Recommendation system

Deep learning approach

"Shallow" approach

Utility matrix

matrix with user, item and rating of user to item

	A	B	C	D	E	F
Mưa nửa đêm	5	5	0	0	1	?
Cỏ úa	5	?	?	0	?	?
Vùng lá me bay	?	4	1	?	?	1
Con cò bé bé	1	1	4	4	4	?
Em yêu trường em	1	0	5	?	?	?

"Shallow" approach
1. Content-Based Recommendations

Recommend based on **item property**

	A	B	C	D	E	F	item's feature vectors
Mưa nửa đêm	5	5	0	0	1	?	$\mathbf{x}_1 = [0.99, 0.02]$
Cỏ úa	5	?	?	0	?	?	$\mathbf{x}_2 = [0.91, 0.11]$
Vùng lá me bay	?	4	1	?	?	1	$\mathbf{x}_3 = [0.95, 0.05]$
Con cò bé bé	1	1	4	4	4	?	$\mathbf{x}_4 = [0.01, 0.99]$
Em yêu trường em	1	0	5	?	?	?	$\mathbf{x}_5 = [0.03, 0.98]$
User's models	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	← **need to optimize**

Hình 2: Giả sử feature vector cho mỗi *item* được cho trong cột cuối cùng. Với mỗi *user*, chúng ta cần tìm một mô hình θ_i tương ứng sao cho mô hình thu được là *tốt nhất*.

Pre caculated

SGD

Derivative

"Shallow" approach

2. User-user, item-item Collaborative Filtering

Recommend based on **similarity of user or item**

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	5	5	2	0	1	?	?
i_1	4	?	?	0	?	2	?
i_2	?	4	1	?	?	1	1
i_3	2	2	3	4	4	?	4
i_4	2	0	4	?	?	?	5
$\bar{u}_j$	3.25	2.75	2.5	1.33	2.5	1.5	3.33

a) Original utility matrix $\mathbf{Y}$ and mean user ratings.

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	1.75	2.25	-0.5	-1.33	-1.5	0	0
i_1	0.75	0	0	-1.33	0	0.5	0
i_2	0	1.25	-1.5	0	0	-0.5	-2.33
i_3	-1.25	-0.75	0.5	2.67	1.5	0	0.67
i_4	-1.25	-2.75	1.5	0	0	0	1.67

b) Normalized utility matrix $\bar{\mathbf{Y}}$.

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
u_0	1	0.83	-0.58	-0.79	-0.82	0.2	-0.38
u_1	0.83	1	-0.87	-0.40	-0.55	-0.23	-0.71
u_2	-0.58	-0.87	1	0.27	0.32	0.47	0.96
u_3	-0.79	-0.40	0.27	1	0.87	-0.29	0.18
u_4	-0.82	-0.55	0.32	0.87	1	0	0.16
u_5	0.2	-0.23	0.47	-0.29	0	1	0.56
u_6	-0.38	-0.71	0.96	0.18	0.16	0.56	1

c) User similarity matrix $\mathbf{S}$.

Need predict missing value to fill utility matrix

	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	1.75	2.25	-0.5	-1.33	-1.5	0.18	-0.63
i_1	0.75	0.48	-0.17	-1.33	-1.33	0.5	0.05
i_2	0.91	1.25	-1.5	-1.84	-1.78	-0.5	-2.33
i_3	-1.25	-0.75	0.5	2.67	1.5	0.59	0.67
i_4	-1.25	-2.75	1.5	1.57	1.56	1.59	1.67

d) $\hat{\mathbf{Y}}$

Predict normalized rating of u_1 on i_1 with $k = 2$

Users who rated i_1 : $\{u_0, u_3, u_5\}$

Corresponding similarities: {0.83, -0.40, -0.23}

$\Rightarrow$ most similar users: $\mathcal{N}(u_1, i_1) = \{u_0, u_5\}$

with **normalized ratings {0.75, 0.5}**

$$\Rightarrow \hat{y}_{i_1,u_1} = \frac{0.83*0.75+(-0.23)*0.5}{0.83+|-0.23|} \approx 0.48$$

