# Jednostavni povratni modeli

Martin Tutek, Petra Bevandić, Josip Šarić, Siniša Šegvić

2023.

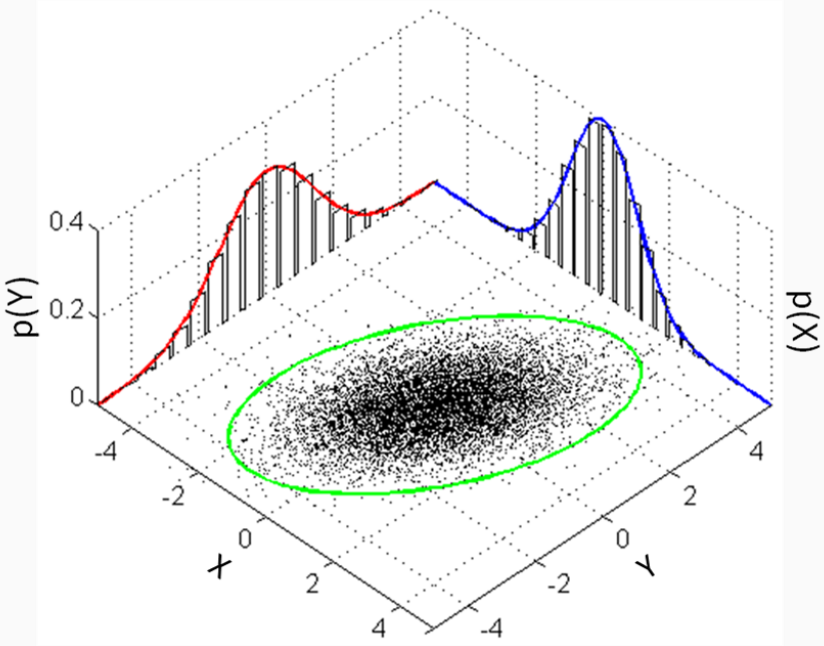
# Uvod

# U prethodnim predavanjima...

## ...bavili smo se modelima za podatke s:

- fiksnom dimenzionalnošću
- eksplicitnom međuovisnošću

primjeri slika iz CIFAR-10

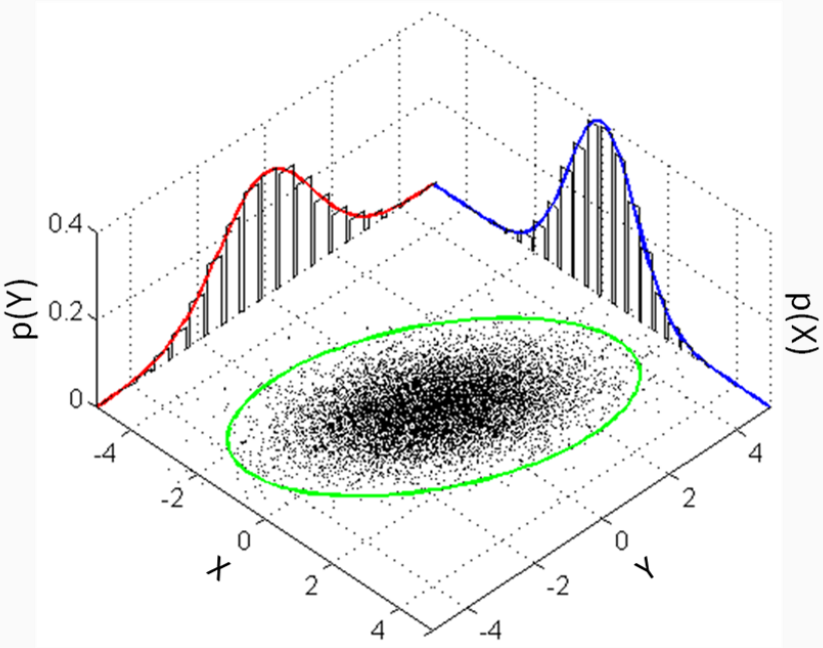



multivarijatna normalna distribucija

# U prethodnim predavanjima...

## ...bavili smo se modelima za podatke s:

- fiksnom dimenzionalnošću
- eksplicitnom međuovisnošću




multivarijatna normalna distribucija

primjeri slika iz CIFAR-10

# U prethodnim predavanjima...

**Fiksna ulazna dimenzionalnost:**

- svi uzorci multivarijatne normalne distribucije (vidi Lab #1) imaju istu dimenzionalnost: $x_i \in \mathbb{R}^d, \forall i$
- svaka slika iz CIFAR-10 ima istu dimenzionalnost 32x32x3 (WxHxC)
- **kako osigurati:** izrezivanjem, nadopunjavanjem, sažimanjem, uvećavanjem (interpolacijom)

**Međuovisnost:**

- općenita normalna razdioba $\mathcal{N}(\mu, \Sigma)$:
  - komponente nisu dekorelirane: $P(x, y) \neq P(x)P(y)$
  - eksplicitnu međuovisnost opisuje parametar $\Sigma$
- slike: bliski pikseli implicitno su jače korelirani od dalekih
- **kako osigurati:** primjenom konvolucijskih slojeva i pozicijskog kodiranja kod transformera

# Obrada prirodnog jezika

Prevedite ovu rečenicu na engleski jezik:

*Duboko učenje je super.* $\rightarrow$ *Deep learning is great.*

Rješenje problema zahtijeva sljedeće korake:

- segmentirati rečenicu (slijed slova, riječi ili slogova) na smislene komponente
- "razumjeti" da "duboko učenje" nije učenje na velikim dubinama (npr. na dnu Atlantika) nego grana računarstva
- shvatiti značenje izvorne rečenice bez da nam je netko rekao o kojem se jeziku radi
- generirati engleski prijevod s istim značenjem.

Tekstni podatci su čudni:

- **Nemaju** fiksnu dimenzionalnost (duljina rečenice može varirati)
- **Nemaju** eksplicitne obrasce međuovisnosti.
  - daleke ovisnosti: **tko** je prema povijesnim izvorima otkrio **Ameriku**?

# Obrada prirodnog jezika

Prevedite ovu rečenicu na vijetnamski jezik:

> 最寄りのインタネットカフェはどこですか
>
> → *Chờ internet ở đâu*
>
> → *Where is the nearest internet shop*

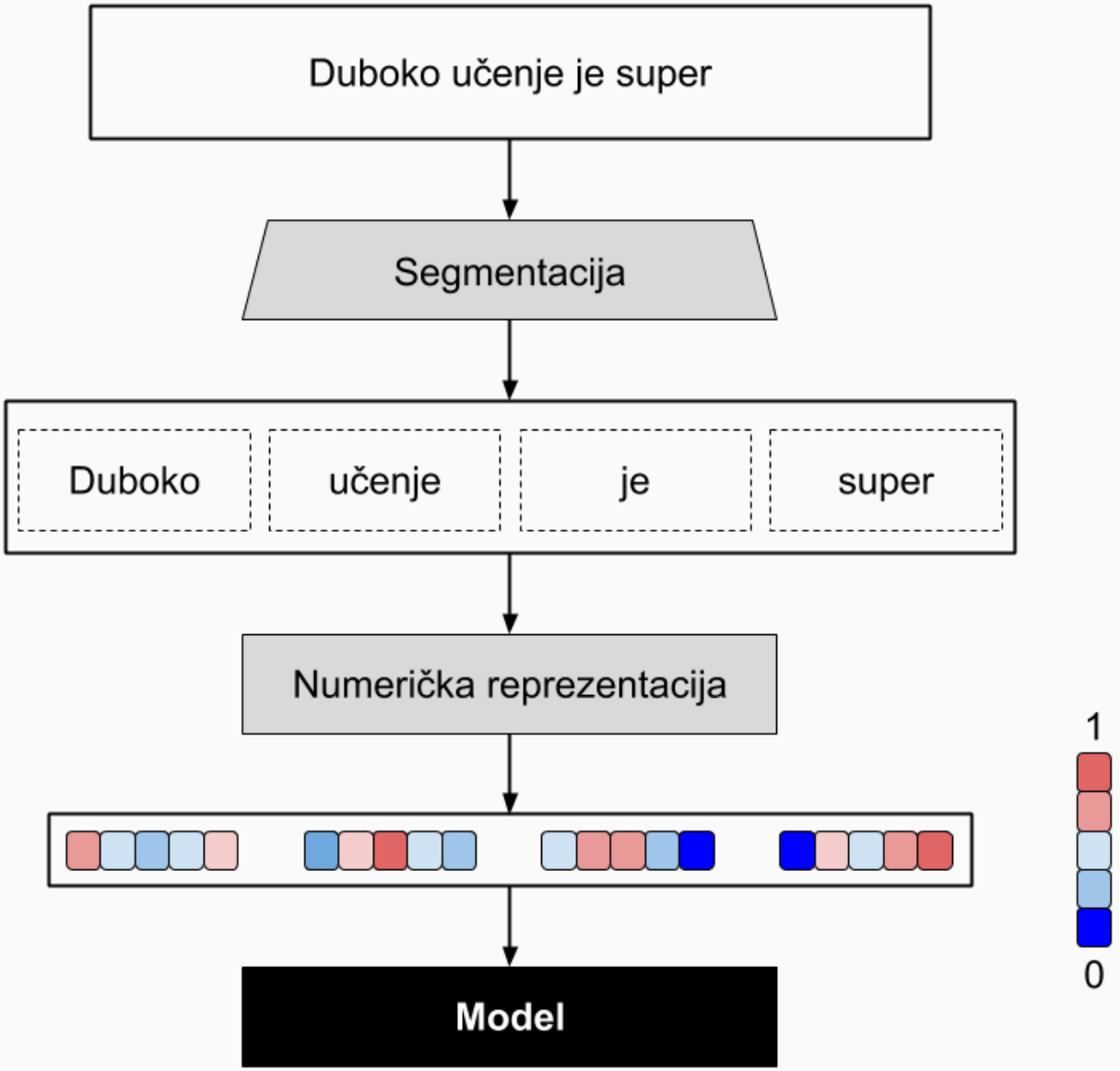
Jezik računalima izgleda otprilike kao što nama izgleda japanski (ali samo za nas koji ne razumijemo japanski)

Značenja i međusobna sličnost jezičnih jedinica su nam nepoznati.

Prije obrade tekstnih podataka algoritmima strojnog učenja možemo riječi prvo prevesti u vektorske reprezentacije.

# Reprezentacije dijelova rečenica

# Obrada teksta



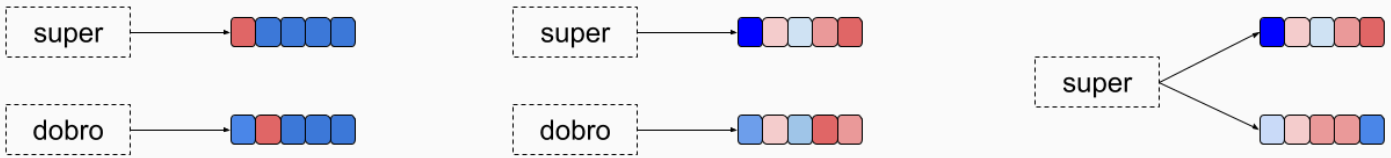
# Ovo predavanje:

Segmentacija teksta:

- tekst možemo segmentirati na **riječi**, slova ili podriječi[1]

Numerička reprezentacija:

- svakoj riječi dodijeliti **gustu** visokodimenzionalnu reprezentaciju
- alternative: jednojedinične (→*vreće riječi*) i višeprototipne.



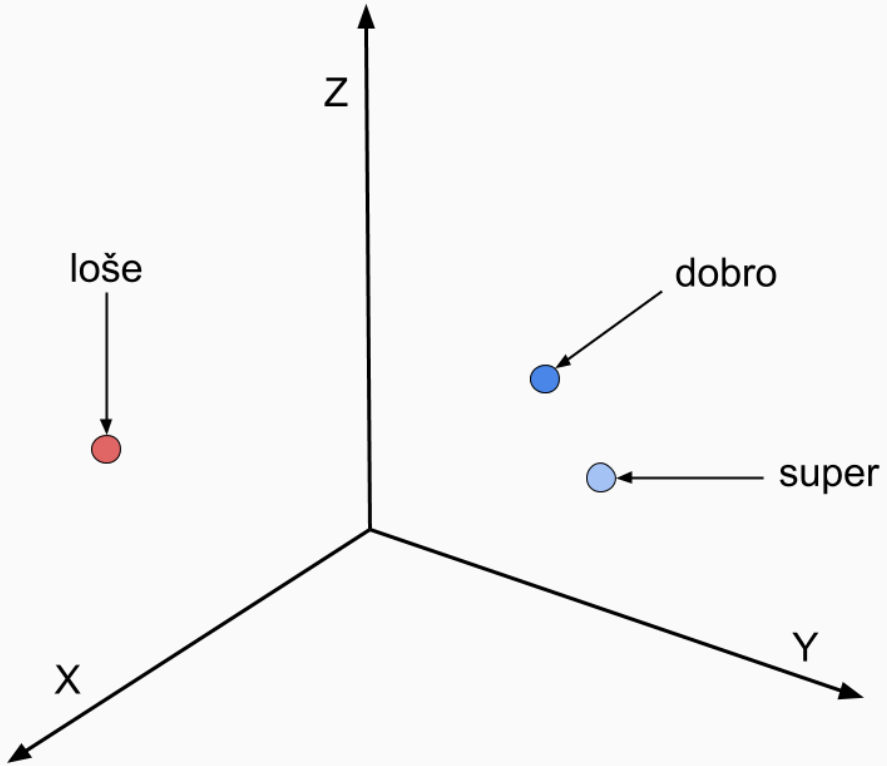

(a) (b) (c)

Jednojedinične **(a)**, guste **(b)** i višeprototipne reprezentacije **(c)**.

