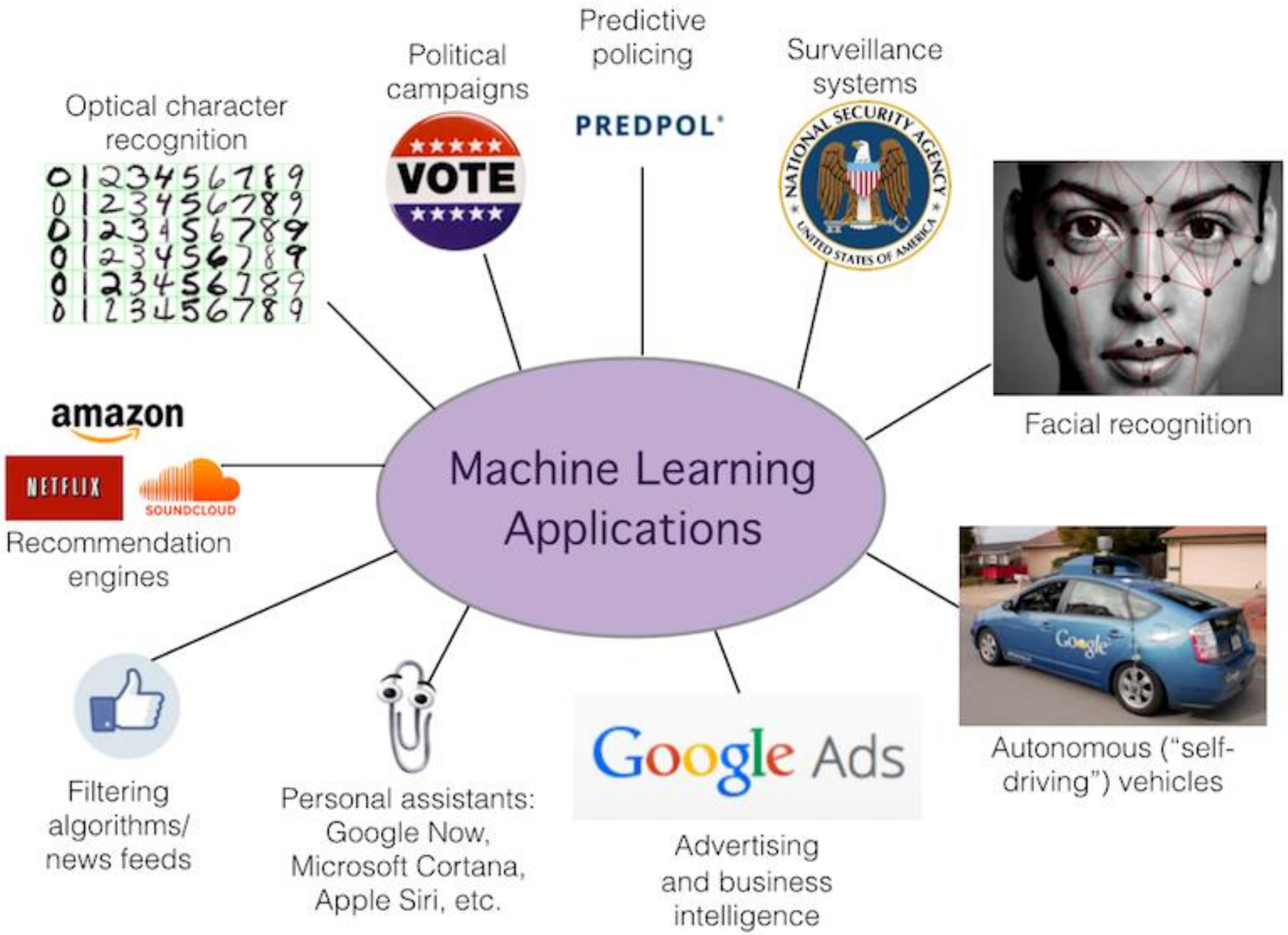
MACHINE LEARNING, NEURAL NETWORKS, DEEP LEARNING

Bagoly Attila

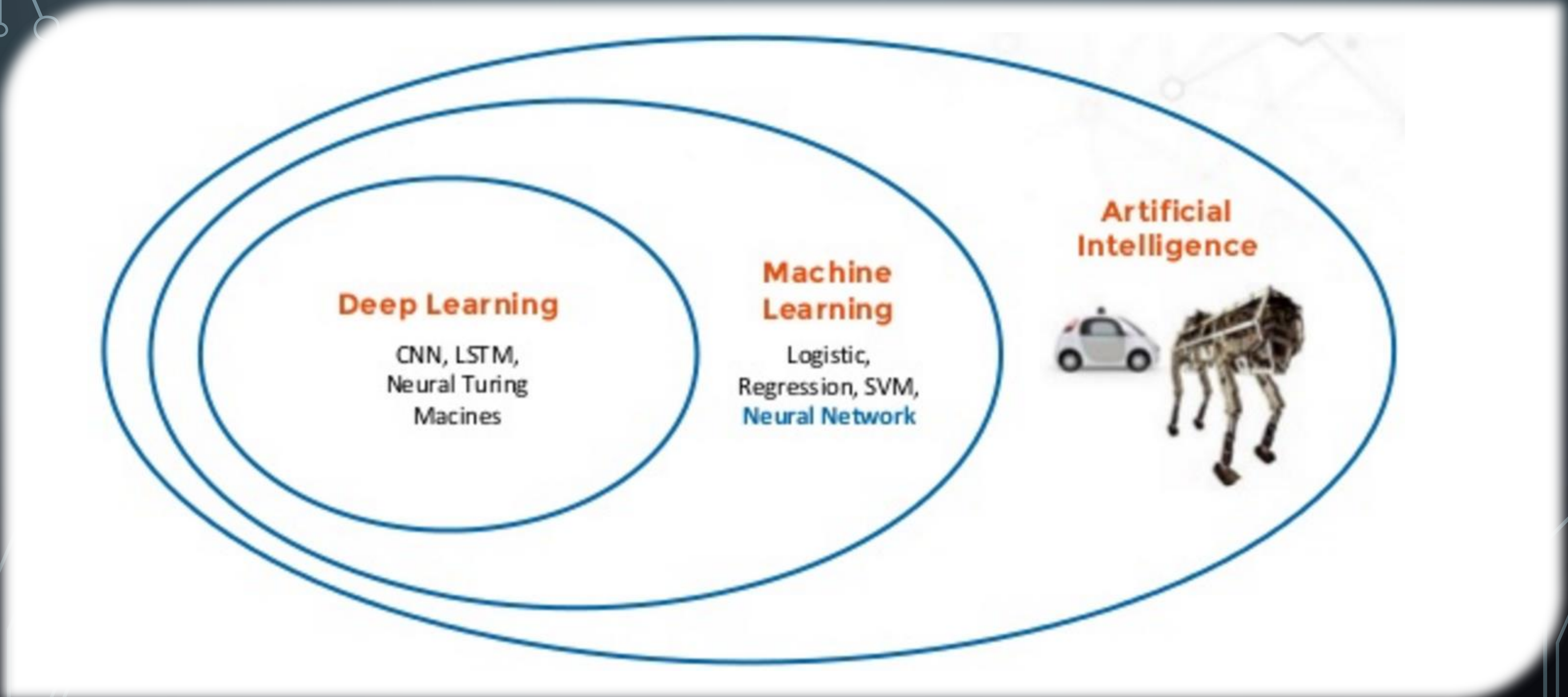
ELTE Fizika MSc, 2. évfolyam

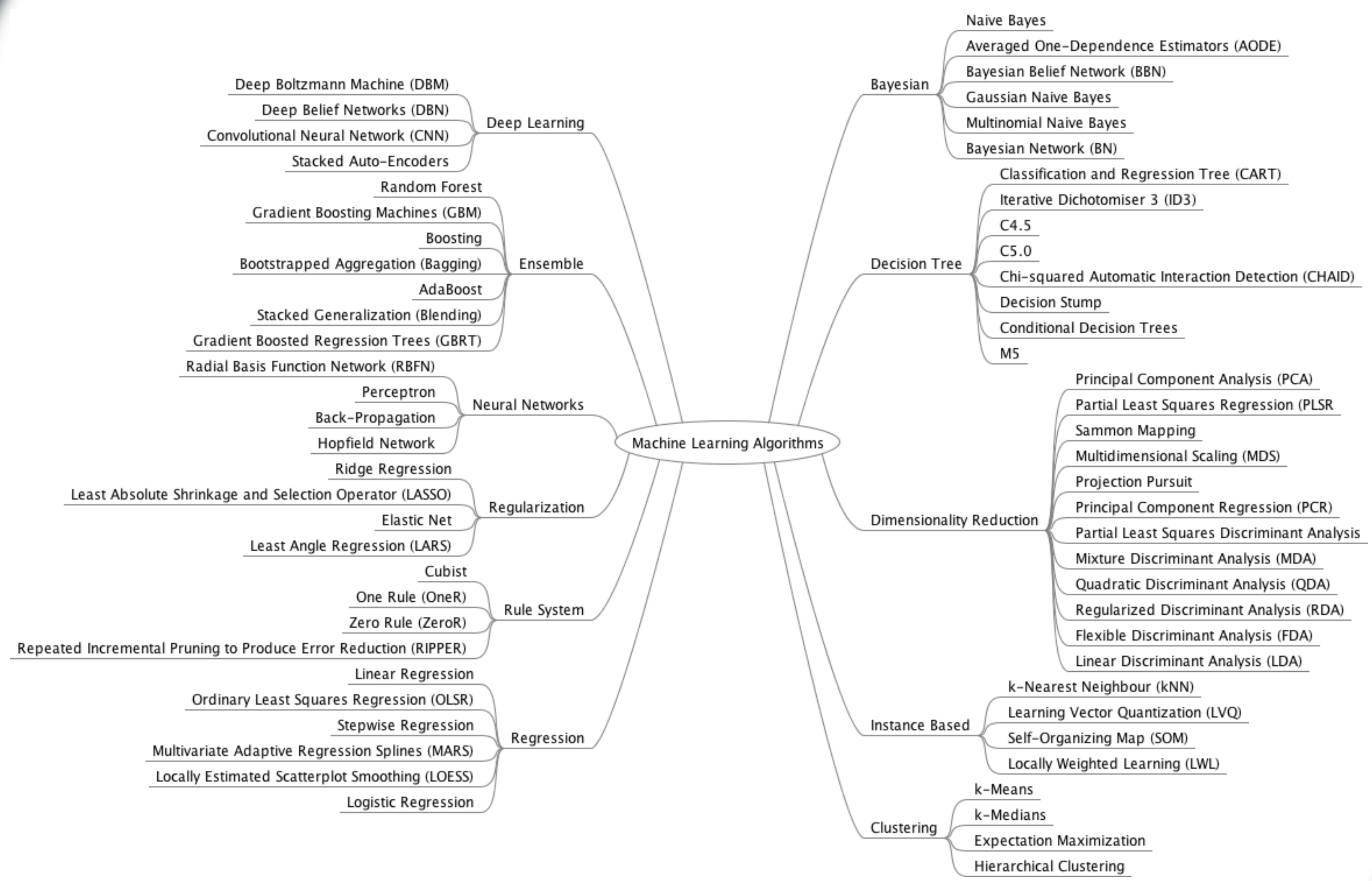
ELMÉLETI FIZIKA SZEMINÁRIUM

GÉPI TANULÁS NÉHÁNY ALKALMAZÁSA



MESTERSÉGES INTELLIGENCIA, GÉPI TANULÁS, DEEP LEARNING





DEEP LEARNING VS. MACHINE LEARNING

GÉPI TANULÁS

- Számitástechnika részterülete
- Olyan cselekvések végrehajtárásra képes, ami expliciten nem volt beprogramozva
- E tapasztalat, T feladatok osztálya, teljesítmény P(T)
- Tanuló program: P(T) nő E növelésével

FELADATOK TÍPUSAI

- Felügyelt tanulás (supervised learning):

  Adott: $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ adathalmaz (X $\ni x$ feature vektor, $Y \ni y$ label).
  Keressük: $f: X \to Y$ leképezést, úgy, hogy $L: X \times Y \to R$ pont fügvény maximális legyen.

