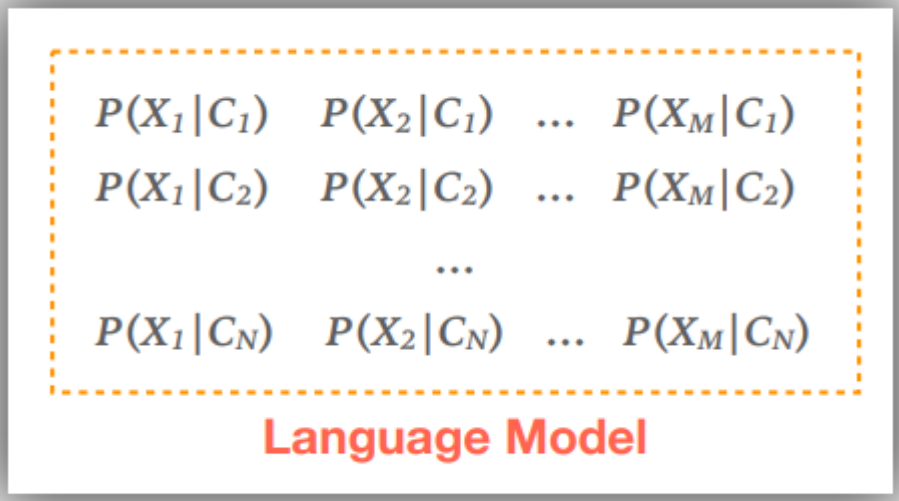
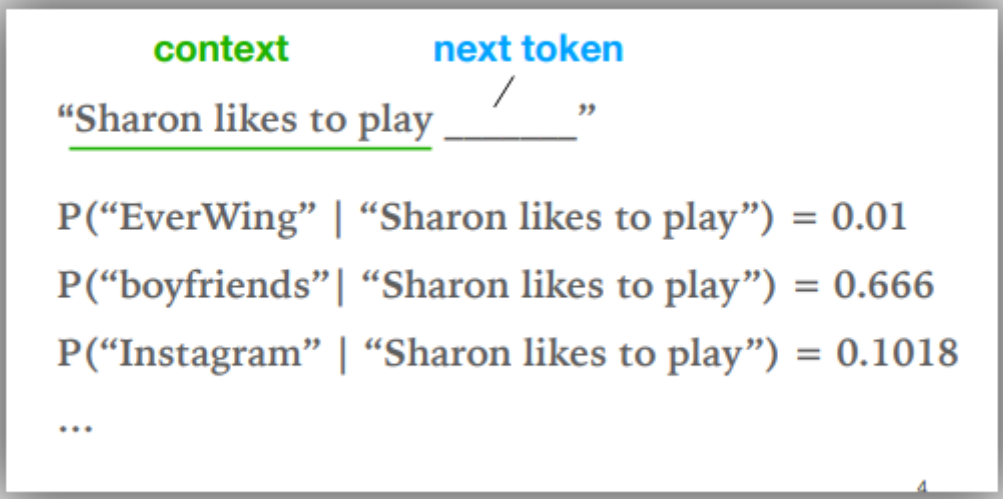
مدل زبان ما:



