

Sequence-Aware Recommender Systems

Tutorial at TheWebConf 2019, San Francisco

Paolo Cremonesi | Politecnico di Milano, Italy

Massimo Quadrana | Pandora, USA

Dietmar Jannach | University of Klagenfurt, Austria

Agenda

- 09:00 – 09:45 Introduction & Problem Definition
- 09:45 – 10:30 Algorithms I
- 10:30 – 11:00 Coffee break
- 11:00 – 11:30 Algorithms II
- 11:30 – 12:00 Evaluation
- 12:00 – 12:20 Hands-on
- 12:20 – 12:30 Conclusion / Questions

Algorithms

Taxonomy

- Sequence Learning
 - Frequent Pattern Mining
 - Sequence Modeling
 - Distributed Item Representations
 - Supervised Models with Sliding Window
- Sequence-aware Matrix Factorization
- Hybrids
 - Factorized Markov Chains
 - LDA/Clustering + sequence learning
- Others
 - Graph-based, Discrete-optimization

Taxonomy

- Sequence Learning
 - Frequent Pattern Mining
 - Sequence Modeling
 - Distributed Item Representations
 - Supervised Models with Sliding Window
- Sequence-aware Matrix Factorization
- Hybrids
 - Factorized Markov Chains
 - LDA/Clustering + sequence learning
- Others
 - Graph-based, Discrete-optimization

Sequence Learning (SL)

- Useful in application domains where input data has an inherent sequential nature
 - Natural Language Processing
 - Time-series prediction
 - DNA modelling
 - **Sequence-Aware Recommendation**

Frequent Pattern Mining (FPM)

1. Discover **user consumption patterns**
 - Association rules, (Contiguous) Sequential Patterns
2. Look for patterns matching partial transactions
3. Rank items by confidence of matched rules

Size 1	Size 2	Size 3
$< A >$ (5)	$< A, B >$ (4)	$< A, B, E >$ (4)
$< B >$ (6)	$< A, C >$ (4)	$< A, E, C >$ (4)
$< C >$ (4)	$< A, E >$ (4)	
$< E >$ (5)	$< B, C >$ (4)	
	$< B, E >$ (5)	
	$< C, E >$ (4)	

Table 3: Frequent Sequential Patterns
Credits ([Nakagawa and Mobasher, 2003](#))

Applications

- Page prefetching and recommendations
- Personalized FPM for next-item recommendation
- Next-app prediction

- Association Rule Mining
 - items co-occurring within the same sessions
 - no check on order
 - if you like A and B, you also like C (aka: learning to rank)
- Sequential Pattern Mining
 - Items co-occurring in the same order
 - no check on distance
 - If you watch A and later watch B, you will later watch C
- Contiguous Sequential Pattern Mining
 - Item co-occurring in the same order and distance
 - If you watch A and B one after the other, you will now watch C

- Two steps approach
 1. Offline: rule mining
 2. Online: rule matching (with current user session)
- Rules have
 - Support: number of examples (main parameter)
 - Confidence: conditional probability
- Lower thresholds -> fewer rules
 - few rules: difficult to find rules matching a session
 - many rules: noisy rules (low quality)

Frequent Pattern Mining (FPM)

- Pros
 - Easy to implement
 - Explainable predictions
- Cons
 - Choice of the minimum support/confidence thresholds
 - Data sparsity
 - Limited scalability

[C. Lu et al] Mining mobile application sequential patterns for usage prediction. GrC '14

[Nakagawa and Mobasher] Impact of site characteristics on recommendation models based on association rules and sequential patterns. IJCAI '03

[Yap et al.] Effective next-items recommendation via personalized sequential pattern mining. DASFAA '12

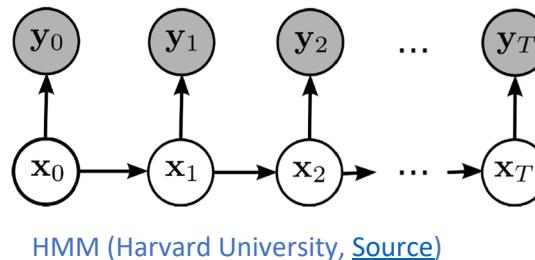
[Zhou et al.] An intelligent recommender system using sequential web access patterns. CIS '04

Sequence Modeling

- Sequences of past user actions as time series with **discrete** observations
 - Timestamps used only to **order** user actions (optionally to model time intervals)
- Aim to learn models from past observations to predict future ones
- Categories of SM models
 - Markov Models
 - Reinforcement Learning
 - Recurrent Neural Networks

Markov Models

- Stochastic processes over **discrete random variables**
 - Finite history (= order of the model)
→ user actions depend on a limited # of most recent actions
 - Extensions: Variable Order MM / Context Trees, HMMs



- Applications
 - Online shops
 - Playlist generation
 - **Variable Order Markov Models** for news recommendation
 - **Hidden Markov Models** for contextual next-track prediction

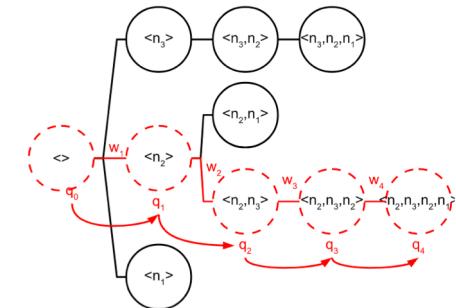


Figure 2: VMM context tree for the sequence $s = (n_1, n_2, n_3, n_2)$. Nodes in red-dashed are active experts $\mu \in A(s)$.

Credits (Garcin et al, 2013)

[Garcin et al.] Personalized news recommendation with context trees. RecSys '13,

[He et al.] Web query recommendation via sequential query prediction. ICDE '09

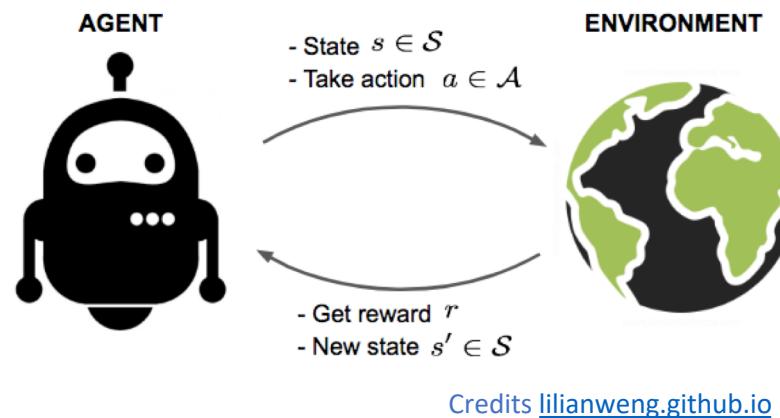
[Hosseinzadeh Aghdam et al.] Adapting recommendations to contextual changes using hierarchical hidden markov models. RecSys '15

[McFee and Lanckriet] The natural language of playlists. ISMIR '11

[Shani et al.] An MDP-based recommender system. J. Mach. Learn. Res. 2005.