e) Example

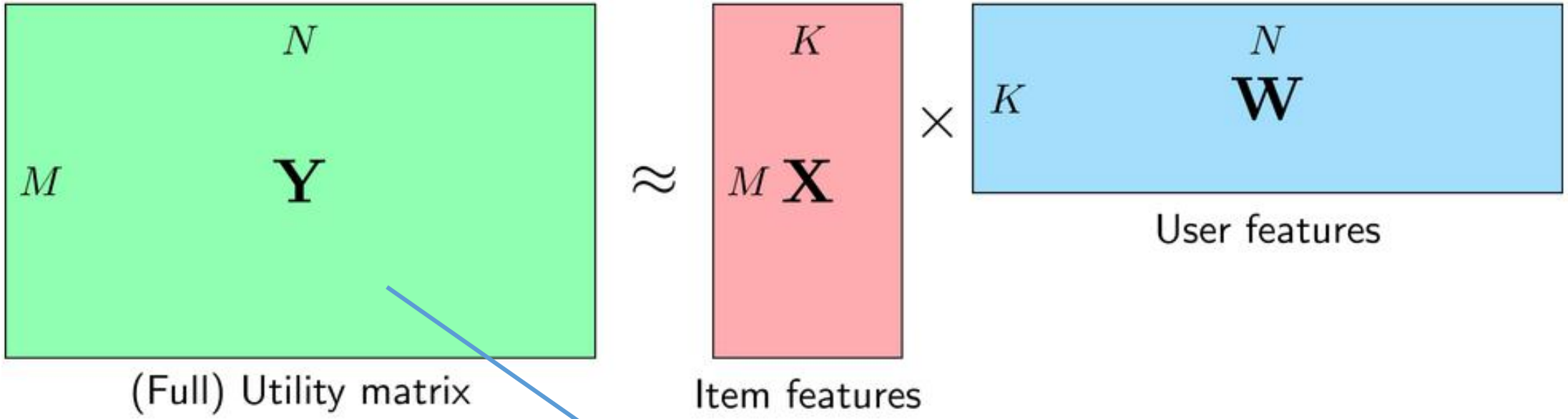
	u_0	u_1	u_2	u_3	u_4	u_5	u_6
i_0	5	5	2	0	1	1.68	2.70
i_1	4	3.23	2.33	0	1.67	2	3.38
i_2	4.15	4	1	-0.5	0.71	1	1
i_3	2	2	3	4	4	2.10	4
i_4	2	0	4	2.9	4.06	3.10	5

f) Full $\mathbf{Y}$

"Shallow" approach
3. Matrix Factorization Collaborative Filtering

Utility matrix ≈ item maxtrix * user matrix (two low rank maxtrix)

$$\mathbf{Y} \approx \hat{\mathbf{Y}} = \mathbf{X}\mathbf{W}$$



Need predict missing value to fill utility matrix

"Shallow" approach
Why is it not accurate?

- We need predict or caculate missing value in utility matrix to fill low rank matrix, ex:
  - Content-Based Recommendations: item feature vector.
  - User-user or item-item Collaborative Filtering: Normalized utility maxtrix.
  - Matrix Factorization Collaborative Filtering: full utility matrix.