- Nem felügyelt tanulás (unsupervised learning):

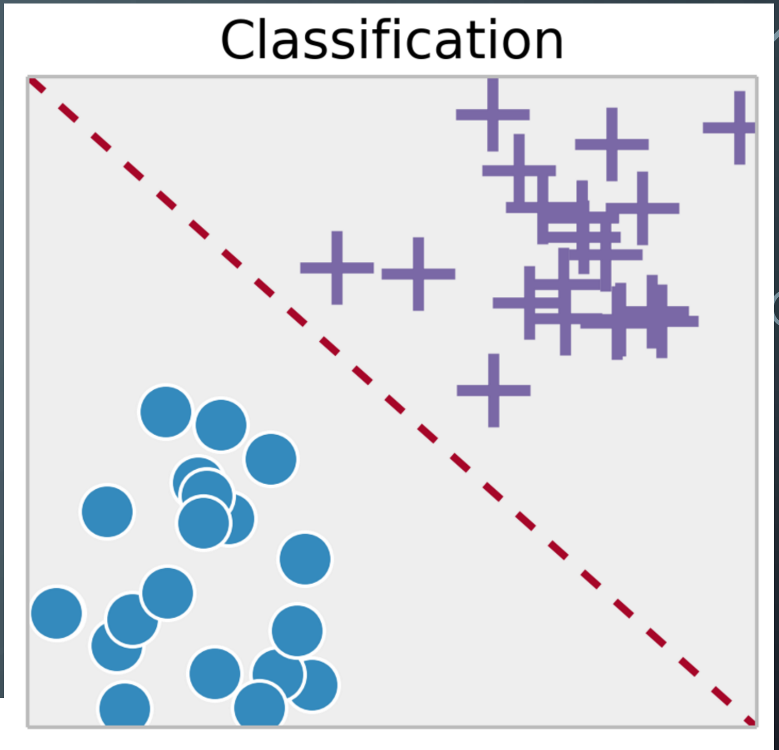
  Adott: $\{(x_1), \ldots, (x_N)\}$ adathalmaz. Nincsennek labelek!
  Keressük: adathalmaz struktúrát

- Megerősítéses tanulás (reinforcement learning):

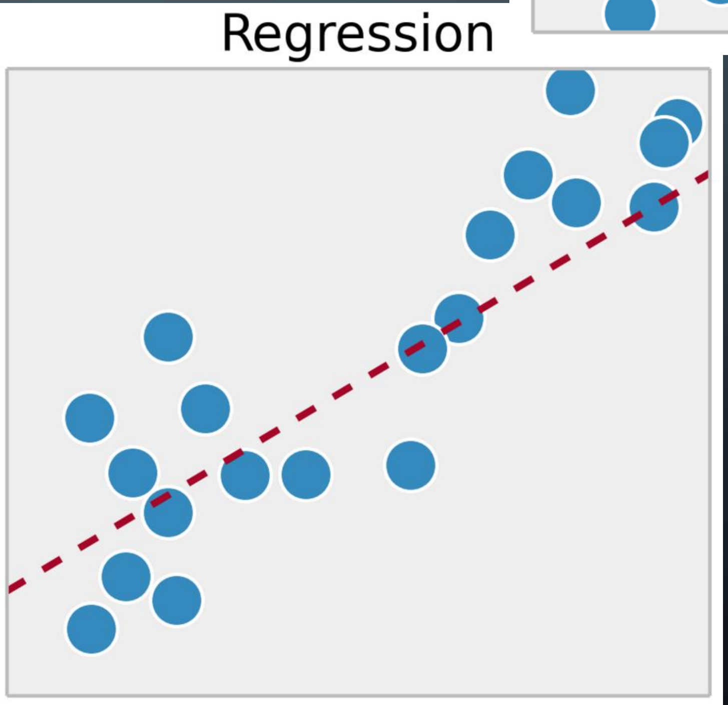
  Adott: megfigyelhető környezet
  Keressük: ügynököt, aki környezetet megfigyelve cselekvéseket tesz, úgy, hogy maximalizálja a jutalmat.

FELÜGYELT TANULÁS

- Klasszifikáció:
  - Diszkrét állapot (y diszkrét): kategoriák
  - Adott egy bemeneti vektor: milyen kategoriában esik?



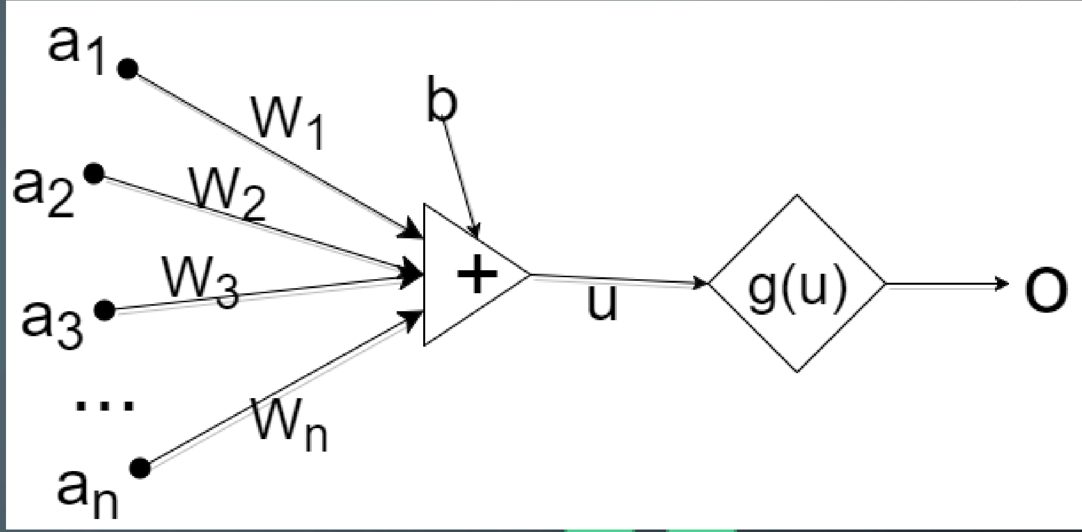
- Regresszió:
  - „Folytonos” állapot (y diszkrét)
  - Nem kategoriába sorolunk
  - Numerikus előrejelzés
  - Megszokott függvényillesztés
  - Fizikusok láteleme



NEURONHÁLÓZATOK

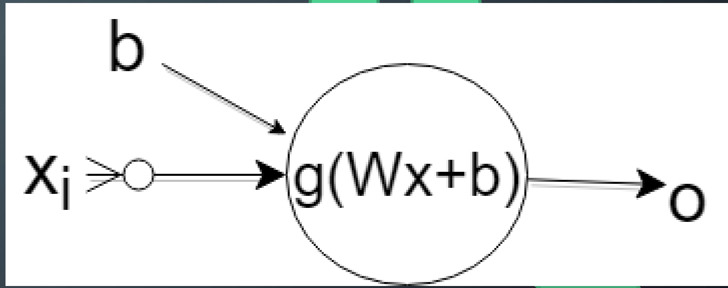
- Fizika: tudjuk a függvényt amit illeszteni szeretnénk
- Rengeteg esetben: nem tudjuk mi a függvény
- Kell: modell, ami bármilyen függvényt tud közeliteni
- Felhasználó: eredmény orientált, nem akarja megérteni a függvényt

