalomban), hogy a tanítás konvergens. A perceptron, azonban még nem az a megoldás amire mi várunk, hiszen nem minden probléma megoldására képes, sőt valójában még egy egyszerű XOR probléma is kifog rajta:



Amint az ábrán látható a két mintahalmazt nem lehet elválasztani egy egyenessel, vagyis lineárisan nem szeparálható. Általánosságban a következő mondható el a perceptron osztályozó képességéről (kapacitásáról).

$$C(P,N)=\begin{cases}1 & \text{ha } P\le N\\ \dfrac{2}{2^P}\displaystyle\sum_{i=0}^{N-1}\binom{P-1}{i} & \text{ha } P>N\end{cases}$$

P a véletlenszerűen kiválasztott pontok száma, *N* a dimenziószám, és *C*(*P*,*N*) jelöli a perceptron kapacitását. Tehát ha a mintahalmaz nagysága kisebb, mint a dimenziószám, akkor biztos, hogy a perceptron képes szeparálni. Ez két dimenzió, vagyis sík esetén azt jelenti, hogy már 3 pont kiválasztása esetén sem biztos a lineáris szeparálhatóság.

Az adaline hasonló a perceptronhoz azzal a különbséggel, hogy a hibát a nemlinearitás előtt képezzük, és az LMS algoritmussal tanítunk

7. Többrétegű perceptron hálózatok (MLP - multi-layer perceptrons)

A gyakorlatban legtöbbször alkalmazott struktúra. A neurális hálózatok újrafelfedezése akkor kezdődött, amikor a 80-as évek elején felfedezték a tanítására alkalmas hibavisszaterjesztéses algoritmust. A többrétegű perceptron hálózatokban perceptronokat kötünk össze, azonban az alkalmazott nemlinearitás valamilyen folytonosan differenciálható függvény (pl. szigmoid típusú). A súlymódosítást ebben az esetben szintén az LMS algoritmussal végezzük.
A többrétegű hálózatok egyes rétegeiben egyszerű perceptronok vannak összekötve. A perceptronok kiemenete, mint említettük, folytonos, mindenütt diferenciálható nemlinearitás. A többrétegű hálózatok rétegeinek neuronszáma eltérő lehet.
A többrétegű hálózatok tanítási összefüggései a következők:
A kimeneti réteg esetén:

$$\mathbf{W}^{(L)}(k+1)=\mathbf{W}^{(L)}(k)+2\mu\varepsilon_i(k)f'(s_i^{(L)})\mathbf{x}^{(L)}(k)=\mathbf{W}^{(L)}(k)+2\mu\delta_i^{(L)}(k)\mathbf{x}^{(L)}(k)$$

A rejtett rétegek esetén:

$$\mathbf{W}^{(l)}(k+1)=\mathbf{W}^{(l)}(k)+2\mu\delta^{(l)}\mathbf{x}^{(l)^T}(k)$$
$$\delta_i^{(l)}(k)=\left(\sum_{r=1}^{N_{l+1}}\delta_r^{(l+1)}(k)w_{ri}^{(l+1)}(k)\right)f'\left(s_i^{(l)}(k)\right)$$

A $\delta_i^{(l)}$ értékeke az *L*. rétegtől kezdve számítandók visszafelé.

A többrétegű hálózatok alkalmazásánál több olyan fontos problémába ütközünk, amely fontos a neurális hálózatok témakörében. Általános jellemző, hogy a felmerülő kérdésekre nem adható egzakt válasz, a legtöbb esetben intuitív módon, előzetes tapasztalatokra alapozva válaszoljuk meg a felmerülő kérdéseket.

A hálózat méretének megválasztása
A rétegek számának, az egyes rétegeken belüli neuronok számának meghatározására nincs egyértelmű módszer. Általánosan elmondható, hogy a több réteg alkalmazása könnyítheti a feladat megoldását, illetve rétegenként kevesebb processzáló elem is elegendő lehet. Kétféle módszer terjedt el a hálózat méretének meghatározására. Az első szerint egy nagyobb hálózatból kiindulva próbáljuk meg csökkenteni annak méretét, a másik szerint egy kisebb hálózat bővítésével próbálunk eljutni a megfelelő megoldásig.

