Practice on Long Sequential User Behavior Modeling for Click-Through Rate Prediction (MIMN)

Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, Kun Gai* Alibaba Group

KDD 2019

Reporter: Yinghuan Zhang Aug. 26th 2020

Click-Through Rate Prediction

在计算广告和推荐系统中，点击率（Click-Through Rate，以下简称CTR）预估是一个重要问题

- **传统CTR预估任务方法**：协同过滤，Matrix Decomposition, Factorization Machines
- **深度学习CTR预估模型**

深度学习CTR模型

Deep Crossing(2016) Microsoft - 最典型和基础性的模型

FNN (2016) UCL - 用FM的隐向量完成embedding初始化

PNN (2016) SJTU - 丰富特征交叉的方式

Wide & Deep (2016) Google -

Memorization和Generalization的综合权衡

DeepFM (2017) 华为 -

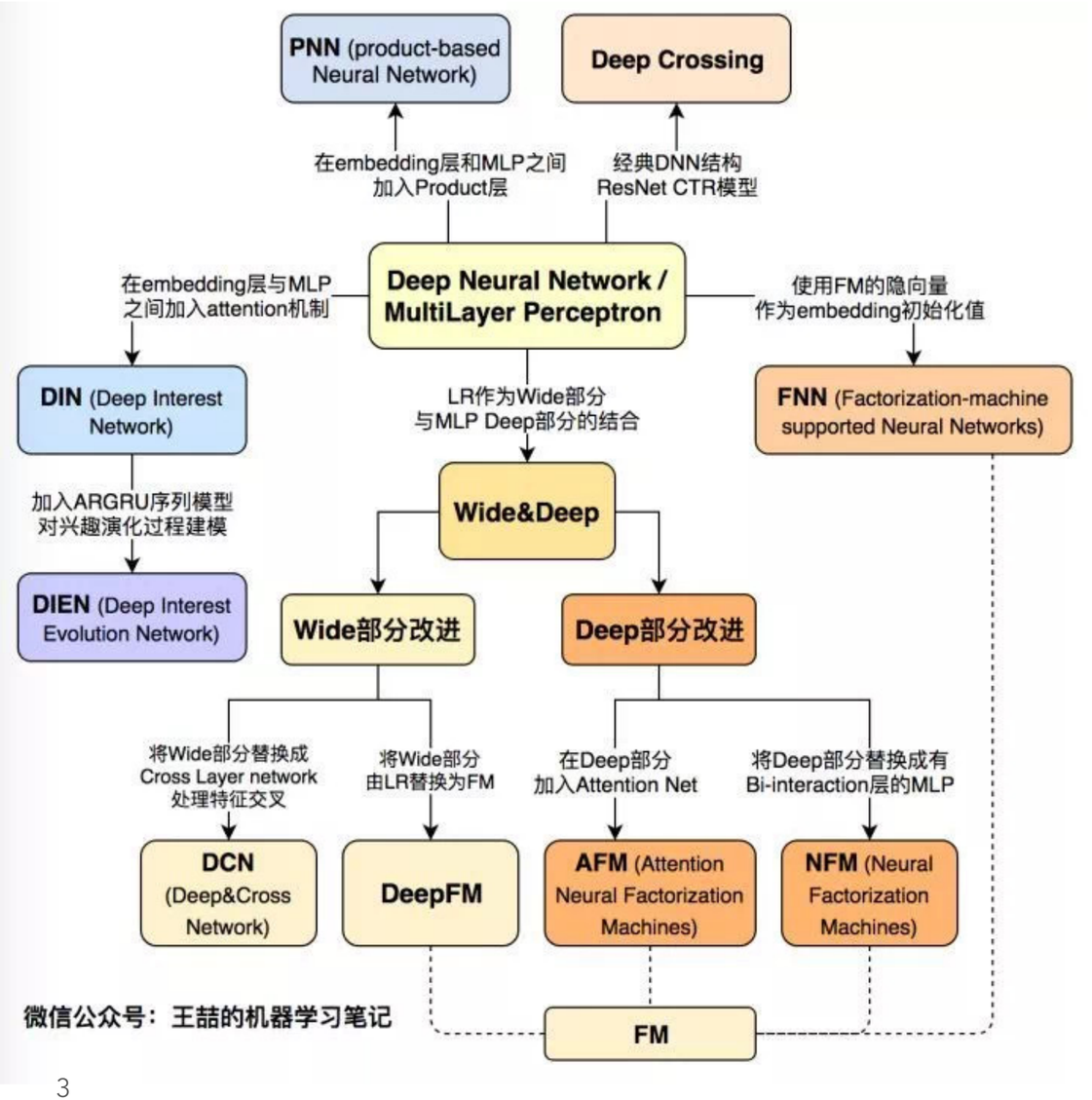
用FM代替Wide部分，加强了浅层网络部分特征组合的能力

DCN (2017) Google -

使用Cross网络代替Wide部分，增加特征之间的交互力度

NFM (2017) NUS - Deep部分改进

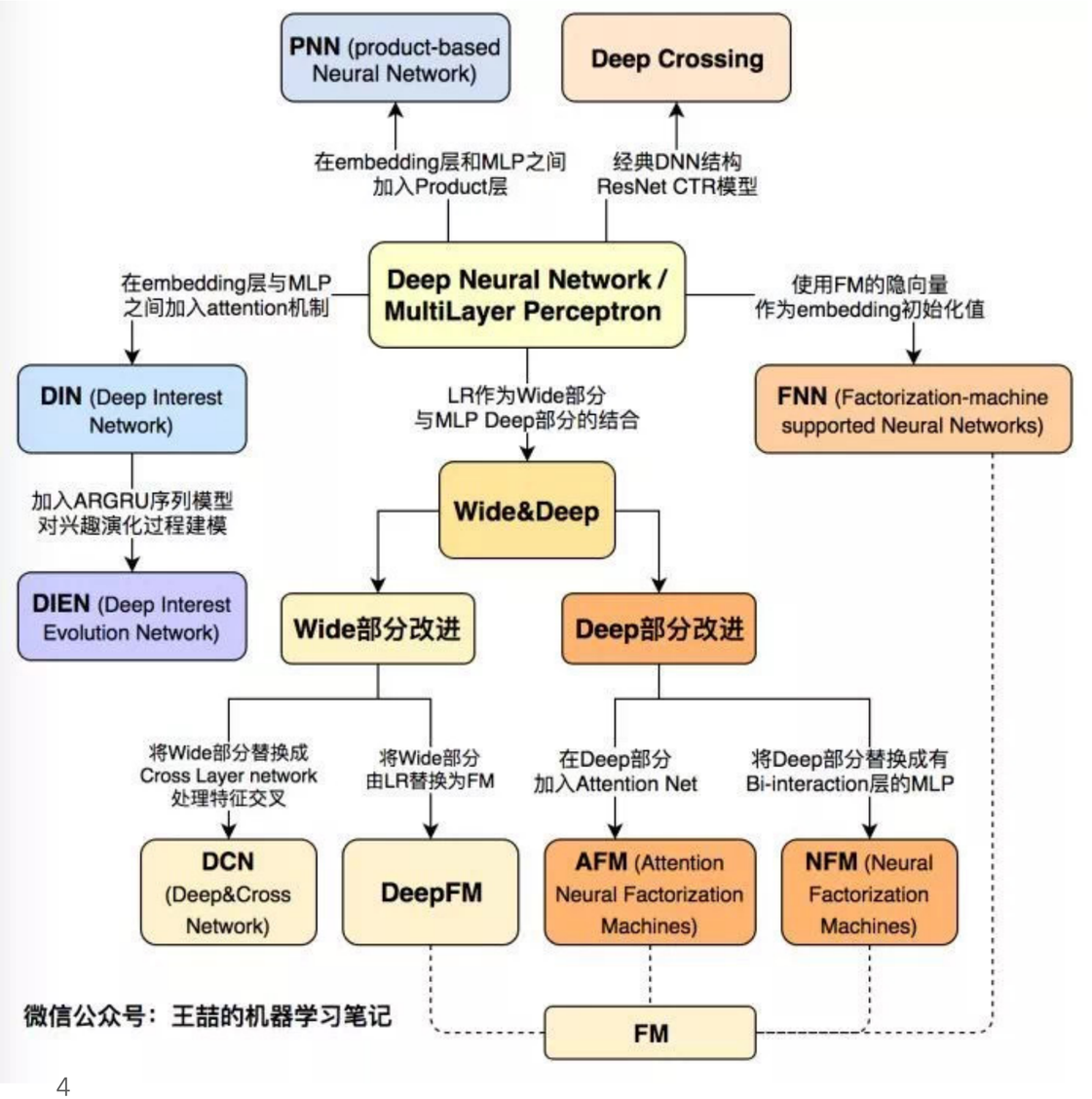
AFM (2017) ZJU - 引入attention机制的FM



深度学习CTR模型的演化图谱

深度学习CTR模型

- Embedding 层：将稀疏特征转化为低维稠密特征
- Stacking 层：将分段特征连接起来
- 多层神经网络：完成特征的组合、转换
- Scoring 层：完成CTR计算



