Obrada prirodnih jezika

Klasifikacija teksta korišćenjem ML-a

Tamara Stanković, Data Scientist 2

Microsoft Development Center Serbia

Email: Tamara.Stankovic@microsoft.com

Obrada prirodnih jezika

Natural Language Processing (*NLP*)

Razvijanje metoda za rešavanje različitih problema povezanih sa prirodnim jezicima, na primer:

- Pronalaženje podataka (*information extraction*)
- Automatsko prepoznavanja govora (*automatic speech recognition*)
- Mašinsko prevođenje (*machine translation*)
- Analiza teksta na osnovu sentimenta (*sentiment analysis*)
- Odgovaranje na pitanja (*question answering*)
- Sumarizacija teksta (*summarization*)

Agenda

- Definicija problema
- Kako predstaviti podatke?
  - *One-hot* vektor vs. *word embedding* vektor
- Kako izabrati model?
  - ''Standardna'' rešenja
  - Rekurente neuronske mreže
  - *Transformer*

Definicija problema

- Klasifikacija *tweet*-ova na pozitivne, neutralne i negativne, na osnovu njihovog sentimenta (*sentiment analysis*).

Otvoreno pitanje #1:
Kako prestaviti podatke?

Otvoreno pitanje #2:
Koji ML model iskoristiti za rešavanje problema?

Kako predstaviti podatke?

- Rečenica/paragraf/*tweet*/itd. ili bilo koji drugi oblik teksta je **sekvenca reči**:

$$[w_1, w_2, \dots, w_k]$$

- Kako predstaviti reč u numeričkom obliku?

$$f(word\_str) = word\_num$$

	Today	**is**	**an**	**awesome**	**day**	**!**
Opcija #1	58	89	76	98	26	-1
Opcija #2	[0.3, 0.25]	[0.98, 0.8]	[0.1, 0.05]	[0.9, 0.8]	[0.7, 0.6]	[0, 0]
Opcija #3	[0, 0, 0, 1]	[0, 1, 0, 0]	[1, 0, 0, 0]	[0, 0, 0, 0]	[0, 0, 1, 0]	[0, 0, 0, 0]
			...			
Opcija #N	...	...	...	...	...	...

Numerička reprezentacija reči

Rečnik (*vocabulary, dictionary*) – skup svih reči koje se mogu pojavljivati u tekstu

Svaka reč može se predstaviti:

- *One-hot* vektorom
  - Vektor koji sadrži sve nule, osim na poziciji koja odgovara toj reči u rečniku
- *Word embedding*-om
  - Transformisana reprezentacija reči u nekom vektorskom prostoru

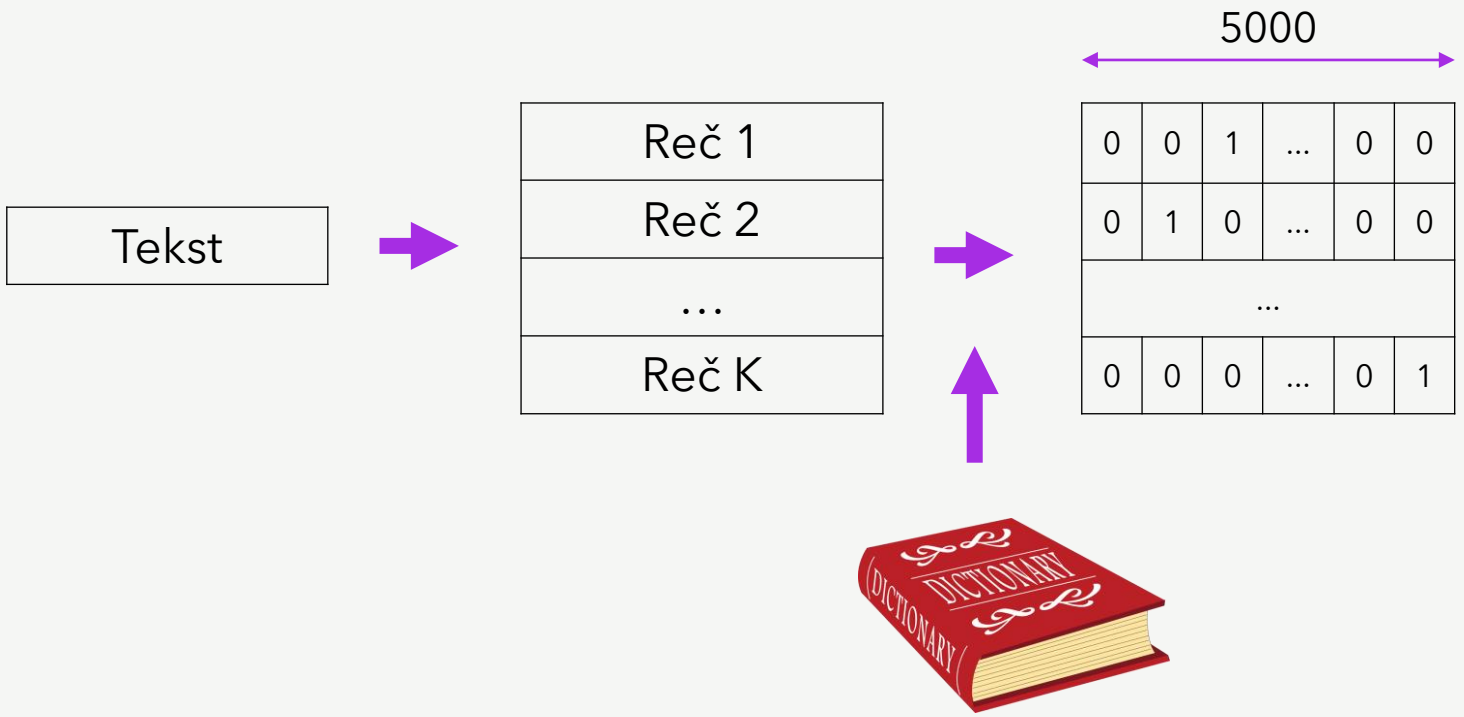
Numerička reprezentacija reči

Rečnik (*vocabulary, dictionary*) – skup svih reči koje se mogu pojavljivati u tekstu

Svaka reč može se predstaviti:

- *One-hot* vektorom
  - Vektor koji sadrži sve nule, osim na poziciji koja odgovara toj reči u rečniku
- *Word embedding*-om
  - Transformisana reprezentacija reči u nekom vektorskom prostoru

