

Osztályozási feladatok képdiagnostikában

Orvosi képdiagnostikai 2017 ősz

Osztályozás

- Szeparáló felületet keresünk
- Leképezéseket tanulunk meg azok mintáiból
 - A tanuláshoz használt minták a tanító minták / pontok
 - A leképezés folytonos / diszkrét bementből képez diszkrét és véges számú halmazba
$$f(\mathbf{x}_i) = d_i, d_i \in \{1, 2, \dots, k\} : \forall i \in \{1, 2, \dots, P\}$$
 - Feltételezhetünk még bemeneti / kimeneti zajt:
$$f(\mathbf{x}_i + \varepsilon_{b,i}) + \varepsilon_{k,i} = d_i$$
 - $f(\cdot)$ szerint lehet lineáris / paramétereiben lineáris / nemlineáris

Terminológia, jelölések

- Osztályozó:
 - $f : \mathbb{R}^N \times \Theta \rightarrow \{1, 2, \dots, d\}$
- Veszteségfüggvény (loss function, hibafüggvény)
 - $L(d, y) \propto -\log(p(y | d))$
 - általában közvetlenül definiáljuk p kihagyásával
- Tanítás:
$$\min_{\mathbf{w} \in \Theta} R(\mathbf{w}) = \mathbb{E}_{(\mathbf{x}, d) \in D} \{L(d, f(\mathbf{x}, \mathbf{w}))\}$$

s.t. $\mathbf{w} \in \Theta$
- Tapasztalati kockázat:
 - $R_{emp}(\mathbf{w}) = \sum_i L(d_i, f(\mathbf{x}_i, \mathbf{w}))$

Orvosi Döntéstámogató Rendszerek

- Computer Aided Detection
 - Cél a leletezések specificitásának, és érzékenységének a növelése
 - Számos megbetegedés esetén már jóval annak diagnosztizálása előtt láthatóak annak jelei
 - Megvalósításuk tanuló rendszerrel történik
 - Klasszikus megközelítés: probléma specifikus jel / képfeldolgozás állítja elő az osztályozás bemeneteit
 - Újabb megközelítés: kimarad a manuális jelfeldolgozás, az MI-re bizzuk a teljes folyamatot

Empirikus kockázatminimalizálás

- Elv: a tapasztalati kockázat (tanítóminták osztályozási hibájának) minimalizálásával minimalizálható a kockázat (valódi minták osztályozási hibája), ha közben rögzítjük az osztályozó VC-dimenzióját:
- Osztályozó hibája $1-\eta$ valószínűséggel

$$R(\mathbf{w}) \leq R_{emp}(\mathbf{w}) + \frac{\varepsilon(h)}{2} \sqrt{1 + \frac{4R_{emp}(\mathbf{w})}{\varepsilon(h)}}$$

$$\varepsilon(h) = 4 \frac{h(\ln(2l/h) + 1) - \ln(\eta/4)}{l}$$

l : tanítópontok száma
 h : osztályozó VC dimenziója
(osztályozó komplexitása)
 $\hat{f}(\cdot)$: megtanult leképezés

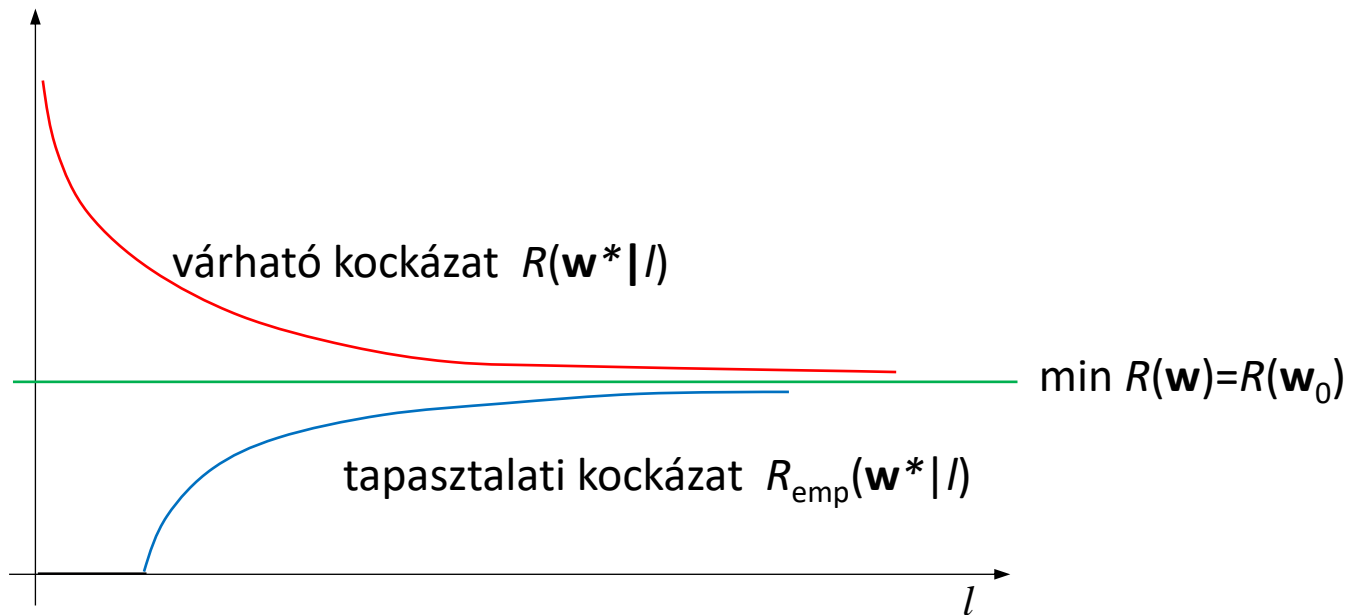
D : mintákat generáló
eloszlás

Empirikus kockázatminimalizálás

- Fontos, hogy ugyanazon háttéreloszlás mintái adják az éles (valódi) és a tanítómintákat
- VC dimenzió:
 - Maximum hány olyan tetszőleges elrendezésű és osztályozású mintapont létezik, melyet az osztályozó képes hibátlanul (adott osztályozás szerint) szeparálni.
- Cél a kockázat ($R(\mathbf{w})$) minimalizálása – inverz probléma
 - Azon belül is leginkább MAP becslés
 - Likelihood tag: tanítóminták osztályozási hibáját bünteti
 - Regularizációs tag: osztályozó VC dimenzióját bünteti

Osztályozók általánosítási hibája

- Lényegében $R(\mathbf{w}) - R_{emp}(\mathbf{w})$:
 - Tanítóminták számának növelésével csökken
 - VC dimenzió növekedésével nő



Példák osztályozókra

- Lineáris / Kernel diszkrimináns analízis
- Rosenbladt perceptron
- Adaline
- Neurális háló (MLP – multi layer perceptron)
- Bázisfüggvényes lineáris hálók (pl. RBF, CMAC)
- Mély neurális hálók
 - Konvolúciós neurális hálók

Lineáris (Fischer) diszkrimináns analízis

- Cél olyan vetítőirány keresése, mely mentén maximális

$$C(\mathbf{w}) = \frac{\left(\mathbf{w}^T (\mathbf{m}_2 - \mathbf{m}_1)\right)^2}{\sum_{k \in \{1,2\}} \sum_{i \in P_k} \left(\mathbf{w}^T (\mathbf{x}_i - \mathbf{m}_k)\right)^2}$$

$$\mathbf{m}_k = \mathbb{E}_{i \in P_k} \{\mathbf{x}_i\}$$

- Vezessük be az alábbi jelöléseket:

