Üzleti Intelligencia

10. Előadás: Visszacsatolásos neurális hálózatok

Kuknyó Dániel
Budapesti Gazdasági Egyetem

2023/24
1.félév

Visszacsatolásos neurális hálózatok alapjai

Alkalmazás	Input	Output
Beszédfelismerés		"Milyen szép időnk van ma!"
Szemantikai értelmezés	"Ez egy rossz film volt."	
DNS szekvencia elemzés	AGCCCTGTACTAG	AGCCCTGTACTAG
Gépi fordítás	"Willst du mit mir tanzen?"	"Szeretnél velem táncolni?"
Videók elemzése		Futás
Nevek felismerése	Tegnap Józsi letörölte a termelési adatbázist.	Tegnap Józsi letörölte a termelési adatbázist.

Visszacsatolásos hálózatok: nevek felismerése a szövegben

Input: Tegnap Józsi letörölte a termelési adatbázist.

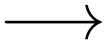
Input reprezentáció: $X = [x_1,\ x_2,\ x_3,\ ...,\ x_t,\ ...,\ x_6]$

Output reprezentáció: $Y = [y_1,\ y_2,\ y_3,\ ...,\ y_t,\ ...,\ y_6]$

Output: $[0,\ 1,\ 0,\ 0,\ 0,\ 0]$

Szavak reprezentálása 1-hot vektorokkal

Osztály
Macska
Kutya
Teknős



Macska	Kutya	Teknős
1	0	0
0	1	0
0	0	1

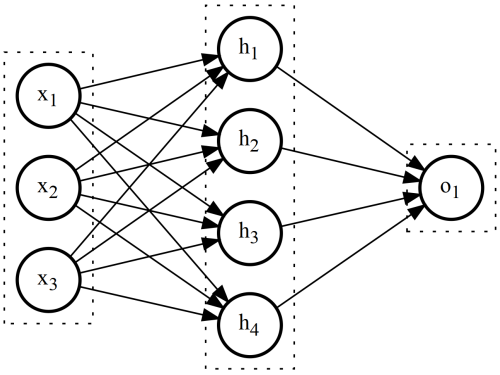
Az egyes szavak ilyen módon való kódolása lehetővé teszi, hogy egy neurális hálózat felépítse a saját **szókincsét**, majd különböző szekvenciákat bináris, azonos hosszúságú vektorok halmazaként reprezentáljon.

Miért alkalmatlanok szekvencia feldolgozásra a hagyományos hálózatok?

- A szekvenciák hossza nem előre meghatározott, ezért nem képes változó hosszúságú input adattal dolgozni.
- Nem képes azonos szekvenciák között súlyokat megosztani.

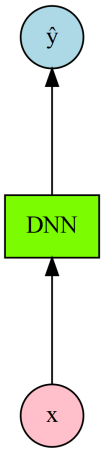
Ezzel szemben az RNN hálózatok:

- Változó hosszúságú sorozatokkal működnek.
- Hosszútávú függőségeket is meg tudnak tanulni.
- Megőrzi az input vektor rendezettségét.
- Képes paramétereket megosztani sorozatok között.

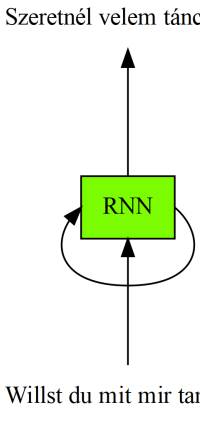


Mélyhálózatok (DNN) vs. visszacsatolásos hálózatok (RNN)