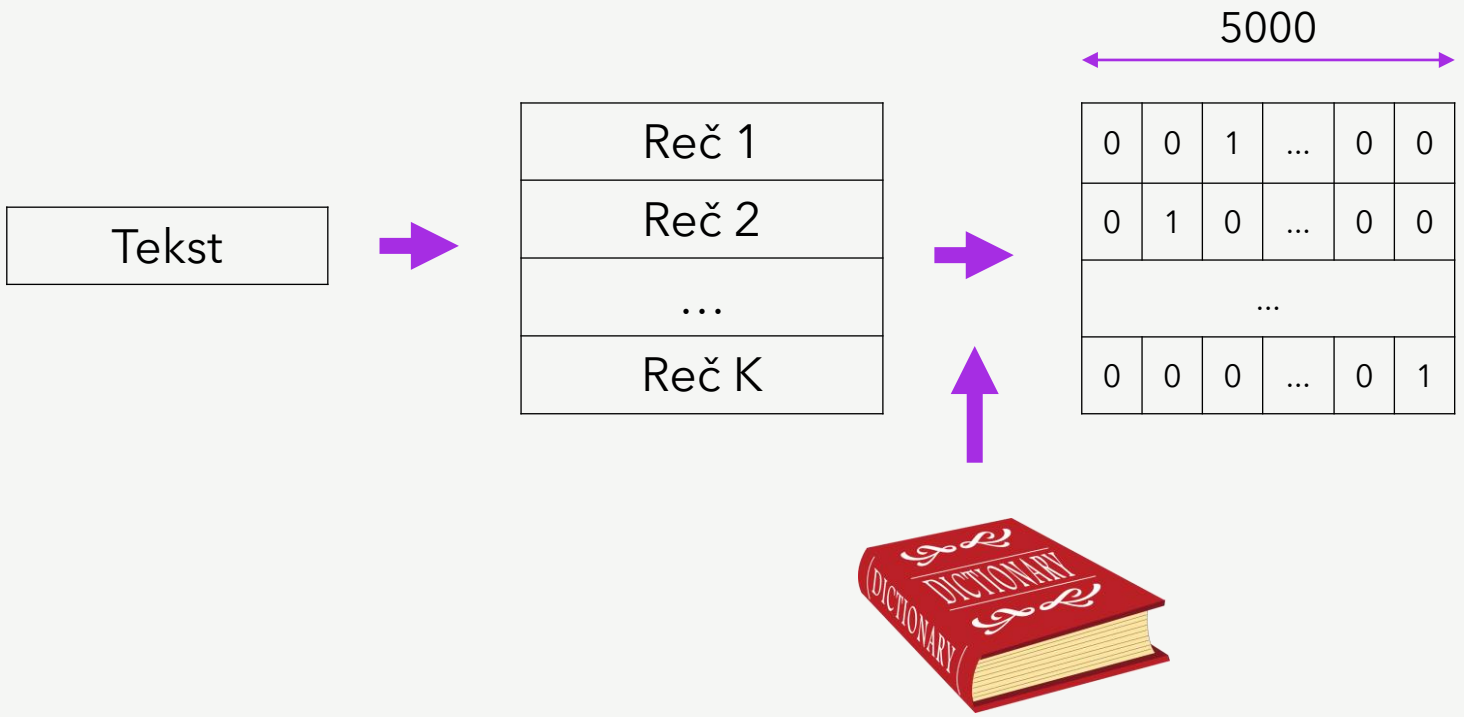
One-hot reprezentacija reči



Rečnik sa 5000 najfrekventnijih reči

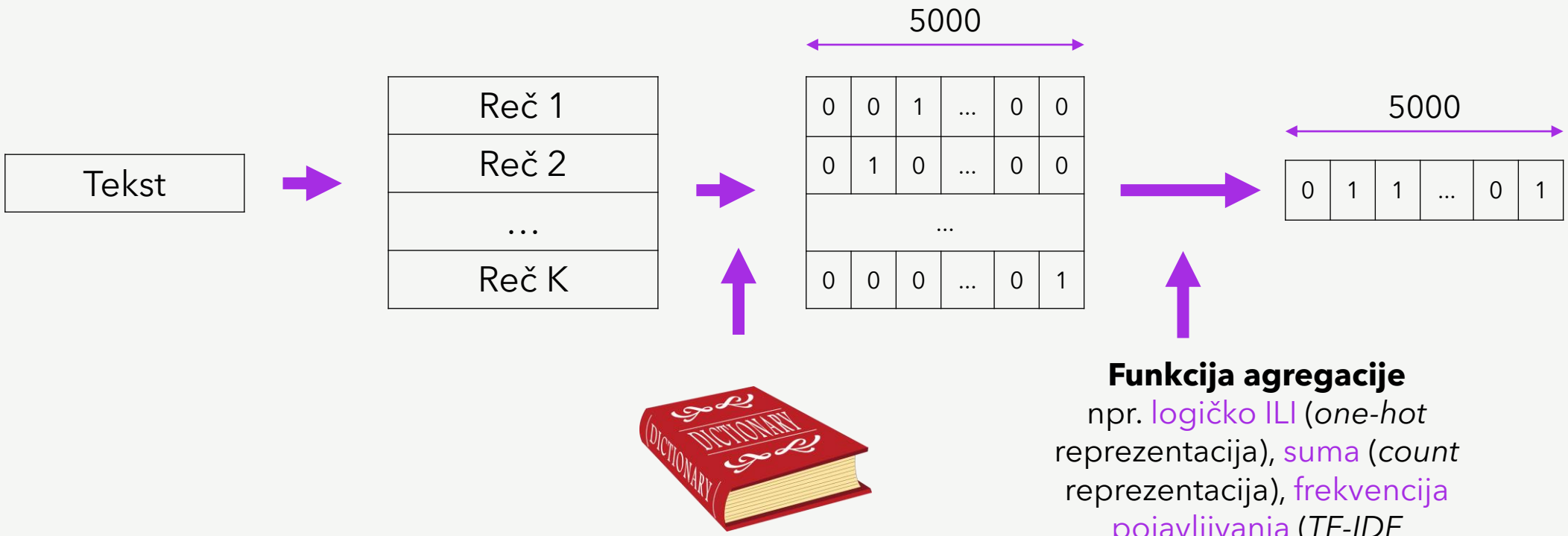
Svaka reč u teksu mapira se u *one-hot* vektor, čija dužina odgovara broju reči u rečniku. Ovaj vektor sadrži sve nule, osim na mestu na koje odgovara indeksu te reči u rečniku.
Ukoliko se reč ne nalazi u rečniku, mapira se u „nepoznatu reč" (*unknown token*), specijalnu reč izdvojenu za ovakve slučajeve.

One-hot reprezentacija reči



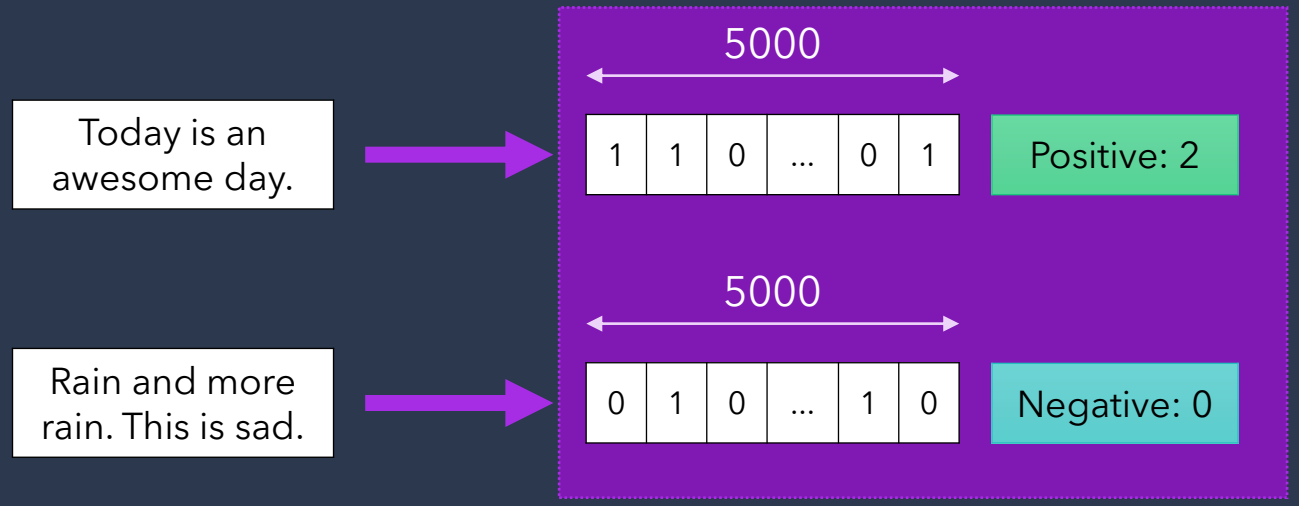
Rečnik sa 5000 najfrekventnijih reči

One-hot reprezentacija teksta

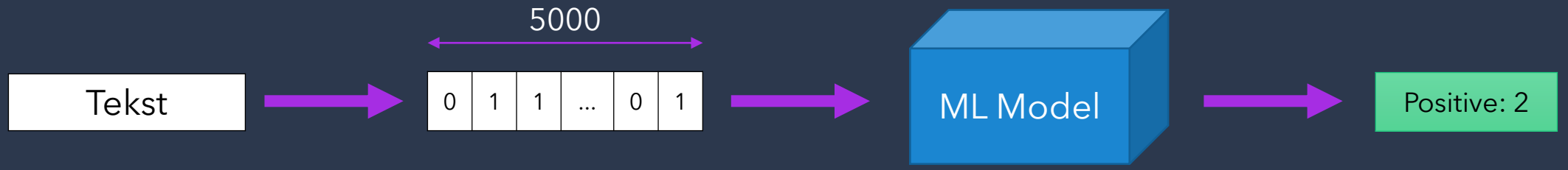


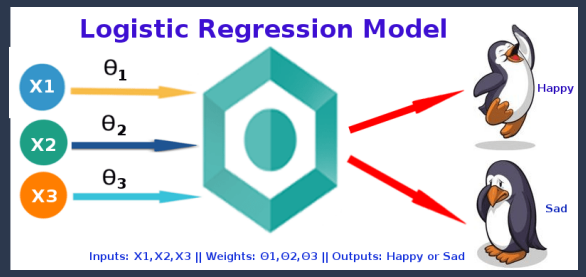
Kako predstaviti podatke? – Rešenje #1

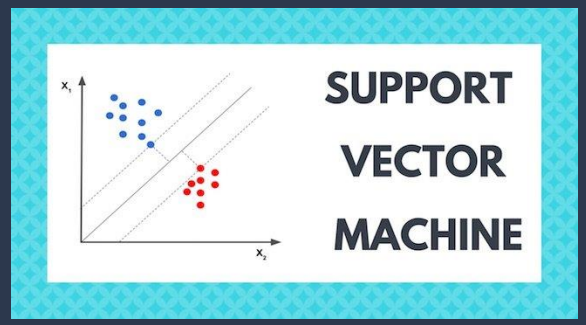
- Za neki tekst definisali smo funkciju koja mapira taj tekst u vektor brojeva, dužine 5000
  - Za svaki ulazni tekst može se dobiti njegov *feature vector* (X)
  - Kako već imamo labelu za svaki tekst (Y), možemo konstruisati parove (X, Y) koje koristimo za treniranje ML modela

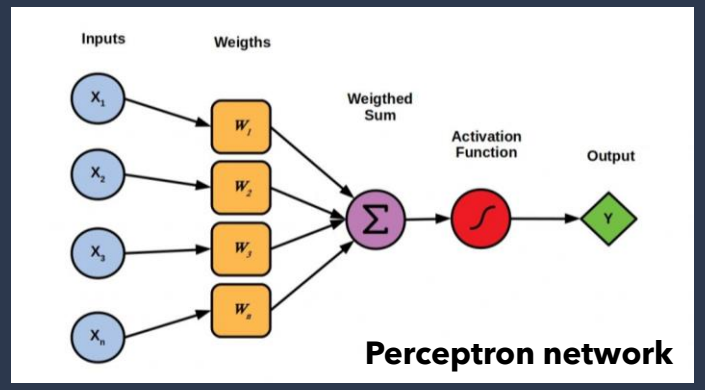


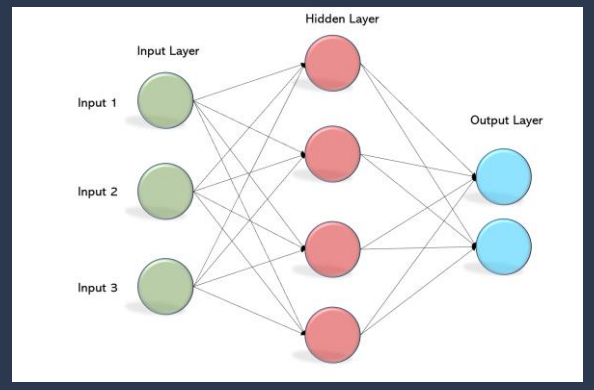
Kako izabrati model? – Rešenje #1











MLP network

Rešili smo problem, ali...

