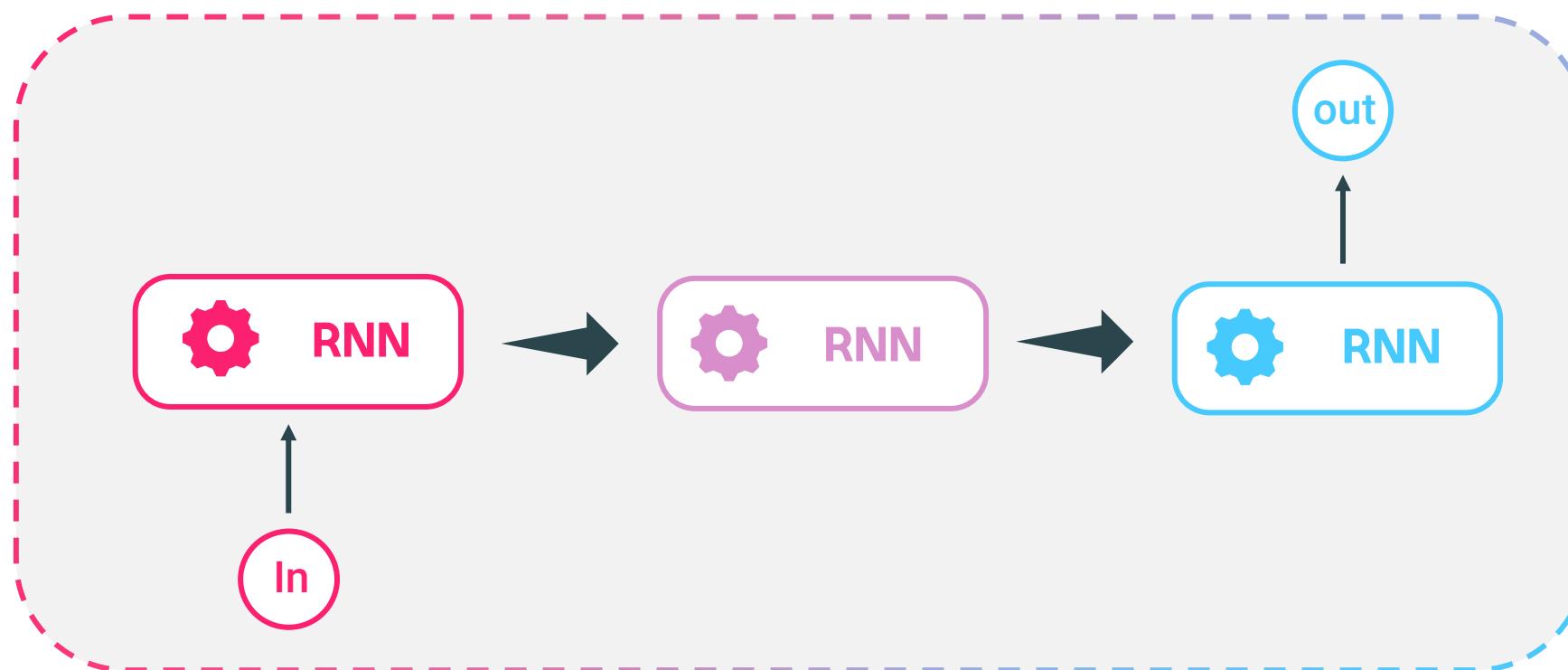


دوره یادگیری عمیق پیشرفته - شبکه های بازگشتی

جلسه اول : مقدمات شبکه های بازگشتی



حامد قاسمی

دانشجوی دکتری هوش مصنوعی دانشگاه تهران



نحوه استفاده شما از دوره:

20 درصد ویدیو

20 درصد مرور شما

60 درصد حل تمرینات و فعالیت در گروه

خطای متداول



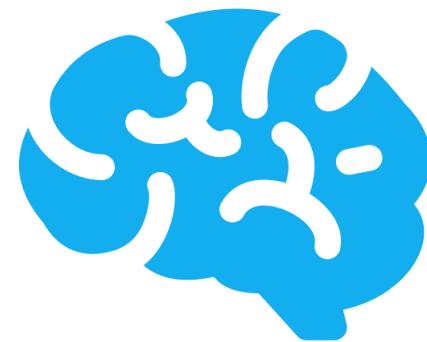
100 → 90
80

40 50

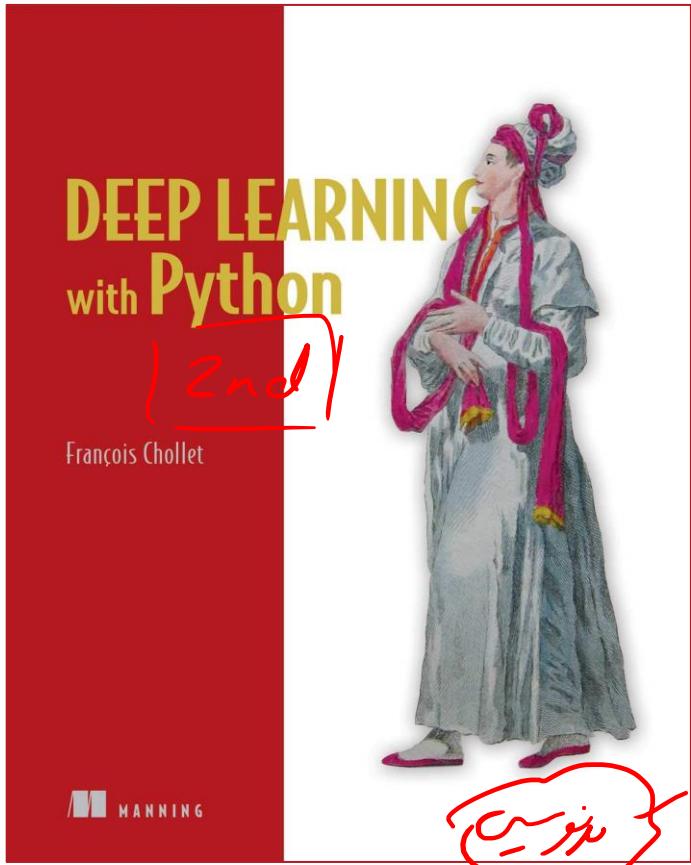


دوره سخته؟ بلم-سخته است.

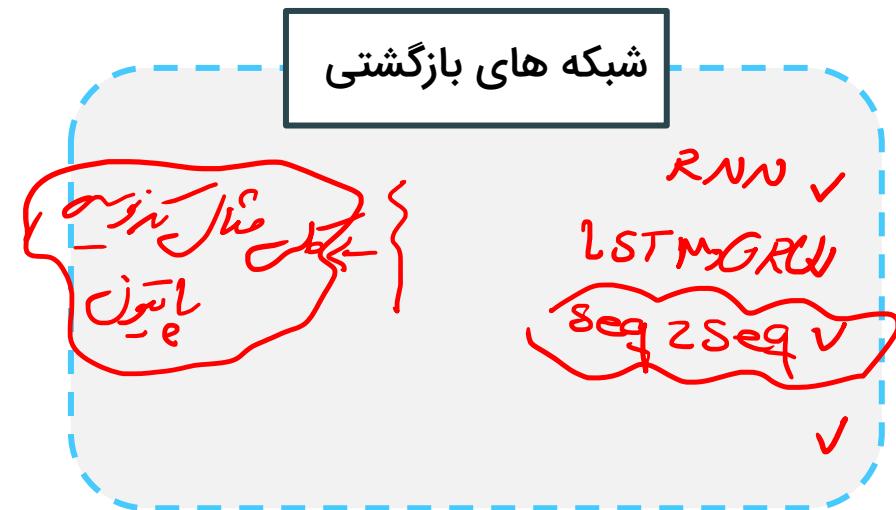
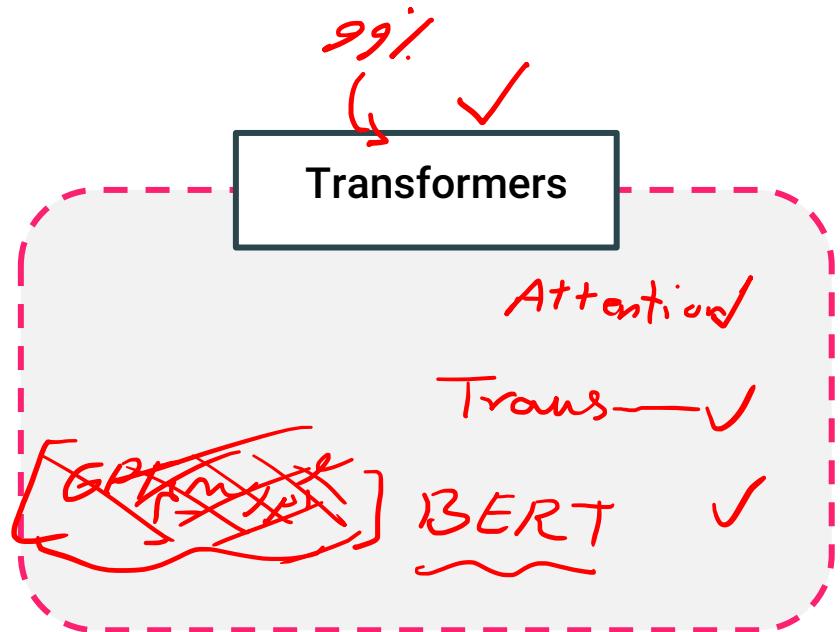
در زمان سختی به ذهنی سرد و قلبی گرم احتیاج داریم.



منابع مورد استفاده در دوره



چه خواهیم گفت :



۱۰:۱۱:۲۰
۱۱:۲۰ → ۱۱:۴۰ ۶
۱۱:۴۰ → ۱۱:۴۰

آنچه در این جلسه گفته خواهد شد :

بخش سوم

❖ دو مثال ساده از این بزرگوار



بخش دوم

❖ توضیح دقیق شبکه های بازگشتی

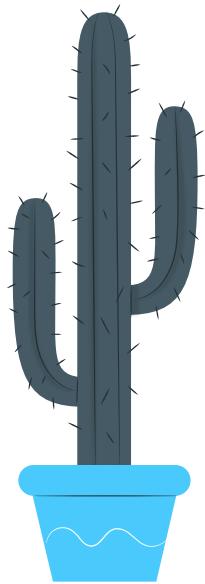
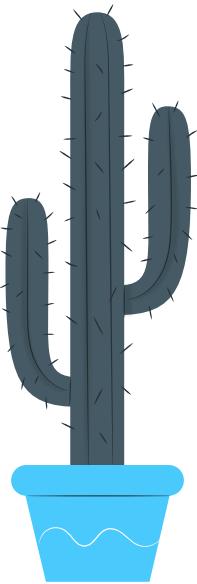
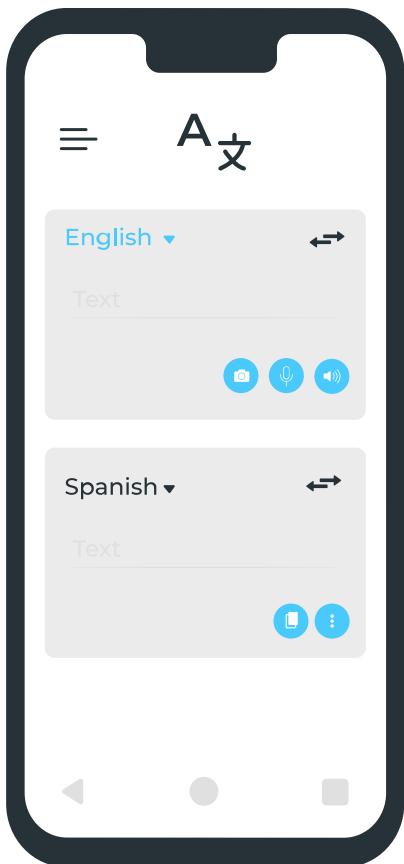


بخش اول

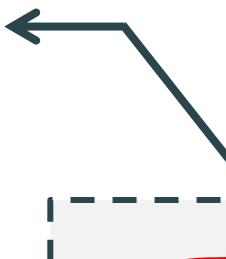
❖ کاربرد شبکه های بازگشتی کجاست ؟



بخش اول کاربرد شبکه های بازگشتی کجاست ؟



اشاره به زبان انسان

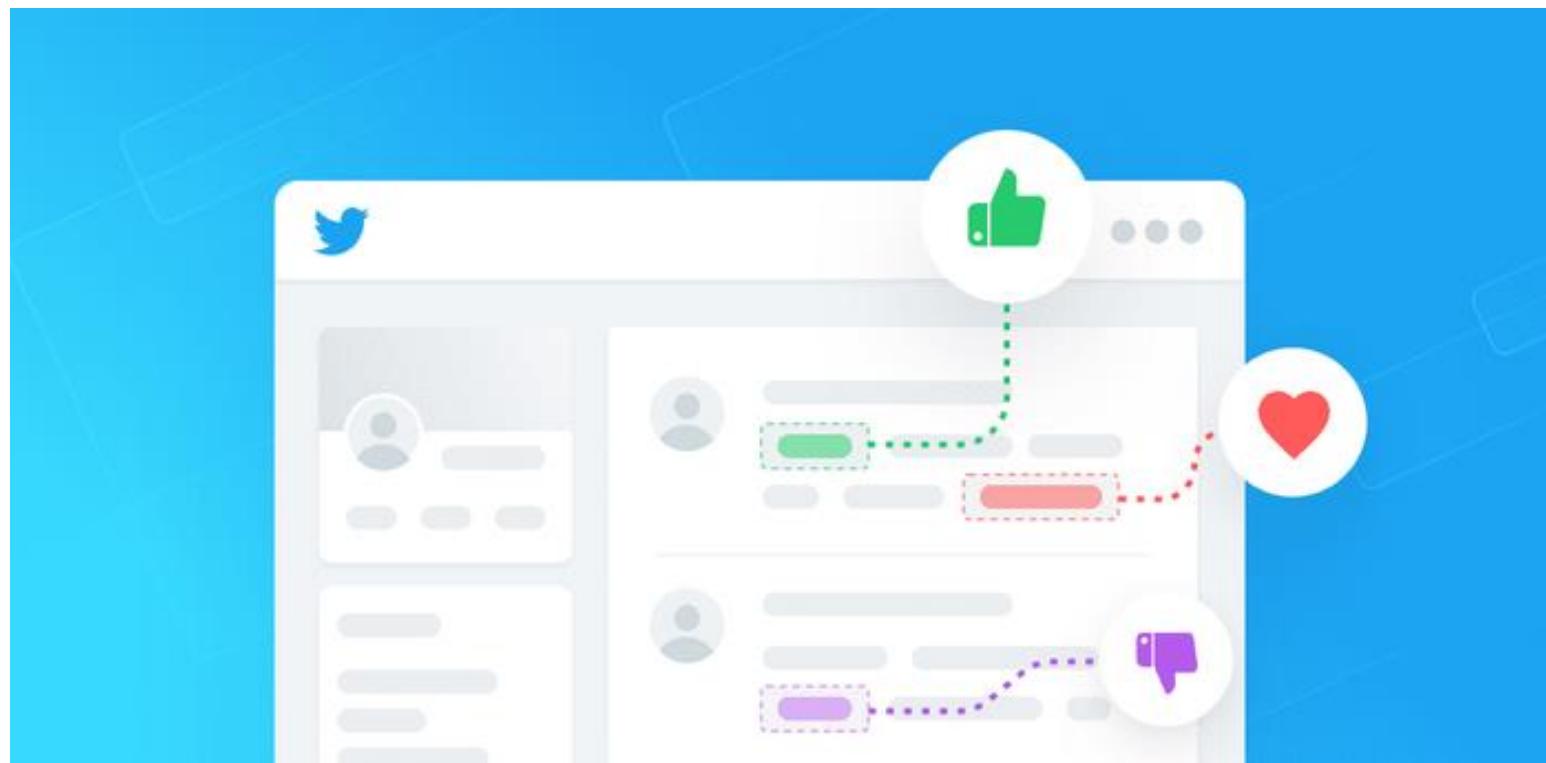


پردازش زبان طبیعی

در مقابل زبان های برنامه نویسی مثل C

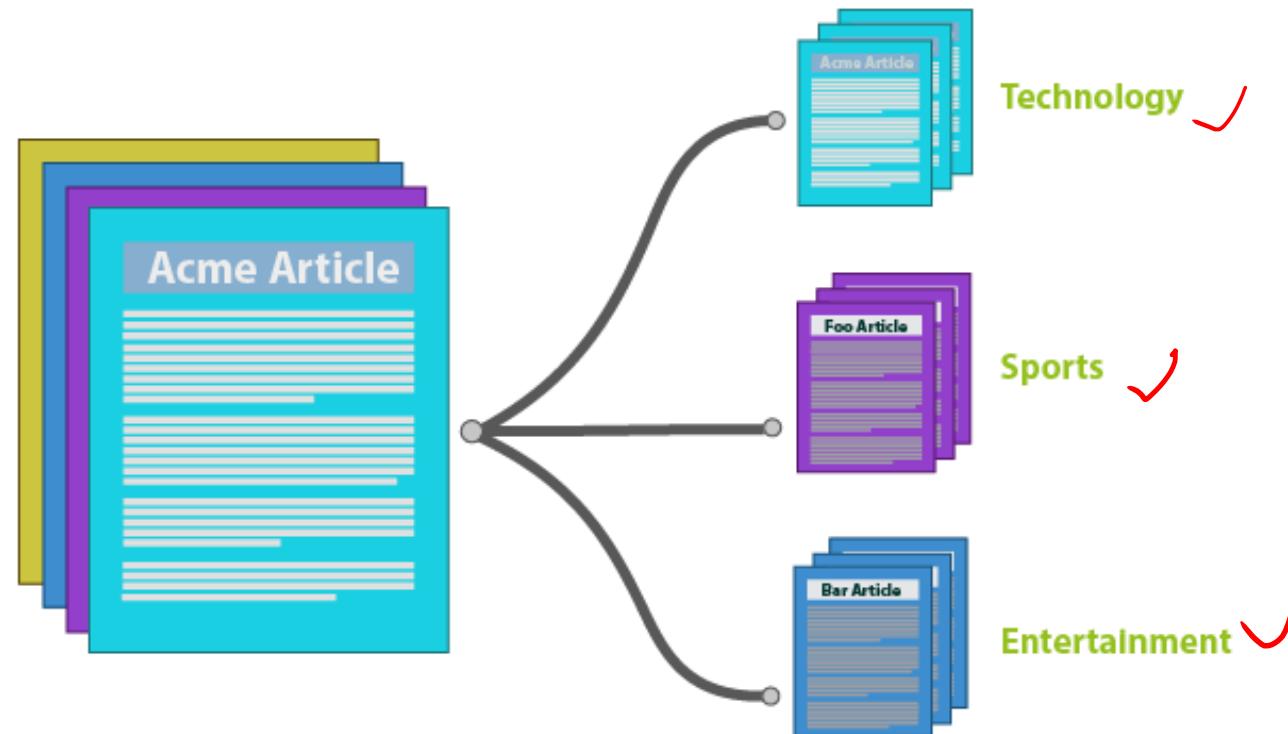


برخی از Task ها در پردازش زبان طبیعی



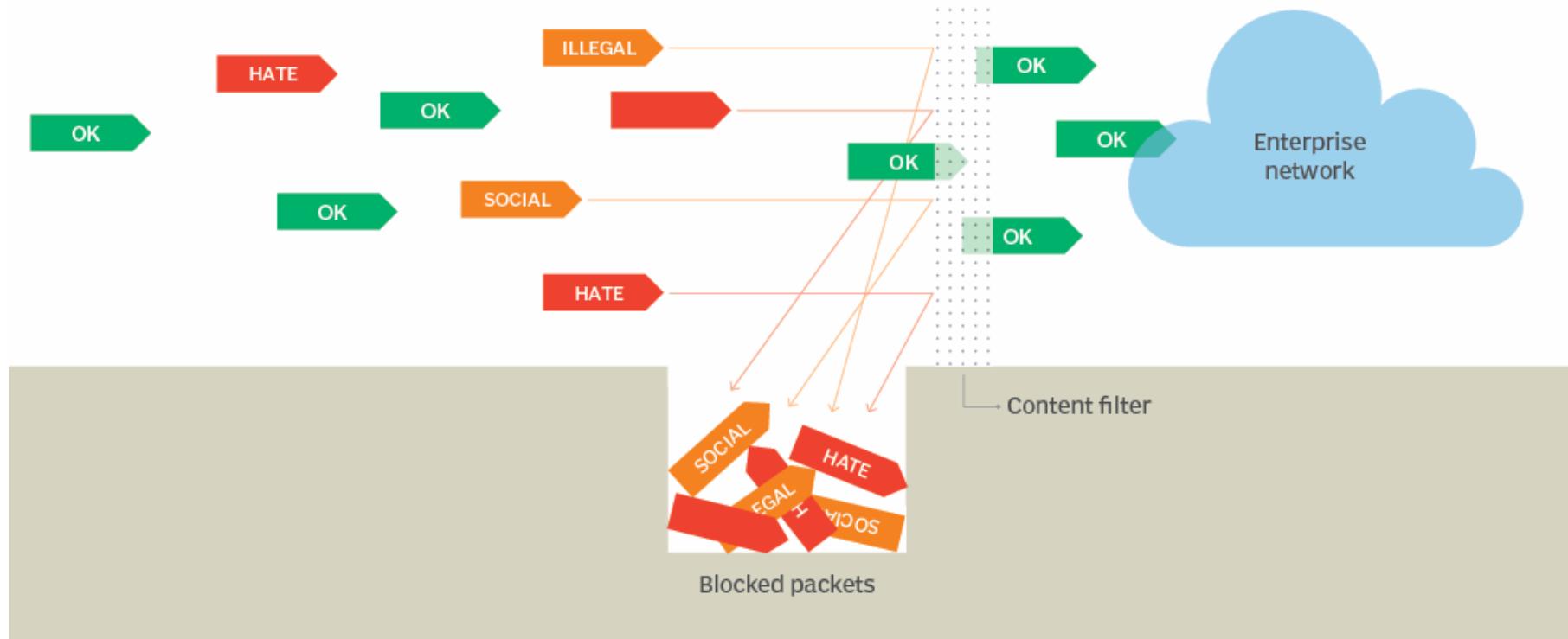
Text Classification

موضوع متن چیست ؟



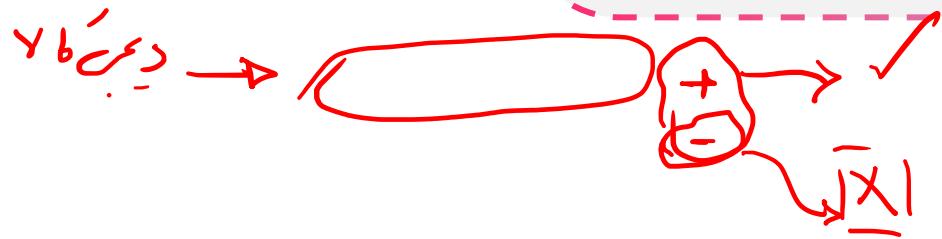
Content Filtering

متن حاوی توهین هست ؟



متن مثبت هست یا منفی؟

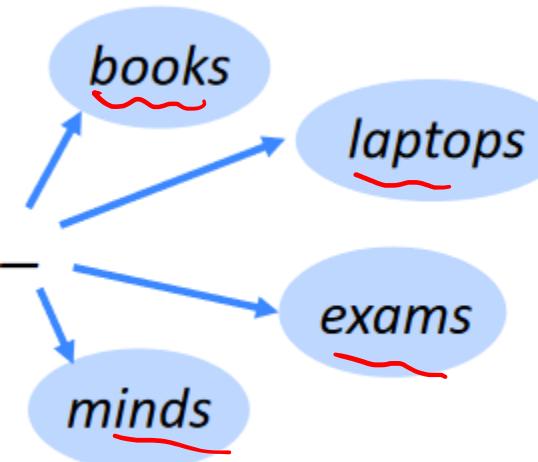
Sentiment Analysis



Language Modeling

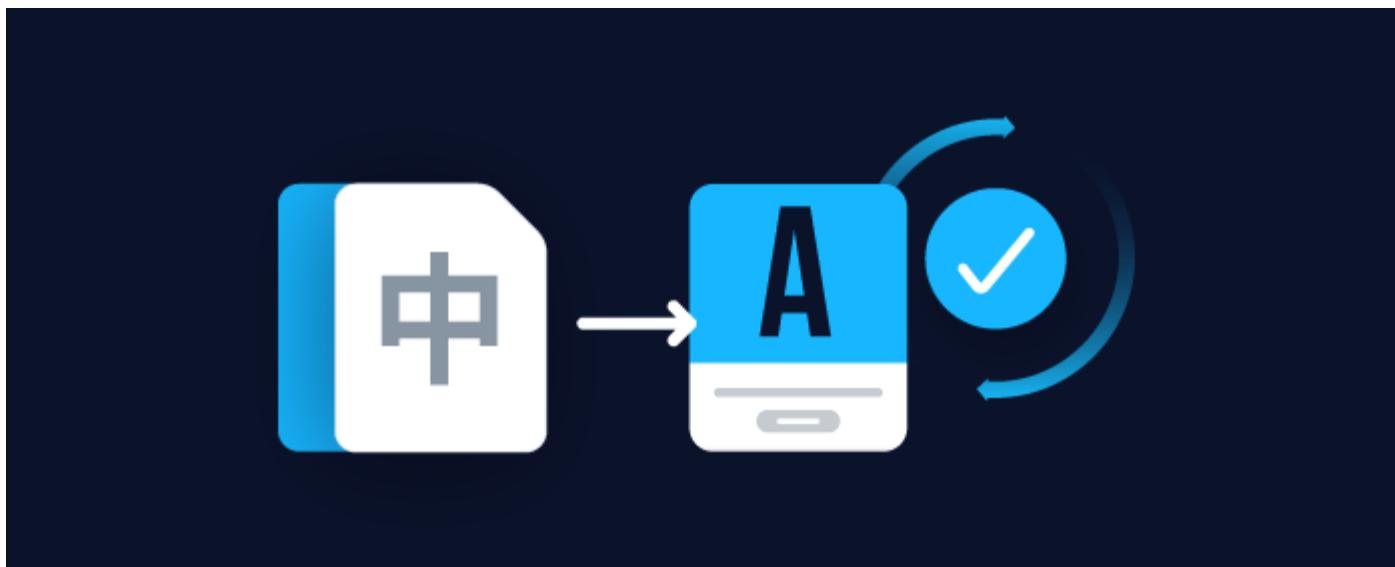
کلمه بعدی چیه ؟

the students opened their _____?