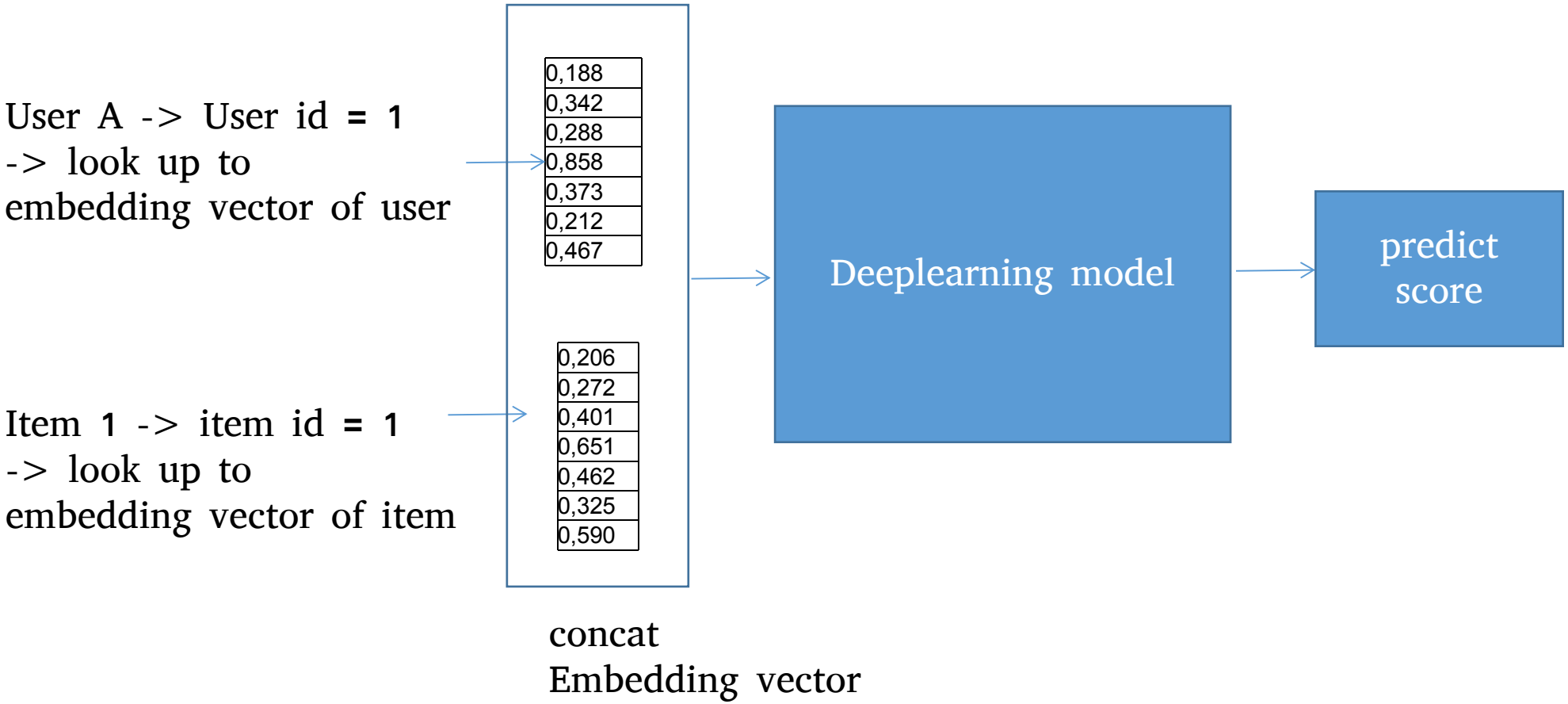
Deeplearning approach

- In deeplearning, we treat each unique value in categorycal variable with a unique id. Each unique id represent by a embedding vector with n dimension.
- The same with item

User	Id	Embedding vector (5 dimension)				
user A	1	0,188	0,769	0,206	0,168	0,075
user B	2	0,342	0,900	0,272	0,486	0,180
user C	3	0,288	0,827	0,401	0,800	0,738
user D	4	0,858	0,817	0,651	0,203	0,786
user E	5	0,373	0,784	0,462	0,327	0,954
user F	6	0,212	0,057	0,325	0,668	0,699
user G	7	0,467	0,197	0,590	0,448	0,494