Numerička reprezentacija reči

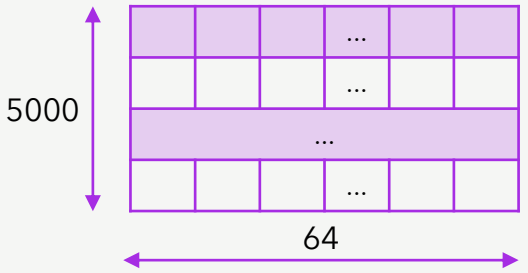
Rečnik (*vocabulary, dictionary*) – skup svih reči koje se mogu pojavljivati u tekstu

Svaka reč može se predstaviti:

- *One-hot* vektorom
  - Vektor koji sadrži sve nule, osim na poziciji koja odgovara toj reči u rečniku
- *Word embedding*-om
  - Transformisana reprezentacija reči u nekom vektorskom prostoru

Word Embedding reprezentacija

Matrica *Word Embedding*-a



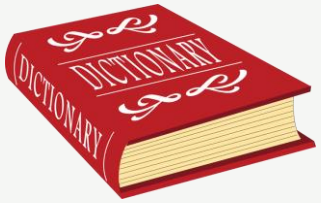
Tekst

Reč 1
Reč 2
…
Reč K

Indeks reči 1
Indeks reči 2
…
Indeks reči K

0.25	0.29	0.84	…	0.77	0.32
0.31	0.18	0.78	…	0.95	0.71
…					
0.67	0.01	0.98	…	0.45	1.61

64



Rečnik sa 5000 najfrekventnijih reči

Matrica *Word Embedding*-a

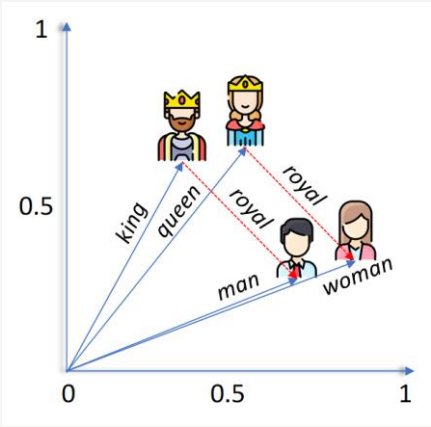
Matrica *Word Embedding*-a

5000

-1.24	-0.36	0.57	...	0.60	-0.18
1.27	-0.96	-0.61	...	0.67	0.35
...					
0.41	-0.21	1.38	...	0.55	1.14

64

Glavna odlika dobrih *word embedding*-a jeste da „hvataju" sintaktičke i semantičke veze između reči, kao i pravila pod kojima se one koriste.



- Kako dobiti matricu *Word Embedding*-a?
  1. Izračunati je za problem koji se rešava
     - Trenirati model koji uči reprezentaciju reči pomoću velikog nelabelovanog skupa podataka, na nekom *supervised* zadatku – npr. predikcija sledeće reči u tekstu (*language modeling task*)
  2. Iskoristiti neku postojeću
     - Moguće je koristiti neku od dostupnih varijanti – originalni *Word2Vec*, *GloVe* sa Standford-a, Facebook-ov *FastText*, itd.
  3. Iskoristiti neku postojeću, ali je prilagoditi problemu
     - Inicijalizovati *Word Embedding* matricu pomoću neke od dostupnih varijatni, a zatim je tokom treninga ažurirati da se što bolje prilagodi problemu

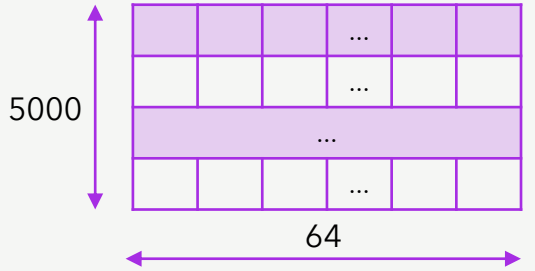
Word Embedding reprezentacija

Tekst

Reč 1
Reč 2
…
Reč K

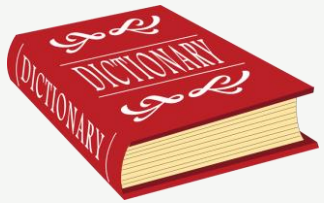
Indeks reči 1
Indeks reči 2
…
Indeks reči K

Matrica *Word Embedding*-a



0.25	0.29	0.84	…	0.77	0.32
0.31	0.18	0.78	…	0.95	0.71
…					
0.67	0.01	0.98	…	0.45	1.61

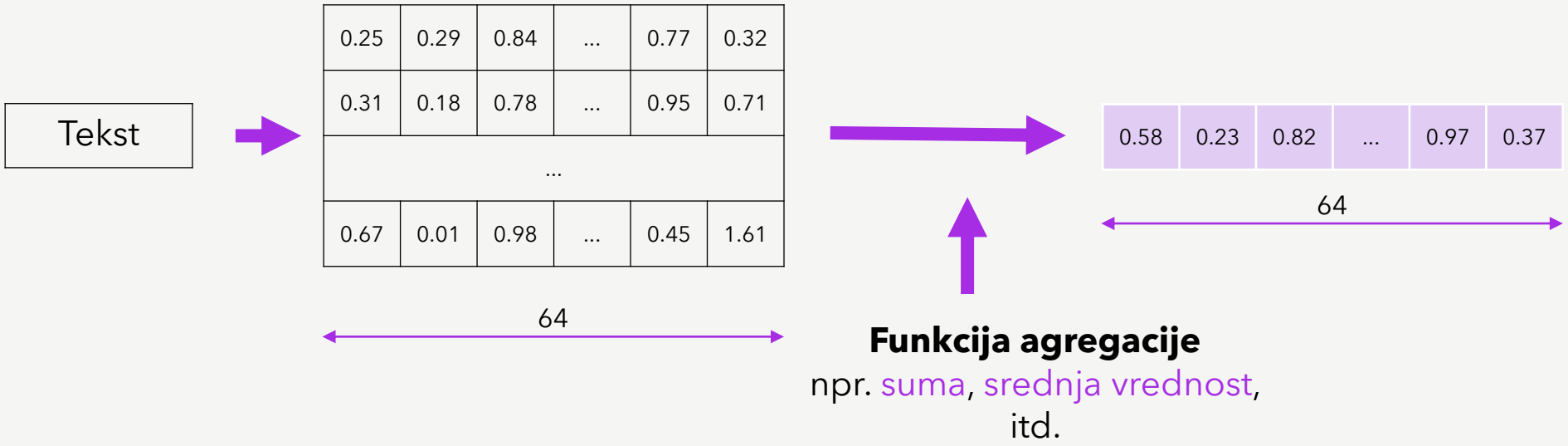
64



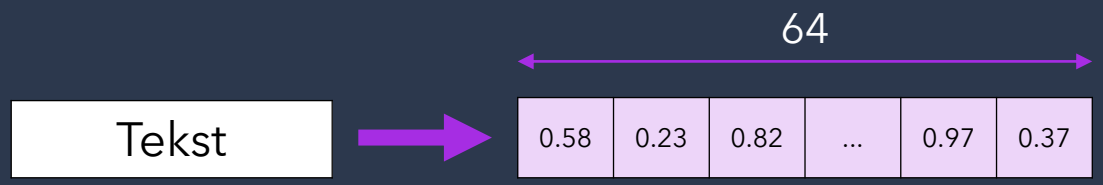
Rečnik sa 5000 najfrekventnijih reči

Word Embedding reprezentacija

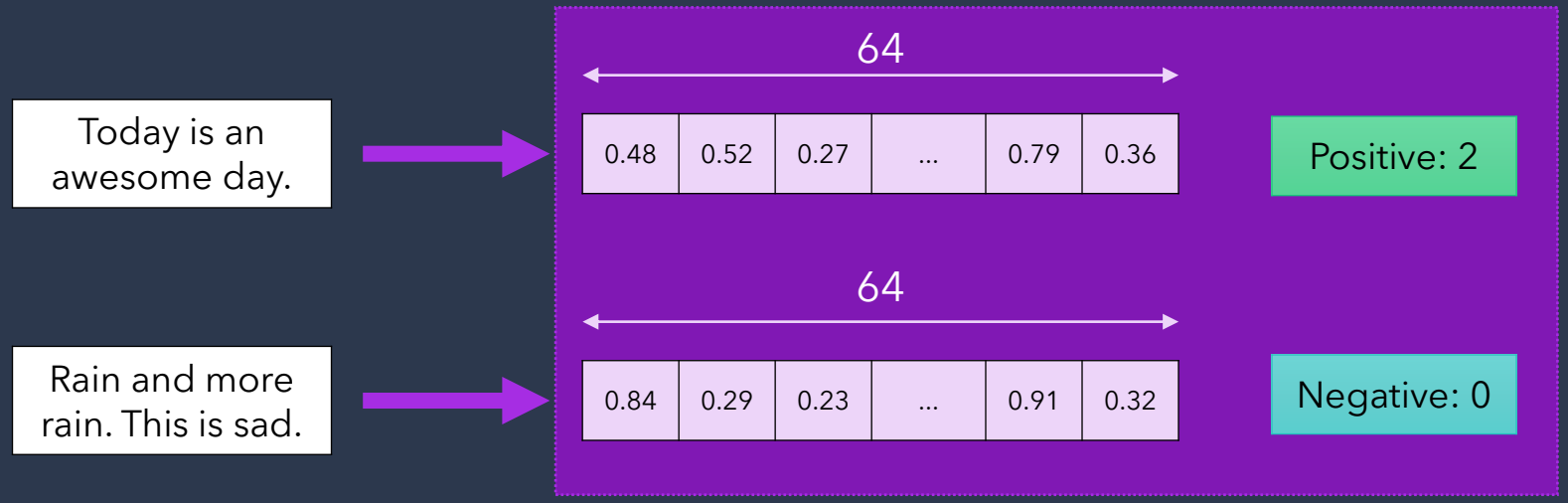
Opcija #1



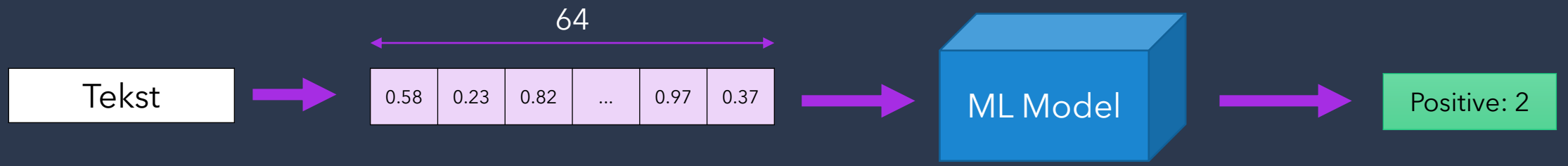
Kako predstaviti podatke? – Rešenje #2

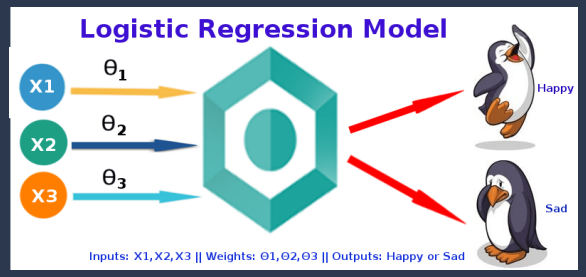


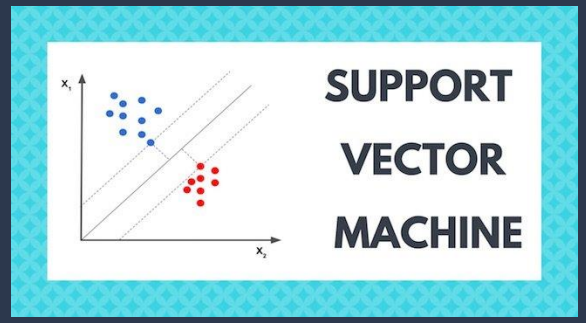
- Za neki tekst definisali smo funkciju koja mapira taj tekst u vektor brojeva, dužine 64
  - Za svaki ulazni tekst može se dobiti njegov *feature vector* (X)
  - Kako već imamo labelu za svaki tekst (Y), možemo konstruisati parove (X, Y) koje koristimo za treniranje ML modela