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1) \cdot (\mathbf{m}_2 - \mathbf{m}_1)^T$$

$$\mathbf{S}_{wk} = \sum_{i \in P_k} (\mathbf{x}_i - \mathbf{m}_k) (\mathbf{x}_i - \mathbf{m}_k)^T$$

$$\mathbf{S}_W = \sum_k \mathbf{S}_{wk}$$

Lineáris (Fischer) diszkrimináns analízis

- Egy Rayleigh hányadost minimalizálunk

$$\arg \max \left\{ \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} \right\} = \arg \max \left\{ \frac{\mathbf{w}^T \mathbf{S}_W^{-1} \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \right\}$$

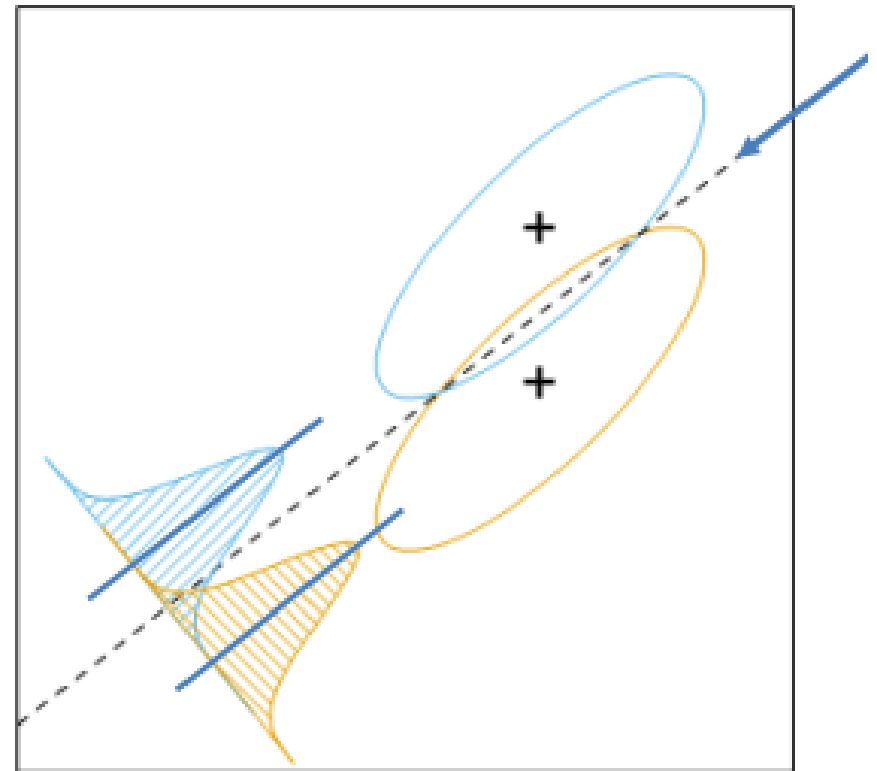
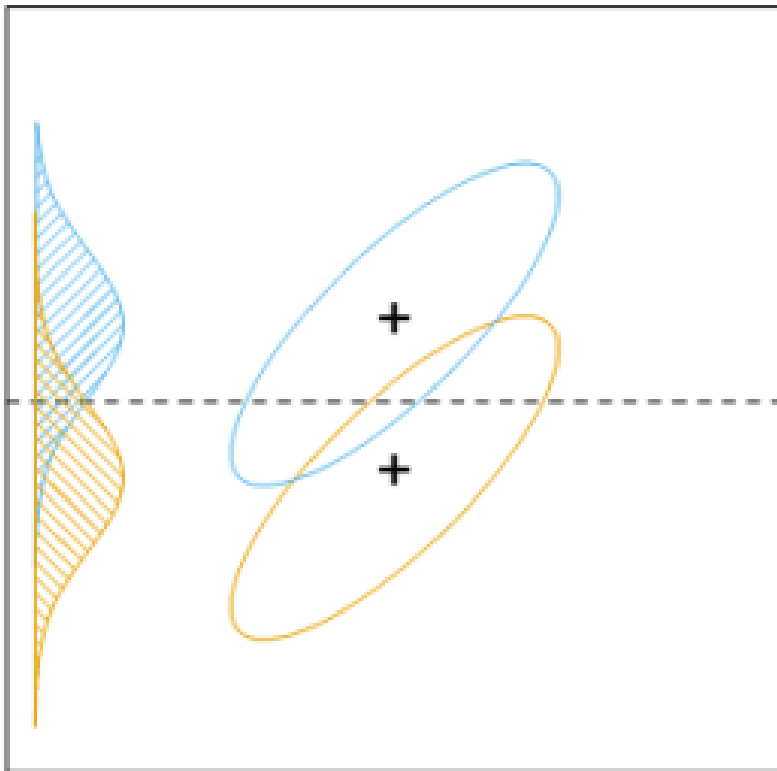
– Ekvivalens optimalizálási probléma

$$\begin{aligned} \max. \quad & \mathbf{w}^T \mathbf{S}_W^{-1} \mathbf{S}_B \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T \mathbf{w} \leq 1 \end{aligned}$$

- Lagrange duális nyeregpontjában: $\mathbf{S}_W^{-1} \mathbf{S}_B \mathbf{w} = \lambda \mathbf{w}$
- Tehát a legkisebb sajátérték, meg hozzá tartozó sajátvektor a megoldás (gyakorlatban azonban nem szükséges a sajátérték egyenlet megoldása)

Lineáris (Fischer) diszkriminációs analízis

- Tekinthető a PCA osztályozási feladatokra adekvát változatának is – ez is dimenziót redukál



Adaptív lineáris neuron

- Négyzetes veszteségfüggvény alkalmazásával

$$\mathbf{y} = \mathbf{X}\mathbf{w} ; \quad \mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ 1 & 1 & \cdots & 1 \end{bmatrix}^T ; \quad R_{emp}(\mathbf{w}) = \|\mathbf{d} - \mathbf{X}\mathbf{w}\|_2^2$$

- Analitikus megoldás: $\mathbf{w}^* = \mathbf{X}^\dagger \mathbf{d} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{d}$
- Ha jobb általánosító képességet akarunk:
 - Csonkoljuk $\mathbf{X}^\dagger = \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{U}^*$ esetén a szinguláris értékeket
 - Folytonos átmenete – Thikhonov regularizáció:
$$\mathbf{X}^\dagger = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T$$

Adaptív lineáris neuron

- Iteratív optimalizáció – Least Mean Squares

- Pillanatnyi négyzetes hiba:

$$C(\mathbf{w}(k)) = \varepsilon^2(k) = \left(d_{i(k)} - \mathbf{x}_{i(k)}^T \mathbf{w}(k) \right)^2$$

- Sztochasztikus Gradiens optimalizáció – μ -LMS

- Négyzetes hiba gradiensét a pillanatnyi négyzetes hiba gradiensével becsüljük:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu \nabla C(k) = \mathbf{w}(k) - 2\mu \varepsilon(k) \mathbf{x}_{i(k)}$$

- SGD kondicionálással – α -LMS

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \alpha \frac{\varepsilon(k)}{\mathbf{x}_{i(k)}^T \mathbf{x}_{i(k)}} \mathbf{x}_{i(k)} \quad 0 < \alpha < 2$$

Bázisfüggvényes osztályozók

- Problémák nagy része nem lineárisan szeparábilis.
- Paramétereiben lineáris, nemlineáris osztályozók:

$$f(\mathbf{x}, \mathbf{w}) = \varphi(\mathbf{x})^T \cdot \mathbf{w} = y$$

- Lényegében $\varphi(\cdot)$ (ú.n. jellemzőtérbe transzformáló leképezés) határozza meg, hogy mire képes az osztályozó
- $\varphi(\cdot)$ is tanulható (pl. OLS eljárás RBF hálók esetén)
- Ha nem inkonzisztensek a tanítóminták, akkor mindig létezik olyan $\varphi(\cdot)$, mellyel lineárisan szeparábilis a probléma
- Cserébe baj lehet a VC dimenzióval / háló általánosító képességével