[1] `https://github.com/google/sentencepiece`

# Guste reprezentacije riječi

Udaljenost u prostoru reprezentacija trebala bi odgovarati semantičkoj ili sintaktičkoj sličnosti među riječima:




- kolika je udaljenost između jednojediničnih reprezentacija?
- kako izračunati udaljenost između višeprototipnih reprezentacija?

# Guste reprezentacije riječi

U praksi, dimenzionalnost reprezentacija puno je veća nego na slici.

Moramo se osloniti na intuicije iz niskodimenzionalnih prostora, iako one mogu biti varljive.

- If you are not used to thinking about hyper-planes in high-dimensional spaces, now is the time to learn.
- To deal with hyper-planes in a 14-dimensional space, visualize a 3-D space and say "fourteen" to yourself very loudly. Everyone does it.
  - But remember that going from 13-D to 14-D creates as much extra complexity as going from 2-D to 3-D.

Predavanja "Neural Networks for Machine Learning", Geoffrey Hinton

# Reprezentacije riječi

Početne reprezentacije riječi mogu se prednaučiti na nekom pomoćnom zadatku:

- pomoćni zadatci pogađaju skrivene riječi u zadanim lokalnim kontekstima na velikim korpusima stvarnog jezika (Wikipedia, Common crawl[2])
- modeli: word2vec[3] (CBOW, Skip-gram), GloVe[4], FastText[5]
- učenje optimira matricu ugrađivanja čiji retci sadrže vektorske reprezentacije odgovarajućih riječi.

U ovom kolegiju nećemo razmatrati pred-učenje reprezentacija:

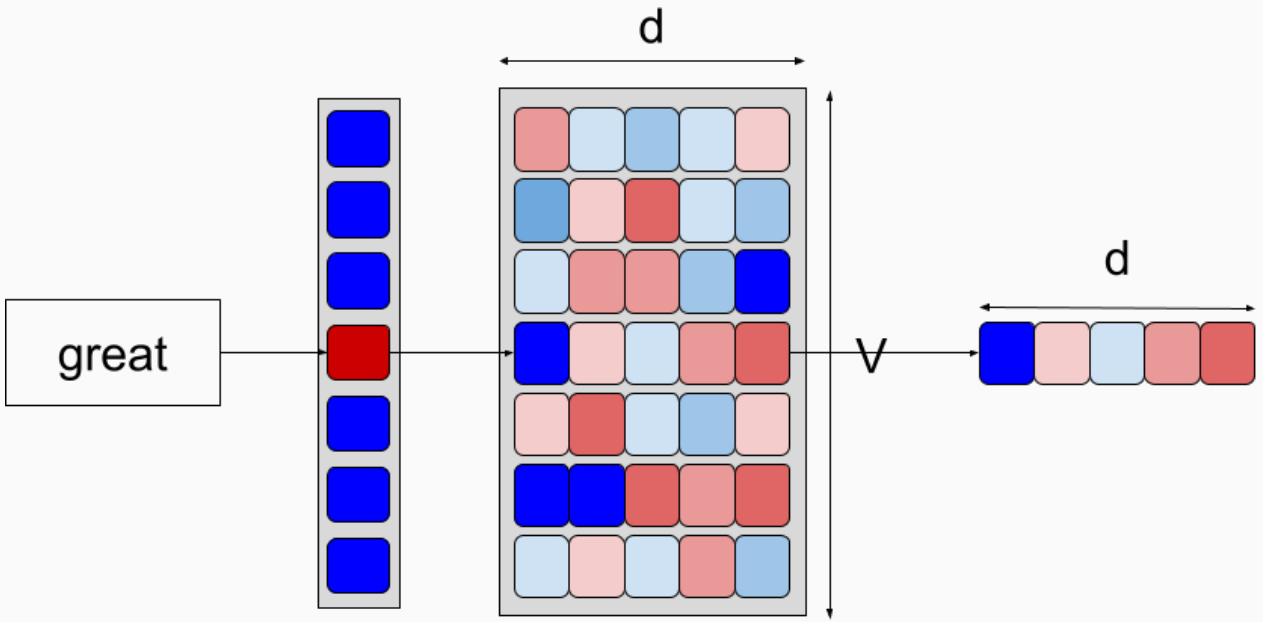
- umjesto toga, koristit ćemo ili slučajne inicijalizacije ili već prednaučene reprezentacije.

[2] https://commoncrawl.org/
[3] https://en.wikipedia.org/wiki/Word2vec
[4] https://nlp.stanford.edu/projects/glove/
[5] https://fasttext.cc/

# Matrice ugrađivanja u praksi

Reprezentacije riječi $\mathbf{x}_i$ dobivamo množenjem retčanog jednojediničnog koda riječi $\mathbf{e}_i$ s matricom ugrađivanja $\mathbf{E} \in \mathbb{R}^{V \times d}$:

$$\mathbf{x}_i = \mathbf{e}_i \cdot \mathbf{E} \qquad \mathbf{e}_i = \underbrace{[0, \ldots, 0, \overbrace{1}^{i}, 0, \ldots, 0]}_{V}$$



# Reprezentacije riječi: hiperparametri

Veličina rječnika: $V$

- broj jedinstvenih riječi koje model prepoznaje
- ovisi o dostupnoj memoriji i važnosti riječi[6]
- u praksi obično uzimamo $V$ najčešćih riječi iz korpusa
- preostale riječi obično i) izostavljamo ili ii) mijenjamo simbolom *<UNK>*

Dimenzionalnost ugrađivanja: $d$

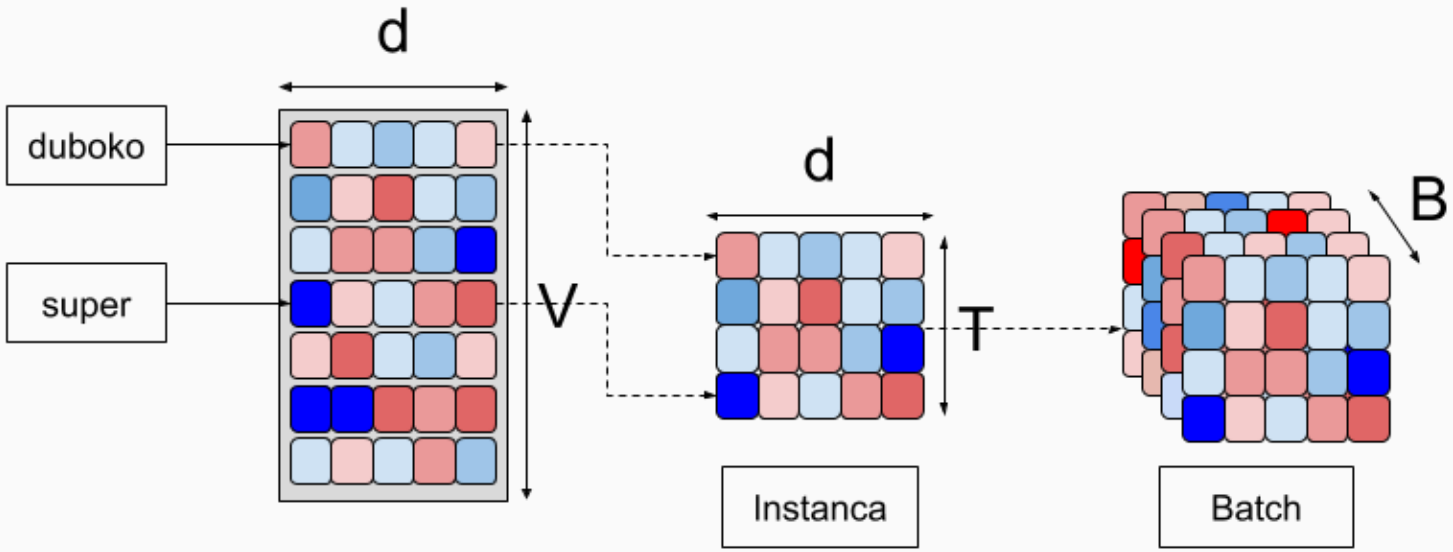
- uobičajeni izbor $d = 300$ pokazuje dobra svojstva na različitim zadatcima [7]
- u praksi, izbor će ovisiti i o dostupnoj memoriji, ciljanoj generalizacijskoj moći, te dostupnosti prednaučenih ugrađivanja.

[6] rijetke i neinformativne riječi obično se izostavljaju
[7] loši izbori mogu dovesti do podnaučenosti ili prenaučenosti

# Predstavljanje teksta

Odlomke i rečenice predstavljamo ulančavanjem reprezentacija riječi.

Duljinu teksta označavamo brojem riječi $T$ (*vremenska* dimenzija).




Ovaj pristup mora riješiti nekoliko problema: možete li ih prepoznati?

# Predstavljanje teksta

**Problem:** vremenska dimenzija **mijenja se** od rečenice do rečenice

- rečenice grupe za učenje imat će različite duljine
- u praksi, ovo rješavamo nadopunjavanjem i odsijecanjem.

**Cilj 1:** model obrađuje svaku riječ na isti način

- ovo ograničava dimenzionalnost modela i smanjuje prenaučenost.

**Cilj 2:** model treba biti osjetljiv na redoslijed riječi

- u suprotnom ne bismo mogli razlikovati rečenice sastavljene od istih riječi:
  - *Dog eats cat* i *Cat eats dog*.

Oba cilja možemo postići **povratnim modelima**

- međutim, prvo ćemo pogledati alternative za postizanje ovih ciljeva.

# Predstavljanje teksta: vremensko agregiranje sažimanjem

Ideja: predstaviti tekst sažimanjem reprezentacija riječi (*eng. mean/average pooling*)

$$r_\mu(text) = \frac{1}{T}\sum_t^\top x^{(t)}$$

- ovakvo agregiranje gubi informaciju o položaju riječi u tekstu, ali ne zahtijeva **parametre**

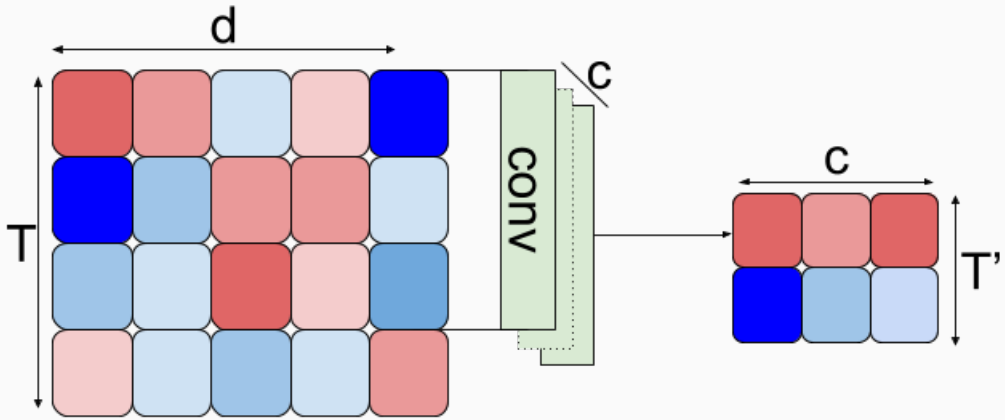
Varijante ovog pristupa: sažimanje zbrajanjem, **otežano sažimanje**

$$r_w(text) = \sum_t^\top w_t x^{(t)} \qquad w_t = f(x^{(t)}, \text{text})$$

- otežano sažimanje množi reprezentacije riječi faktorom koji ovisi o tekstu i o riječi (eg. IDF)

# Predstavljanje teksta: 1D convolution

Konvolucije zadržavaju informaciju o redoslijedu riječi:




I dalje trebamo sažimanje ako nam treba predikcija za cijeli odlomak.

Jednostavne konvolucijske arhitekture propuštaju *globalni* kontekst:

- može se ublažiti povećanjem dubine...
- ... ili dilatiranim konvolucijama ili vremenskim sažimanjem.