RÉTEGEK: SOK NEURON



- Bemenet: a$\in R^n$
- Egy neuron kimenete: o$\in R$

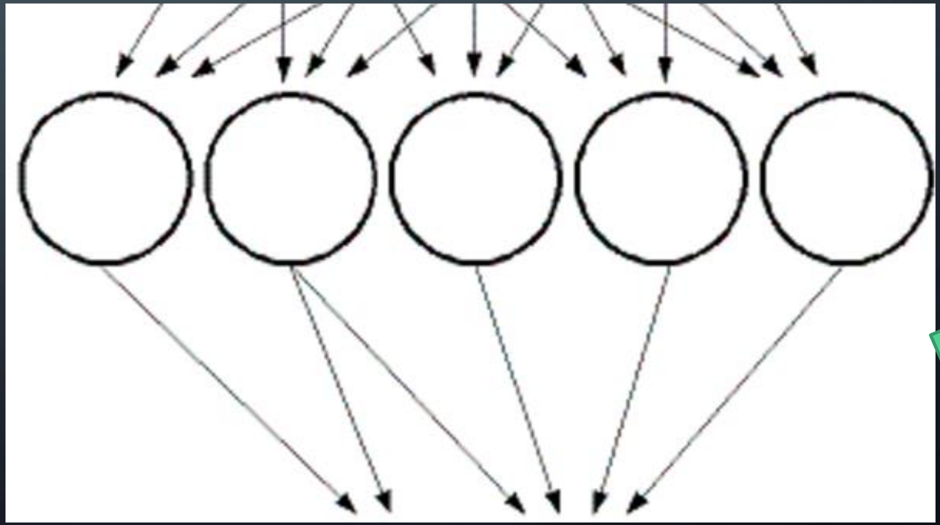
$$o^{neuron} = g(W^{1\times n}a + b)$$

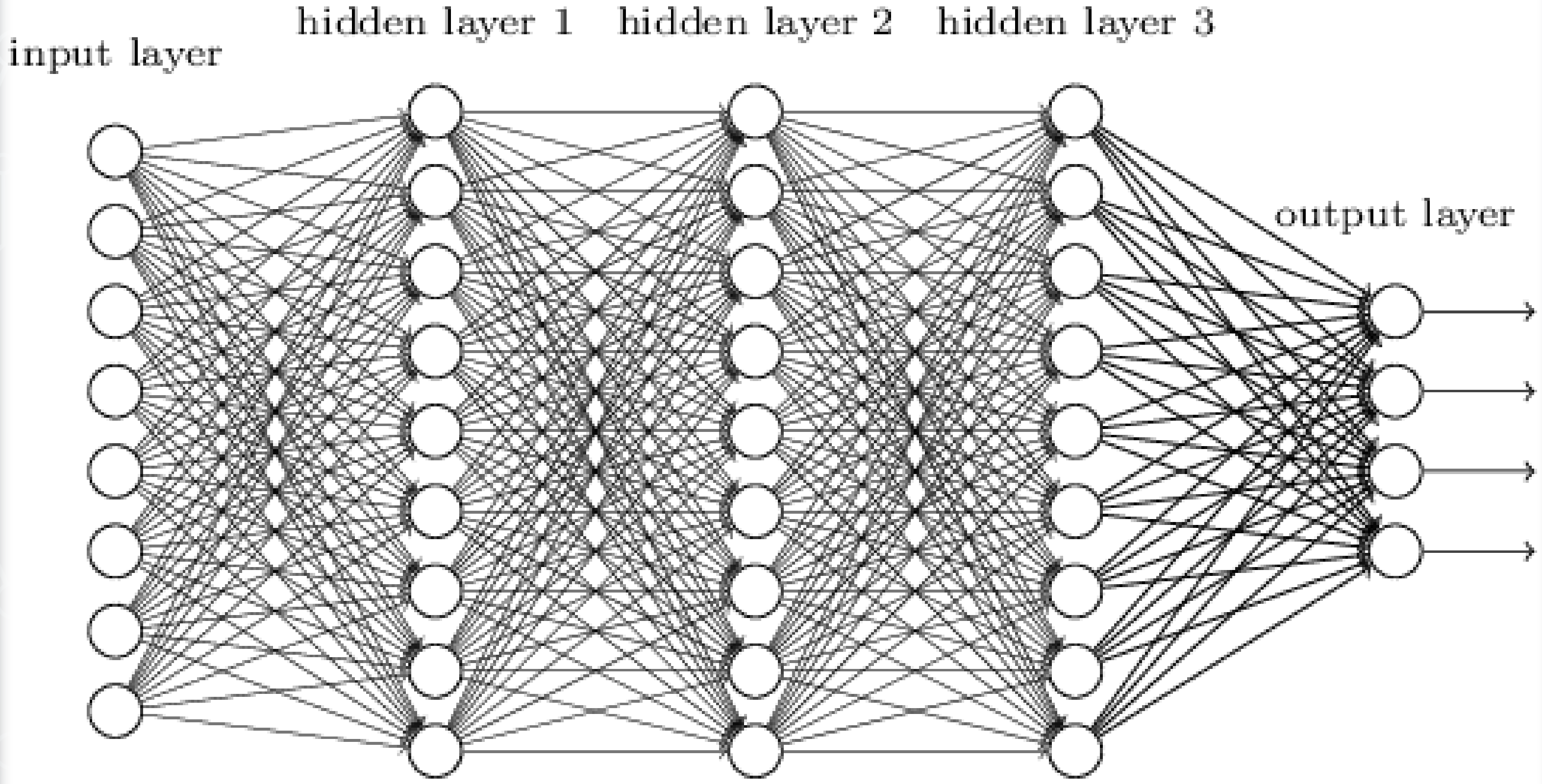


- M neuron a rétegben: kimenet: o$\in R^m$

$$\text{o} \qquad = g(W^{m\times n}a + b)$$

- W: súlymátrix
- g: nem lineáris függvény





NEURONHÁLÓ

- Rétegeket pakolunk egymás mögé
- Ez függvénykomopozició
- Sok réteg: bonyolult modell
- Teljes hálózat a következő:

$$f = g_{W_h b_h} \circ \ldots \circ\, g_{W_2 b_2} \circ g_{W_1 b_1}$$

- Deep learning: mély hálózat (sok sok réteg)

TANULÁS 1

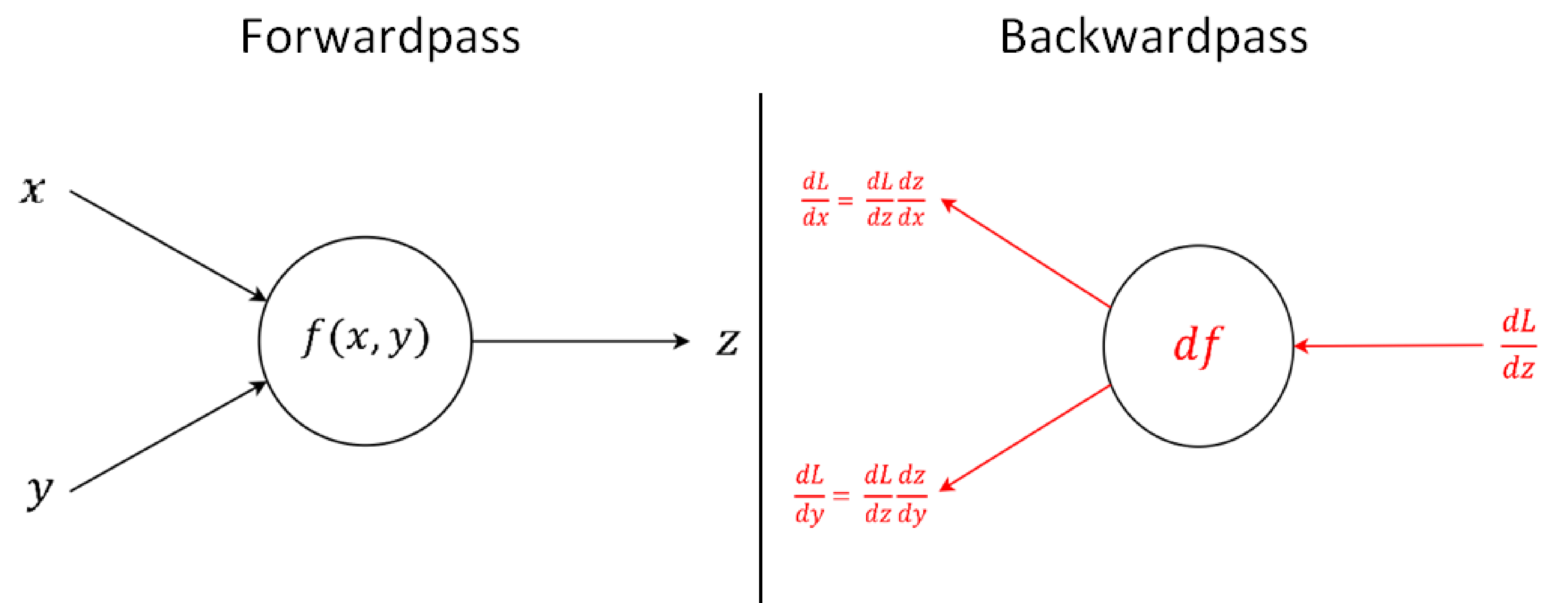
- Terveztünk egy hálózatot, azaz definiáltuk:

$$f = g_{W_h b_h} \circ \ldots \circ g_{W_2 b_2} \circ g_{W_1 b_1}$$

- Adott: $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ adathalmaz
- Keressük: $W_1, \ldots, W_h$ mátrix halmazt, és $b_1, \ldots, b_h$ vektor halmazt
- Feltétel: $f(x_i)$ „lehető legközelebb" legyen y_i-hez, minden i-re
- Azaz, minimalizálni szeretnénk valamilyen távolságot (költségfüggvényt):
  - Euklideszi távolság: $L = \frac{1}{N} \sum_{i=1}^{N} \| f(x_i) - y_i \|_2$
  - Cross-entropy: $L = \frac{1}{N} \sum_{i=1}^{N} y_i \log f(x_i)$

TANULÁS 2

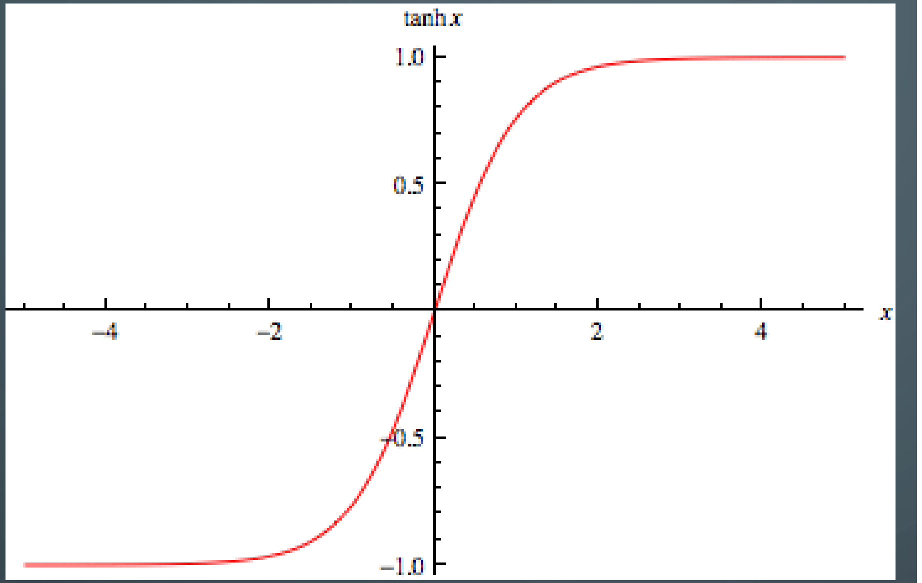
- Tehát, mi $W_1, \dots, W_h, b_1, \dots, b_h$ súlyok, hogy L minimális legyen?
- Megoldás: gradiens módszer: W,b-ket mindig L gradiens irányába változtatjuk
- Gradiens-t hogy határozzuk meg?
- Válasz: összetett függvény deriválási szabály → backpropagation algoritmus