Reinforcement Learning

- Learn by **sequential interactions** with the **environment**
- Generate recommendations (**actions**) and collect user feedback (**reward**)
- Markov Decisions Processes (MDPs)



Credits lilianweng.github.io

- Applications
 - Online e-commerce services
 - Sequential relationships between the attributes of items explored in a user session

[Moling et al.] Optimal radio channel recommendations with explicit and implicit feedback. RecSys '12

[Shani et al.] An MDP-based recommender system. J. Mach. Learn. Res. 2005.

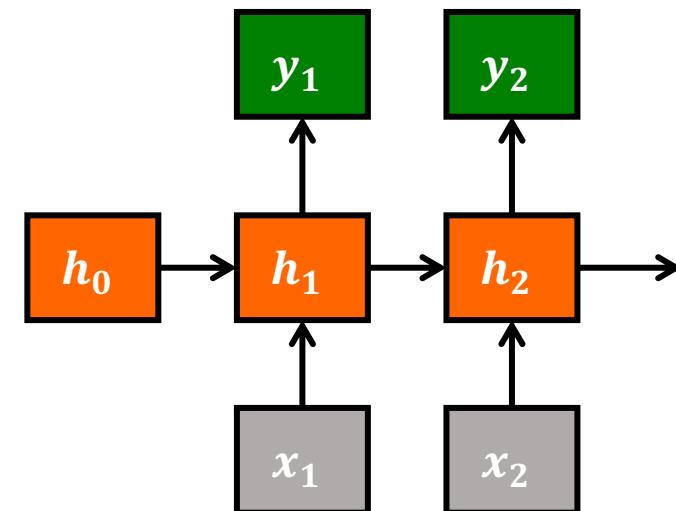
[Tavakol and Brefeld] Factored mdps for detecting topics of user sessions. RecSys '14

Recurrent Neural Networks (RNN)

- Distributed real-valued hidden state models with non-linear dynamics
 - Hidden state: latent representation of user state within/across sessions
 - Update the hidden state on the current input and its previous value, then use it to predict the probability for the next action
- Applications
 - Next-click prediction with RNNs
 - Session-based recommendation
 - Long-term/short-term user modeling

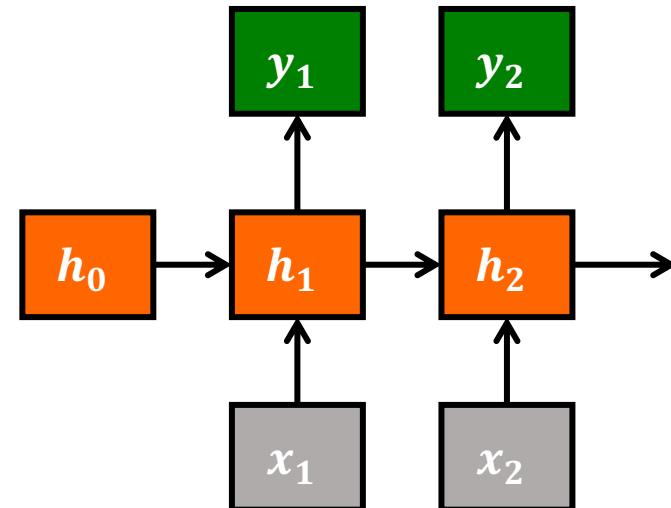
Simple Recurrent Neural Network

- Hidden state → used to predict the output
 - Computed from next input and previous hidden state



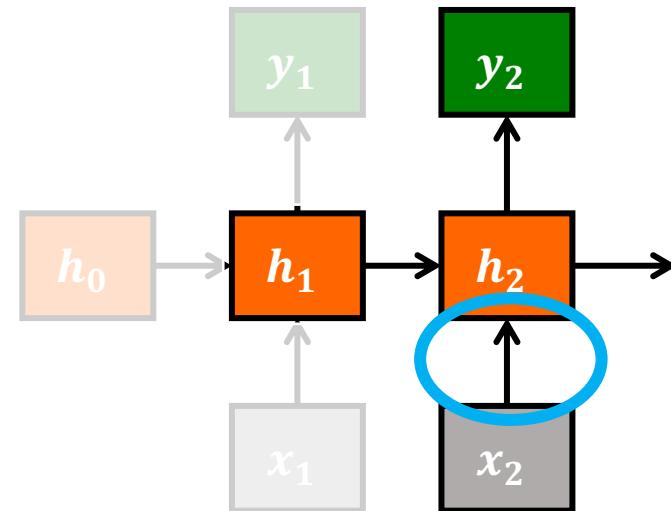
Simple Recurrent Neural Network

- Hidden state → used to predict the output
 - Computed from current input and previous hidden state
- Three weight matrices
 - $h_t = f(x_t^T W_x + h_{t-1}^T W_h + b_h)$
 - $y_t = f(h_t^T W_y)$



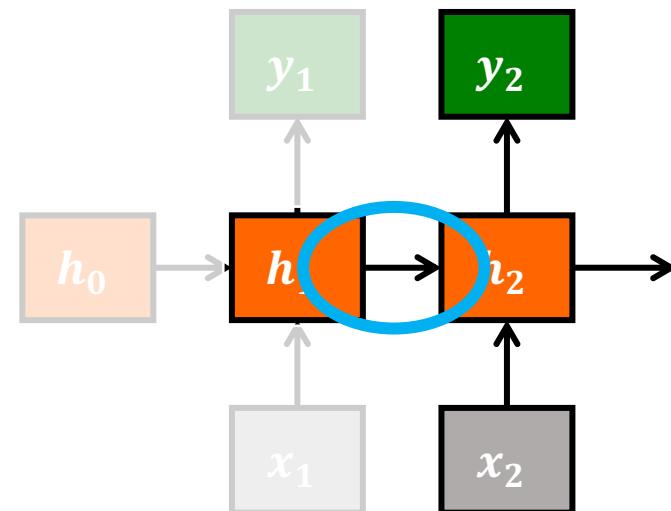
Simple Recurrent Neural Network

- Hidden state → used to predict the output
 - Computed from next input and previous hidden state
- Three weight matrices
 - $h_t = f(x_t^T W_x + h_{t-1}^T W_h + b_h)$
 - $y_t = f(h_t^T W_y)$