深度学习CTR模型的演化图谱

深度学习CTR模型

Deep Crossing(2016) Microsoft - 最典型和基础性的模型

FNN (2016) UCL - 用FM的隐向量完成embedding初始化

PNN (2016) SJTU - 丰富特征交叉的方式

Wide & Deep (2016) Google -

Memorization和Generalization的综合权衡

DeepFM (2017) 华为 -

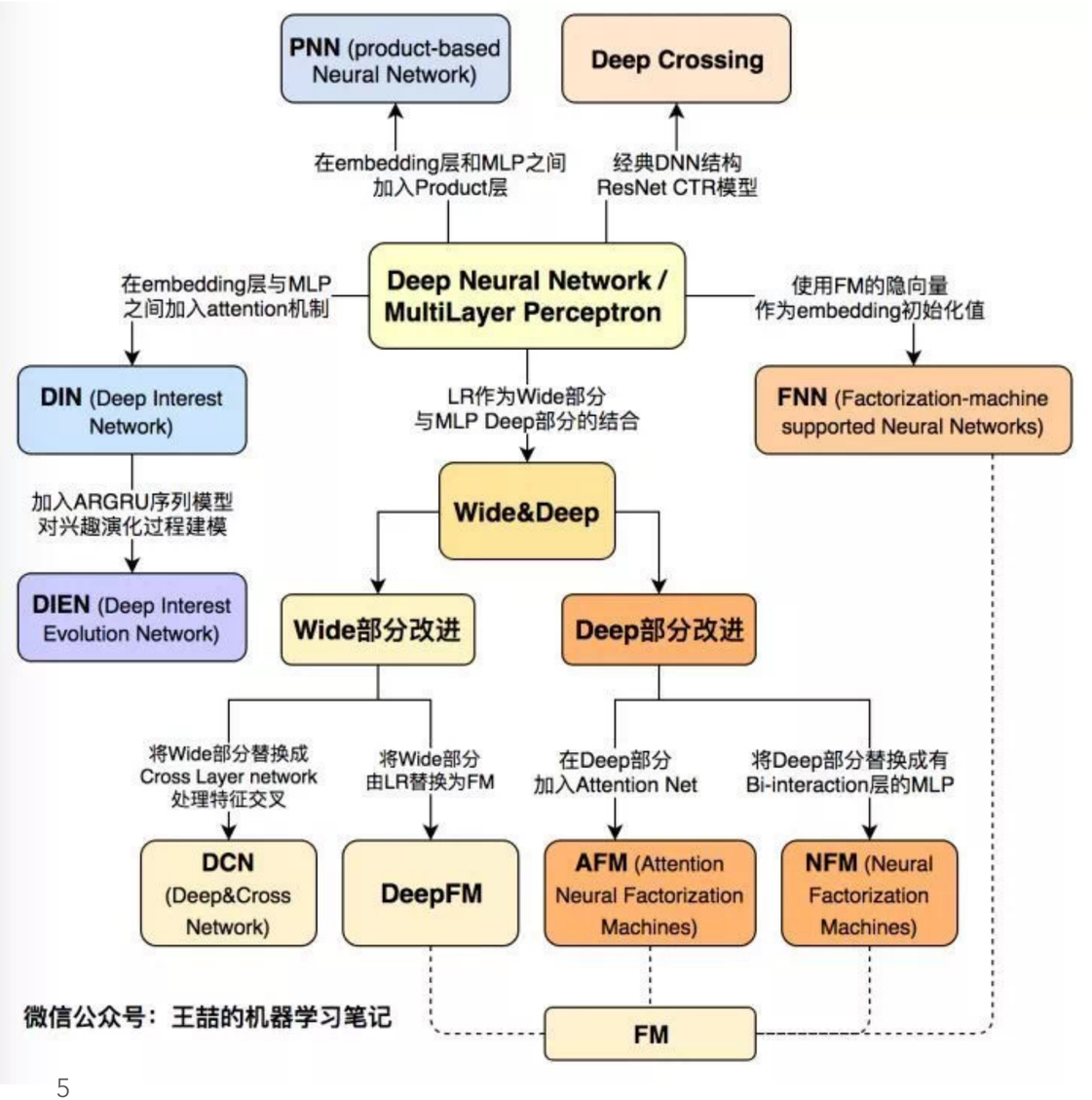
用FM代替Wide部分，加强了浅层网络部分特征组合的能力

DCN (2017) Google -

使用Cross网络代替Wide部分，增加特征之间的交互力度

NFM (2017) NUS - Deep部分改进

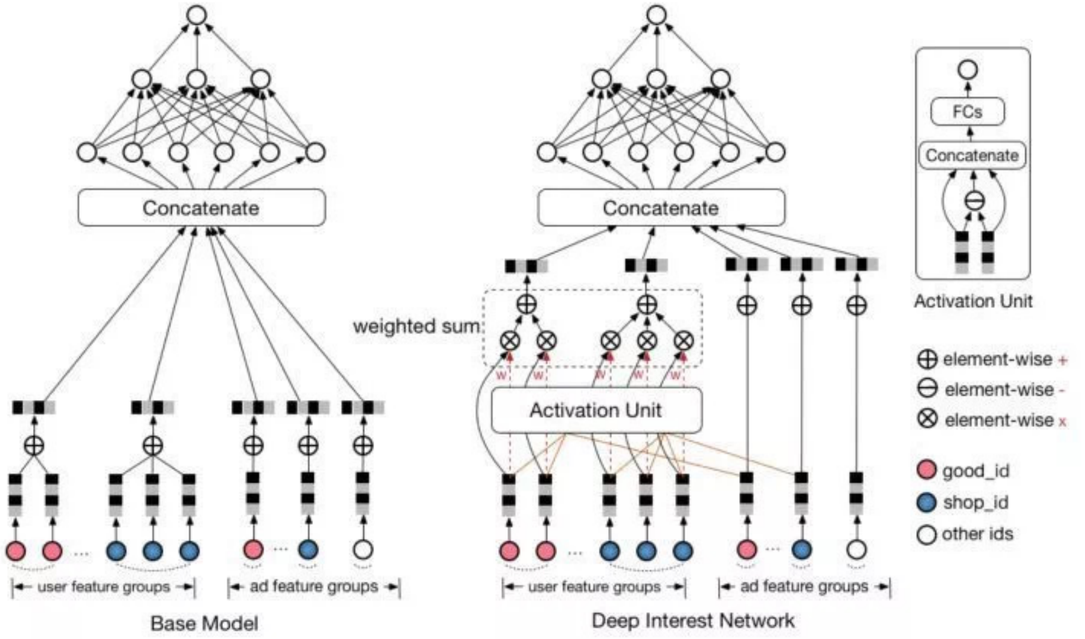
AFM (2017) ZJU - 引入attention机制的FM

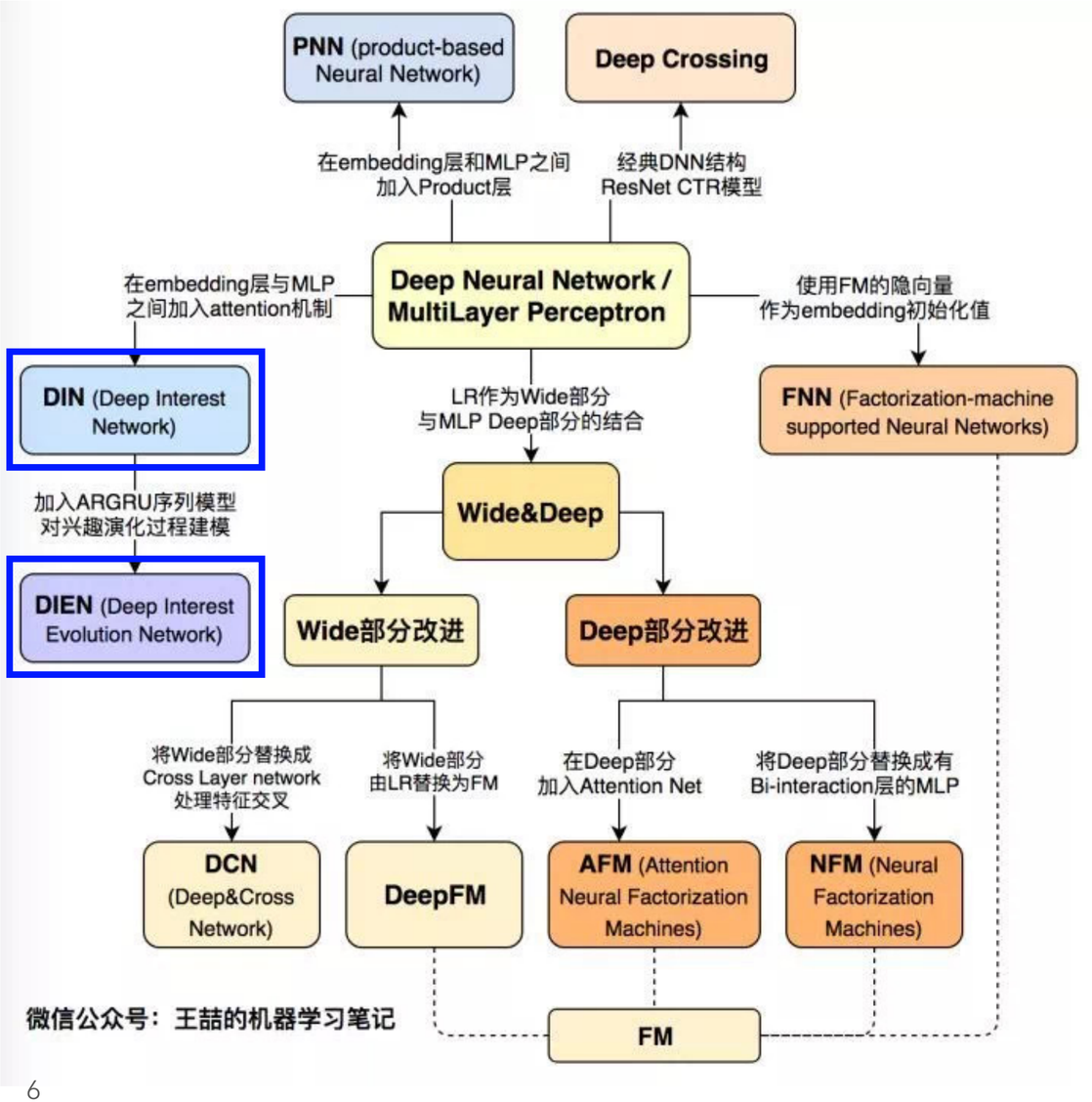


深度学习CTR模型的演化图谱

深度学习CTR模型

DIN (2018) Alibaba





深度学习CTR模型的演化图谱

深度学习CTR模型

DIEN (2019) Alibaba

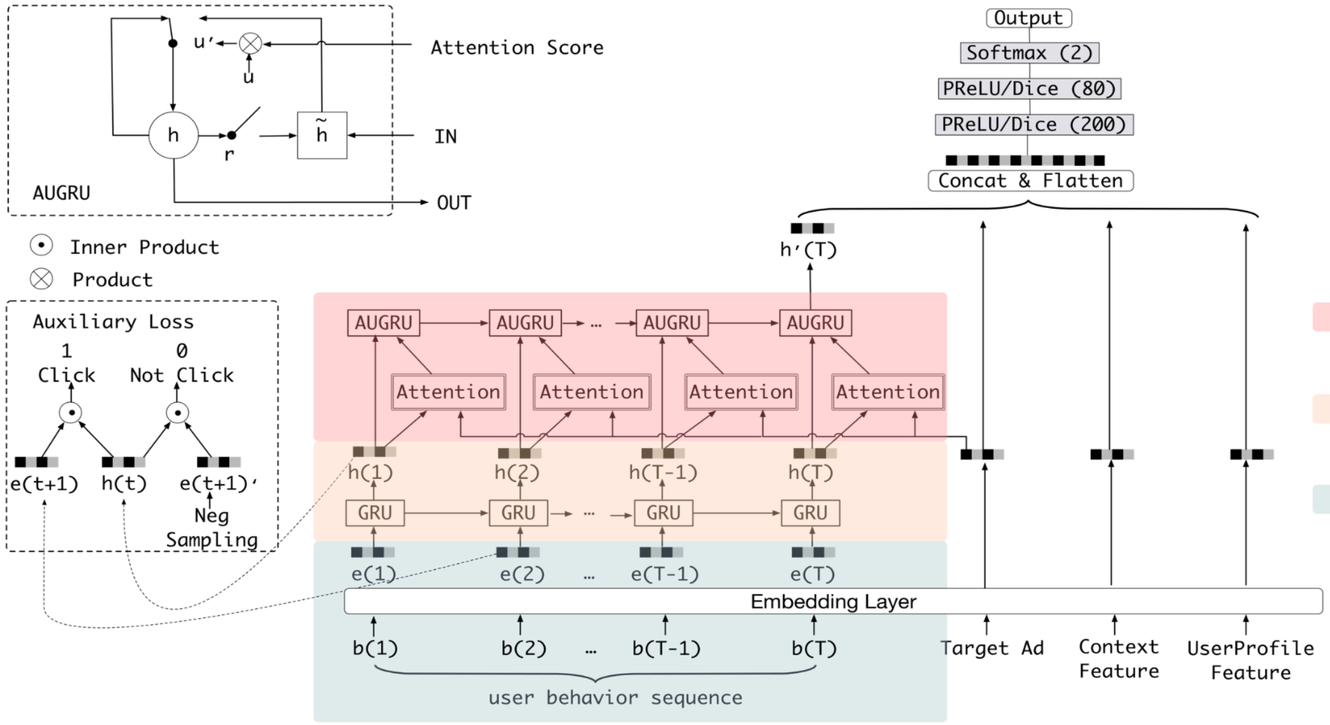