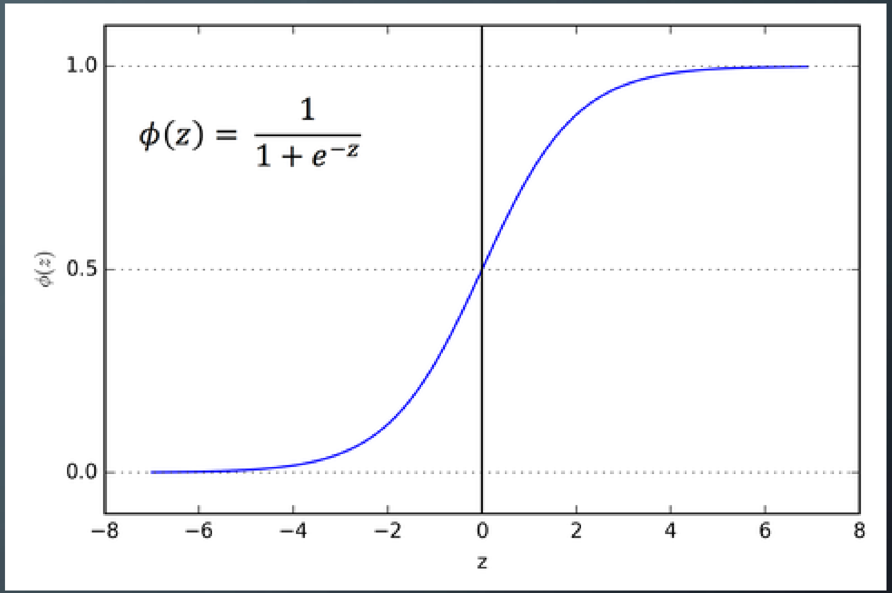


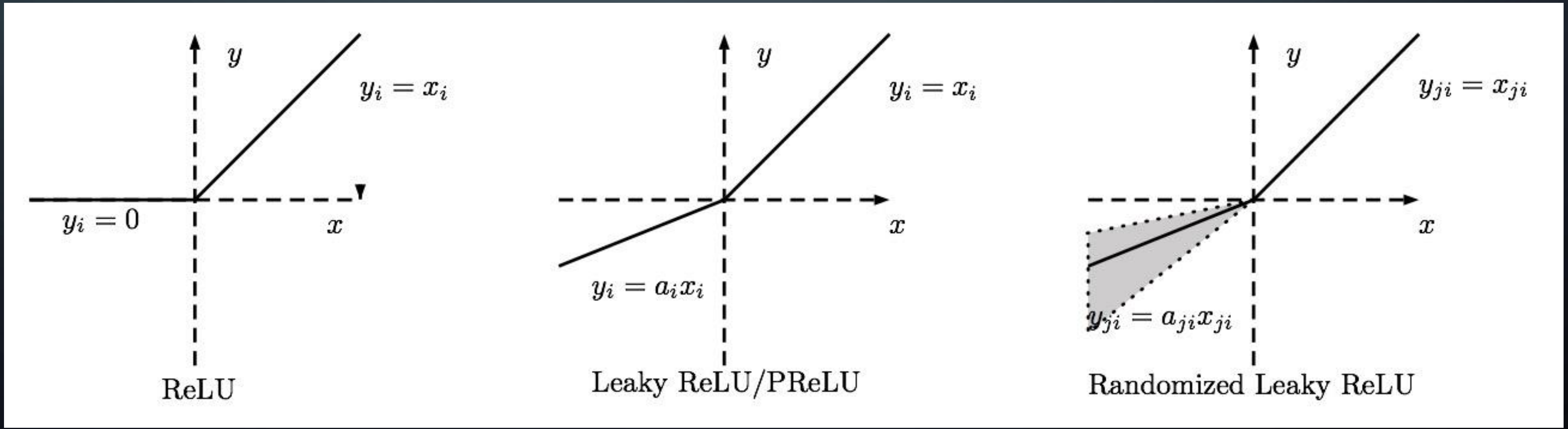
NEM LINEARITÁSOK

Mély hálózatok nagy problémája: eltűnő gradiens

MEGOLDÁS

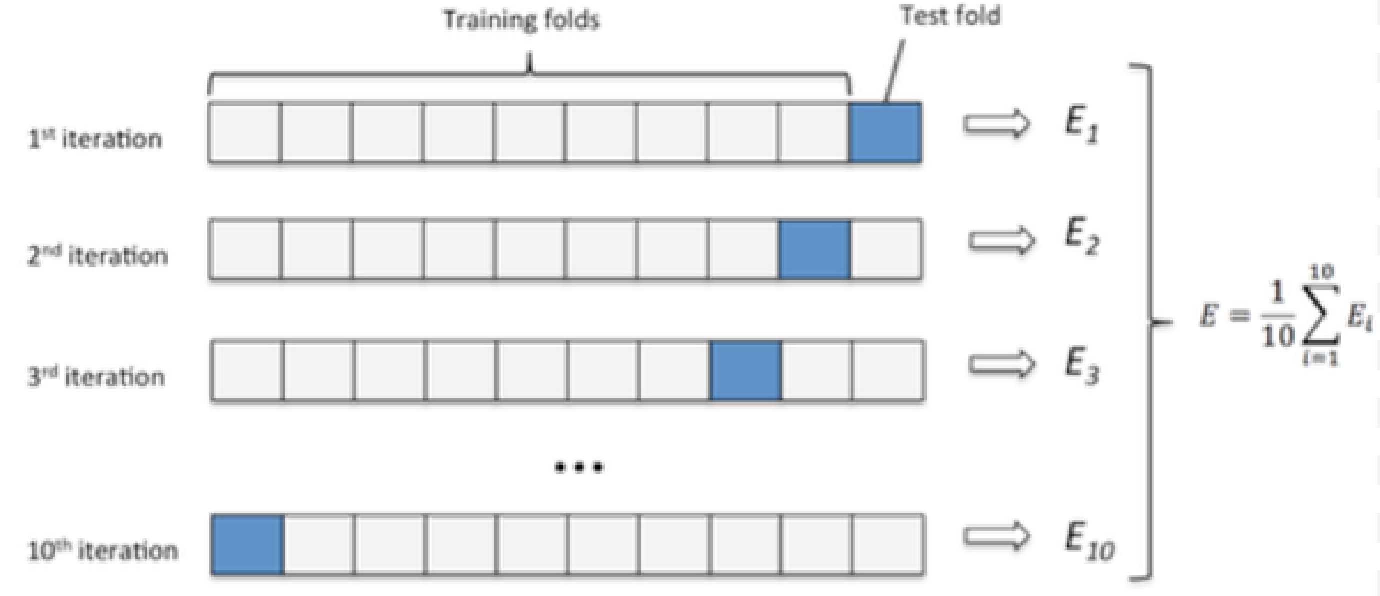






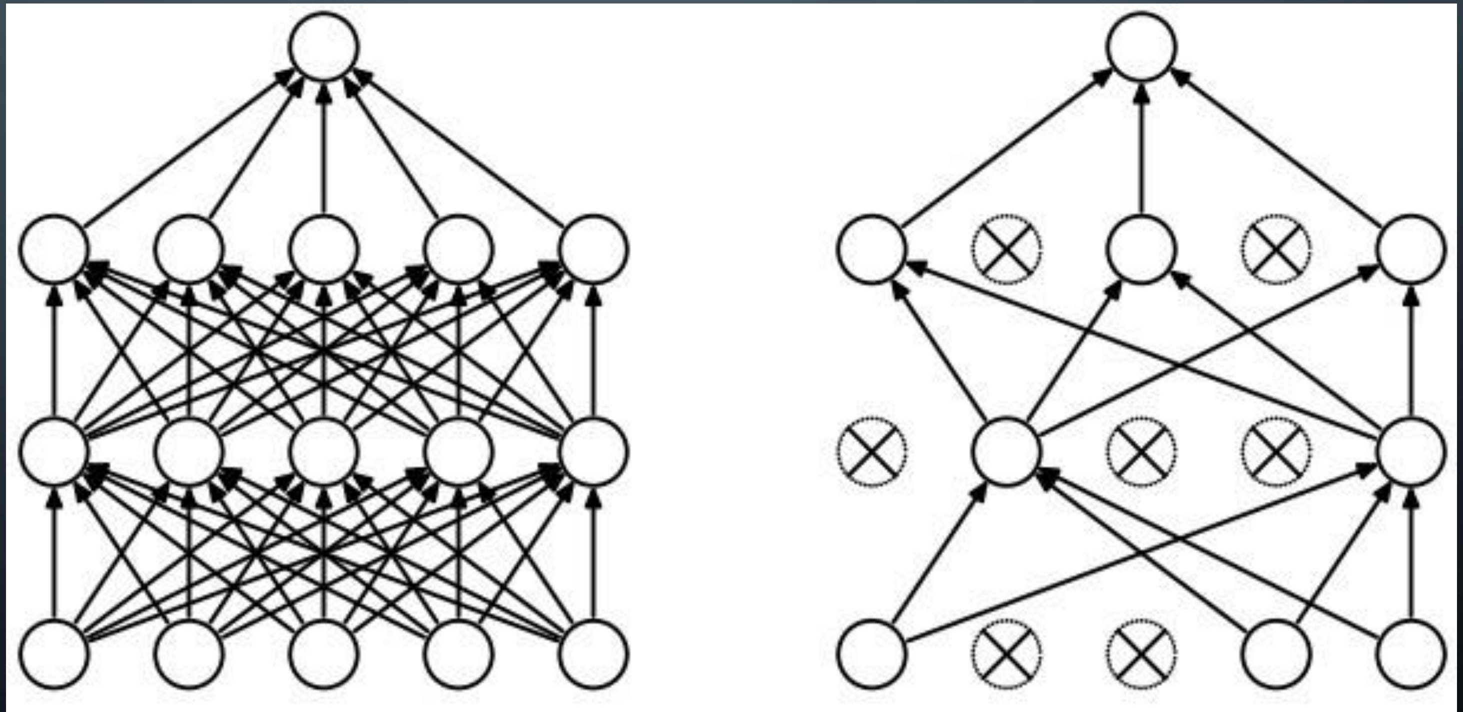
MILYEN JÓ A HÁLÓZAT?

