Deep Learning

Deep Recommender System

U Kang
Seoul National University

In This Lecture

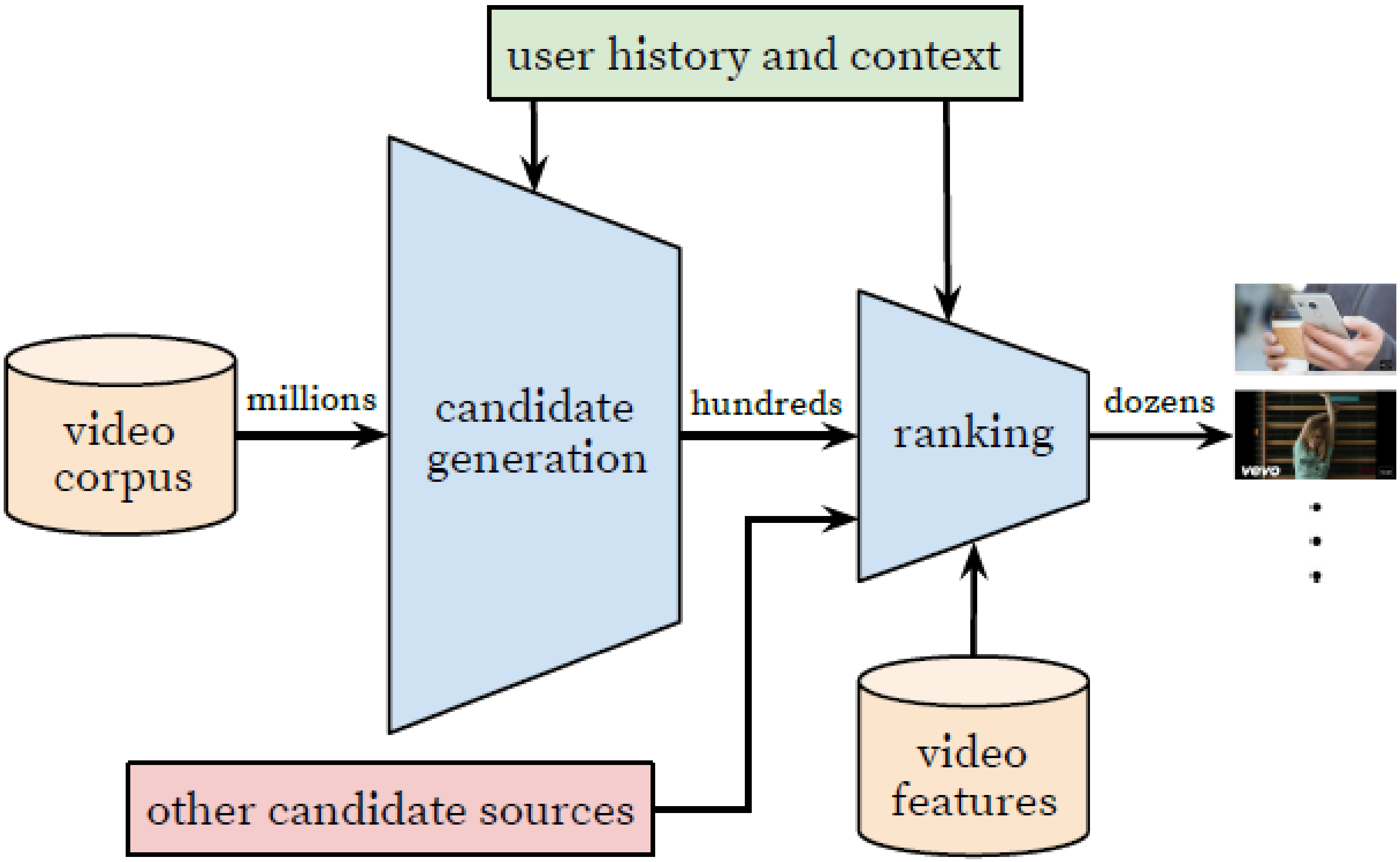
- Deep recommender system
  - MLP Based System
  - AE Based System
  - CNN Based System
  - RNN Based System

Outline

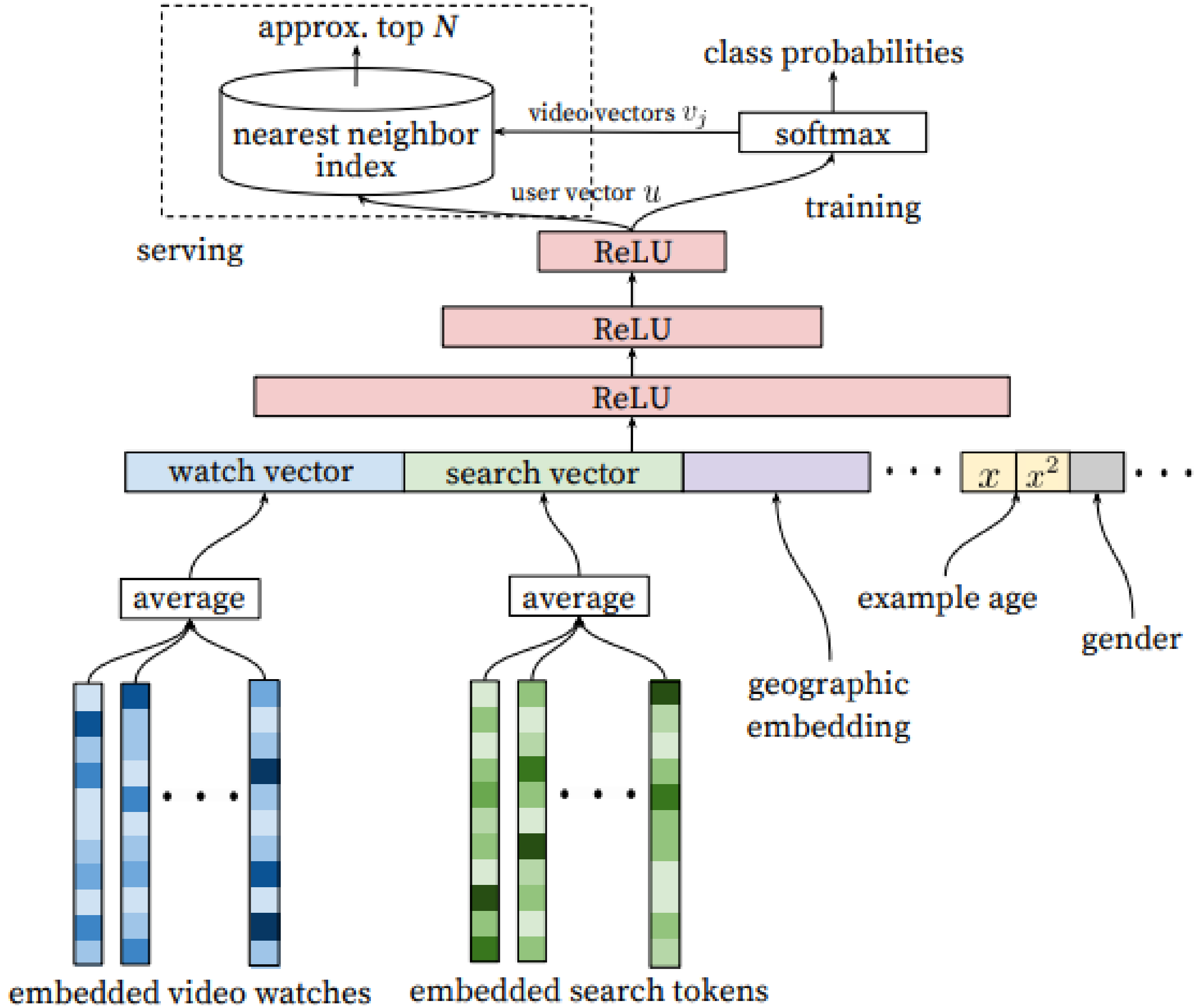
- ☐ **MLP Based System**
- ☐ AE Based System
- ☐ CNN Based System
- ☐ RNN Based System

Covington et al., Deep Neural Networks for Youtube Recommendations, RecSys'16

Deep Neural Networks for Youtube Recommendations

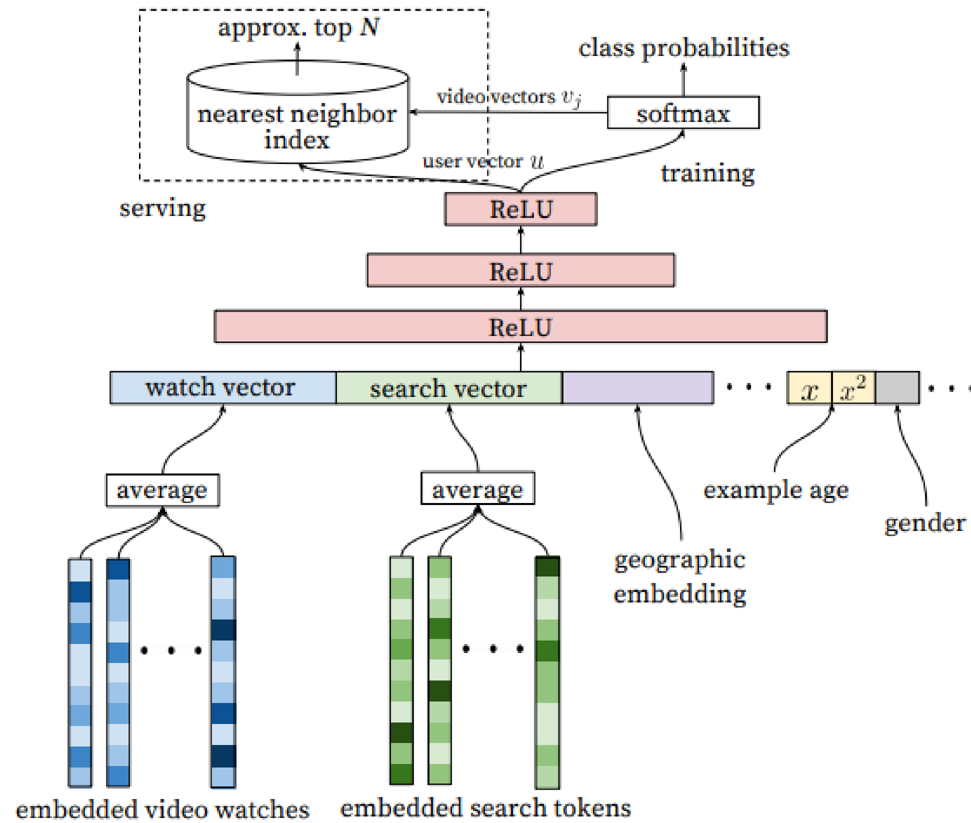


Candidate Generation



Embedding

- Embedded video watches, and embedded search tokens
- The vectors are learned together, using backpropagation

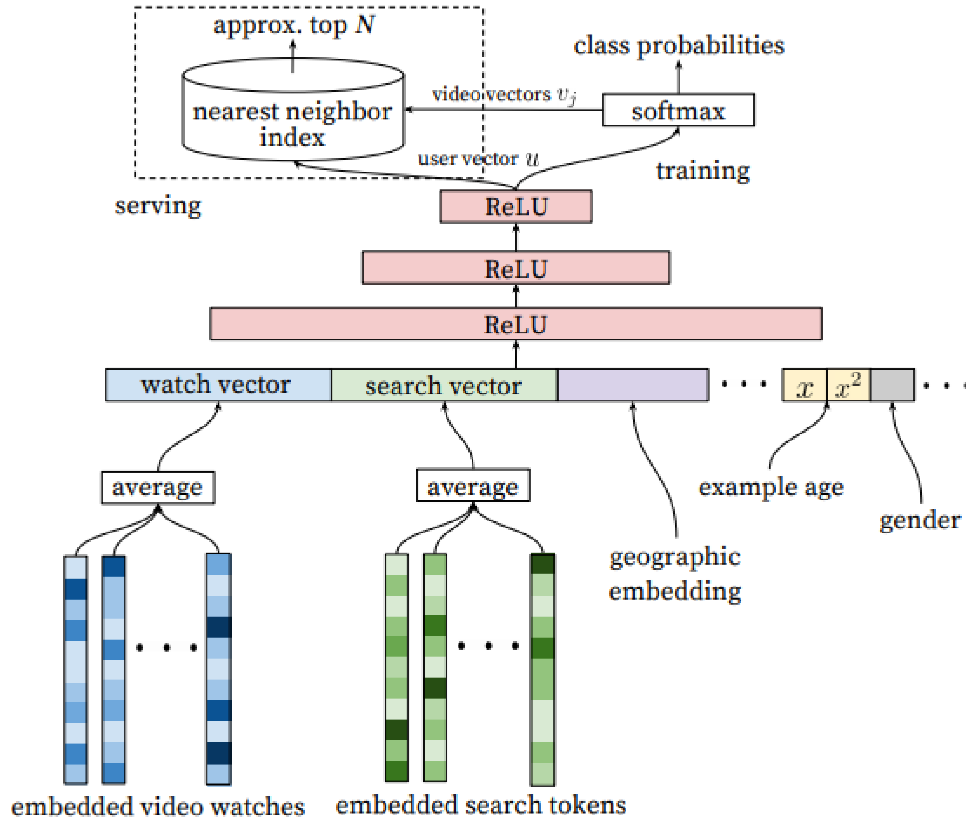


Recommendation as Classification

- Recommendation can be viewed as extreme multiclass classification to accurately classify a specific video watch w_t at time t among millions of video i from a corpus V, based on user U and context C
  - $P(w_t = i|U, C) = \frac{e^{v_i u}}{\sum_{j \in V} e^{v_j u}}$