# Predstavljanje teksta: sažetak

Obrada prirodnog teksta (NLP) složen je zadatak:

- segmentirati tekst u riječi, **podriječi** ili slova
- detektirati i ispraviti tipografske greške
- istu misao možemo izraziti na različe načine

Pretpostavke u okviru ovih predavanja:

- tekst segmentiran na riječi
- reprezentacije riječi prednaučene ili slučajno inicijalizirane
- svakoj riječi dodjeljujemo točno *jedno* ugrađivanje

Pristupi za klasifikaciju teksta varijabilne duljine:

- konvolucije i sažimanja
- povratni modeli

# Obični povratni modeli

# Povratni modeli: motivacija

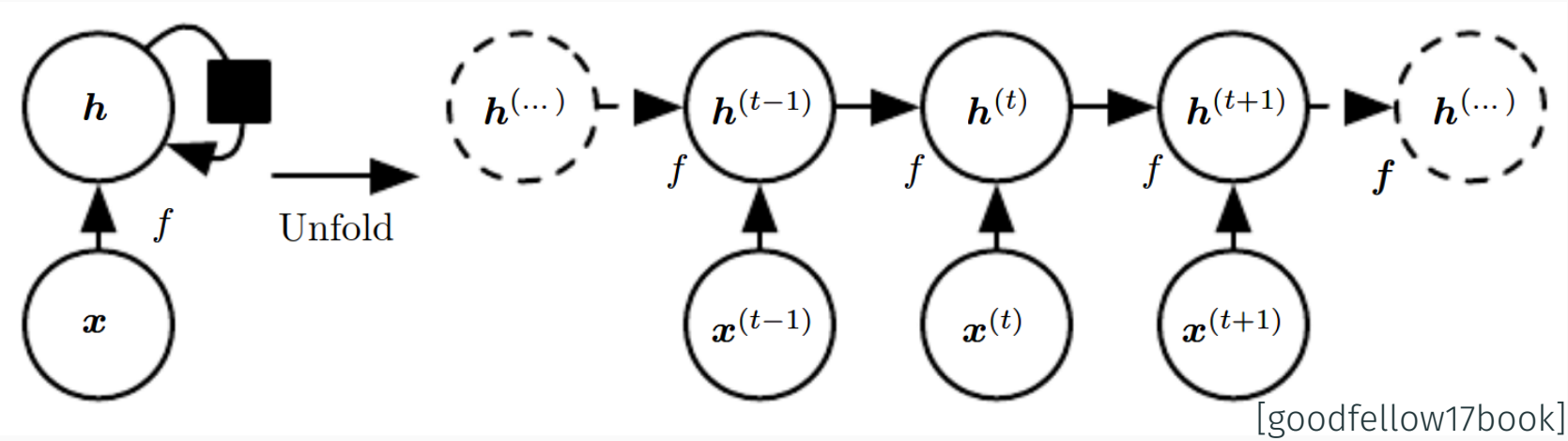
Tražimo prikladni model za slijedne podatke:

- proizvoljna duljina
- broj parametara ne ovisi o duljini slijeda
- model je osjetljiv na redoslijed podataka

Inspirirani **dinamičkim sustavima**:

- model $f$ ažurira **skriveno stanje** $h^{(t)}$ s obzirom na ulaz $x^{(t)}$:

$$h^{(t)} = f(x^{(t)}, h^{(t-1)})$$




[goodfellow17book]

# Povratni modeli: formulacija

Povratni model $f$ ekvivarijantan je s obzirom na vrijeme (položaj u slijedu):

$$h^{(t)} = f(x^{(t)}, h^{(t-1)})$$

Jednostavni povratni model imaju jednu nelinearnost po ulazu:

$$h^{(t)} = g(\underbrace{W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h}_{a^{(t)}})$$

- $W_{hh}$, $W_{xh}$, $b_h$ - parametri povratne afine transformacije
- $a^{(t)}$ - linearna povratna mjera
- $g$ - nelinearnost (sigmoida, tanh,...)

# Povratni modeli: intuicija

Stanje $h^{(t)}$ predstavlja **viđeni** dio slijeda:

- ako je model dobro naučen, $h$ će kodirati značenje teksta

Matrica $W_{xh}$ projicira ulaz u prostor reprezentacije stanja

- ovo bi trebalo odbaciti nepotrebne informacije

Matrica $W_{hh}$ modelira evoluciju stanja:

- ona opisuje utjecaj vremena i ulaza i odbacuje nepotrebne informacije iz stanja

$$h^{(t)} = g(W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h)$$

Dimenzionalnosti ulaza i stanja **mogu se razlikovati**:

$$h \in \mathbb{R}^h \qquad x \in \mathbb{R}^d$$

# Povratni modeli: izlazni sloj

Stanje $h^{(t)}$ predstavlja svu viđenu informaciju:

- osim značenja, stanje pamti i neke dodatne informacije
  - npr. dvosmislene riječi ili navode osoba ili lokacija
- neke od ovih informacija mogu biti **nebitne** za predikciju.

*Izlazni sloj* projicira stanje na prostor predikcija:

$$o^{(t)} = W_{hy} h^{(t)} + b_o \tag{1}$$

- ovaj korak *filtrira* nevažne informacije
- vektor $o^{(t)}$ sadrži logite predikcija u trenutku $t$
- **[!!]** RNN ćelija Pytorcha ne sadrži izlaznu projekciju – morate je sami dodati.

# Povratni modeli: formulacija

Jednostavni povratni model definiran je sljedećim jednadžbama

1. ažuriranje skrivenog stanja:

$$h^{(t)} = tanh(W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h)$$

2. projekcija u prostor izlaznih logita:

$$o^{(t)} = W_{hy}h^{(t)} + b_o$$

U praksi, stanje obično aktiviramo hiperbolnim tangensom:

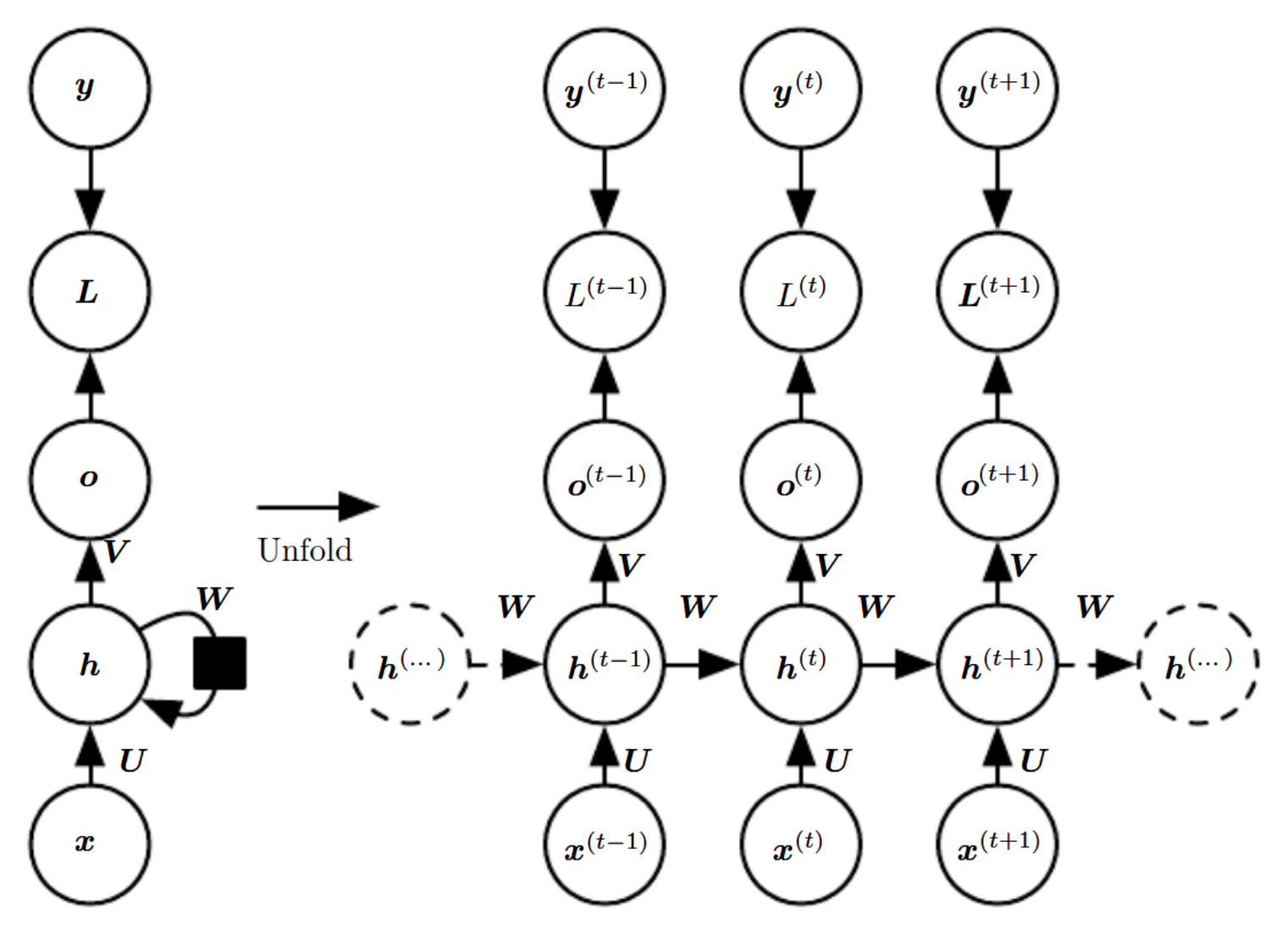
- preostale mogućnosti su sigmoida i zglobnica.

Dimenzije parametara: $W_{hh} \in \mathbb{R}^{h \times h}$, $W_{xh} \in \mathbb{R}^{h \times d}$, $W_{hy} \in \mathbb{R}^{y \times h}$

- $d$ i $y$ označavaju ulaznu i izlaznu dimenzionalnost

**[!!]** Notacija iz knjige: $W := W_{hh}$, $U := W_{xh}$, $V := W_{hy}$, $b := b_h$, $c := b_o$

# Povratni modeli: vizualizacija



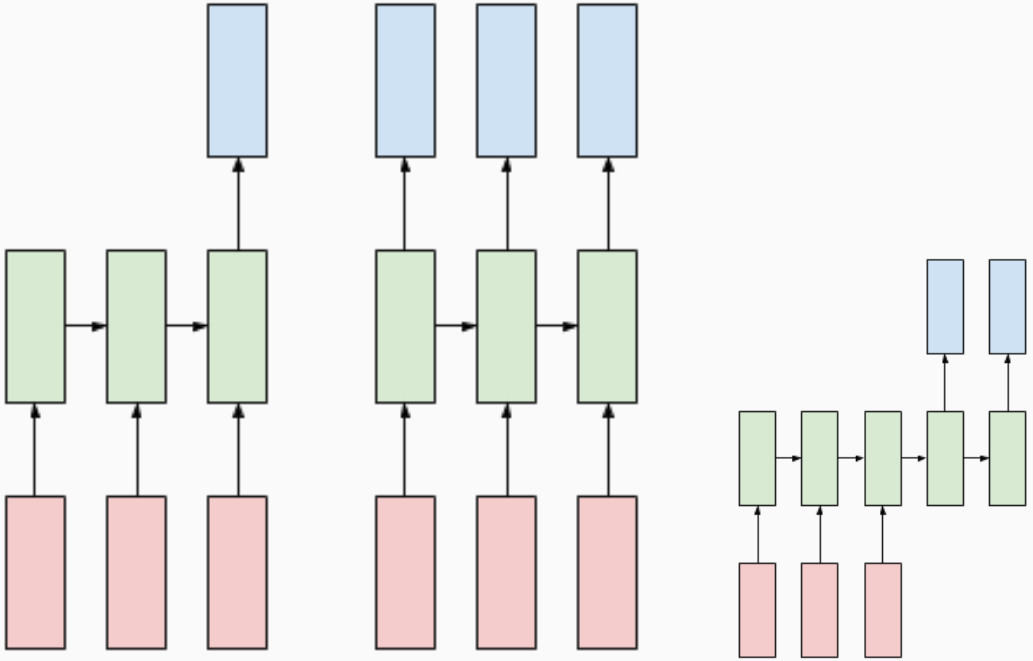

Razmotani povratni model s ulazima $x^{(t)}$, gubitkom $L^{(t)}$ i izlazima $o^{(t)}$.

# Povratni modeli: zadatci

Povratni model može generirati izlaz u svakom trenutku $t$ – ali treba li nam to?

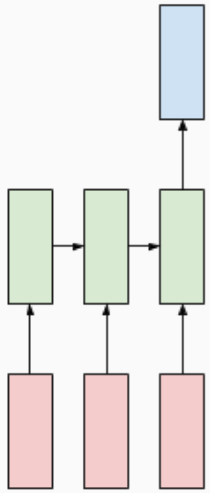
- konfiguracija izlaza modela ovisi o zadatku!

Pokušajte smisliti zadatke koji zahtijevaju samo jedan izlaz **(a)** po jedan izlaz za svaki ulaz **(b)**, ili promjenljivi broj izlaza ali više od jednog **(c)**.



# Jezični zadatci

Klasifikacija slijeda: temeljni problem u razumijevanju jezika.



Klasifikacija slijeda

Potrebno je odrediti jednu izlaznu varijablu za cijeli ulazni niz

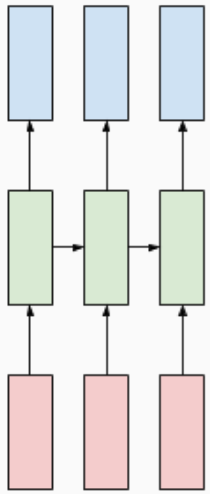
Različiti klasifikacijski problemi:

- analiza sentimenta,
- kategorizacija dokumenata,
- određivanje glazbenog žanra, ...

Izlazni sloj prima samo **posljednje** latentno stanje.

# Jezični zadatci

”Označavanje” slijeda: zahtijevamo predikciju u svakom ulazu.



”Označavanje” slijeda

Potrebno odrediti slijed varijabli: svaka predikcija odgovara točno jednom ulazu.

Različiti zadatci guste predikcije:

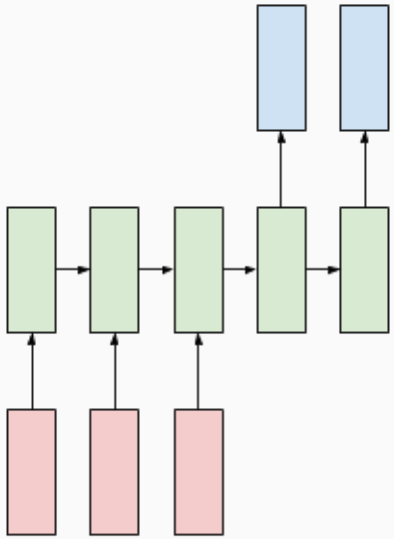
- predikcija vrste riječi,
- raspoznavanje imenovanih entiteta,
- sažimanje teksta ekstrakcijom,
- detekcija okvira videa, ...