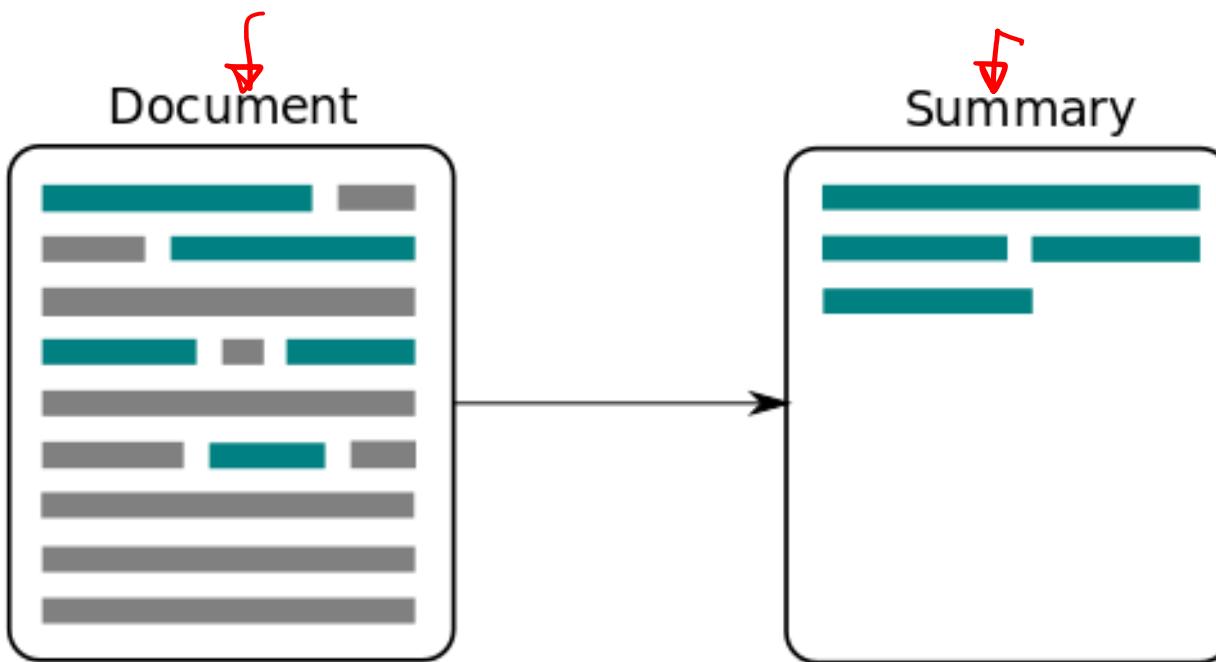
Translation

ترجمه متن به زبانی دیگر

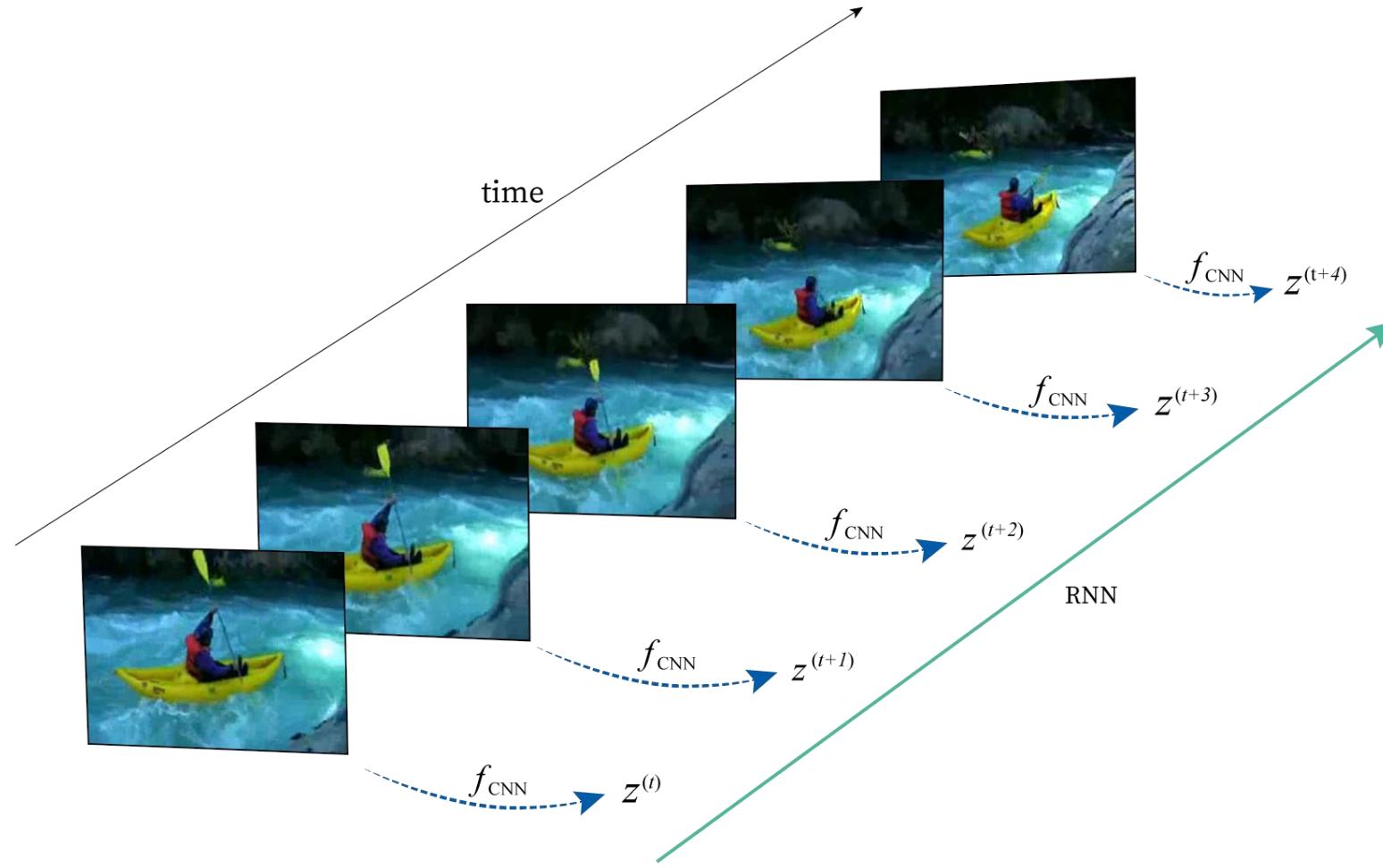


Text Summarization

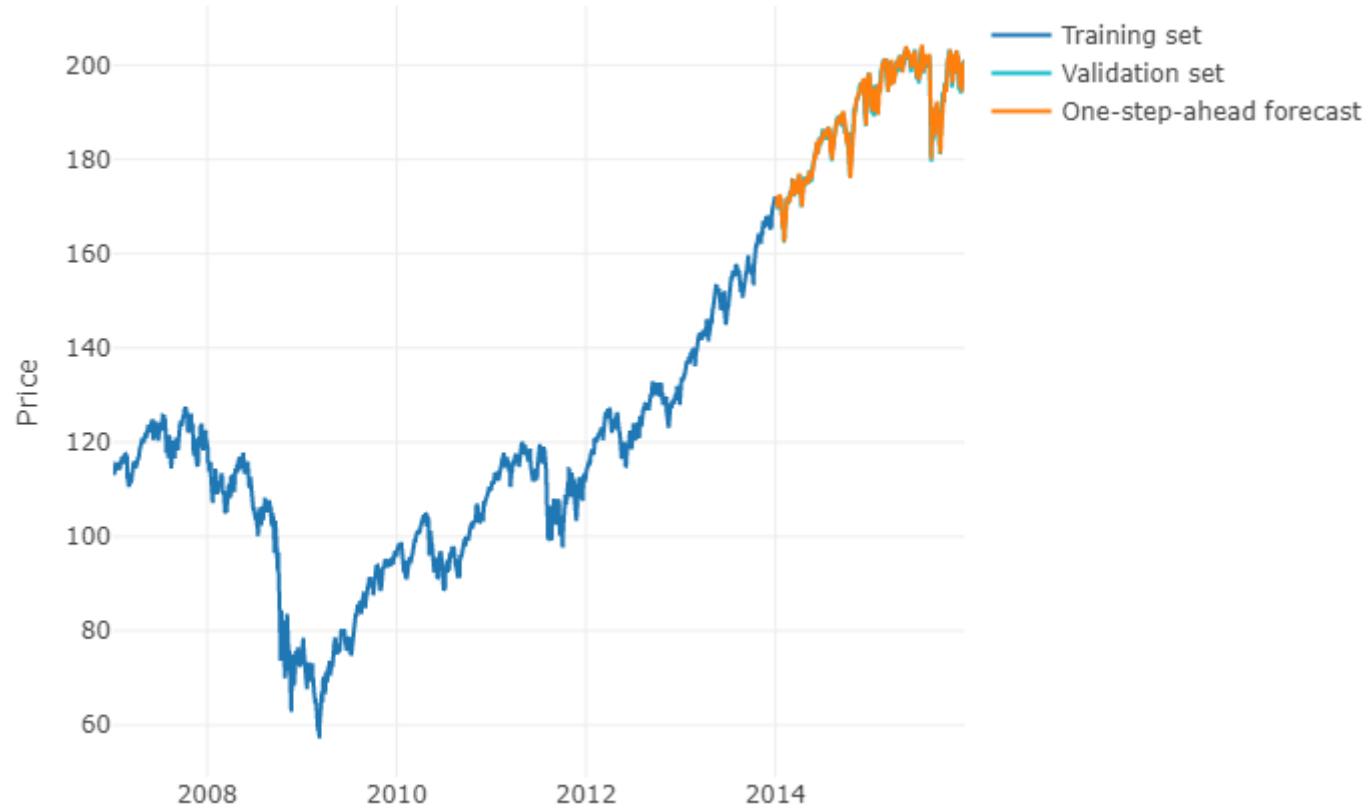
خلاصه سازی یک متن در یک پارگراف



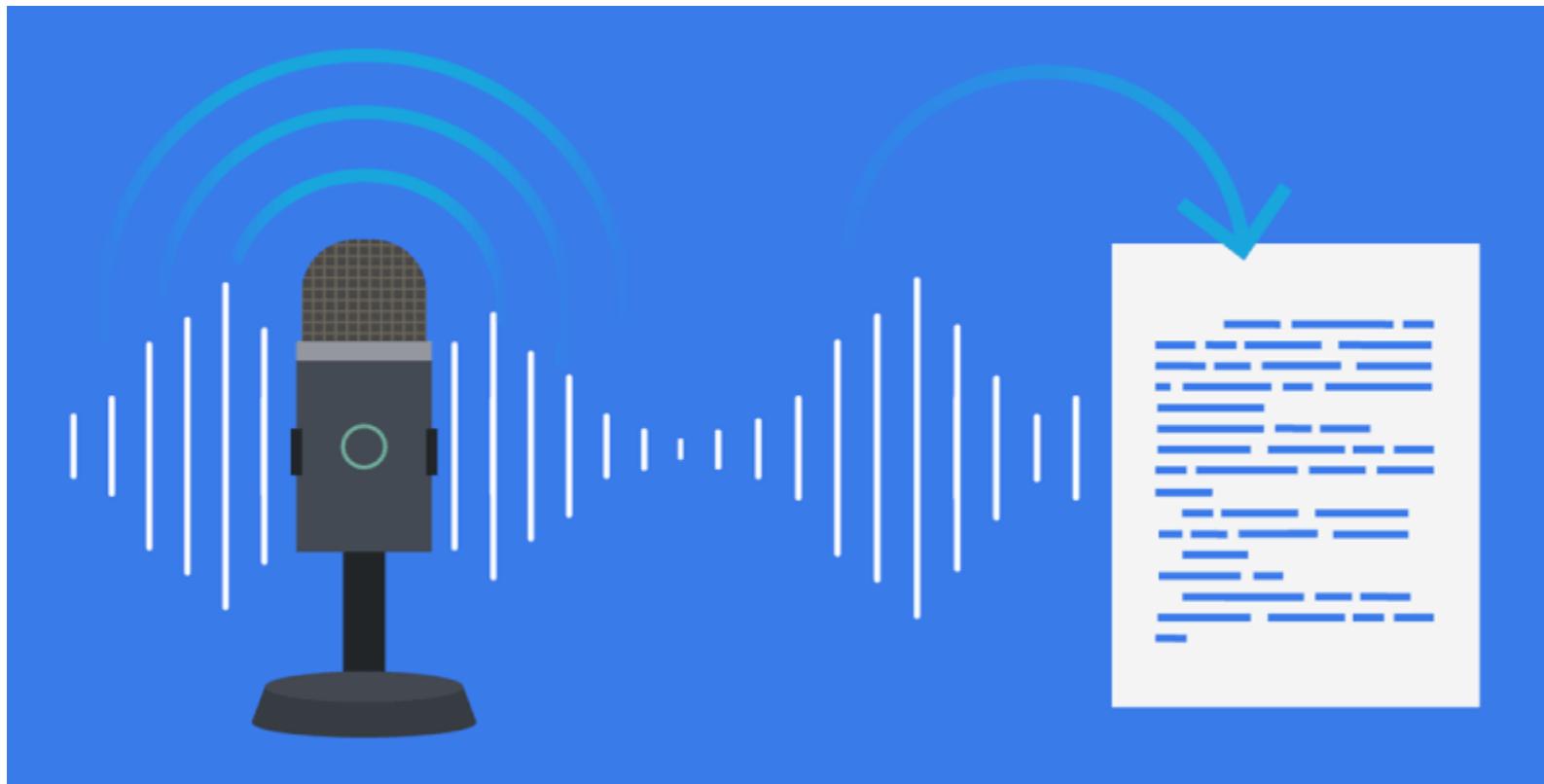
Video Classification



Time series forecasting



Speech Recognition



تا الان :

بخش سوم

- ❖ دو مثال ساده از این بزرگوار



بخش سوم

بخش دوم

- ❖ توضیح دقیق شبکه های بازگشتی

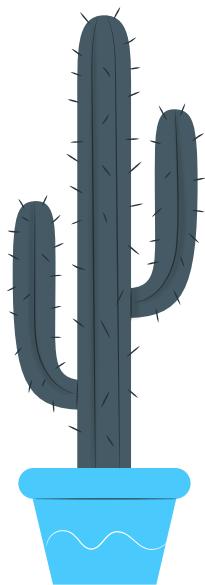
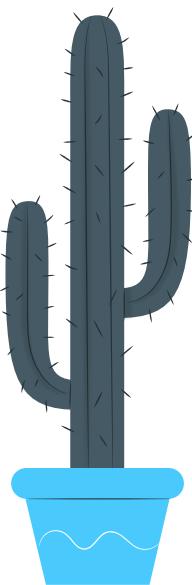
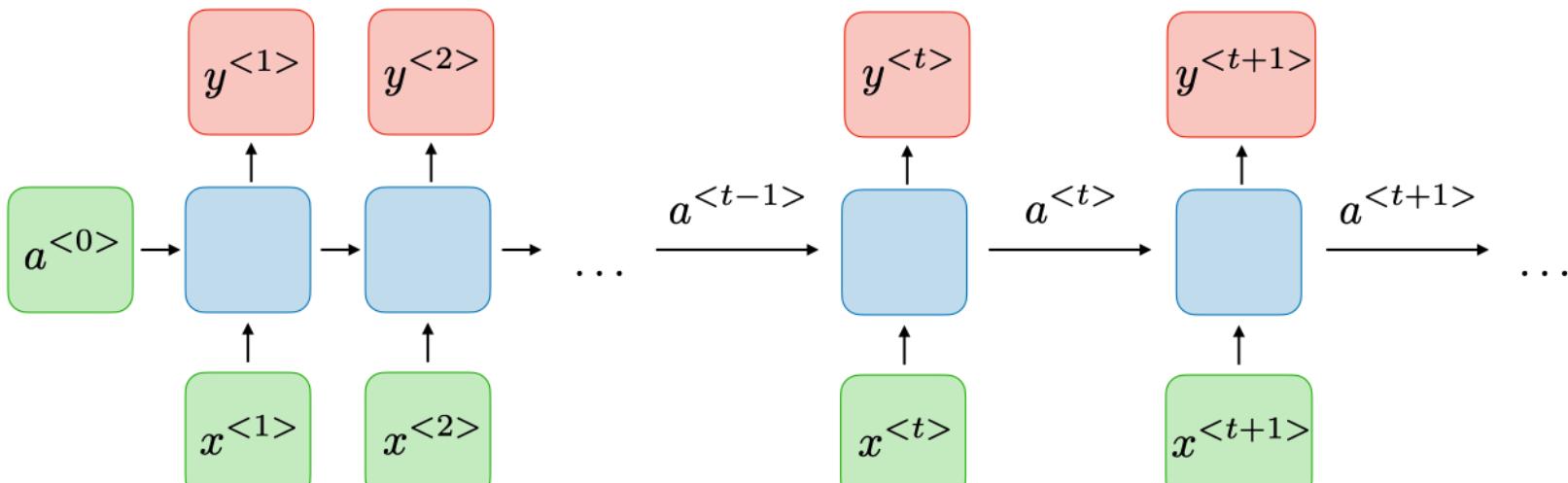


بخش اول

- ❖ کاربرد شبکه های بازگشتی کجاست ؟



بخش دوم : توضیح دقیق شبکه های بازگشتی



قبل از ادامه باید سه نکته و مفهوم مرور شود.

!

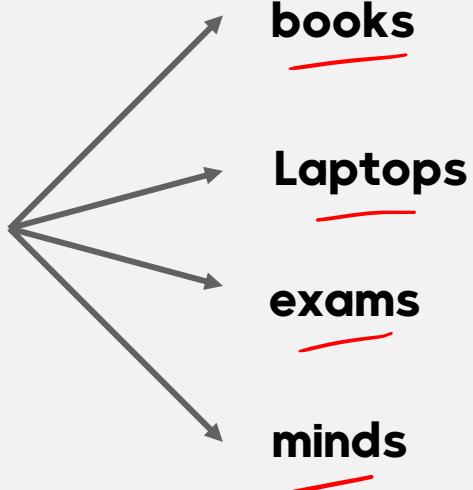


مفهوم Language Modeling

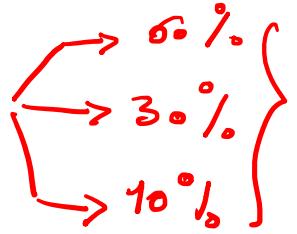
به صورت ساده پیش‌بینی کلمه بعدی است.



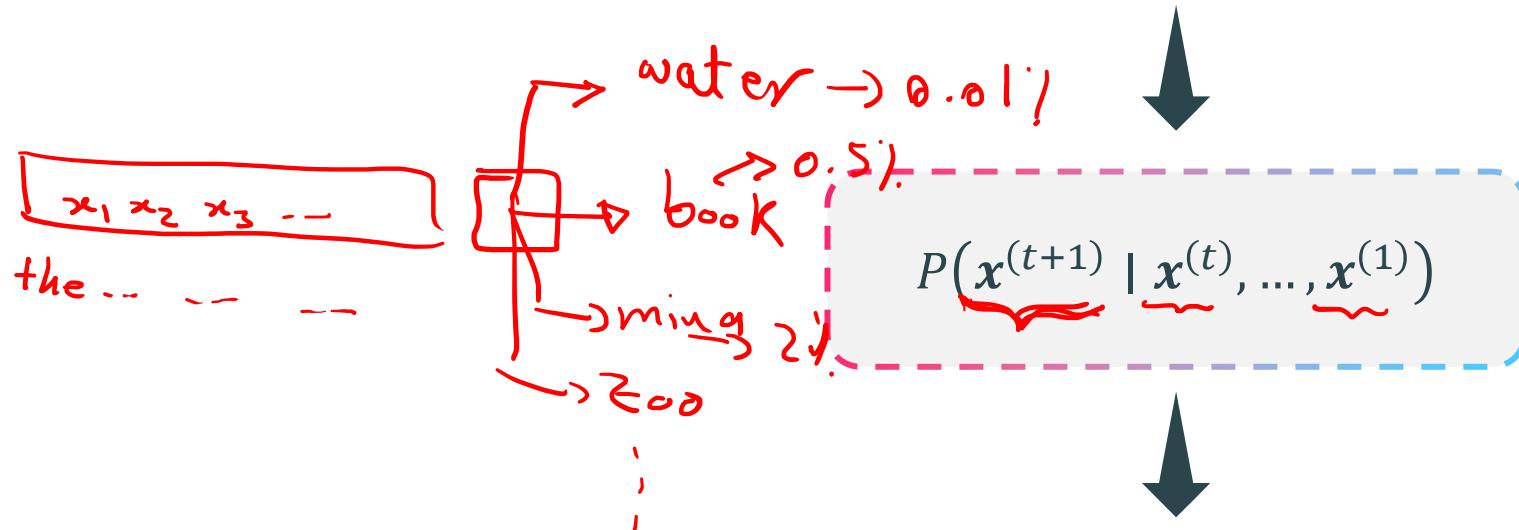
The Students Opened Their



کمی رسمی تر:



اگر مجموعه کلمات x_1 و x_2 و ... و x_t را داشته باشیم ، احتمال $x^{(t+1)}$ را محاسبه کنیم.