- L-et minimalizáljuk, konvergál $\rightarrow^?$ konvergált hiba jellemzi a pontosságot
- NEM! Mert: nagyon könnyű overfittelni (mély háló, sok millió illesztési paraméter)
- Megoldás: adatszetet szét kell osztani tanuló és teszt halmazra
- Teszt halmazon L → pontosság (általánosít-e?)
- Általánosabb: k-fold Cross-validation



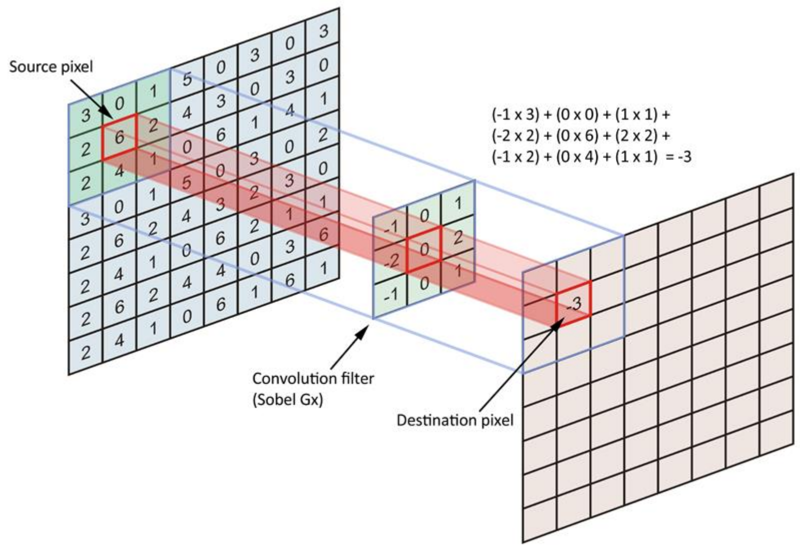
MÉLY HÁLÓZATOK: TÚL ILLESZTÉS

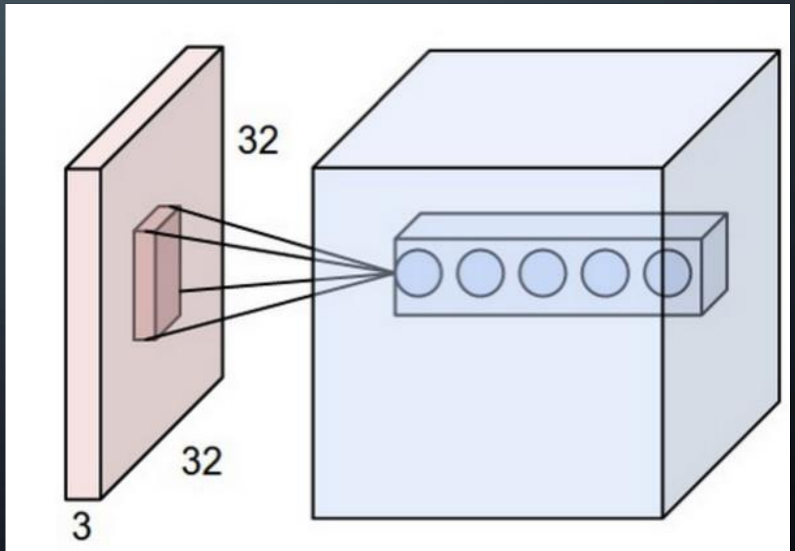
- Sok millió paraméter lehet egy mély hálózatban
- Könnyen overfitteljük az adatokat:
- Hatékony megoldások:
  - Regularizáció bevezetése: $L \to L + \gamma \sum \sum \sum \|W\|_2$
  - Droupout:



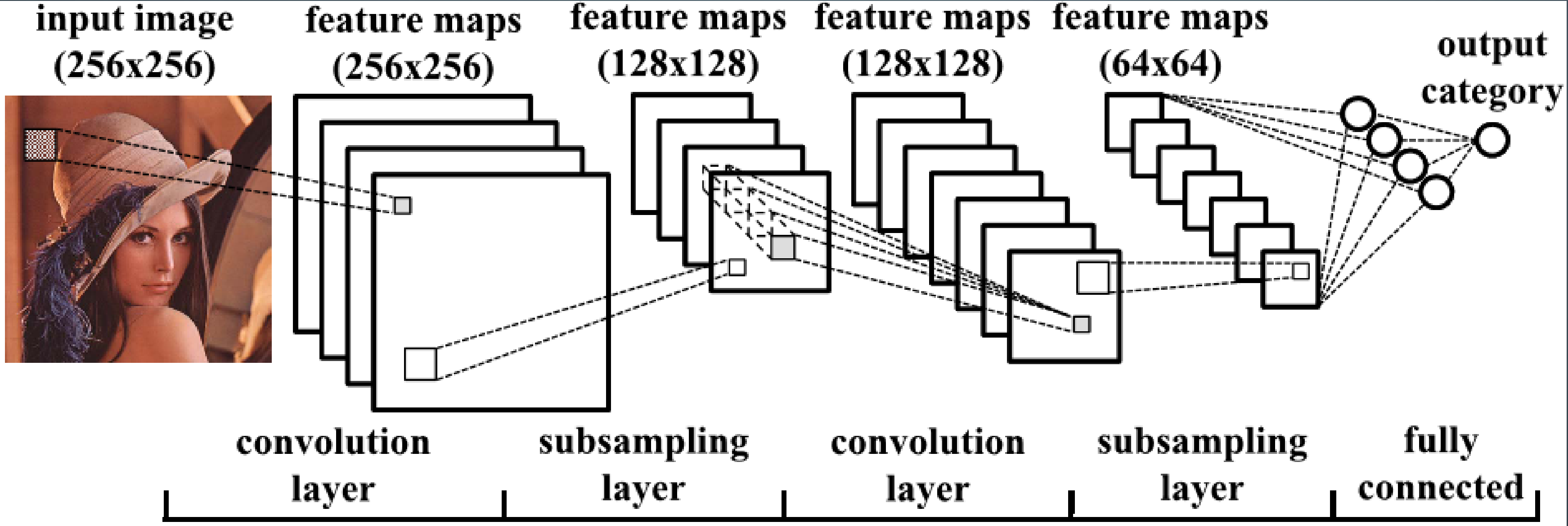
KONVOLUCIÓS NEURONHÁLÓZATOK

- Feladat: képek felismerése
- Feature vektor (x): most egy 3 dimenziós mátrix: width x height x 3
- CNN: speciális struktúra a képfelismeréshez kitalálva
- Kép: eltolás invariancia, lokális objektumok
- Lokalitás: neuronba nem az egész kép van bekötve, csak néhány szomszéd
- Eltolás invariancia: neuron csináljon egy map-et, úgy, hogy végigpásztázza a képet





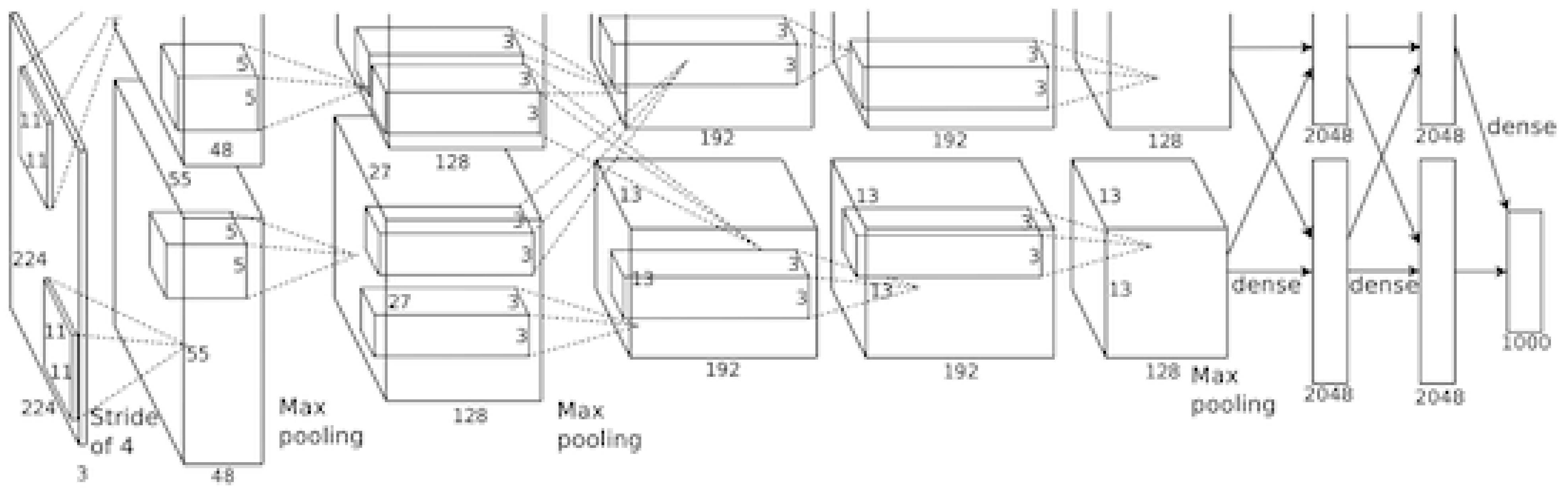
CNN



- 1989-ben vezette be Yann LeCun: LeNet5 (CIFAR10)
- De ekkor még nem lett nagyon népszerű
- Mi változott?
- Sok adat
- Erős GPU

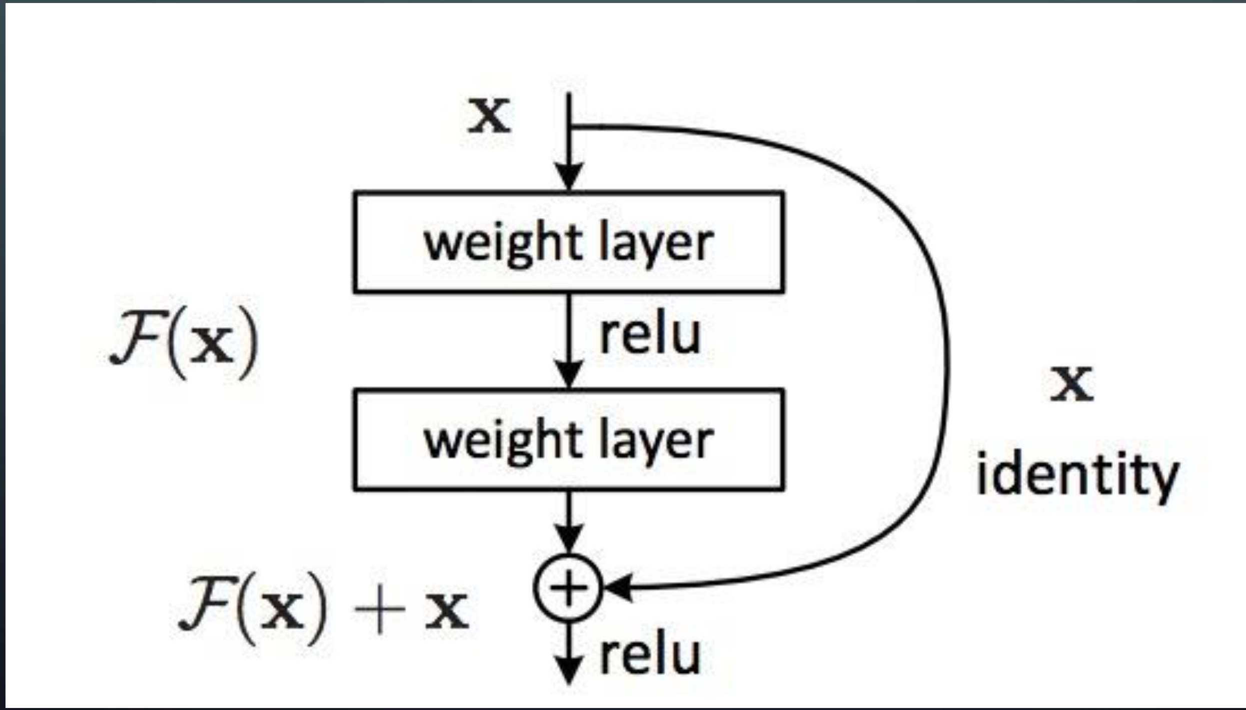
ALEXNET 2012

- 61 millió paraméter
- ImageNnet: top5 error15.4% (ember 5% körüli)
- Conv, pool rétegek váltakozása + végén teljesen összekötött réteg
- VGGNet (138 millió paraméter, 2013, 7.3% top5 error)