Predikcije često generiramo s malim vremenskim pomakom u odnosu na odgovarajuće ulaze.

Naveli smo navodnike jer riječ *označavanje* obično koristimo kad radnju vrše osobe (bolje: označavanje slijeda $\rightarrow$ gusta predikcija).

# Jezični zadatci

Zadatci oblika slijed-u-slijed (*seq2seq*) podrazumijevaju složenije oblike preslikavanja ulaznog slijeda u izlazni.



slijed-u-slijed

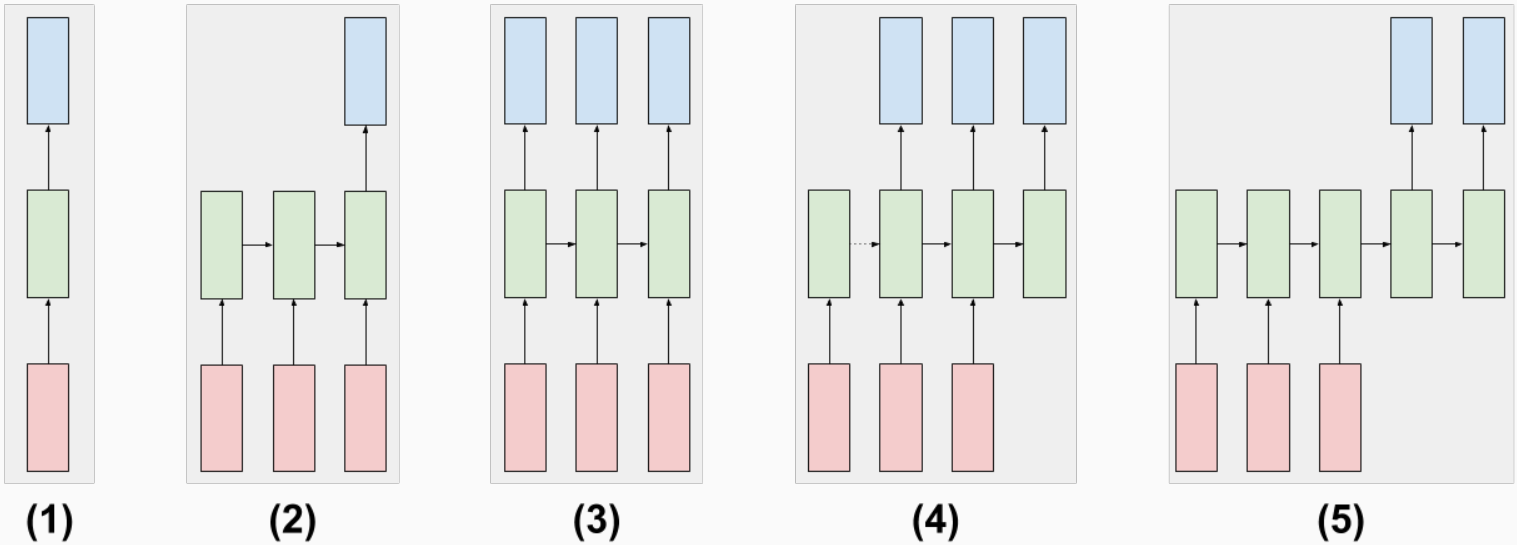
Zadatak je ulazni slijed preslikati u ciljni slijed nepoznate duljine

- strojno prevođenje,
- apstraktivno sažimanje teksta, ...

Razmotrite probleme koji bi se ovdje mogli javiti:

- kako započeti prijevod?
- bismo li uvijek trebali prediktirati najvjerojatniju riječ?
- kako odlučiti kada zaključiti predikciju?

# Jezični zadatci




Smjernice za prepoznavanje zadataka:

- kad god broj izlaza odgovara broju ulaza, zadatak odgovara gustoj predikciji, neovisno o tome postoji li pomak **(4)** ili ne **(3)**
- zadatke oblika slijed-u-slijed **(5)** često razdvajamo na apsorbiranje ulaza (gdje ne provodimo predikciju) i proizvođenje izlaza (gdje generiramo čitavi izlazni slijed)

# Analiza: klasifikacija slijeda

# Analiza sentimenta

Klasifikacijski problem: odrediti sentiment ulaznog teksta

- binarna formulacija: pozitivan/negativan
- kategorička formulacija: brojčana ocjena [1,10]
  - ovo bi se moglo formulirati i kao regresija...
- skupovi podataka: IMDB[8], YELP[9]
- pozitivan primjer iz IMDB:

```
... was the story that blew me away. hurray for Takahisa
```

Naša analiza pretpostavlja binarnu formulaciju i zanemaruje (značajne) probleme predobrade teksta.

[8] https://ai.stanford.edu/~amaas/data/sentiment/
[9] https://www.yelp.com/dataset/

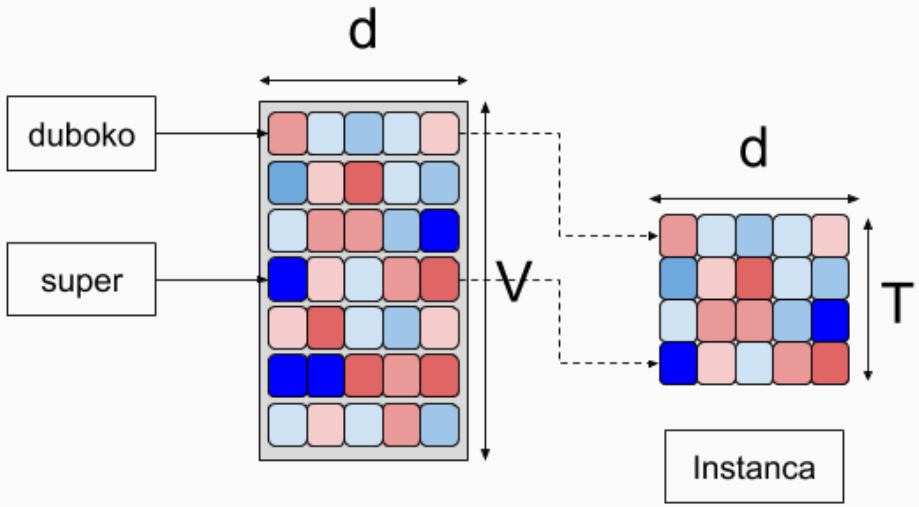
# Analiza sentimenta: uvod

Ciljna varijabla:

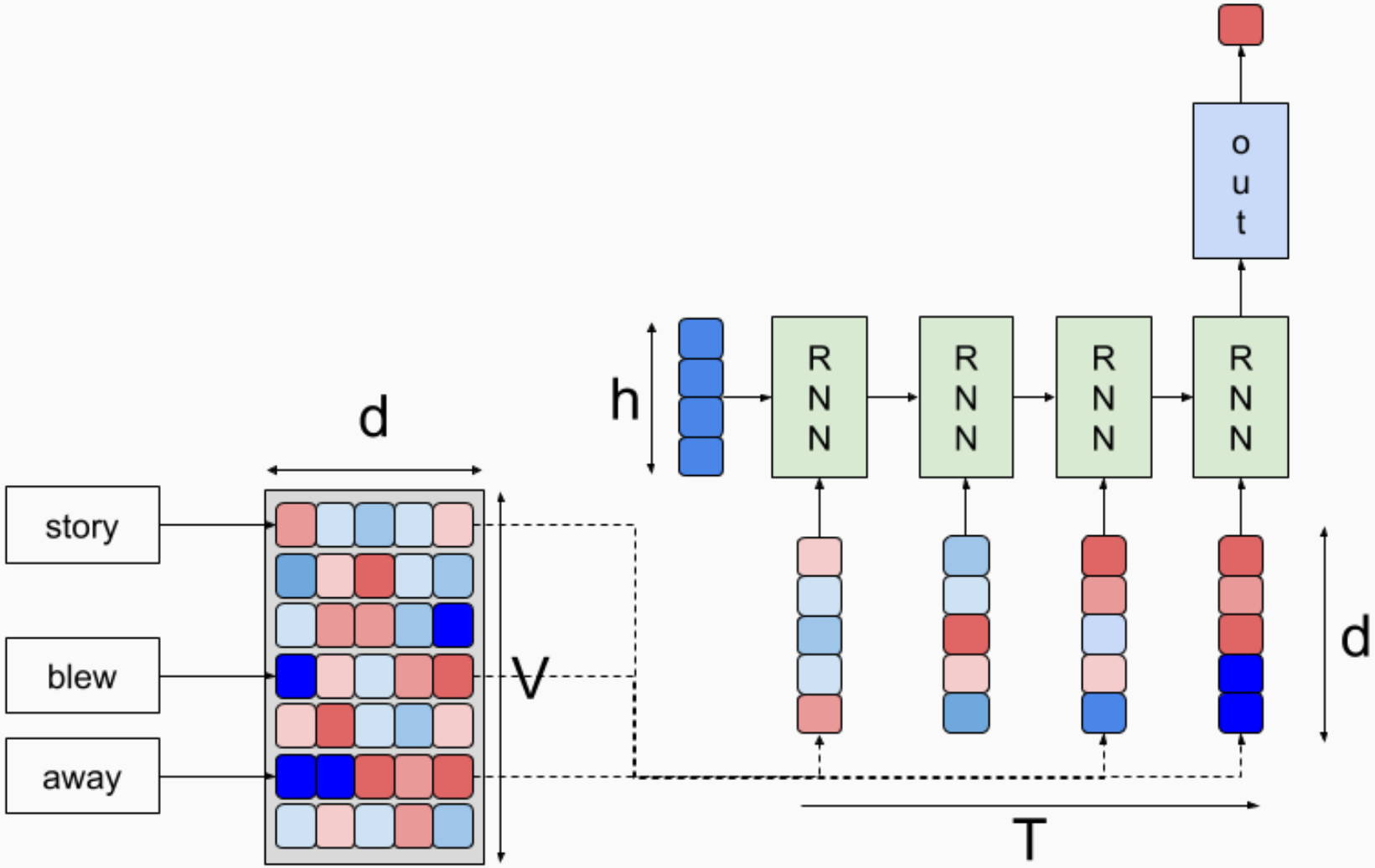
- Sentiment: $\{\text{pozitivan}, \text{negativan}\}$
- Vrijednosti varijable predstavljamo indeksima: $\{0, 1\}$

Ulazne varijable:

- slijed reprezentacija riječi, podriječi ili slova
- moramo odabrati veličinu vokabulara i svaku riječ zamijeniti s odgovarajućim ugrađivanjem ili simbolom `<UNK>`



# Analiza sentimenta: skica



# Analiza sentimenta: pseudokod

**Algoritam 1:** Plain RNN for sequence classification

**Single output RNN** $(X, W_{hh}, W_{xh}, W_{hy}, b_h, b_o)$

**inputs :** A sequence of vectors $X$, parameters $W_{...}$, $b_{...}$

**output:** Logits $o$

$h_t \leftarrow zero\_init$;

**foreach** $x$ *in* $X$ **do**

$a_t \leftarrow W_{hh} h_t + W_{xh} x + b_h$;

$h_t \leftarrow tanh(a_t)$;

**end**

$o \leftarrow W_{hy} h_t + b_y$;

**return** $o$;

Ovaj algoritam možemo prilagoditi za gustu predikciju tako da generiranje izlaznih elemenata jednostavno pomaknemo u petlju.

# Analiza: gusta predikcija

# Raspoznavanje vrste riječi

Gusta predikcija vrste riječi za svaku riječ ulaza: raspoznava

- višerazredni izlaz: imenica, pridjev, glagol, ...