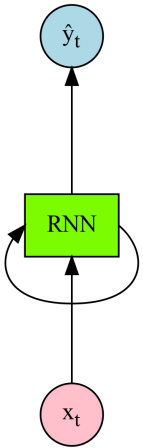
Hagyományos mélyhálózat



Visszacsatolásos mélyhálózat működése

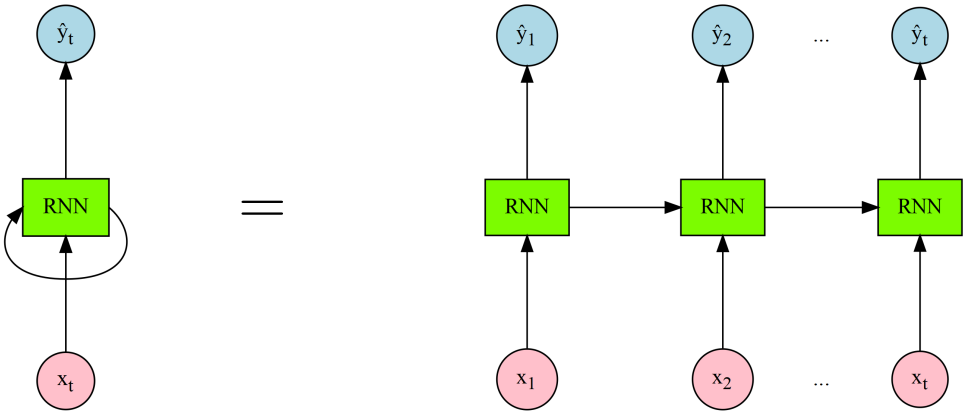


Visszacsatolásos mélyhálózat jelölése



Reprezentáció

A visszacsatolásos neurális hálókat kétféle módon lehet jelölni: összehajtott és lehajtott állapotban. Az összehajtott jóval kompaktabb, a lehajtott viszont egy tiszta és intuitív nézőpontot ad a hálózat architektúrájára.



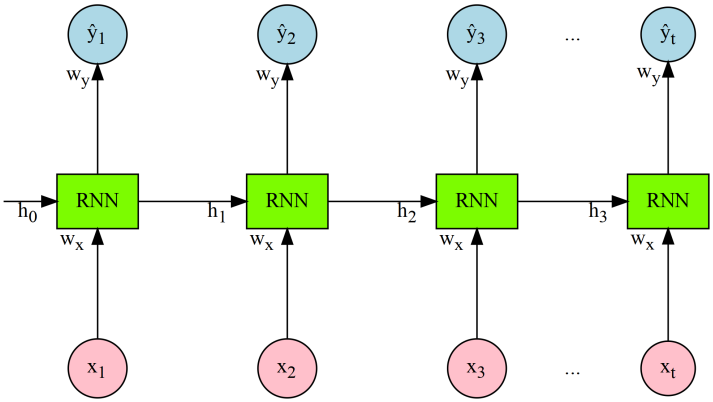
Súlyok és kapcsolatok

- x_t: Input vektor t. eleme.
- $\hat{y}_t$: Output vektor t. eleme.
- h_t: Rejtett réteg aktivációja (**cella állapota**) t időben.
- W_x: Input súlyai (időben állandó, tanítható).
- W_y: Output súlyai (időben állandó, tanítható).

Ebben az esetben az output

$$\hat{y}_t = f(x_t, h_{t-1})$$

x_t aktuális input és h_{t-1} előző állapot függvénye.



Számítások az RNN-ben

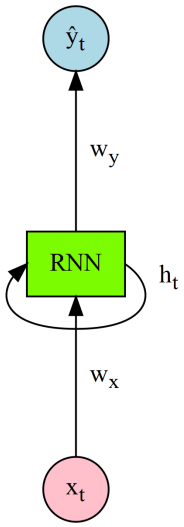
Rejtett állapot számítása

$$h_t = tanh\left(W_h \cdot h_{t-1} + W_x \cdot x_t + b_h\right)$$

Ahol $tanh(\cdot)$ a hiperbolikus tangens függvény, h_{t-1} az előző cella állapota, x_t az input vektor aktuális eleme, b pedig a cella torzítása.

Output számítása

$$\hat{y} = tanh\left(W_y \cdot h_t + b_y\right)$$



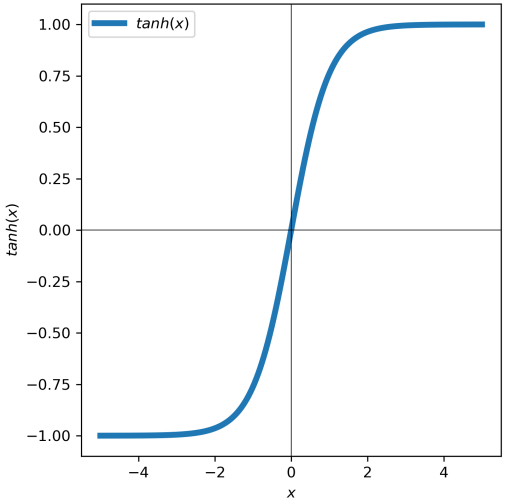
A hiperbolikus tangens függvény

A hiperbolikus tangens függvény az egyik gyakori aktivációs függvény visszacsatolásos neurális hálózatokban. Előnyei a **nemlinearitás**, **erős gradiens**, **nulla középpontúság**.