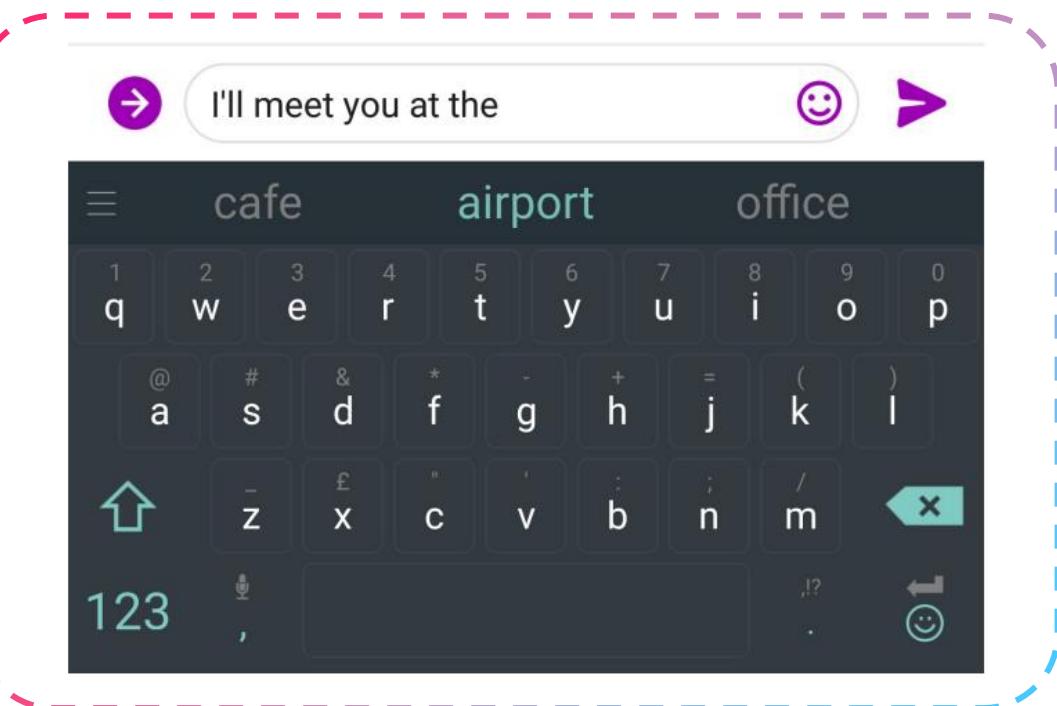
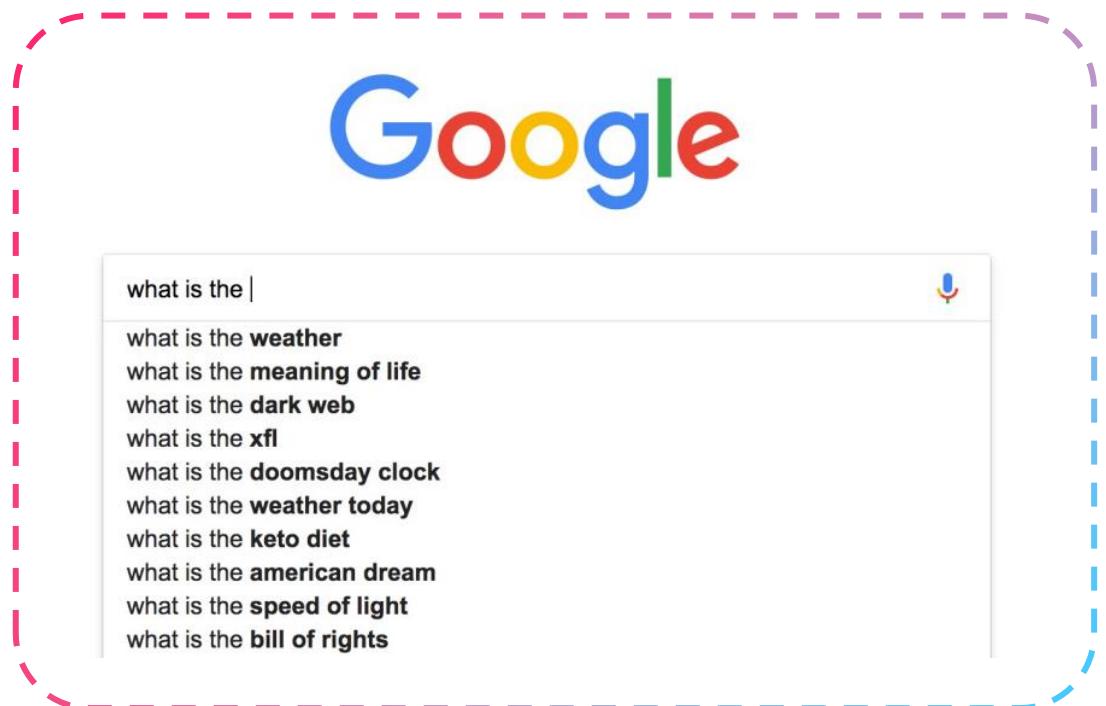


$$V = \{w_1, \dots, w_{|V|}\}$$

میتواند هر واژه در دیکشنری V باشد.



LM ہا کجا کاربرد دارہ ؟



۲

صـ

hello →

[]

یک نکته :

کلمه خام

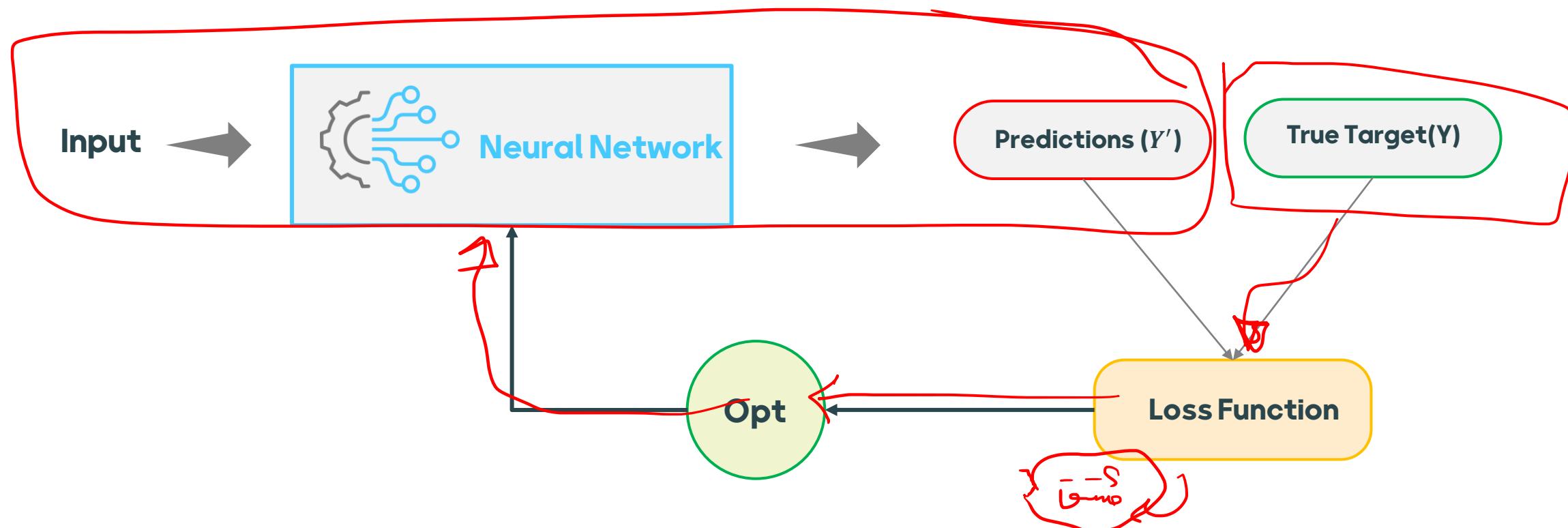


تابع E



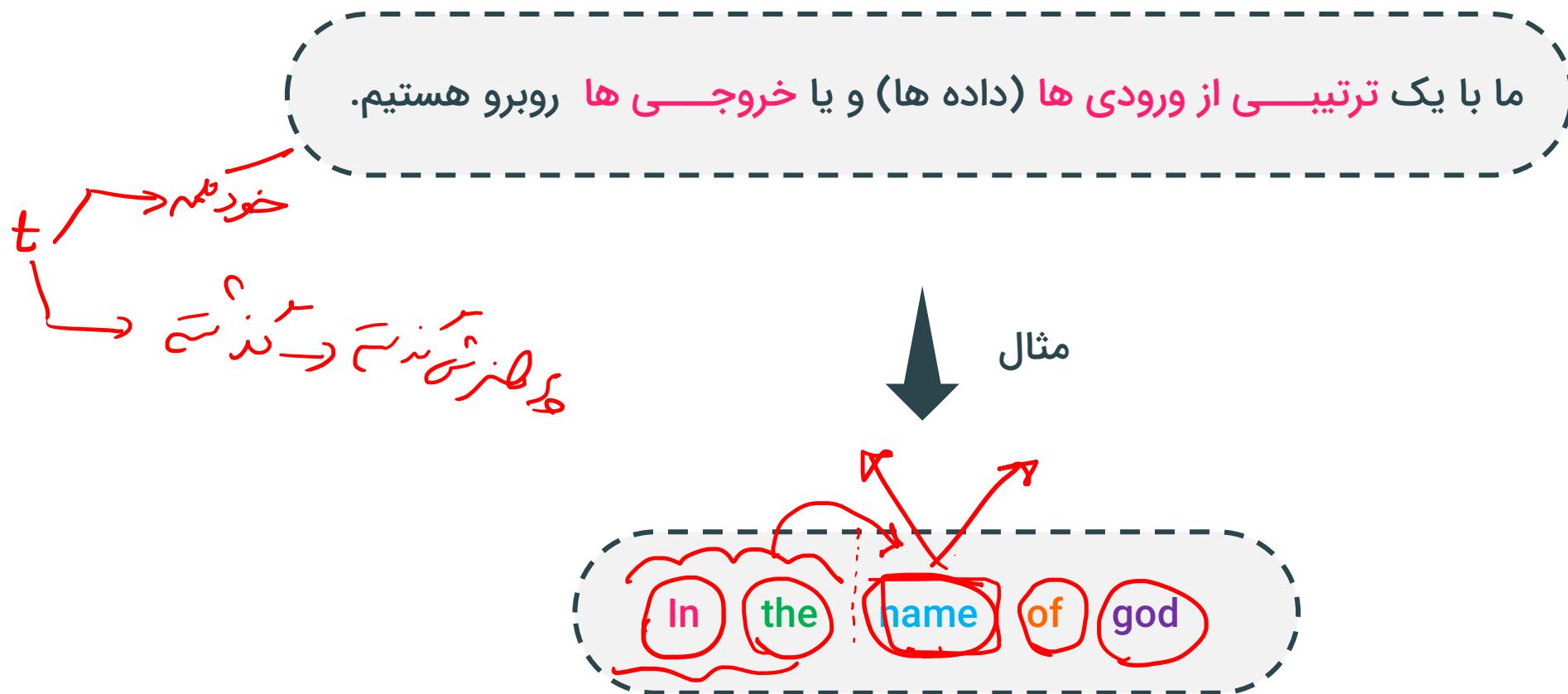
یک نکته عدـ

فرآیند کلی یک شبکه عصبی



دیگه بريم سر وقت اصل مطلب

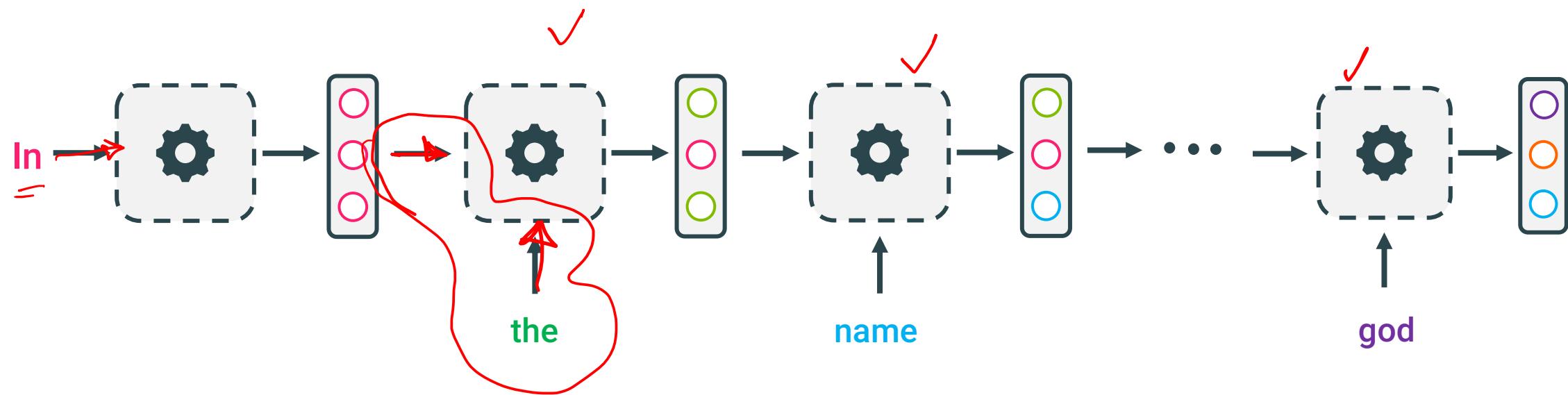
تفاوت اصلی کجاست ؟



تفاوت اصلی کجاست ؟

و به خاطر **جایگاه** نیاز هست تا ما اینها رو **به ترتیب پردازش** کنیم.

به عبارت دیگر :



In the name of god

چند نکته :

به هر کدام از بردارهای پردازش شده **hidden State** می گوییم.



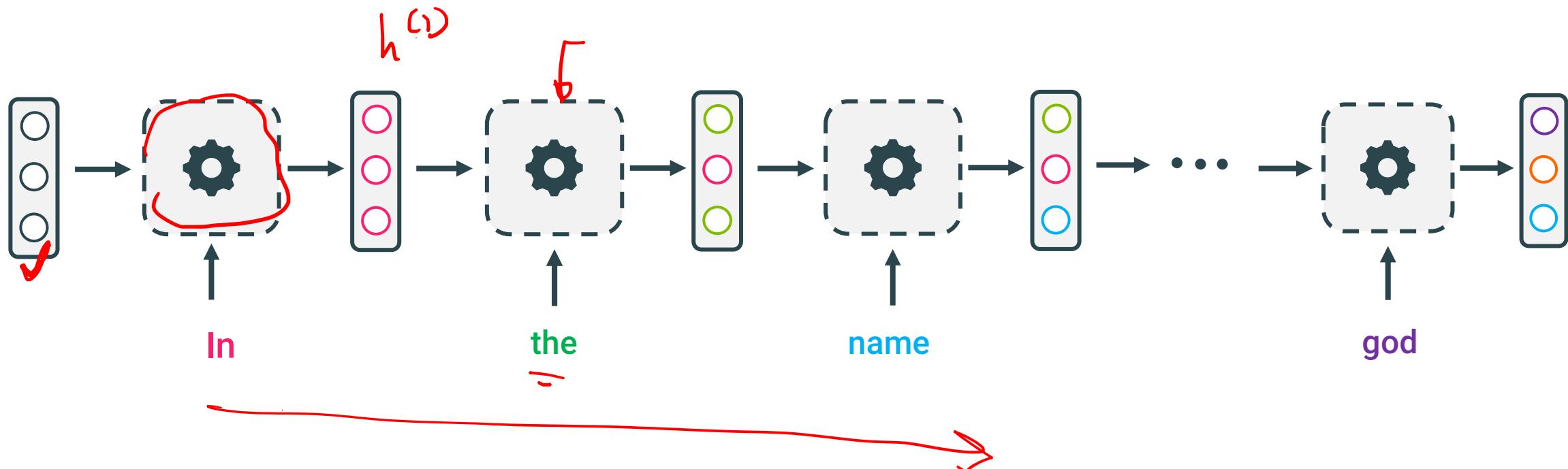
هر واحد پردازشی ما به جای یک ورودی، دو ورودی دارد. خود کلمه و بردار

حاصل از پردازش کل خروجی های قبلی (به جز اولی که کار رو خراب کرده !)



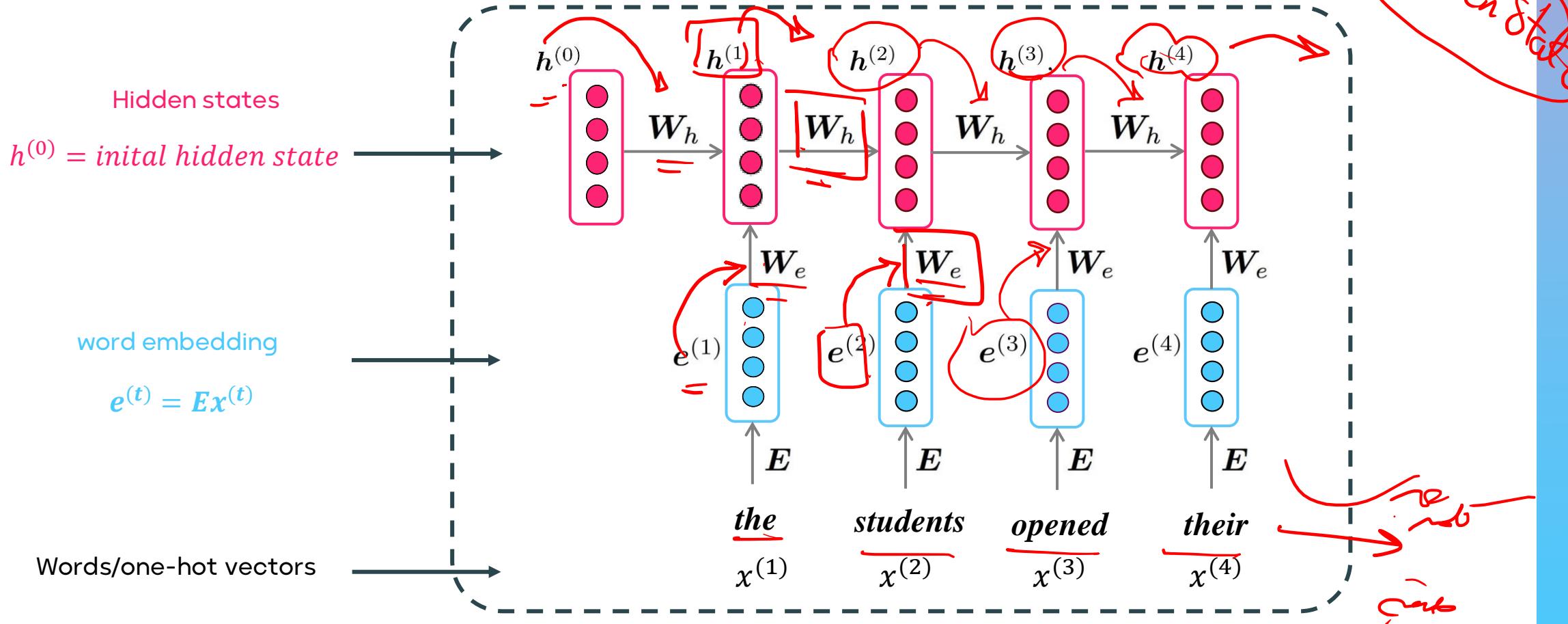
چجور مشکل اولی را حل کنیم؟

یه بردار رندوم در نظر میگیریم برای شروع که بتوانیم فرمول بندی واحدی رو ارایه بدیم



لَهْ :

كمي رسمي تر:

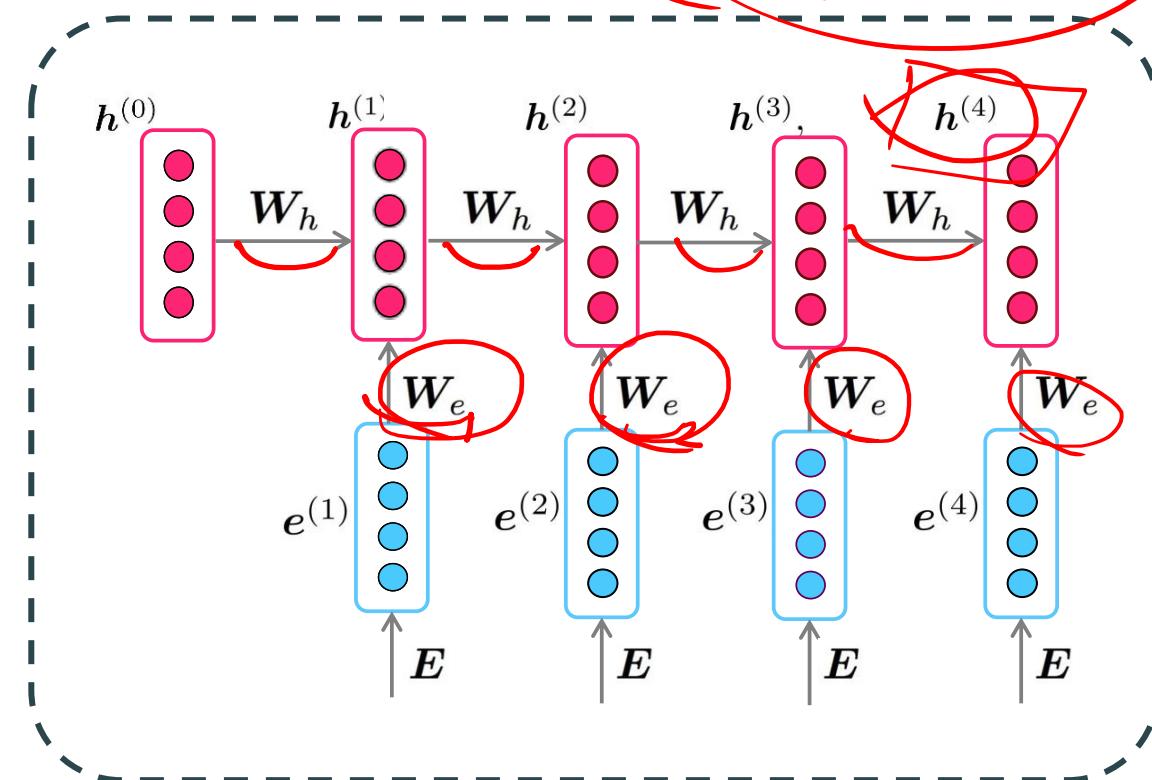
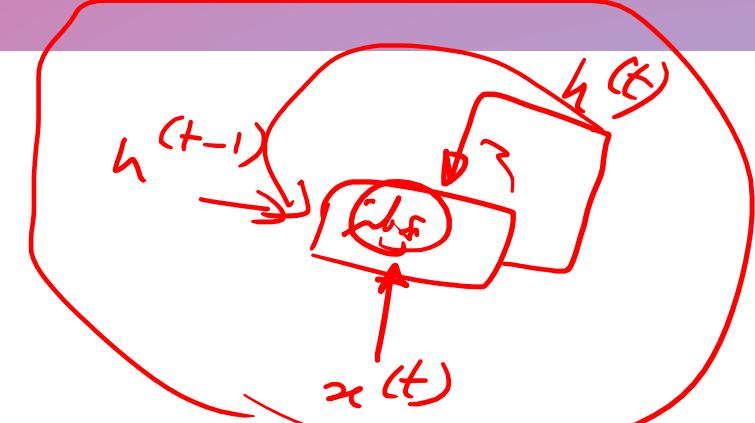


$$h^{(t)} = \sigma(\omega X + b)$$

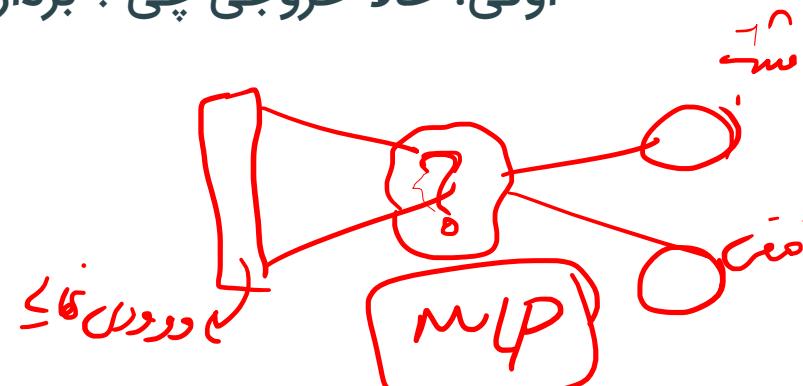
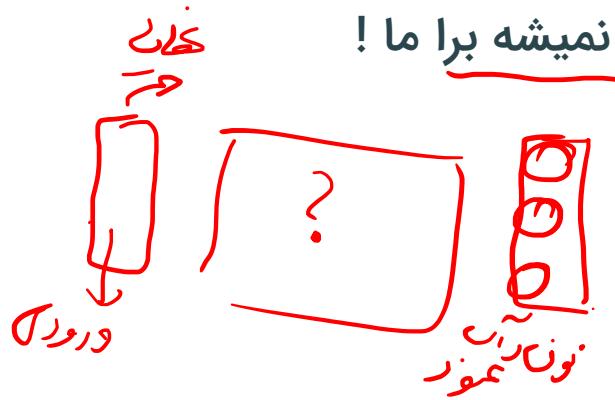
hidden state رابطه ریاضی