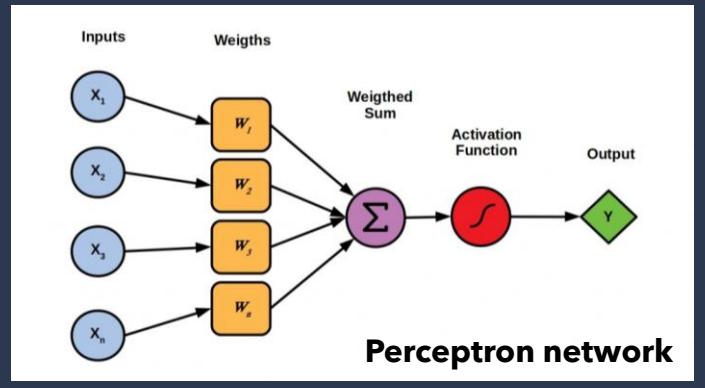


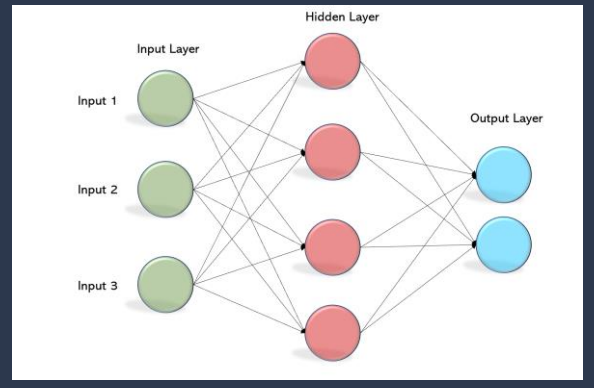
Kako izabrati model? – Rešenje #2











Ponovo smo rešili problem, ali...

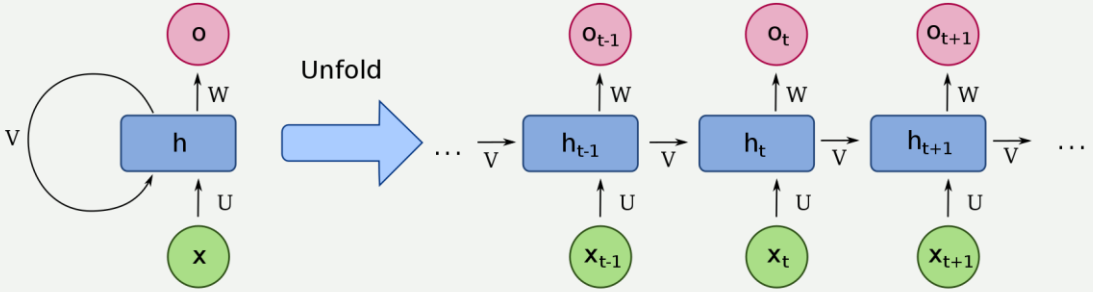
Rekurente Neuronske Mreže (RNN)

- Tip neuronskih mreža koji se često koristi za obradu podataka u sekvenci
- Informacije se propagiraju kroz mrežu, ali tako da jedan element koristi samo one informacije koje su pre njega u sekvenci
  - Trenutno „skriveno" stanje (*hidden state*) se računa na osnovu predhodnog „skrivenog" stanja i trenutnog ulaza (*input*)
  - Trenutni izlaz (*output*) se računa an osnovu trenutnog „skivenog" stanja
- U svim koracima, koriste se isti parametri za izračunavanje trenutnog „skrivenog" stanja i izlaza.

$$h_t = f_1(\mathrm{U} \times x + V \times h_{t-1})$$

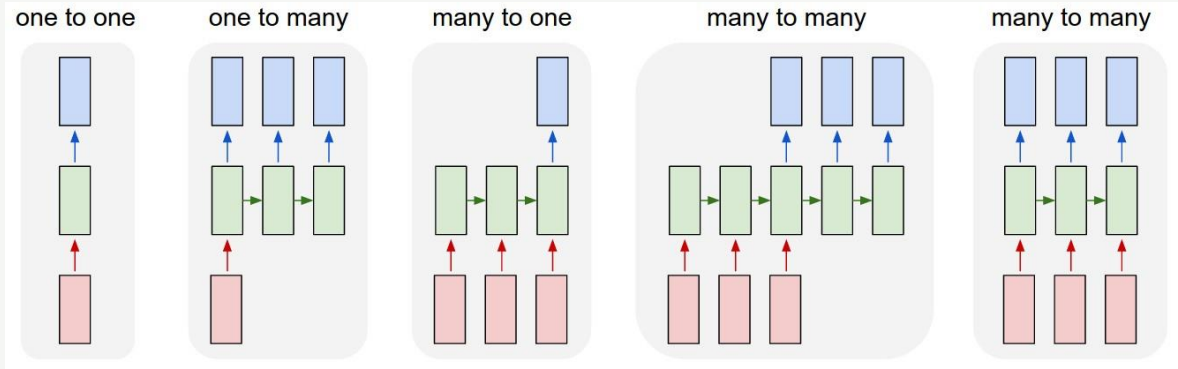
$$o_t = f_2(W \times h_t)$$

U – matrica parametara za ulazne podatke
V – matrica parametara za „skriveno" stanje
W – matrica parametara za računanje izlaza
f_1 - funkcija aktivacije, npr. $tanh$
f_2 - funkcija aktivacije, npr. $sigmoid$ ili $softmax$

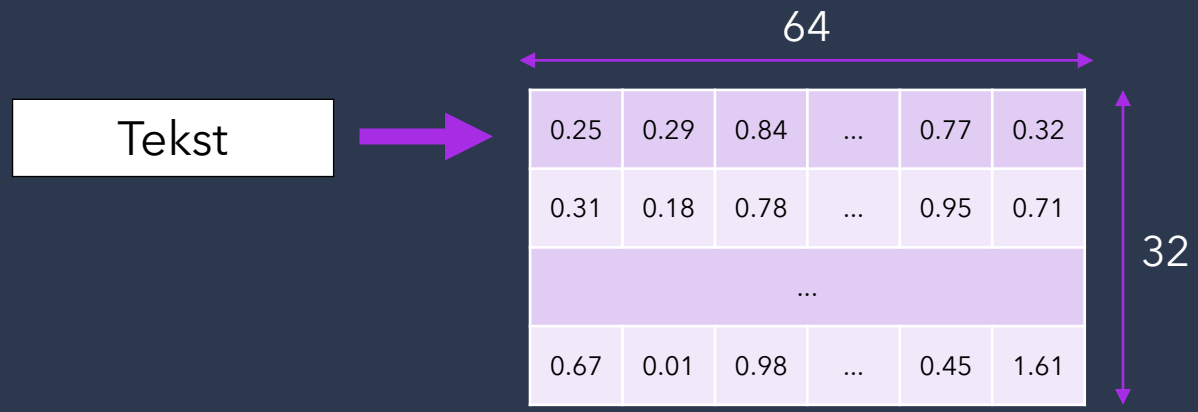


Tipovi *RNN*-ova

1. **Jedan-na-jedan** (*one to one*) – standardna neuronska mreža
2. **Jedan-na-više** (*one to many*) – npr. generisanje sekvence
3. **Više-na-jedan** (*many to one*) – npr. klasifikacija teksta na osnovu sentimenta
4. **Više-na-više** (*many to many*) sa različitim brojem ulaza i izlaza – encoder/decoder arhitektura – npr. mašinsko prevođenje
5. **Više-na-više** (*many to many*) sa istim brojem ulaza i izlaza – npr. prepoznavanje imenovanih entiteta