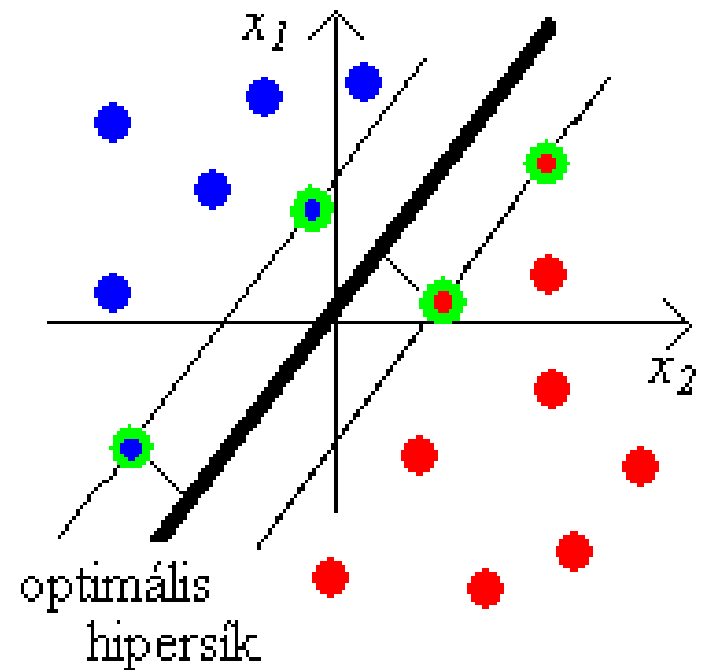
Szupport vektor gépek

- Keressük az alábbi szeparáló síkot:

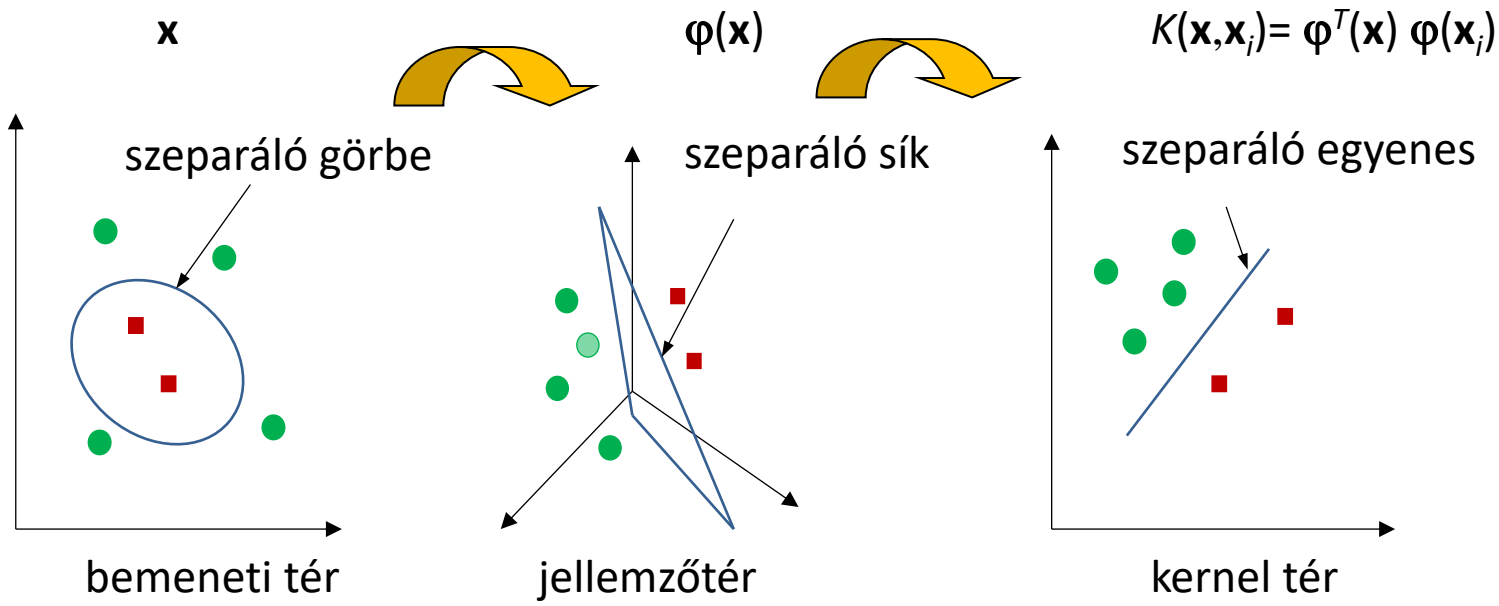
$$\min. \quad 1/2 \mathbf{w}^T \mathbf{w}$$

$$s.t. \quad d_i \left(\mathbf{w}^T \varphi(\mathbf{x}_i) + b \right) \geq 1$$

- Lényegében Thikhonov regularizálunk (hasonlóan, ahogy azt az SART / SIRT-nél tettük)
- Lagrange duális alapján:
 - Megússzuk $\varphi(\cdot)$ definiálását
 - Helyette $K(\mathbf{x}, \mathbf{x}_i) = \varphi(\mathbf{x})^T \varphi(\mathbf{x}_i)$ úgynevezett kernelt kell definiálnunk (Kernel trükk)
 - Végtelen dimenziós φ is lehet véges VC dimenzió mellett



Szupport vektor gép kernel leképezése



$$y = \sum_{j=0}^M w_j \varphi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x})$$

$$\mathbf{w} = [w_0, w_1, \dots, w_M]^T$$

$$\boldsymbol{\varphi} = [\varphi_0(\mathbf{x}), \varphi_1(\mathbf{x}), \dots, \varphi_M(\mathbf{x})]^T$$

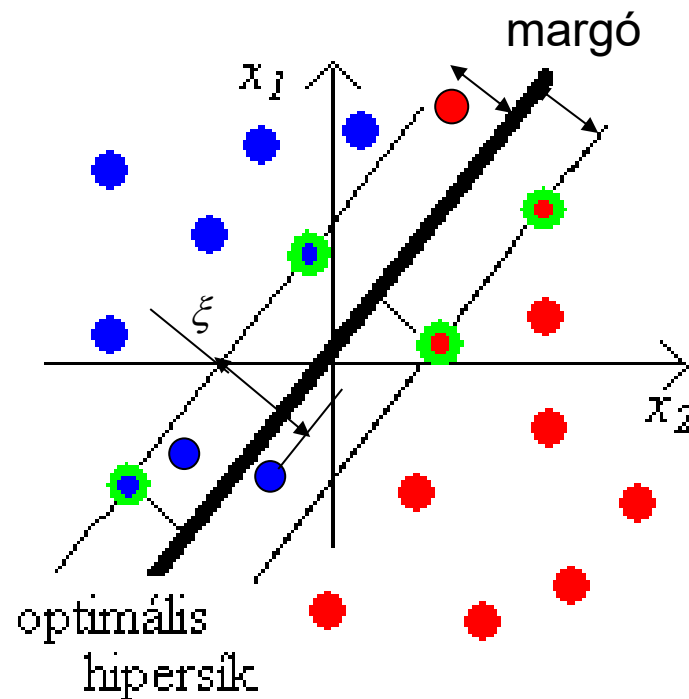
Szupport vektor gépek - gyengítés

- Keressük az alábbi szeparáló síkot:

$$\min. \quad 1/2 \mathbf{w}^T \mathbf{w} + c \cdot \mathbf{1}^T \cdot \xi$$

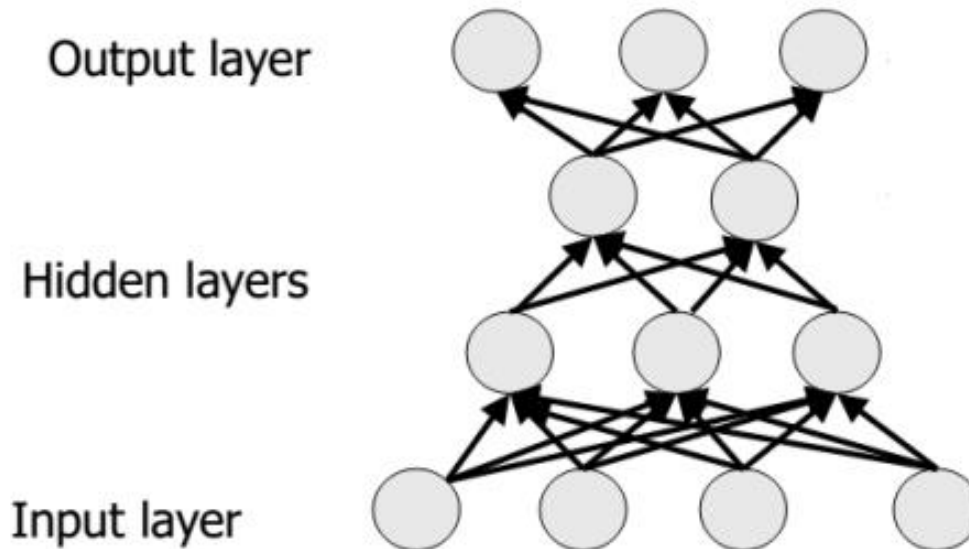
$$s.t. \quad d_i \left(\mathbf{w}^T \varphi(\mathbf{x}_i) + b \right) \geq 1 - \xi_{(i)} \quad \xi \in \mathbb{R}_+^{|P|}$$

- Megengedjük, hogy bizonyos minták belógjanak a margóba, esetleg rosszul osztályozzuk őket
- c lényegében egy trade – off szabályozó az empirikus kockázat (likelihood tag, gyengítő változó), és a regularizációs ($\|\mathbf{w}\|_2^2$) tag súlya között.
- Lényegében $R(\mathbf{w})$ tagjainak súlyát szabályozzuk

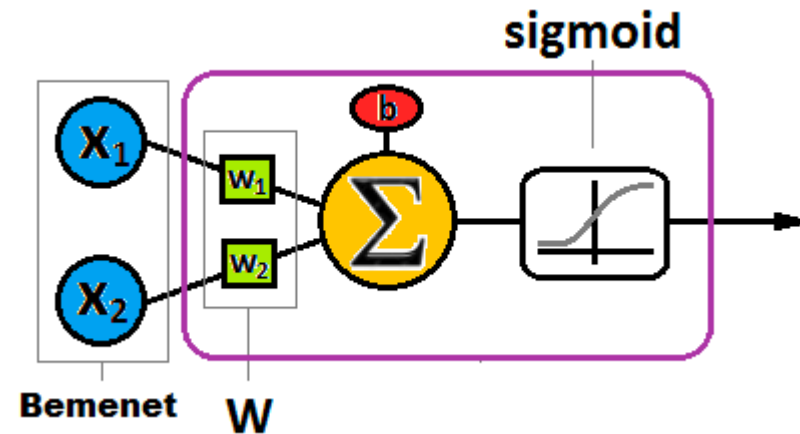


Többrétegű Perceptron (MLP)

- Klasszikus neurális hálónak is hívják



Egy neuron felépítése:



– Nem linearitásokra példák:

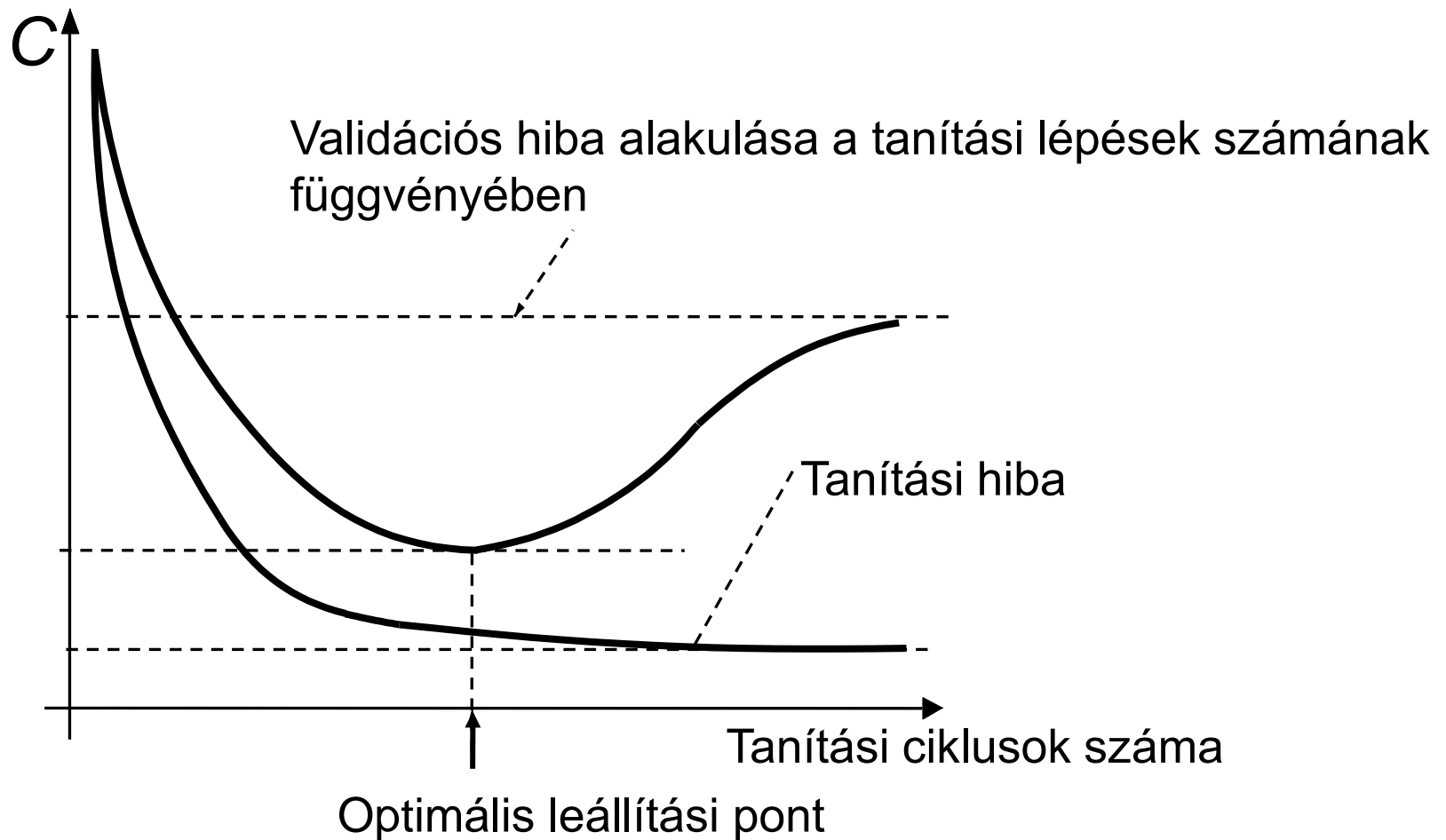
- Pl. tanh , Logisztikus szigmoid ($1/(1 + \exp(-s))$)
- Kritérium, hogy gradiens módszeres optimalizációnál (hiba visszaterjesztésnek hívjuk) ne „sikkassa el” a hibát