$$h^{(t)} = \sigma(W_h h^{(t-1)} + W_e e^{(t)} + b_1)$$

σ 100



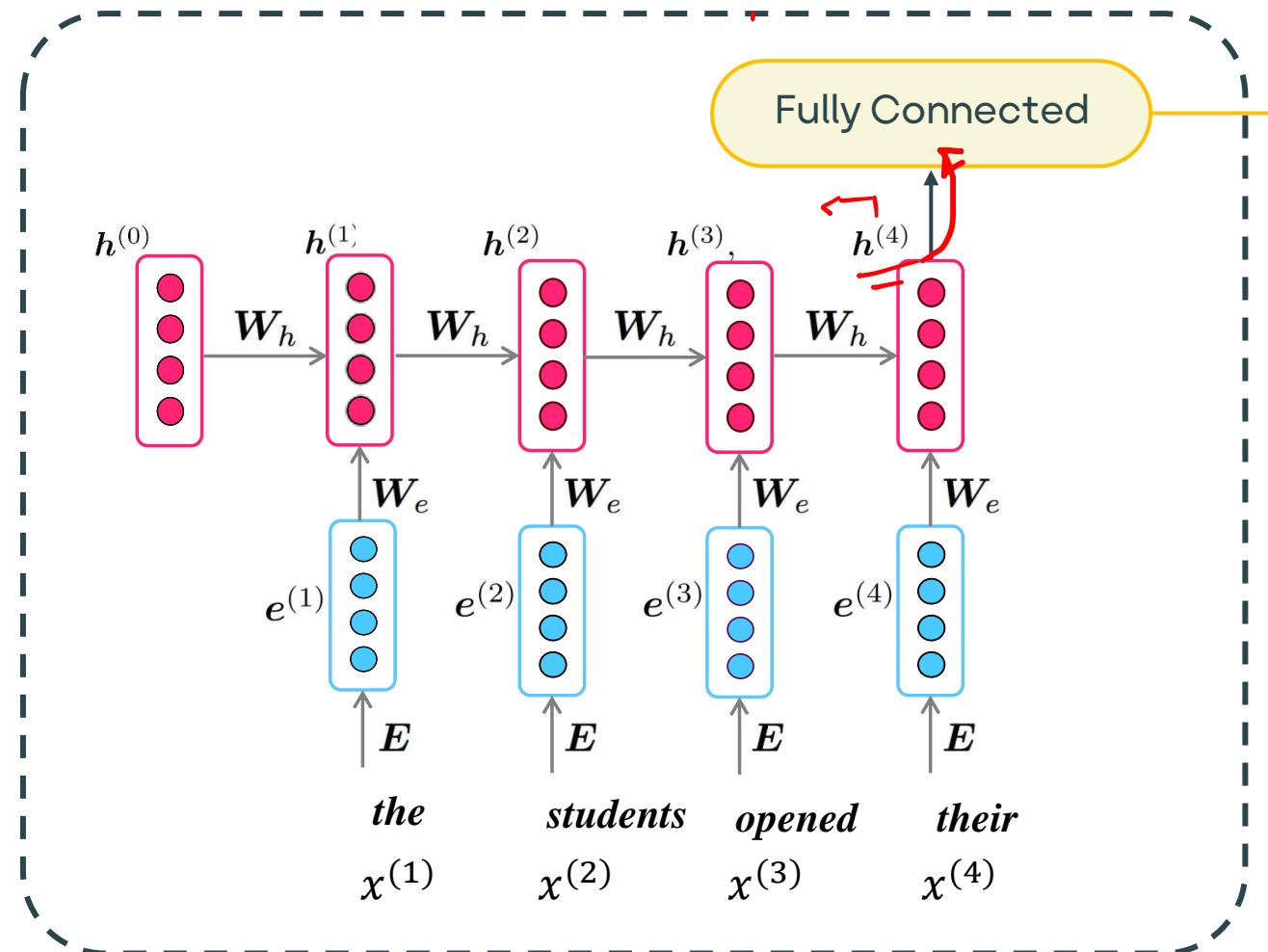
اوکی. حالا خروجی چی؟ بردار hidden state که نون و آب نمیشه برا ما!



100

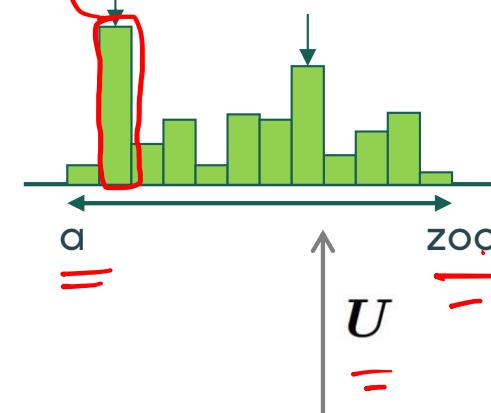
اضافه کردن واحد پردازشی به آخرین بردار h

language modeling \rightarrow Softmax



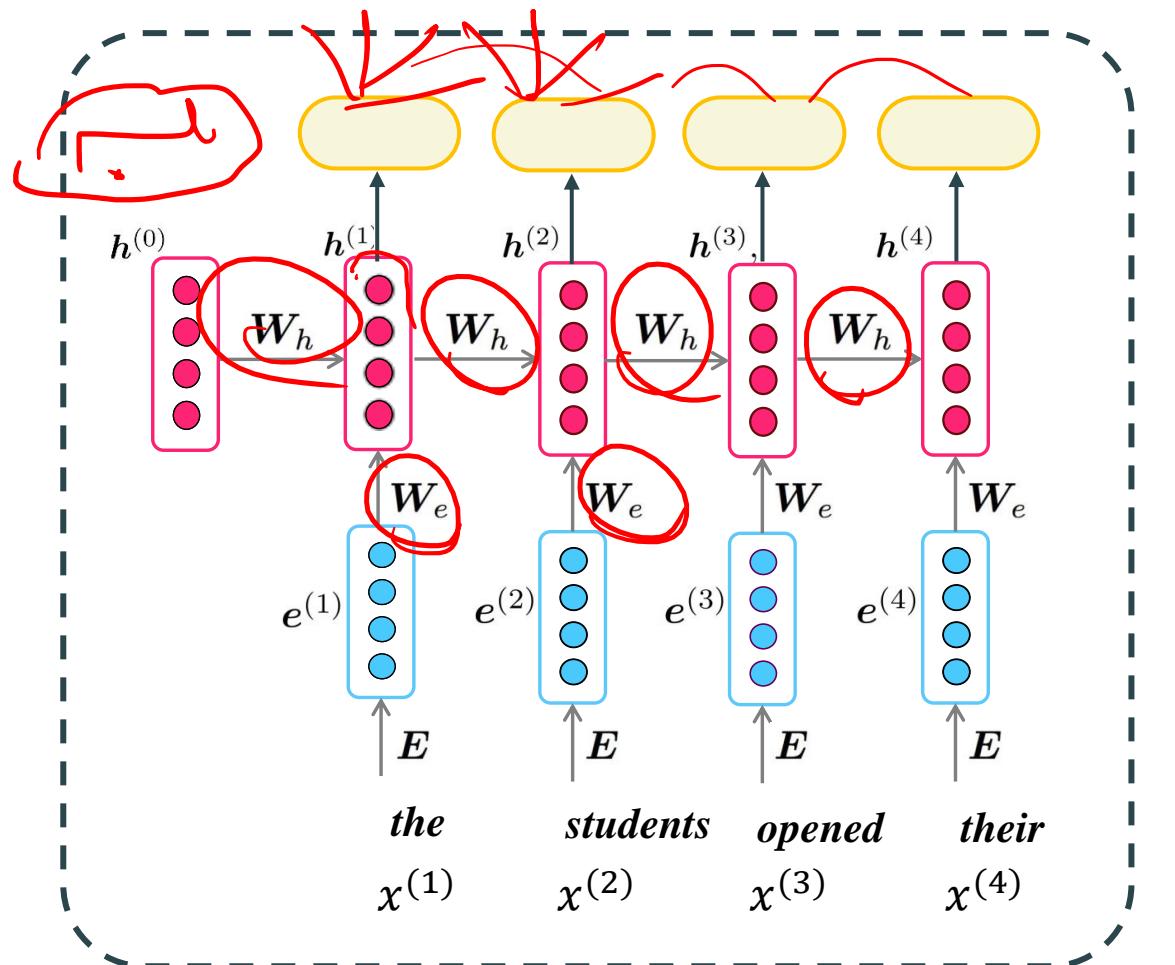
نهایت زان را

$\hat{y}^{(4)} = P(x^{(5)} | \text{the students opened their})$



(
نسمِ عصبی مخواهی)

نکته: این واحد پردازشی میتواند به تمامی h ها وصل شود.



کجا کاربرد داره این کار؟



کلمات رو بخوایم به یه سری **Category** مشخص تقسیم کنیم.

اما فرآیند Training به چه شکل است ؟

محاسبه Loss

دستورات جمع سر

گام ۱: جمع آوری یک corpus بزرگ از نوشه ها که تشکیل شده از کلمات x_1 تا x_t

گام ۲: ورودی کلمات به مدل RNN و محاسبه y_t در هر گام زمانی (احتمالا کلمات)

کلمه
کار

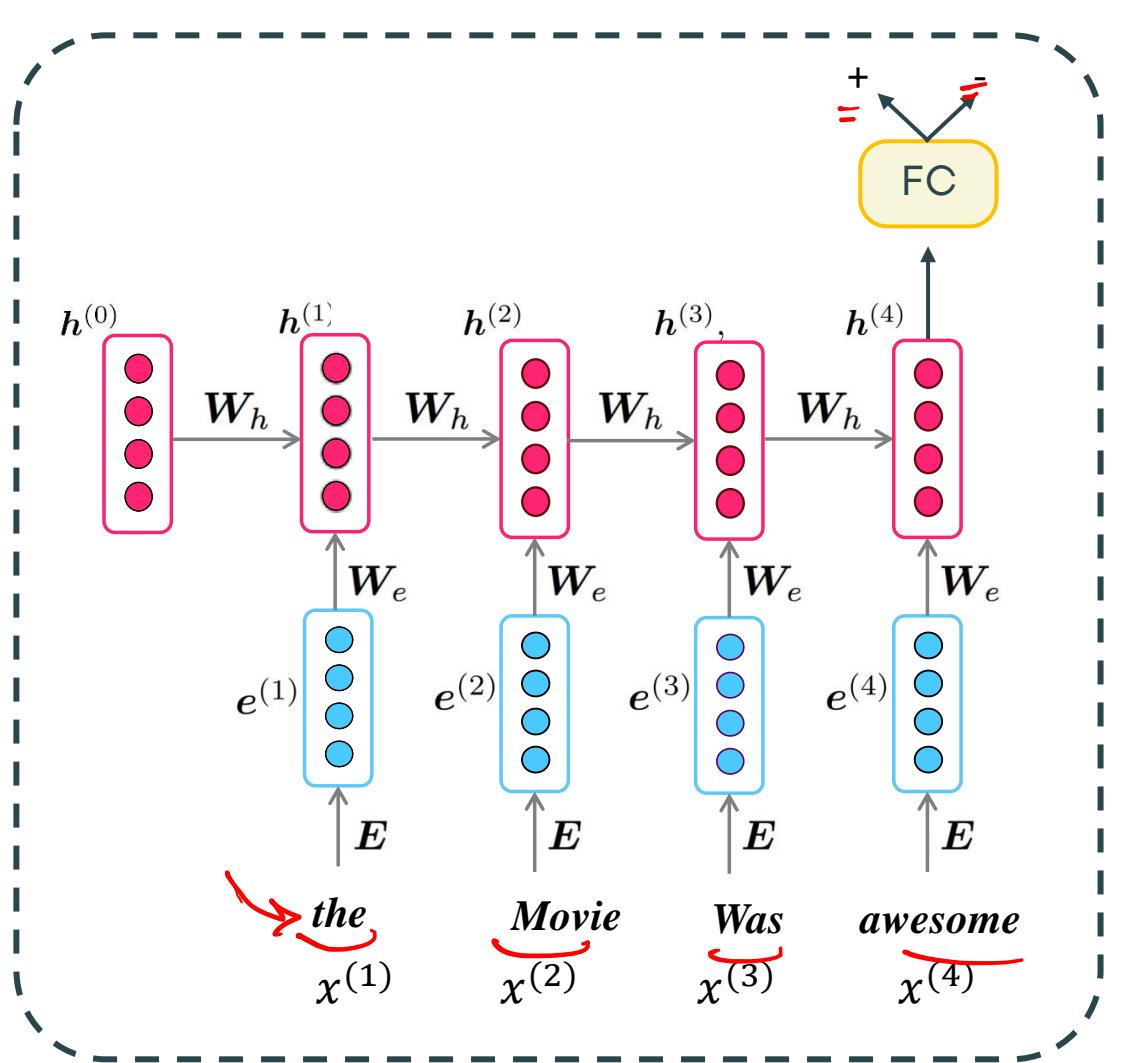
گام ۳: محاسبه Loss بین توزیع y_t و y_{t+1} واقعی

Batch

گام ۴: میانگین گیری Loss داده ها



Binary Classification



categorical cross entropy

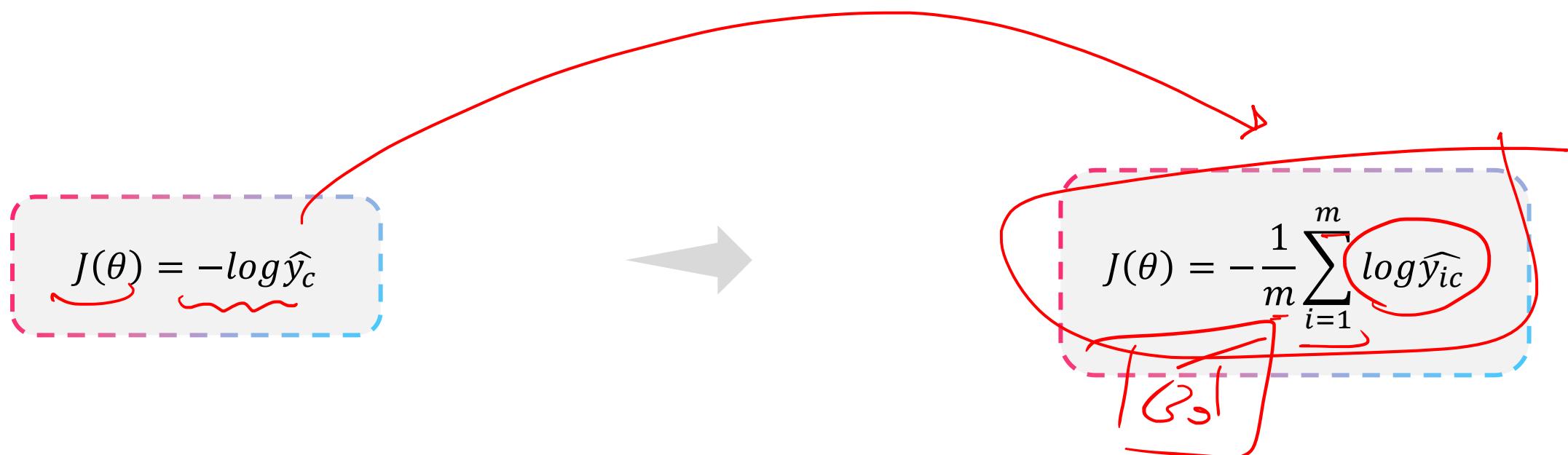
$y = [0, 1]$

$\hat{y} = [0.3, 0.7]$

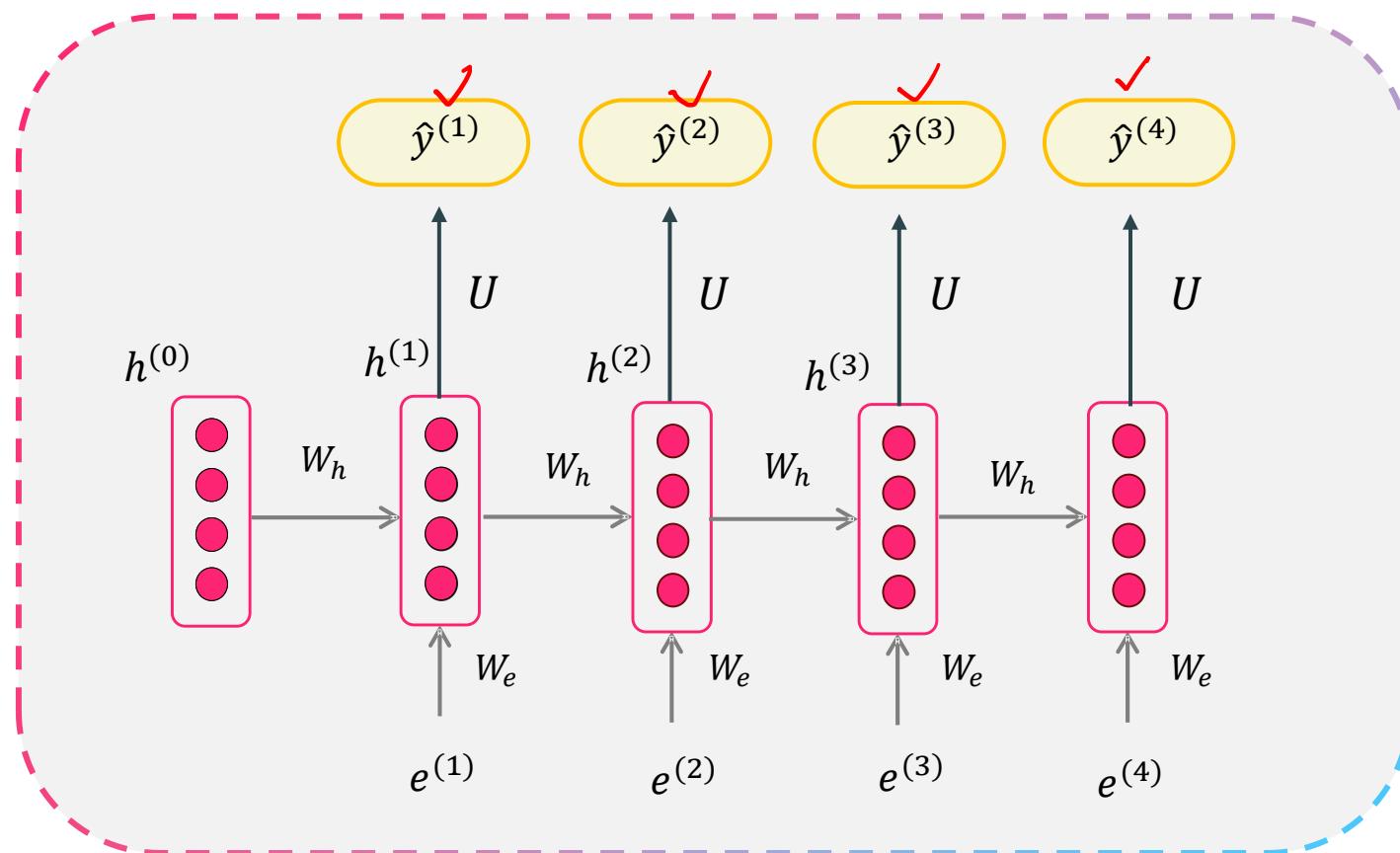
$$J(\theta) = - \sum_{i=0}^1 y[i] \log \hat{y}[i] = -\log \hat{y}_c = -\log(0.7)$$