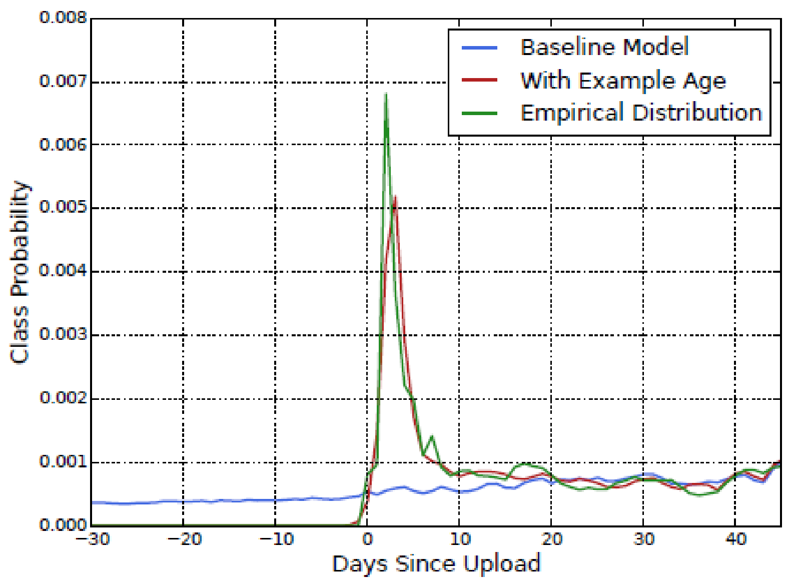
Serving

- Return top N results
- Nearest neighbor using dot-products

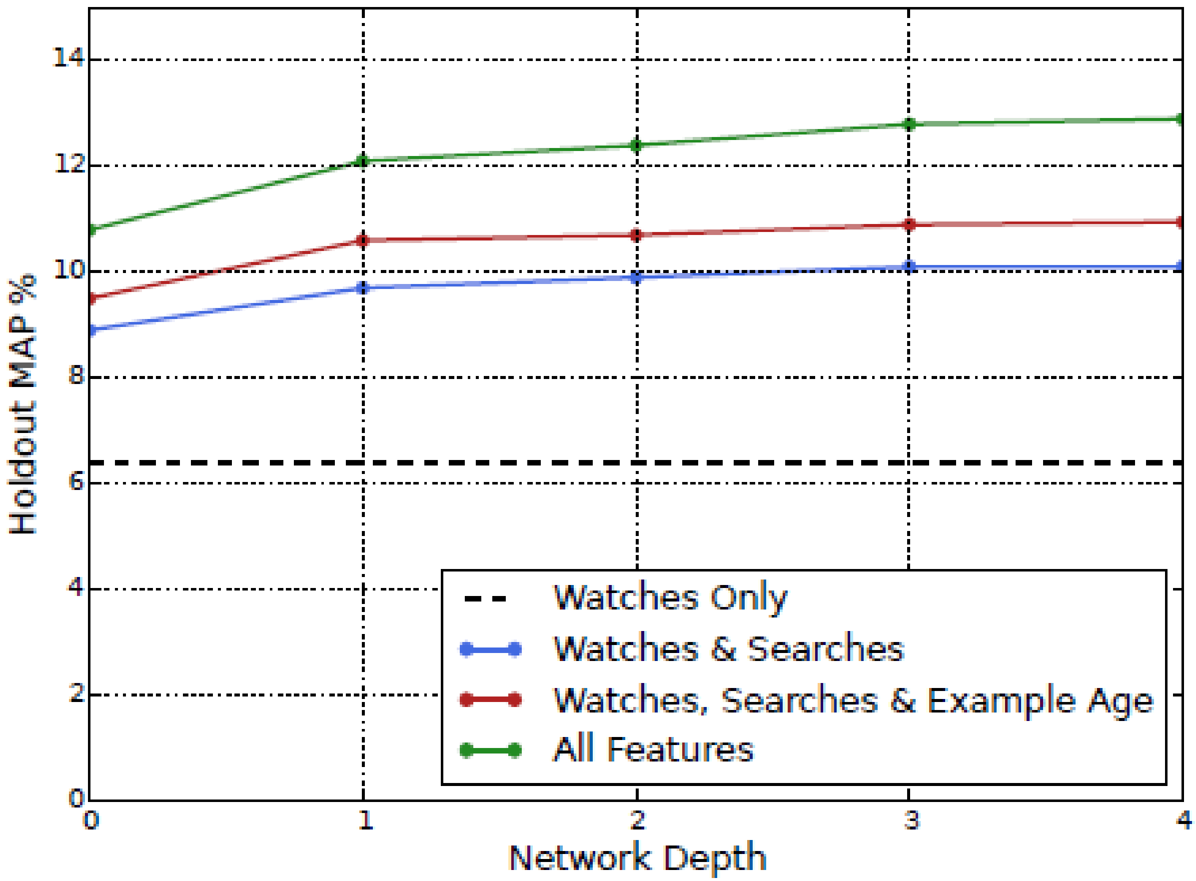


Example Age Feature

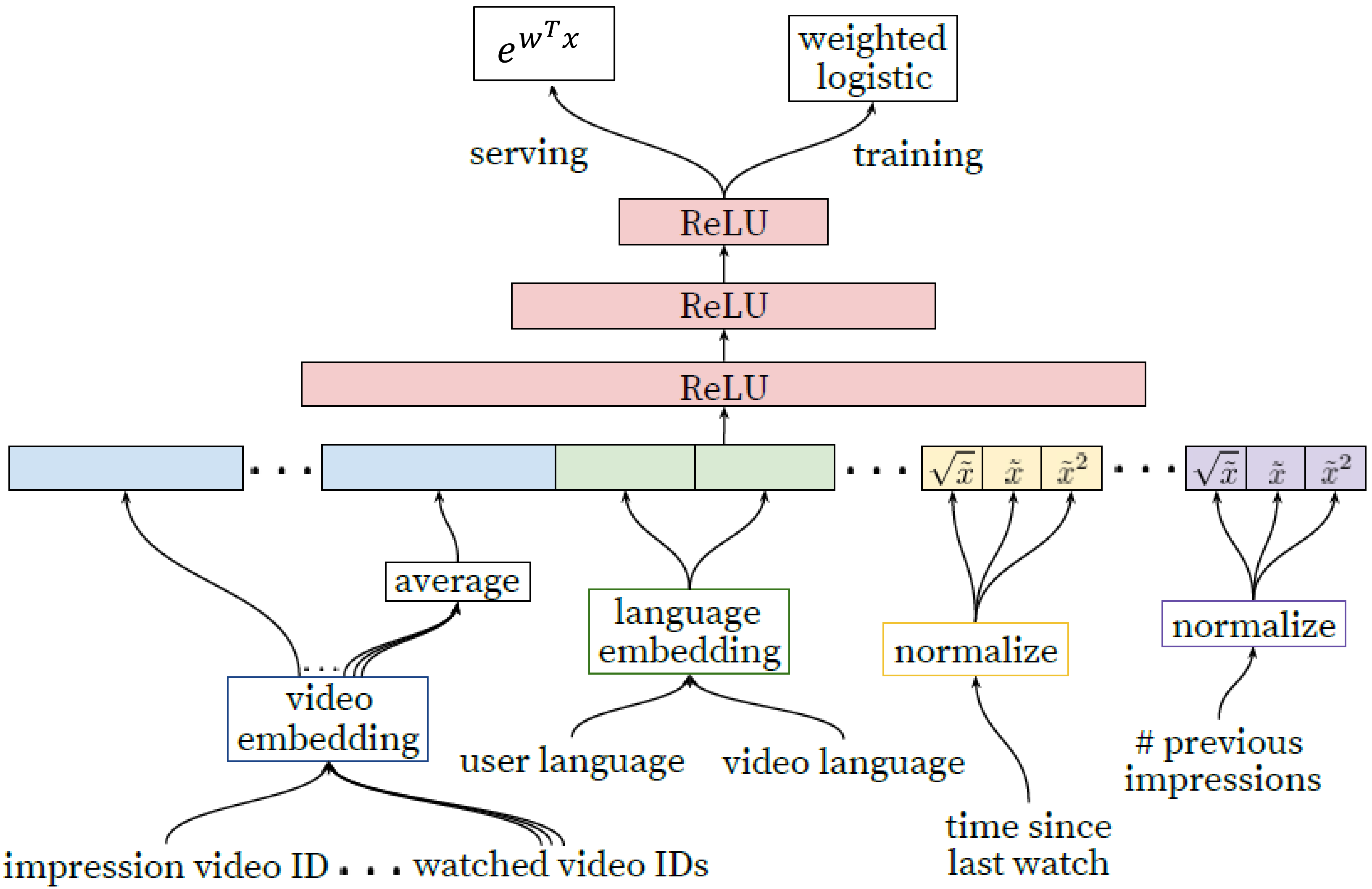
- Machine learning systems often exhibit an implicit bias towards the past because they are trained to predict future behavior from historical examples
- Correction
  - Feed the age of the training example as a feature during training
  - At serving time, this feature is set to zero (or slightly negative) to reflect that the model is making predictions at the very end of the training window



Effects of Features



Ranking



$e^{w^T x}$
weighted logistic
serving
training
ReLU
ReLU
ReLU
$\sqrt{\tilde{x}}$ $\tilde{x}$ $\tilde{x}^2$
$\sqrt{\tilde{x}}$ $\tilde{x}$ $\tilde{x}^2$
average
language embedding
normalize
normalize
video embedding
user language
video language
\# previous impressions
time since last watch
impression video ID . . . watched video IDs

Normalization

- A continuous feature x with distribution f is transformed to $\tilde{x}$ by scaling the values such that the feature is equally distributed in [0, 1) using the cumulative distribution $\tilde{x} = \int_{-\infty}^{x} p(x)dx$
- In addition to raw normalized feature $\tilde{x}$, add powers $\tilde{x}^2$ and $\sqrt{\tilde{x}}$, giving the network more expressive power by allowing it to easily form super- and sub-linear functions of the feature.

Model Training

- Model is trained with logistic regression under cross-entropy loss
  - Positive impressions are weighted by the observed watch time on the video
  - Negative impressions all receive unit weight

Experiments with Hidden Layers

- Increasing width and depth improves performance

Hidden layers	weighted, per-user loss
None	41.6%
256 ReLU	36.9%
512 ReLU	36.7%
1024 ReLU	35.8%
512 ReLU $\rightarrow$ 256 ReLU	35.2%
1024 ReLU $\rightarrow$ 512 ReLU	34.7%
1024 ReLU $\rightarrow$ 512 ReLU $\rightarrow$ 256 ReLU	34.6%

Covington et al.: Summary

- FNN-based recommender system
  - Candidate generation followed by ranking of candidates
  - Partially consider sequence information in FNN framework (by averaging vectors for watches)
  - Interesting features
    - Video embedding
    - “Age” of videos

Outline

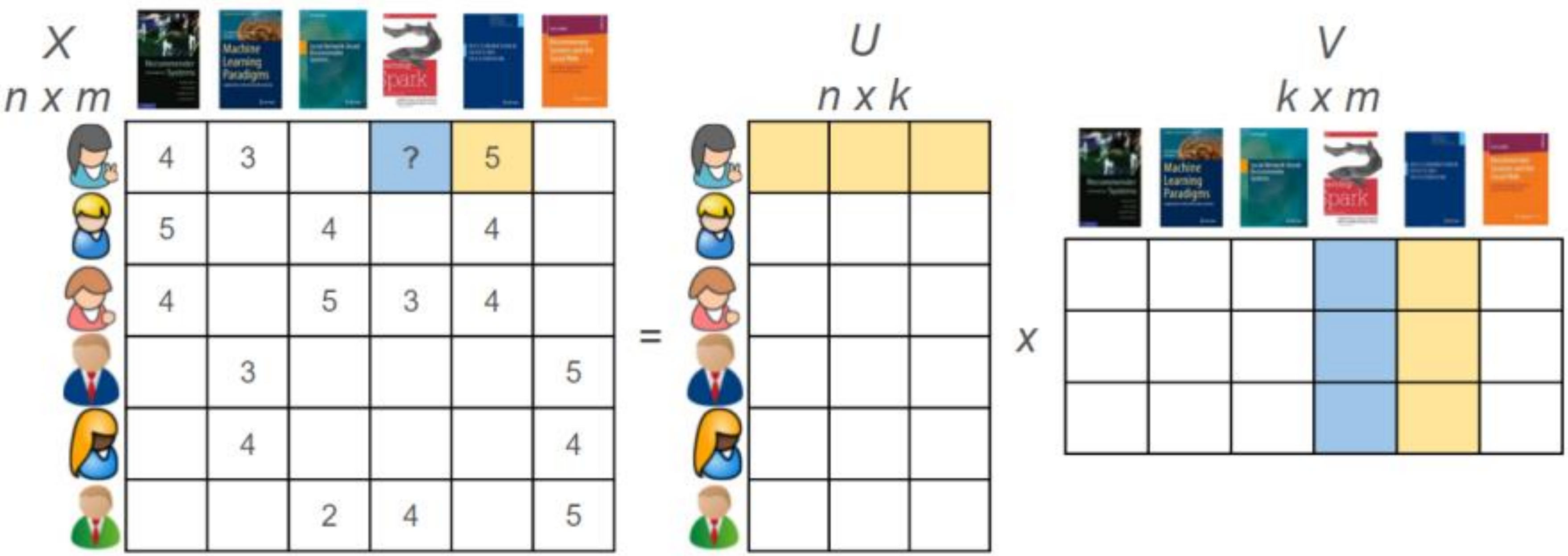
- ☑ MLP Based System
- ☐ **AE Based System**
- ☐ CNN Based System
- ☐ RNN Based System