Kako predstaviti podatke? – Rešenje #3

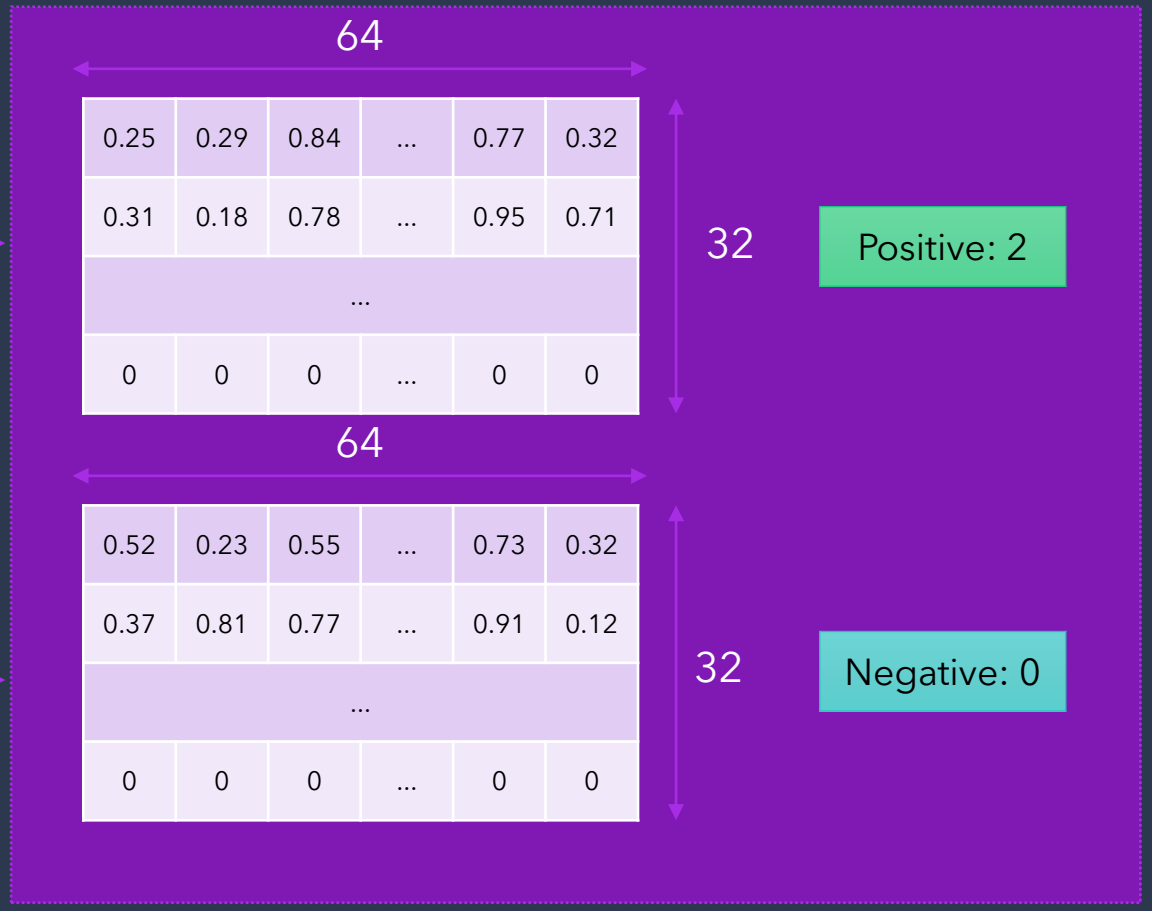


- Za neki tekst definisali smo funkciju koja mapira taj tekst u **matricu** brojeva, veličine 32 x 64.
  - 32 odgovara predefinisanoj maksimalnoj dužini ulazne sekvence:
    - Ako je sekvenca duža od ove predefinisane maksimalne dužine, uzimaju se prve 32 reči
    - Ako je sekvenca kraća od ove predefinisane maksimalne dužine, matrica se dopunjava *padding* vektorima, sa predefinisanim vrednostima (npr. nulama)
  - 64 odgovara dužini *word embedding* vektora jedne reči
- Za svaki ulazni tekst može se dobiti njegov *feature vector* (X)
- Kako već imamo labelu za svaki tekst (Y), možemo konstruisati parove (X, Y) koje koristimo za treniranje ML modela

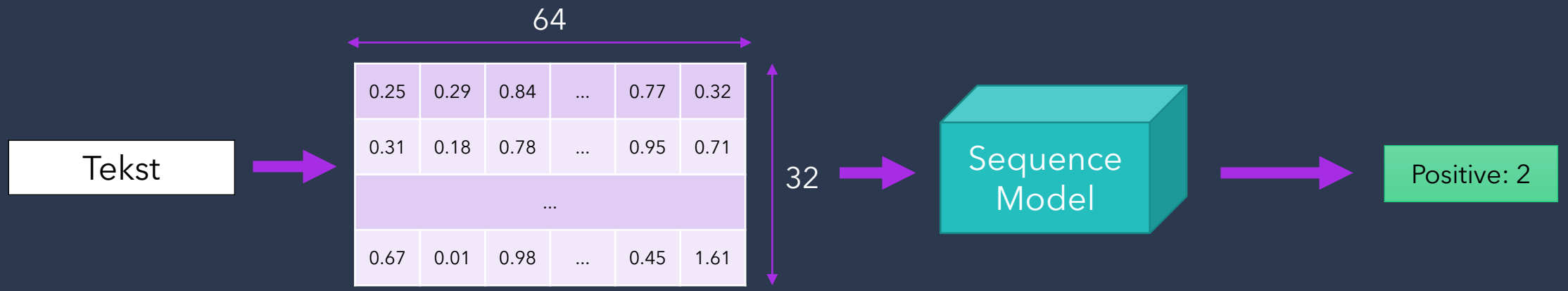
Kako predstaviti podatke? – Rešenje #3

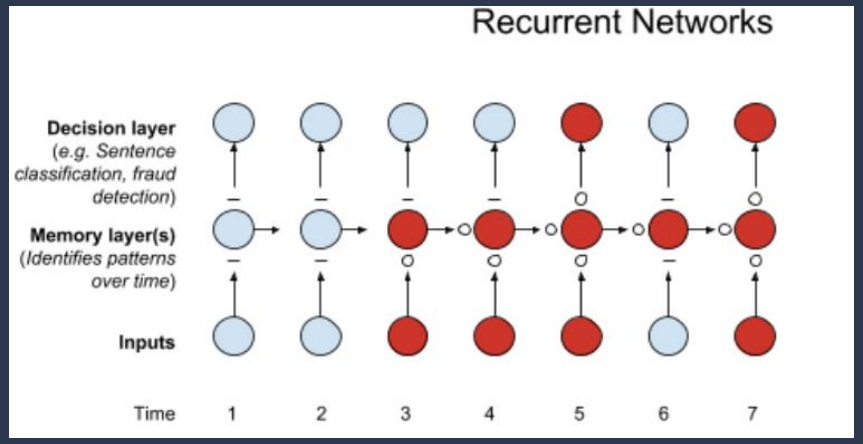
Today is an awesome day.

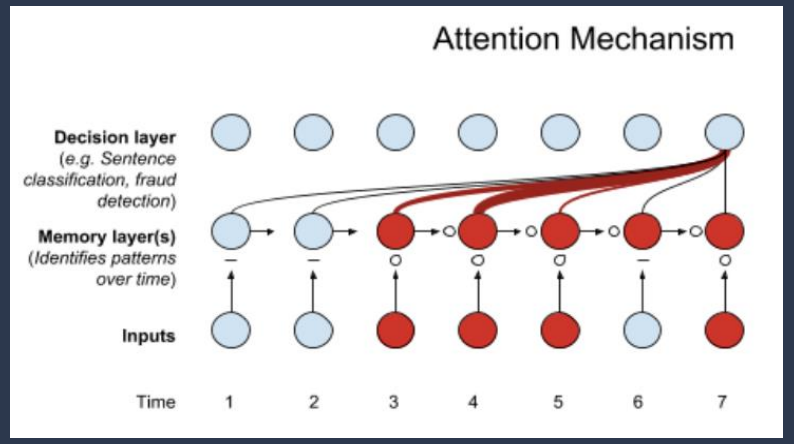
Rain and more rain. This is sad.