$- [0 \log 0.3 + 1 \times \log 0.7]$
 $= -\log 0.7$

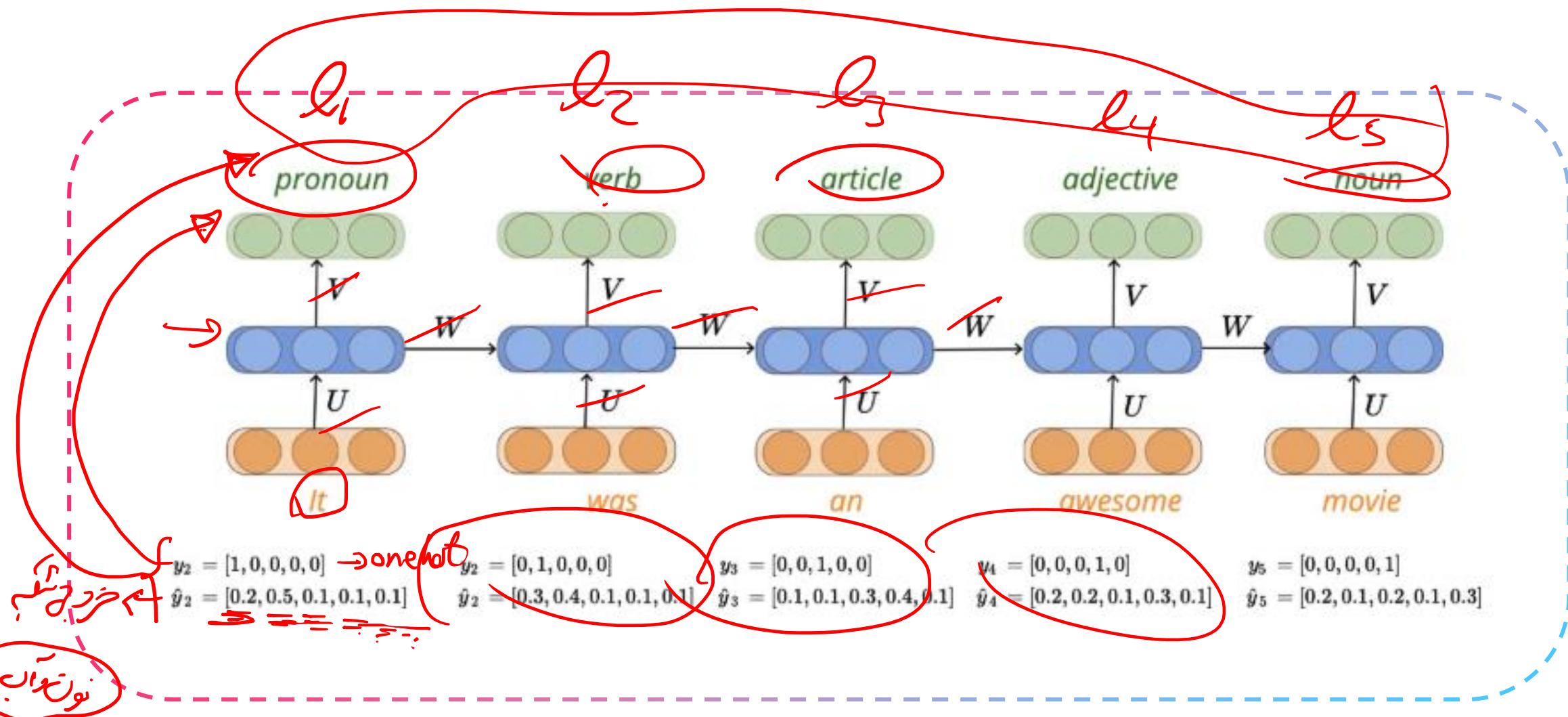
Loss اصلی : یه میانگین روی تمام داده ها



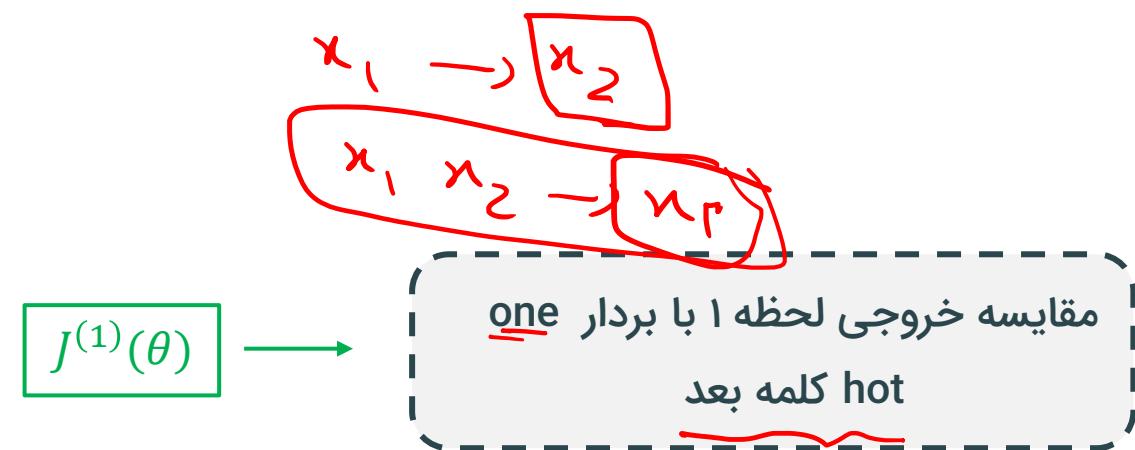
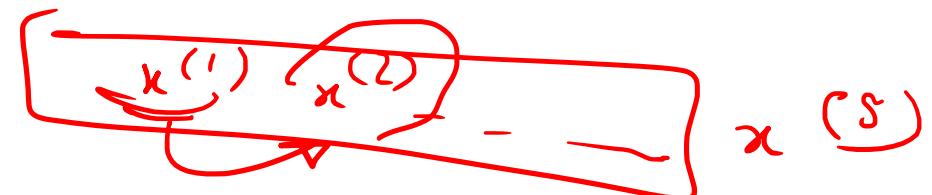
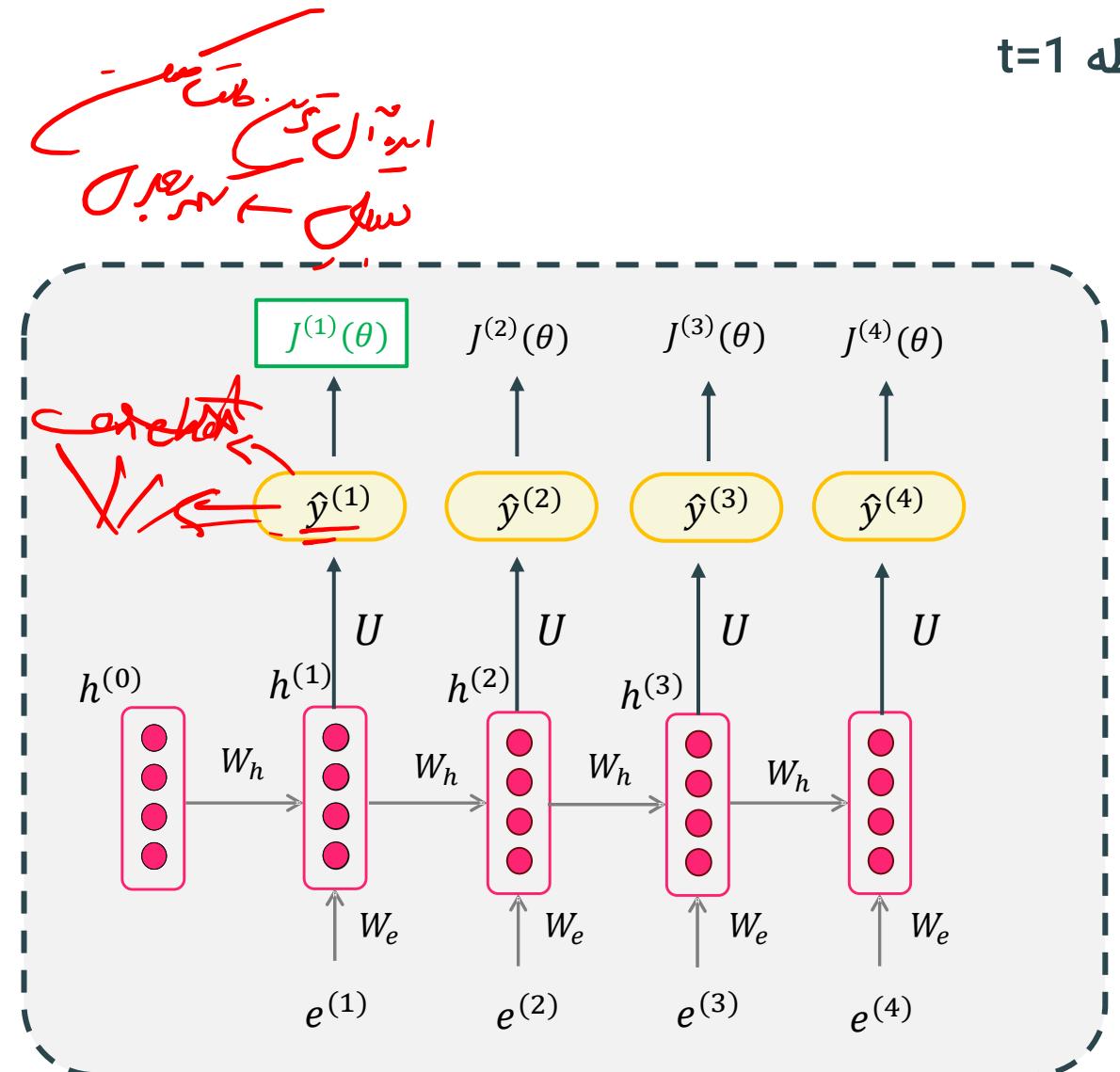
اگر برای هر گام زمانی یک خروجی داشتیم چه؟



ابتدا یک شکل برای درک بهتر

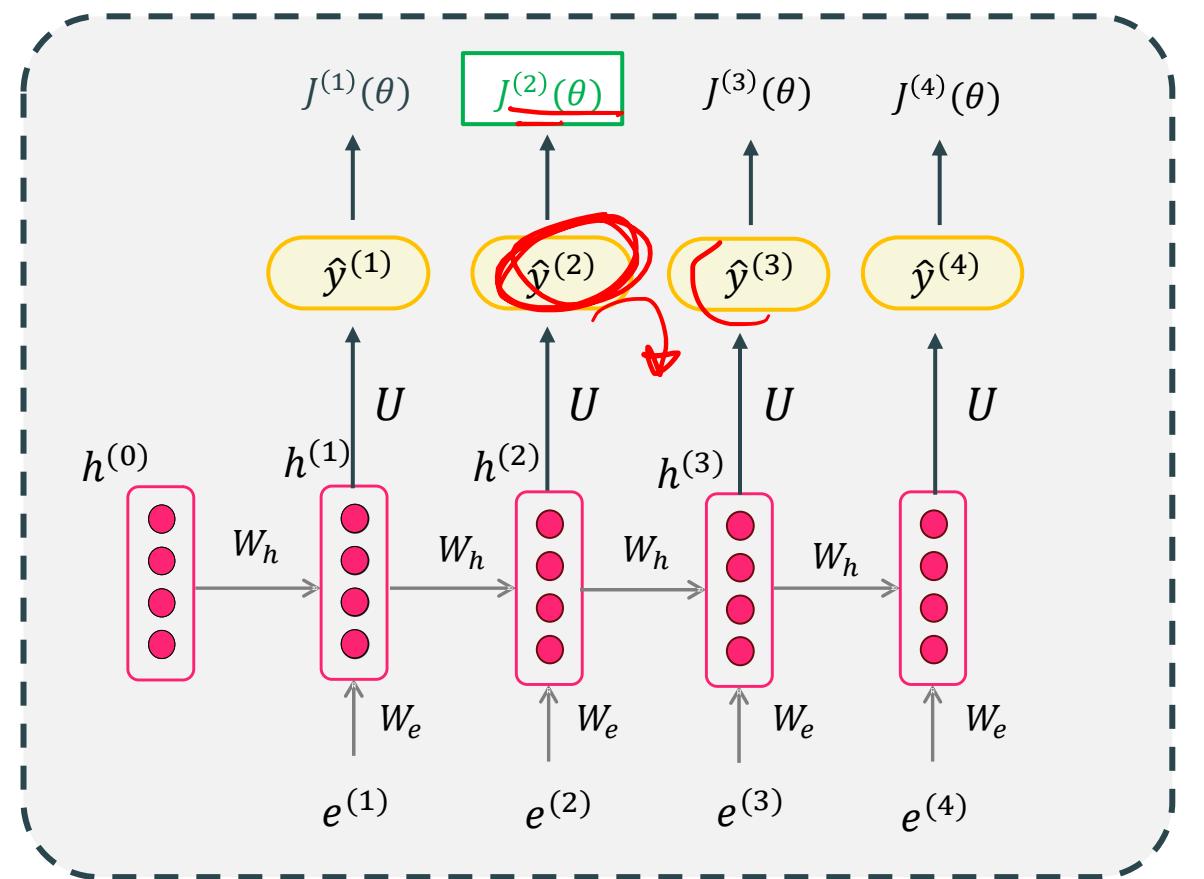


در لحظه 1



چرا سلسله خطون

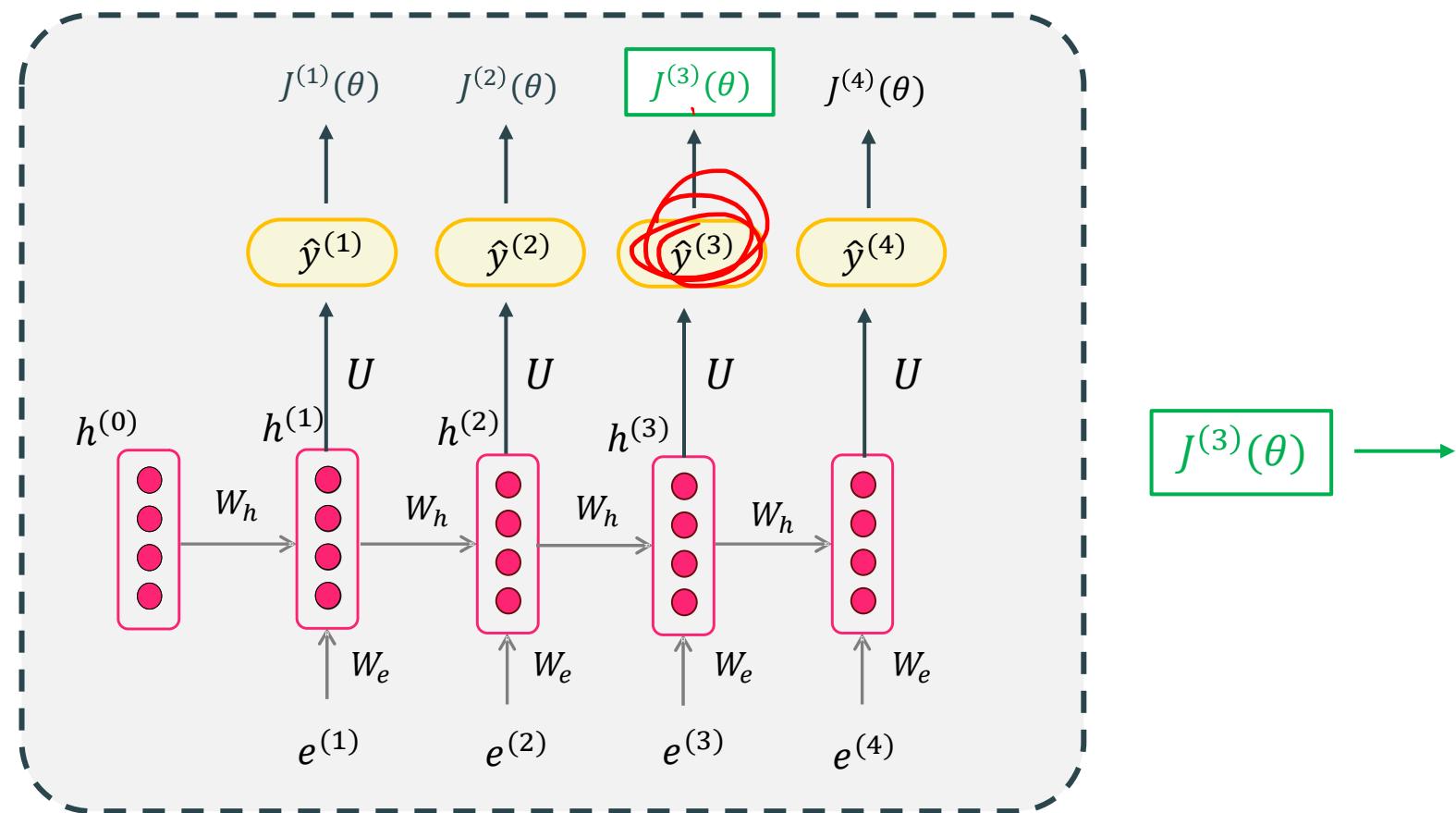
در لحظه 2



$J^{(2)}(\theta)$

مقایسه خروجی لحظه 2 با بردار کلمه بعد hot

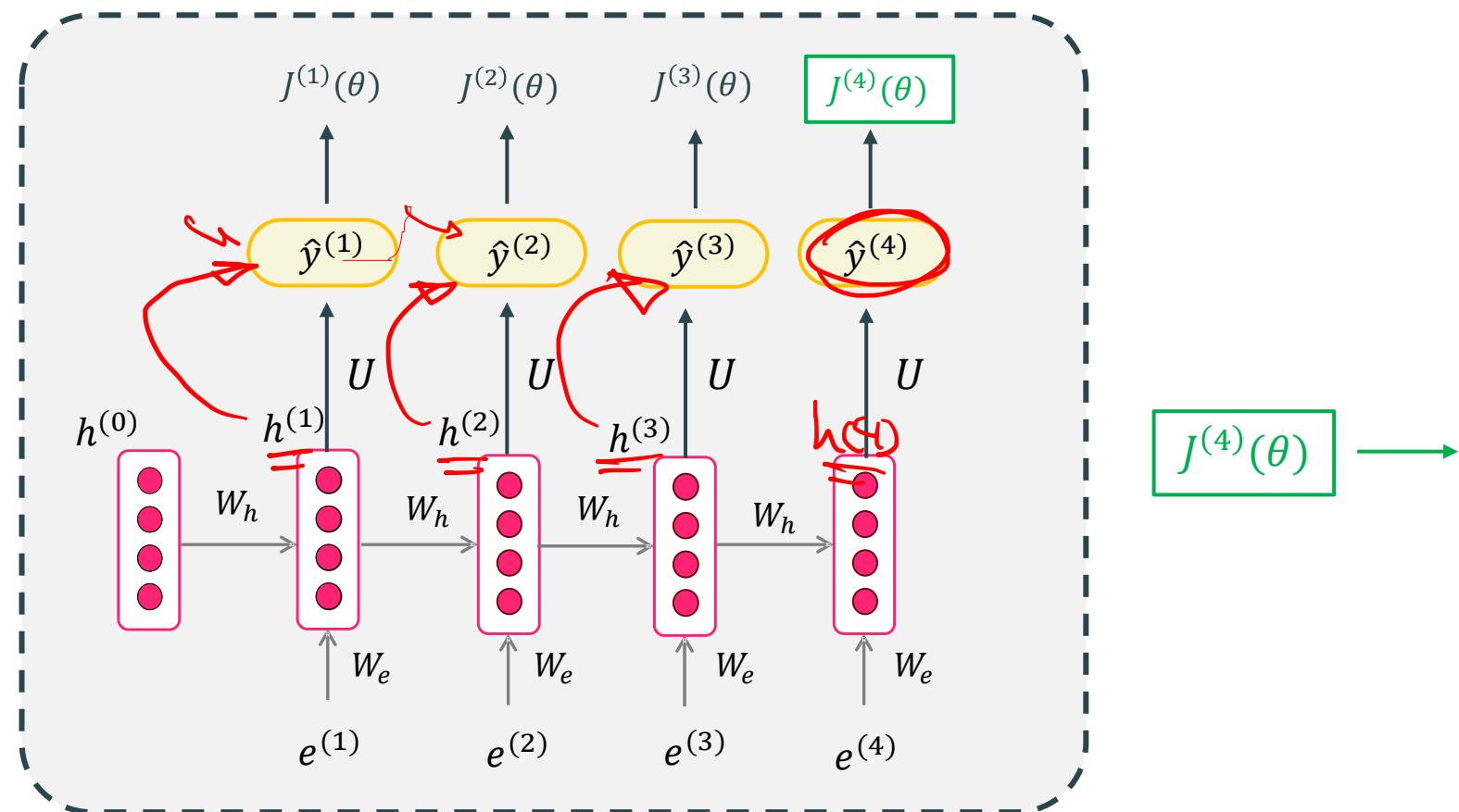
در لحظه 3



$J^{(3)}(\theta)$

مقایسه خروجی لحظه ۳ با بردار کلمه بعد hot

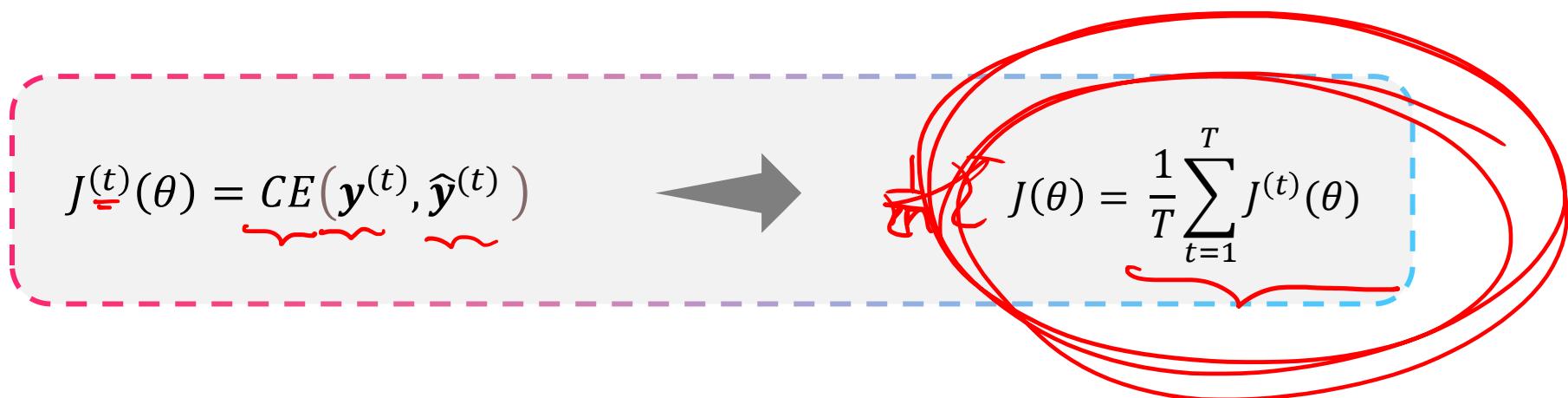
در لحظه $t=4$



$$J^{(4)}(\theta)$$

مقایسه خروجی لحظه ۴ با بردار کلمه بعد
کلمه hot

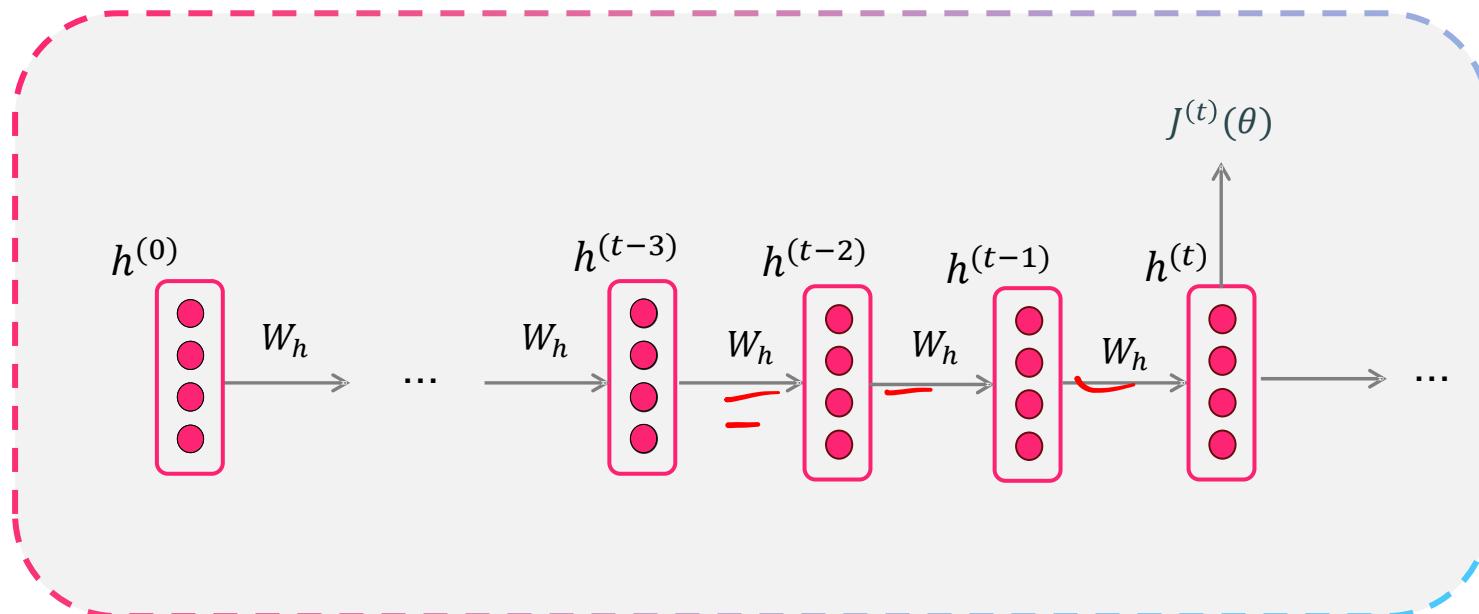
بنابراین Loss برای یک داده به شکل زیر است



اوکی. ولی ما به مشتق Loss نیاز داریم !

$$\theta^+ = \theta^- - \alpha \frac{\partial J}{\partial \theta}$$

حالا مشتق لاس نسبت به وزن ها چی میشه ؟



$$J(\theta) = \frac{1}{T} \sum_{t=1}^T J^{(t)}(\theta)$$

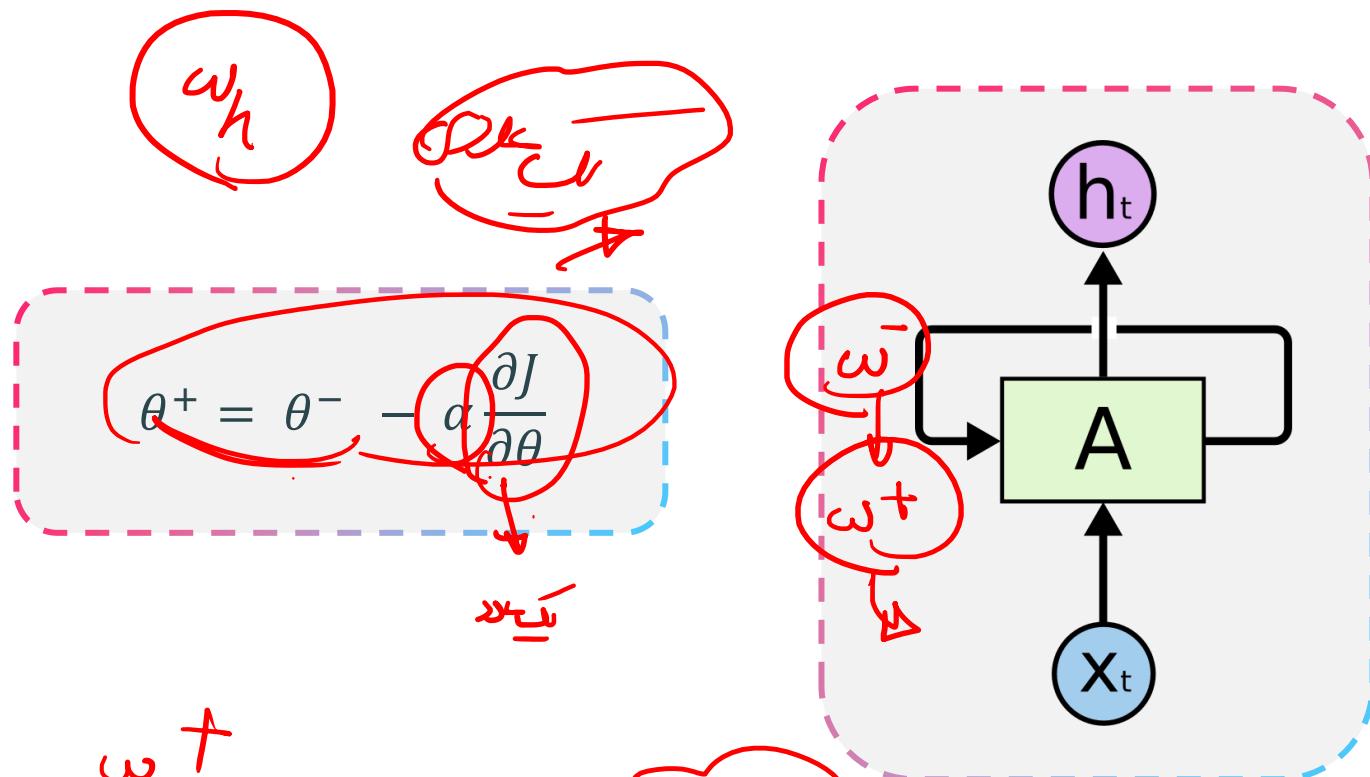
$\xrightarrow{\quad}$

$$\frac{\partial J^{(t)}}{\partial W_h} = \sum_{i=1}^t \left. \frac{\partial J^{(t)}}{\partial W_h} \right|_{(i)}$$