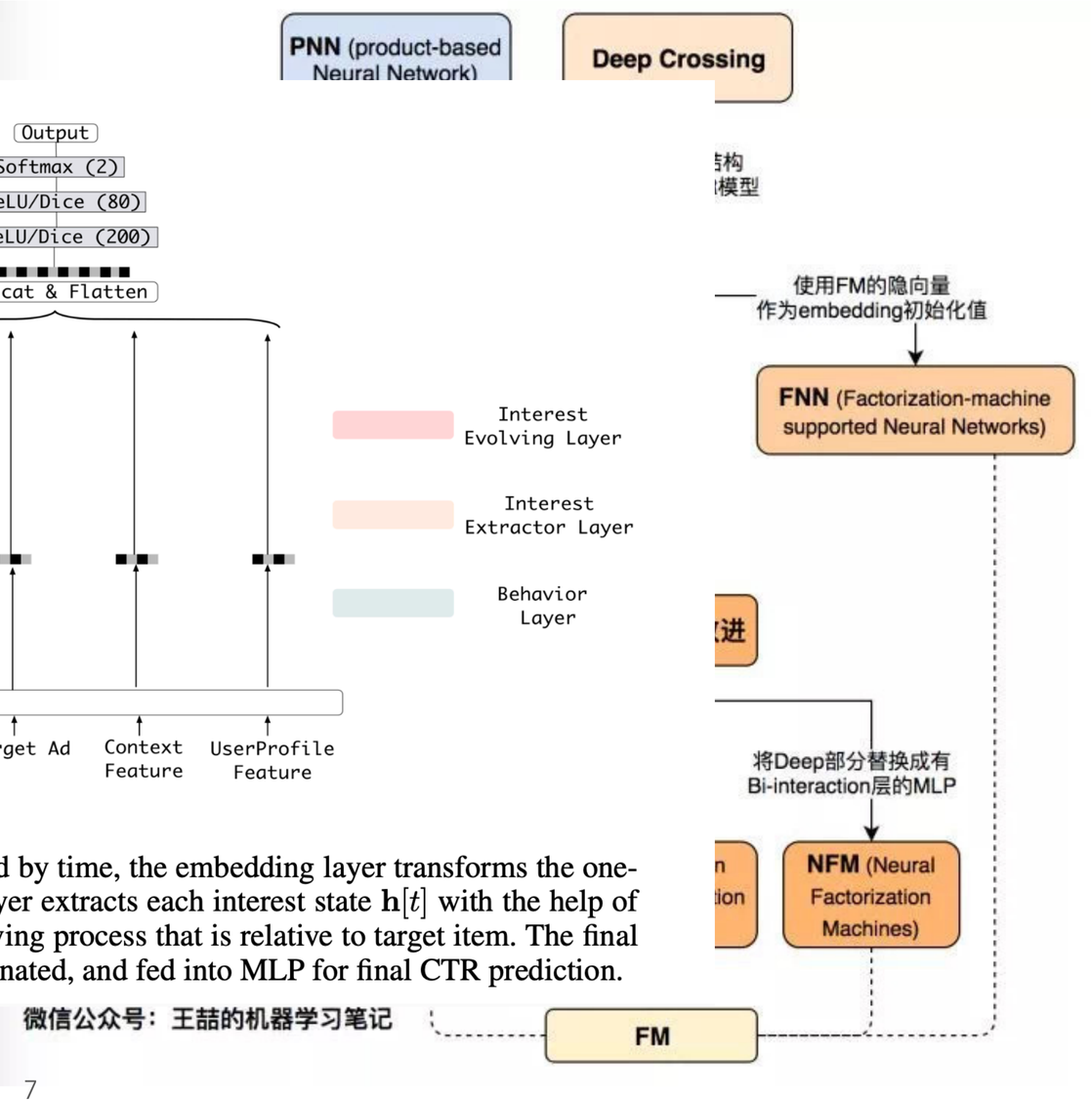


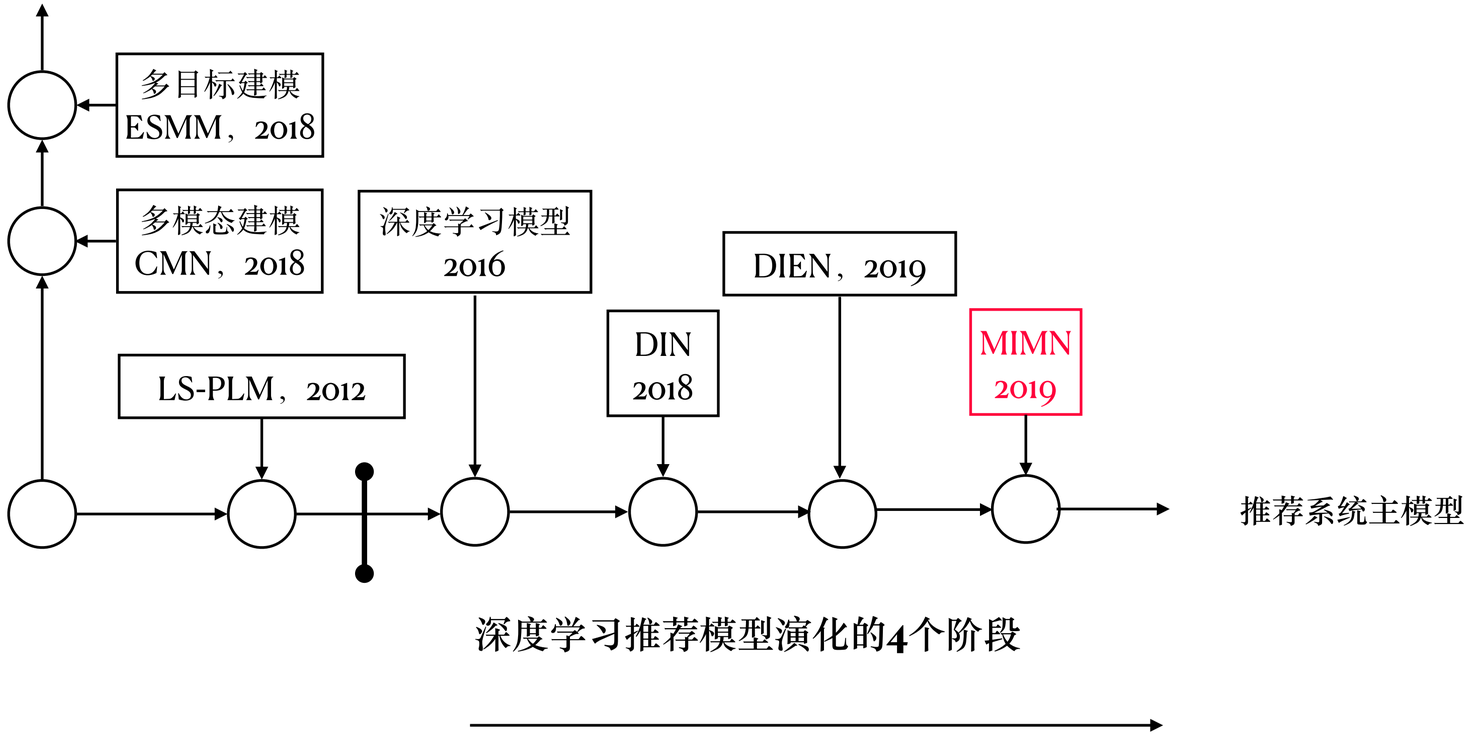
Figure 1: The structure of DIEN. At the behavior layer, behaviors are sorted by time, the embedding layer transforms the one-hot representation $\mathbf{b}[t]$ to embedding vector $\mathbf{e}[t]$. Then interest extractor layer extracts each interest state $\mathbf{h}[t]$ with the help of auxiliary loss. At interest evolving layer, AUGRU models the interest evolving process that is relative to target item. The final interest state $\mathbf{h}'[T]$ and embedding vectors of remaining feature are concatenated, and fed into MLP for final CTR prediction.