Többrétegű Perceptron (MLP)

- Hány rejtett réteg legyen
 - Egy darab elegendően sok (de véges számú) neuronnal elég – univerzális approximátor
 - Mostanában mégis az 1-nél jóval több a népszerű
 - Összetettebb leképezésekhez így kevesebb neuron elég
 - Eltérő típusú rejtett rétegek
- Egy rétegen belül hány neuron legyen
 - Elegendően sok a problémához
 - De ne legyen túl sok
 - Túl sok szabad paraméter, nagyobb VC dimenzió

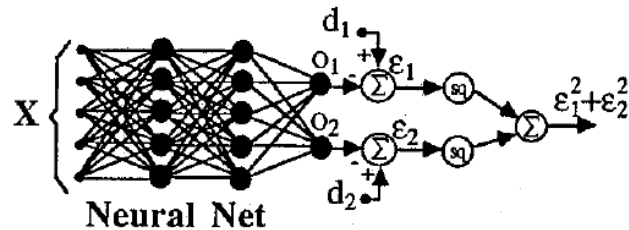
MLP tanítása

- Korai leállítás (a túltanulás elkerüléséért):

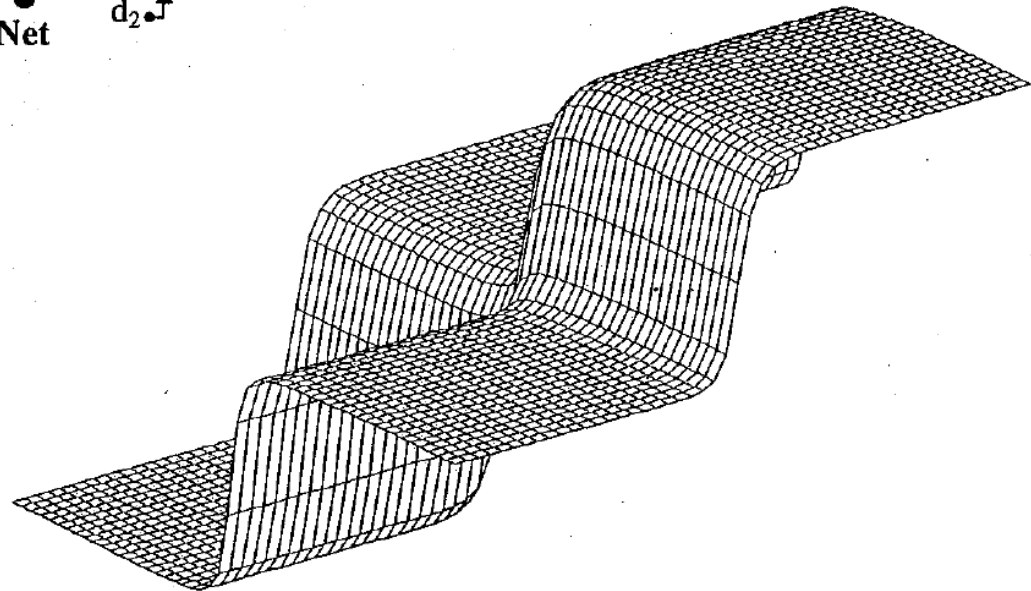


MLP hibafelület példa (1)

Ellaposodó /
felrobbanó
hibafelület



- 4 inputs
- 3-layer network: 5 feed 8 feed 2
- sigmoids

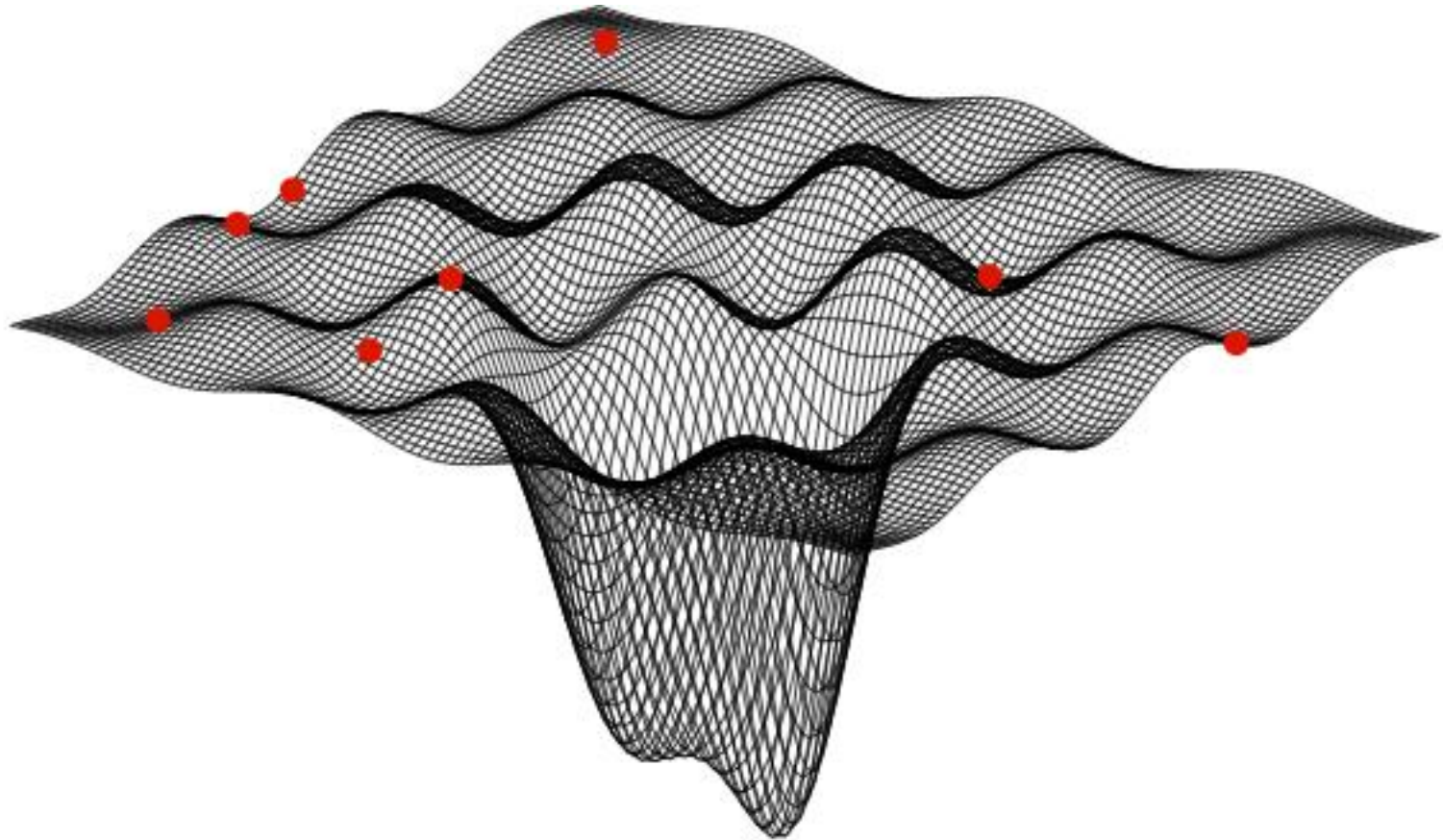


EXPERIMENT 3a.
Randomize weights,
then vary two
first-layer weights.

