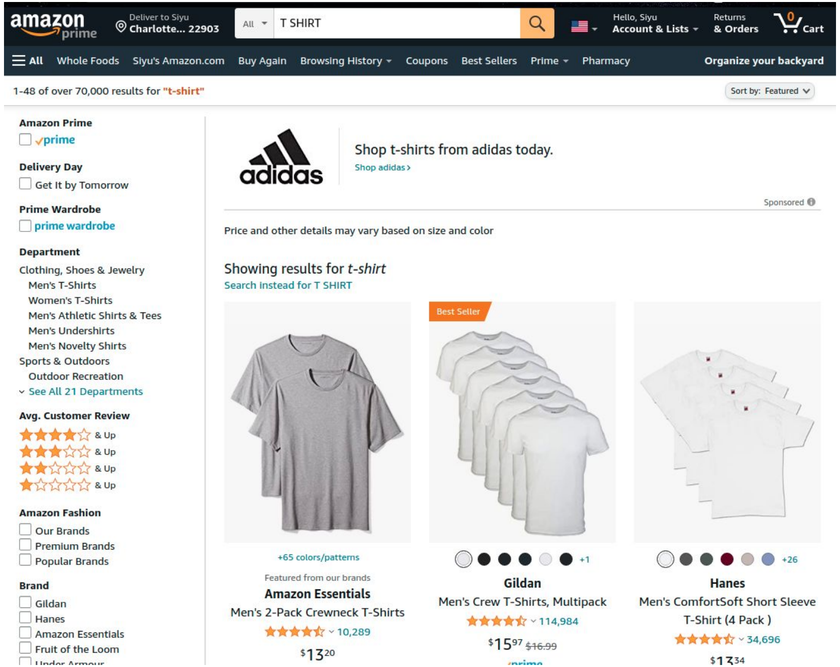
Neural Attentive Session-Based Recommendation

● Introduction and related work	*Siyu Jian*
● Method	*Veena Ramesh*
● Experimental setup	*Meng Hua*
● Analysis and conclusion	*Johannes Johnson*

Introduction:

- E-commercial scenarios we always want to provide the customer the best product they want
    - Guess what the customer want
    - Provide the best matched product

- Provide the product matches the customer's expectation is key to success
    - Provide better shopping experience
    - beat Business rival
    - **Rewards**



Introduction:

- E-commercial scenarios we always want to provide the customer the best product they want
  - Guess what the customer want
  - Provide the best matched product

- Provide the product matches the customer's expectation is to success
  - Provide better shopping experience
  - beat Business rival

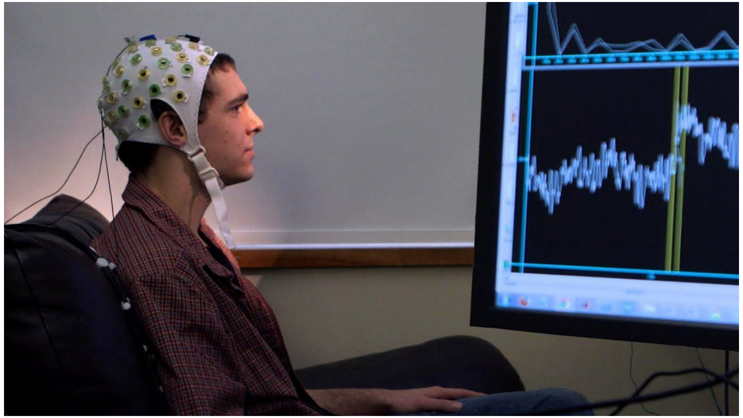
  - **Rewards** is sample !

Introduction:

- E-commercial scenarios we always want to provide the customer the best product they want
  - Guess what the customer want
  - Provide the best matched product

- Provide the product matches the customer’s expectation is key to success
  - Provide better shopping experience
  - beat Business rival
  - **Rewards** is sample !