深度学习CTR模型的演化图谱

阿里巴巴推荐模型体系

基于独立问题的通用模型



用户行为序列长度对AUC的影响

两大挑战：

- Storage Constraints
- Latency Constraints

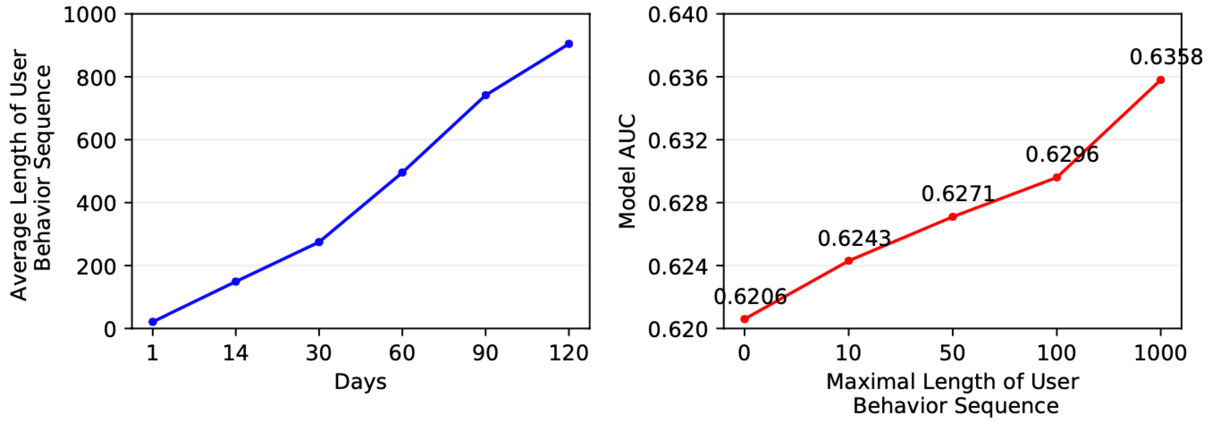


Figure 1: Statistics of sequential user behavior data and corresponding model performance in the display advertising system in Alibaba.

Multi-channel user Interest Memory Network
多通道兴趣记忆网络

(a)基础深度学习模型

(b)DIN 模型

(c)DIEN 模型

(d)MIMN 模型

将用户兴趣细分为不同兴趣通道
进一步模拟用户在不同兴趣通道上的演化过程
生成不同兴趣通道的记忆向量

MIMN

Co-design of machine learning algorithm and online serving system for CTR prediction task.

UIC (User Interest Center)

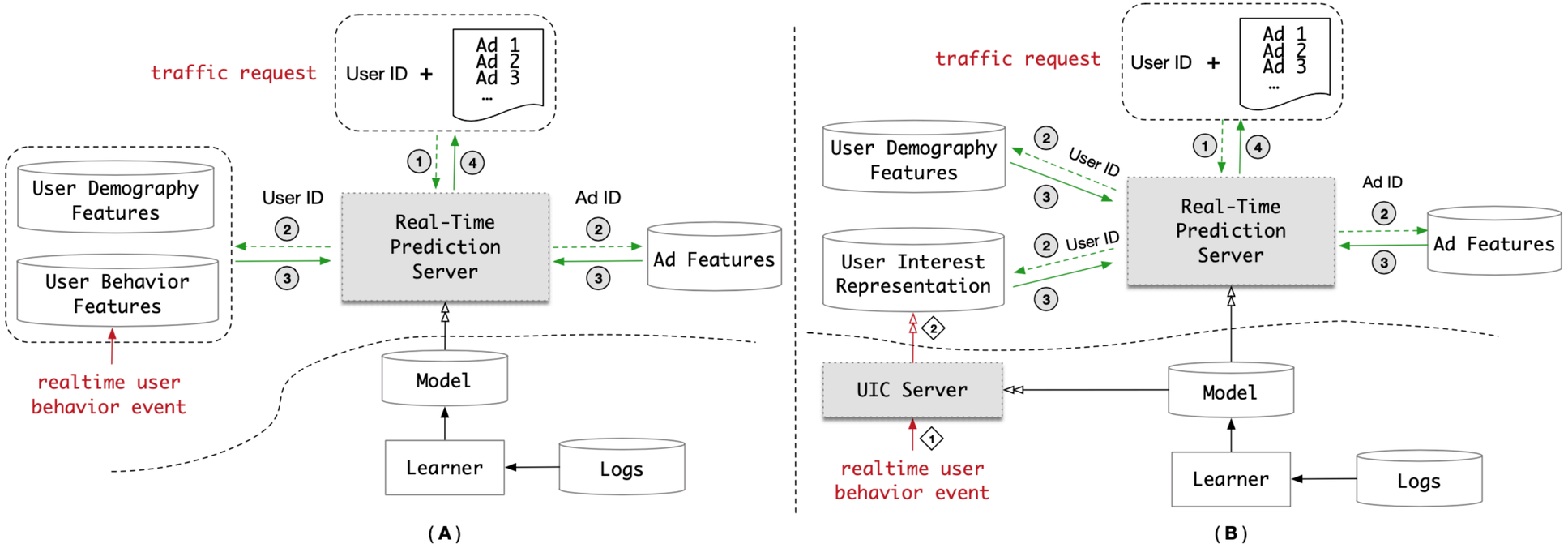
☆ 实时推断

MIMN (Multi-channel user Interest Memory Network)

☆ 长序列行为数据建模

User Interest Center (UIC)

在线环境

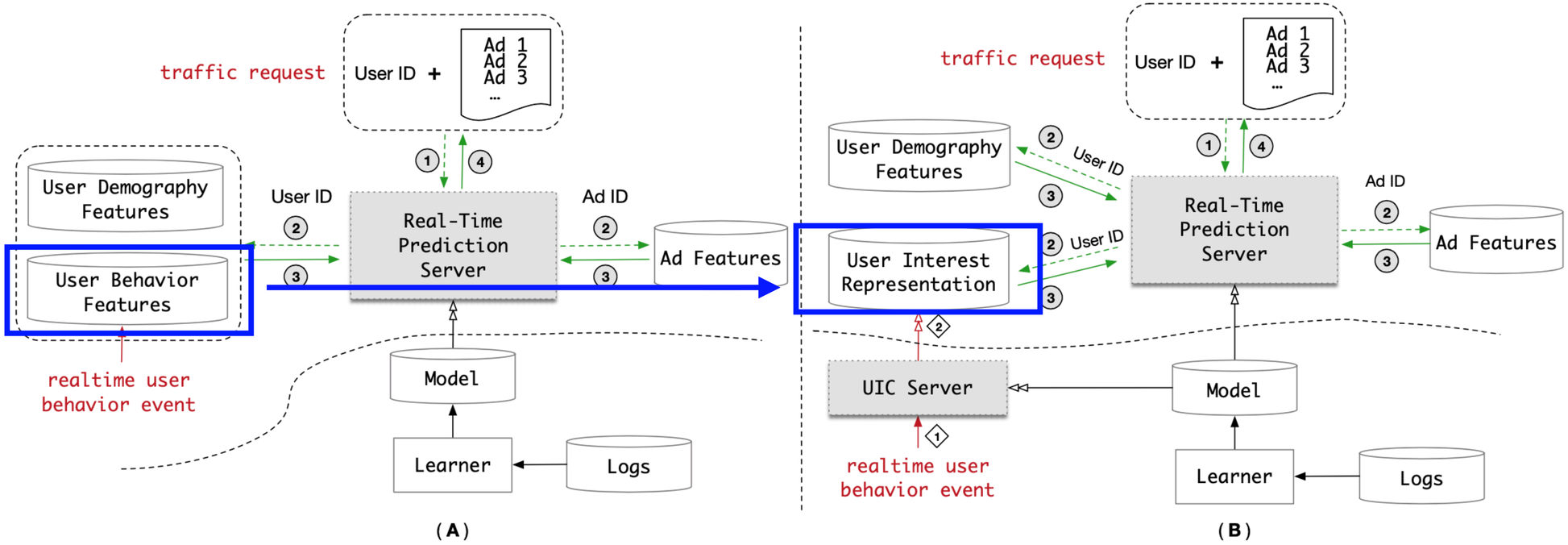


离线环境

Illustration of Real-Time Prediction (RTP) system for CTR task.

User Interest Center (UIC)

在线环境



离线环境

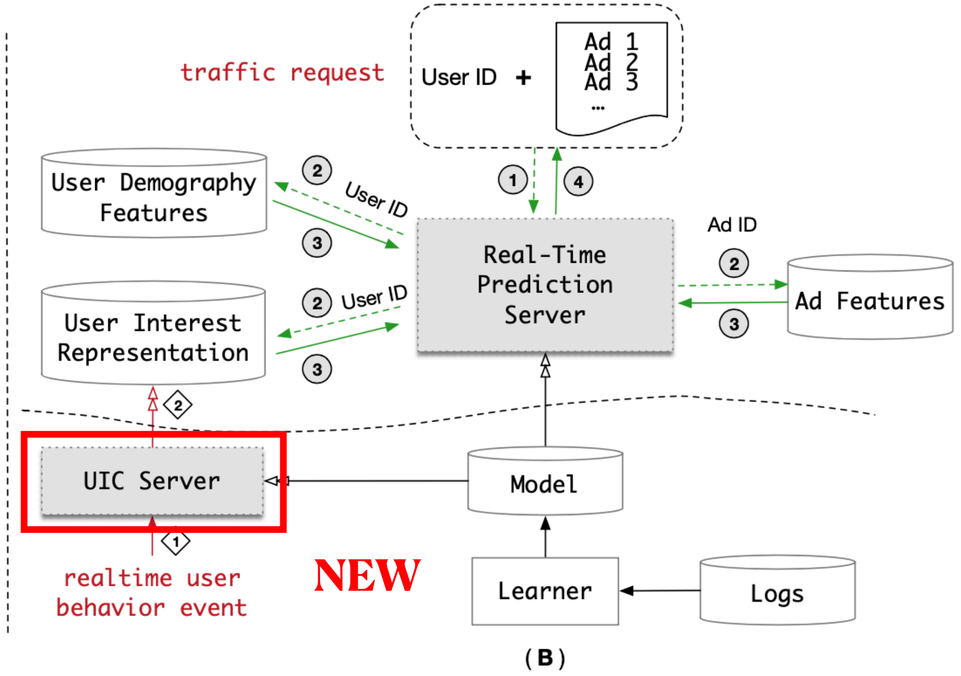
Illustration of Real-Time Prediction (RTP) system for CTR task.

User Interest Center (UIC)

★ 实时推断

☆ 长序列行为数据建模

1. UIC server maintains the **latest interest representation** for each user.
2. key point of UIC : The update of user-wise state, depends only on realtime user behavior trigger event, rather than the request. That is, UIC is **latency free** for realtime CTR prediction.