Kako izabrati model? – Rešenje #3

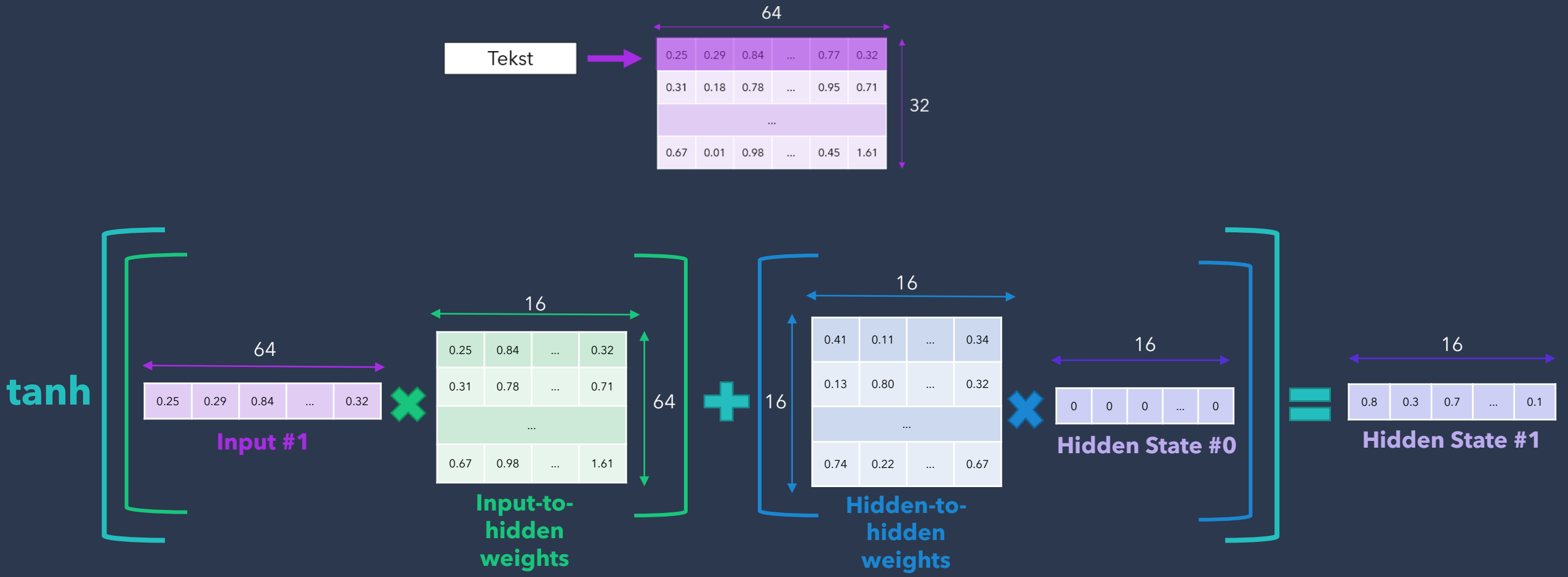






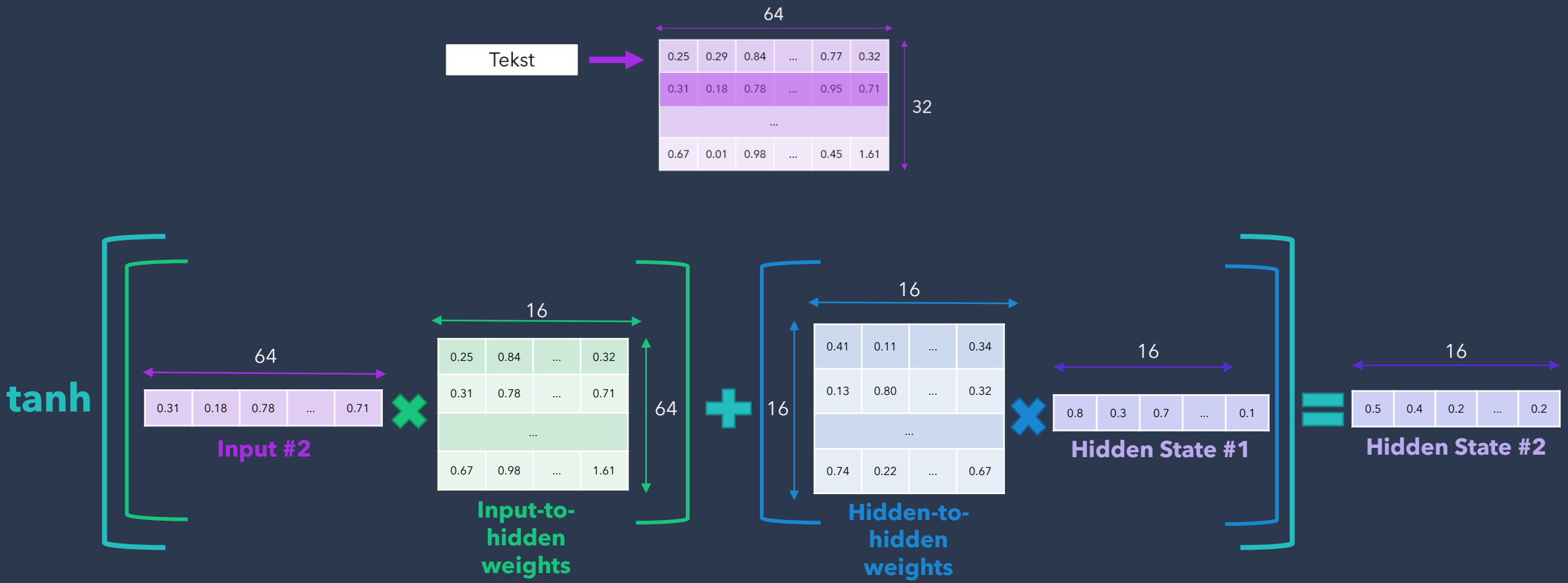
Kako izabrati model? – Rešenje #3 – RNN

Korak 1



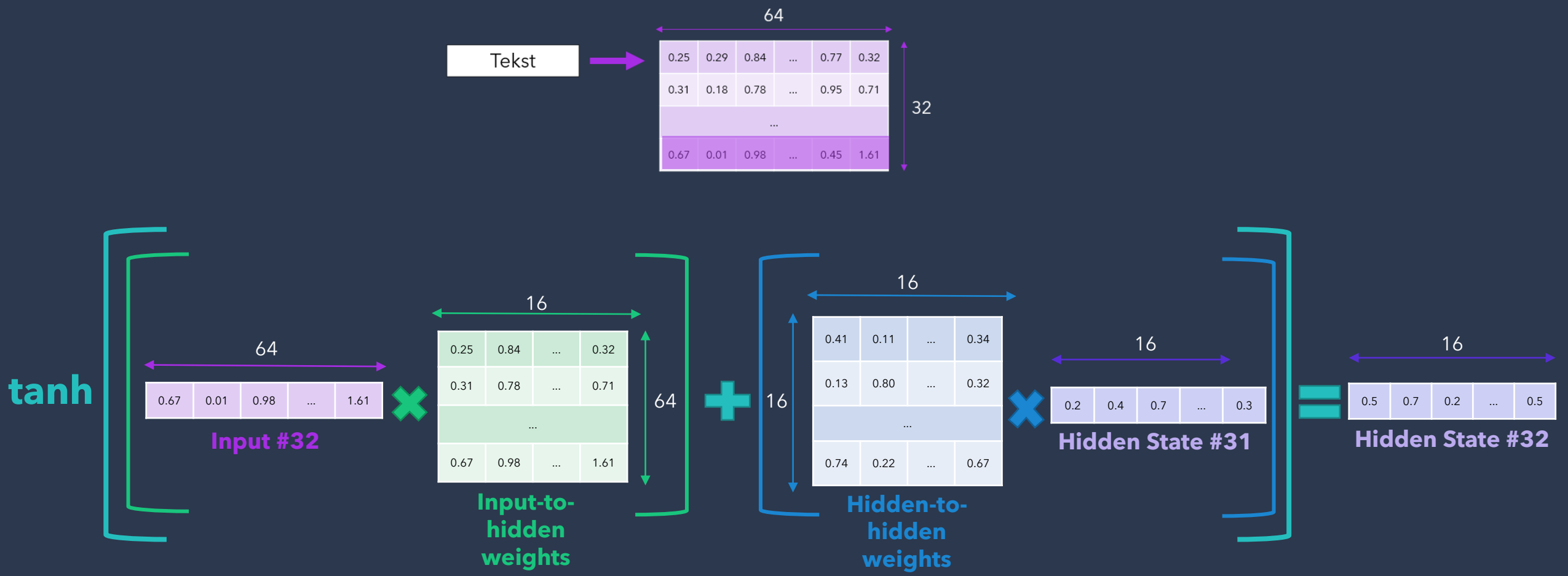
Kako izabrati model? – Rešenje #3 – RNN

Korak 2



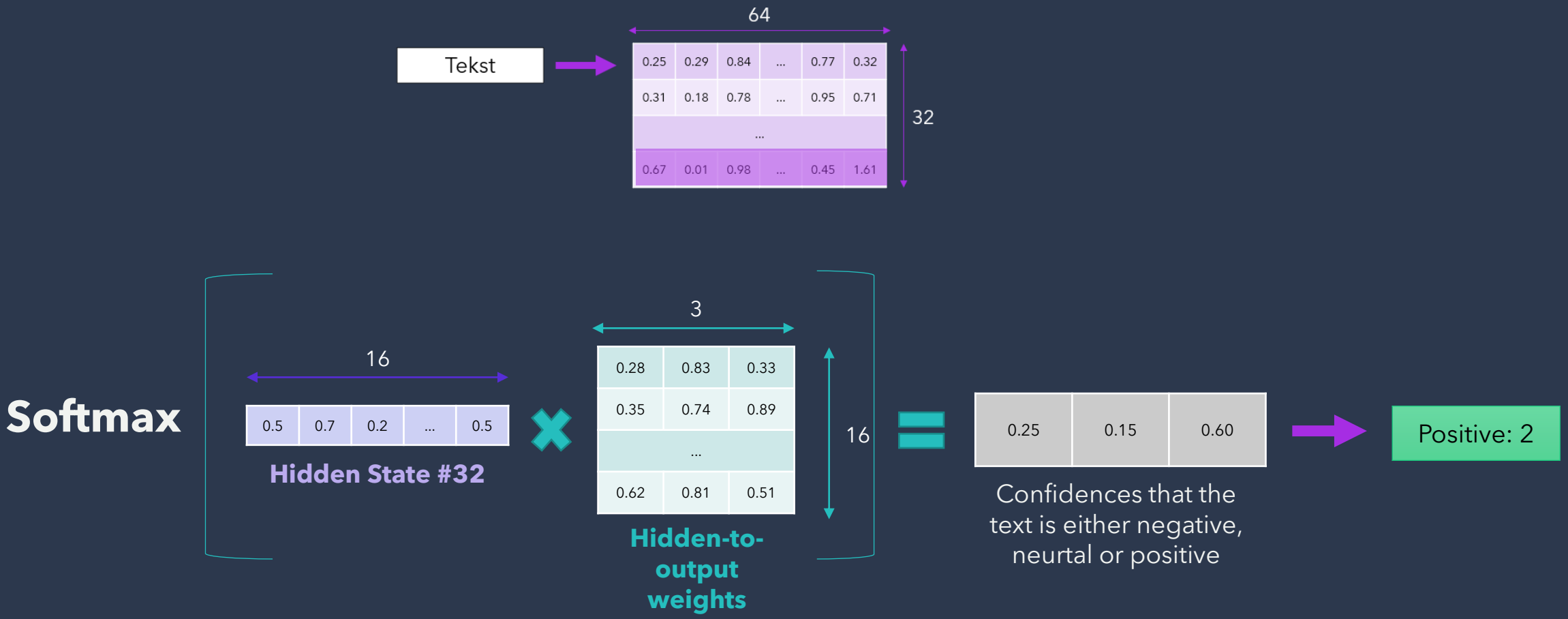
Kako izabrati model? – Rešenje #3 – RNN

Korak 32



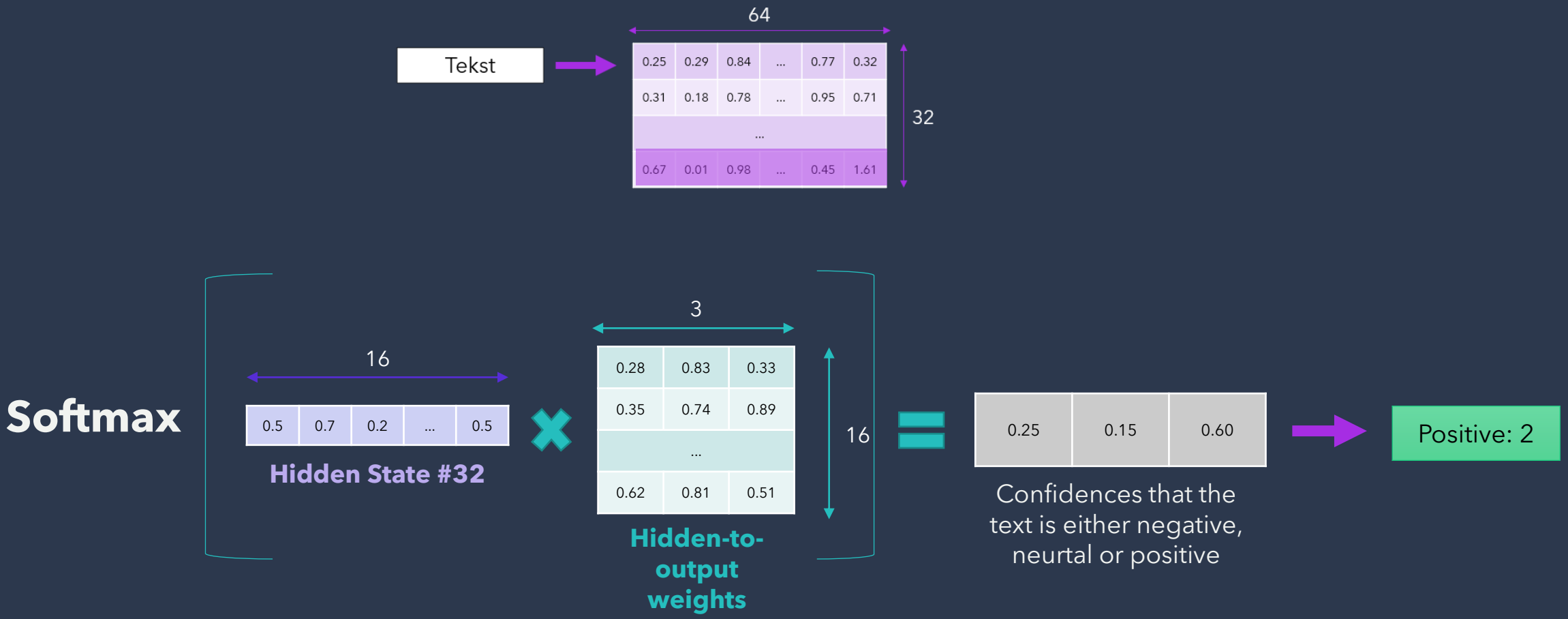
Kako izabrati model? – Rešenje #3 – RNN

Korak 32



Kako izabrati model? – Rešenje #3 – RNN

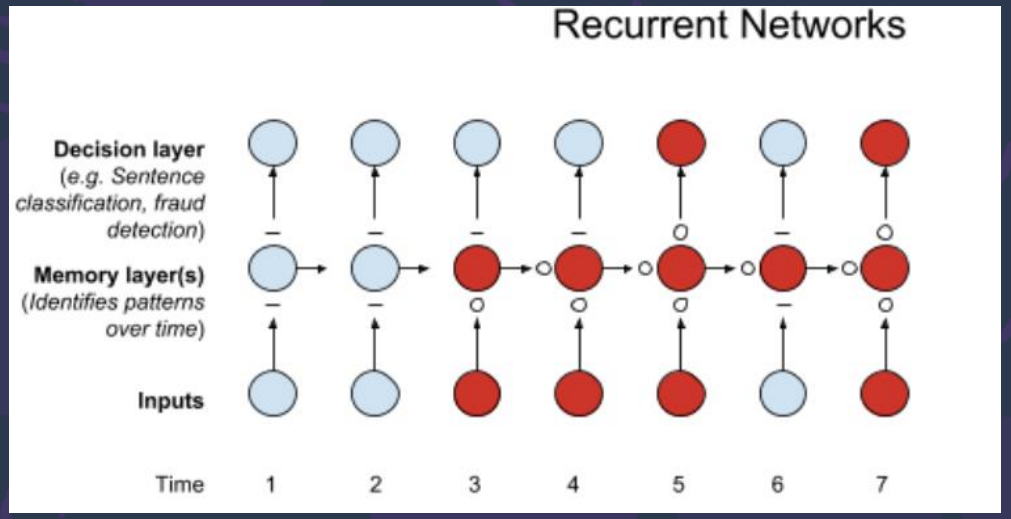
Korak 32

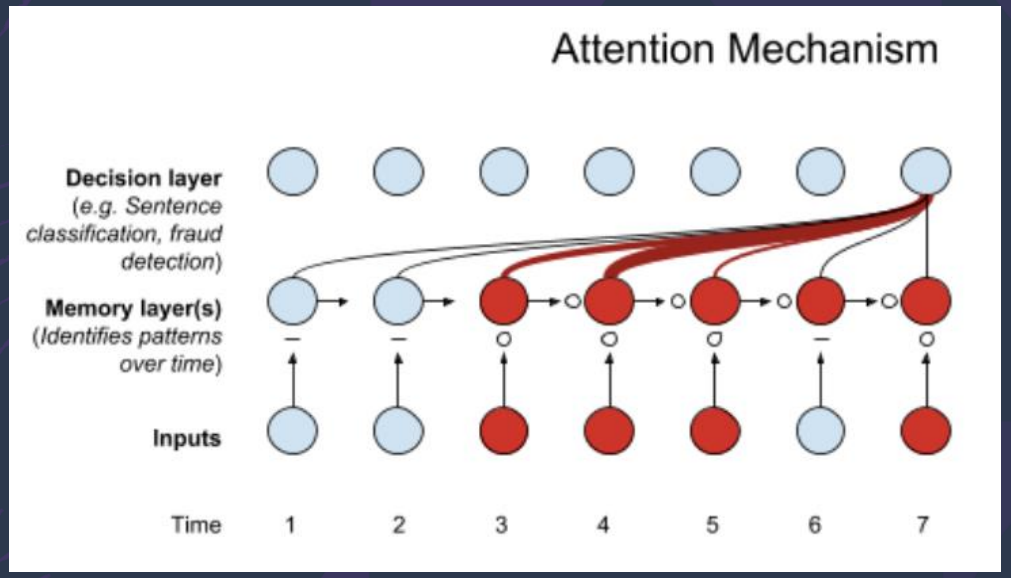


Ponovo smo rešili problem, ali...

Attention **mehanizam**