- How to Guess What the customer intent
  - Customer Query Keywords
  - Click items
  - Time spend on the session
  - Clicks in this session
  - Customer click another item, start a new session



All those information will reflect what the customer’s mind

Related Works:

- **General Recommender**
  - Items that often clicked together
  - K-nearest neighbor approach

- **Sequential Recommender**
  - Markov chain

- **Deep Learning Based Method**
  - Neural network recommender is mostly focusing on the classical collaborative filtering
  - Deep neural networks
  - Recurrent Neural networks [Hidasi et al.]

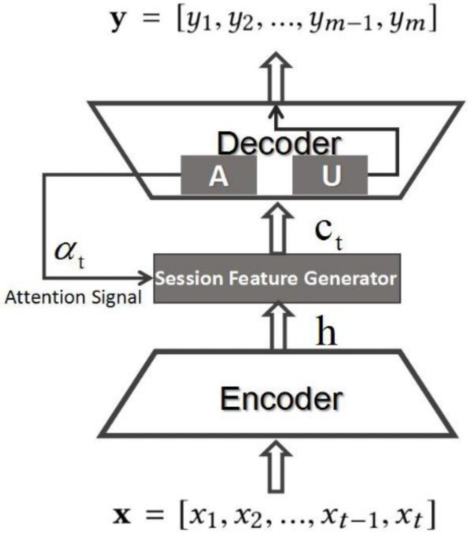
Neural **A**ttentive **R**ecommendation **M**achine (**NARM**)

- User sequential behavior
- Main purpose in the current session

Methods

Overview

- **Session based recommendation task:** predicting what the user would click next
    - Given: current sequential transaction data
    - Want**: Y** = $[y_1, y_2, y_3, ..., y_m]$
        - Each is the recommender score of an item
        - A ranked list of all of the items that occur in the session
- <u>**NARM**</u>
    - Build a representation of the current session and generate predictions based on it
        - Encoder
            - Global
            - Local
        - Decoder



Methods

Global Encoder

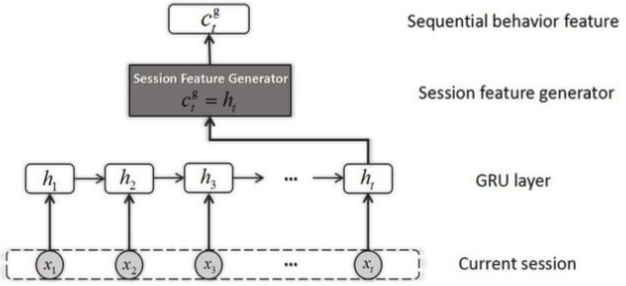
- Input: all previous clicks
- Output: feature of the user's sequential behavior in the current session

We are using a Recurrent Neural Network (**RNN**) with Gated Recurrent Units (**GRU**)

- *Hidsai et al*. determined that GRU can outperform LSTM units for session-based recommendation tasks
- GRU aims at dealing with the vanishing gradient problem
    - Activation is a linear interpolation between $\boldsymbol{h_{t-1}}$ and $\boldsymbol{h_t}$

Drawbacks

- Vectorial summarization of the whole sequence behavior cannot really capture the 'intention' of the user



(a) The graphical model of the global encoder in NARM, where the last hidden state is interpreted as the user's sequential behavior feature $c_t^g = h_t$.

Methods

Local Encoder

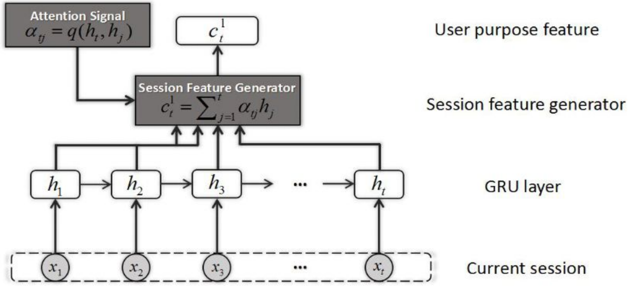
- To capture the ‘main purpose,’ we add **item-level attention mechanism**
- Weighted factors (alpha) determine which inputs should be ignored/emphasized
- *q* computes the similarity between $\boldsymbol{h_t}$ and $\boldsymbol{h_j}$