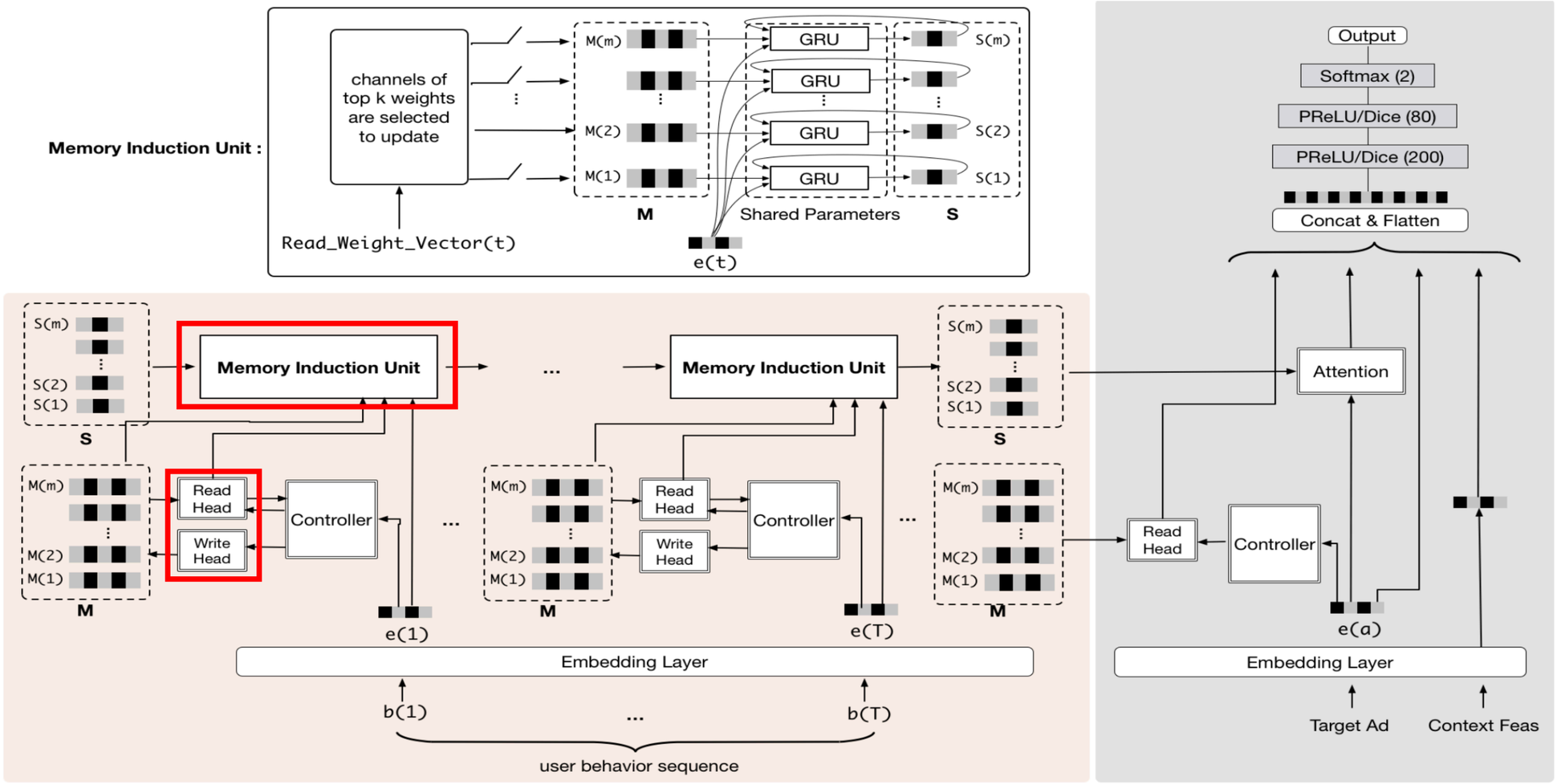
UIC can reduce the latency of DIEN model with 1000 length of user behavior from 200 ms to 19ms with 500 QPS.

Illustration of Real-Time Prediction (RTP) system for CTR task.

Multi-channel user Interest Memory Network (MIMN)

核心一：**Neural Turing machine 的 memory read/write 操作**

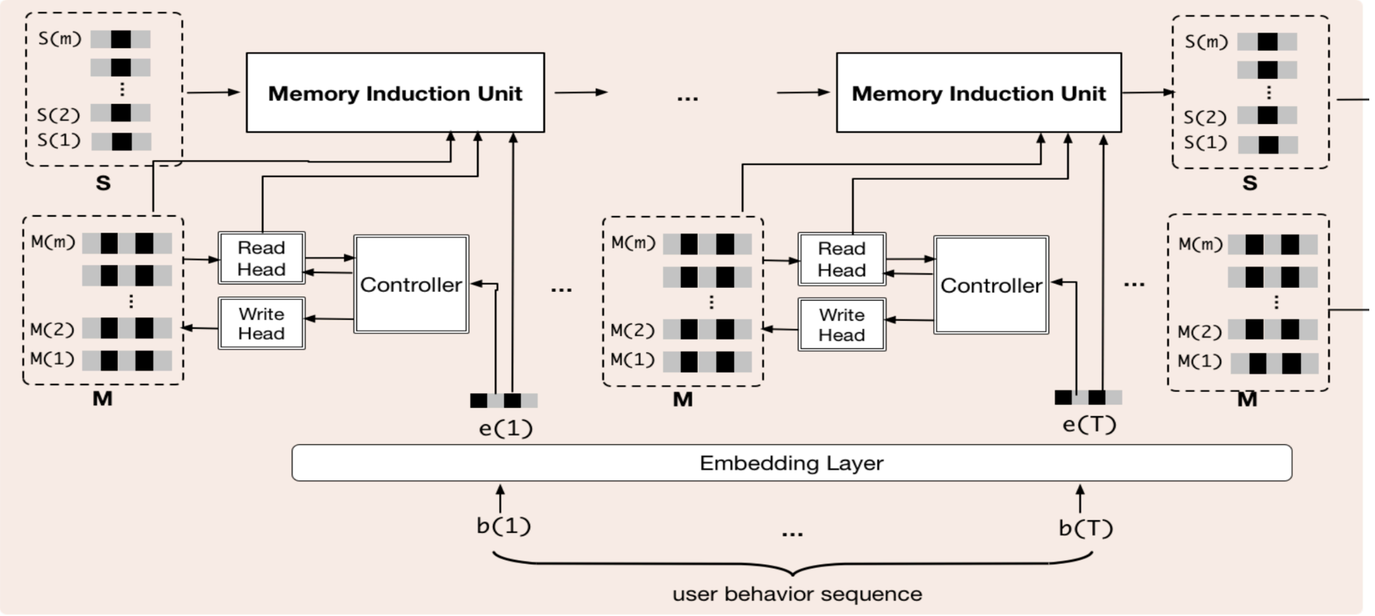
核心二：**提取高阶信息采用多通道GRU的Memory Induction Unit**



Network architecture of the proposed MIMN model

MIMN: Neural Turing Machine

以time step t 为例，记忆网络参数可以表示为矩阵M_t，包含有m个memory slot，第i个slot记忆向量记为$M_t(i)$，NTM通过controller对memory进行 memory read 和 memory write 操作



MIMN: Neural Turing Machine

传统NTM问题：memory utilization unbalance

解决方案：Memory Utilization Regularization

MUR核心思想：对不同slot写入权重的方差进行约束

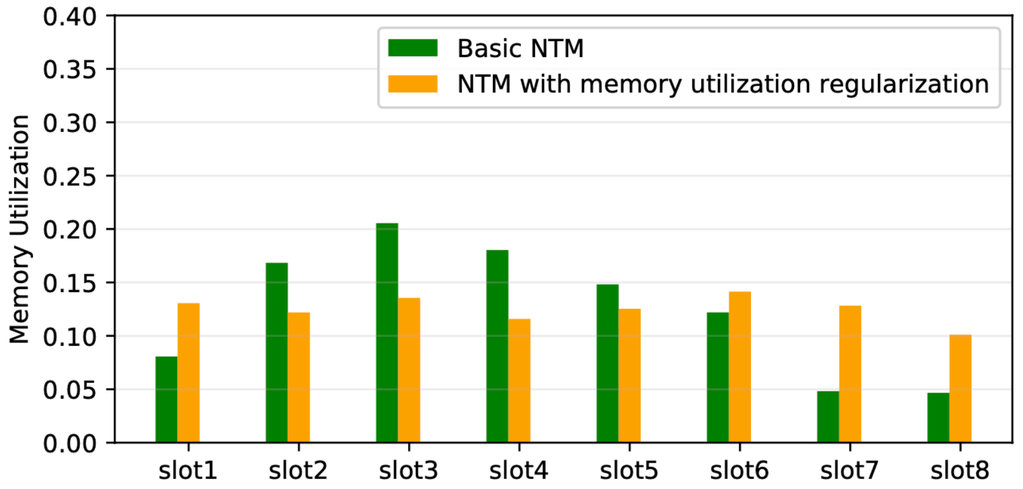
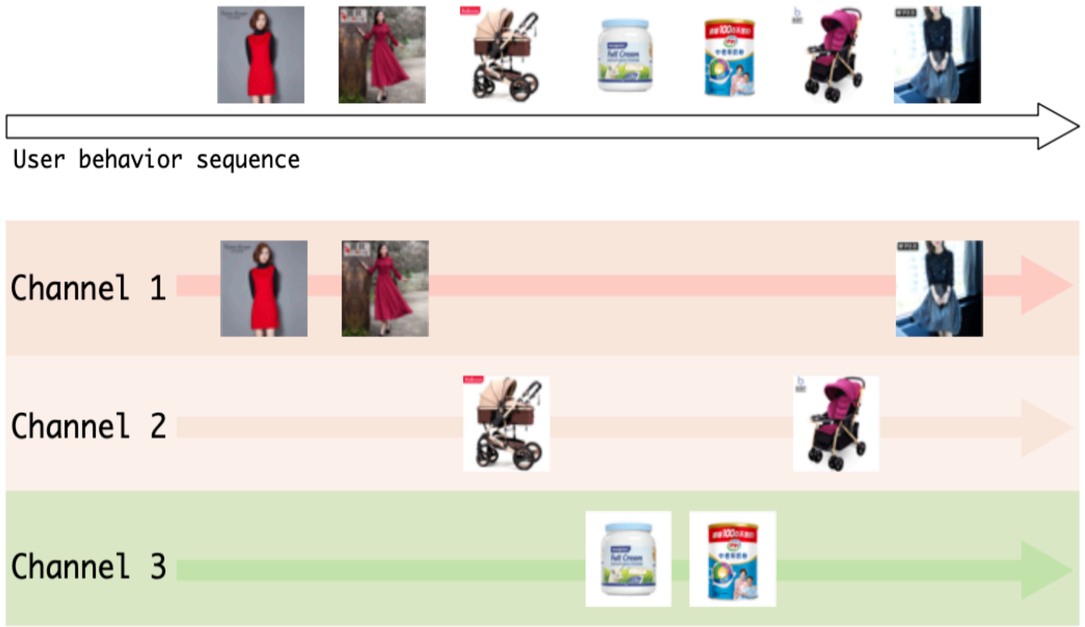
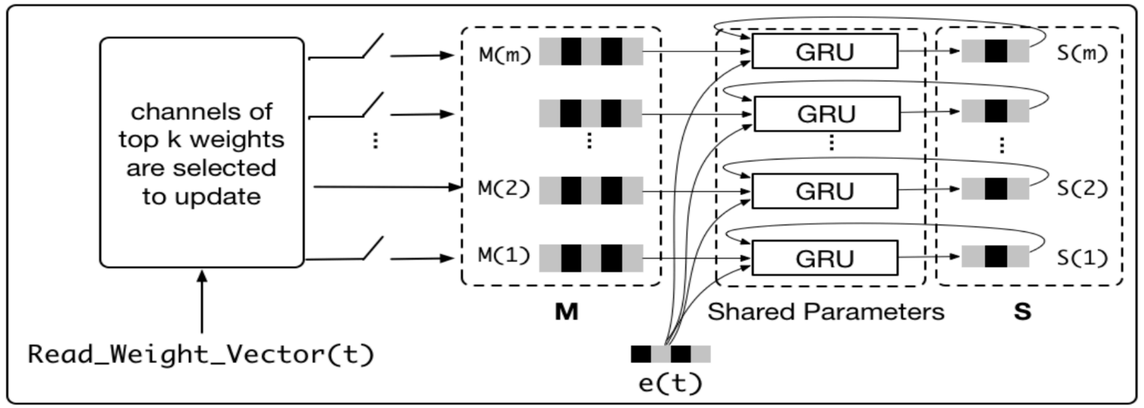