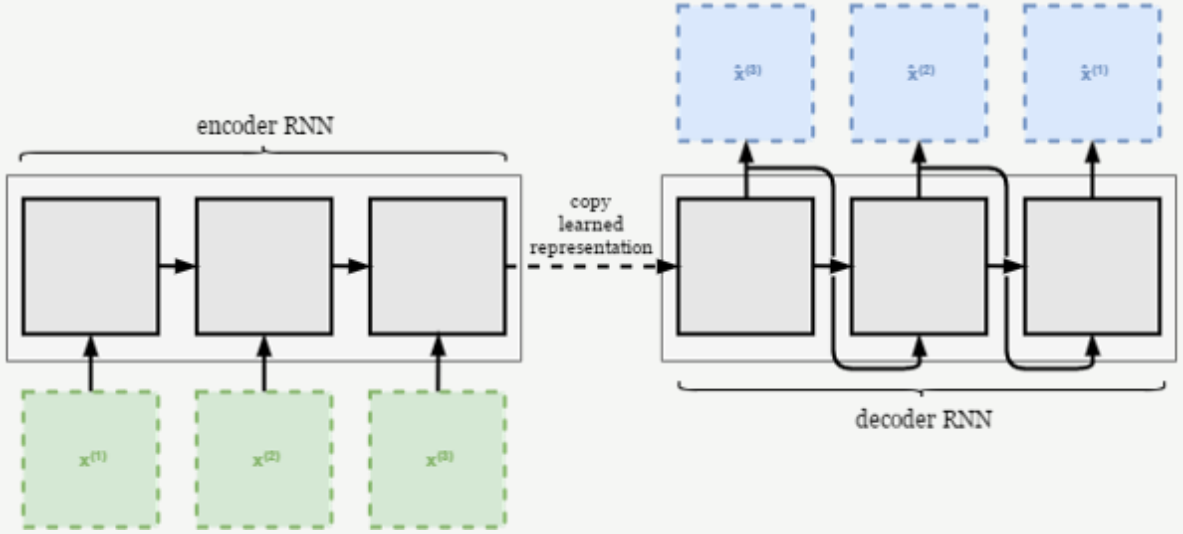
- Čest problem kod RNN-ova: predugačke zavisnosti
- ***Attention*** **mehanizam** se koristi kao prenosnik informacija, tako što prilagođava vrednosti težina koje su dodeljene svakom ulazu
- ***Self-attention*** – „pažnja" koju svaka reč pruža drugim rečima u okviru sekvence – za svaku reč, indentifikuje se koliko svaka druga reč doprinosi kontekstu posmatrane reči.
- RNN sa *attention* mehanizmom:
  - U tipičkom RNN-u, koriste se trenutni ulaz i prethodno „skriveno" stanje da bi se izračunalo trenutno „skriveno" stanje
  - Zahvanjujući *attention* mehanizmu, različite težine se pridružuju „skrivenim" stanjima i ulazima.