اون مشتق رو محاسبه کن کامل و یک بار وزن ها رو آپدیت کن !

$$J(\theta) = \frac{1}{T} \sum_{t=1}^T J^{(t)}(\theta)$$

متوسط



$$w_h^+ = w_h^- - \alpha \frac{\partial J}{\partial w_h}$$

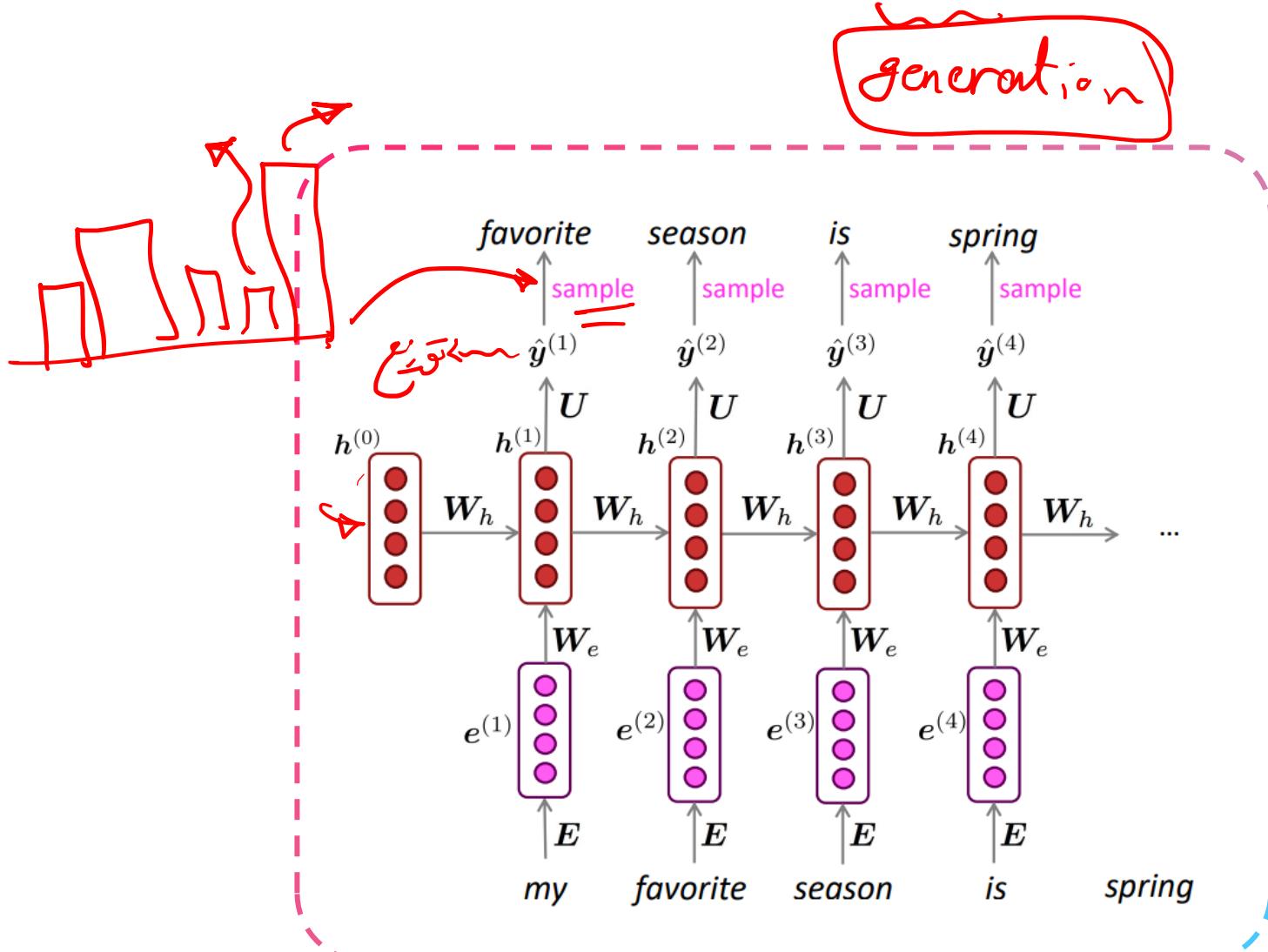
در عمل : Truncated BPTT



خیلی اوقات جهت این که محاسبات زیاد نشه تا 2^0 گام زمانی مشتق ها محاسبه میشه.



تولید متن به کمک RNN های ساده



نمونه متن تولیدی (سخنان اوباما)

The United States will step up to the cost of a new challenges of the American people that will share the fact that we created the problem. They were attacked and so that they have to say that all the task of the final days of war that I will not be able to get this done.



جمع‌بندی

یک سیستم که کلمه بعدی را پیش‌بینی می‌کند. : Language Model

یک sequence از ورودی با هر طولی را دریافت می‌کند.
از وزن‌های یکسانی در هر گام زمانی استفاده می‌کند.
بسته به نیاز می‌توانیم در هر مرحله خروجی داشته باشیم.

خوب تر است

RNN : خانواده از شبکه‌های عصبی که

==



$RNN \neq LM$



RNN ها گزینه مناسبی برای LM ها هستند.



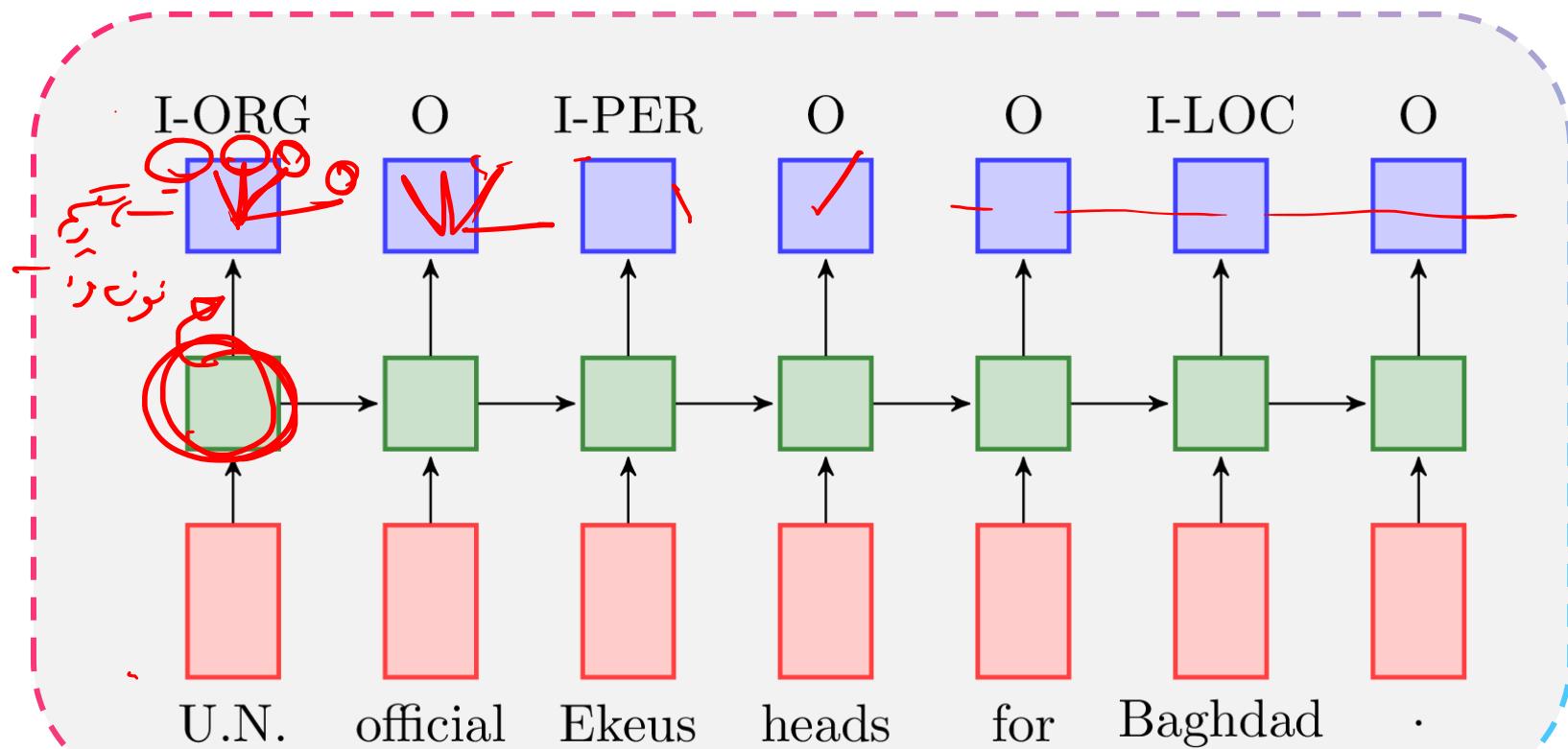
چطور از RNN ها در سایر کاربردها استفاده کنیم ؟

Named Entity Recognition

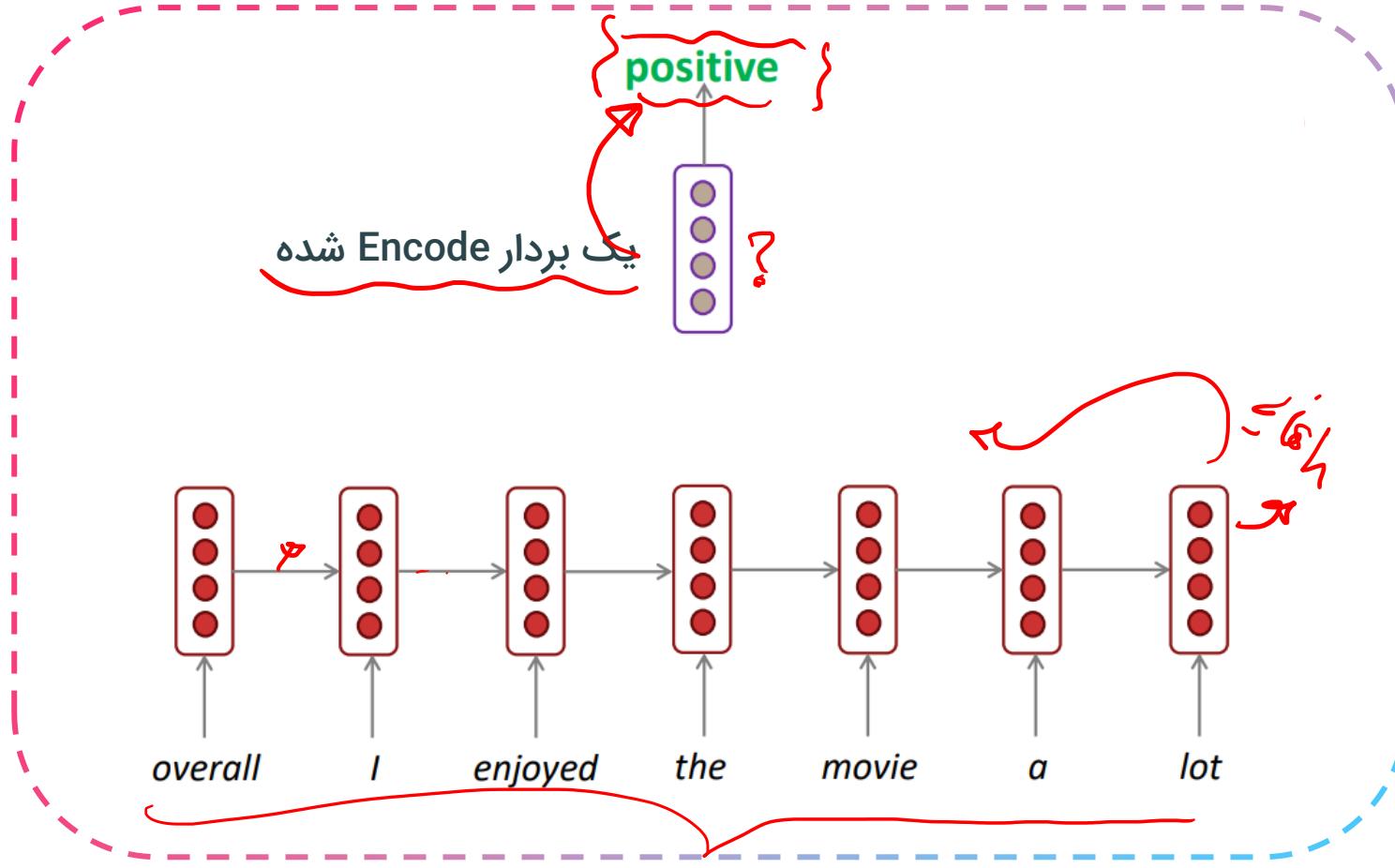
Person p Loc l Org o Event e Date d Other z

Barack Hussein Obama II (born August 4, 1961) is an American attorney and politician who served as the 44th President of the United States from January 20, 2009, to January 20, 2017. A member of the Democratic Party, he was the first African American to serve as president. He was previously a United States Senator from Illinois and a member of the Illinois State Senate.

استفاده از RNN در NER

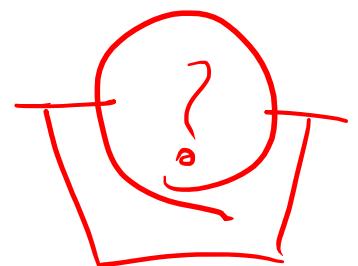
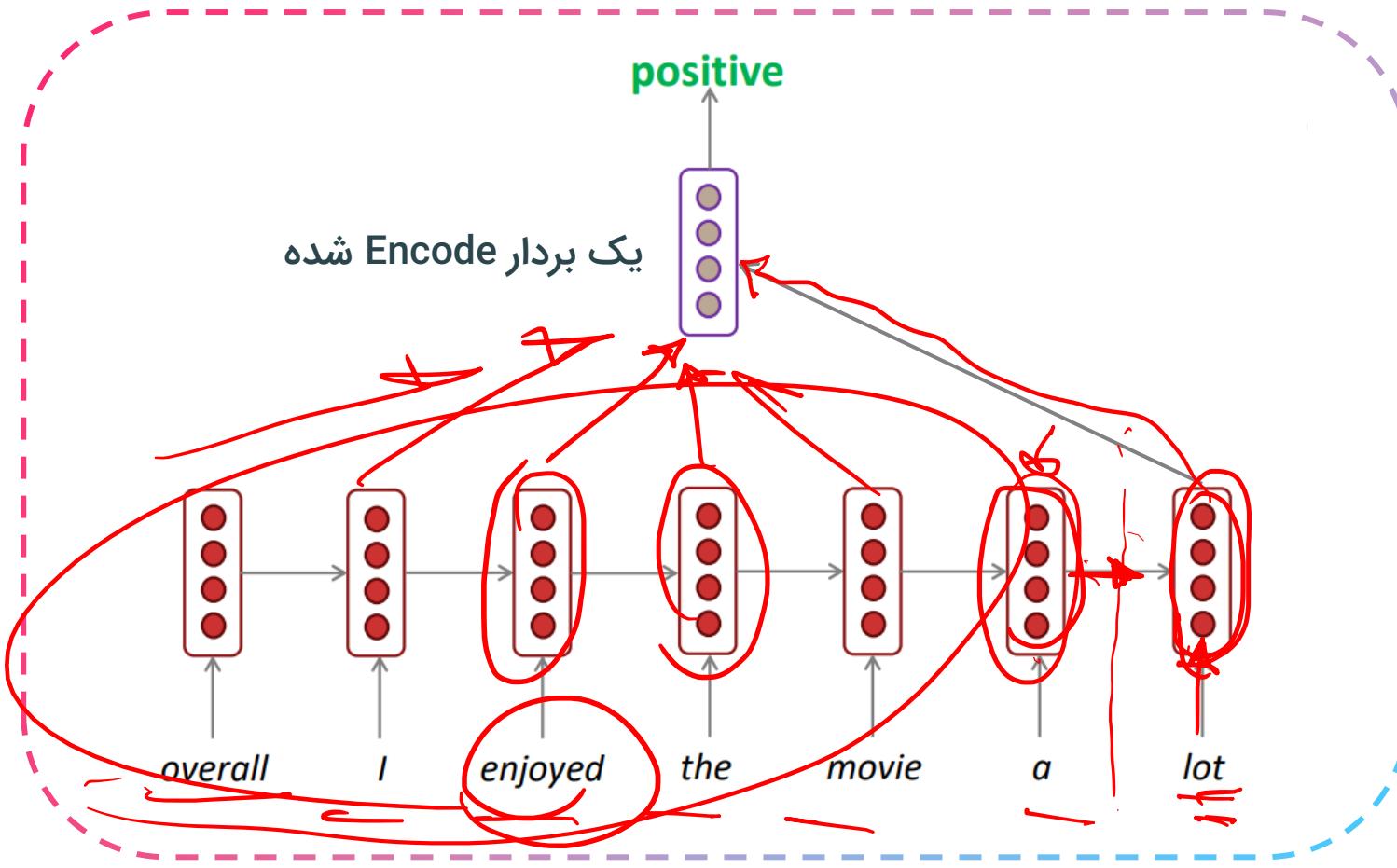


Sentiment Classification (تحليل احساسات)

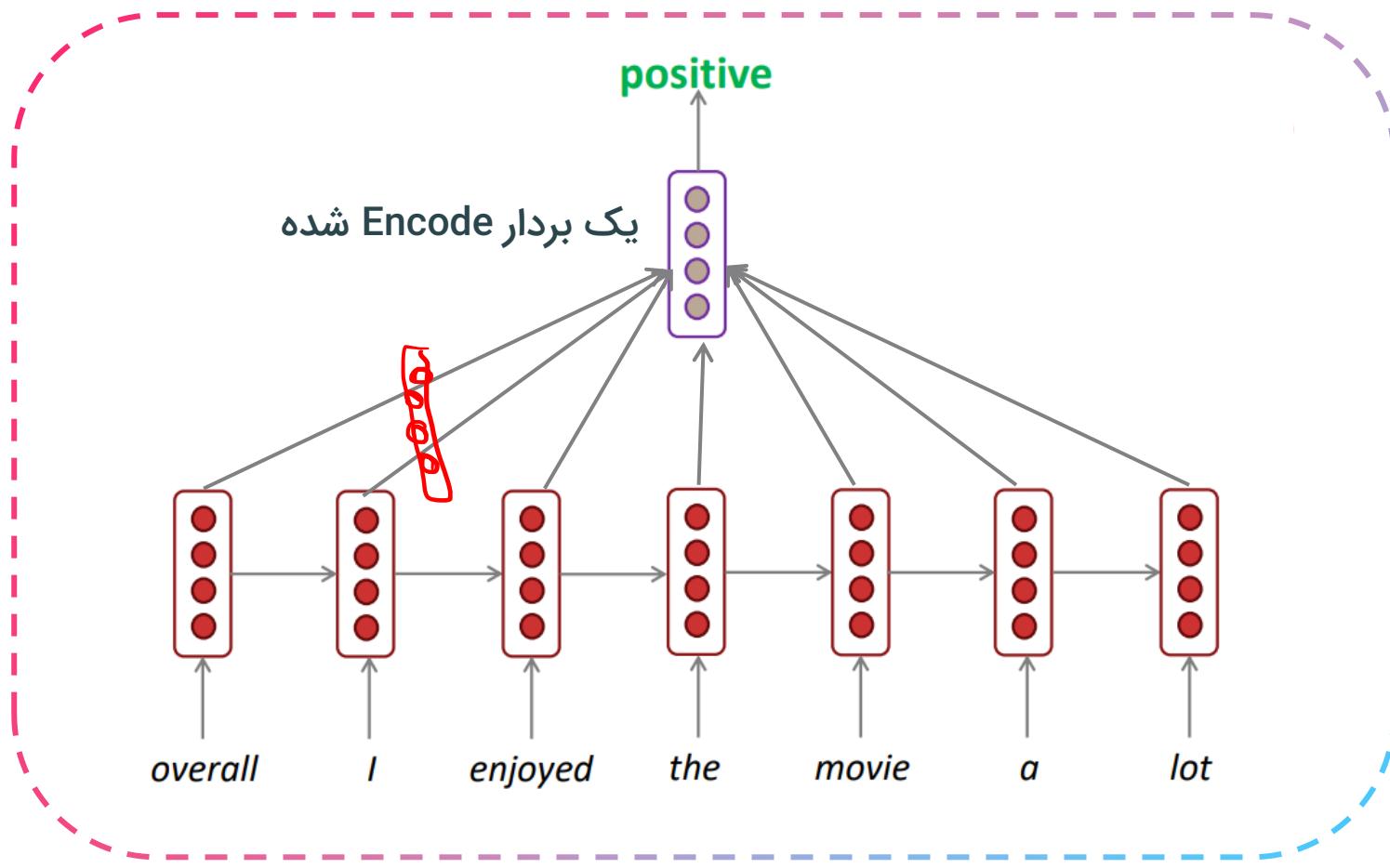


پیشنهادتان برای استخراج بردار Encode شده چیست ؟

گزینه ۱ : استفاده از آخرین hidden state



گزینه ۲ : استفاده از میانگین یا ماکزیمم روی المان های hidden state



تا الان :

بخش سوم

- ❖ دو مثال ساده از این بزرگوار



توضیح دقیق شبکه های بازگشتی

بخش دوم

- ❖ توضیح دقیق شبکه های بازگشتی

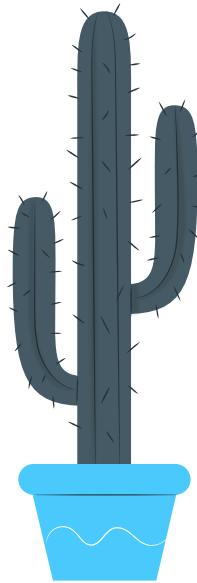
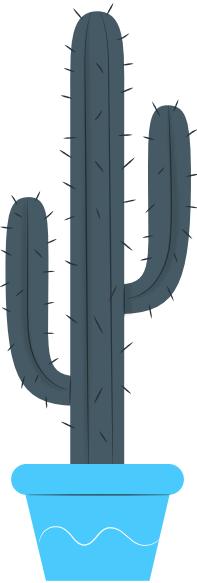