Mean Square Error Surface

MLP hibafelület példa (2)

- Sok a lokális szélsőérték



Objektum / mintafelismerés megközelítése

Traditional Pattern Recognition: Fixed/Handcrafted Feature Extractor



Mainstream Pattern Recognition

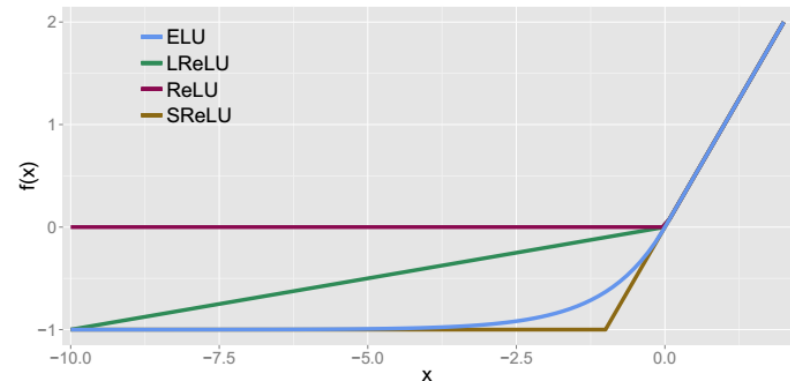


Deep Learning: Multiple stages/layers trained end to end



Konvolúciós neurális hálózat (CNN)

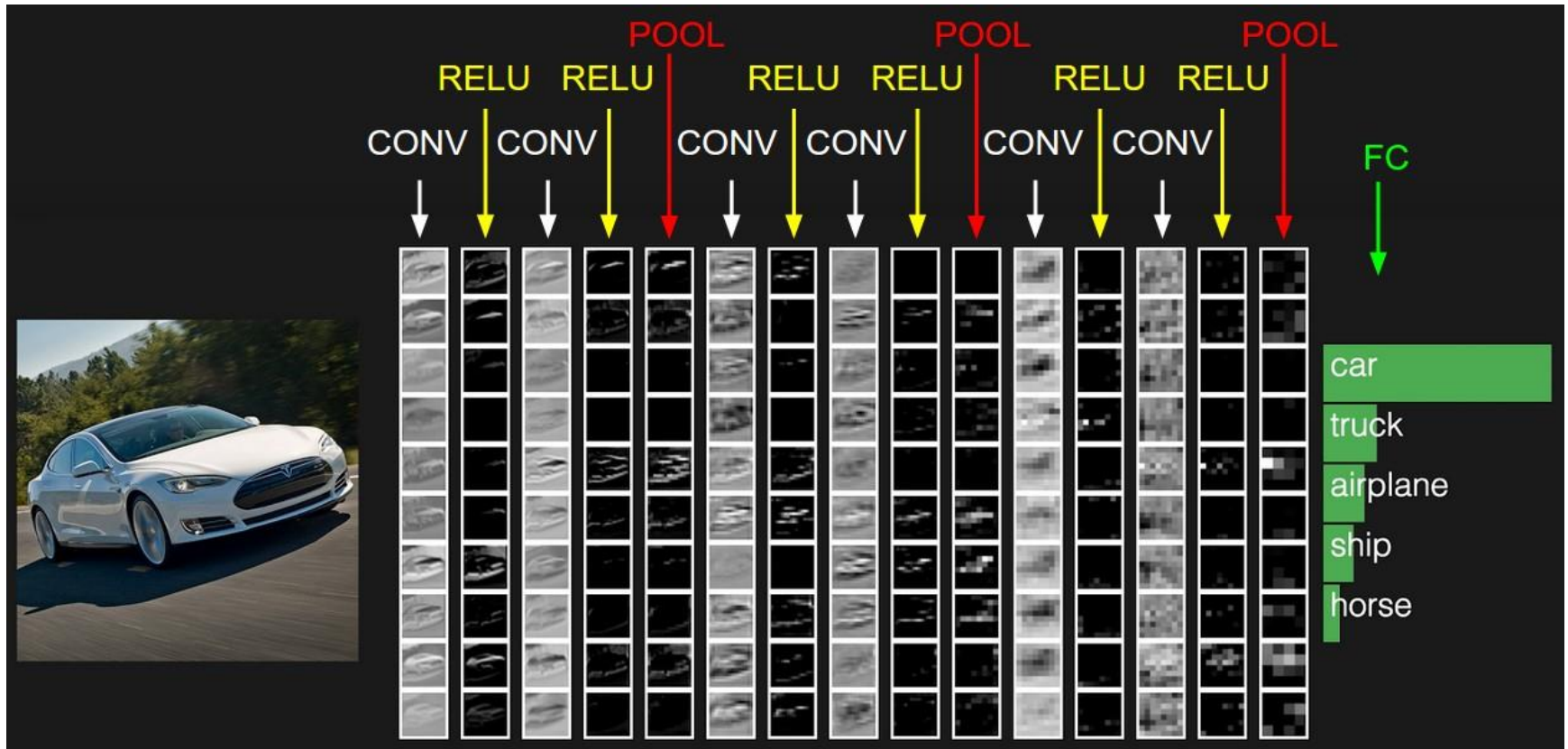
- Alapötlet – a szűrőkernelt is tanulja meg a háló:
 - Így nem kell megtippelni, hogy mi alapján ismerünk fel egy adott objektumot
 - A neurális háló bizonyos rétegei LSI lineáris rendszerek kernelét tanulják
- Cserébe nagy idő és memória igény:
 - Mélyebb hálók – kisebb kernel is elég
 - Viszont a klasszikus MLP-nél használt nem-linearitások „elnyelik” a hibát
 - Újítás nem-linearitások:



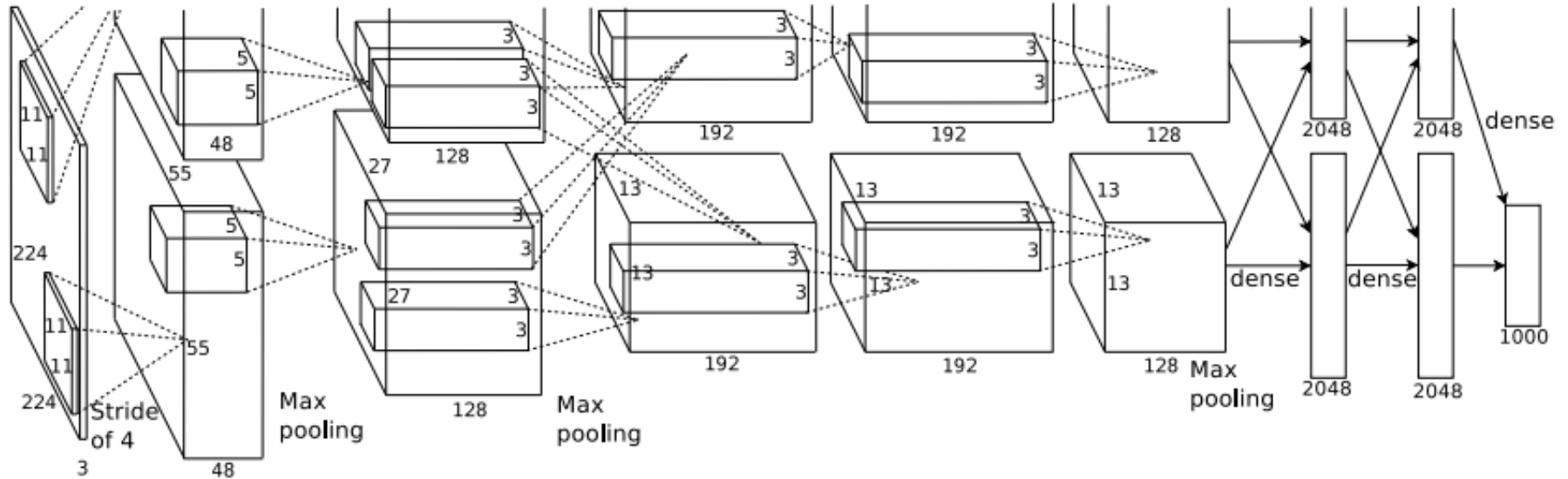
CNN

- 4 különböző típusú réteg:
 - Konvolúciós: kernelt tanul (tipikusan 3×3 , 5×5)
 - Paraméterek: kernel méret, stride, kép szélének kezelése
 - Nem-linearitás:
 - ReLU, szivárgó ReLU, zajos ReLU, MaxOut, ...
 - Pooling: szűrt képek alul-mintavételezése
 - Általában nemlineáris (maximum pooling)
 - Paraméterek számát korlátozza, de a bijektív jelleg biztosan elveszik – jelentősen nehezíti az interpretációt
 - Fully Connected Layer: klasszikus MLP
 - Osztálycímkeket ettől várjuk