Hiperbolikus tangens függvény

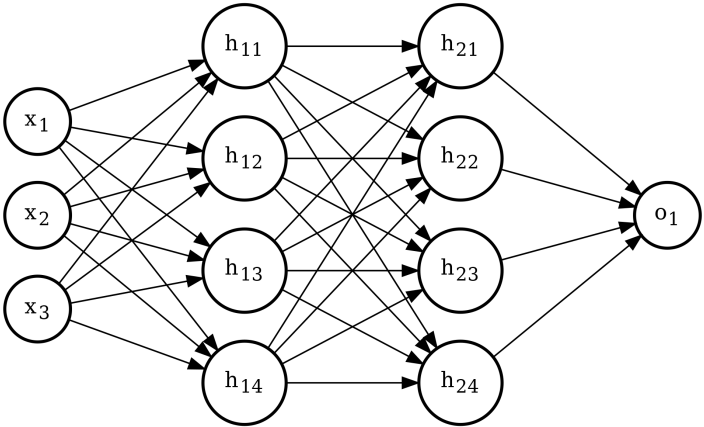
$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Ahol $e \approx 2.71828$ a természetes logaritmus értéke.



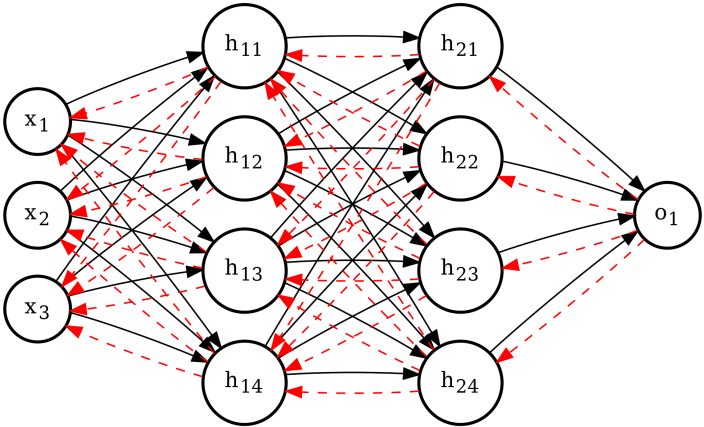
Hiba visszaáramoltatás hagyományos hálózatok esetén

1. Köteg előre áramoltatása a hálózaton.
2. Költség kiszámítása.
3. Költség gradiensének meghatározása minden paraméter szerint.
4. Paraméterek frissítése a költség minimalizálása érdekében.



Hiba visszaáramoltatás hagyományos hálózatok esetén

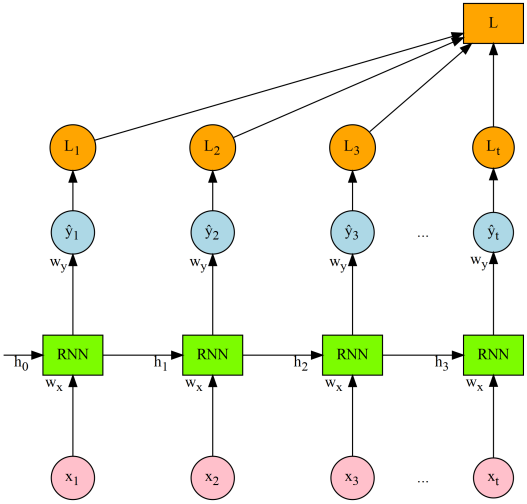
1. Köteg előre áramoltatása a hálózaton.
2. Költség kiszámítása.
3. Költség gradiensének meghatározása minden paraméter szerint.
4. Paraméterek frissítése a költség minimalizálása érdekében.