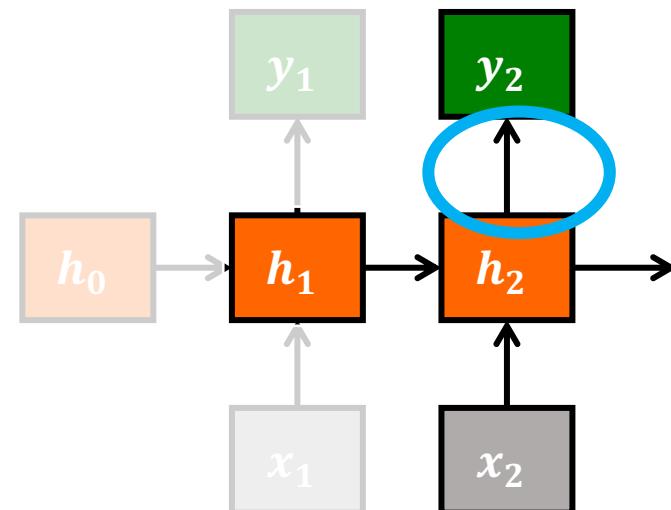
Simple Recurrent Neural Network

- Hidden state → used to predict the output
 - Computed from next input and previous hidden state
- Three weight matrices
 - $h_t = f(x_t^T W_x + h_{t-1}^T W_h + b_h)$
 - $y_t = f(h_t^T W_y)$



Simple Recurrent Neural Network

- Hidden state → used to predict the output
 - Computed from next input and previous hidden state
- Three weight matrices
 - $h_t = f(x_t^T W_x + h_{t-1}^T W_h + b_h)$
 - $y_t = f(h_t^T W_y)$



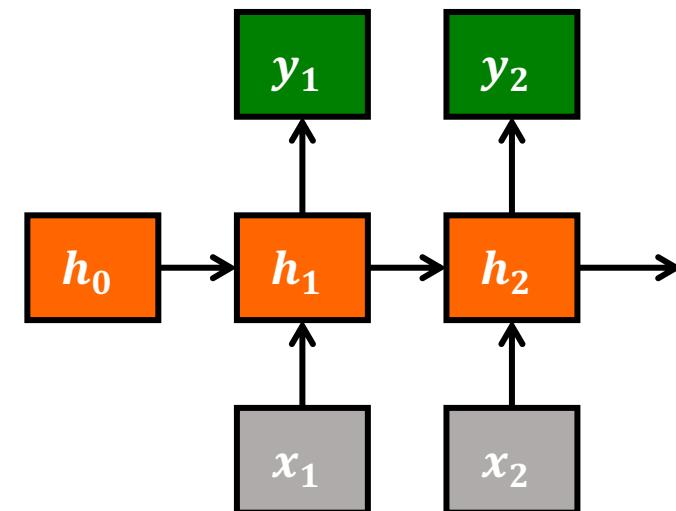
Simple Recurrent Neural Network

- Item subject to user interaction as **1-of-N** coded vector
- Example:
 - Output: item 4

$$y_2 = \begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

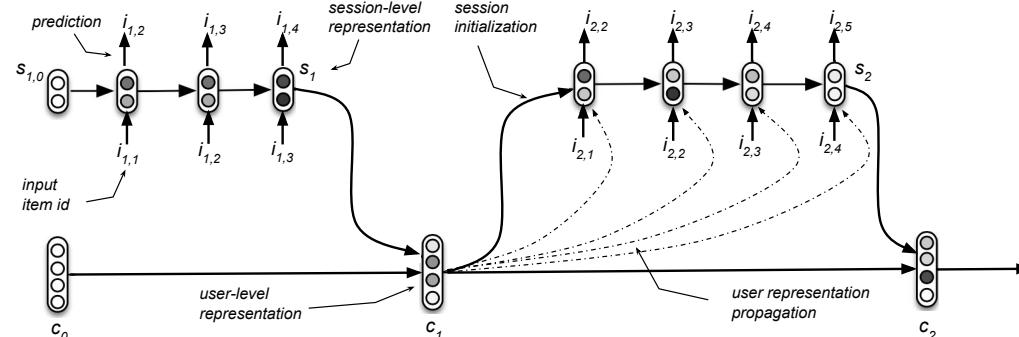
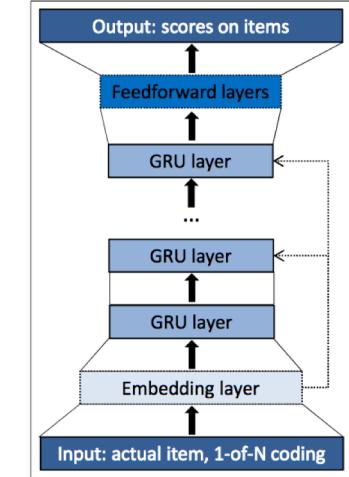
- Input: item 3

$$x_2 = \begin{array}{|c|c|c|c|c|} \hline 0 & 0 & 1 & 0 & 0 \\ \hline \end{array}$$



RNNs for Session-based Recommendation

- GRU4Rec/GRU4Rec+ [Hidasi, 2018]
 - Gated Recurrent Units (GRU), 1-hot item encoding, parallel mini-batching
 - Optimized **negative item sampling** and **ranking loss function**
- HGRURec [Quadrana, 2017]
 - Hierarchical RNN, **long+short-term** user profiling, **personalized** in-session recommendations
 - Bonus: **Time intervals** between sessions with Point Processes [Vassøy, 2019]



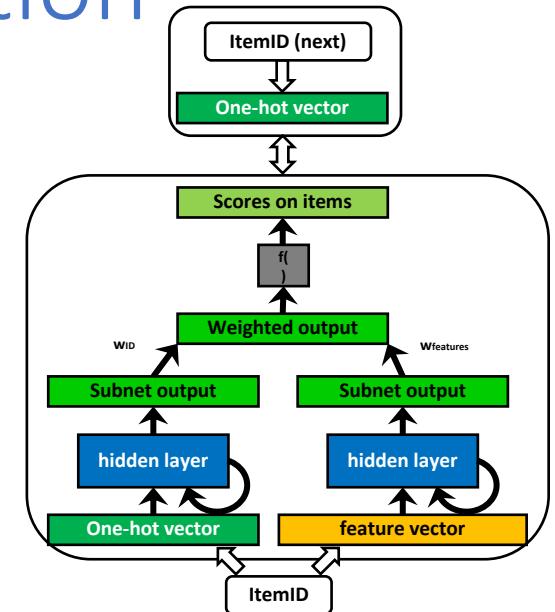
[Quadrana et al.] Personalizing session-based recommendations with hierarchical recurrent neural networks. RecSys 2017

[Hidasi and Karatzoglou] Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. CIKM 2018