Figure 6: Memory utilization over different slots in NTM

MIMN: Memory Induction Unit



Multi-channel memory induction process



Memory Induction Unit

MIMN: Implementation for Online Serving

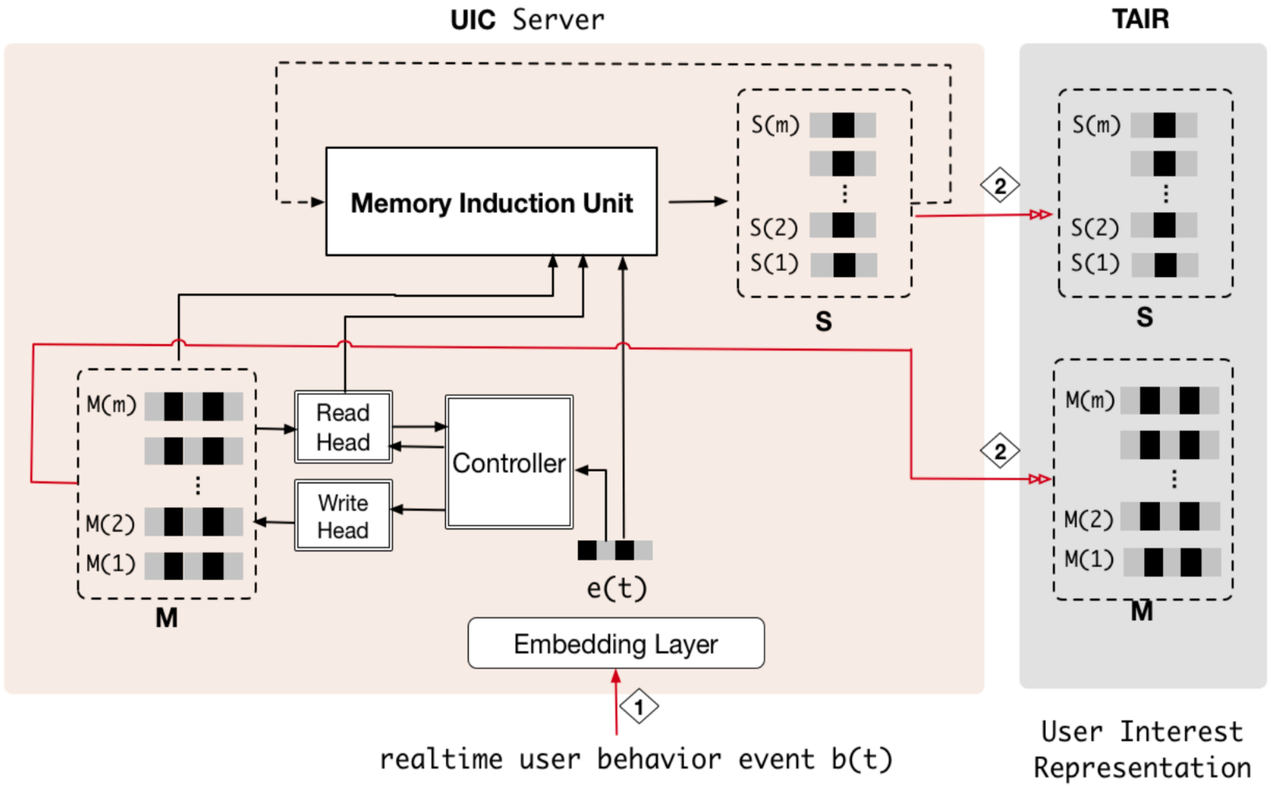
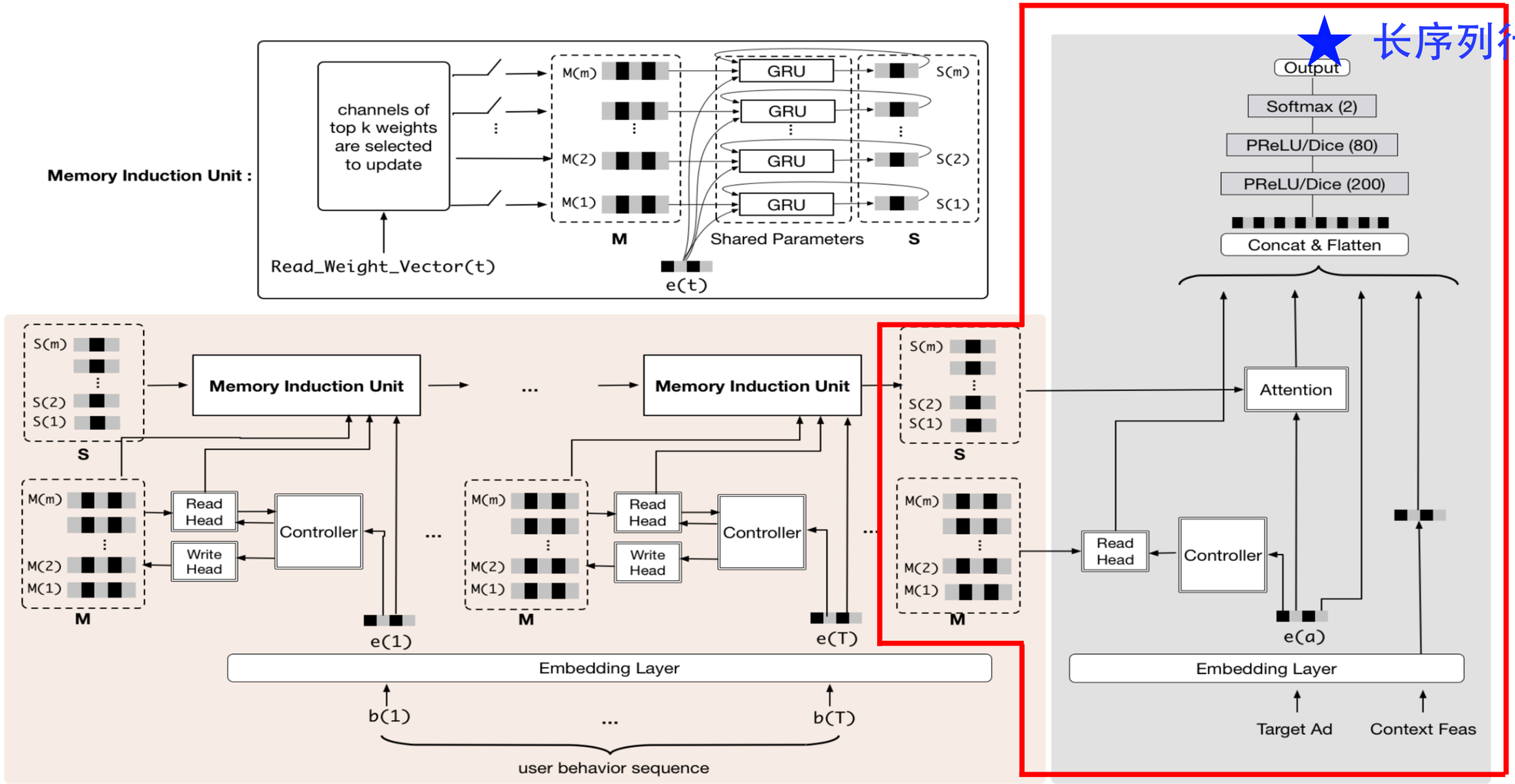


Figure 5: Implementation of the sub-network for user interest modeling with NTM and MIU in UIC Server.