Hiba kiszámítása RNN esetén: előre áramoltatás

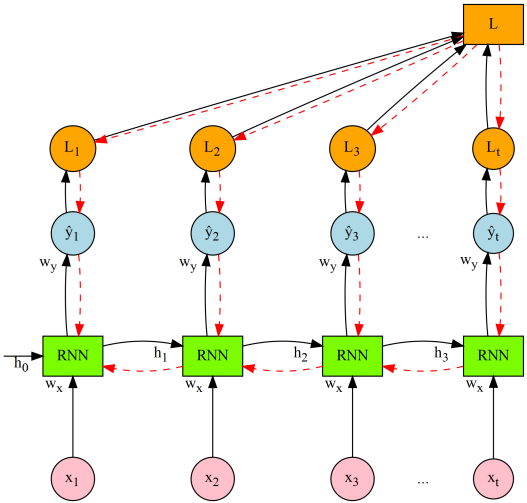
Előre áramoltatás során az RNN az input szekvencia elemeit **egyesével dolgozza fel időlépésenként**.

Minden t időlépésben kiszámolja L_t költséget, amelyet a végén aggregál valamilyen módszerrel, például **átlagolással vagy összegzéssel**.



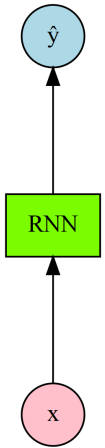
Hiba kiszámítása RNN esetén: időbeni visszaáramoltatás

1. Az output értékek kiszámítása minden időlépésre.
2. A hálózatot lehajtva minden időlépésre a költség kiszámítása.
3. A hálózatot feltekerve frissíteni a paramétereket.
4. Ismétlés a meghatározott lépésszámig.