Skupovi: Penn Treebank (PTB) (paywall), Universal Dependencies[10] (UD)

- PTB taxonomy: `https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html`

Primjer rečenice iz korpusa SETimes (dio UD) i odgovarajuće oznake:

```
U   Bruxellesu izrazili su  zabrinutost.
ADP PROPN      VERB     AUX NOUN        .
```

[10] `https://universaldependencies.org/`

# Raspoznavanje vrste riječi: uvod

Ciljne varijable:

- vrste riječi: $\{ADP, PROPN, ...\}$
- koristimo indekse razreda: $\{1, \ldots, Y\}$

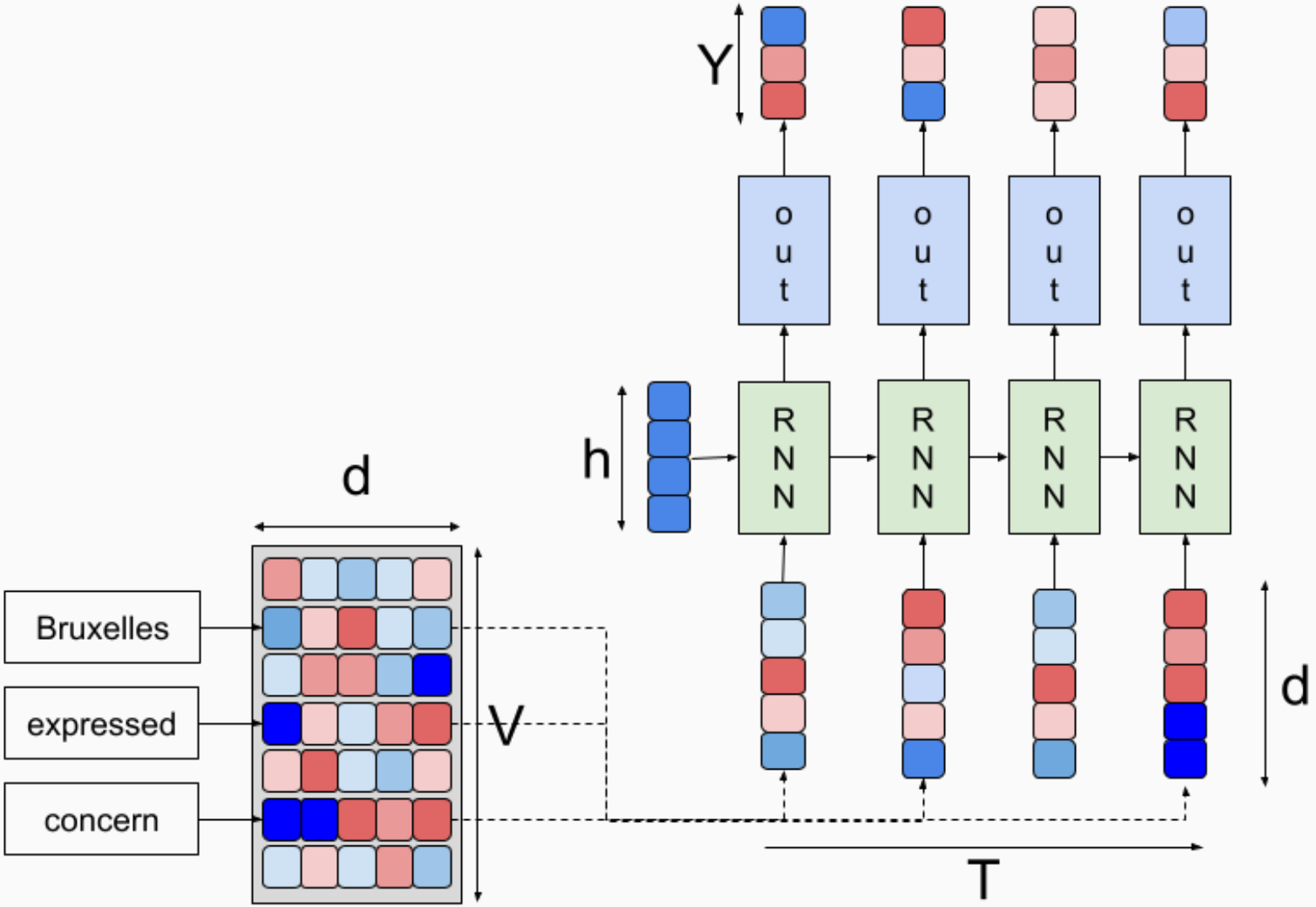
Ulazne varijable:

- slijed reprezentacija riječi, podriječi ili slova
- moramo odabrati veličinu vokabulara i svaku riječ zamijeniti s odgovarajućim ugrađivanjem ili simbolom `<UNK>`

Vrlo slično klasifikaciji slijeda

- važna razlika: dimenzija ciljne varijable (2 vs N)
- koristimo guste predikcije umjesto predikcija koje obuhvaćaju cijeli slijed

# Part-of-speech tagging: top-level sketch



# Raspoznavanje vrste riječi: pseudokod

**Algoritam 2:** Pseudokod RNNa za višerazrednu predikciju na razini simbola

**Dense sequence prediction RNN** $(X, W_{hh}, W_{xh}, W_{hy}, b_h, b_o)$

**inputs :** A sequence of vectors $X$, parameters $W_{...}$, $b_{...}$

**output:** A sequence of logits $\hat{y}$

$h_t \leftarrow zero\_init$;

$o \leftarrow [\,]$;

**foreach** $x$ *in* $X$ **do**

$a_t \leftarrow W_{hh}h_t + W_{xh}x + b_h$;

$h_t \leftarrow tanh(a_t)$;

$o_t \leftarrow W_{hy}h_t + b_y$;

$o.append(o_t)$

**end**

**return** $o$;

# Povratni modeli: sažetak

Povratni model djeluje kao dinamički sustav:

- ulazna informacija ugrađuje se u latentno stanje korak po korak
- latentno stanje iterativno se ažurira s obzirom na staro stanje i tekući ulaz

Stanje najčešće ne sadržava predikcije:

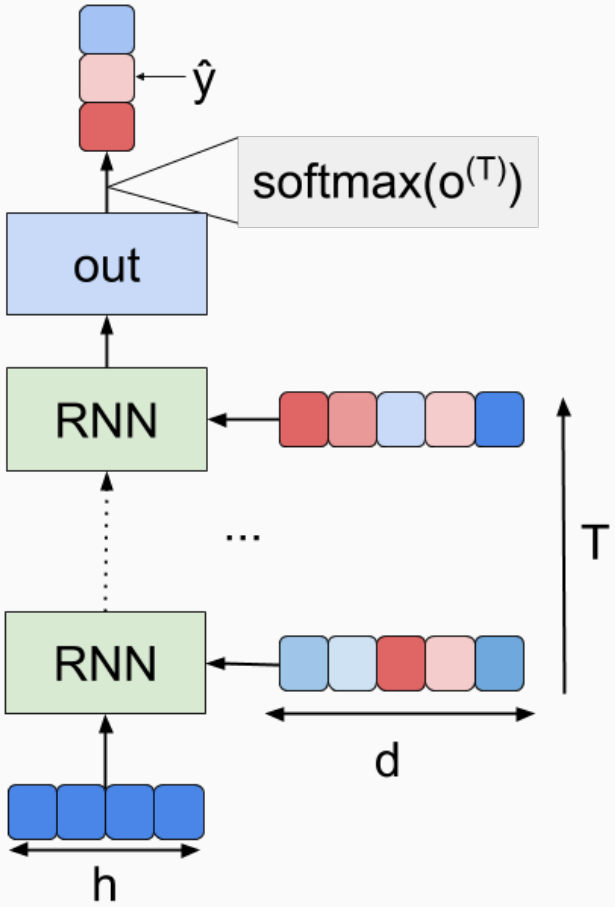
- predikcije izražavamo projiciranjem stanja
- broj izlaznih projekcija ovisi o vrsti zadatka (samo jedna, jedna po ulazu ili varijabilan broj)
- ako ima više izlaza, parametri izlaznih projekcija se **dijele**

Tri glavna zadatka raspoznavanja slijedova su **raspoznavanje slijeda**, **gusta predikcija** i **prevođenje iz slijeda u slijed**.

# Učenje povratnih modela

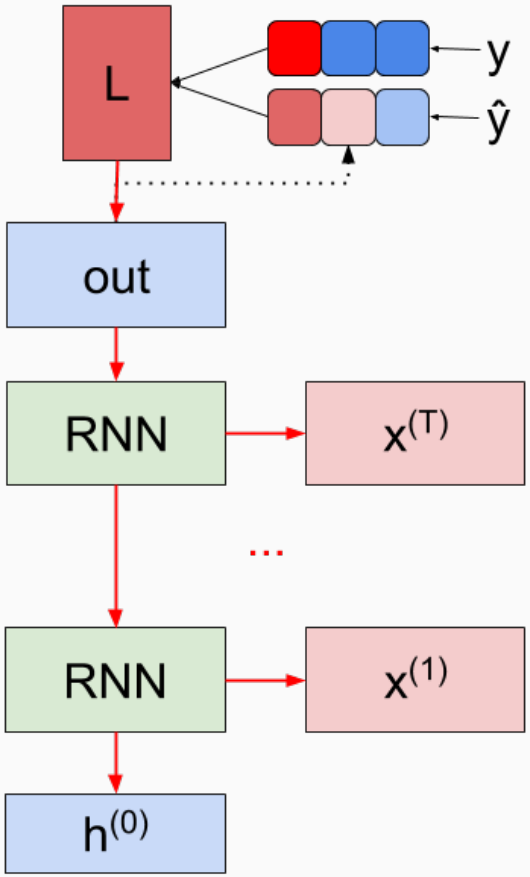
# Uvod

Povratni model možemo razmotati u klasični potpuno povezani model s odvojenim ulazima i dijeljenim parametrima:

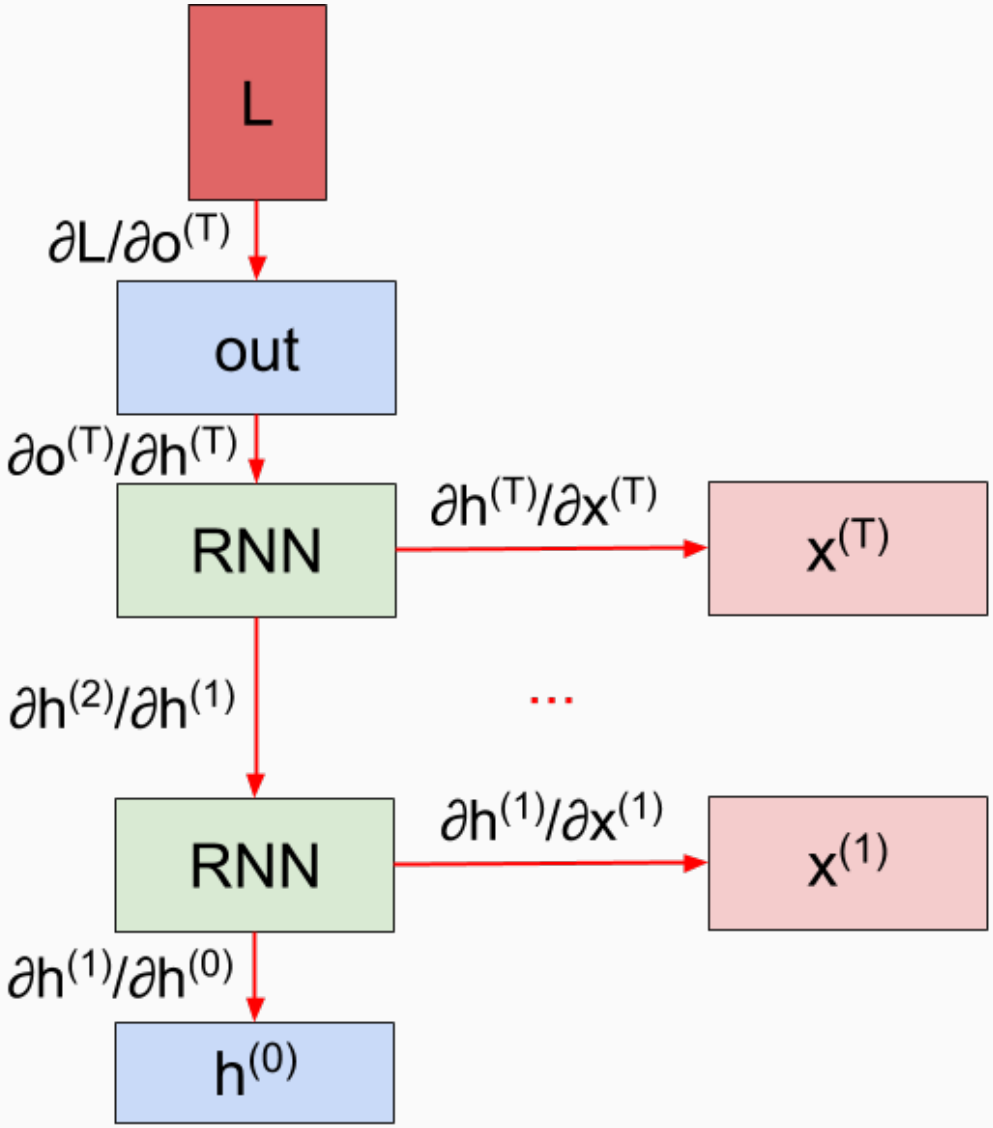


# Primjer: klasifikacija niza

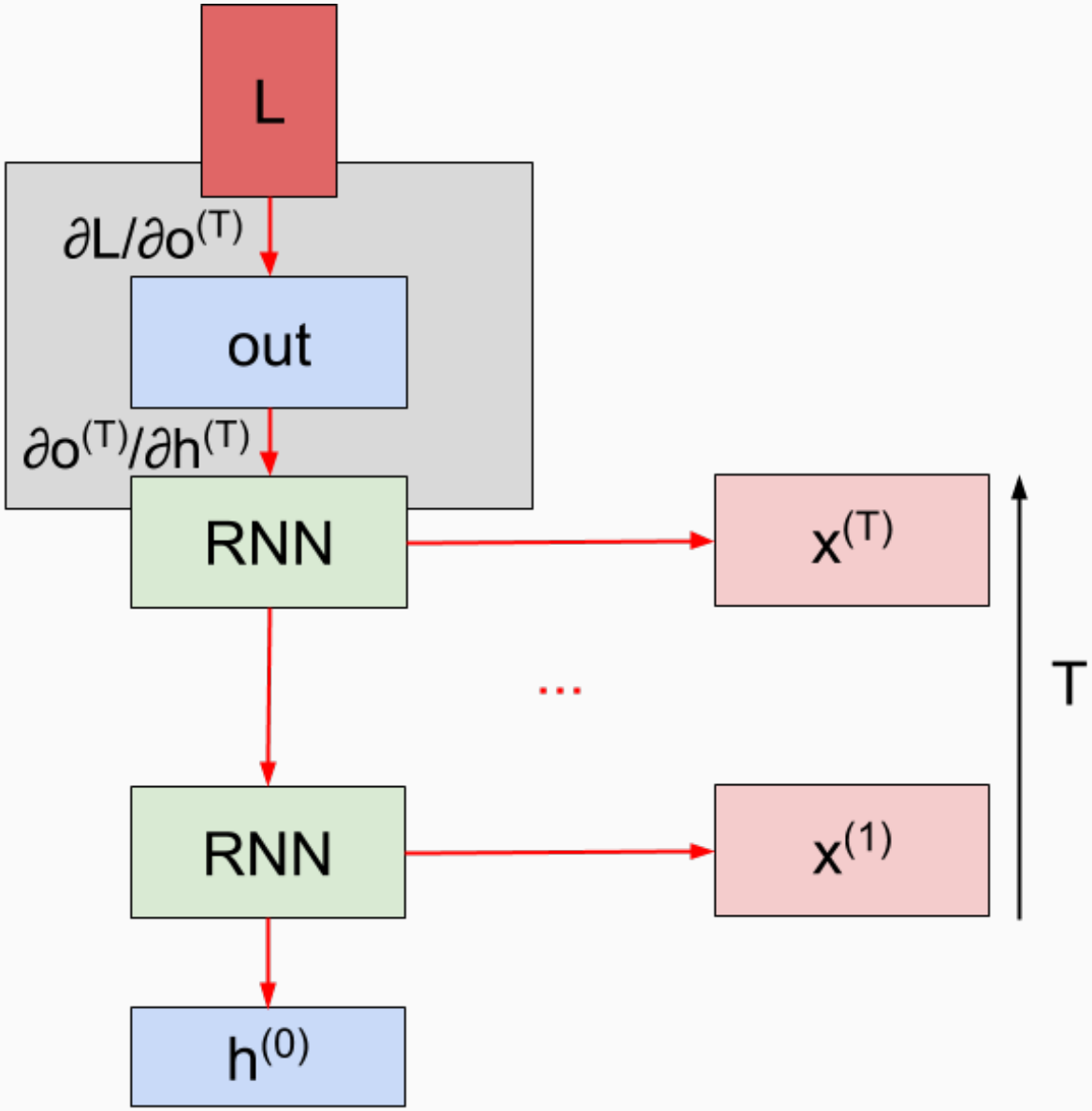
Model za prediktiranje na razini cijelog niza možemo učiti standardnim backpropom:



# Primjer: klasifikacija niza (2)



# Povratno učenje: prolaz kroz izlazni sloj



# Povratno učenje: prolaz kroz izlazni sloj (2)

Predikcije dobivamo aktiviranjem izlaza (podsjetnik):

$$o^{(T)} = W_{hy} h^{(T)} + b_y$$
$$\hat{y} = \text{softmax}(o^{(T)})$$

Pretpostavljamo unakrsnu entropiju kao standardni gubitak:

$$\mathbb{L} = -\sum_j y_j \cdot \log \hat{y}_j$$

Gradijenti s obzirom na izlazne aktivacije:

$$\frac{\partial \mathbb{L}}{\partial o^{(T)}} = (\underbrace{softmax(\mathbf{o}^{(\mathsf{T})})}_{\hat{y}} - y)^\top$$

# Povratno učenje: prolaz kroz izlazni sloj (3)

Jednadžbe predikcije izlaza (podsjetnik):

$$o^{(T)} = W_{hy} h^{(T)} + b_y$$

Gradijenti s obzirom na parametre izlaznog sloja $W_{hy}$ i $b_y$:

$$\frac{\partial \mathbb{L}}{\partial W_{hy}} = \frac{\partial \mathbb{L}}{\partial o^{(T)}} \frac{\partial o^{(T)}}{\partial W_{hy}} = \ldots = (\hat{y} - y) \cdot (h^{(T)})^\top$$

$$\frac{\partial \mathbb{L}}{\partial b_y} = \frac{\partial \mathbb{L}}{\partial o^{(T)}} \frac{\partial o^{(T)}}{\partial b_y} = (\hat{y} - y)^\top$$

Gradient s obzirom na posljednje stanje:

$$\frac{\partial \mathbb{L}}{\partial h^{(T)}} = \frac{\partial \mathbb{L}}{\partial o^{(T)}} \frac{\partial o^{(T)}}{\partial h^{(T)}} = (\hat{y} - y)^\top W_{hy}$$

# Povratno učenje: ažuriranje stanja

Stanje u trenutku $t$ ovisi o stanjima u svim prethodnim trenutcima:

$$h^{(t)} = f(h^{(t-1)}, \ldots, h^{(0)})$$

Unatražni prolaz u koraku $t = T$ (posljednji korak, podsjetnik)

$$\frac{\partial \mathbb{L}}{\partial h^{(T)}} = \frac{\partial \mathbb{L}}{\partial o^{(T)}} \frac{\partial o^{(T)}}{\partial h^{(T)}} = \frac{\partial \mathbb{L}}{\partial o^{(T)}} W_{hy} = (\hat{y} - y)^\top W_{hy}$$

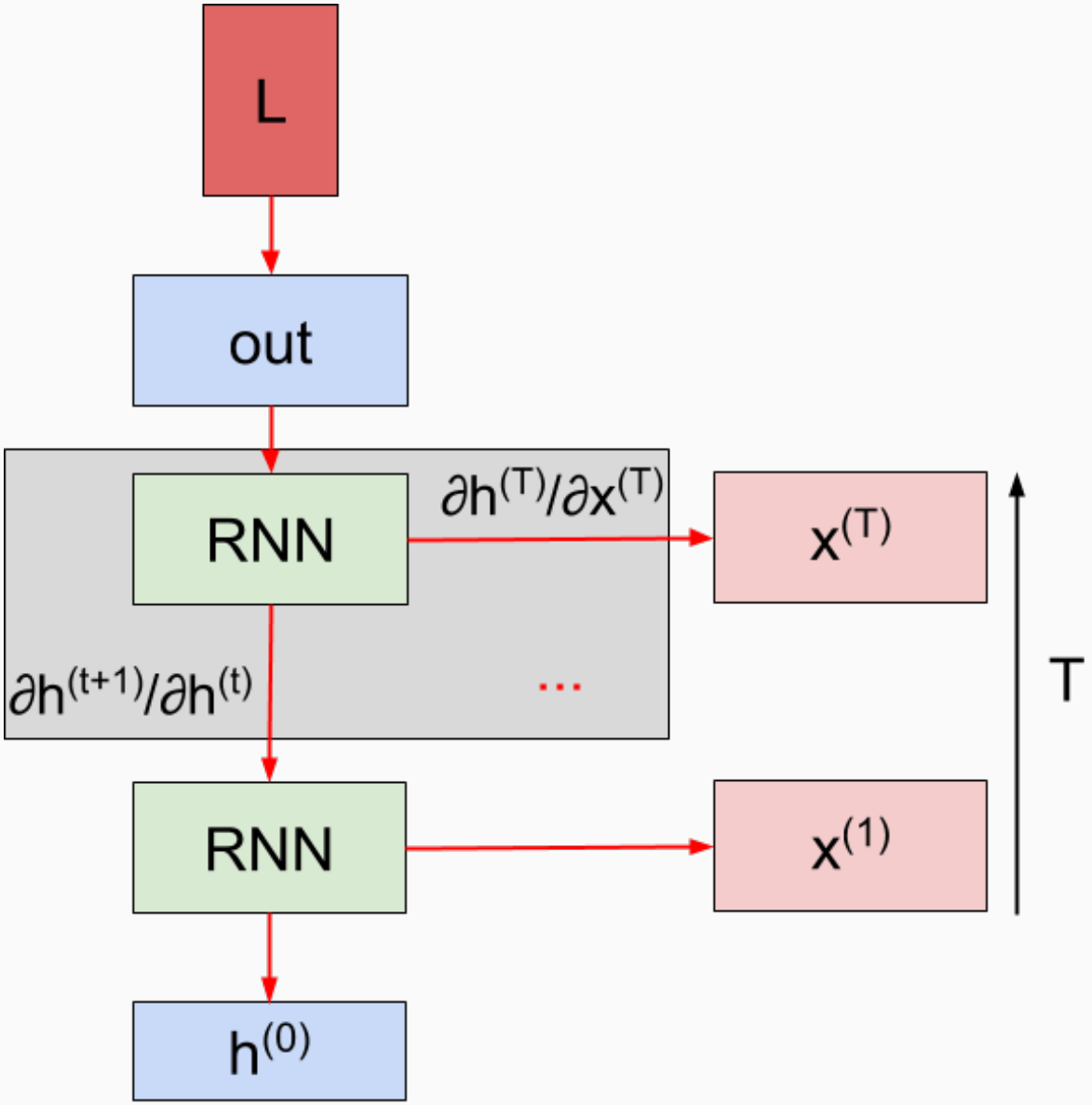
Za $t < T$ (preostale vremenske korake)

$$\frac{\partial \mathbb{L}}{\partial h^{(t)}} = \frac{\partial \mathbb{L}}{\partial h^{(T)}} \frac{\partial h^{(T)}}{\partial h^{(T-1)}} \cdots \frac{\partial h^{(t+1)}}{\partial h^{(t)}}$$

Općenito, do svakog stanja mogu doći gradijenti sljedeće dvije vrste:

1. gradijenti iz **odgovarajućih** predikcija (odozgo)
   (samo kod guste predikcije)
2. gradijenti iz **budućih** skrivenih stanja (zdesna)

# Povratno učenje: prolaz kroz ažuriranje stanja (2)



# Povratno učenje: prolaz kroz ažuriranje stanja (3)

Jednadžba ažuriranje stanja (podsjetnik):

$$h^{(t)} = tanh(\underbrace{W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h}_{a^{(t)}})$$

Treba nam derivacija hiperbolnog tangensa:

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

$$\frac{dtanh(x)}{dx} = 1 - tanh^2(x)$$

Hiperbolni tangens je sigmoida čiji izlaz je *rastegnut* preko $[-1, 1]$

$$tanh(x) = 2\sigma(2x) - 1$$

# Povratno učenje: prolaz kroz ažuriranje stanja (4)

Jednadžba ažuriranja stanja (podsjetnik):

$$h^{(t)} = tanh(\underbrace{W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h}_{a^{(t)}})$$

Gradijent s obzirom na predaktivaciju $a^{(t)}$:

$$\frac{\partial \mathbb{L}}{\partial a^{(t)}} = \frac{\partial \mathbb{L}}{\partial h^{(t)}} \frac{\partial h^{(t)}}{\partial a^{(t)}} = \frac{\partial \mathbb{L}}{\partial h^{(t)}} \frac{\partial tanh}{\partial a^{(t)}} = \ldots = \frac{\partial \mathbb{L}}{\partial h^{(t)}} \odot (1 - {h^{(t)}}^2)$$

Gradijenti s obzirom na parametre povratne ćelije ($W_{hh}$, $W_{xh}$, $b_h$):

$$\frac{\partial \mathbb{L}}{\partial W_{hh}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} \frac{\partial a^{(t)}}{\partial W_{hh}} = \ldots = \left(\frac{\partial \mathbb{L}}{\partial a^{(t)}}\right)^\top \left(h^{(t-1)}\right)^\top$$

$$\frac{\partial \mathbb{L}}{\partial W_{xh}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} \frac{\partial a^{(t)}}{\partial W_{xh}} = \ldots = \left(\frac{\partial \mathbb{L}}{\partial a^{(t)}}\right)^\top \left(x^{(t)}\right)^\top$$

$$\frac{\partial \mathbb{L}}{\partial b_h} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} \frac{\partial a^{(t)}}{\partial b_h} = \frac{\partial \mathbb{L}}{\partial a^{(t)}}$$