A tanulási aránytényező megválasztása
Nagyobb értékek választásakor a tanulás konvergenciája gyorsabb lehet, de a minimumhely közelében problémák mutatkozhatnak, sőt egyes esetekben divergenciához is juthatunk. Kisebb aránytényező választása esetén a konvergencia lassabb, de a minimumhely megtalálása pontosabban lehetséges. Általában célszerű adaptív megoldásokat alkalmazni, ahol a tanulási aránytényező lépésenként változik, sőt akár rétegről rétegre is.

A kezdeti súlyok megállapítása
Amennyiben rendelkezünk a problémáról valamilyen előzetes ismerettel, úgy a kiindulási súlyokat érdemes ennek megfelelően megválasztani. Általában azonban ilyen ismereteink nincsenek, így a legmegfelelőbb a kezdeti súlyok véletlenszerű meghatározása. Amennyiben a tanítás sikertelen, célszerű más súlyokkal megpróbálni.

A tanító lépések száma
A tanító lépések száma nem határozható meg előre, azonban figyelnünk kell arra, hogy a hálózat túltanulhatja magát, ami az általánosítóképesség rovására megy.

A megtanított hálózat minősítése
A megtanított hálózat minősítése a hálózat ismeretlen adatokra adott hibája alapján lehetséges. Amennyiben a hiba elfogadható a tanítás (a struktúra megválasztása) sikeres volt.

A tanító mintahalmaz
A hálózat tanításához szükséges minták száma természetesen függ a feladattól, de elmondható, hogy a legtöbb esetben kevés minta áll rendelkezésre. A tanító mintakészletet több részre szokták osztani, amelyek a hálózat tanításánál, és a tanítás sikerességének minősítésénél lehetnek fontosak. A tanító készletet használjuk a tanításra, míg a kiértékelő mintakészletet a tanítás kiértékelésére. A hálózat általánosítóképessége miatt várható, hogy ezeket a mintákra is helyes választ ad. A kiértékelő mintakészlet felhasználása is lehetséges a tanítás során. Szokás még egy teszt mintakészlet meghatározása is, amelyet a tanításnál egyáltalán nem használunk fel. Kevés adat esetén a csoportok elemeit cseréljük (cross validation), és minden mintát felhasználunk tanításra és kiértékelésre is.

Backpropagation változatok
A tanítási algoritmus előbb bemutatott változata viszonylag lassú konvergenciát biztosít. A konvergencia gyorsítására, a lokális minimumba való beragadás elkerülésére különböző módszereket dolgoztak ki.
Konjugált gradiens módszer
Levenberg-Marquardt eljárás

Kalman-szűrő alapú eljárás
Momentum módszer
Felejtő tanulás

8. Egyéb hálózati struktúrák

Ellenőrzött tanítású hálózatok
RBF (Radial Basis Function) hálózat
CMAC (Cerebellar Model Articulation Controller) hálózat
CPN (Counterpropagation) hálózat
PNN (Probabilistic Neural Network) hálózat
GRNN (Generalized Regression Neural Network) hálózat
Különböző visszacsatolt hálózati struktúrák
Előrecsatolt hálózati struktúrák FIR szűrőkkel, vagy késleltetett bemenetekkel

Nemellenőrzött tanítású hálózatok
Kohonen hálózat (SOM)
Altér hálózatok (Oja, Földiák)
Főkomponens hálók (APEX, Sanger) (PCA)
Független komponens analízis hálózatok (ICA)

Analitikus tanítású hálózatok
Hopfield hálózat
Boltzmann gépek
Mean-field hálózatok
CNN hálózat

Források:
1. Horváth Gábor (szerkesztő): Neurális hálózatok és műszaki alkalmazásaik, Műegyetemi Kiadó, Budapest, 1998. 6-112. oldal.
2. http://www.icos.ethz.ch/teaching/NNcourse/brain.html
3. http://www.shef.ac.uk/psychology/gurney/notes/l1/l1.html