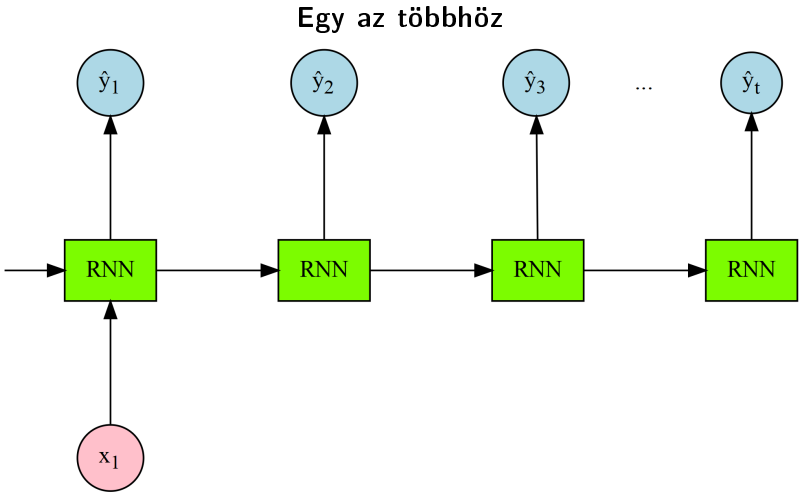


RNN típusok

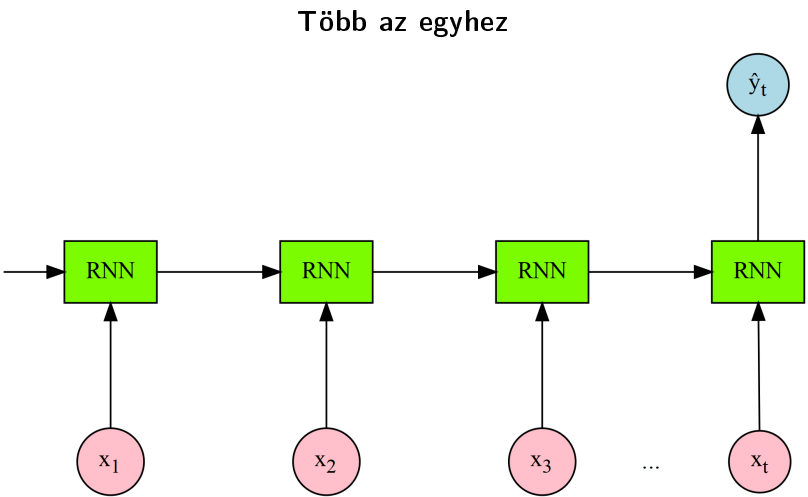
Egy az egyhez



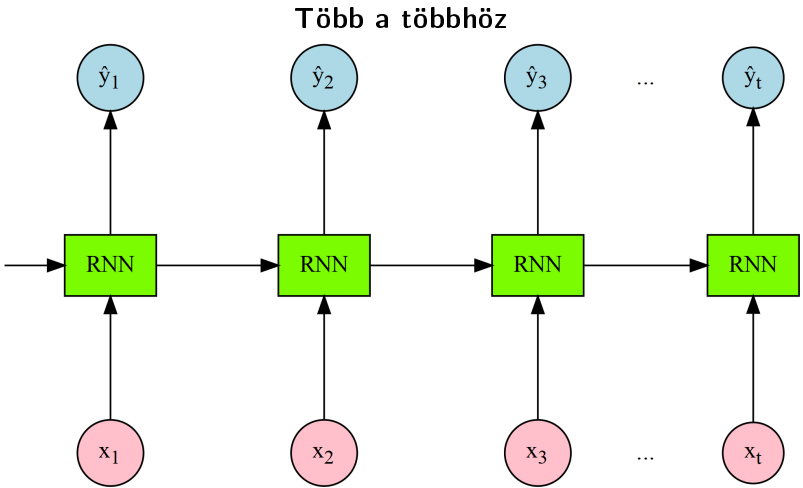
RNN típusok



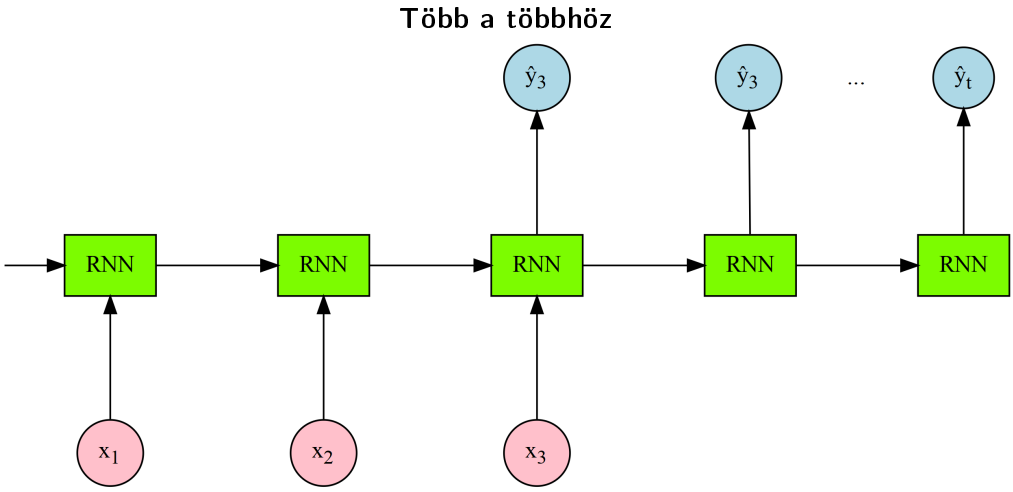
RNN típusok



RNN típusok

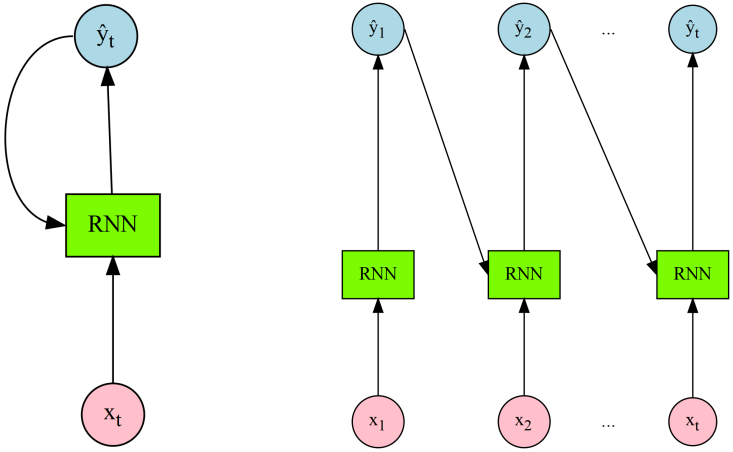


RNN típusok



RNN típusok

Output szerint becsatolt



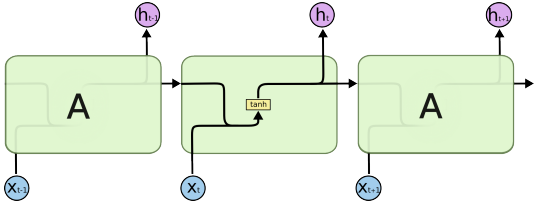
Hagyományos RNN hálózat

A hagyományos LSTM hálózat két bemenete az x_t **input vektor aktuális eleme** és a h_{t-1} **előző cella aktiváció**. Ez alapján állítja elő az aktuális cella állapotot:

$$h_t = tanh\left(W_h \cdot h_{t-1} + W_x \cdot x_t + b_h\right)$$

Ennek az architektúrának több hátránya is van:

- Igazából **csak egy nagyon mély hálózat**.
- A hiperbolikus tangens függvény gradiensei **a szélsőértékek felé haladva eltűnnek**.