# povratno učenje: gradijenti s obzirom na ulaze

Jednadžba ažuriranja stanja (podsjetnik):

$$h^{(t)} = tanh(\underbrace{W_{hh}h^{(t-1)} + W_{xh}x^{(t)} + b_h}_{a^{(t)}})$$

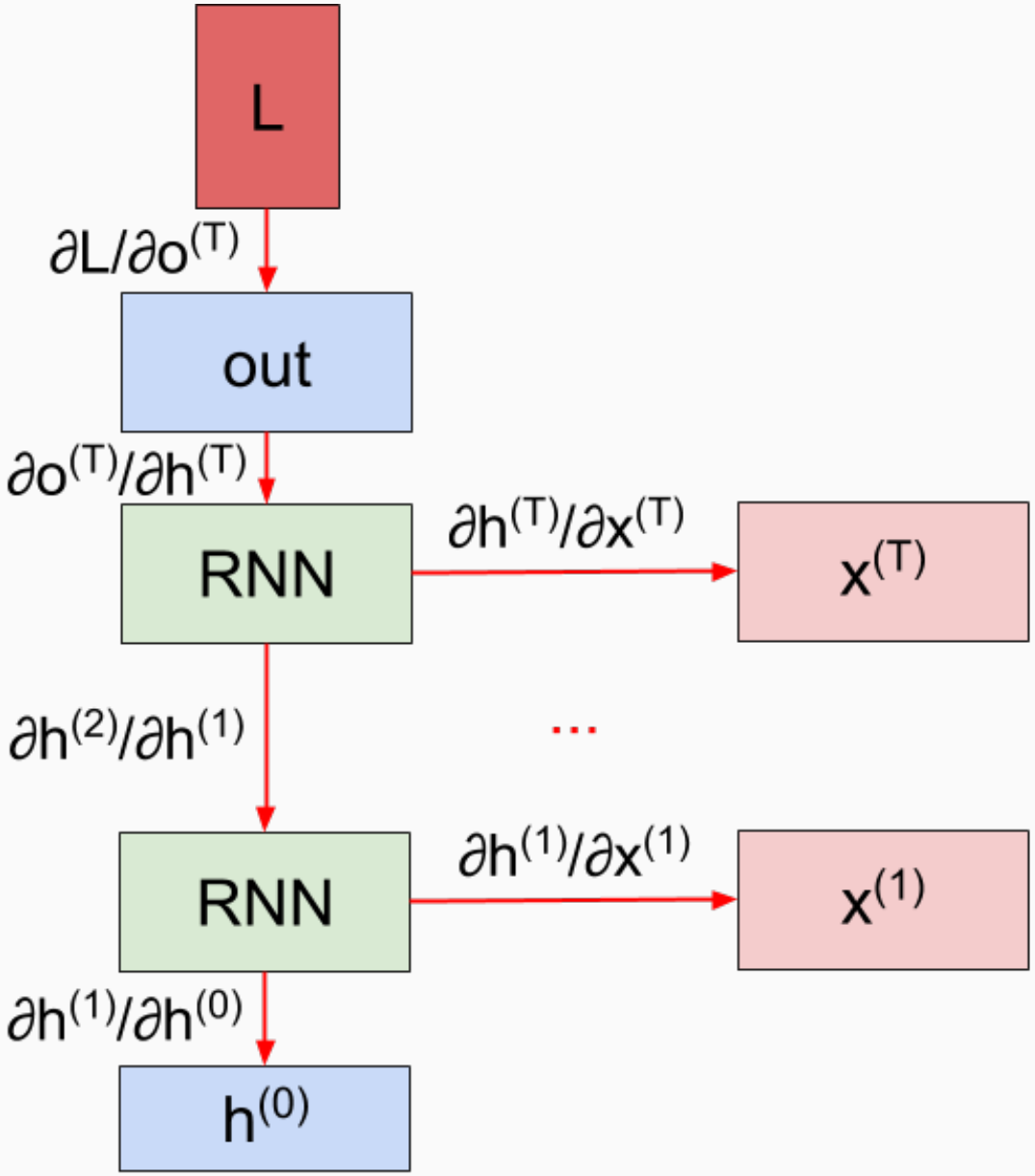
1. gradijent s obzirom na prethodno stanje:

$$\frac{\partial \mathbb{L}}{\partial h^{(t-1)}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} \frac{\partial a^{(t)}}{\partial h^{(t-1)}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} W_{hh}$$

2. gradijenti s obzirom na ulaz (razmislite zašto!):

$$\frac{\partial \mathbb{L}}{\partial x^{(t)}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} \frac{\partial a^{(t)}}{\partial x^{(t)}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} W_{xh}$$

# povratno učenje: klasifikacija slijeda, sažetak



# povratno učenje: klasifikacija slijeda, sažetak (2)

Gradijenti gubitka dolaze do parametara ažuriranja stanja kroz svako skriveno stanje:

$$\frac{\partial \mathbb{L}}{\partial W_{hh}} = \frac{\partial \mathbb{L}}{\partial a^{(t)}} \frac{\partial a^{(t)}}{\partial W_{hh}} = \underbrace{\left(\frac{\partial \mathbb{L}}{\partial a^{(t)}}\right)^\top \left(h^{(t-1)}\right)^\top}_{\forall t \in \{1,2,\ldots,T\}}$$

Ti gradijenti **akumuliraju** se kroz vrijeme (jednako za $W_{xh}$!):

$$\frac{\partial \mathbb{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \frac{\partial \mathbb{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \left(\frac{\partial \mathbb{L}}{\partial a^{(t)}}\right)^\top \left(h^{(t-1)}\right)^\top$$

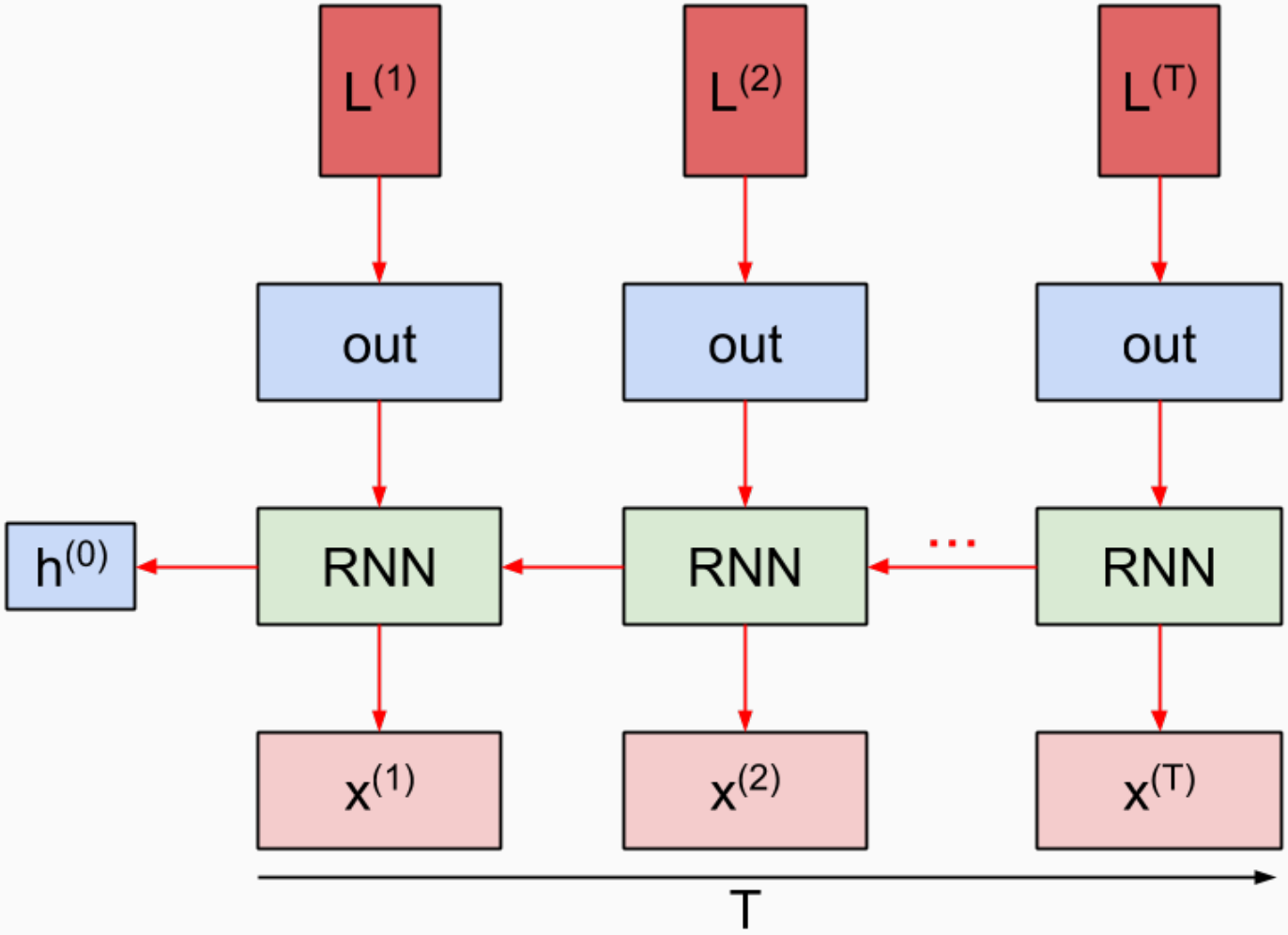
Optimizacijski korak koristi akumulirani gradijent.

Motivacija za računanje gradijenata s obzirom na ulaze ($x^{(t)}$):

1. učenje (ugađanje) ugrađivanja riječi
2. ulaz može odgovarati izlazu drugog povratnog modula

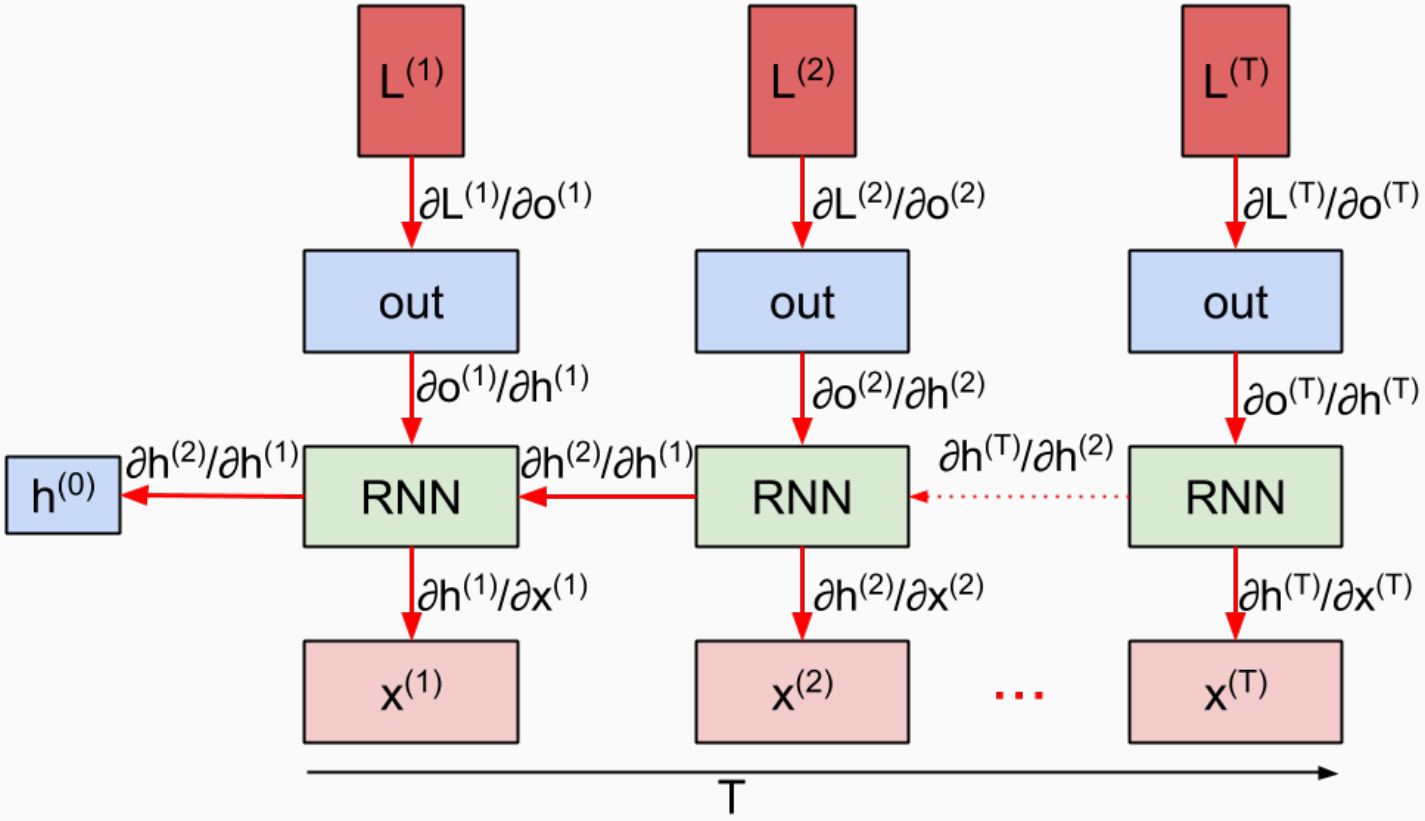
# Povratno učenje: gusta predikcija - gubitak

Gusta predikcija implicira gubitak u **svakom** vremenskom koraku



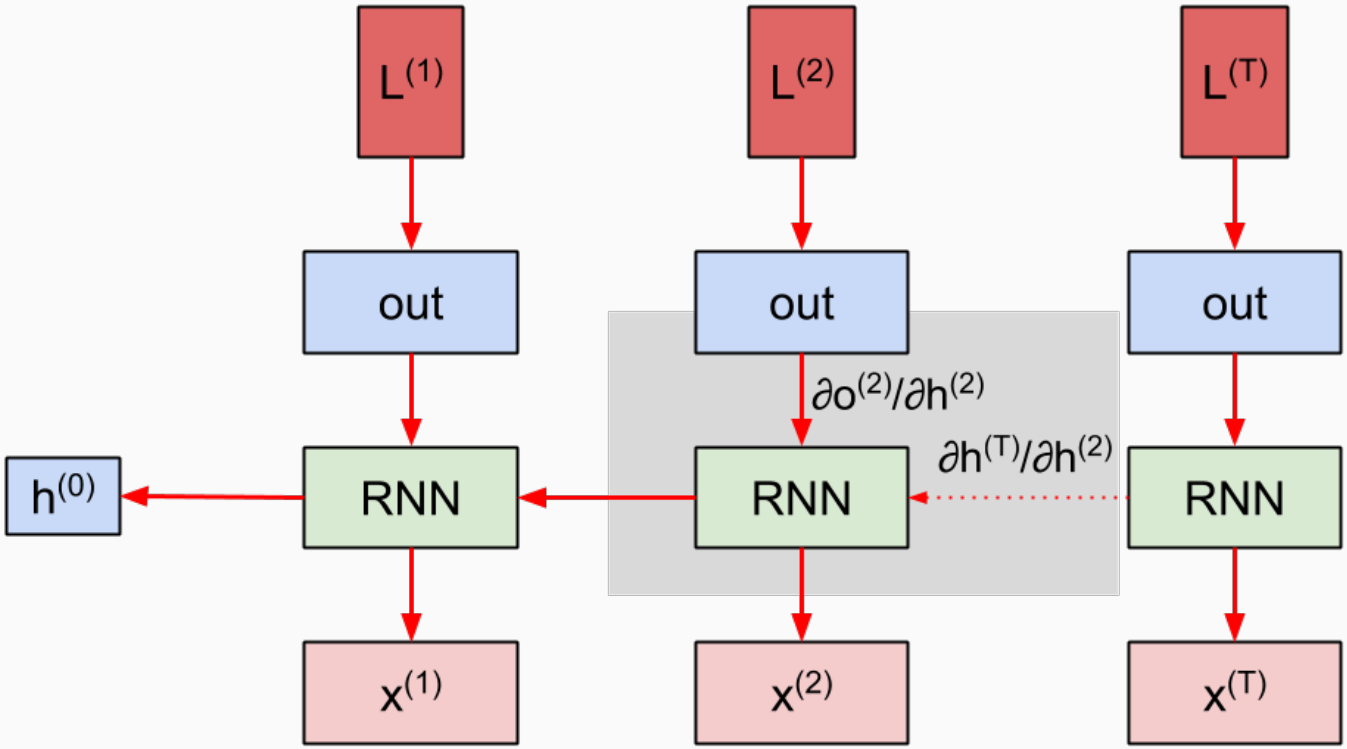
# Povratno učenje: gusta predikcija - propagiranje gradijenta