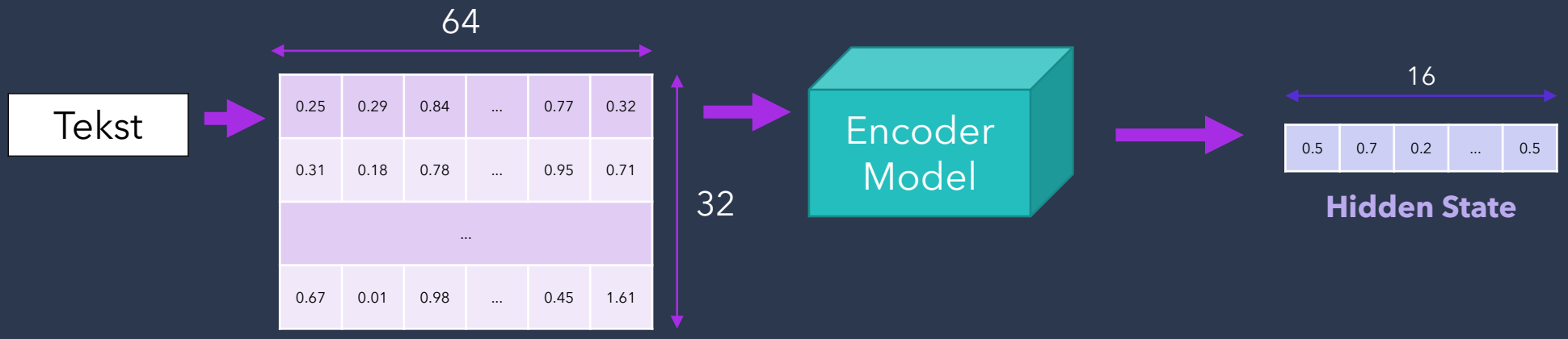


Transformer

- ***Transformer*** – model prvi put opisan u *"Attention is All You Need"*, ,koji se bazira samo na *attention* mehanizmu, bez korišćenja RNN-ova ili konvolucije
- Ima tipičnu ***encoder-decoder*** strukturu:
  - ***Encoder*** mapira ulaznu sekvencu u neku reprezentaciju
  - ***Decoder*** generiše izlaz na osnovu te reprezentacije, element po element, koristeći prethodno generisane elemente kao dodatne ulaze prilikom generisanje novih

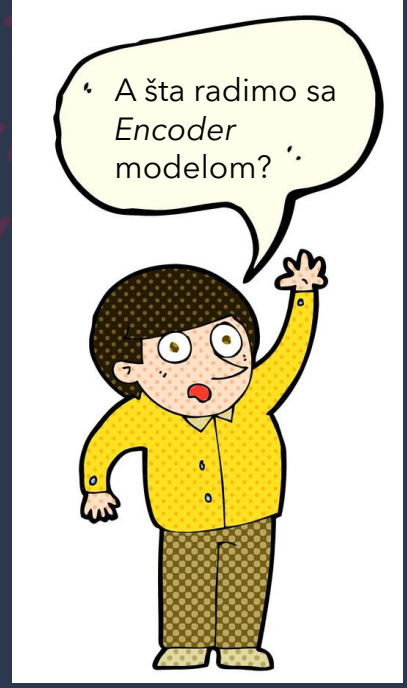


Kako predstaviti podatke? – Rešenje #4



- Za neki tekst **Encoder Model** predstavlja funkciju koja mapira taj tekst u vektor brojeva, dužine 16, što odgovara predefinisanoj veličini „skivenog“ sloja, odnosno generisane reprezentacije.
  - Za svaki ulazni tekst može se dobiti njegov *feature vector* (X)
  - Kako već imamo labelu za svaki tekst (Y), možemo konstruisati parove (X, Y) koje koristimo za treniranje ML modela
- Idealno bi bilo ako bi smo ovu reprezentaciju mogli da koristimo za različite probleme i samo prilagođavamo vrednosti **Hidden-To-Output** matrici.

Kako izabrati model? – Rešenje #4



Tekst

64

0.25	0.29	0.84	...	0.77	0.32
0.31	0.18	0.78	...	0.95	0.71
...					
0.67	0.01	0.98	...	0.45	1.61

32

Encoder Model

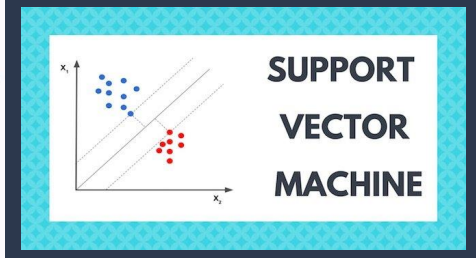
16

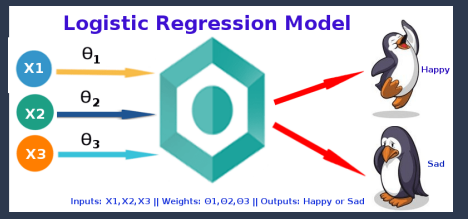
0.5	0.7	0.2	...	0.5

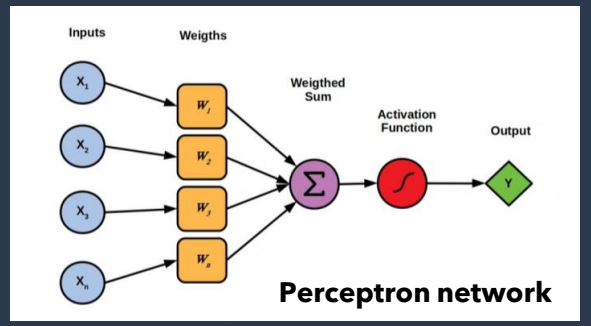
Hidden State

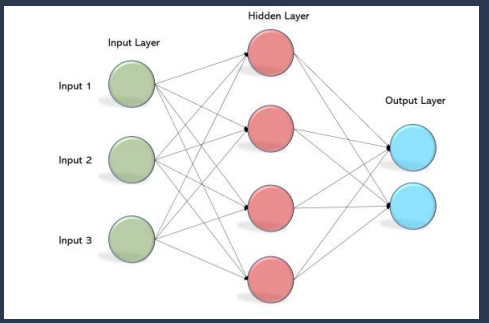
ML Model

Positive: 2



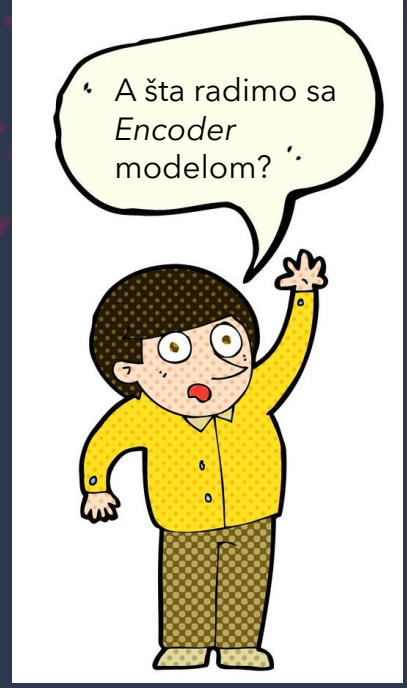


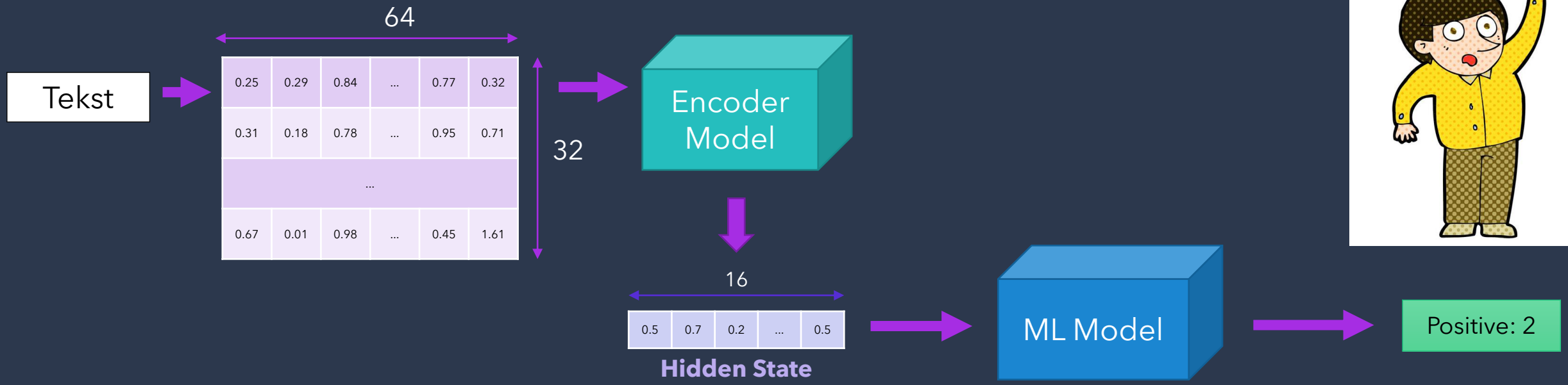


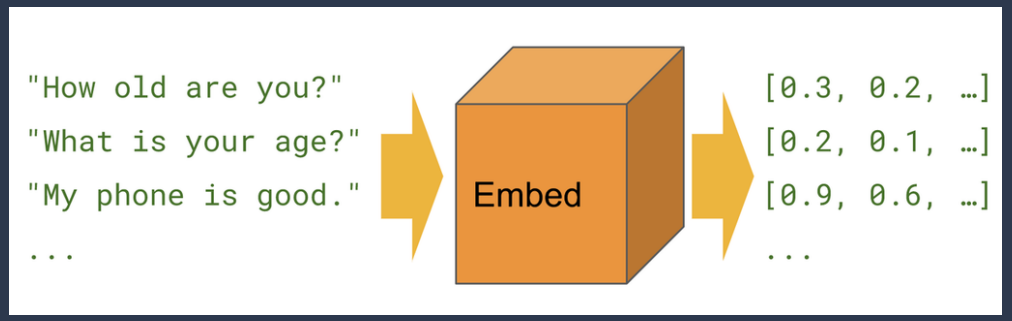


MLP network

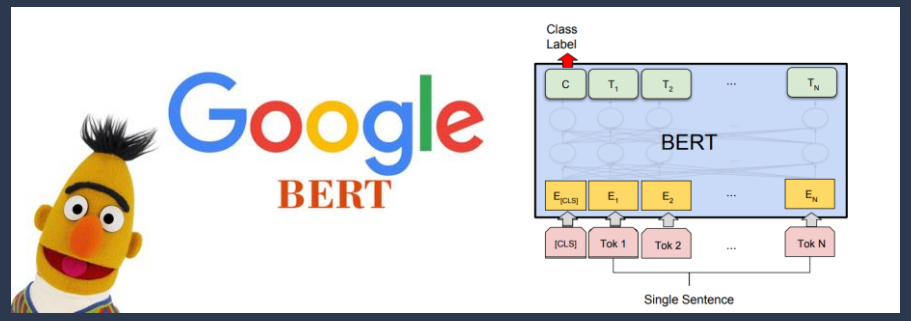
Kako izabrati model? – Rešenje #4







Universal Sentence Encoder, Cer et al. 2018



BERT, Devlin et al. 2018

Open GPT, Radford et al. 2018

Pitanja?


A kako ovo da implementiramo?
Da li ovo radi samo za engleski?
Da li je uredu da koristimo neka već gotova rešenja?
Kako se mogu generisati lažne vesti?
Kako izabrati neko od ovih rešenja?
A šta je sa drugim problemima u *NLP*-u?

Pitanja?