Focuses on more important items to capture the main purpose of the current session

$$\boldsymbol{c}_t^{l} = \sum_{j=1}^{t} \alpha_{tj} \boldsymbol{h}_j \ ,$$

$$\alpha_{tj} = q(\boldsymbol{h}_t, \boldsymbol{h}_j) \ .$$

$$q(\boldsymbol{h}_t, \boldsymbol{h}_j) = \boldsymbol{v}^{\mathrm{T}} \sigma(\boldsymbol{A}_1 \boldsymbol{h}_t + \boldsymbol{A}_2 \boldsymbol{h}_j) \ ,$$



(b) The graphical model of the local encoder in NARM, where the weighted sum of hidden states is interpreted as the user’s main purpose feature $\boldsymbol{c}_t^1 = \sum_{j=1}^{t} \alpha_{tj} \boldsymbol{h}_j$.

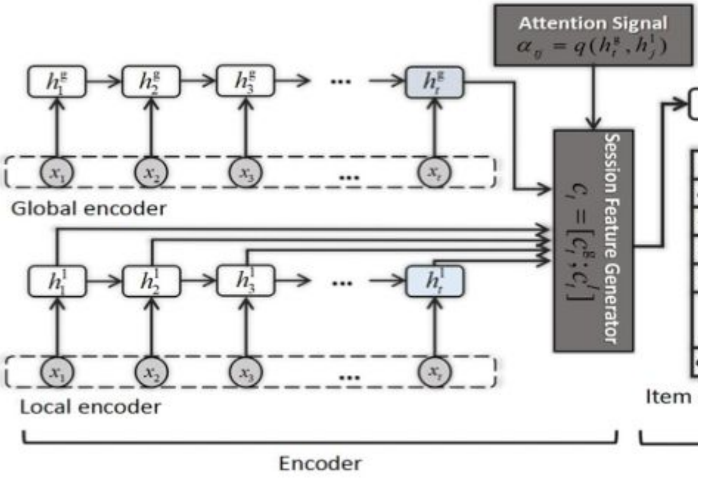
Methods

NARM Encoding

- **Global encoder:** summarized whole sequence behavior
- **Local encoder**: dynamically selected items that are important in the current session (main purpose)

We combine these encoders for an <u>extended representation</u>

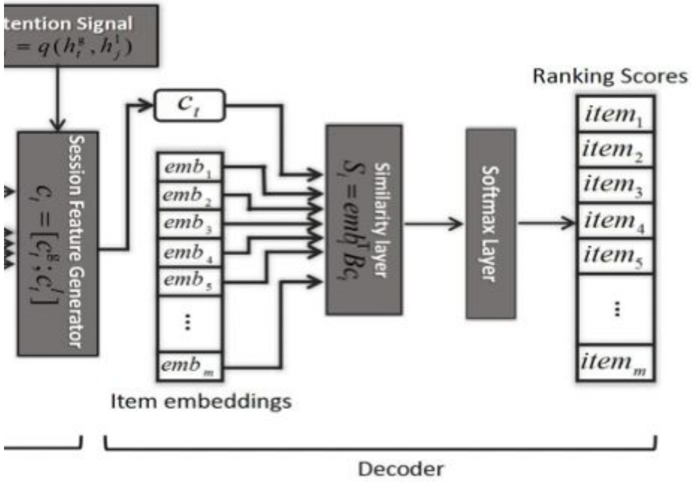
- $\mathbf{h_t}^{\mathbf{g}}$ encodes entire behavior
- $\mathbf{h_t}^{\mathbf{l}}$ is used to compute alpha with previous hidden states



Methods

Decoding

- Usually RNNs use a fully connected layer
  - The number of parameters to be learned in this layer is |*H*| * |*N*|
    - |*H*| is the dimension of session representation
    - |*N*| is the number of candidate items for prediction
- Bi-linear decoding scheme
  - Reduces the number of parameters
  - Improves the performance of NARM