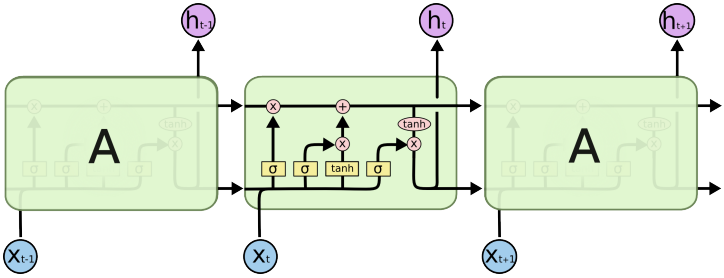


LSTM mint az RNN kiterjesztése

LSTM architektúra

Az LSTM (Long Short Term Memory) egy speciális neurális hálózat **architektúra szekvenciális adatok feldolgozására**.

Memóriacellákból és különböző **kapukból** (**input**, **felejtés**, **output**) áll amelyek segítik az információfolyam irányítását.

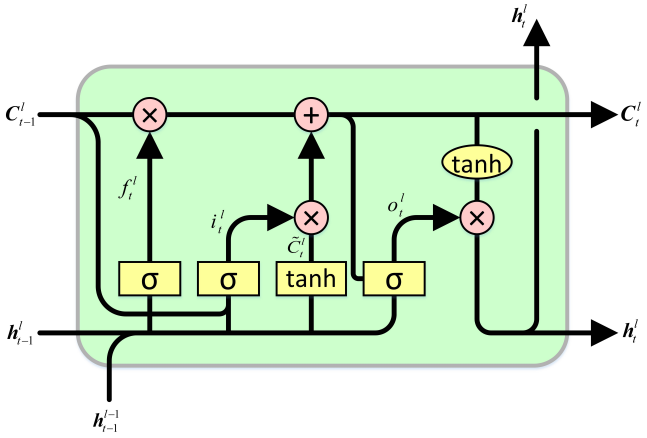


LSTM cella felépítése

Cella output

A cella outputja (h_t) **a hálózat aktivációja** t időlépésben.

A cella outputja az input vektor t. eleme és az előző cella aktivációja h_{t-1} alapján áll elő és a következő, $t+1$-edik cella inputjául szolgál.

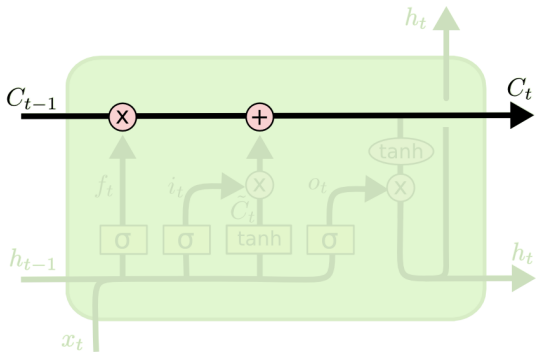


Elemi műveletek az LSTM cellában

Cella állapot

A cella állapot (C_t) LSTM hálózatokban egy **hosszútávú memória** ami **több időlépésen keresztül** képes információt eltárolni.