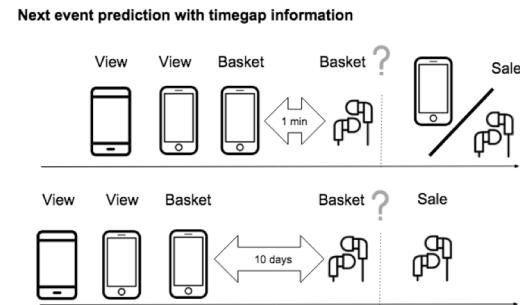
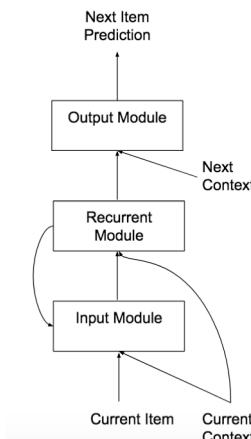
[Vassøy et al.] Time is of the Essence: a Joint Hierarchical RNN and Point Process Model for Time and Item Predictions. WSDM 2019

RNNs for Session-based Recommendation

- Parallel RNNs [Hidasi, 2016]
 - Jointly modelling **item features** and **identifiers**
 - **Alternated training** procedure to learn robust feature representations
 - Experiments with product **images** and **descriptions**

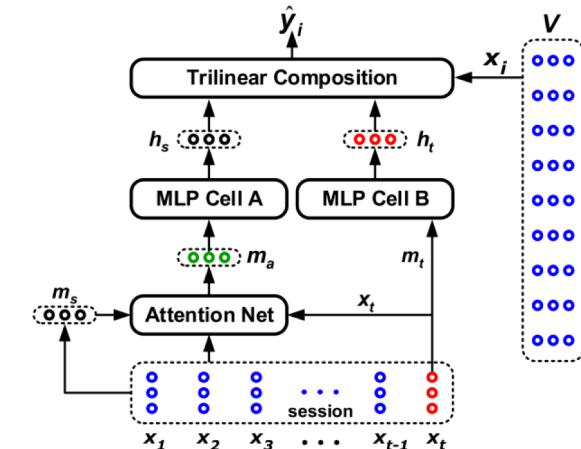
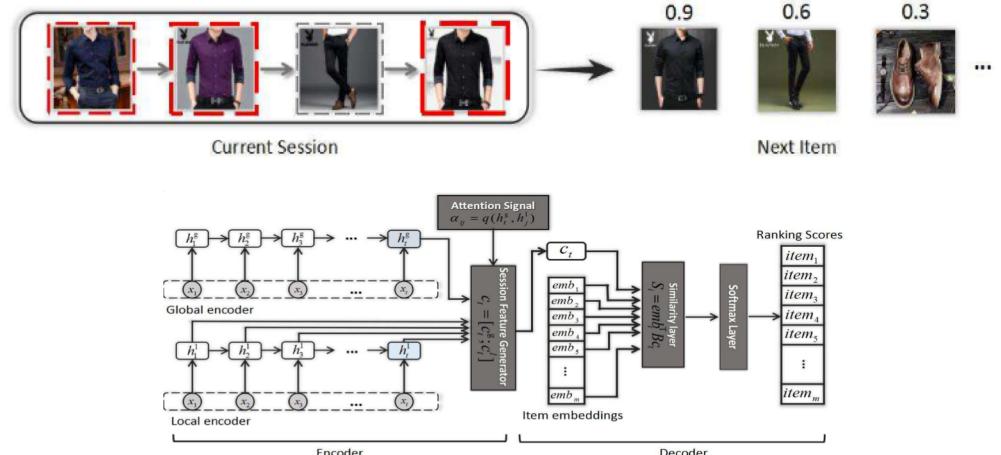


- Contextual Sequence Modeling [Smirnova, 2017]
 - Temporal ctx, action type, etc.
 - Concatenation $[x_t; c_t]$
 - Mult. interaction $x_t \odot Cc_t$
 - Combined $[x_t \odot Cc_t; c_t]$



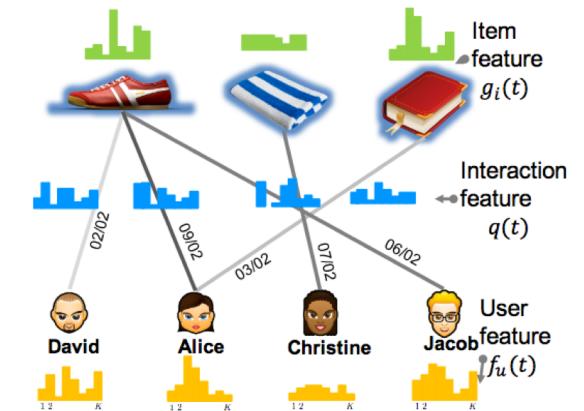
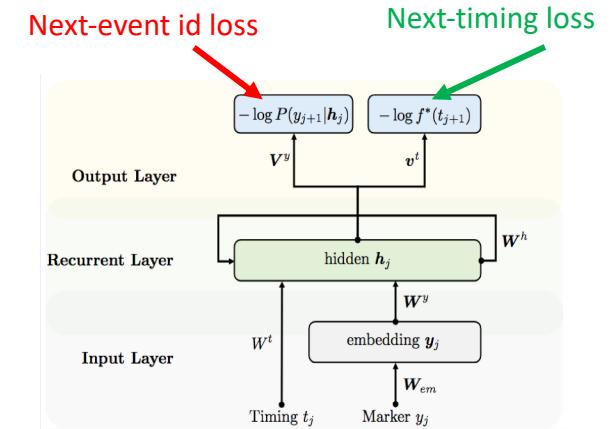
RNNs for Session-based Recommendation

- Neural Attention Networks [Liu, 2017]
 - Standard RNN → **global** session repr.
 - Attention RNN → **main purpose** of the session
 - Combine local/global features with bi-linear matching scheme
- Attentive memory networks [Liu, 2018]
 - Replace RNN with **attention/memory module**
 - Memory retains short-term user interests
 - **Weighted attention** to memory
 - Only Feed-forward networks!