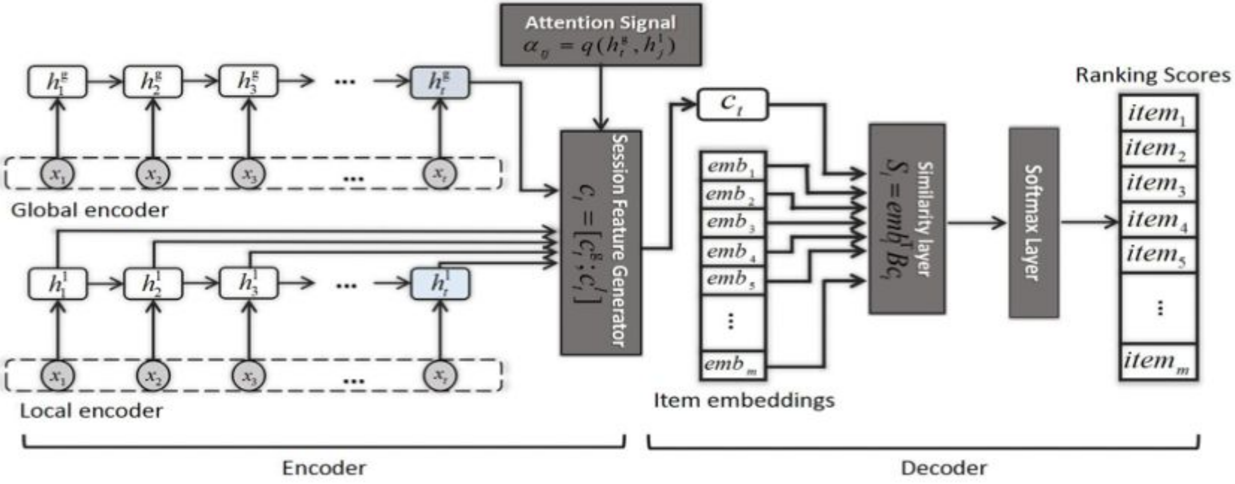
Bi-linear Decoding Scheme

- A bi-linear similarity function between the current session and each candidate items is used to compute a similarity score S_i
  - $S_i = emb_i^T \boldsymbol{Bc_t}$
    - **B** is a |*D*| * |*H*| matrix
      - |*D*| is the dimension of each item embedding
- **Total number of parameters learned is now |*D*| * |*H*|**

Methods

Training

- To fit the attention mechanism in local encoder, NARM processes each input sequence separately
    - Not session parallel, sequence-to sequence
- Standard mini-batch gradient descent with cross entropy loss
- Back-propagation Through Time (BPTT) method is used to train



Experimental Setup

Two standard transaction dataset: **YOOCHOOSE**, **DIGINETICA**

Data preprocessing:

- Filter out sessions of length 1 and items that appear less than 5 times
- Filter out the clicks from test set where the clicked items did not appear in the training set
- Generate the sequences with corresponding labels(targets)

[x1, x2, … xn]--->

([x1],V(x2)), ([x1,x2],V(x3)),..., ([x1,x2,...,xn−1],V(xn))

V is the last click in the current session

- Take the recent fraction 1/64 and ¼ of **YOOCHOOSE** for better performance

Datasets	all the clicks	train sessions	test sessions	all the items	avg.length
YOOCHOOSE 1/64	557248	369859	55898	16766	6.16
YOOCHOOSE 1/4	8326407	5917746	55898	29618	5.71
DIGINETICA	982961	719470	60858	43097	5.12

Evaluation Metrics

(Look at the top 20 items from the recommender)

1. Recall@20

$$= \frac{\text{\# of cases that the final click falls into the top 20 recommendation}}{\text{Total \# of cases}}$$

2. MRR(Mean reciprocal rank)@20

$$\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}.$$

Rank is the ranking from our recommender

1/rank set to 0 if it is larger than 20

Performance

Table 2: The comparison of different decoders in NARM.

Decoders	YOOCHOOSE 1/64		YOOCHOOSE 1/4		DIGINETICA	
	Recall@20(%)	MRR@20(%)	Recall@20(%)	MRR@20(%)	Recall@20(%)	MRR@20(%)
Fully-connected decoder	67.67	29.17	69.49	29.54	57.84	24.77
Bi-linear similarity decoder	**68.32**	28.76	**69.73**	29.23	**62.58**	**27.35**

Table 3: Performance comparison of NARM with baseline methods over three datasets.