Deeplearning approach Example

User A rate item 1 with score = 7

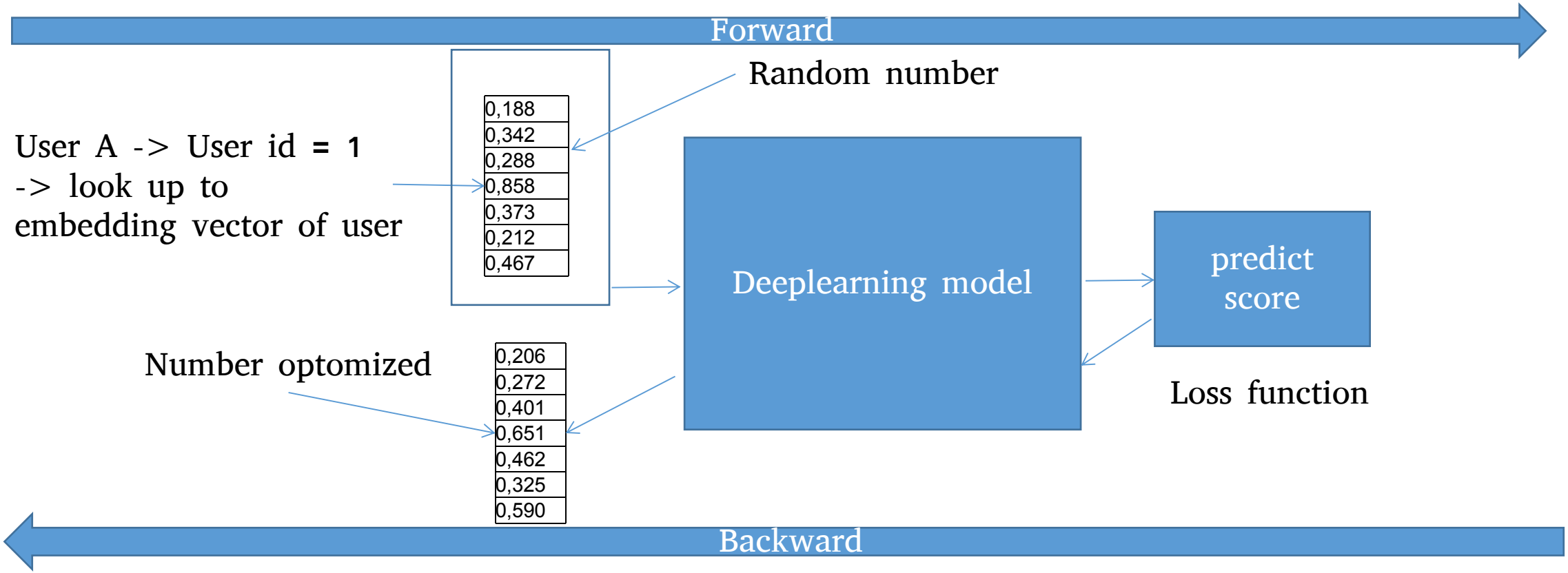


"Shallow" vs Deeplearning approach What's Different?

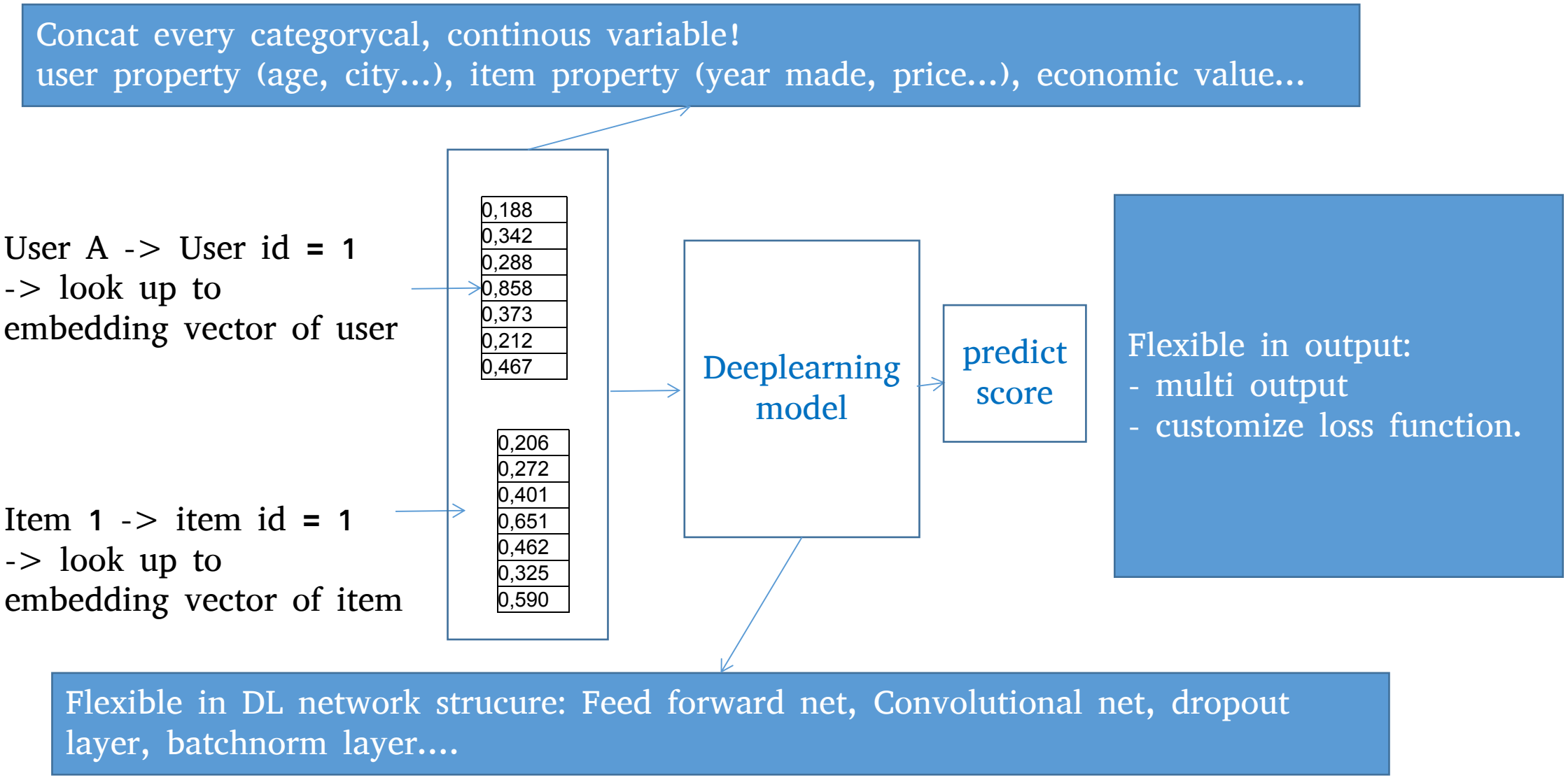
Deeplearning approach only use data of **user already rate item** to fit model!