MIMN: Implementation for Online Serving

★ 实时推断

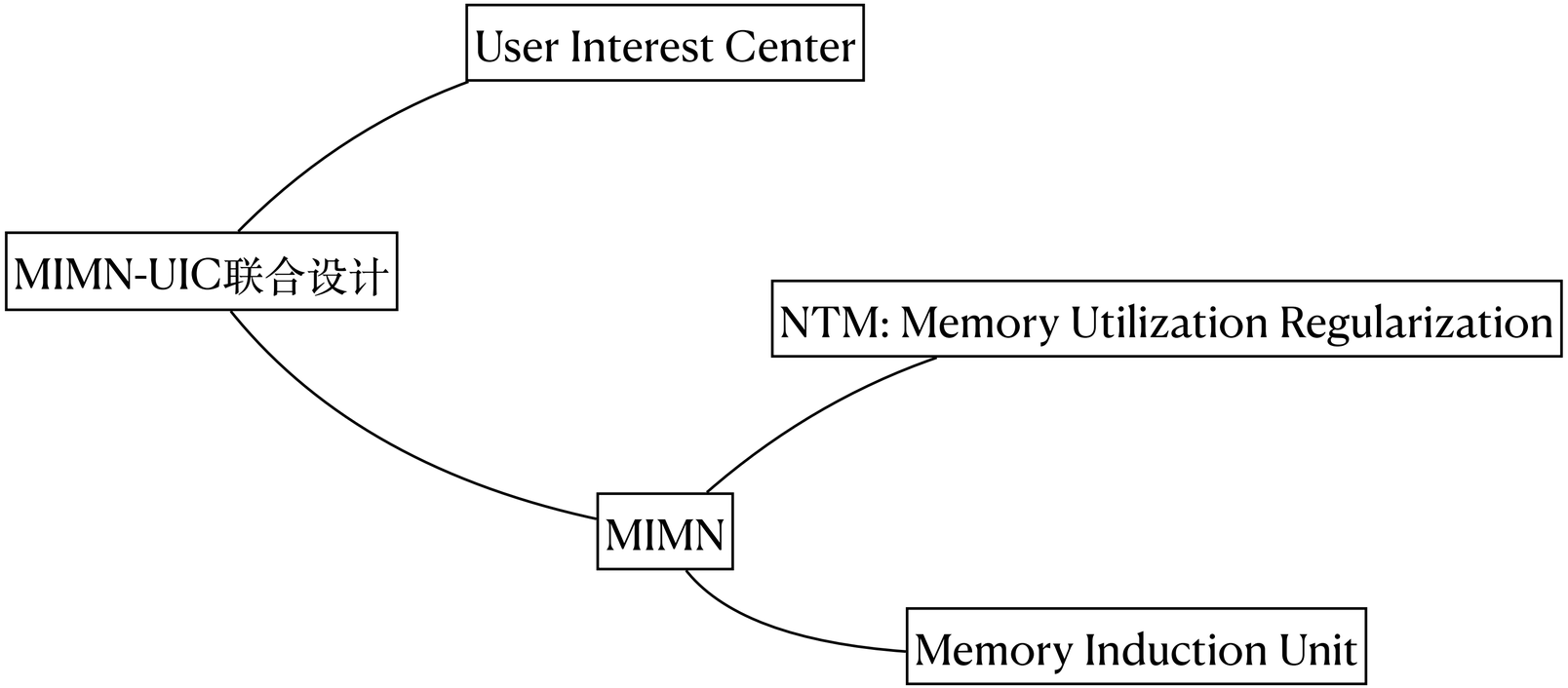
★ 长序列行为数据建模



In this way, user behavior data is not needed to be stored.

The huge volume of longtime user behaviors could be reduced from 6T to 2.7T in our system.

Conclusion


User Interest Center
MIMN-UIC联合设计
NTM: Memory Utilization Regularization
MIMN
Memory Induction Unit

Experiment

Table 1: Statistics of datasets used in this paper.

Dataset	Users	Items[a]	Categories	Instances
Amazon(Book).	75053	358367	1583	150016
Taobao.	987994	4162024	9439	987994
Industrial.	0.29 billion	0.6 billion	100,000	12.2 billion

[a] For industrial dataset, items refer to be the advertisement.

Table 2: Model performance (AUC) on public datasets

Model	Taobao (mean ± std)	Amazon (mean ± std)
Embedding&MLP	0.8709 ± 0.00184	0.7367 ± 0.00043
DIN	0.8833 ± 0.00220	0.7419 ± 0.00049
GRU4REC	0.9006 ± 0.00094	0.7411 ± 0.00185
ARNN	0.9066 ± 0.00420	0.7420 ± 0.00029
RUM	0.9018 ± 0.00253	0.7428 ± 0.00041
DIEN	0.9081 ± 0.00221	0.7481 ± 0.00102
MIMN	**0.9179 ± 0.00325**	**0.7593 ± 0.00150**

Experiment

Table 5: Model performance (AUC) on industrial dataset

Model	AUC
DIEN	0.6541
MIMN	0.6644
MIMN under out-synch setting (within one day)	0.6644
MIMN trained with big-sale data	0.6627

Online A/B Testing. We have deployed the proposed solution in the display advertising system in Alibaba. From 2019-03-30 to 2019-05-10, we conduct strict online A/B testing experiment to validate the proposed MIMN model. Compared to DIEN (our last product model), MIMN achieves 7.5% CTR and 6% RPM (Revenue Per Mille) gain. We attribute this to the mining of additional information from long sequential behavior data that the proposed co-design solution enables.

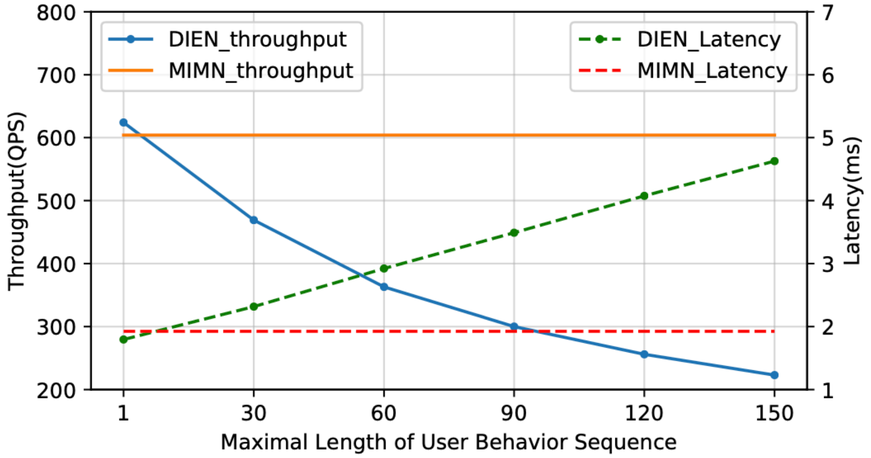


Figure 7: System performance of realtime CTR prediction system w.r.t. different length of user behavior sequences, serving with MIMN and DIEN model. MIMN model is implemented with the design of UIC server.

THANK YOU!

Reporter: Yinghuan Zhang Aug. 26th 2020

MIMN: Neural Turing Machine

Memory Read. Input with t-th behavior embedding vector, the controller generates a read key $\mathbf{k}_t$ to address memory. It first traverses all memory slots, generating a weight vector $\mathbf{w}_t^r$

$$\mathbf{w}_t^r(i) = \frac{\exp\big(K\big(\mathbf{k}_t, \mathbf{M}_t(i)\big)\big)}{\sum_j^m \exp\big(K\big(\mathbf{k}_t, \mathbf{M}_t(j)\big)\big)}, \quad for \ \ i = 1, 2, ...m \tag{1}$$

where

$$K\big(\mathbf{k}_t, \mathbf{M}_t(i)\big) = \frac{\mathbf{k}_t^T \mathbf{M}_t(i)}{\| \mathbf{k}_t \| \| \mathbf{M}_t(i) \|}, \tag{2}$$

then calculates a weighted memory summarization as output $\mathbf{r}_t$,

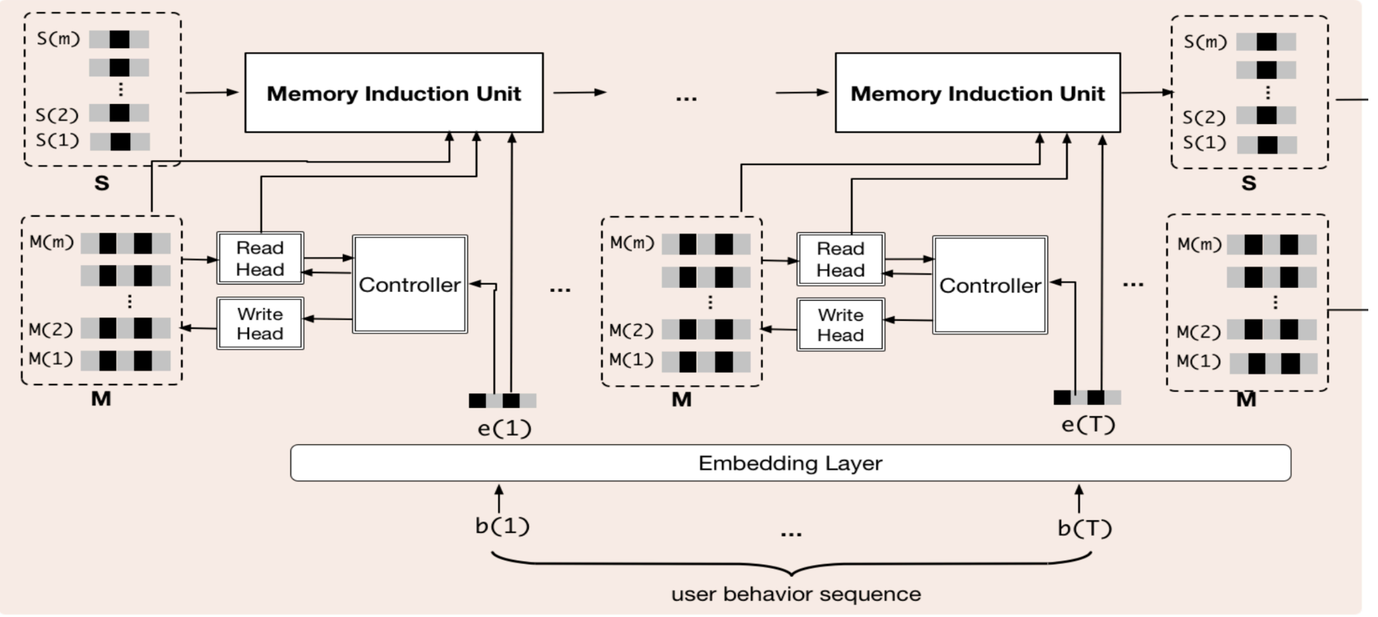
$$\mathbf{r}_t = \sum_i^m w_t^r(i)\mathbf{M}_t(i). \tag{3}$$

Memory Write. The weight vector $\mathbf{w}_t^w$ for memory write addressing is generated similar to *memory read* operation of Eq.(1). Two additional keys of add vector $\mathbf{a}_t$ and erase vector $\mathbf{e}_t$ are also generated from the controller, which control the update of memory.

$$\mathbf{M_t} = (1 - \mathbf{E_t}) \odot \mathbf{M_{t-1}} + \mathbf{A_t}\ , \tag{4}$$

where $\mathbf{E_t}$ is the erase matrix, $\mathbf{A_t}$ is the add matrix, with $\mathbf{E_t} = \mathbf{w}_t^w \otimes \mathbf{e}_t$ and $\mathbf{A_t} = \mathbf{w}_t^w \otimes \mathbf{a}_t$. Here $\odot$ and $\otimes$ means dot product and outer product respectively.