Methods	YOOCHOOSE 1/64		YOOCHOOSE 1/4		DIGINETICA	
	Recall@20(%)	MRR@20(%)	Recall@20(%)	MRR@20(%)	Recall@20(%)	MRR@20(%)
POP	6.71	1.65	1.33	0.30	0.91	0.23
S-POP	30.44	18.35	27.08	17.75	21.07	14.69
Item-KNN	51.60	21.81	52.31	21.70	28.35	9.45
BPR-MF	31.31	12.08	3.40	1.57	15.19	8.63
FPMC*	45.62	15.01	-	-	31.55	8.92
GRU-Rec	60.64	22.89	59.53	22.60	43.82	15.46
Improved GRU-Rec	67.84	**29.00**	69.11	29.22	57.95	24.93
NARM	**68.32**	28.76	**69.73**	**29.23**	**62.58**	**27.35**

* On YOOCHOOSE 1/4, we do not have enough memory to initialize FPMC. Our available memory is 120G.

State-of-art

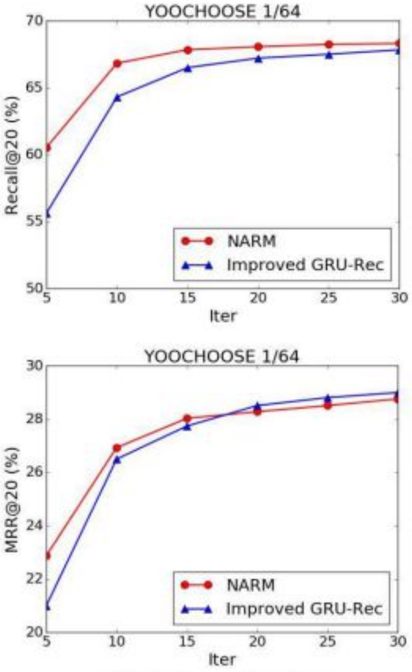
Analysis:

Session Features:

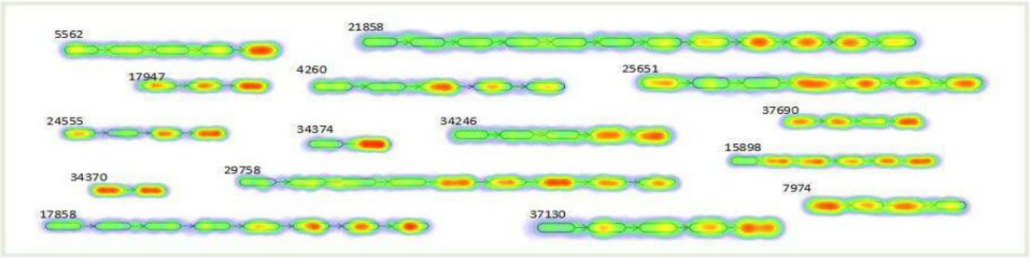
- Different session features have different levels of effectiveness.
- This suggests that the NARM architecture is suited to learning a good recommendation model.

Session Lengths:

- NARM architecture performs well on long sessions, but performance increases over baseline models are less impressive over very long and very short sequences.



(a) YOOCHOOSE1/64



(a) Performance comparison on YOOCHOOSE 1/64

	d=50		d=100	
Models	Recall@20	MRR@20	Recall@20	MRR@20
$NARM_{global}$	67.26	26.95	68.15	28.37
$NARM_{local}$	67.07	26.79	68.10	28.38
$NARM_{hybrid}$	**68.28**	**28.10**	**68.32**	**28.76**

Conclusion:

- NARM is able to exploit properties of user click sessions that are difficult for standard RNN models to capture.
- The architecture allows sequential behavior and main purpose to be captured.

- Performance can be increased by considering item attributes.
- Is there a possible way to capture any more properties of the session?
- How does NARM architecture stand up on long sequences compared to state of the art models?

The End, Thank you !