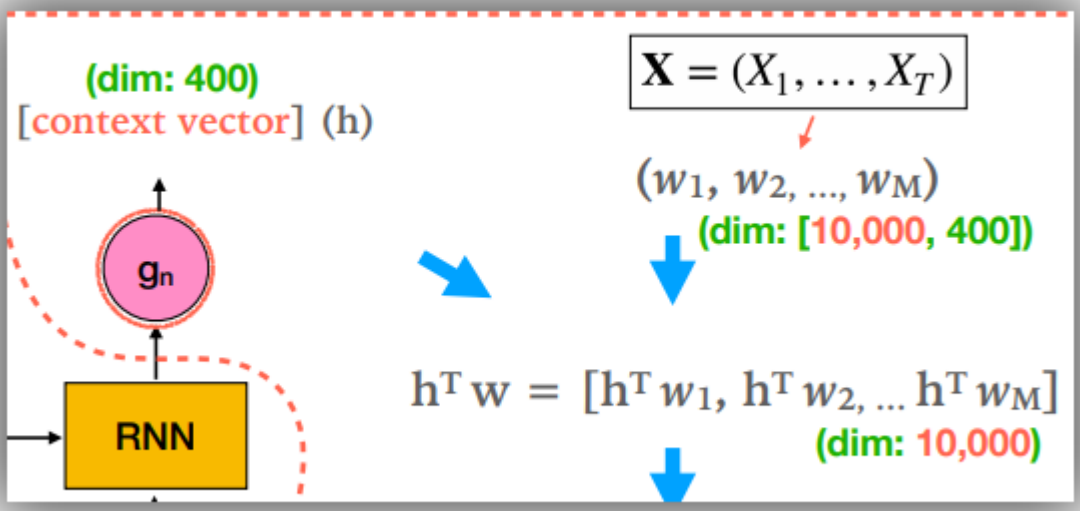
X: next token

C: context

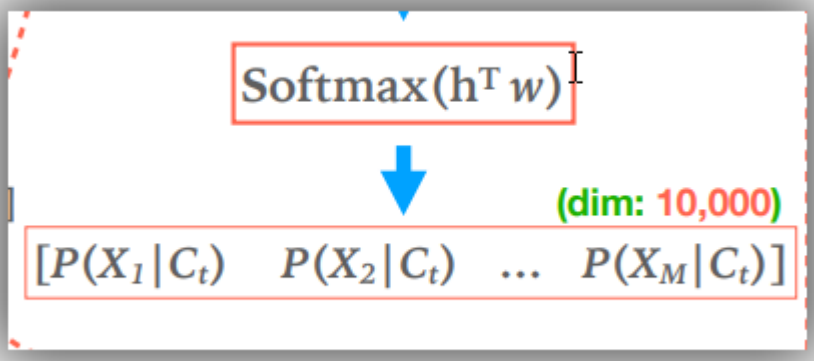
مثلا میخواهیم احتمال توکن بعدی به ازای کلمات everwings و boyfriends و Instagram و ... را در جمله زیر حساب کنیم.



rnn یک نمونه از مدل زبانی است که تک تک X ها رو با word embedding تبدیل میکنه به بردار های عددی w و context هامون رو encode میکنه به بردار های h و ضرب داخلی این دو رو به دست میاره.



بعد از اون فرمول softmax رو اعمال میکنه تا احتمالات رو به دست بیاره.



اگر یک بردار از context و کلمات زیر را داشته باشیم:

C_t = "Sharon likes to play"
$\Rightarrow$ context vector (h)

X_1 = "EverWing" $\Rightarrow$ w_1
X_2 = "boyfriends" $\Rightarrow$ w_2
X_3 = "Instagram" $\Rightarrow$ w_3
X_4 = "MaLuLian" $\Rightarrow$ w_4

ضرب داخلی آن ها به طور مثال به این صورت میباشد:

$[h^T w_1, h^T w_2, h^T w_3, h^T w_4]$

[1, 12, 7, 11]

اگر softmax بزنیم:

Goal:

$[P(X_1|C_t), P(X_2|C_t), P(X_3|C_t), P(X_4|C_t)]$ = [0.0, 0.73, 0.0, 0.27]

که هدف ما به دست آوردن چنین احتمالاتی میباشد.