بخش اول

- ❖ کاربرد شبکه های بازگشتی کجاست ؟



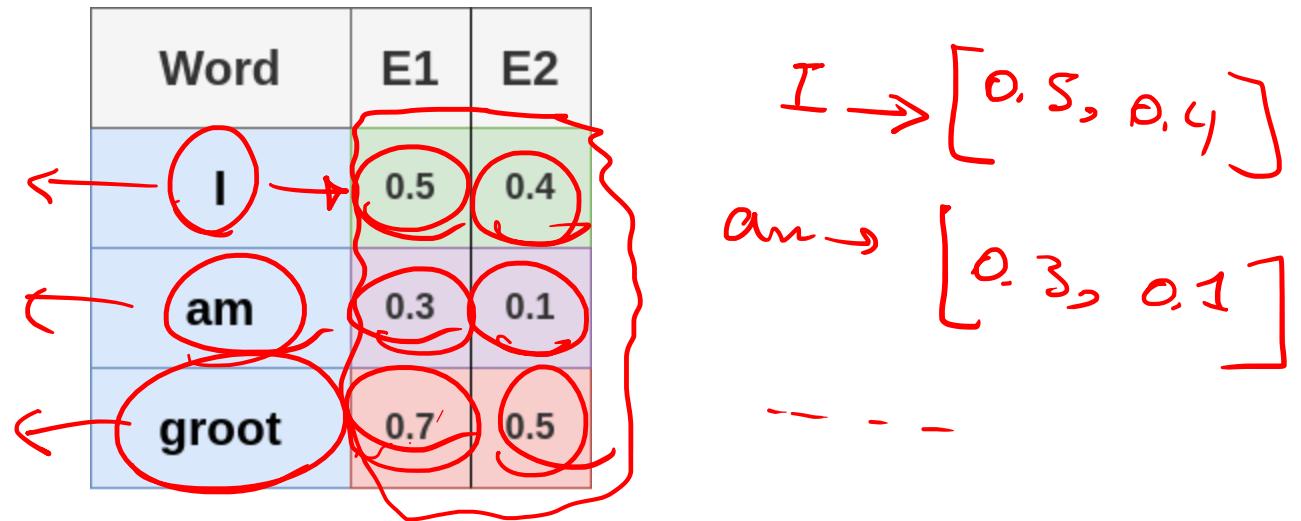
بخش سوم : دو مثال ساده از این بزرگوار



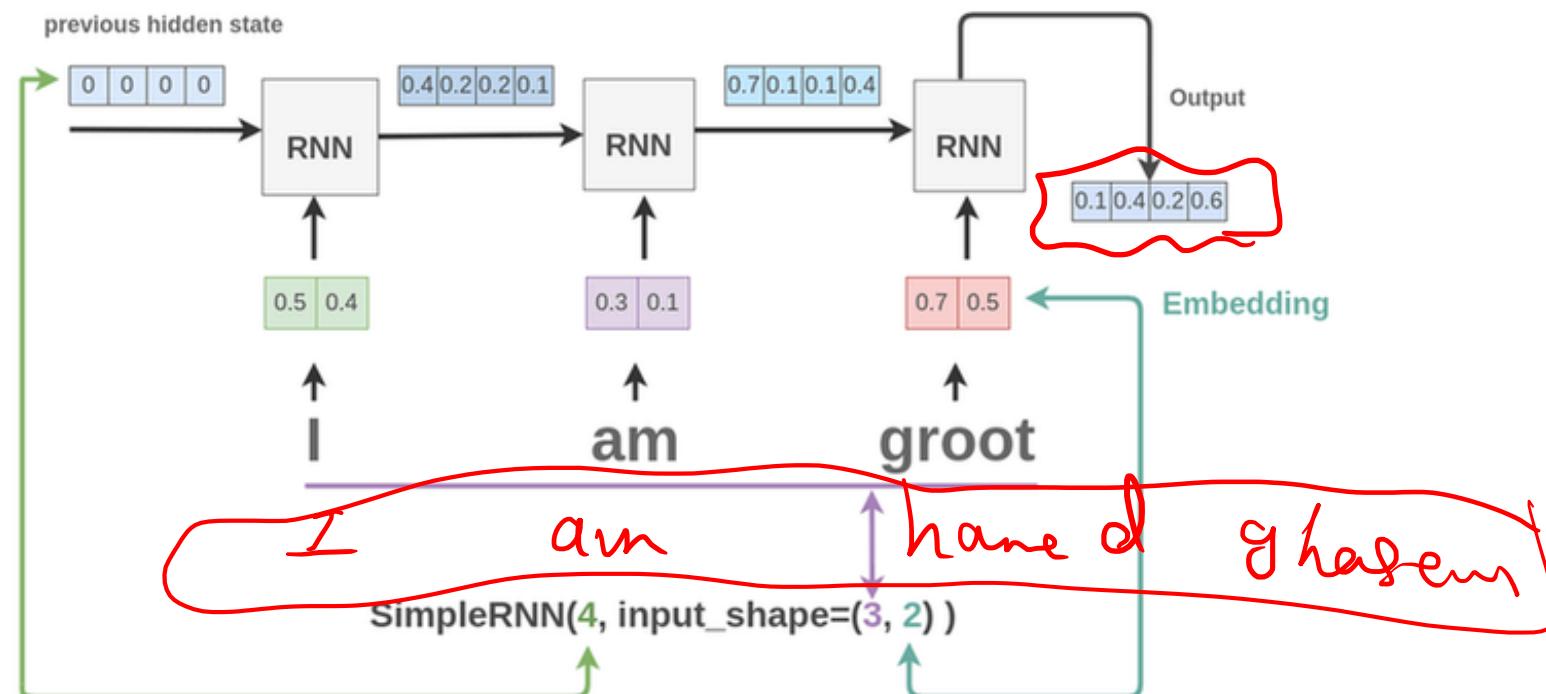
یک مثال آموزشی



فرض: تبدیل کلمات به بردارهای عددی



```
model = Sequential()  
  
model.add(SimpleRNN(4, input_shape=(3, 2)))
```



مثال عددی

۸
3 2, 3 2, 3

ابعاد
x = tf.random.normal((1, 3, 2))
layer = SimpleRNN(4, input_shape=(3, 2))
output = layer(x)
print(output.shape)
(1, 4)

ابعاد خروجی در این حالت برابر (batch_size, outputfeatures) است.

مطمئن هستیم همیشه گام زمانی ۳ است؟

```
model = Sequential()  
model.add(SimpleRNN(4, input_shape=(None, 2)))
```



وقتی **None** میداریم تا **هر میزان** که لازم هست برو جلو

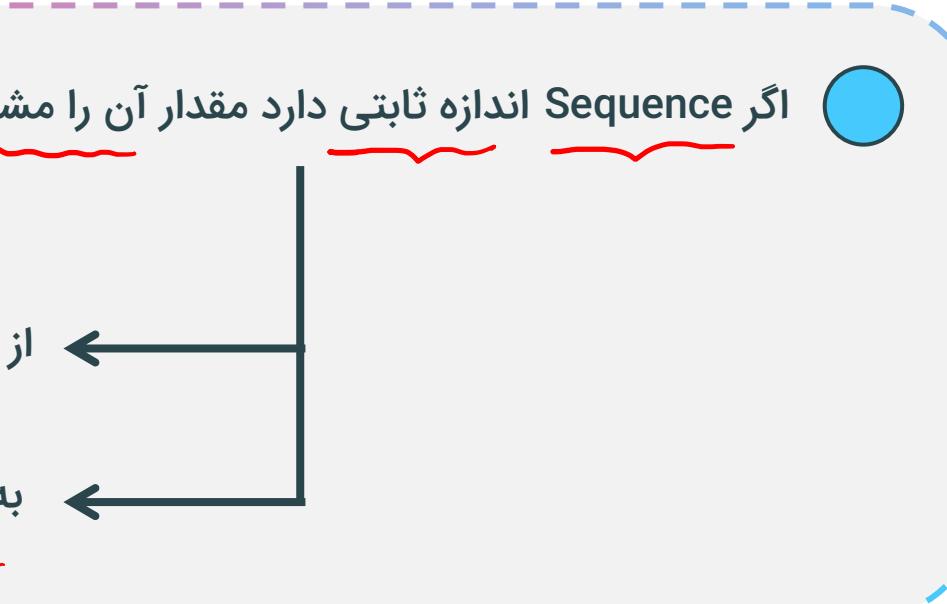
نکته شوله



اگر Sequence اندازه ثابتی دارد مقدار آن را مشخص کنید به دو دلیل :

از نعمت `model.summary()` استفاده کنید.

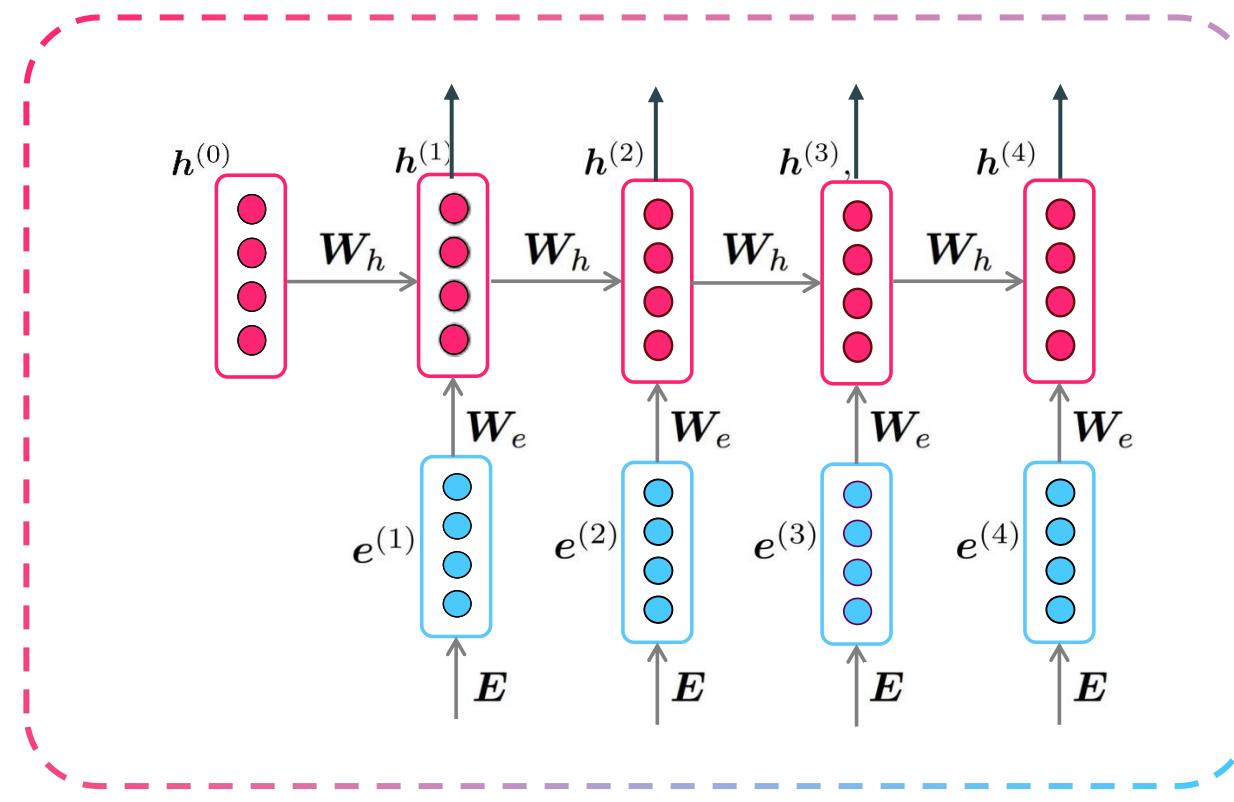
به لحاظ محاسبات نیز بهینه سازی هایی انجام می شود.



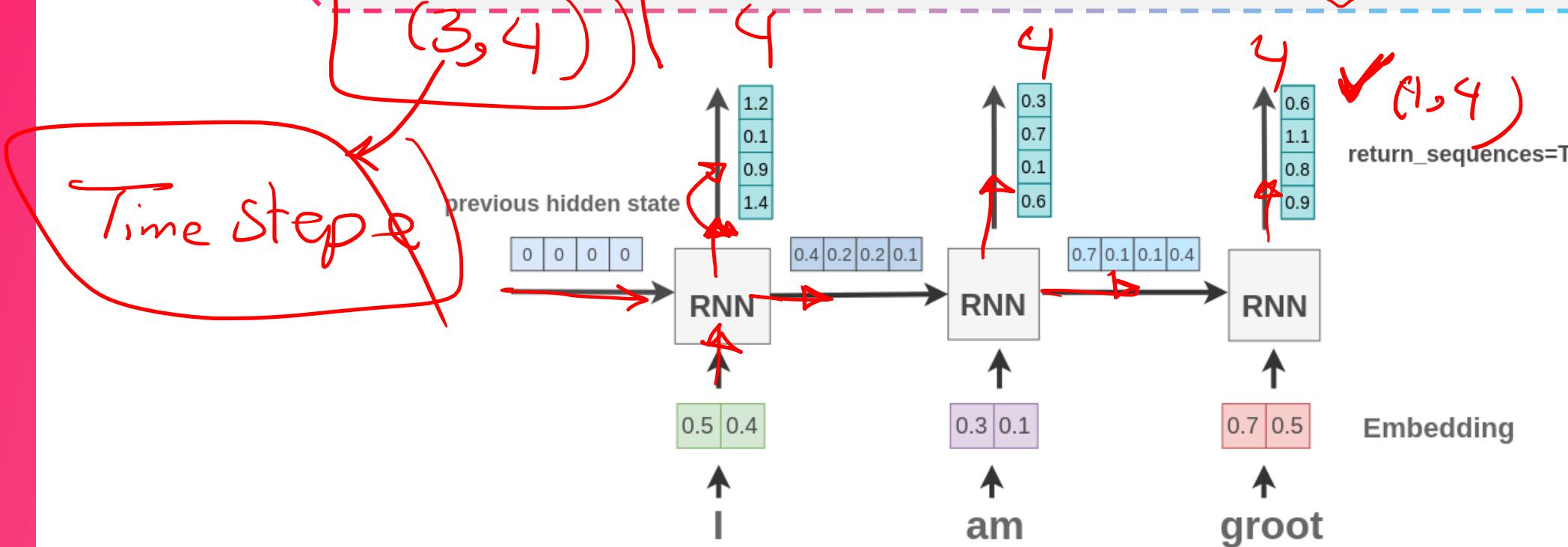
برای طبقه بندی 

```
model = Sequential()  
  
model.add(SimpleRNN(4, input_shape=(3, 2)))  
  
model.add(Dense(1))   
Dense(2, activation='softmax')  
1 Sigma'el
```

اگر همه hidden state ها را خواستیم چه ؟



```
model = Sequential()  
model.add(SimpleRNN(4, input_shape=(3, 2), return_sequences=True))
```



ابعاد خروجی در این حالت برابر `(batch_size, timesteps, output_features)` است.

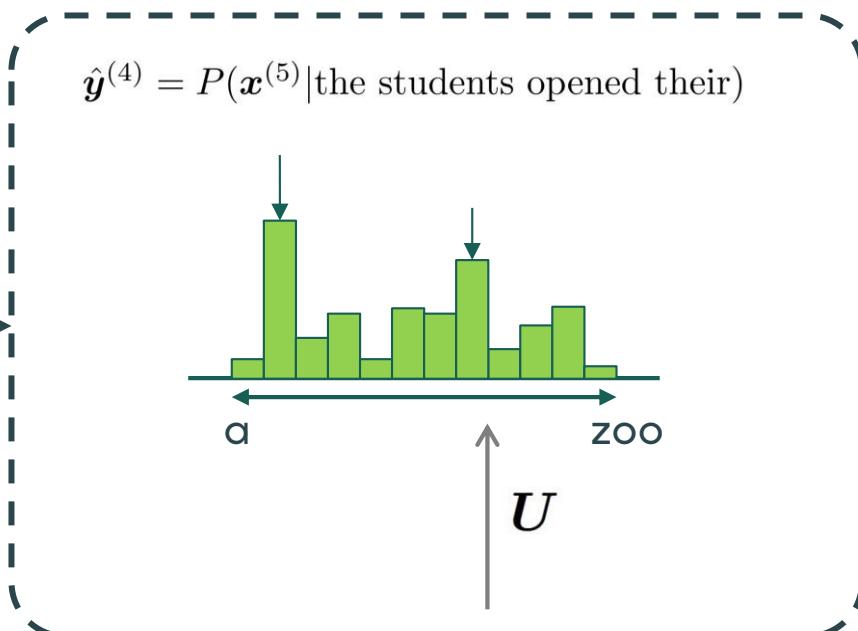
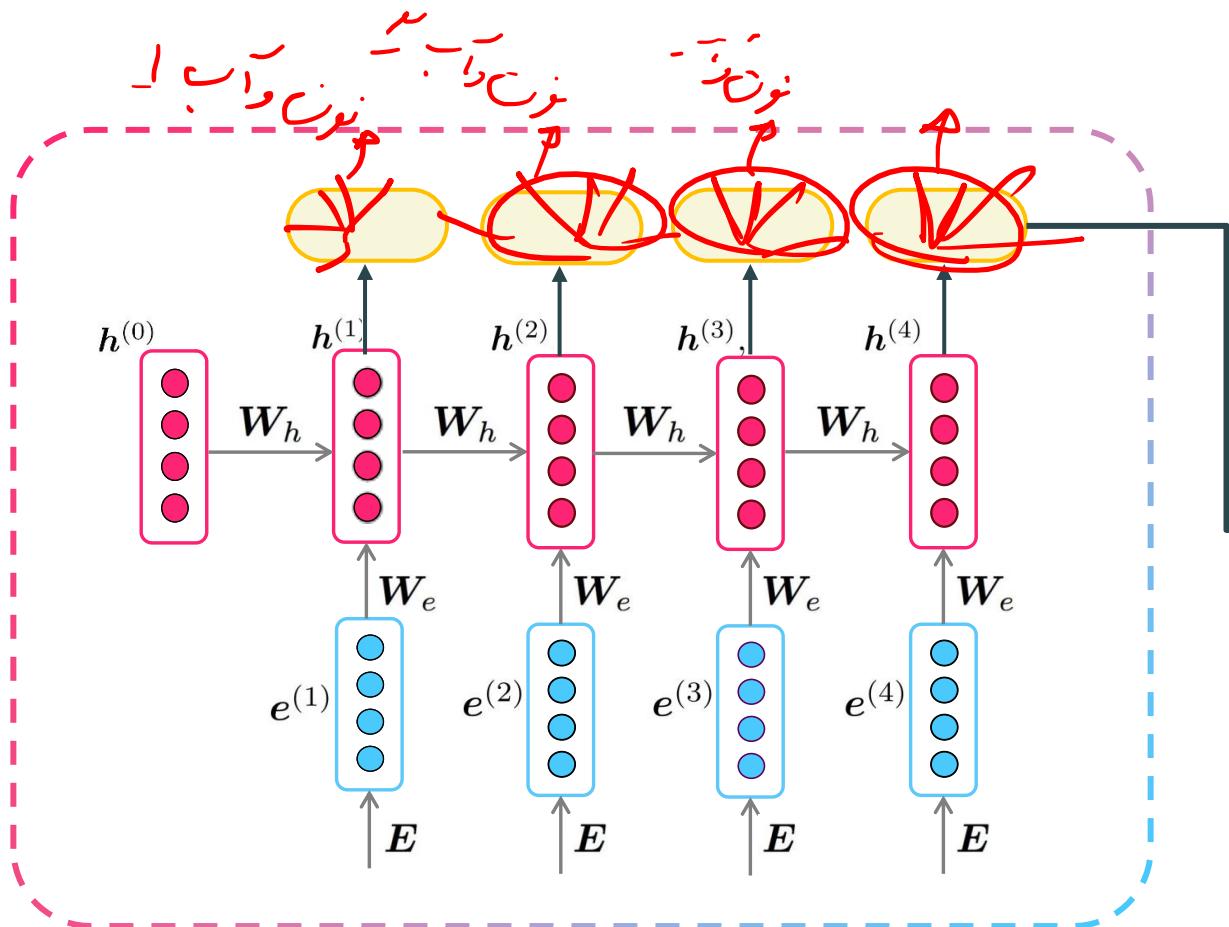


مثال :

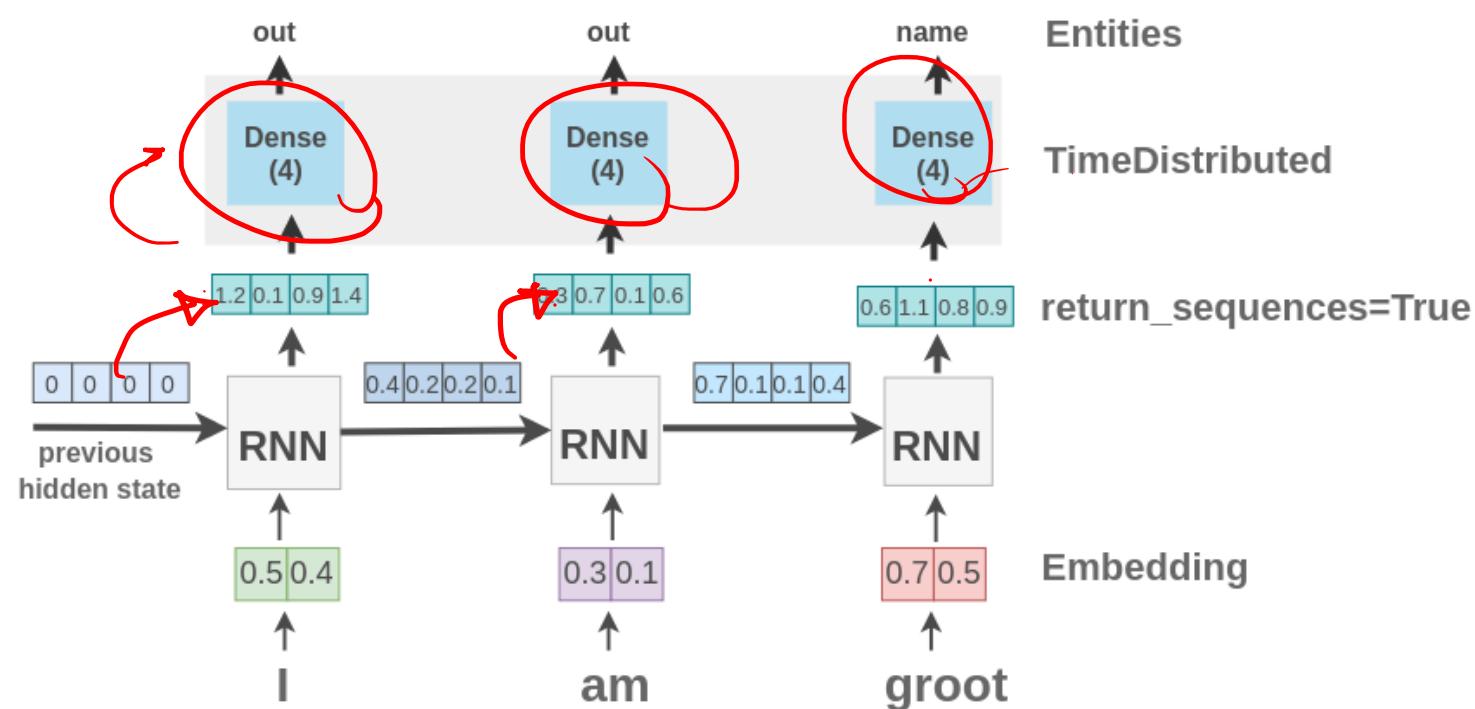
```
import tensorflow as tf
from tensorflow.keras.layers import SimpleRNN

x = tf.random.normal((1, 3, 2))
layer = SimpleRNN(4, input_shape=(3, 2), return_sequences=True)
output = layer(x)
print(output.shape)
# (1, 3, 4)
```

چطور از همه خروجی بگیریم حالا ؟



```
model = Sequential()  
model.add(SimpleRNN(4, input_shape=(3, 2), return_sequences=True))  
model.add(TimeDistributed(Dense(4, activation='softmax')))
```

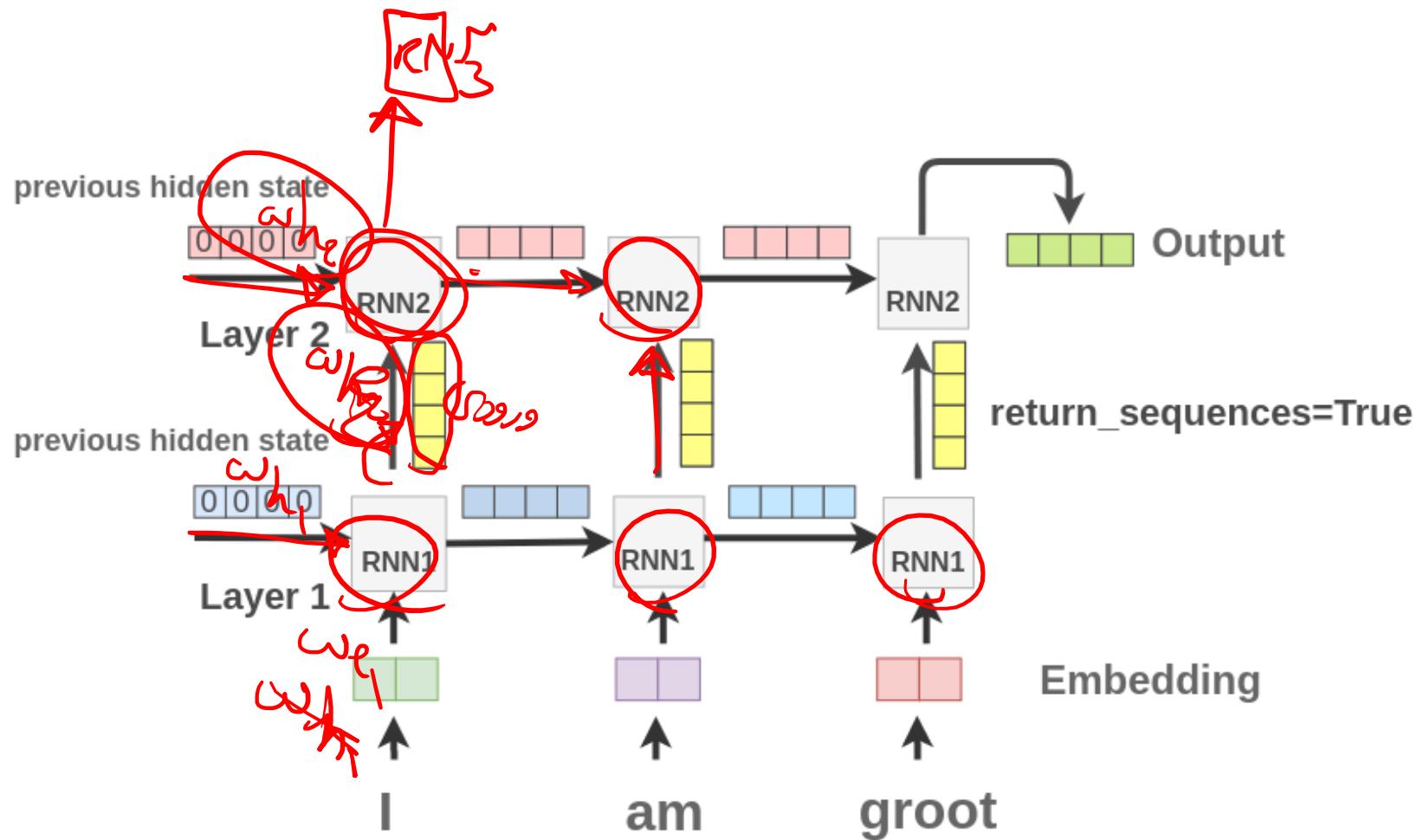


تعداد نرون های لایه آخر به تعداد کلاس هایی است که میخواهیم باشد.