Wang et al., Collaborative Deep Learning for Recommender Systems, KDD'15

Problem Definition

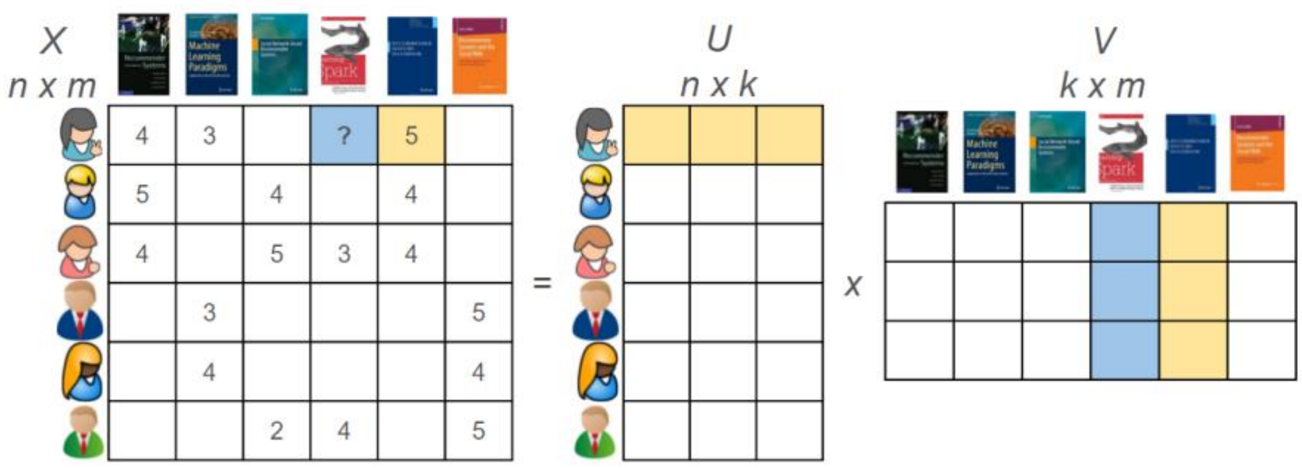
- Given
  - Rating information of users and items, and additional information (e.g. text) for items
- Goal
  - Infer the ratings of unrated items by users

Matrix Factorization



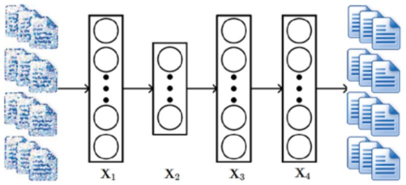
Collaborative Deep Learning

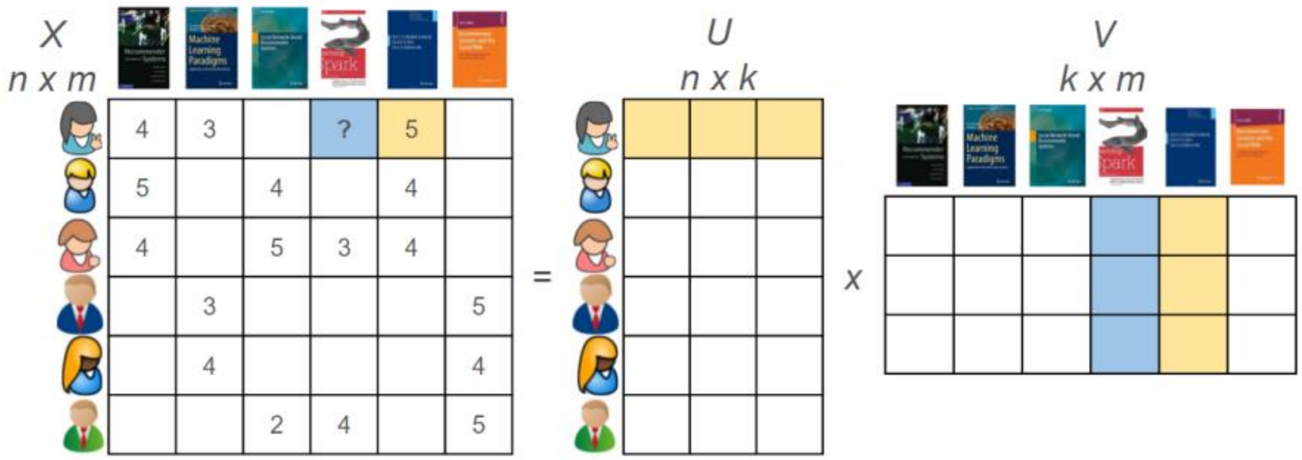
- Challenges
  - How to incorporate text information of items in its embeddings, such that items with similar contents are more likely to have similar embeddings?



Collaborative Deep Learning

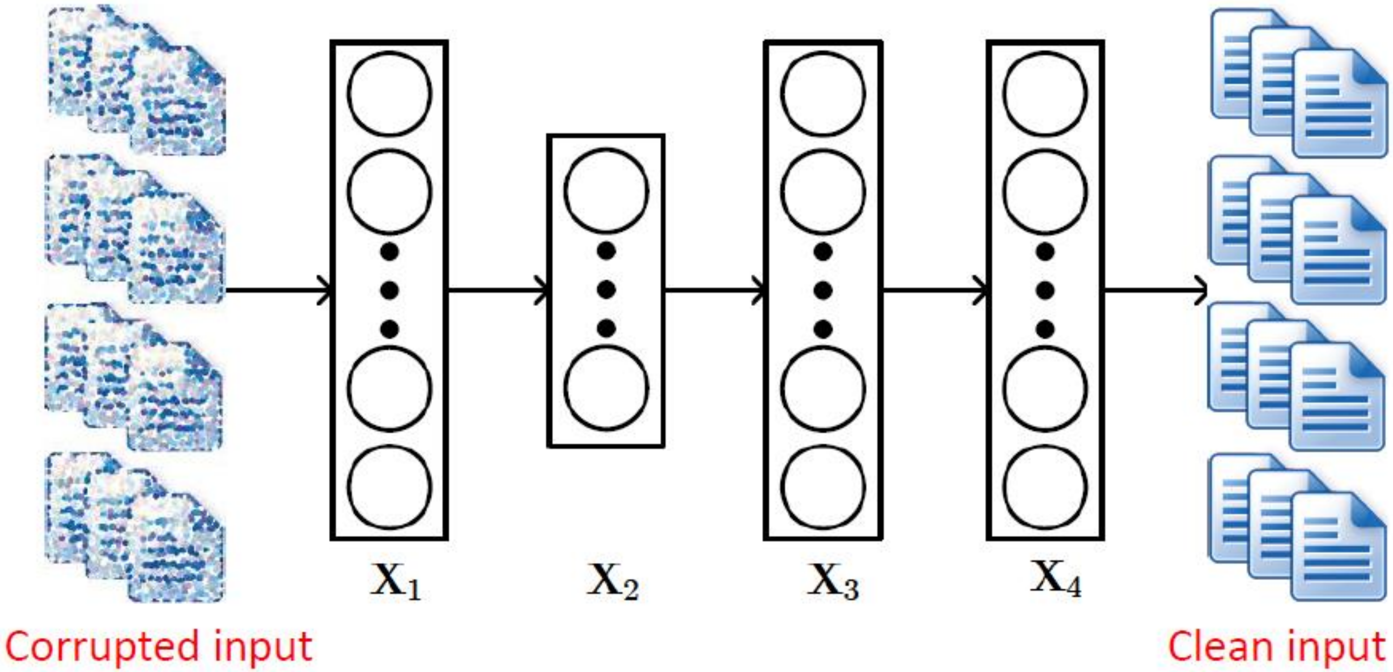
- Main Idea
  - Use code vectors of an “autoencoder” as item vectors!
  - The autoencoder is jointly trained in the MF framework





Stacked Denoising Autoencoder (SDAE)

Feature:
bag of words



SDAE solves the following optimization problem:

$$\min_{\{\mathbf{W}_l\},\{\mathbf{b}_l\}} \|\mathbf{X}_c - \mathbf{X}_L\|_F^2 + \lambda \sum_l \|\mathbf{W}_l\|_F^2,$$

where λ is a regularization parameter and $\|\cdot\|_F$ denotes the Frobenius norm.

Vincent et al. 2010

Learning

$$\mathscr{L} = \boxed{-\frac{\lambda_u}{2}\sum_i \|\mathbf{u}_i\|_2^2 - \frac{\lambda_w}{2}\sum_l (\|\mathbf{W}_l\|_F^2 + \|\mathbf{b}_l\|_2^2)}$$

$$\boxed{-\frac{\lambda_v}{2}\sum_j \|\mathbf{v}_j - \mathbf{X}_{\frac{L}{2},j*}^T\|_2^2} - \boxed{\frac{\lambda_n}{2}\sum_j \|\mathbf{X}_{L,j*} - \mathbf{X}_{c,j*}\|_2^2}$$

$$\boxed{-\frac{\lambda_s}{2}\sum_l \sum_j \|\sigma(\mathbf{X}_{l-1,j*}\mathbf{W}_l + \mathbf{b}_l) - \mathbf{X}_{l,j*}\|_2^2}$$

$$\boxed{-\sum_{i,j}\frac{\mathbf{C}_{ij}}{2}(\mathbf{R}_{ij} - \mathbf{u}_i^T\mathbf{v}_j)^2.}$$

SDAE loss

Use code as item vector

SDAE layer-next layer relation

MF loss

Datasets

	citeulike-a	citeulike-t	Netflix
#users	5551	7947	407261
#items	16980	25975	9228
#ratings	204987	134860	15348808

Content information

Titles and abstracts

Titles and abstracts

Movie plots

Evaluation Metrics

Recall:

$$\text{recall@}M = \frac{\text{number of items that the user likes among the top } M}{\text{total number of items that the user likes}}$$

Mean Average Precision (mAP):

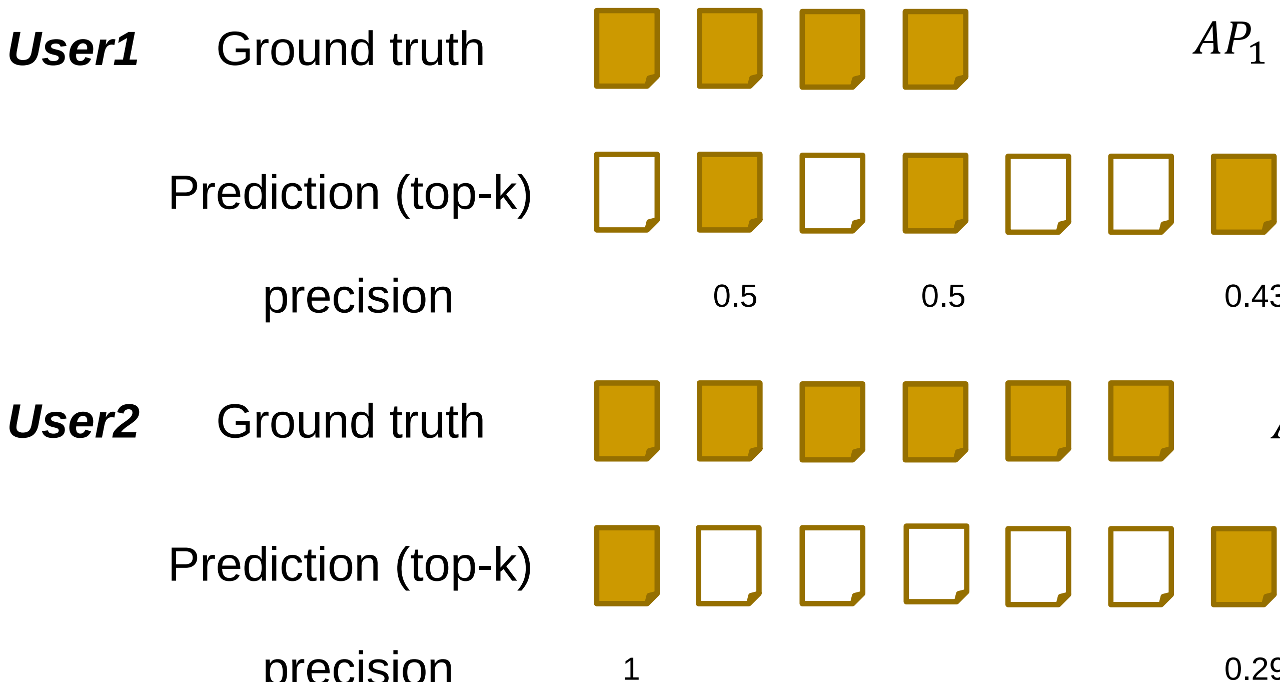
$$mAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$

P(k): precision at k
rel(k): 1 if relevant, 0 otherwise

$$AveP = \frac{\sum_{k=1}^{n} (P(k) \times rel(k))}{\text{number of relevant items}}$$

Higher recall and mAP indicate better recommendation performance

MAP



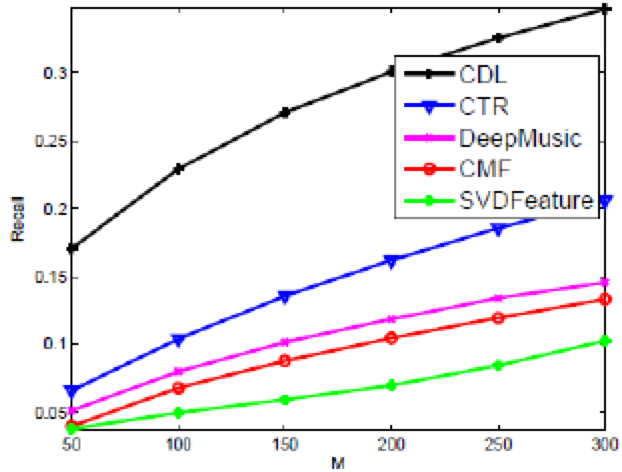
$$AP_1 = \frac{0.5 + 0.5 + 0.43}{4} = 0.36$$

$$AP_2 = \frac{1 + 0.29}{6} = 0.22$$

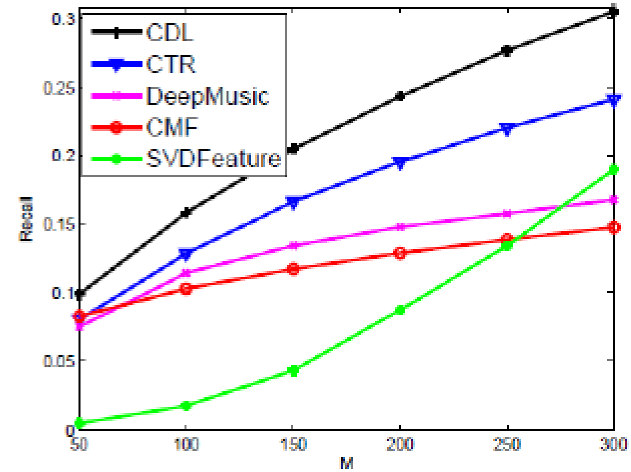
$$MAP = \frac{AP_1 + AP_2}{2} = 0.29$$

Recall

When the ratings are **very sparse**:

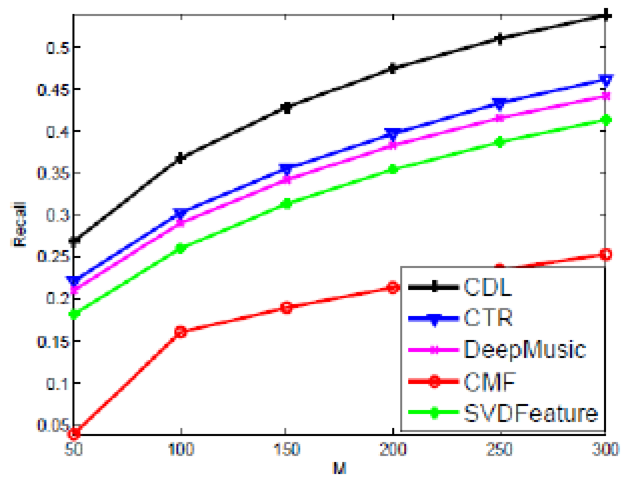


***citeulike-t*, sparse setting**

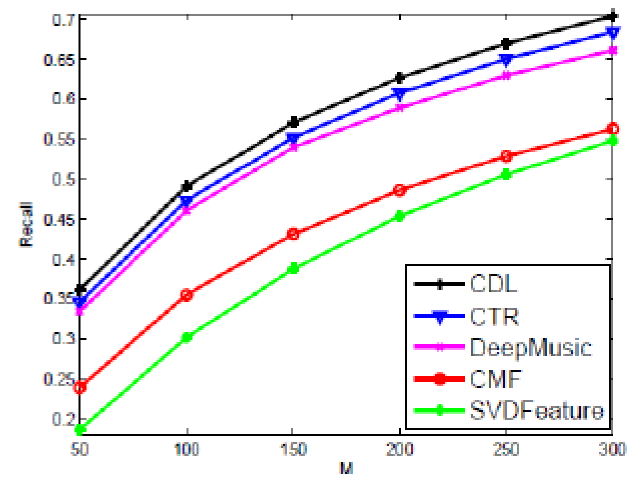


***Netflix*, sparse setting**

When the ratings are **dense**:



***citeulike-t*, dense setting**



***Netflix*, dense setting**

Mean Average Precision (mAP)

	citeulike-a	*citeulike-t*	*Netflix*
CDL	**0.0514**	**0.0453**	**0.0312**
CTR	0.0236	0.0175	0.0223
DeepMusic	0.0159	0.0118	0.0167
CMF	0.0164	0.0104	0.0158
SVDFeature	0.0152	0.0103	0.0187

CDL: Summary

- AE-based recommender system
  - Combined PMF framework with deep learning
  - Learn item representation from stacked denoising autoencoder (SDAE)
  - Effective especially when the rating is very sparse

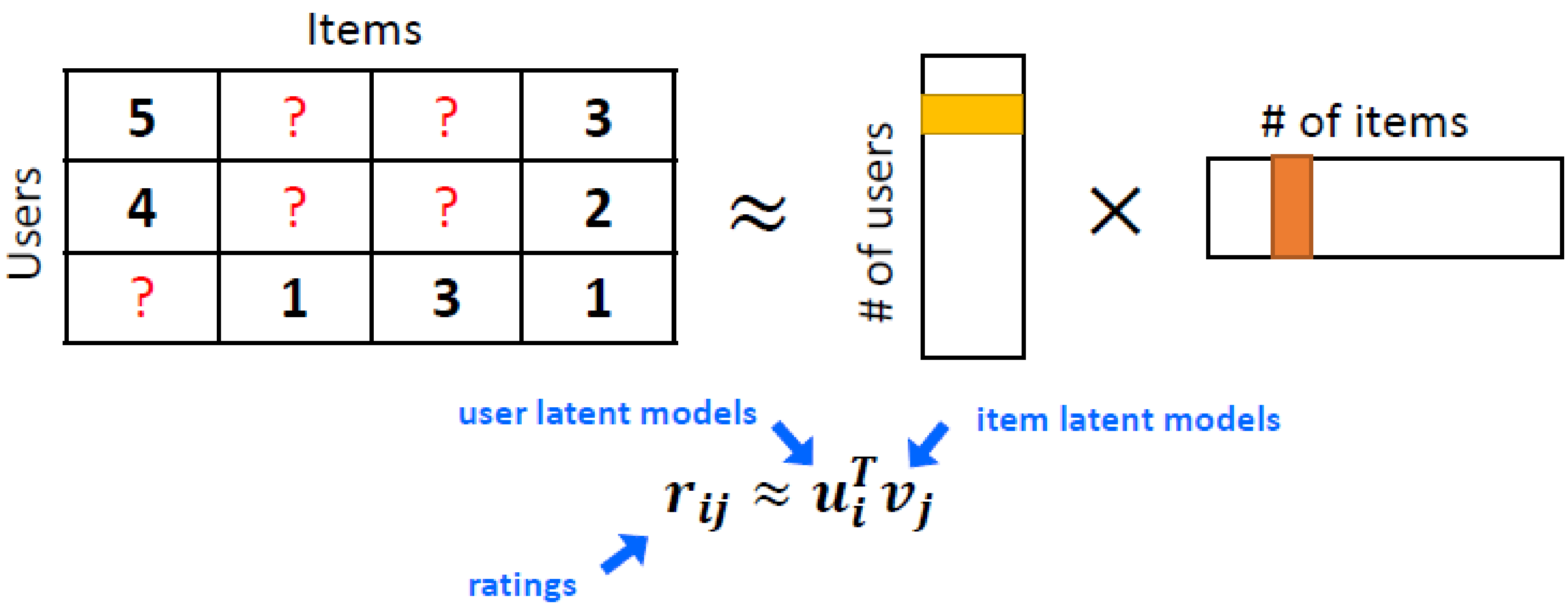
Outline

- ☑ MLP Based System
- ☑ AE Based System
- ☐ **CNN Based System**
- ☐ RNN Based System

Kim et al., Convolutional Matrix Factorization for Document Context-Aware Recommendation, RecSys'16

Matrix Factorization

- A popular model-based CF method



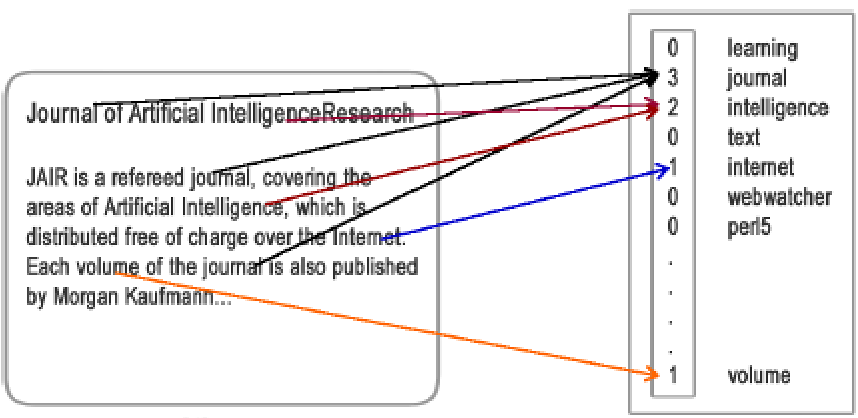
Use of Text

- To handle sparseness of a rating matrix, text information (review, synopsis, abstract etc.) can be used

a description document

Previous Work

- Collaborative deep learning for recommender system (CDL): use Stacked Denoising Autoencoder (SDAE)
  - Limitation: bag of words models



Ignore

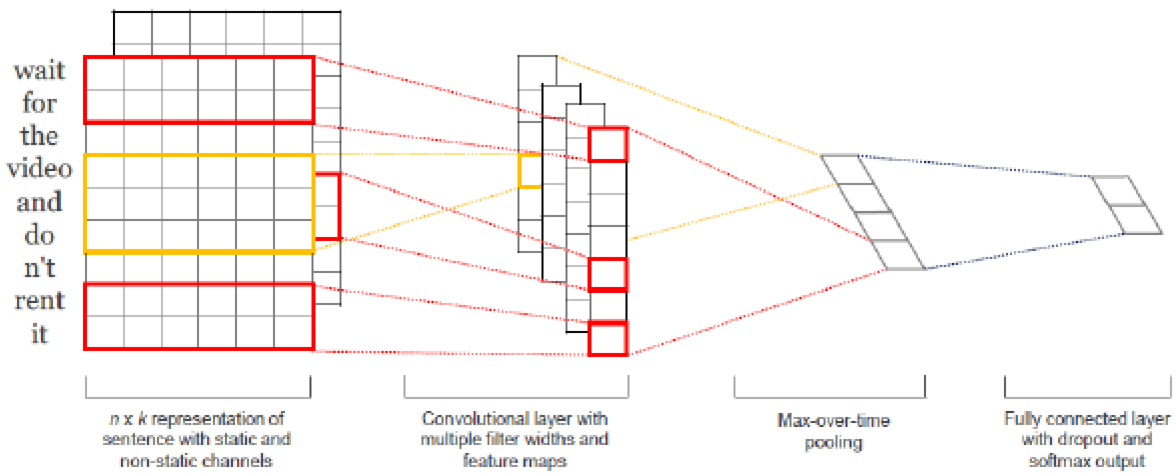
- ✓ surrounding words of a word
- ✓ word order

Convolutional MF (ConvMF)

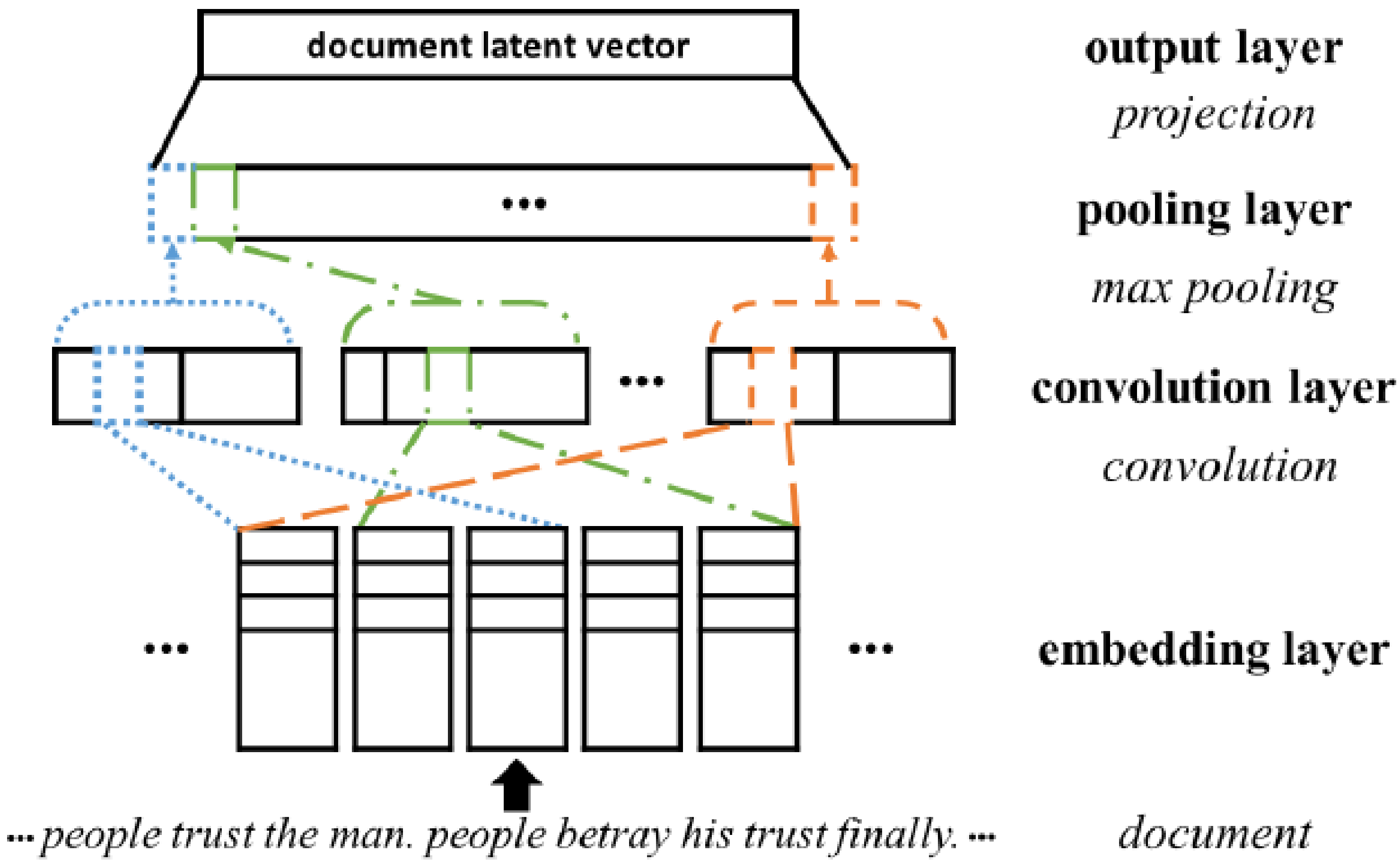
- Consider contextual information
  - Considering surrounding words and word order as "contextual information" improves the accuracy of word vectors in the word embedding
    - Word2vec
- Effectively exploit both ratings and description documents
- Jointly optimize the recommendation model in order to properly predict ratings to items of users