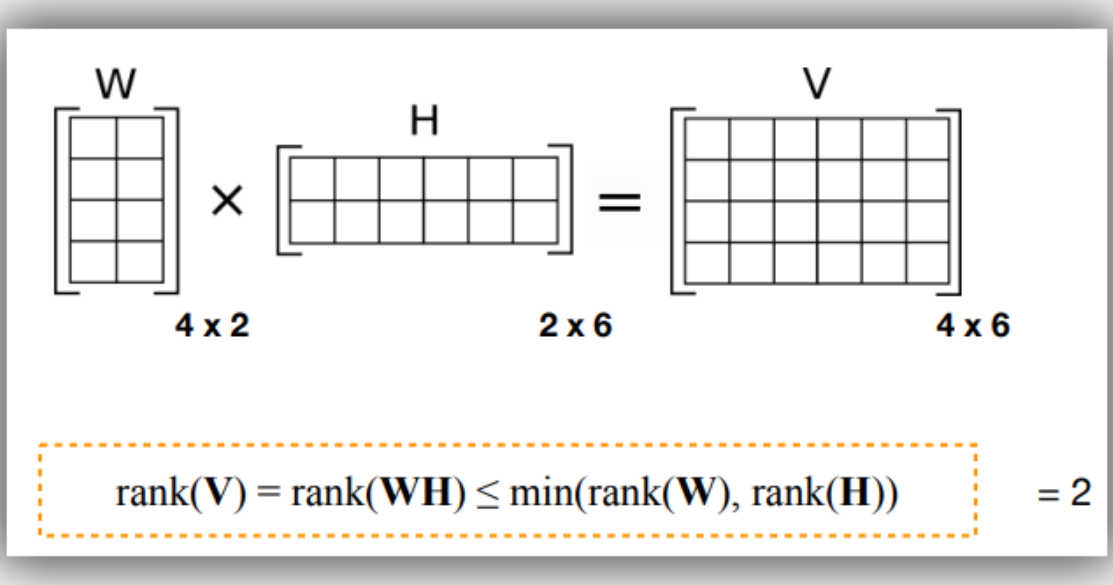
حالا سوال ما اینه که آیا این مدل مبتنی بر softmax میتونه زبان ما رو مدل کنه؟

اگر جواب نه باشه ینی عملکرد مدل ما محدود میشه در نتیجه softmax bottleneck پیش میاد.

حالا میخوایم با معیار رنک آشنا بشیم. رنک از فرمول زیر به دست میاد:

- If **A** is a *m x n* matrix: $\text{rank}(\mathbf{A}) \leq \min(m, n)$

برای مثال رنک یه ماتریس به صورت زیر 2 میباشد:



فرم ماتریس مدل ما و احتمال آن:

$$\mathbf{H}_\theta = \begin{bmatrix} \mathbf{h}_{c_1}^\top \\ \mathbf{h}_{c_2}^\top \\ \cdots \\ \mathbf{h}_{c_N}^\top \end{bmatrix} \in \mathbb{R}^{N \times d}$$

our model — (N: # of context)

$$\mathbf{W}_\theta = \begin{bmatrix} \mathbf{w}_{x_1}^\top \\ \mathbf{w}_{x_2}^\top \\ \cdots \\ \mathbf{w}_{x_M}^\top \end{bmatrix} \in \mathbb{R}^{M \times d}$$

our model — (M: # of token)

$$\text{Softmax}(\mathbf{H}_\theta \mathbf{W}_\theta^\top) = P_\theta(X|c)$$

Ground truth:

$$\mathbf{A} = \begin{bmatrix} \log P^*(x_1|c_1), & \log P^*(x_2|c_1) & \cdots & \log P^*(x_M|c_1) \\ \log P^*(x_1|c_2), & \log P^*(x_2|c_2) & \cdots & \log P^*(x_M|c_2) \\ \vdots & \vdots & \ddots & \vdots \\ \log P^*(x_1|c_N), & \log P^*(x_2|c_N) & \cdots & \log P^*(x_M|c_N) \end{bmatrix} \in \mathbb{R}^{N \times M}$$

ground truth

star (*) means: true data distribution

$$\text{Softmax}(\mathbf{A}) = P^*(X|c)$$

Pθ: مدل ما

P*: توزیع دیتای صحیح

میریم سراغ تعریف F(A):

$$F(\mathbf{A}) = \{\mathbf{A} + \mathbf{\Lambda}\mathbf{J}_{N,M} | \mathbf{\Lambda} \text{ is diagonal and } \mathbf{\Lambda} \in \mathbb{R}^{N \times N}\}$$

$J_{N,M}$ is a all-one matrix

***F* is an infinite set.**

ویژگی های F(A):

Property 1. *For any* $\mathbf{A}_1 \neq \mathbf{A}_2 \in F(\mathbf{A})$, $|rank(\mathbf{A}_1) - rank(\mathbf{A}_2)| \leq 1$.