CNN példák



CNN példák

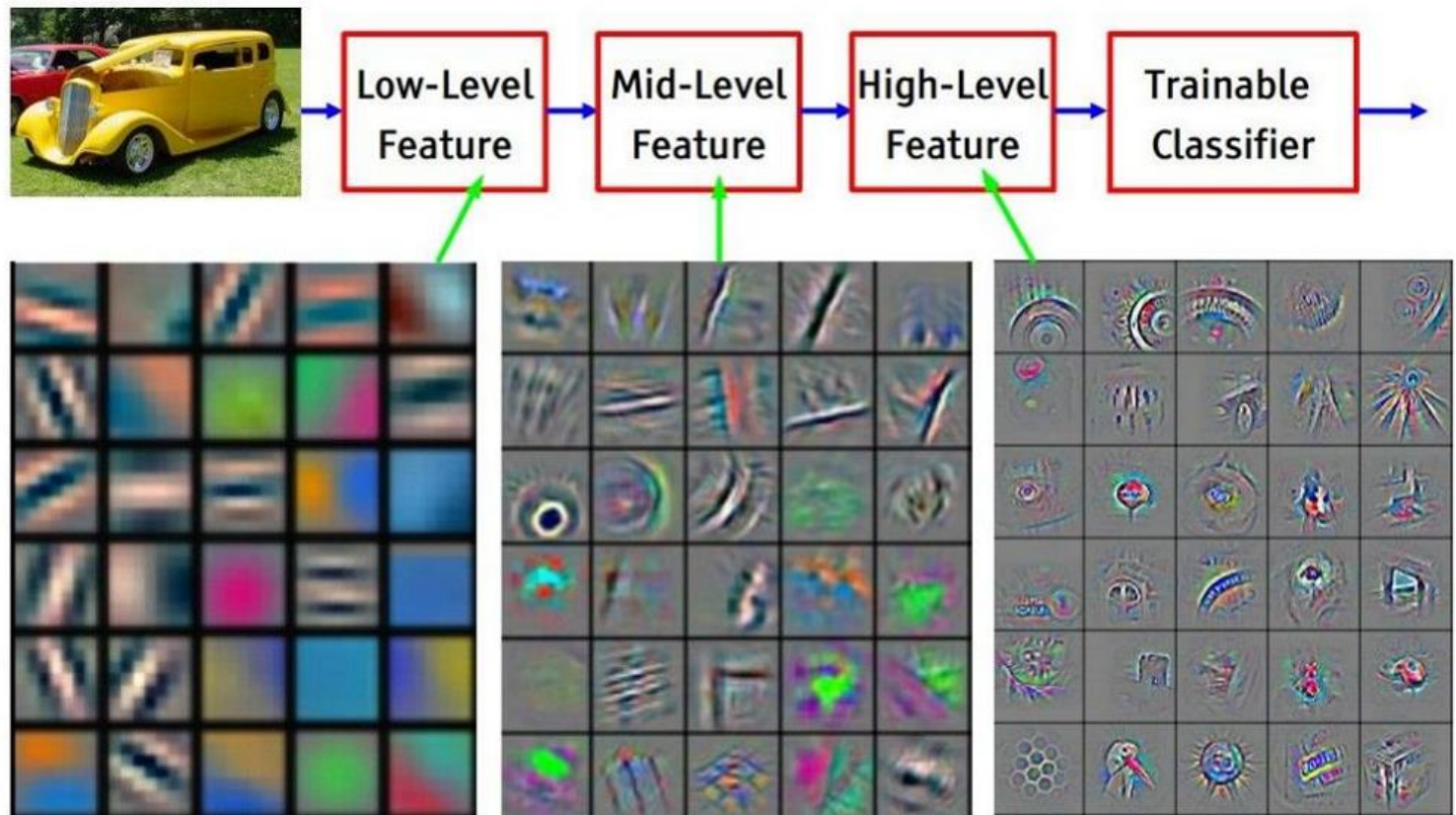


Bemenet: 150.528-dimenziós

a neuronok száma a háló további rétegeiben: 253.440; 186.624;
64.896; 64.896; 43.264; 4096;4096; 1000.

CNN példák

A szűrők, mint tulajdonság-érzékelők



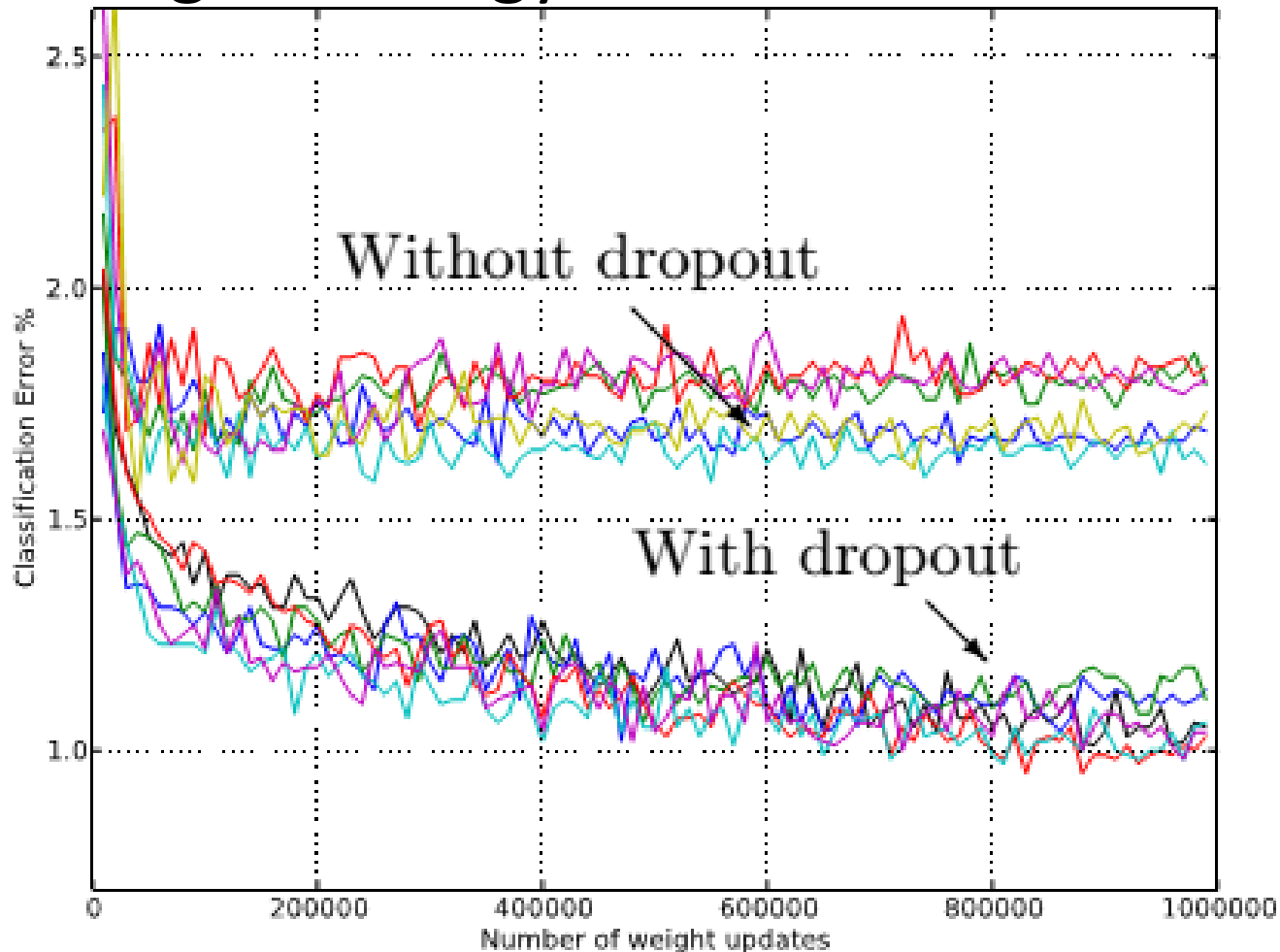
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