CNN

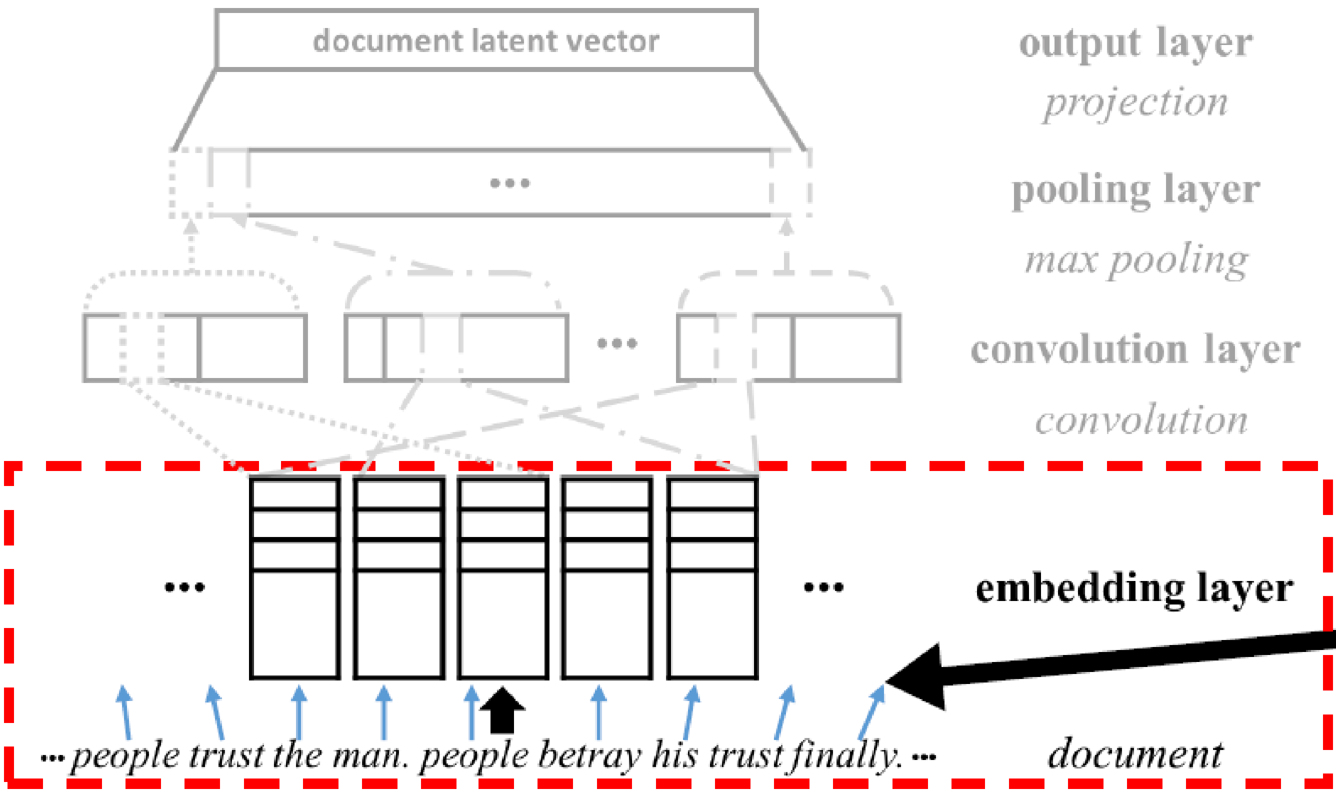
- For NLP and IR tasks, CNN have been mainly developed to consider local contextual information in a document
- Example of CNN architecture for sentiment classification



Overview of CNN in ConvMF



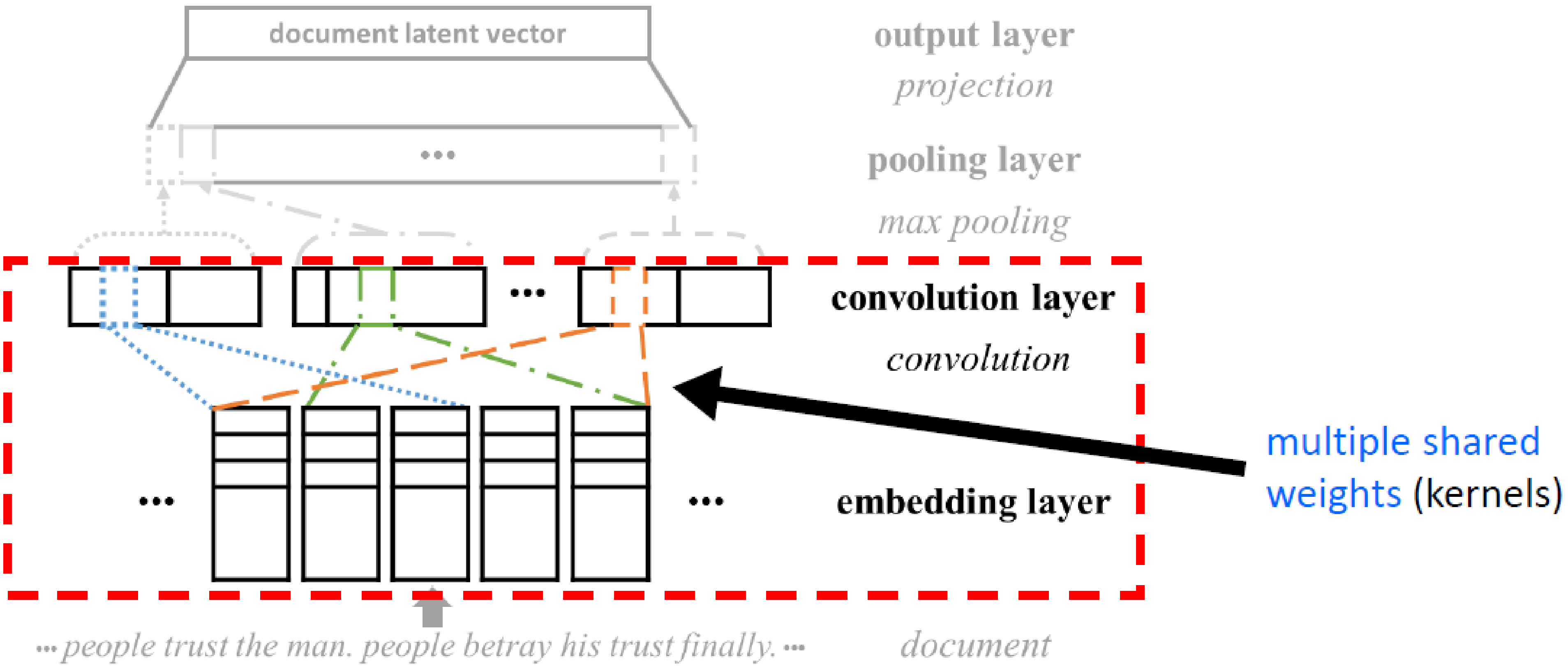
Embedding Layer – Word Embedding



(*pre-trained*) word embedding models.

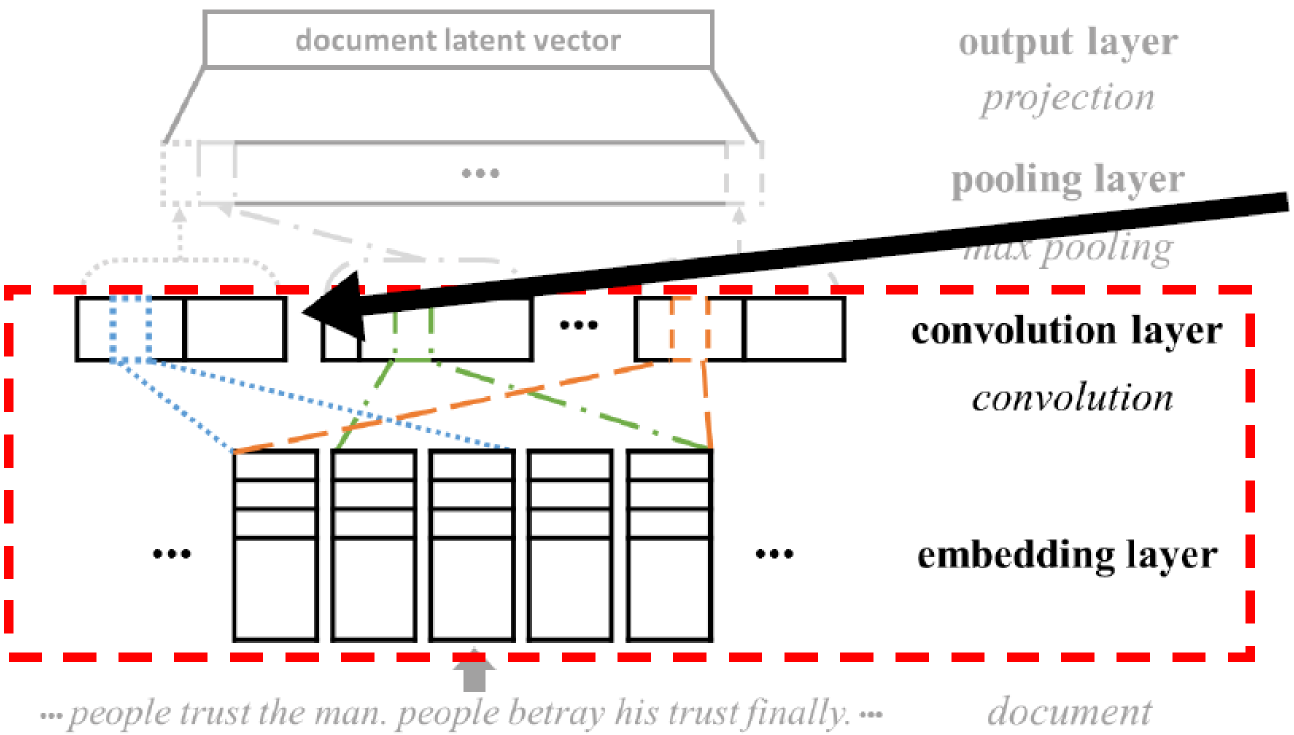
Convolution Layer – Contextual information

- Extract contextual features from a document matrix



Convolution Layer – Contextual information

- Example (window size: 3)



$c = [c_1, c_2, \ldots, c_i, \ldots, c_{l-ws+1}]$

c_2

... people betrav his trust finally ...

c_3

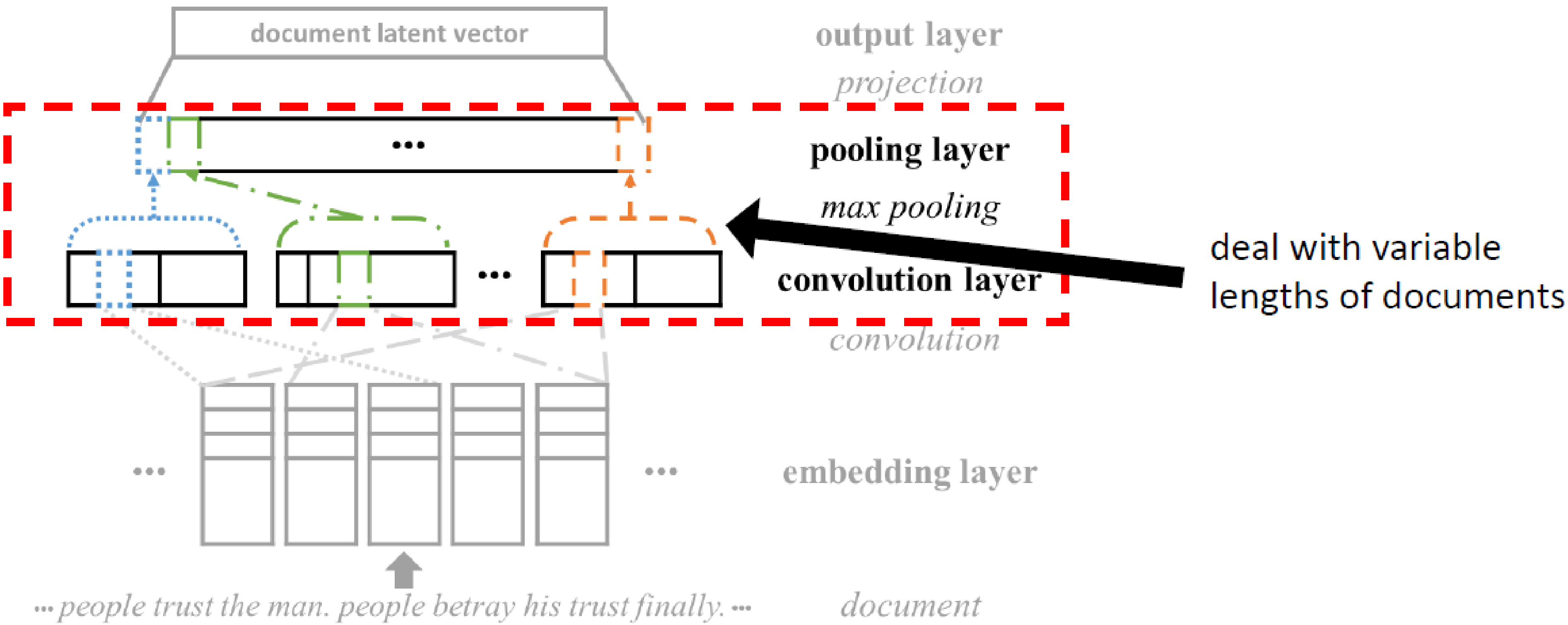
... people betray his trust finally ...

c_4

... people betray his trust finally ...