U gustom slučaju moramo akumulirati doprinose članova gubitka iz svih budućih koraka:



# Povratno učenje: gusta predikcija - gradijenti

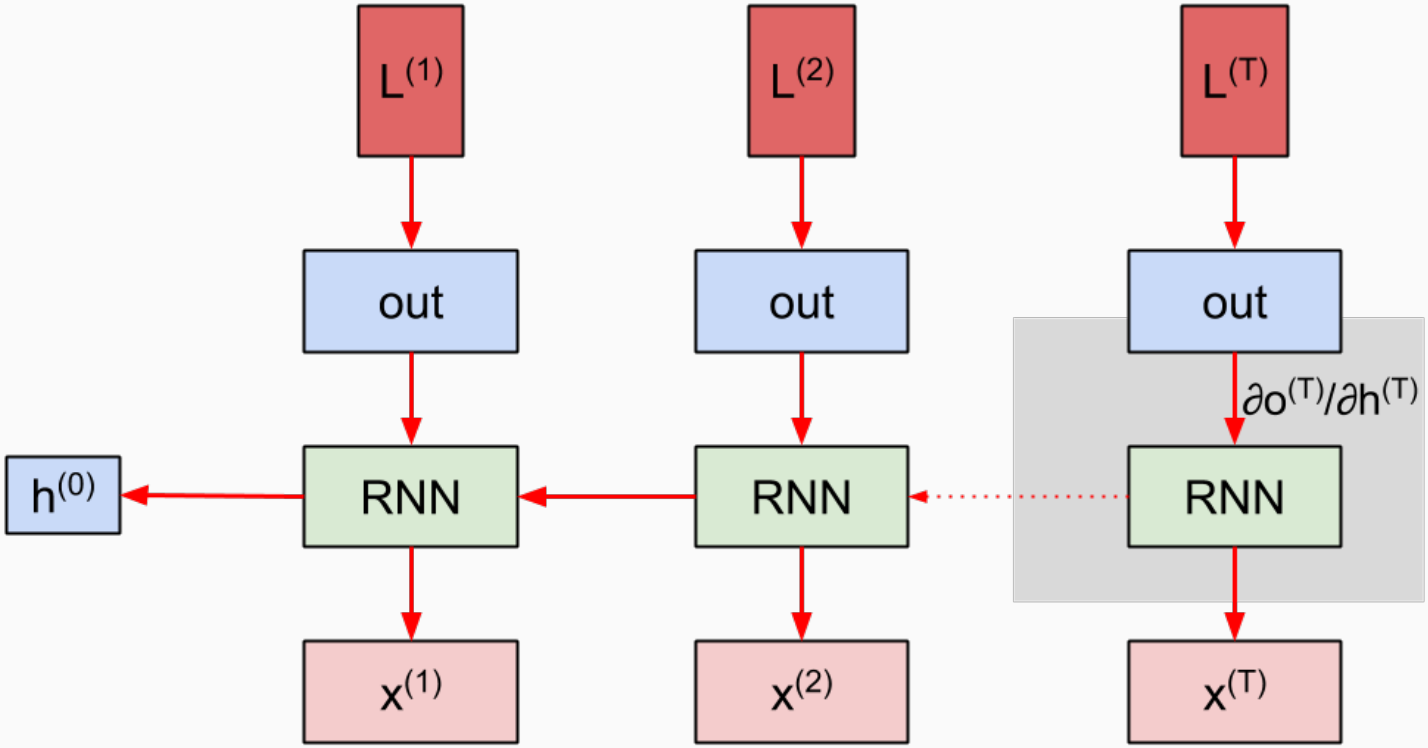
Na ažuriranje stanja utječu tekući gubitak (vertikalna ovisnost) i svi budući članovi gubitka (vodoravna ovisnost):



# Povratno učenje: gusta predikcija - gradijenti (2)

Na ažuriranje stanja utječu tekući gubitak (vertikalna ovisnost) i svi budući članovi gubitka (vodoravna ovisnost):

- … osim u posljednjem vremenskom koraku kad ne trebamo nikakav doprinos iz budućnosti



# Povratno učenje: gusta predikcija - gradijenti (3)

Gubitak odgovara **zbroju** vremenskih komponenata $\mathbb{L} = \sum_t \mathbb{L}^{(t)}$

- ako gradijente računamo *ručno* trebamo posebno paziti na $\frac{\partial \mathbb{L}}{\partial h^{(t)}}$

Posljednji vremenski korak ($t = T$) isti kao i u klasifikacijskom slučaju:

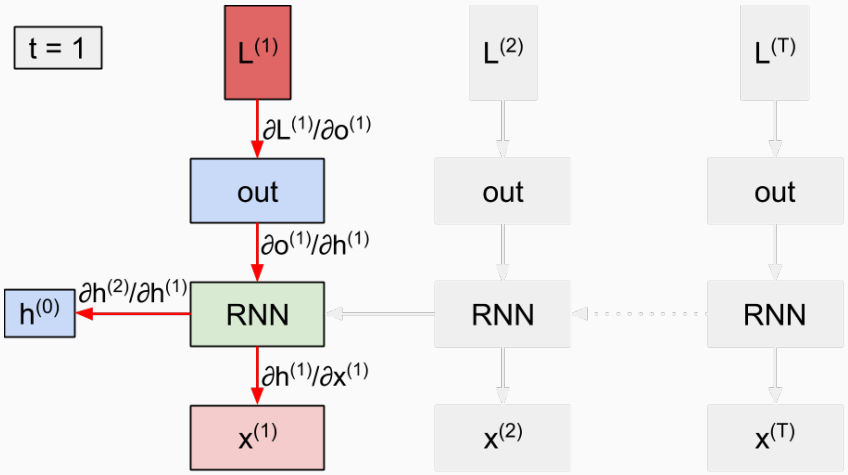
$$\frac{\partial \mathbb{L}}{\partial h^{(t)}} = \frac{\partial \mathbb{L}^{(t)}}{\partial h^{(t)}} = \frac{\partial \mathbb{L}^{(t)}}{\partial o^{(t)}} \frac{\partial o^{(t)}}{\partial h^{(t)}} = \frac{\partial \mathbb{L}^{(t)}}{\partial o^{(t)}} W_{hy}$$

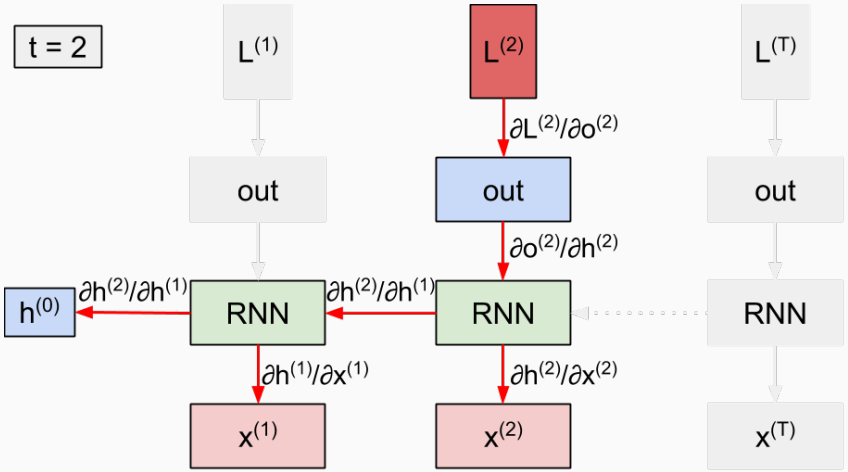
Preostali vremenski koraci ($t < T$) zahtijevaju posebnu pažnju:

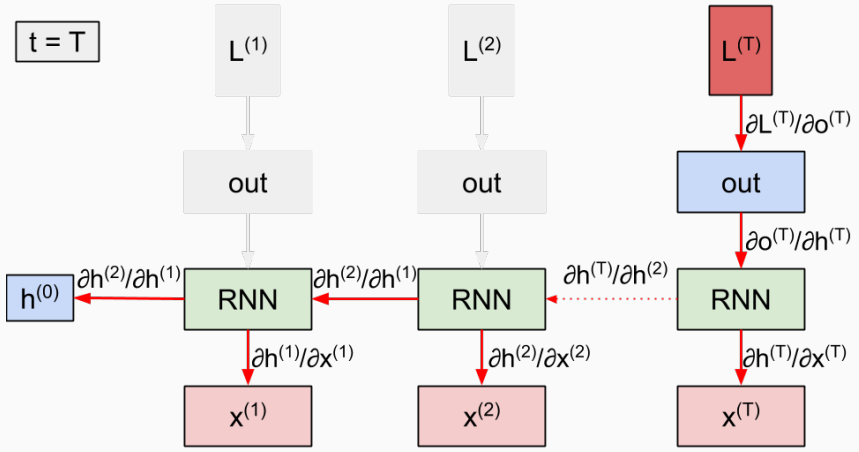
$$\frac{\partial \mathbb{L}}{\partial h^{(t)}} = \underbrace{\frac{\partial \mathbb{L}^{(t)}}{\partial h^{(t)}}}_{\text{tekući gubitak}} + \underbrace{\frac{\partial \mathbb{L}^{(t^*>t)}}{\partial h^{(t)}}}_{\text{budući gubitci}}$$

$$\frac{\partial \mathbb{L}^{(t^*>t)}}{\partial h^{(t)}} = \sum_{t^*>t}^{\top} \frac{\partial \mathbb{L}^{(t^*)}}{\partial h^{(t)}} = \sum_{t^*>t}^{\top} \frac{\partial \mathbb{L}^{(t^*)}}{\partial h^{(t^*)}} \cdots \frac{\partial h^{(t+1)}}{\partial h^{(t)}}$$

# Povratno učenje: gusta predikcija - intuicija




Svaka povratna ćelija prima gradijente odozgo (trenutak t) i zdesna (budući gubitak)

U praksi možemo provesti:

- ili *T* unatražnih prolaza (s `retain_graph=True`)
- ili 1 unatražni prolaz kroz ukupni gubitak

# Povratno učenje: gusta predikcija (sažetak)

Računanje gradijenata povratnog modela često nazivamo širenjem unatrag kroz vrijeme (BPTT):

- širenje unatrag kroz vrijeme odgovara standardnom backpropu kroz razmotani model
- gustu predikciju možemo interpretirati kao združeno optimizirani višezadaćni klasifikacijski problem
- povratnu ćeliju možemo promatrati kao sloj običnog potpuno povezanog modela

Zbog dijeljenih parametara i akumulacije gradijenata, učenje može biti **nestabilno**

- česte pojave nestajućih i eksplodirajućih gradijenata
- prikazana osnovna formulacija rijetko se koristi u praksi

# Praktični savjeti

Dimenzionalnost latentnog stanja $h$:

- što veća - to bolje, ostali hiperparametri mogu biti važniji
- dubina modela također može biti važnija (sljedeće predavanje)

Duljina slijeda $T$:

- obični povratni modeli **kratkoročno** pamte ($T \approx 20$ za tekst)
- često podrezujemo ulazne slijedove na zadanu duljinu
- prag ovisi o podatcima i zadatku

Dobar početak: Adam, podrezivanje gradijenta (čak i za LSTM)

Alternative povratnim modelima **postoje**:

- SVM za jednostavnu klasifikaciju teksta:
  `https://github.com/mesnilgr/nbsvm`
- **pažnja** može biti sve što nam treba
- zvuk, govor: (dilatirane) konvolucije, pažnja

Pitanja?

# Literatura

Literatura (pored knjige):

1. Peter's notes: Implementing a NN / RNN from scratch `http://peterroelants.github.io/`
2. Andrej Karpathy: Unreasonable Effectiveness of Recurrent Neural Networks `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`
3. Cristopher Olah: Understanding LSTM's `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`
4. CS224d: Deep Learning for Natural Language Processing `http://cs224d.stanford.edu/syllabus.html`