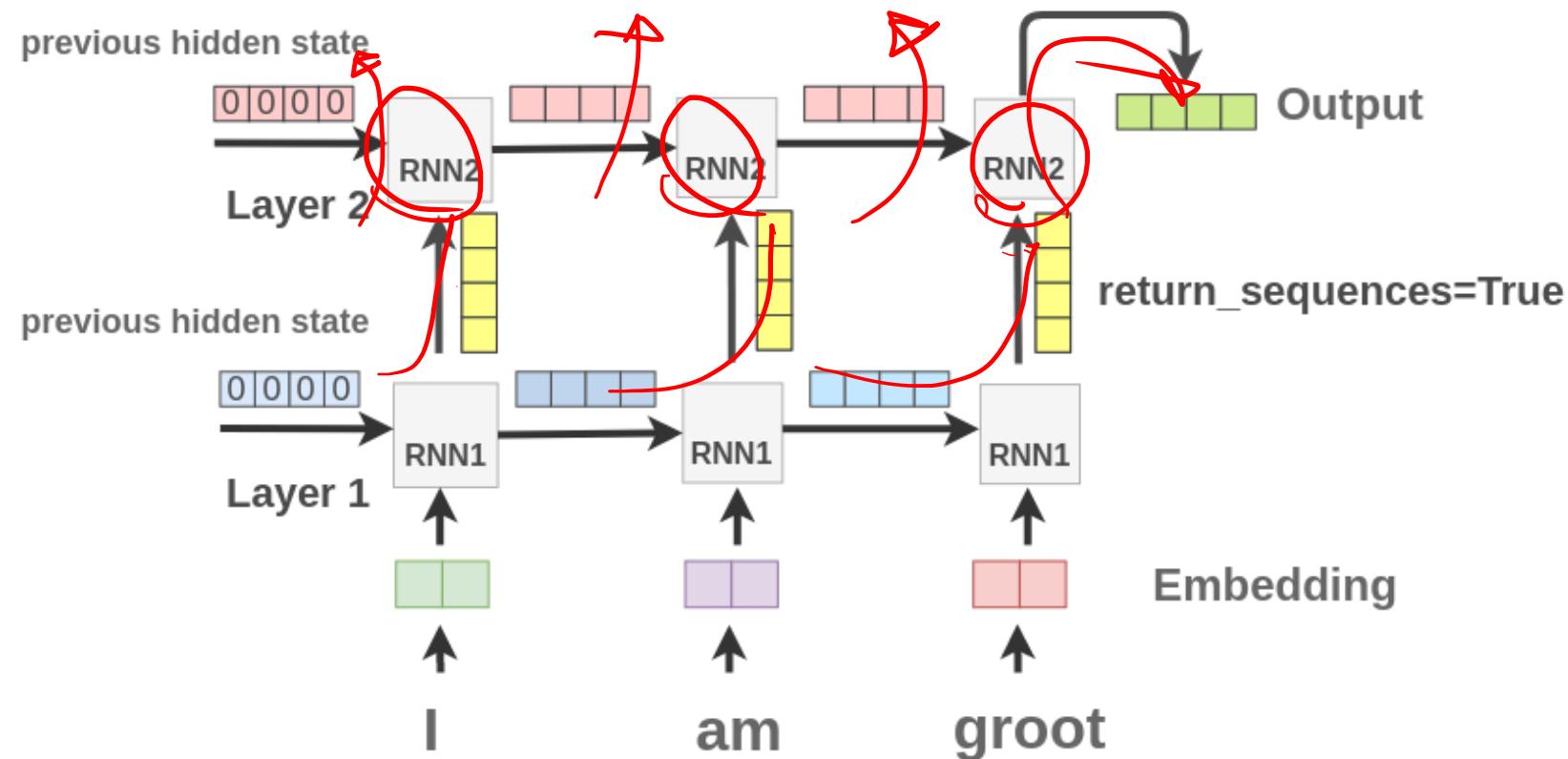


Deep RNN

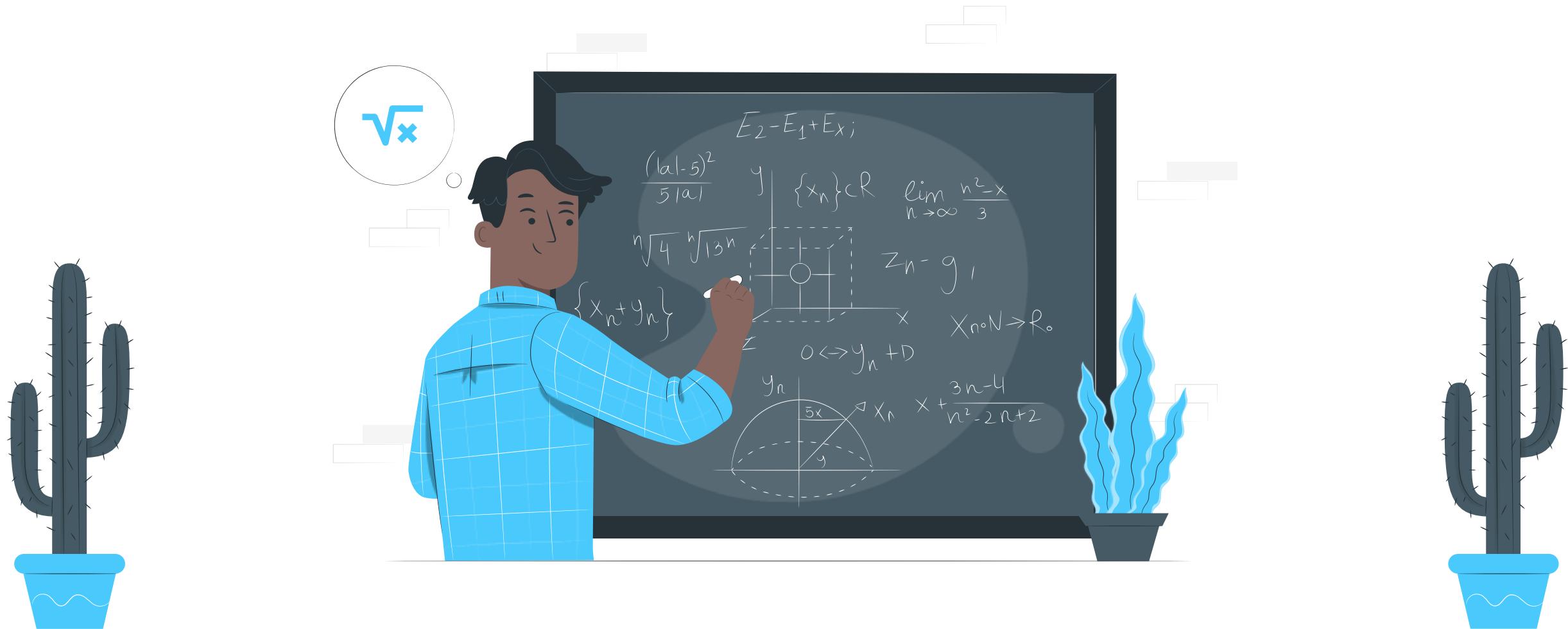
چرا یک لایه ! چند لایه چرا نداشته باشیم ؟



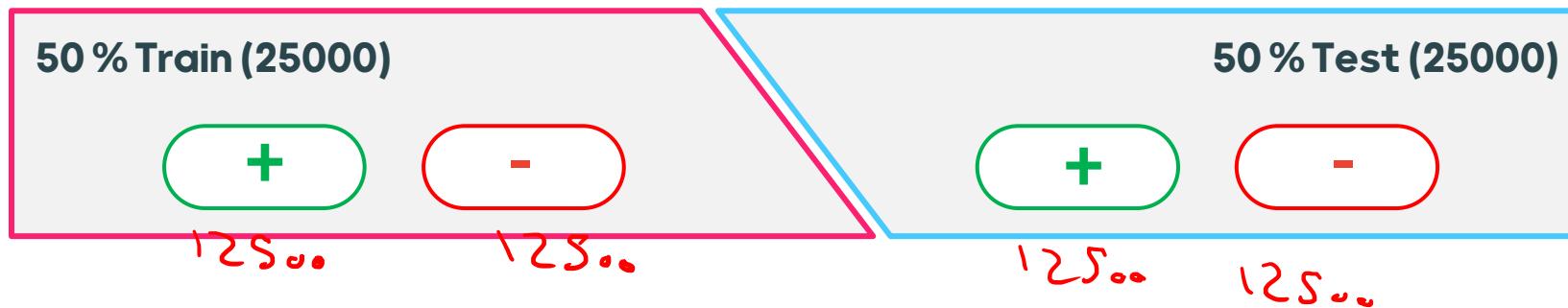
```
model = Sequential()  
model.add(SimpleRNN(4, input_shape=(3, 2), return_sequences=True))  
model.add(SimpleRNN(4))
```



مثالی کمی جدی تر



پیش‌نیاز ۱: آشنایی با دیتاست Imdb



نکته :

در Keras این دیتاست به صورت کامل پردازش شده و آماده در اختیار ما قرار داده شده است.



در Keras کلمات به تعدادی `integer` تبدیل شده اند که هر `Integer` متناظر با یک کلمه در دیکشنری است.



به عبارت دیگر

```
[  
  ["Hello", "world", "!"],  
  ["How", "are", "you", "doing", "today"],  
  ["The", "weather", "will", "be", "nice", "tomorrow"],  
]
```



```
[  
  [71, 1331, 4231],  
  [73, 8, 3215, 55, 927],  
  [83, 91, 1, 645, 1253, 947],  
]
```

خواندن دیتاست در Keras

```
from keras.datasets import imdb  
(train_data, train_labels), (test_data, test_labels) = imdb.load_data(num_words=10000)
```

8

۱۰ هزار واژه ای که بیشترین فراوانی در Training data را دارند، نگه دار. (کلمات نادر و کم تکرار را حذف کن)

~~12~~

[~~12~~, ~~17~~, ~~25~~, ~~7~~]

نکات تکمیلی در این باره

اگر محدودیتی روی این کلمات نگذاریم باید با ۸۸۵۸۵ کلمه کار کنیم که بی خودی بزرگ است !



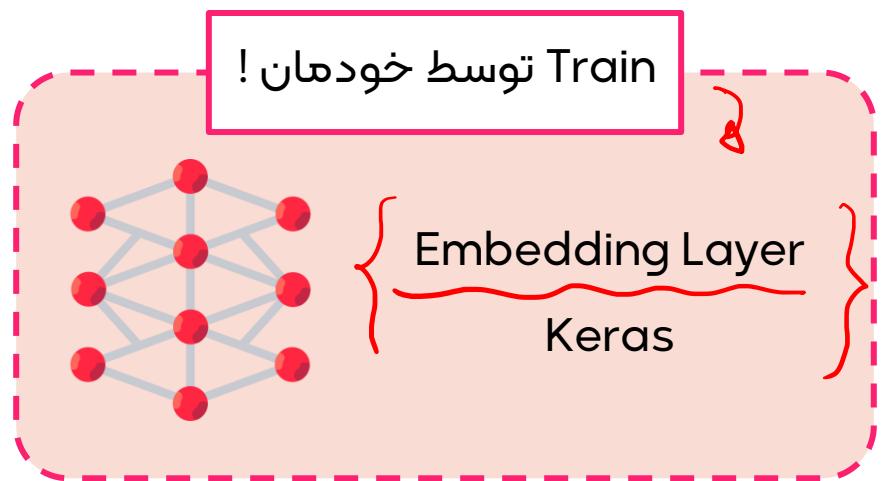
خیلی از کلمات فقط یک بار در جمله ای آمده اند و پردازش آنها تاثیر خاصی در طبقه بندی ندارد.

نمونه دیتای Train و Test

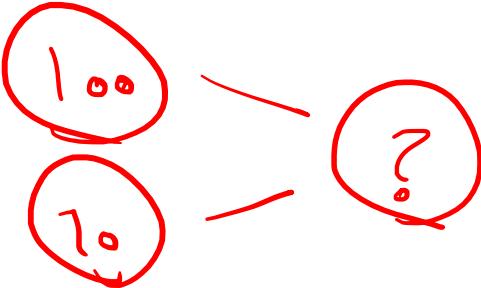
```
>>> train_data[0]
[1, 14, 22, 16, ... 178, 32]
>>> train_labels[0]
1
```

پیشناز ۲: تبدیل کلمات به بردارهای عددی

راه های تبدیل کلمات به بردارهای عددی



نکته کنکوری

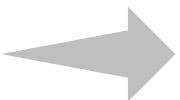


اما در Keras همه جملات باید هم اندازه باشند تا بتوانیم از Embedding Layer استفاده کنیم.

راه حل: استفاده از keras pad_sequences

جملات رو هم اندازه کنیم !

```
[ [ 71, 1331, 4231 ], [ 73, 8, 3215, 55, 927 ], [ 83, 91, 1, 645, 1253, 947 ] ]
```



```
[[ [ 711, 632, 71, 0, 0, 0 ], [ 73, 8, 3215, 55, 927, 0 ], [ 83, 91, 1, 645, 1253, 947 ] ] ]
```

تابع همین موضوع در keras

```
keras.preprocessing.sequence.pad_sequences(  
    sequences, maxlen=None, padding="pre", value=0.0)
```

ورودی: لیست تودرتو

```
[[1], [2, 3], [4, 5, 6]]
```

که همچو
که همچو

خروجی: آرایه numpy دو بعدی

```
array([[0, 0, 1],  
       [0, 2, 3],  
       [4, 5, 6]])
```

مثال ۱ و ۲

```
>>> sequence = [[1], [2, 3], [4, 5, 6]]  
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence)  
array([[0, 0, 1],  
       [0, 2, 3],  
       [4, 5, 6]], dtype=int32)
```

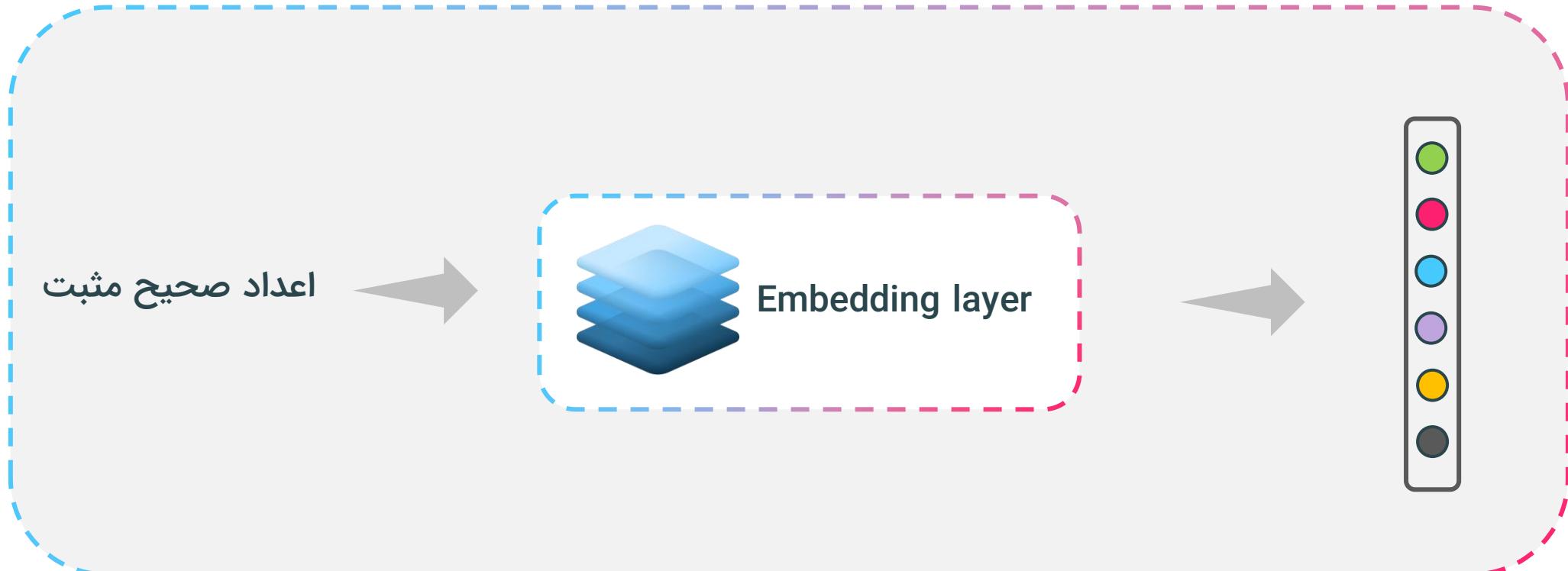
```
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence, value=-1)  
array([[ -1, -1,  1],  
       [ -1,  2,  3],  
       [ 4,  5,  6]], dtype=int32)
```

مثال ۳ و ۴

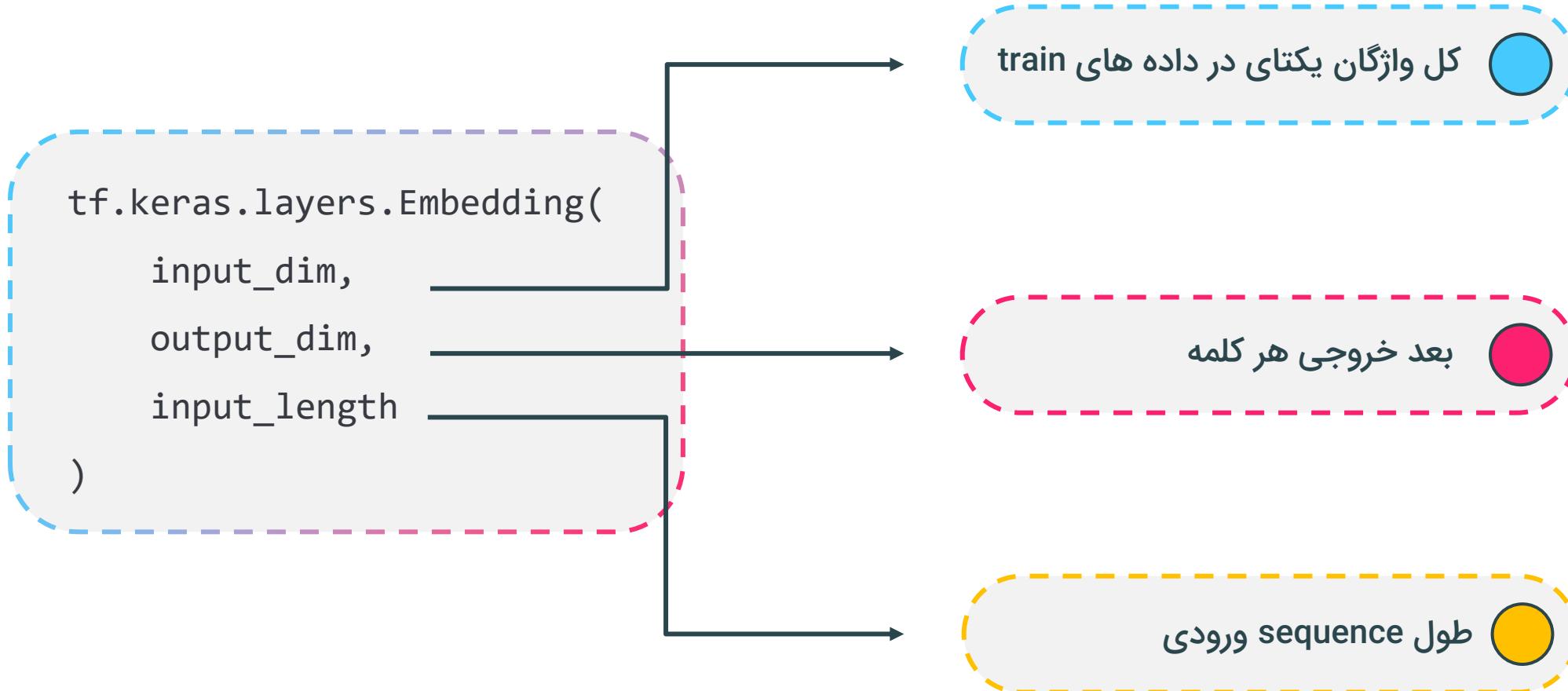
```
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence, padding='post')  
array([[1, 0, 0],  
       [2, 3, 0],  
       [4, 5, 6]], dtype=int32)
```

```
>>> tf.keras.preprocessing.sequence.pad_sequences(sequence, maxlen=2)  
array([[0, 1],  
       [2, 3],  
       [5, 6]], dtype=int32)
```

پیشیاز ۳ : لایه Embedding Keras از نگاهی کلی



embedding لایه های ورودی



حالا دیگه برمی‌وارد که بشیم.

خسته نباشید بچه ها

بخش سوم

- ❖ دو مثال ساده از این بزرگوار



❖

بخش دوم

- ❖ توضیح دقیق شبکه های بازگشتی



- ❖

بخش اول

- ❖ کاربرد شبکه های بازگشتی کجاست ؟



❖