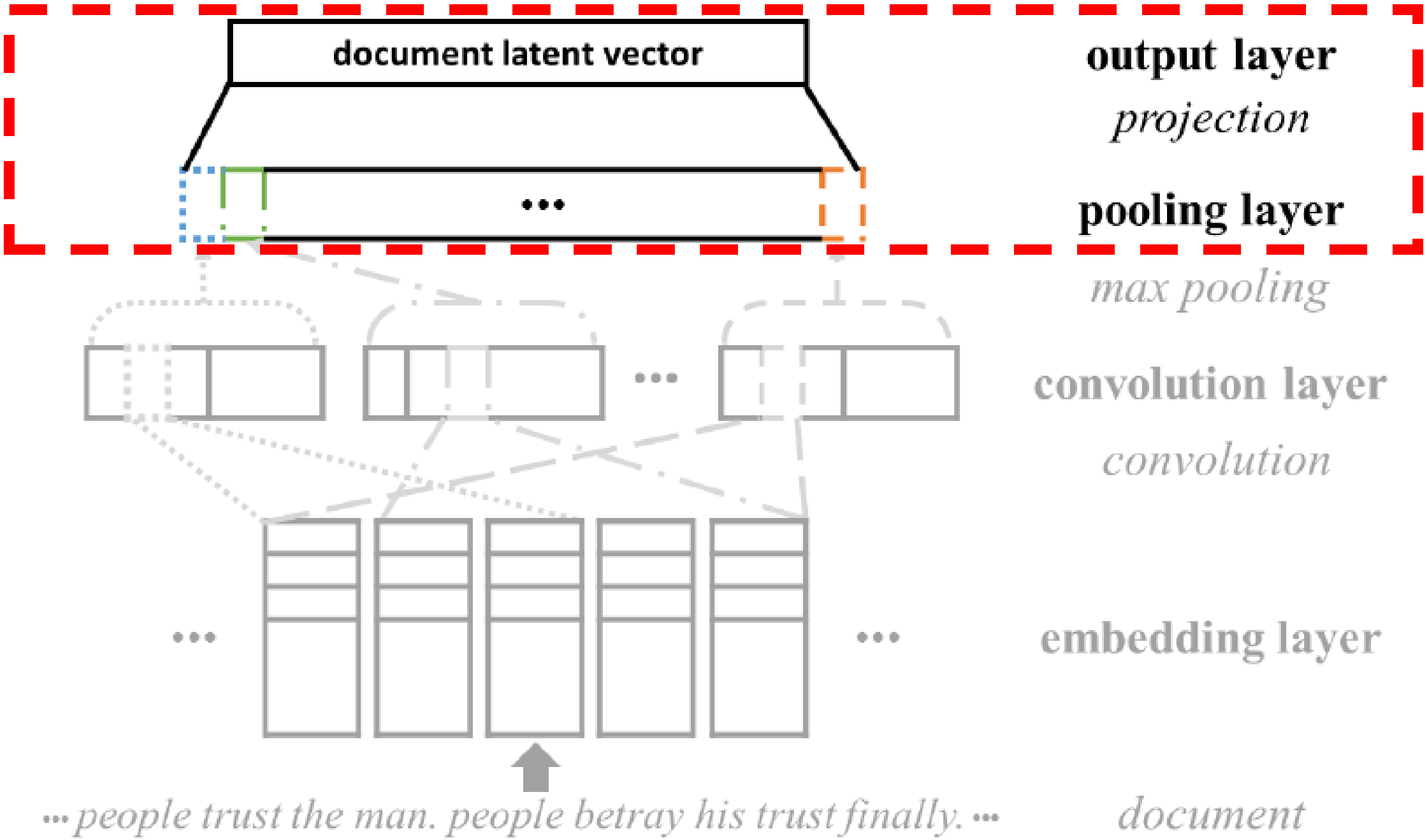
Pooling Layer – Representative Information

- Extract representative features from the convolution layer



Output Layer – High Level Features of Documents

- Project representative features to a k-dim. space



Objective Function

$$\mathcal{L}(U,V,W) = \sum_{i}^{N}\sum_{j}^{M}\frac{I_{ij}}{2}(r_{ij} - u_i^T v_j)^2 + \frac{\lambda_U}{2}\sum_{i}^{N}\|u_i\|_2$$

$$+ \frac{\lambda_V}{2}\sum_{j}^{M}\|v_j - cnn(W, X_j)\|_2 + \frac{\lambda_W}{2}\sum_{k}^{|w_k|}\|w_k\|_2,$$

Performance

- RMSE – training/valid/test dataset (80%/10%/10%)

ConvMF and ConvMF+ achieve significant improvements on all the datasets.

Model			V
PMF	0.8971 (0.0020)	0.8311 (0.0010)	1.4118 (0.0105)
CTR	0.8969 (0.0027)	0.8275 (0.0004)	1.5496 (0.0104)
CDL	0.8879 (0.0015)	0.8186 (0.0005)	1.3594 (0.0139)
ConvMF	**0.8531** (0.0018)	**0.7958** (0.0006)	**1.1337** (0.0043)
ConvMF+	**0.8549** (0.0018)	**0.7930** (0.0006)	**1.1279** (0.0073)
Improve	3.92%	2.79%	16.60%

Improvement by pre-trained word embedding

extremely sparse dataset!

Outline

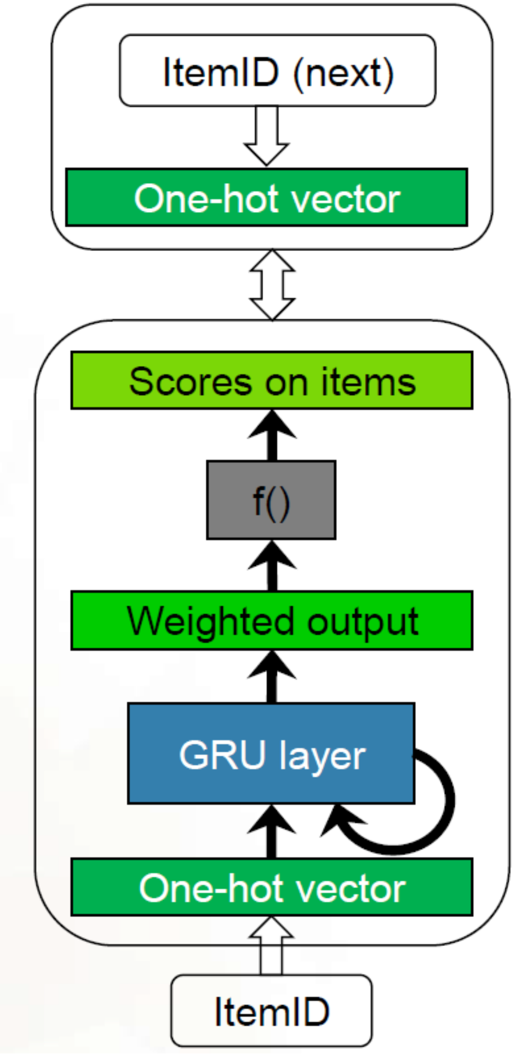
- ☑ MLP Based System
- ☑ AE Based System
- ☑ CNN Based System
- ☐ **RNN Based System**

Hidasi et al., Session-based Recommendations with Recurrent Neural Networks, ICLR'16

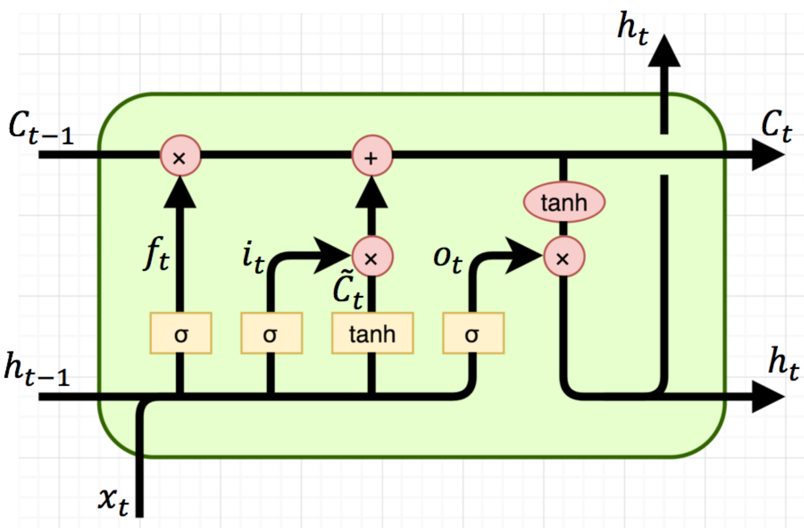
GRU4Rec

- GRU-based recommender system
- GRU trained on session data, adapted to the recommendation task
  - Input: current item ID
  - Hidden state: session representation
  - Output: likelihood of being the next item

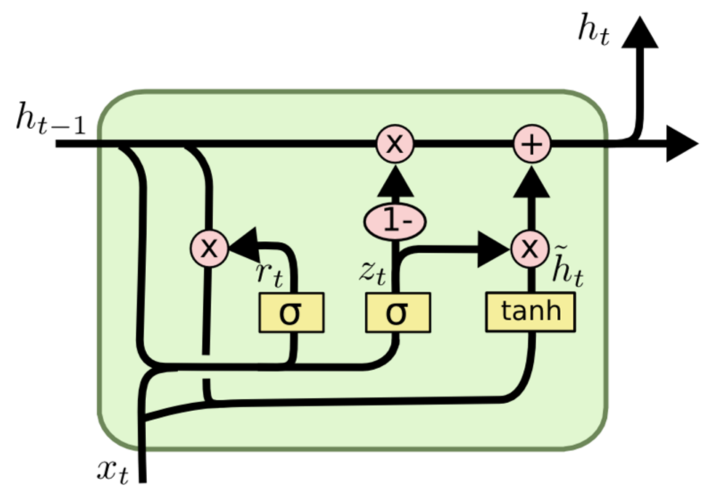
GRU4Rec


ItemID (next)
One-hot vector
Scores on items
f()
Weighted output
GRU layer
One-hot vector
ItemID

LSTM vs. GRU



(a) Long Short-Term Memory



(b) Gated Recurrent Unit

$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right)$$
$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right)$$
$$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right)$$
$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right)$$
$$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right)$$
$$h_t = \tanh(C_t) * o_t$$

$$z_t = \sigma\left(x_t U^z + h_{t-1} W^z\right)$$
$$r_t = \sigma\left(x_t U^r + h_{t-1} W^r\right)$$
$$\tilde{h}_t = \tanh\left(x_t U^h + (r_t * h_{t-1}) W^h\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

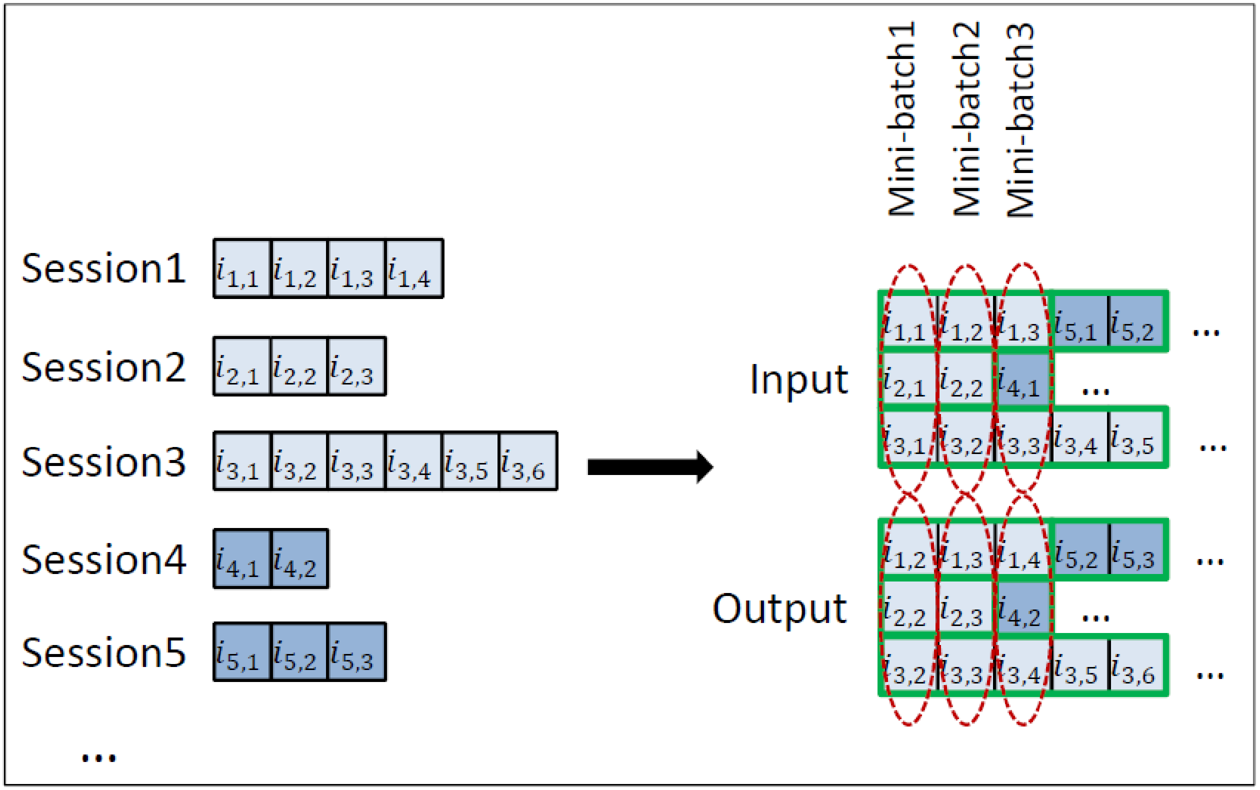
r: reset gate
z: update gate

Key Ideas of GRU4Rec

- Session-parallel mini-batches
- Output sampling
- Loss functions: cross-entropy, BPR, TOP1

Session-parallel Mini-batches

- Mini-batch is defined over sessions
  - Lots of sessions are very short
  - Mix long and short sessions using session-parallel mini-batches



Output Sampling

- Computing scores for all items (100k – 1m) in every step is slow
- One positive item (target) + several negative samples
- Which negative samples to choose?
  - Missing event = dislike?
  - The more popular an item is, the more probable that the use knows about it, and thus it is more likely that a missing event expresses dislike
- Solution: scores on mini-batch targets
  - Use items from the other training examples of the mini-batch as negative examples
    - This is popularity-based sampling!
    - Further reduce computational times by skipping the sampling
    - Implementation side: make the code less complex to faster matrix operations

Loss Functions

- Cross-entropy
- BPR (Bayesian Personalized Ranking)
  - $L_s = -\frac{1}{N_S}\sum_{j=1}^{N_S} \log \sigma(\hat{r}_{s,i} - \hat{r}_{s,j})$
  - N_S: sample size, i: desired item, j: negative samples
- TOP1
  - $L_s = \frac{1}{N_S}\sum_{j=1}^{N_S} I\{\hat{r}_{s,j} > \hat{r}_{s,i}\}$
  - Approximate I{·} with sigmoid; however this is unstable as certain positive items also act as negative examples, and thus scores tend to become increasingly higher
  - To avoid the problem, add regularization term:

$$L_s = \frac{1}{N_S}\sum_{j=1}^{N_S} \sigma\left(\hat{r}_{s,j} - \hat{r}_{s,i}\right) + \sigma\left(\hat{r}_{s,j}{}^2\right)$$

Experimental Results

Table 3: Recall@20 and MRR@20 for different types of a single layer of GRU, compared to the best baseline (item-KNN). Best results per dataset are highlighted.