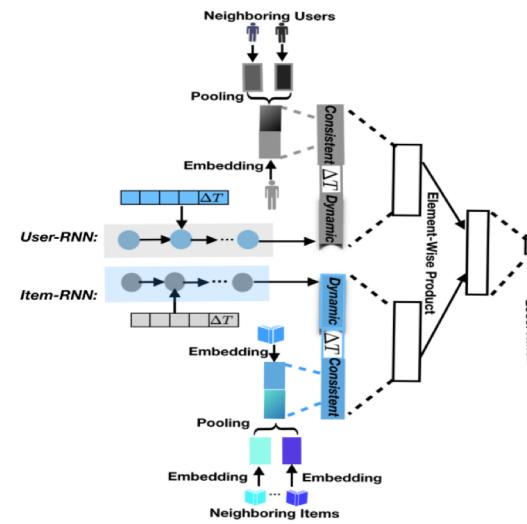
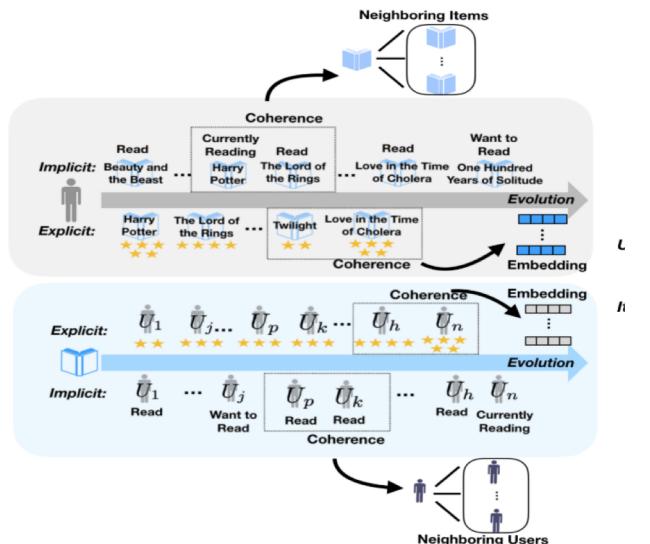
Other RNN Applications

- Recurrent Marked Temporal Point Process (RNN) [Du, 2016]
 - Joint learning **next target item** and **its timing** with a combination of Hawkes process + RNNs
 - Predict **when and what** event will likely occur next
- Deep Co-evolutionary networks [Dai, 2016]
 - Jointly model **user-item feature co-evolution** and **influence each-other** over time



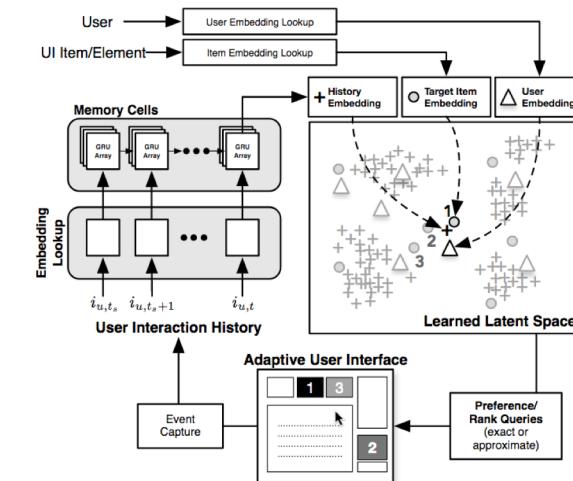
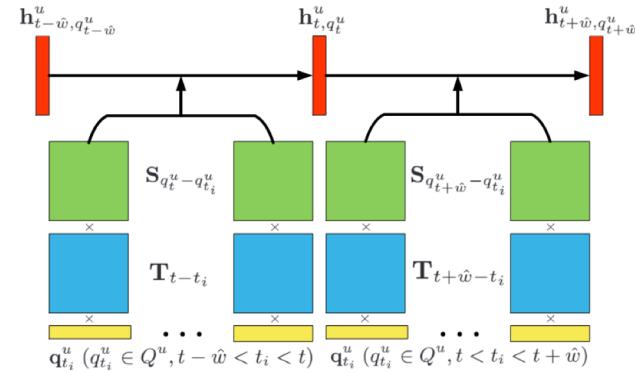
Other RNN Applications

- Recurrent Recommendation with Local Coherence
 - Short-term user/item neighborhood are **coherent** (users who rate or read the same book in a shorter time period are more similar)
 - Coherence-based neighbors with skip-grams
 - Dynamic latent-factors with parallel user/item RNN



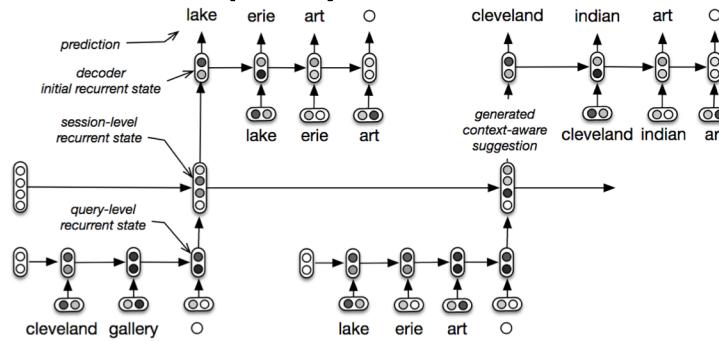
Other RNN Applications

- Spatio-Temporal RNN (ST-RNN) [Liu, 2016]
 - Next-location prediction with custom Spatial and Temporal hidden-to-hidden transition matrices
- Sequential recommendation for dynamic adaptation of user interfaces [Soh, 2017]

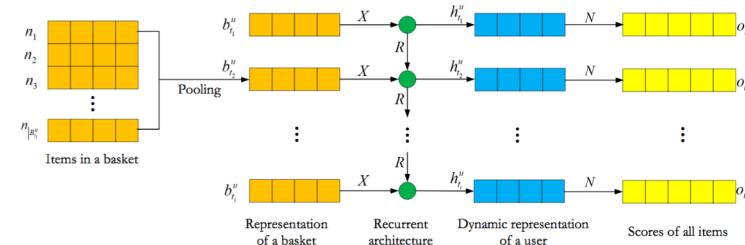


Other RNN Applications

- Generative context-aware next-query recommendation [Sordoni, 2015]
 - Hierarchical RNN to model query-level + session-level features

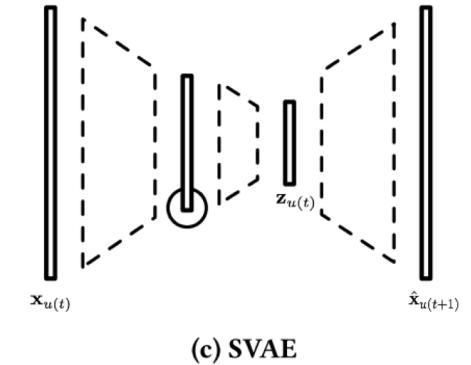


- Dynamic REcurrent bAsket Model (DREAM) [Yu, 2016]
 - Pooling to generate basket representations

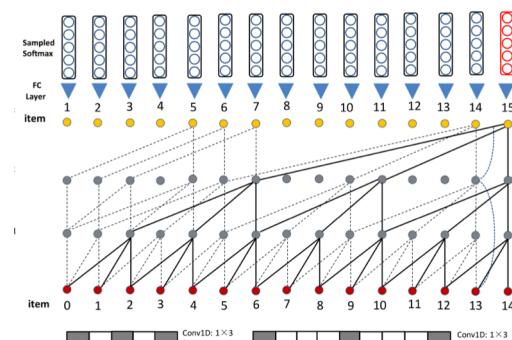
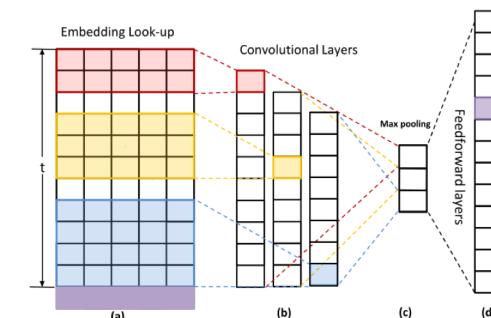