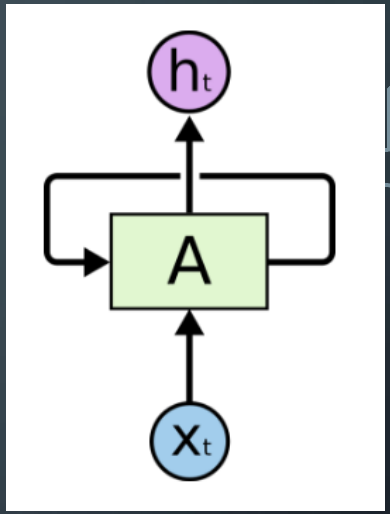
RESNET: 2015

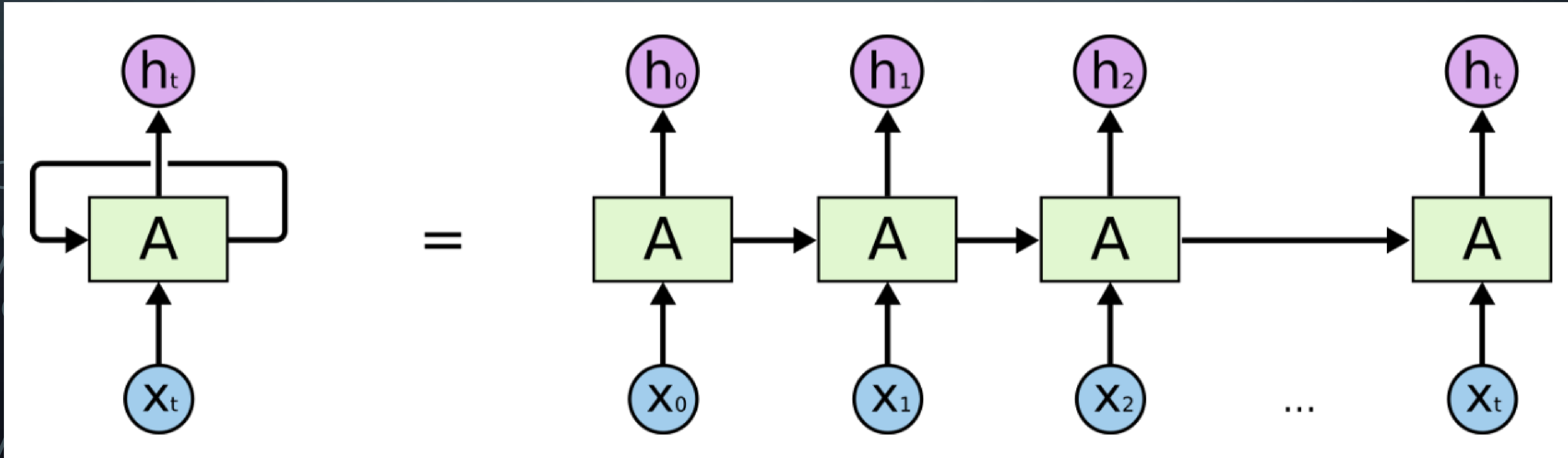
- ResNet 2015: 3.6% top5 error (ember kb. 5%)
- AlexNet: 8 layer, GoogleNet (2014 legjobbja) 22 layer
- Sok layert nem lehet pakolni: eltűnik a gradines a hálózat alján
- ResNet: 152 layer
- Trükk:



REKURENS HÁLÓZATOK

- Szöveg, beszéd, videó, idősorok: számítanak az előzmények
- Ember: nem dob el minden előző infót, és kezdi elölről megérteni a szöveget
- Neuronhálózat ezt nem tudja
- Megoldás: rekurens hálózat

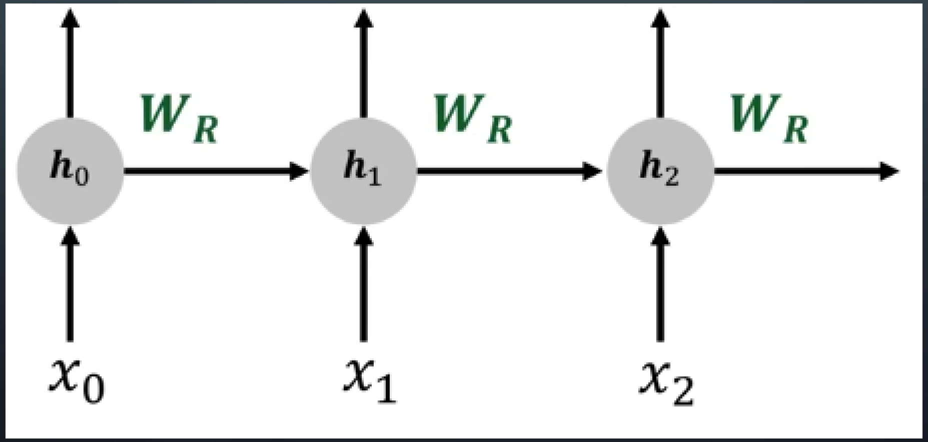




RNN

- Idősorokat szeretnénk feldolgozni: feature vektor időfüggő: $x_t \in R^n$
- Loop: memóriát vittünk a rendszerbe
- Neuron kimenete t-ben: $h_t = g(W_I x_t + W_R h_{t-1} + b)$
- RNN: $P(y_t | y_{t-1} \dots y_t)$ modelt tanul
- $\frac{dL}{dW_R} = \frac{dL}{dh_t}\frac{dh_t}{da}, a = W_I x_t + W_R h_{t-1} + b$, de h_{t-1} is függ $W_R -$ től