Loss / #Units	RSC15		VIDEO	
	Recall@20	MRR@20	Recall@20	MRR@20
TOP1 100	0.5853 (+15.55%)	0.2305 (+12.58%)	0.6141 (+11.50%)	0.3511 (+3.84%)
BPR 100	0.6069 (+19.82%)	0.2407 (+17.54%)	0.5999 (+8.92%)	0.3260 (-3.56%)
Cross-entropy 100	0.6074 (+19.91%)	0.2430 (+18.65%)	0.6372 (+15.69%)	0.3720 (+10.04%)
TOP1 1000	0.6206 (+22.53%)	**0.2693 (+31.49%)**	**0.6624 (+20.27%)**	**0.3891 (+15.08%)**
BPR 1000	**0.6322 (+24.82%)**	0.2467 (+20.47%)	0.6311 (+14.58%)	0.3136 (-7.23%)
Cross-entropy 1000	0.5777 (+14.06%)	0.2153 (+5.16%)	–	–

GRU4Rec: Summary

- GRU-based recommender system
- GRU trained on session data, adapted to the recommendation task
  - Input: current item ID
  - Hidden state: session representation
  - Output: likelihood of being the next item
- Key ideas
  - Session-parallel mini-batches
  - Output sampling
  - Loss functions: cross-entropy, BPR, TOP1

News Recommendation with RNN

Koo et al., Accurate News Recommendation Coalescing Personal and Global Temporal Preferences, PAKDD 2020

Overview

- News recommendation on online news service
- News data patterns
  - *Popularity/Freshness* patterns
- News Recommendation Coalescing Personal and Global Temporal Preferences (PGT)
  - How well does PGT exploit news data patterns to provide accurate news recommendation?

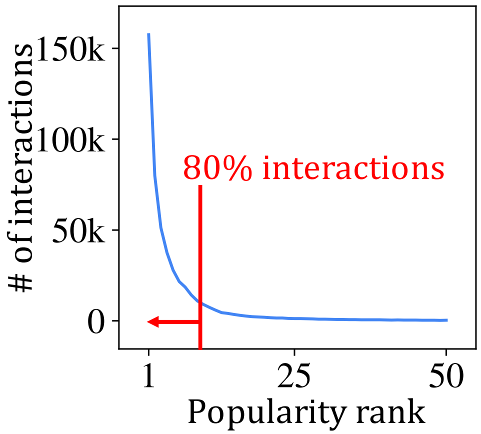
Online News Service

- Online news service
  - Thousands of news everyday
  - Millions of users

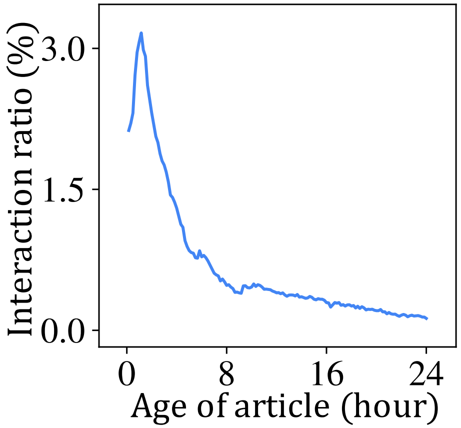
- Challenges
  - **Newly published** news articles everyday!
    - The cold-start problem
  - News recommendation considering users' interests
    - Individual personal preference
    - Time-dependent preference

Popularity/Freshness patterns

- News data patterns
  - Popularity pattern
    - Users mostly prefer popular news
  - Freshness pattern
    - # of interactions of news rapidly decreases over age



(a) Popularity pattern



(b) Freshness pattern

Problem Definition

- **Input**
  - News watch history of each user u
  - Candidate news articles at time t
  - Contents of news article
- **Output**
  - Ranks of candidates for each user u at time t
- **To Address/Consider**
  - The cold-start problem
  - Popularity/Freshness patterns of news

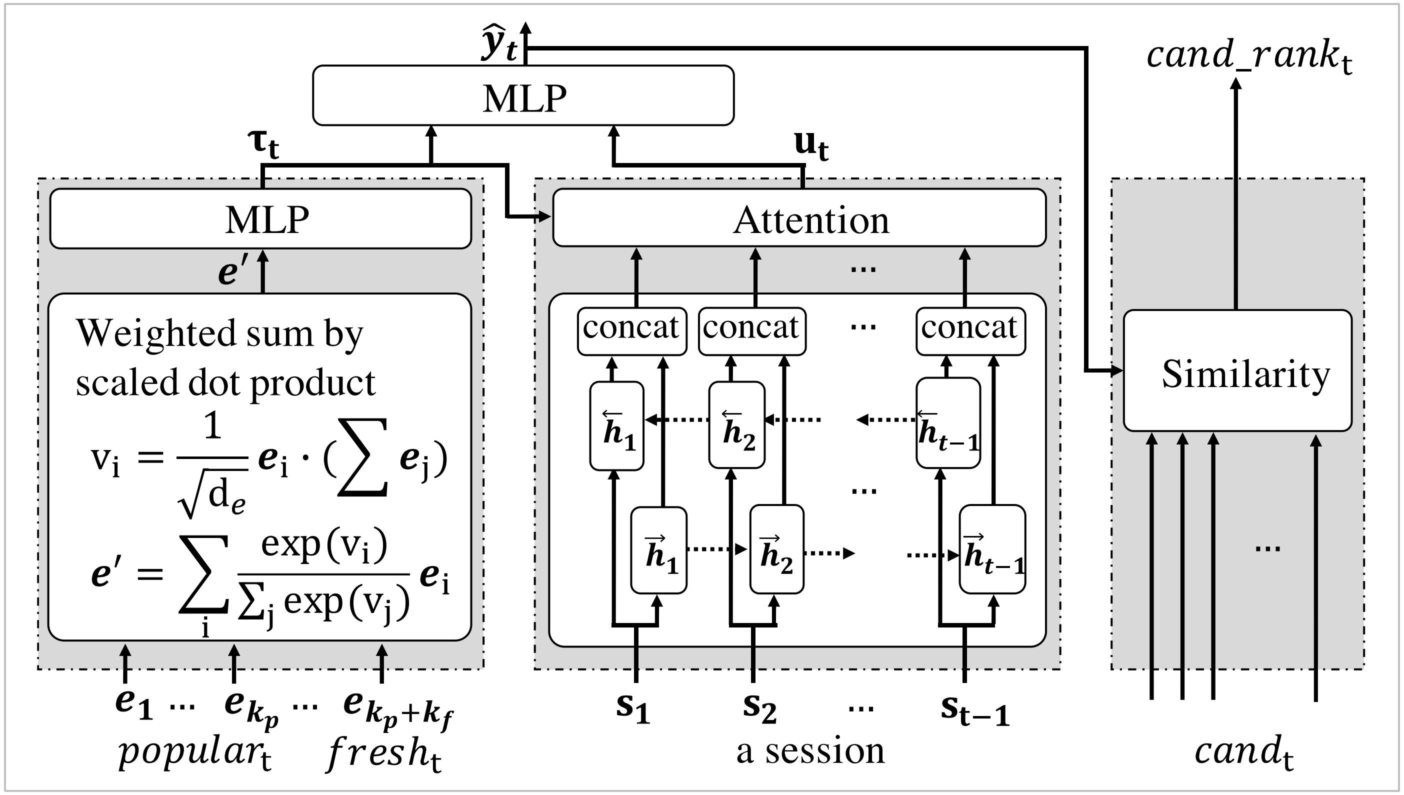
Proposed Method

- **PGT** (News Recommendation Coalescing **P**ersonal and **G**lobal **T**emporal Preferences)

- **Main intuition**
  - Global temporal preference
    - Comprehensive preference of all users at recommendation time
  - Attention network for the personal preference
    - To deal with a quick change of personal preference
    - The global temporal preference vector is used as context

Proposed Method

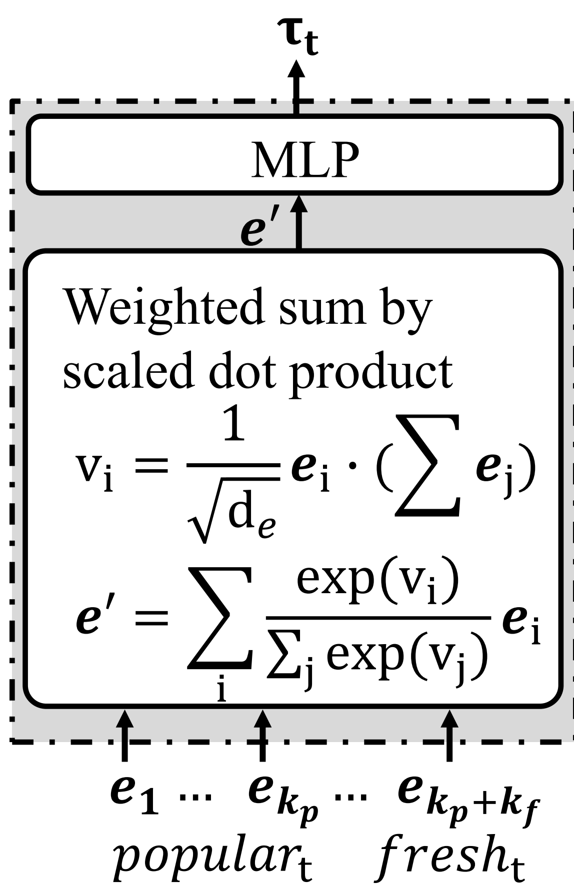
- Overview of PGT



(a) Global temporal preference (b) Personal preference (c) Ranking candidates

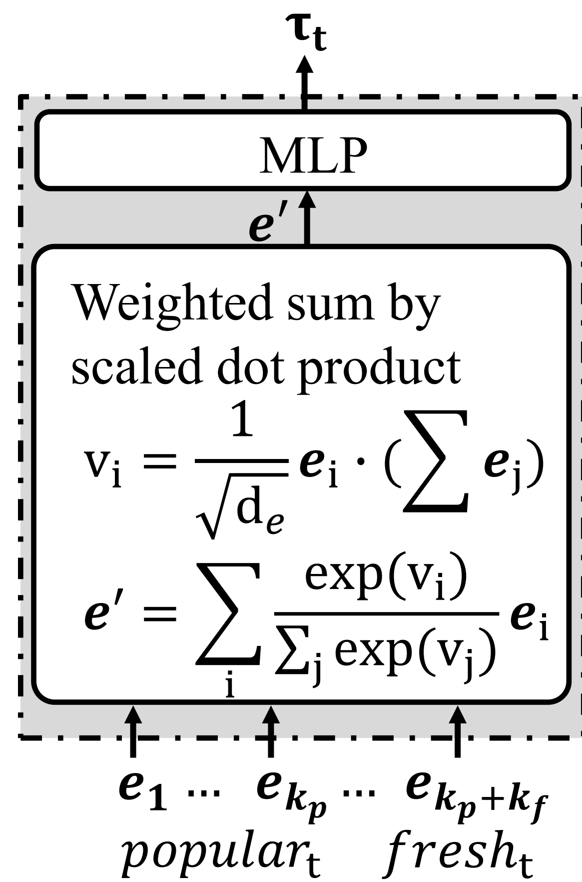
Proposed Method

- Global temporal preference
  - **Intuition**
    - Comprehensive preference of all users at recommendation time t
    - Extract time-dependent features
      - To deal with popularity and freshness patterns
    - To recommend newly published articles well
      - To better handle the cold-start problem