A cella **állapot kapukon keresztül képes változni**, amik meghatározzák, hogyan adódik hozzá vagy vonódik ki információ a cella állapotból.



Elemi műveletek az LSTM cellában

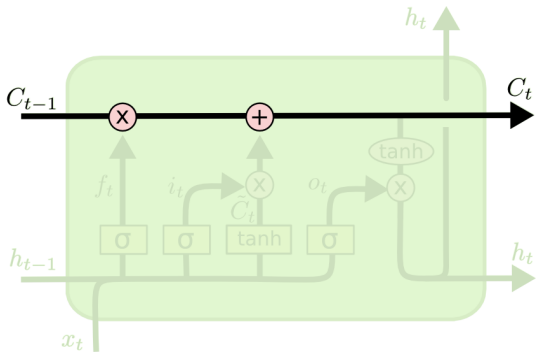
A cella állapot két operátora:

- $\bigotimes$: Elemenkénti szorzás:

$$a \otimes b = [a_1 \cdot b_1, a_2 \cdot b_2, \ldots, a_n \cdot b_n]$$

- $\bigoplus$: Elemenkénti összeadás:

$$a \oplus b = [a_1 + b_1, a_2 + b_2, \ldots, a_n + b_n]$$



Az LSTM cella kapui

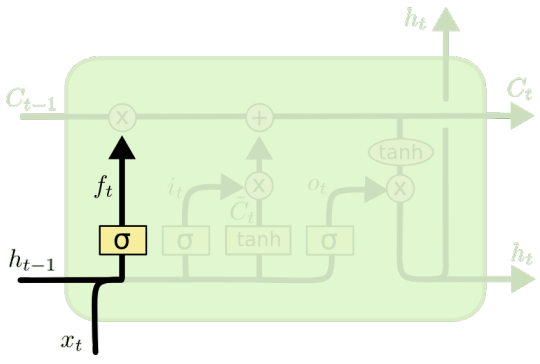
Felejtési kapu

A felejtési kapu az LSTM cellában egy matematikai kapu, amely két inputot fogad: C_{t-1} előző cella állapotot és x_t aktuális állapotot.

A kapu segítségével **a modell szelektíven tud törölni információt az előző cella állapotából**. A felejtési kapu outputja:

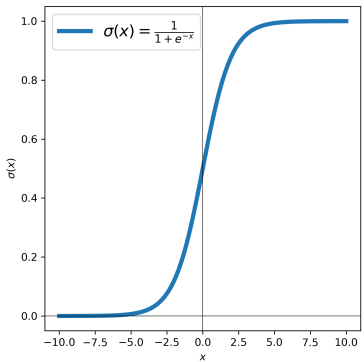
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

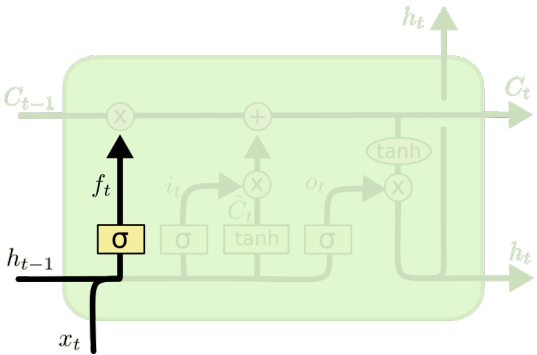
Ahol σ a szigmoid függvény, W_f a kapu súlymátrixa és b_f a torzítása.



Az LSTM cella kapui

A szigmoid függvény célja, hogy $]0, 1[$ intervallumba szorítsa be az input értékeket. Például felejtés esetén a 0 **közeli érték azt jelenti, hogy az információ nem fontos**.