Other DL approaches

- Sequential Variational Autoencoders [Sachdeva, 2019]
 - VAE with recurrent dependency on the latent representation z .
- CNN-based methods
 - CASER [Tang and Wang, 2018]
 - Embedding, 1d conv, max pooling and FFD layers
 - NextItNet [Yuan, 2019]
 - Dilated convolutions



(c) SVAE



[Sachdeva, 2019] Sequential Variational Autoencoders for Collaborative Filtering. WSDM 2019

[Tang and Wang, 2018] Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. WSDM 2018

[Yuan et al., 2019] A Simple Convolutional Generative Network for Next Item Recommendation. WSDM 2019

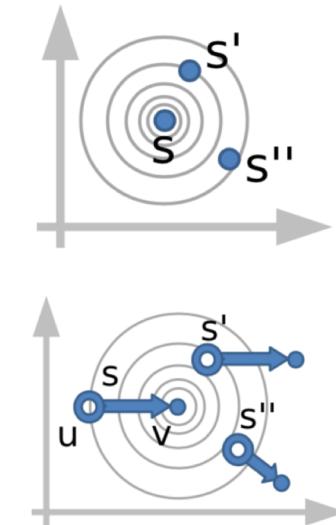
Distributed Item Representations

- Dense, lower-dimensional representations of items
 - derived from sequences of events
 - preserve the sequential relationships between items
- Similar to latent factor models
 - “similar” items are projected to similar vectors
 - every item is associated with a real-valued embedding vector
 - it is projected into a lower-dimensional space
 - certain item transition properties are preserved
 - e.g., co-occurrence of items in similar contexts
- Different approaches are possible
 - Latent Markov Embedding (LME)
 - Prod2Vec

Latent Markov Embeddings

- Transition probability related with the **Euclidean distance** between the **embeddings** of **subsequent** items

$$\Pr(p^{[i]}|p^{[i-1]}) = \frac{e^{-\|X(p^{[i]}) - X(p^{[i-1]})\|_2^2}}{\sum_{j=1}^{|S|} e^{-\|X(s_j) - X(p^{[i-1]})\|_2^2}}$$



- Symmetric transitions → **single-point model**
- Asymmetric transitions → **dual-point model**
- MLE over the existing sequences
- Recommendation through **sampling**
- Applications:
 - Playlist generation
 - POI recommendation

[Chen et al.] Playlist prediction via metric embedding. KDD '12 Multi-space probabilistic sequence modeling. KDD '13

[Feng et al.] Personalized ranking metric embedding for next new POI recommendation. IJCAI '15

[Wu et al.] Personalized next-song recommendation in online karaoke. RecSys '13

Prod2Vec

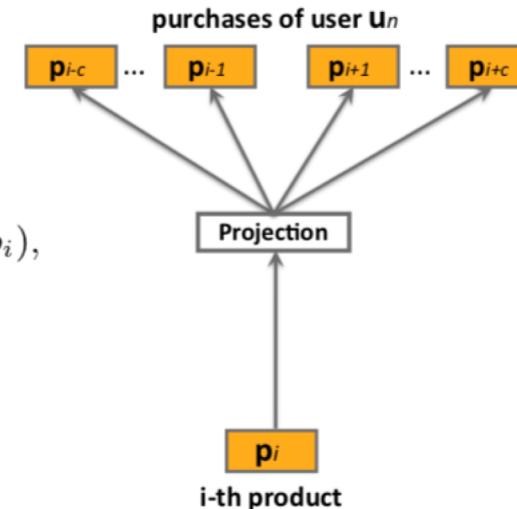
- Transition probability related to the **dot product** between the **embeddings** of items within the **context window**

$$\mathbb{P}(p_{i+j}|p_i) = \frac{\exp(\mathbf{v}_{p_i}^\top \mathbf{v}'_{p_{i+j}})}{\sum_{p=1}^P \exp(\mathbf{v}_{p_i}^\top \mathbf{v}'_p)},$$

- MLE + **Skip-gram**
- Recommendation with (decayed) **KNN**

$$\mathcal{L} = \sum_{s \in \mathcal{S}} \sum_{p_i \in s} \sum_{-c \leq j \leq c, j \neq 0} \log \mathbb{P}(p_{i+j}|p_i),$$

- Variants
 - Bagged-prod2vec [Grbovic, 2015]
 - Meta-prod2vec [Vasile, 2016]
 - User representation with paragraph vectors [Tagami, 2015][Grbovic, 2015]
- Applications
 - E-commerce, music, pretrained embeddings for other sequence models



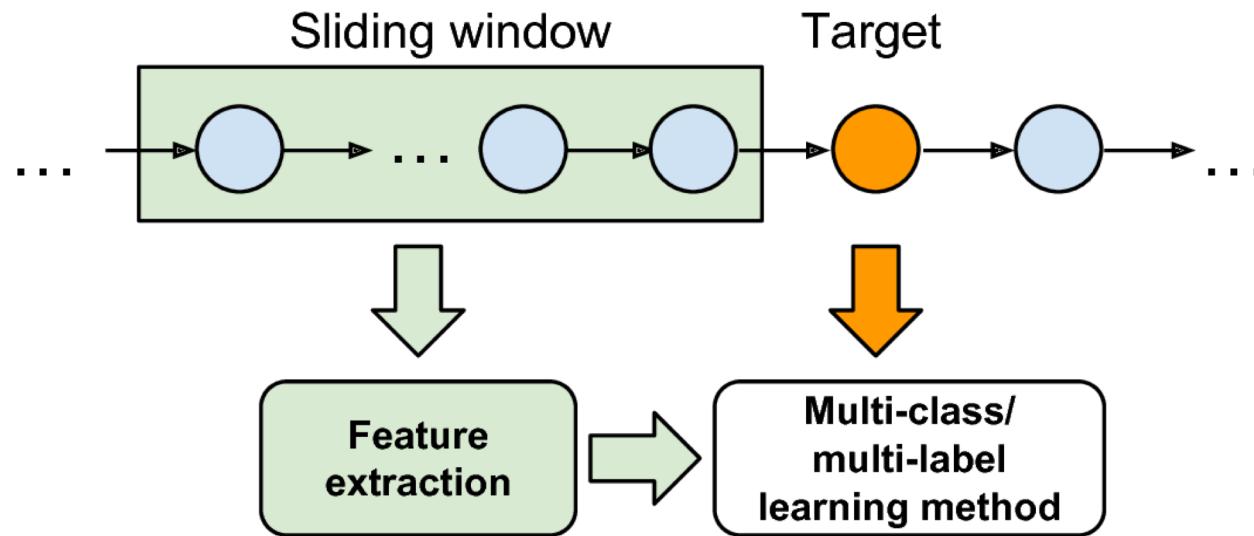
[Grbovic et al.] E-commerce in your inbox: Product recommendations at scale. KDD 2015