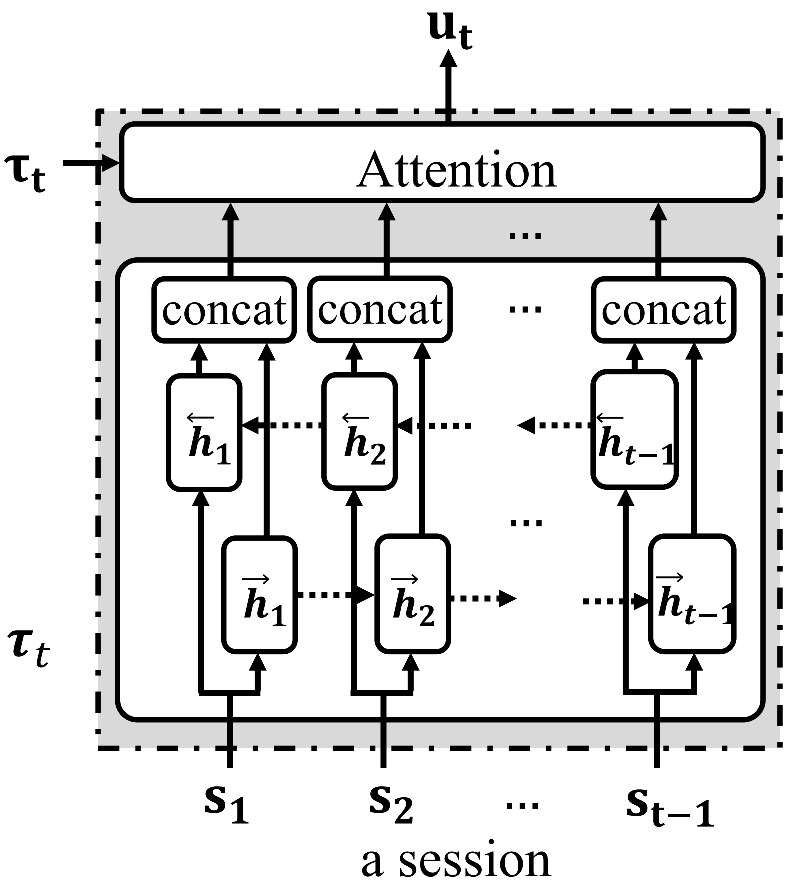
Proposed Method

- Global temporal preference
  - **Input**
    - $\boldsymbol{e}_1, \dots, \boldsymbol{e}_{k_p}, \dots, \boldsymbol{e}_{k_p+k_f}$
      - Popular/Fresh articles
      - k_p: # of popular articles
      - k_f: # of fresh articles
  - **Output**
    - $\boldsymbol{\tau}_t$: Global temporal preference
  - **How**
    - Weighted sum by attention network
      - v_i: unnormalized attention score



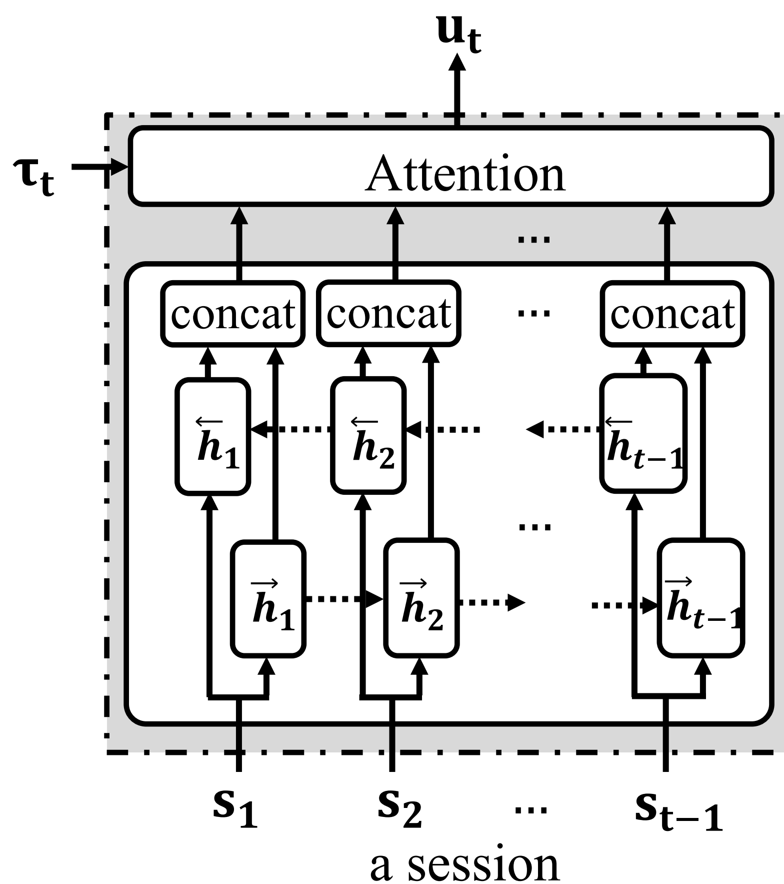
Proposed Method

- Personal preference
  - **Intuition**
    - Individual personal preference from previous user behaviors
    - Highlight important behaviors using the attention network
      - Time-dependent highlighting by $\boldsymbol{\tau}_t$



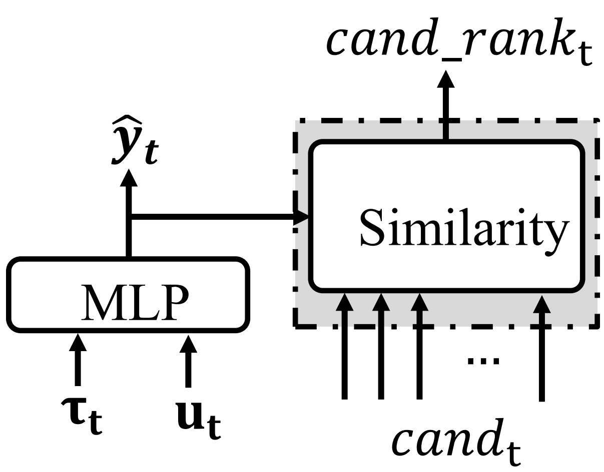
Proposed Method

- Personal preference
  - **Input**
    - $\boldsymbol{s}_1, \boldsymbol{s}_2, \dots, \boldsymbol{s}_{t-1}$
      - Previous watches of a user
    - $\boldsymbol{\tau}_t$: global temporal preference
  - **Output**
    - $\boldsymbol{u}_t$: personal preference
  - **How**
    - Bidirectional RNN
    - Weighted sum of hidden states
      - By attention network using $\boldsymbol{\tau}_t$ as context



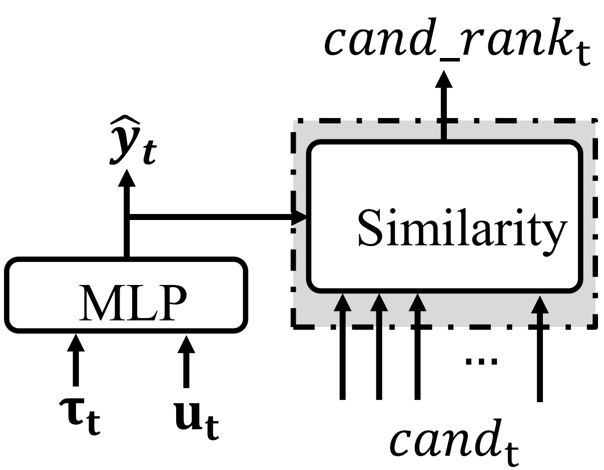
Proposed Method

- Ranking candidates
  - **Intuition**
    - Generate prediction vector $\hat{\boldsymbol{y}}_t$ from two preferences $\boldsymbol{\tau}_\mathbf{t}$, and $\mathbf{u}_\mathbf{t}$
    - Scores each candidate articles by utilizing $\hat{\boldsymbol{y}}_t$, then ranks candidates
  - **Similarity**
    - Inverse of L2 distance between $\hat{\boldsymbol{y}}_t$ and candidate article vector



Proposed Method

- Ranking candidates
  - **Input**
    - $\boldsymbol{\tau}_t$: global temporal preference
    - $\boldsymbol{u}_t$: personal preference
    - $cand_t$: candidate articles
  - **Output**
    - $cand\_rank_t$: rank of candidates
  - **How**
    - Measure the similarity between prediction vector $\widehat{\boldsymbol{y}}_t$ and candidate article vector



Experimental Question

- Q1. **Accuracy** on news recommendation
- Q2. **Effect of** modeling the **global temporal preference**
- Q3. **Effect of** modeling the **attention network** in modeling personal preference

Datasets

- Datasets
  - Adressa: user-news interaction of ‘Adresseavision’ in Norway
  - Globo: user-news interaction of ‘G1’ in Brazil
- Summary of datasets

Dataset	# Sessions	# Events	# Articles	Period
Adressa 1W[1]	112,405	487,961	11,069	7 days
Adressa 10W[1]	655,790	8,167,390	43,460	90 days
Globo[2]	296,332	2,994,717	46,577	16 days

[1]: http://reclab.idi.ntnu.no/dataset

[2]: https://www.kaggle.com/gspmoreira/news-portal-user-interactions-by-globocom

Competitors

- Competitors
  - Only popularity
    - POP
  - RNN-based method
    - Park et al. [CIKM’17]
    - Okural et al. [SIGKDD’17]
  - 3-D CNN method
    - Weave&Rec [Khattar et al. CIKM’18]
  - Attention-based method
    - HRAM [Khattar et al. CIKM’18]
    - NPA [Wu, C. et al. SIGKDD’19]

Experimental Setup

- Training method
  - Divide data into training, validation, and test sets with ratio of 8:1:1 based on the interaction time
  - To maximize the similarity between 1) a prediction vector 2) and the corresponding selected article vector
    - PGT
      - Loss function: mean squared error (MSE) of two vector
      - Optimizer: Adam optimizer
    - Competitors: follow their best setting
  - Mini-batched inputs of size 512

Experimental Setup

- Metric
  - HR@5: Hit Rate
  - MRR@20: Mean Reciprocal Rank

$$HR@5 = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} |\{r_i | r_i <= 5\}|$$

$$MRR@20 = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} c_i, \qquad c_i = \begin{cases} \frac{1}{r_i}, & \text{if } r_i \leq 20 \\ 0, & \text{otherwise} \end{cases}$$

Q1. Accuracy

- Q1. How well does PGT recommend news articles?
  - PGT shows the best performance for all datasets

Dataset	Metric	POP	Park et al. [11]	Okura et al. [10]	Weave&Rec [5]	HRAM [4]	NPA [16]	PGT
Adressa 1W	HR@5	0.4988	0.4714	0.4569	0.4377	0.5347	0.6512	**0.8668**
	MRR@20	0.3291	0.3361	0.3341	0.3013	0.3452	0.4983	**0.6857**
Adressa 10W	HR@5	0.5672	0.3677	0.3477	0.3007	0.3941	0.5819	**0.7106**
	MRR@20	0.3735	0.2461	0.2320	0.2101	0.2531	0.3818	**0.6197**
Globo	HR@5	0.2845	0.3551	0.3537	-	0.4474	-	**0.5663**
	MRR@20	0.2001	0.2483	0.2500	-	0.3101	-	**0.5116**

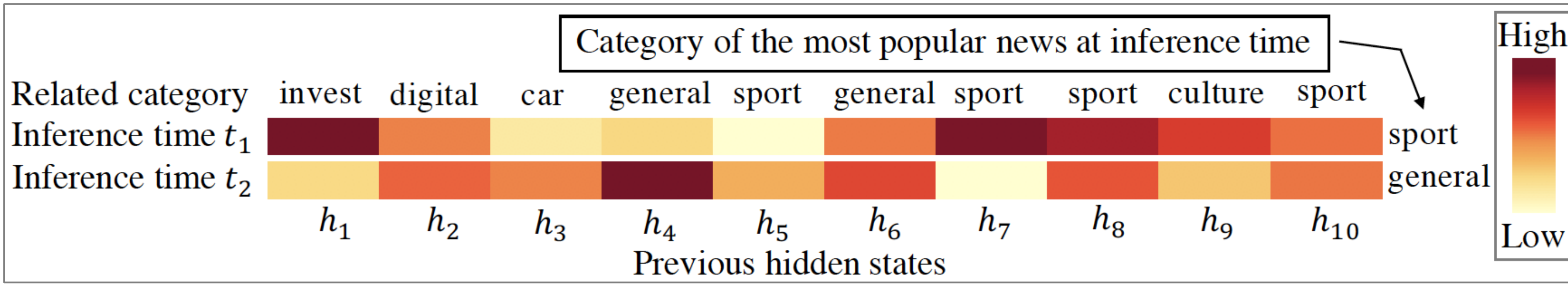
Q2. Effect of modeling the global temporal preference

- Q2. Does the modeling of **global temporal preference** help improve the accuracy?
  - PGT_{-T}: without the global temporal preference
  - PGT_{-A}: without the attention network of BiLSTM

Dataset	Metric	PGT$_{-T}$	PGT$_{-A}$	PGT
Adressa 1W	HR@5	0.6662	0.8497	**0.8668**
	MRR@20	0.5647	0.6756	**0.6857**
Adressa 10W	HR@5	0.6360	0.6946	**0.7106**
	MRR@20	0.5423	0.5610	**0.6197**
Globo	HR@5	0.5366	0.5562	**0.5663**
	MRR@20	0.4923	0.5035	**0.5116**

Q3. Effect of modeling the attention network in personal preference

- Case study of the attention network



  - **Different attention weights** to the same news watch history when the **inference time is changed**
  - When 'sport' or 'general' is popular
    - The attention network gives **more weights** to articles in **the same categories**

Summary

- Proposed **PGT** for recommendation on an online news service
  - To provide accurate recommendation
- **Main idea**: Let's extract time-dependent features by the **global temporal preference**
  - The global temporal preference and attention network in personal preference
    - Better handle the **popularity and freshness patterns** of news
    - Improve the accuracy compared to other competitors

What You Need to Know

- Deep recommender system
  - MLP Based System
  - AE Based System
  - CNN Based System
  - RNN Based System

Questions?