időben is kell backpropagationt csinálni

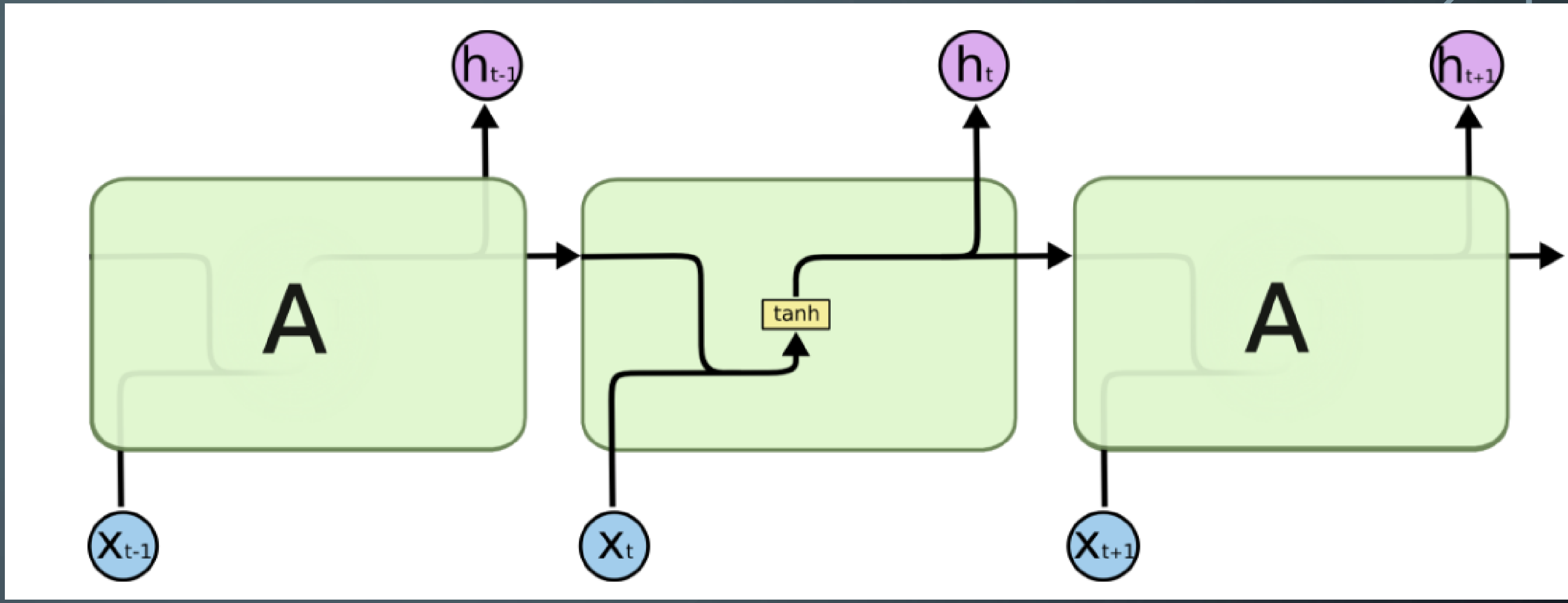


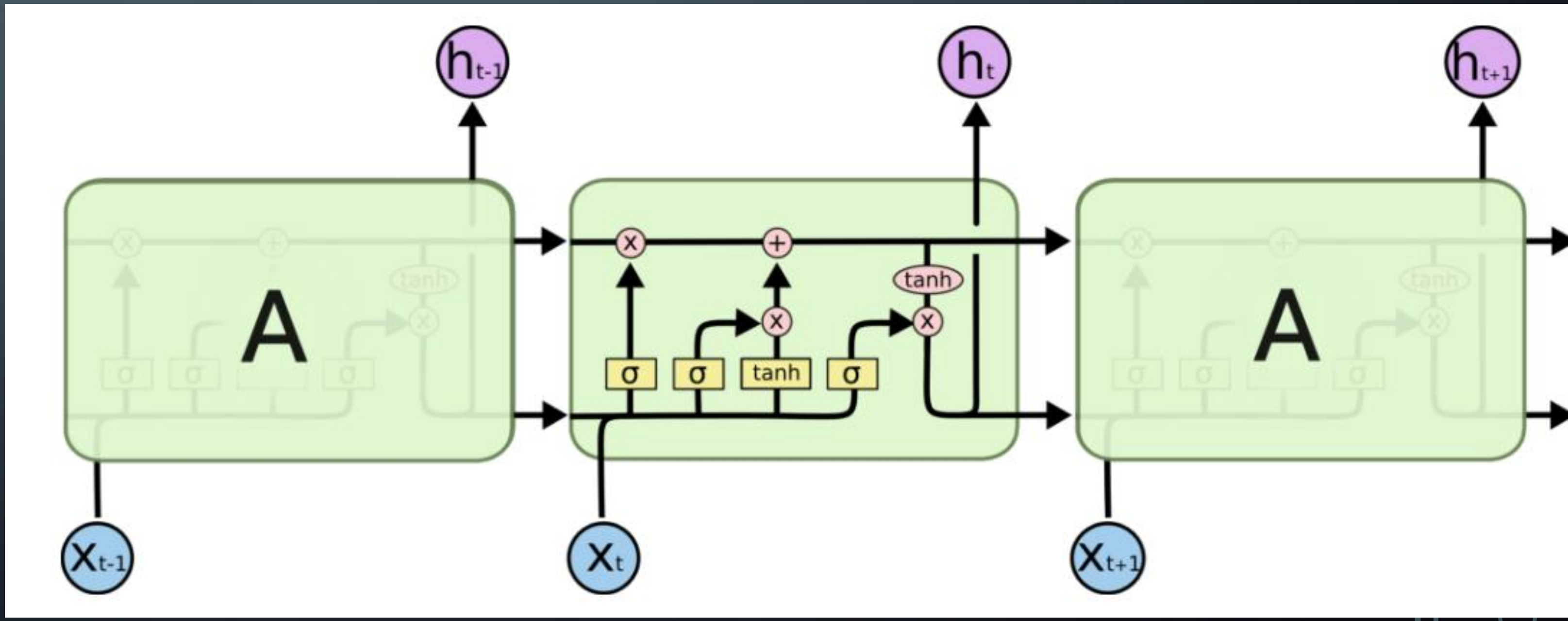
PROBLÉMÁK AZ RNN-EL

- Időben sokat akarunk hátramenni
- De gradiensek: felrobbannak vagy eltűnnek
- RNN kb. 10 időlépést tud hátramenni: előtte látott információkat elfelejti
- Felrobbanás:
  - Észlelés: könnyű (Loss függvény elszáll)
  - Orvoslás: pl. gradiens küszöb értékben maximalizálás
- Eltűnés:
  - Észlelés: nehéz
  - Orvoslás: népszerű: RNN helyet LSTM hálózat

LSTM

- 1997-ben vezették be, manapság lettek népszerűek
- Lényeg: cella állapot
- Cella állapot csak lineárisan változik időben
- Tudunk információt beírni, törölni, kiolvasni a cella állapotból

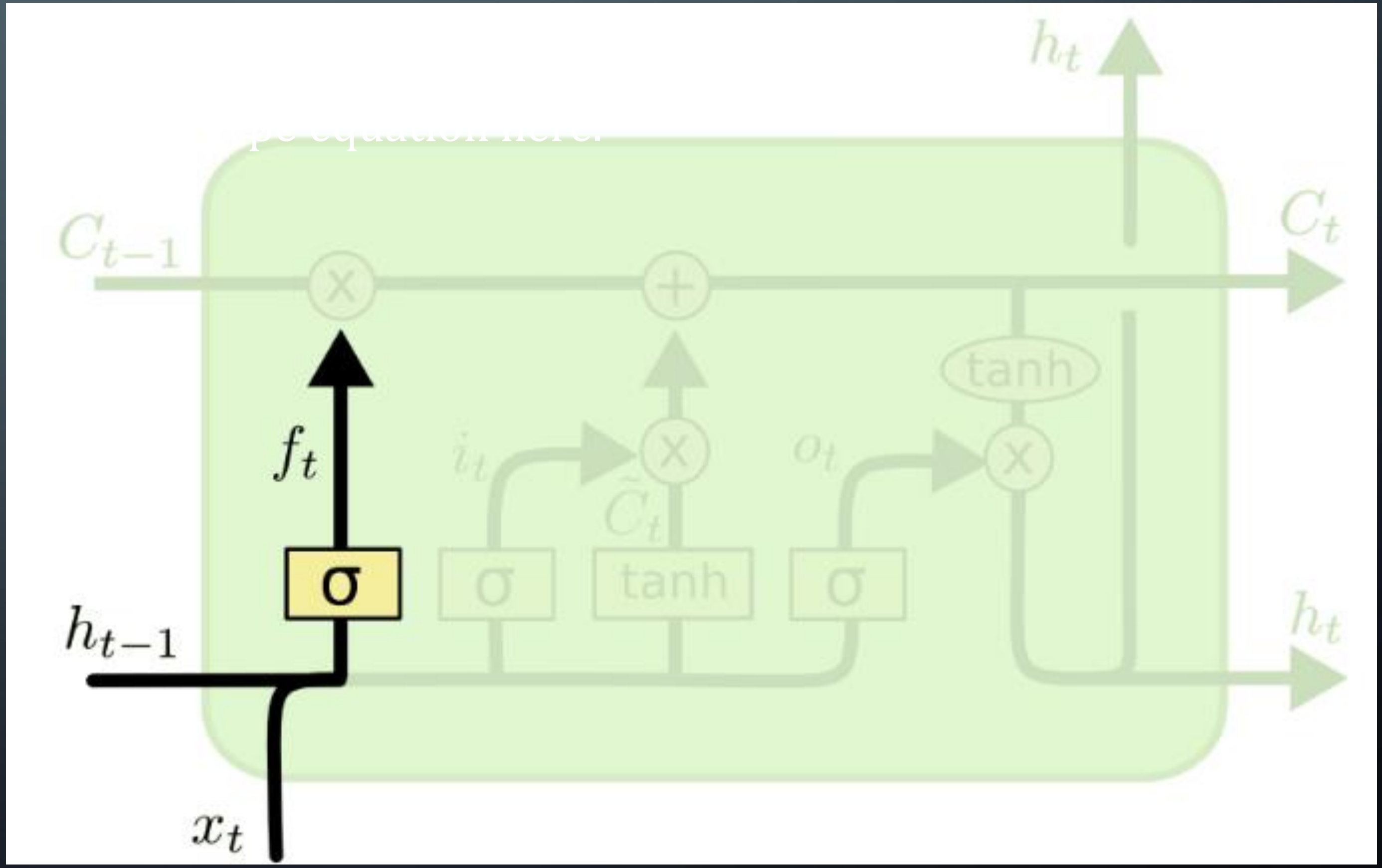




LSTM: INFORMÁCIÓ TÖRLÉSE

- Felejtő kapu: x_t, h_{t-1} alapján egy 0 és 1 közti számokból álló vektor (Sigmoid)
- Szorozzuk a cella állapotot (c_{t-1}) → mennyi információt örzünk meg

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

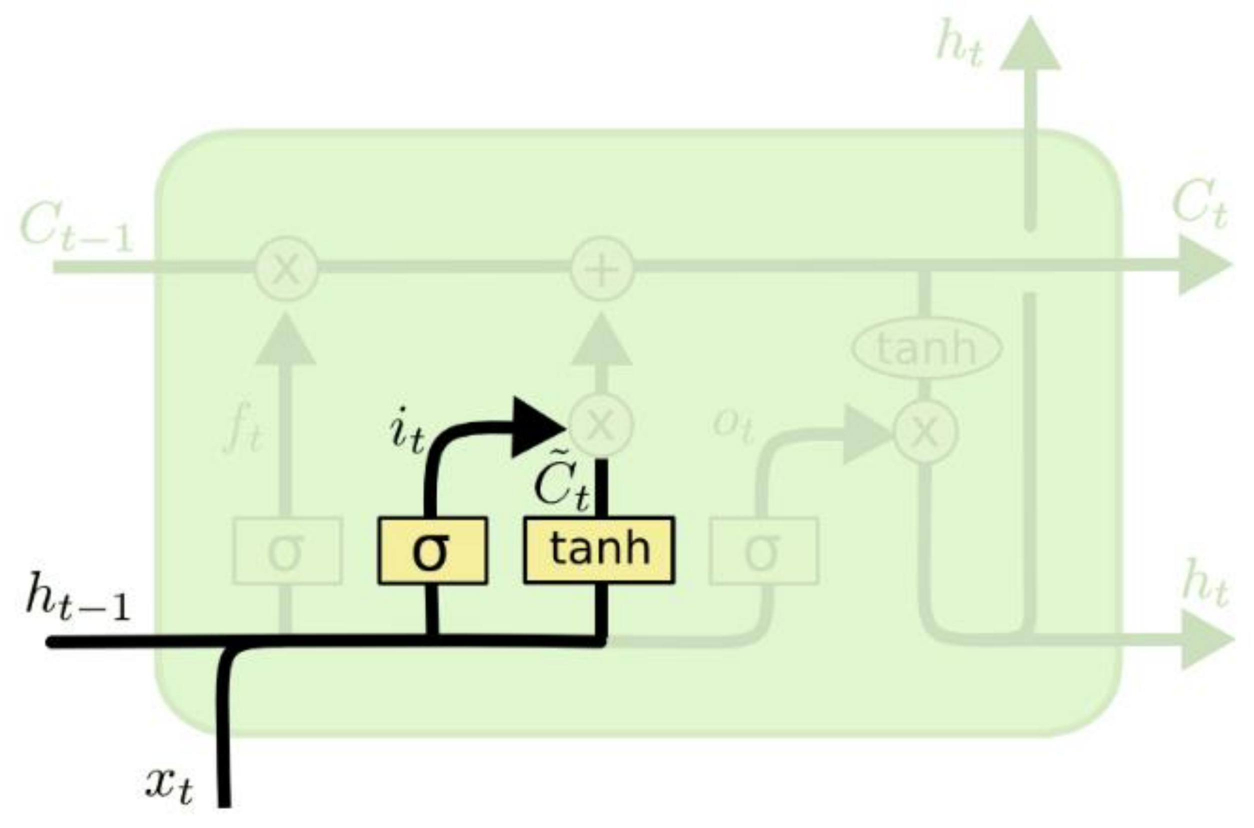


- f_t forget gate layer
- Kapun átmegy: $f_t c_{t-1}$

LSTM: INFORMÁCIÓ HOZZÁADÁSA

- Két rész: miből mennyit adunk a cella állapothoz
- Input layer gate (sigmoid): milyen értékeket mennyire frissítünk
- Célérték layer (tanh): milyen infót szeretnénk hozzáadni

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$



- Állapotváltozás:

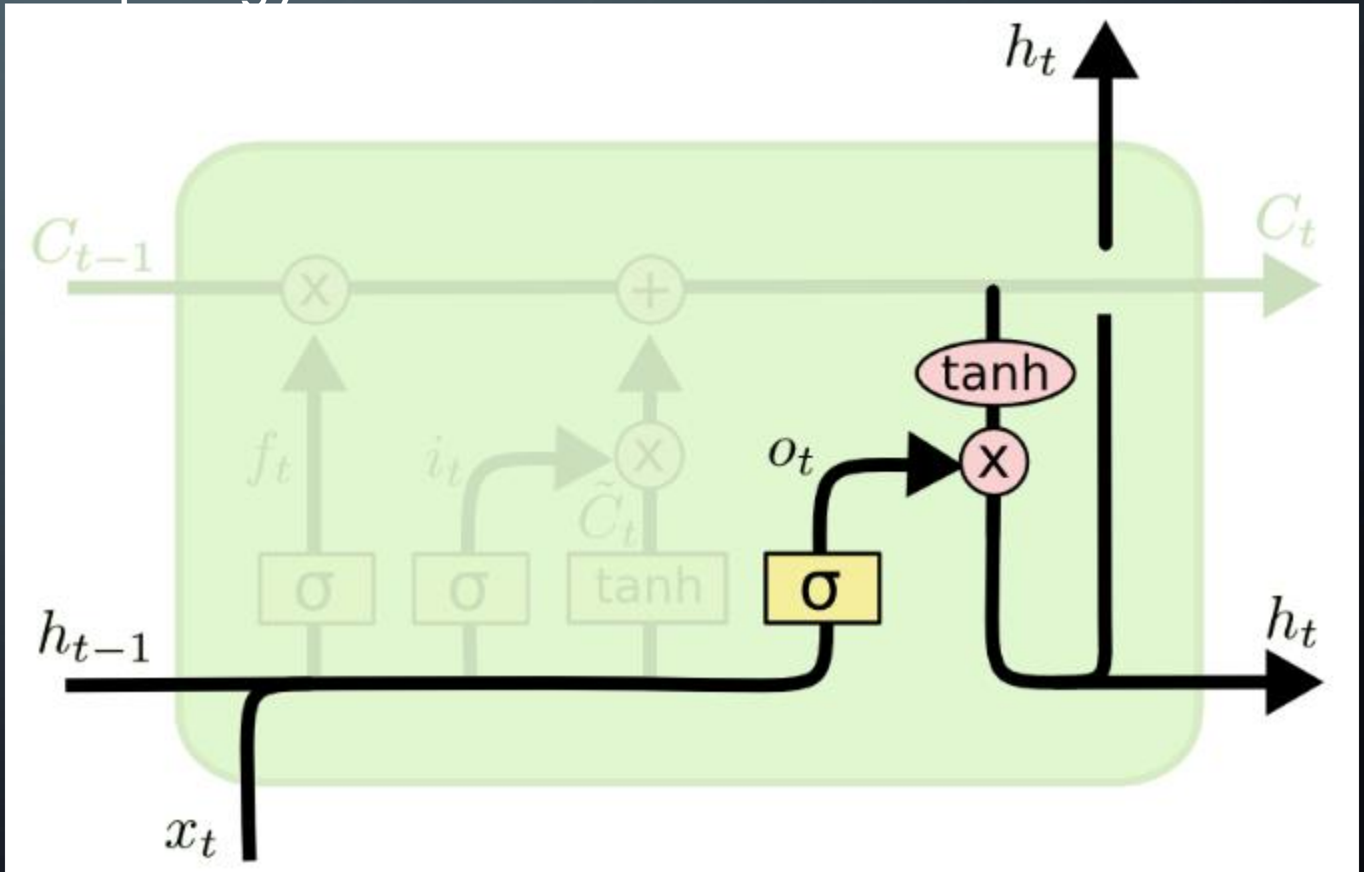
$$c_t = f_t c_{t-1} + i_t \tilde{c}_t$$

LSTM: KIMENET

- Első lépésben eldöntjük a cella állapotából mit kapcsolunk a kimenetre

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

- Következőben a cella állapotát átvezetjük egy tanh-n
- Kimenet: $h_t = o_t \tanh c_t$
- Ezzel a konstrukcióval időben sokáig tudunk visszatekinteni
- Memoria változtatás pl.:
  angol szöveg: emlékezni he/she
  új szövegrész: váltás



LSTM: MATEK GENERÁLÁS

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{étale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

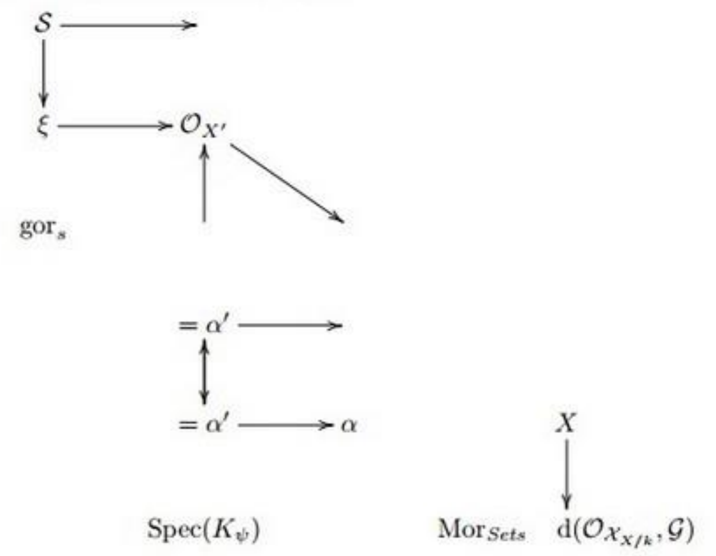
be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type $f_\star$. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{étale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1} \mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

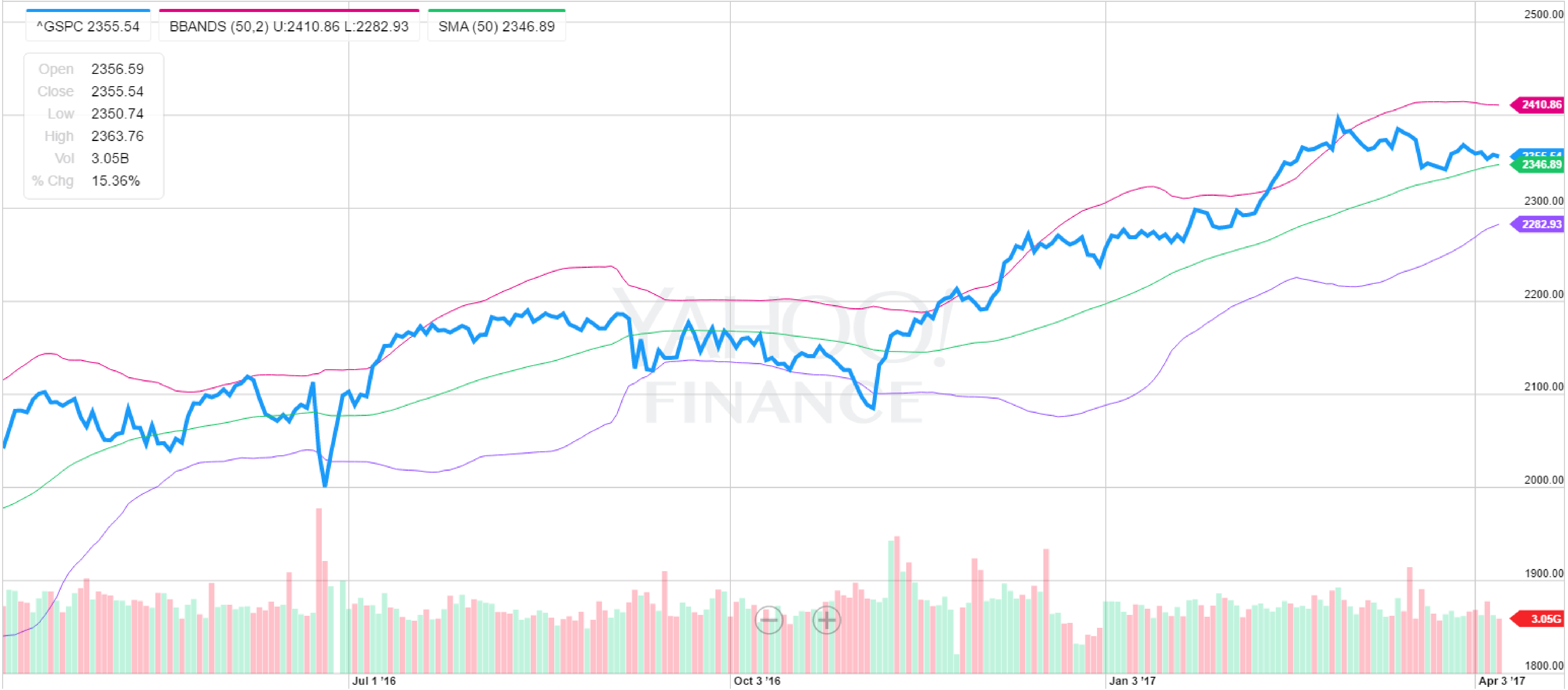
LSTM: KÓDÍRÁS

```
/*
 *  Copyright (c) 2006-2010, Intel Mobile Communication
 *
 *   This program is free software; you can redistribut
 * under the terms of the GNU General Public License ve
 * the Free Software Foundation.
 *
 *        This program is distributed in the hope that
 * but WITHOUT ANY WARRANTY; without even the implied w
 *   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOS
 *
 *  GNU General Public License for more details.
 *
 *   You should have received a copy of the GNU General
 *    along with this program; if not, write to the Fre
 *  Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

#include <linux/kexec.h>
#include <linux/errno.h>
```

```
/*
 * If this error is set, we will need anything right after that BSD.
 */
static void action_new_function(struct s_stat_info *wb)
{
  unsigned long flags;
  int lel_idx_bit = e->edd, *sys & ~((unsigned long) *FIRST_COMPAT);
  buf[0] = 0xFFFFFFFF & (bit << 4);
  min(inc, slist->bytes);
  printk(KERN_WARNING "Memory allocated %02x/%02x, "
    "original MLL instead\n"),
    min(min(multi_run - s->len, max) * num_data_in),
    frame_pos, sz + first_seg);
  div_u64_w(val, inb_p);
  spin_unlock(&disk->queue_lock);
  mutex_unlock(&s->sock->mutex);
  mutex_unlock(&func->mutex);
  return disassemble(info->pending_bh);
}
```

TŐZSDEI ELŐREJELZÉS



TŐZSDEI ELŐREJELZÉS

- Árfolyam: idősor
- Egy több rétegű LSTM hálózatott építhetünk
- Elkezdjük múltbéli adatokkal tanítani, a tett előrejelzésre van adat, amiből Loss függvényt csinálhatunk
- Árfolyamon végigmenve: jövőbeli előrejelzéseket tehetünk