CNN regularizáció - dropout

- Implicit regularizáció – Dropout:
 - P valószínűséggel kihagyja a neuronokat adott epoch-ban
 - Robosztus, redundáns hálót hoz létre. Ennyire sok paraméternél hatásosabb a konvencionális módnál
 - Kieszközöli, hogy olyan feature-öket tanuljon a háló, melyek használhatóak akkor is, ha bizonyos feature-ök hiányoznak.
 - Könnyen számolható
 - Elméleti oldalról paraméter megosztásos szakértő együtttest generál
 - Gyakorlatban $P=0,5$ esetén duplázza a konvergenciához szükséges iteráció számot.

CNN regularizáció - dropout

- Tanulási görbékre gyakorolt hatás:



CNN tanítása

- SGD, illetve annak különböző variánsai
 - Sztochasztikus jelleg a mini-batch-ből következik: tanító minták egy részhalmaza alapján becsli adott epochban a háló hibáját
 - Nagy rétegszám – jelentősen le tud lassulni a tanulás kedvezőtlen inicializáció esetén
 - Batch normalizáció:
 - Minden réteg bemenetét 0 várható értékűvé, 1 szórásúvá transzformálja, majd ez alapján módosítja a hálót – elkerülhető a gradiens eltűnés / robbanás
 - Jelentősen gyorsítja a konvergenciát

CNN tanítása – transfer learning

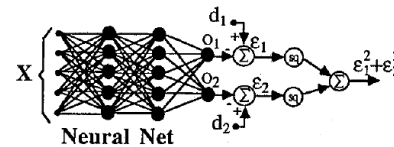
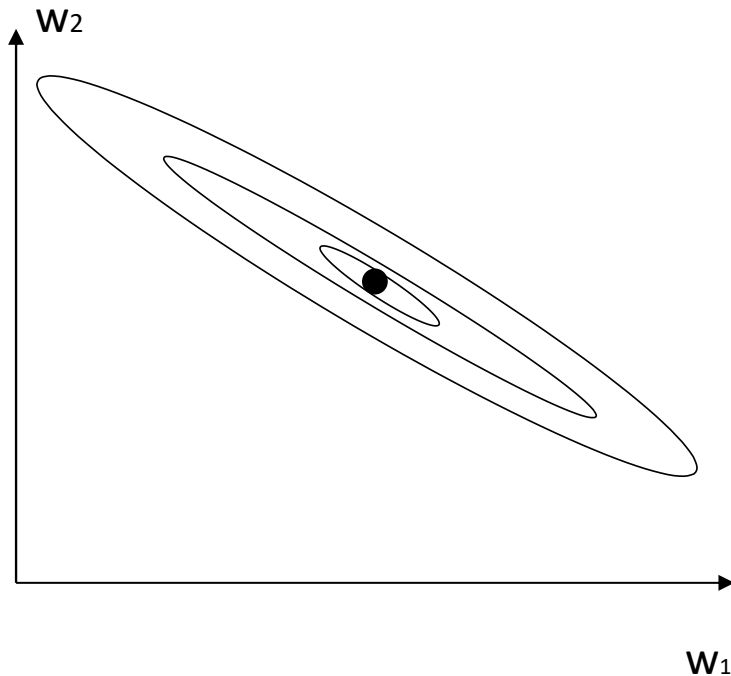
- Sok paraméterhez sok tanítóminta szükséges
 - Isd. $R(\mathbf{w})$ majorálása
 - Alacsonyabban lévő rétegek kis absztrakciójú jellemzőket detektálnak (pl. él, sarokpont. stb.)
 - A probléma specifikusabb jellemzők az FC réteg bemenetének közelében állnak elő
 - Transfer learning: Csak az FC-t tanítjuk, jellemző kiemelő rétegek paramétereit más (több mintás) problémára tanítjuk
 - Fine tuning: Csak a kimenethez közeli jellemző kiemelő rétegeket tanítjuk
- Minta augmentáció:
 - Olyan torzításokkal generálunk még mintákat, melyekre irreleváns osztályozást várunk el (pl. elforgatás, stb.).

Optimalizáló eljárások

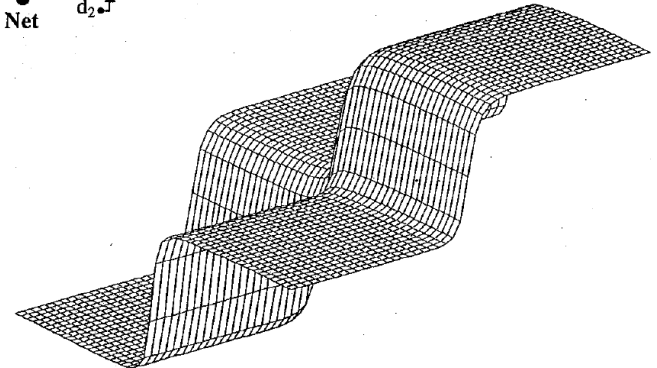
- Másodrendű módszerek:
 - Newton módszer: hibafüggvény lokális másodfokú közelítését minimalizálja iterációnként
 - Kvázi Newton módszer: Hesse mátrixot / annak invertálását csak közelíti (pl. BFGS, L-BFGS)
 - Levenberg Marquardt: regularizált Newton / két megközelítés kombinációja
- Elsőrendű módszerek:
 - Konjugált gradiens: konjugált irányú hibajavítás, (kvadrátikus függvény esetén másodrendű módszer)
 - Nesterov Momentumos / adaptív gradient descent eljárások
 - Sztochasztikus gradiensek esetén hatékonyabbak

Elsőrendű optimalizáló eljárások

- Rosszul kondicionált hibafelületek – bajban a GD:
 - Ellaposodó platók / hirtelen letörések
 - Erősen nem izotropikus felületek



- 4 inputs
- 3-layer network: 5 feed 8 feed 2
- sigmoids



EXPERIMENT 3a.
Randomize weights,
then vary two
first-layer weights.

Mean Square Error Surface

Elsőrendű optimalizáló eljárások

- Ötlet: szűrjük a súlymódosító vektorokat
 - Momentum alapú módszerek – simítja a lépés irányát az iterációk felett (tehetetlenségi szemléltetés)
 - Nesterov momentum asszimptotikusan optimális elsőrendű eljárás, ha gradienst és nem sztochasztikus gradienst használunk (tehát itt általában nem az)
 - Adagrad / RMSProp: átlagolja az elmúlt iterációkon belüli paraméter módosításokat
 - Adaptive Momentum (Adam): kombinálja az előző két módszert

Elsőrendű optimalizáló eljárások