[Tagami et al.] Modeling user activities on the web using paragraph vector. WWW 2015

[Vasile et al.] Meta-prod2vec: Product embeddings using side-information for recommendation. RecSys '16

Supervised Learning w/ Sliding Windows

- Frame sequential recommendation as a classification problem

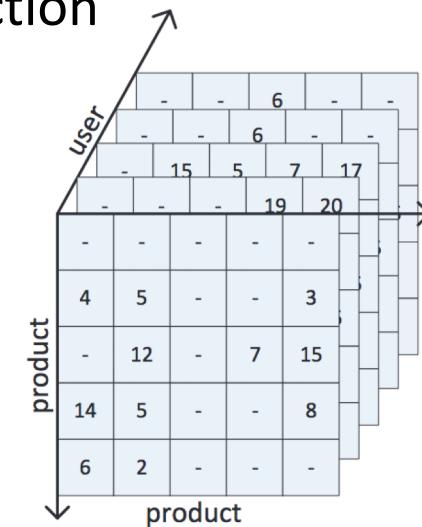
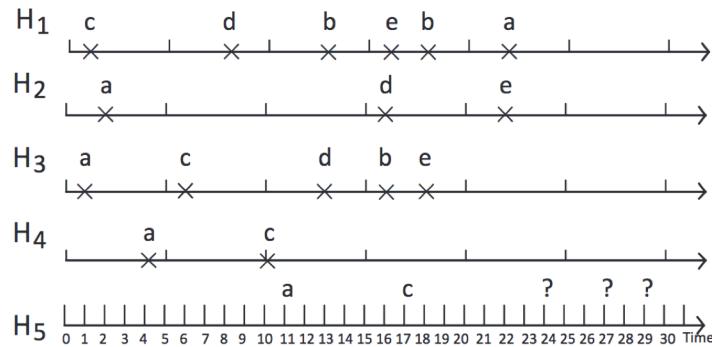


Taxonomy

- Sequence Learning
 - Frequent Pattern Mining
 - Sequence Modeling
 - Distributed Item Representations
 - Supervised Models with Sliding Window
- **Sequence-aware Matrix Factorization**
- Hybrids
 - Factorized Markov Chains
 - LDA/Clustering + sequence learning
- Others
 - Graph-based, Discrete-optimization

Sequence-aware Matrix Factorization

- Sequence information usually derived from timestamps
→ Time-aware recommender systems
- Purchase sequences → personalized purchase interval prediction
 - Framed as factorized (SVD) maximum utility prediction



[Zhao et al.] Increasing temporal diversity with purchase intervals. SIGIR '12

[Zhao et al.] Utilizing purchase intervals in latent clusters for product recommendation. SNAKDD '14

Taxonomy

- Sequence Learning
 - Frequent Pattern Mining
 - Sequence Modeling
 - Distributed Item Representations
 - Supervised Models with Sliding Window
- Sequence-aware Matrix Factorization
- Hybrids
 - Factorized Markov Chains
 - LDA/Clustering + sequence learning
- Others
 - Graph-based, Discrete-optimization

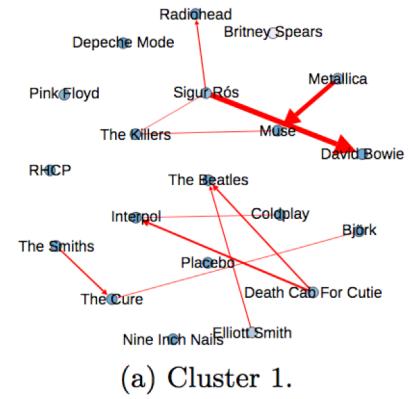
Hybrid Methods

- Sequence Learning + Matrix Completion (CF or CBF)
 - Factored Personalized Markov Chains (FPMC) [Rendle, 2010]
 - Transition cube factorization with Pairwise Loss
- Topic modelling/clustering → sequence learning
 - LDA on sequences → FPM on topic sequences [Hariri, 2012]
 - Clustering over order-1 Markov transition matrices (behavioural clustering) → personalized Page-rank [Natarajan, 2013]

user	?	?	?	?	?	?
?	?	?	?	?	?	?
1	0	1	0	0	?	?
0	1	1	0	0	?	0
0.5	1	0.5	0	0	?	1
0.5	1	0.5	0	0	?	1
?	?	?	?	?	?	?
?	?	?	?	?	?	?

from item

to item



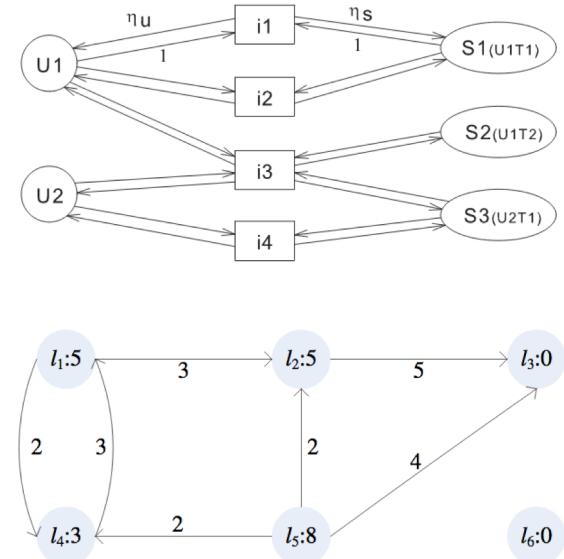
[Rendle et al.] Factorizing personalized markov chains for next-basket recommendation. WWW '10

[Hariri et al.] Context-aware music recommendation based on latent topic sequential patterns. RecSys '12

[Natarajan et al.] Which app will you use next?: Collaborative filtering with interactional context. RecSys '13

Other approaches

- Graph-based
 - Session-based Transition Graph [Xiang, 2010]
 - Location-location Transition Graph [Zhang, 2014]
- Discrete optimization via constraint satisfaction [Jannach, 2015][Paws, 2006][Xu, 2016]



[Xiang et al.] Temporal recommendation on graphs via long- and short-term preference fusion. KDD '10