以time step t 为例，记忆网络参数可以表示为矩阵M_t，包含有m个memory slot，第i个slot记忆向量记为$M_t(i)$，NTM通过controller对memory进行 memory read 和 memory write 操作



MIMN: Neural Turing Machine

Memory Read. Input with t-th behavior embedding vector, the controller generates a read key $\mathbf{k}_t$ to address memory. It first traverses all memory slots, generating a weight vector $\mathbf{w}_t^r$

$$\mathbf{w}_t^r(i) = \frac{\exp\big(K\big(\mathbf{k}_t, \mathbf{M}_t(i)\big)\big)}{\sum_j^m \exp\big(K\big(\mathbf{k}_t, \mathbf{M}_t(j)\big)\big)}, \quad for \ \ i = 1, 2, \ldots m \tag{1}$$

where

$$K\big(\mathbf{k}_t, \mathbf{M}_t(i)\big) = \frac{\mathbf{k}_t^T \mathbf{M}_t(i)}{\| \mathbf{k}_t \| \| \mathbf{M}_t(i) \|}, \tag{2}$$

then calculates a weighted memory summarization as output $\mathbf{r}_t$,

$$\mathbf{r}_t = \sum_i^m w_t^r(i)\mathbf{M}_t(i). \tag{3}$$

Memory Write. The weight vector $\mathbf{w}_t^w$ for memory write addressing is generated similar to *memory read* operation of Eq.(1). Two additional keys of add vector $\mathbf{a}_t$ and erase vector $\mathbf{e}_t$ are also generated from the controller, which control the update of memory.

$$\mathbf{M_t} = (1 - \mathbf{E_t}) \odot \mathbf{M_{t-1}} + \mathbf{A_t}\ , \tag{4}$$

where $\mathbf{E_t}$ is the erase matrix, $\mathbf{A_t}$ is the add matrix, with $\mathbf{E_t} = \mathbf{w}_t^w \otimes \mathbf{e}_t$ and $\mathbf{A_t} = \mathbf{w}_t^w \otimes \mathbf{a}_t$. Here $\odot$ and $\otimes$ means dot product and outer product respectively.

Memory Utilization Regularization

Weight Transfer Matrix，Re-balanced 写权重

$$P_t = softmax(W_g \mathbf{g}_t)$$

$$\mathbf{w}_t^{\tilde{w}} = \mathbf{w}_t^w P_t.$$

截止到时间t的累计更新权重：

$$\mathbf{g}_t = \sum_{c=1}^{t} \mathbf{w}_c^{\tilde{w}}$$

Parameter Matrix 根据一个 regression loss 学习得到

$$\mathbf{w}^{\tilde{w}} = \sum_{t=1}^{T} \mathbf{w}_t^{\tilde{w}},$$

$$\mathbf{L}_{reg} = \lambda \sum_{i=1}^{m} \Big(\mathbf{w}^{\tilde{w}}(i) - \frac{1}{m} \sum_{i=1}^{m} \mathbf{w}^{\tilde{w}}(i)\Big)^2,$$

首先，$g_{t-1} = \sum_{c=1}^{t-1} w_c^{\tilde{w}}$ （同样，我觉得这里用g_{t-1}更为合适，论文中是g_t），$w_c^{\tilde{w}}$代表c时刻的调整后权重，调整权重的计算方式如下：

？

$$P_t = softmax(W_g g_{t-1})$$
$$w_t^{\tilde{w}} = w_t^w P_t$$

MIMN: Neural Turing Machine

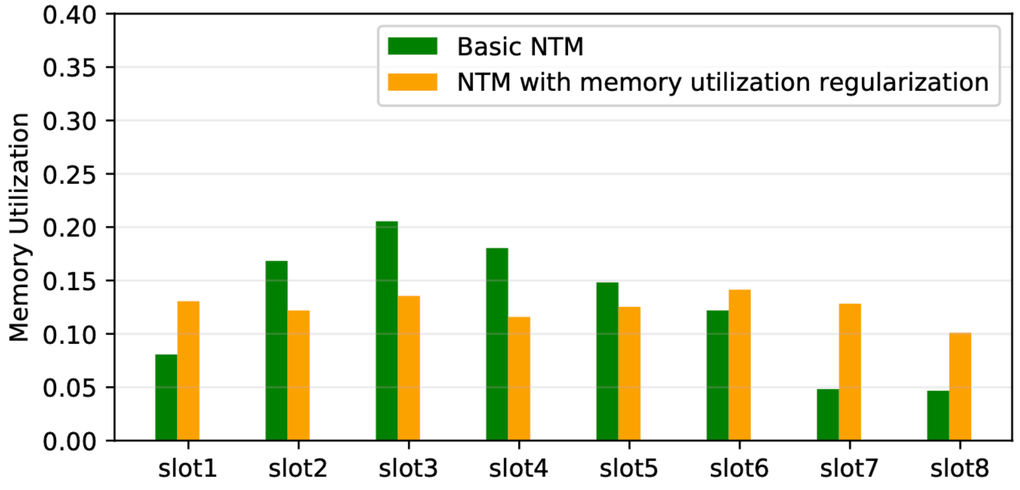


Figure 6: Memory utilization over different slots in NTM

Memory Utilization Regularization

Weight Transfer Matrix，Re-balanced 写权重

$$P_t = softmax(W_g \mathbf{g}_t)$$

$$\mathbf{w}_t^{\tilde{w}} = \mathbf{w}_t^{w} P_t .$$

截止到时间t的累计更新权重：

$$\mathbf{g}_t = \sum_{c=1}^{t} \mathbf{w}_c^{\tilde{w}}$$

Parameter Matrix 根据一个 regression loss 学习得到

$$\mathbf{w}^{\tilde{w}} = \sum_{t=1}^{T} \mathbf{w}_t^{\tilde{w}},$$

$$\mathbf{L}_{reg} = \lambda \sum_{i=1}^{m} \left(\mathbf{w}^{\tilde{w}}(i) - \frac{1}{m} \sum_{i=1}^{m} \mathbf{w}^{\tilde{w}}(i)\right)^2,$$

首先，$g_{t-1} = \sum_{c=1}^{t-1} w_c^{\tilde{w}}$ （同样，我觉得这里用g_{t-1}更为合适，论文中是g_t），$w_c^{\tilde{w}}$代表c时刻的调整后权重，调整权重的计算方式如下：

？

$$P_t = softmax(W_g g_{t-1})$$
$$w_t^{\tilde{w}} = w_t^{w} P_t$$

MIMN: Memory Induction Unit

MIU 和NTM一样，有一个 m slot 的内部 memory $\mathbf{S}$

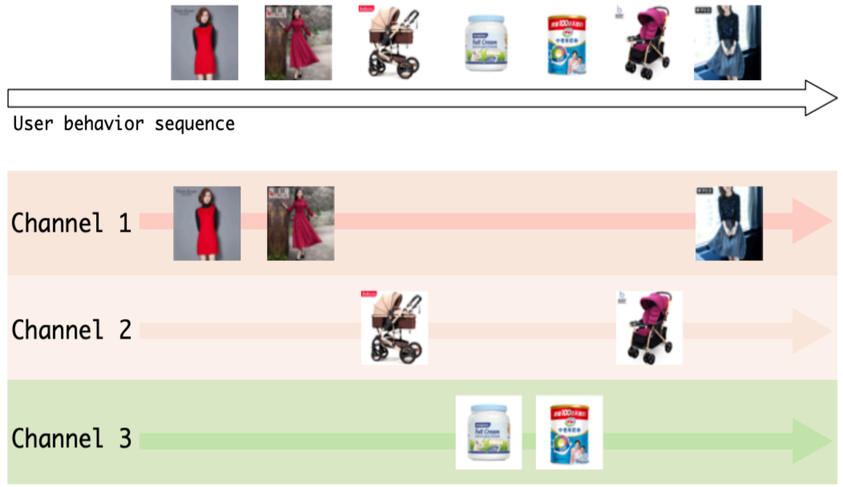
将每个 memory slot 看作一个用户兴趣channel，在第 t 个时间步，MIU：

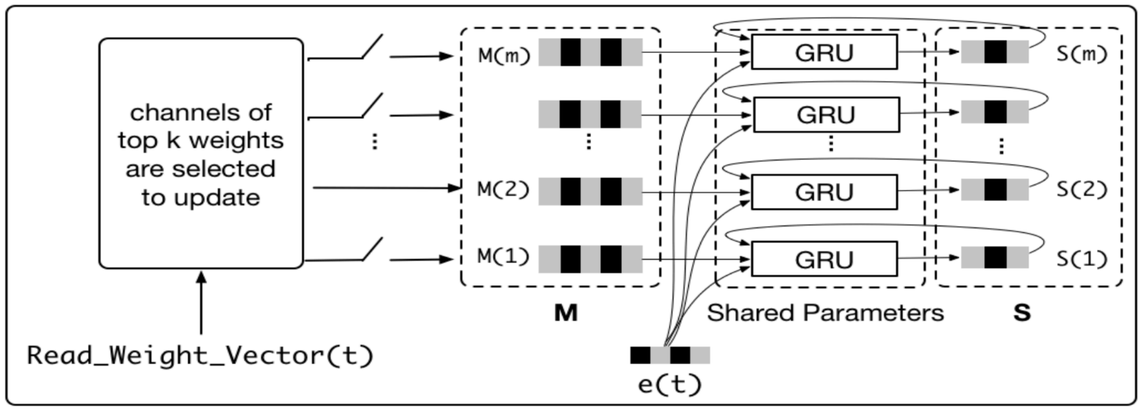
1. 选择写权重 $\mathbf{w}_t^r(i)$ 最高的 k 个 channels
2. 对第 i 个被选择的channel，更新 $\mathbf{S}_t(i)$

$$\mathbf{S}_t(i) = \mathrm{GRU}(\mathbf{S}_{t-1}(i), \mathbf{M}_t(i), e_t),$$

其中 $\mathbf{M}_t(i)$ 是NTM第 i 个memory slot，e_t 是 behavior embedding vector

MIU 同时捕捉了原始行为信息和NTM中存储的信息，如下图所示





Memory Induction Unit