Deeplearning approach
How to caculate embedding vector?

backpropagation is the key!

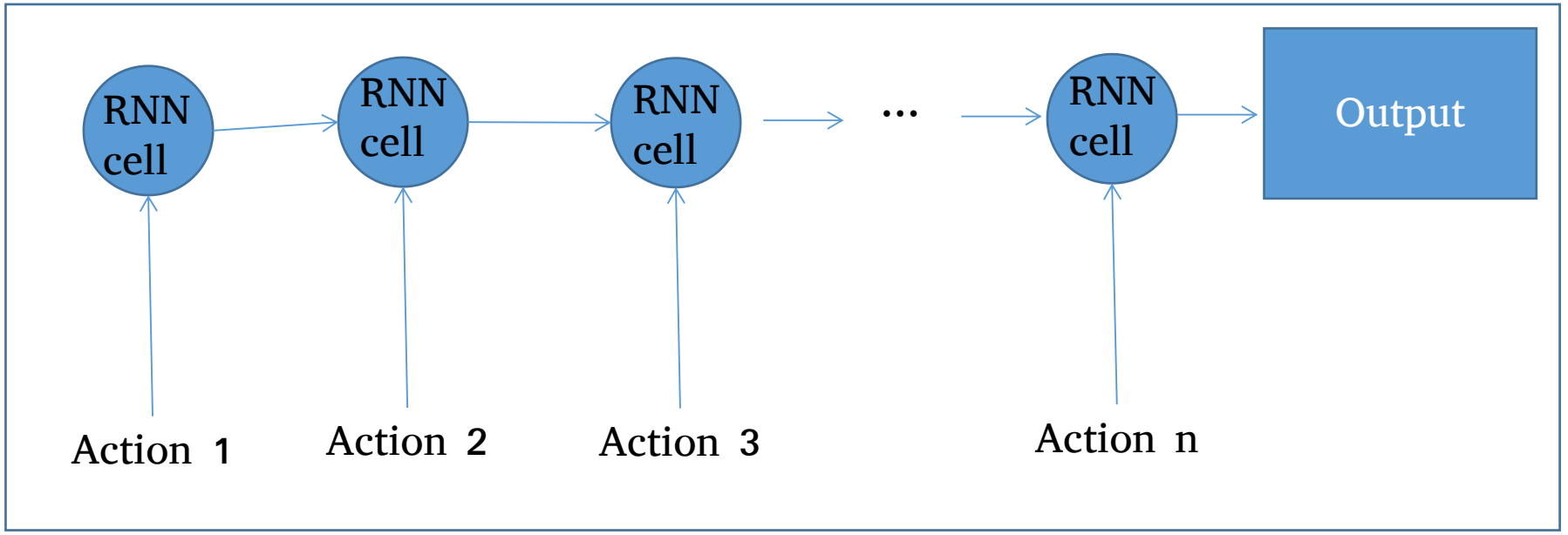


Deeplearning approach
Very flexible!



Deeplearning approach
One last thing. Session base recommendation

Recommend base on **order of user action with sequence model.**



Sequence model