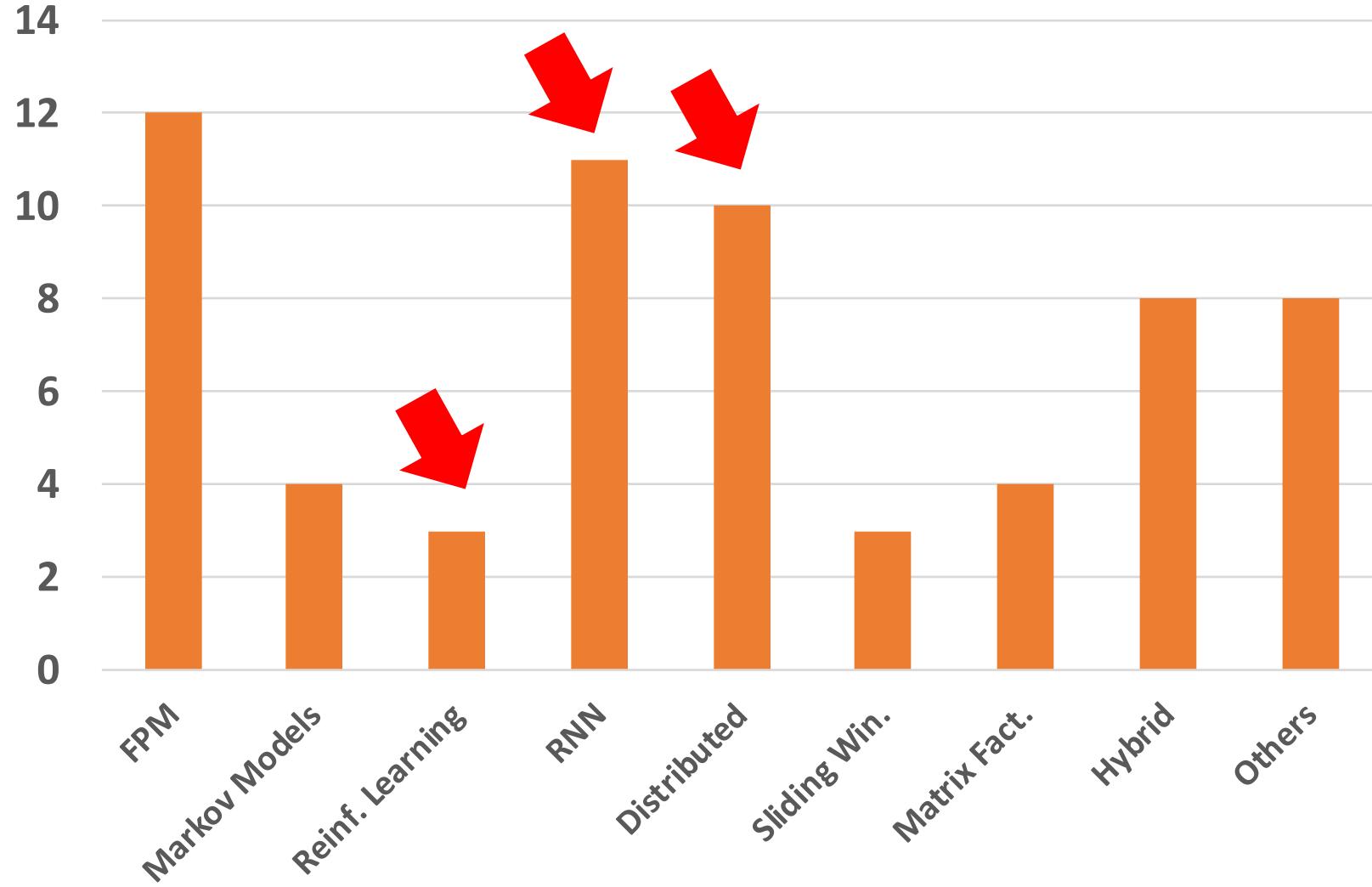
[Zhang et al.] Lore: Exploiting sequential influence for location recommendations. SIGSPATIAL '14

[Jannach et al.] Beyond "hitting the hits": Generating coherent music playlist continuations with the right tracks. RecSys '15

[Pauws et al.] Fast generation of optimal music playlists using local search. ISMIR '06

[Xu et al.] Personalized course sequence recommendations. IEEE Trans. Signal Process. 2016.

Statistics



Algorithm Summary

Algorithm	Main idea	Pros	Cons
FPM	Discover patterns in user action sequences	<ul style="list-style-type: none"> • Easy implementation • Explainable results 	<ul style="list-style-type: none"> • Complex configuration • Suffers from data sparsity • Limited scalability
MC	Compute transition probabilities over fixed-length sequences	<ul style="list-style-type: none"> • Explainable results 	<ul style="list-style-type: none"> • Fixed transition order • Suffers from data sparsity • Limited scalability
VMM	Compute transition probabilities over variable-length sequences	<ul style="list-style-type: none"> • Variable transition orders • Explainable results 	<ul style="list-style-type: none"> • Suffers from data sparsity
HMM	Model the causal factors in user sequences as transitions between <i>discrete</i> hidden states	<ul style="list-style-type: none"> • Learns from variable-length inputs • Robust to data sparsity 	<ul style="list-style-type: none"> • Limited explainability • Huge number of discrete parameters
RL	Directly maximize the customer and seller reward over time	<ul style="list-style-type: none"> • Dynamically adapt recommendations to future (unknown) rewards • Under active research 	<ul style="list-style-type: none"> • MDP-based approaches have same issues as MCs • Limited explainability
RNN	Model the causal factors in user sequences with <i>non-linear</i> transitions between <i>continuous</i> hidden states	<ul style="list-style-type: none"> • Learns from variable-length inputs • Learns long-term dependencies • Robust to data sparsity • Compact hidden states • Under active research 	<ul style="list-style-type: none"> • Complex configuration • Limited explainability • Benefits not fully clear in some domains
EMB	Embed items into latent spaces that preserves sequential transition properties	<ul style="list-style-type: none"> • Robust to data sparsity • Visually interpretable embeddings • Under active research 	<ul style="list-style-type: none"> • Need auxiliary methods to make recommendations • Limited explainability
SL	Use supervised learning over features extracted from fixed-size sliding windows over sequences	<ul style="list-style-type: none"> • Easy implementation • Use off-the-shelf supervised algorithms 	<ul style="list-style-type: none"> • Explainability depends on the chosen supervised method • Feature engineering
MF	Define new inputs and loss functions for MF to handle sequences	<ul style="list-style-type: none"> • Extensive literature available • Robust to data sparsity 	<ul style="list-style-type: none"> • Non-trivial input and loss design • Concerns regarding scalability

Algorithm: FPM: Frequent Pattern Mining, MC: Markov Chains, VMM: Variable-order Markov Models, HMM: Hidden Markov Models, RL: Reinforcement Learning, RNN: Recurrent Neural Networks, EMB: Distributed Item Representations, SL: Supervised Learning w/ Sliding Windows, MF: Matrix Factorization