➡ max rank difference = 1

Property 2. *For any matrix* $\mathbf{A}'$, $\mathbf{A}' \in F(\mathbf{A})$ *if and only if* $Softmax(\mathbf{A}') = P^*$.

➡ **all** possible *logits* are in $F(\mathbf{A})$

ویژگی اول: تفاوت رنک دو ماتریس در F(A) نهایتا یک میباشد.

ویژگی دوم: F(A) تمام logit های ممکن متناظر با توزیع داده های صحیح ما میباشد.

بر اساس ویژگی دوم، لم زیر را داریم:

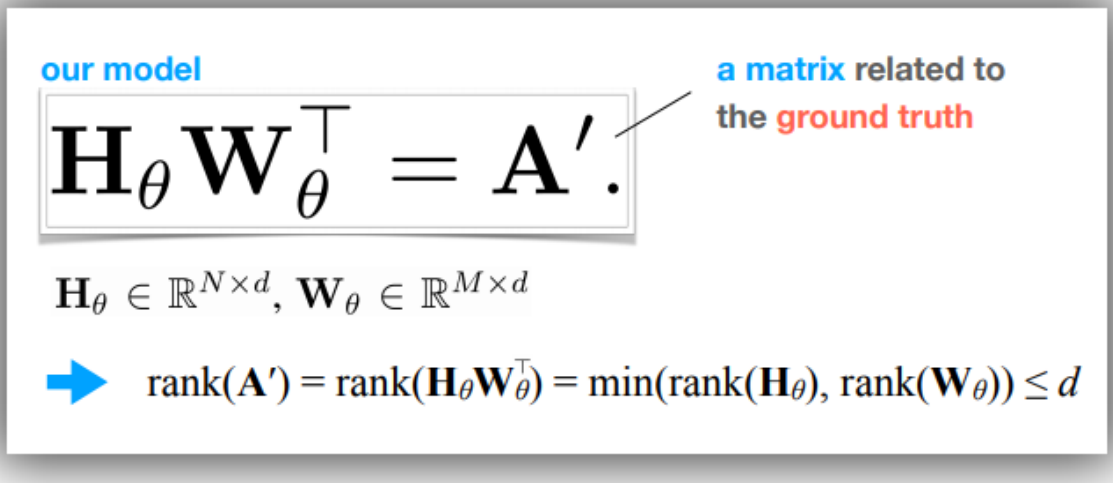
Lemma 1. *Given a model parameter* θ, $\mathbf{H}_\theta \mathbf{W}_\theta^\top \in F(\mathbf{A})$ *if and only if* $P_\theta(X|c) = P^*(X|c)$

➡ $\mathbf{H}_\theta \mathbf{W}_\theta^\top = \mathbf{A}'$.

حالا سوال ما اینه که آیا پارامتر تتا و A' ای وجود دارد که :

$$\mathbf{H}_\theta \mathbf{W}_\theta^\top = \mathbf{A}'.$$

به این مسئله matrix factorization گفته میشود. ما میخوایم مدل ما H و W ای رو یاد بگیره که A’ رو فاکتورایز کنه. یعنی میخوام از ضرب داخلی این دو ماتریس به جواب صحیح برسیم.