Az LSTM cella kapui

Input kapu

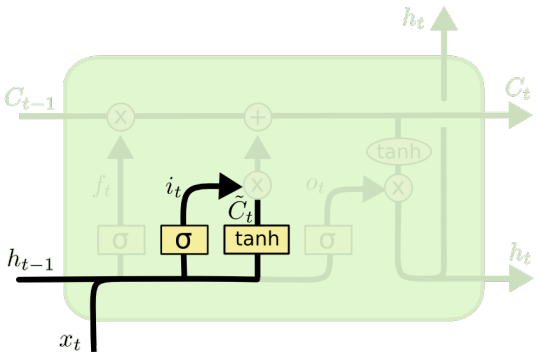
Az input kapu eldönti, **melyik új információ adódik hozzá a memóriához** az input vektor aktuális eleme alapján.

- Input kapu: az új információ memóriába áramlását irányítja:

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)$$

- Cellaállapot jelölt kapu: a már meglévő információ memóriába áramlását irányítja:

$$\tilde{C}_t = tanh\left(W_{\tilde{C}} \cdot [h_{t-1}, x_t] + b_{\tilde{C}}\right)$$

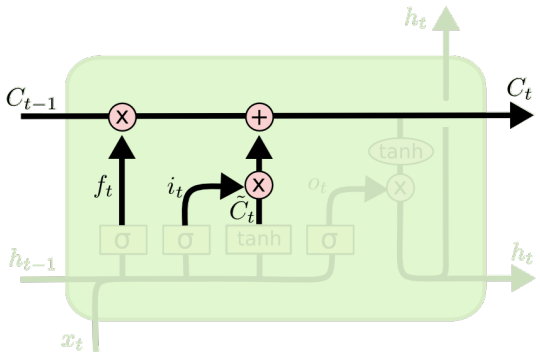


Az LSTM cella kapui

Cellaállapot előállítása

Az előző cellaállapot C_{t-1} és az aktuális cellaállapot jelölt $\tilde{C}_t$ alapján. **Ez a komponens adja a cella memóriáját, és feladata a fontos információk hosszú szakaszokon át való megtartása.**

$$C_t = f_t \otimes C_{t-1} \oplus i_t \otimes \tilde{C}_t$$



Az LSTM cella kapui

Output kapu

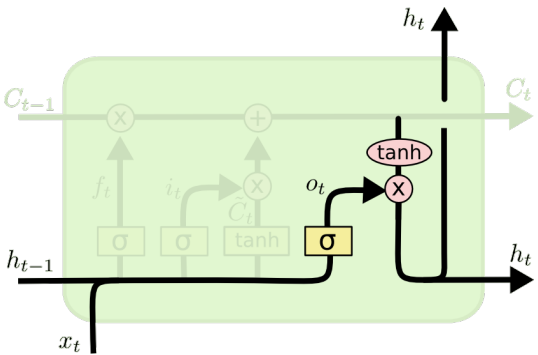
Az output kapu képes **szelektíven átadni fontos információt a következő cella inputjának**.

- Output kapu kimenete: megadja a cella állapotából mi mentődjön át a következő cella állapotába:

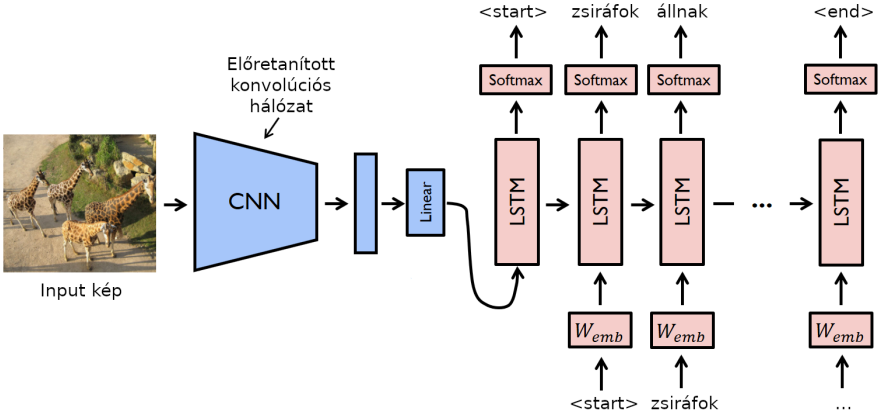
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

- Rejtett állapot: a cella állapota t időlépésben:

$$h_t = o_t \otimes tanh\left(\tilde{C}_t\right)$$



Példa LSTM használatára: képfeliratozás



Példa LSTM használatára: videó feliratozás