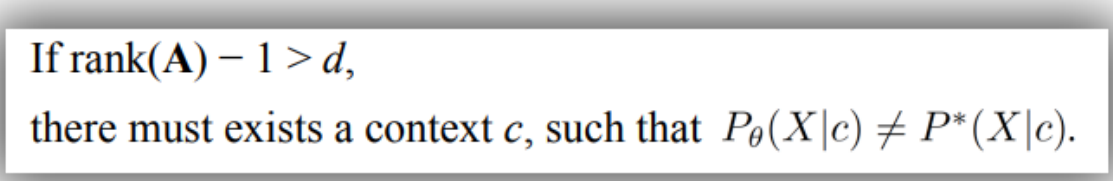
اگر رنک A’ را به دست اوریم متوجه میشویم از d کوچکتر میشود چون باید بین رنک H و W طبق فرمولی که در تصویر بالا نوشتیم مینیمم بگیریم.

N: تعداد context های ممکن.

M: تعداد توکن های ممکن در زبان.

d: بعد بردار w_x که از word embedding به دست آمده است.

Softmax bottleneck:



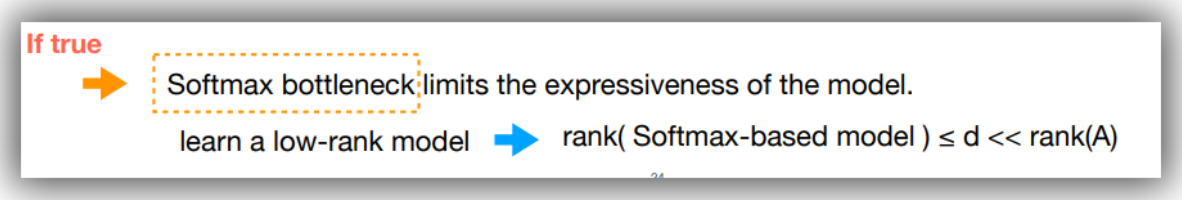
اگر رنک ماتریس A که توزیع داده های صحیح را نشان میدهد از d – 1 بزرگتر باشه یعنی context ای وجود داده که احتمال شرطی مدل ما با توزیع داده های صحیح ما برابر نیست.

اگر d بسیار کوچک باشد ینی مدل ما مبتنی بر softmax قابلیت نشان دادن توزیع داده های صحیح را ندارد.

حالا سه دلیل میاریم که زبان طبیعی یا همان A رنک بالایی داره:

1- زبان طبیعی بسیار وابسته به محتوا یا زمینه است. مثلا برای کلمه "north" در مقالات خبری قسمت سیاست بین الملل احتمال میدهیم که با کلمه "korea" یا "Korean" همراه شود ولی در کتاب های تاریخ داخلی ایالت متحده آمریکا چنین احتمالی وجود ندارد. این وابستگی به محتوا یعنی بالا بودن رنک زبان.
2- اگر زبان طبیعی رنک پایینی داشت ما میتوانستیم تمام semantic meanings را با پایه های کمی بسازیم.
3- به صورت تجربی نشان داده شده که مدل زبانی رنک بالا بهتر از مدل زبانی رنک پایین عمل میکند.

پس:



یعنی یاد گرفتن مدل رنک پایین برابر است با اینکه رنک مدل ما کوچکتر از رنک مدل زبان طبیعی شود و رسایی مدل محدود میشه.

راه حل های راحت برای softmax bottleneck:

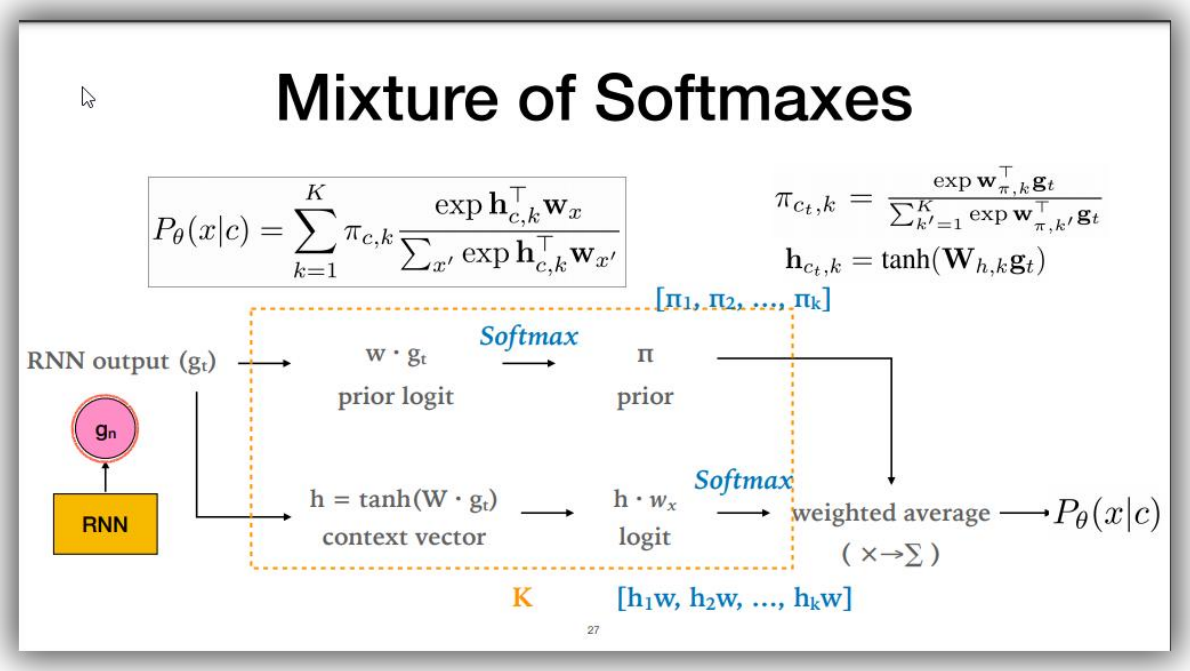
1- استفاده از مدل های ngram که با فرم های پارامتریک محدود نمیشوند و با تعداد پارامتر کافی میتوانند هر مدل زبانی ای رو تخمین بزنند.
2- افزایش بعد d که مدل با رنک بالا را نتیجه میدهد.

هر دو این راه حل ها تعداد پارامتر های مدل را به طور چشمگیر افزایش میدهد که باعث overfitting میشود.

همچنین افزایش بیشتر از چند صد عدد بعد d کمک کننده نیست.

بین expressiveness و generalization مدل trade off داریم ینی هر چی بخوایم رسایی مدل را بالا ببریم به تعمیم مدل ضربه میخورد اما در زیر روش Mos را معرفی میکنیم که رسایی را زیاد میکند بدون اینکه تعداد پارامتر ها انفجاری زیاد بشه.

Mos:



این روش k توزیع از softmax رو محاسبه میکنه و از میانگین وزن دار برای محاسبه توزیع احتمال توکن بعدی استفاده میکنه.

$\pi_{c,k}$ وزن قبلی یا ترکیب وزن ها در k مرحله می باشد. ضرب این π با h.w میانگین وزن دار به ما میدهد.

این مدل به صورت تئوری بسیار رساتر از softmax (با d یکسان) میباشد و میتوانیم این را با توجه به این واقعیت که اگر k = 1 در نظر بگیریم به softmax میرسیم متوجه بشیم.

ازون مهمتر Mos میتونه ماتریس A رو تقریب بزنه بدون اینکه محدودیت رنک ایجاد کنه.

$$\hat{\mathbf{A}}_{\text{MoS}} = \log \sum_{k=1}^{K} \boldsymbol{\Pi}_k \exp(\mathbf{H}_{\theta,k} \mathbf{W}_\theta^\top)$$

$\hat{\mathbf{A}}_{\text{MoS}}$ یک تابع غیرخطی (log_sum_exp) از بردار context و word embedding میباشد که به صورت دلخواه میتواند رنک بالا باشد.

به این صورت باعث رسا تر شدن مدل میشود در حالیکه توانایی تعمیم را پایین نمیاورد. چرا؟ میتوانیم d را کاهش دهیم تا افزایش پارامتر مدل که با ساختار mixture ایجاد شده است جبران شود در حالیکه رنک A هَت کاهش نمیابد چون یک تابع غیرخطی میباشد.