KDD2020 | 阿里团队最新的多元兴趣推荐模型ComiRec

深度传送门 8月13日

以下文章来源于潜心的Python小屋 ， 作者潜心

潜心的Python小屋
记录与分享自己的学习内容，励志做一个推荐算法的小齿轮⚙。

前言

本次分享一篇2020年阿里团队发表在KDD的文章“Controllable Multi-Interest Framework for Recommendation”。本篇文章提出的模型ComiRec是基于推荐系统中匹配/召回的方法，且是一个序列化推荐的解决方案。由于是当前最新的序列化推荐匹配模型，因此会详细的介绍其背景、方法与实验。

本文约4.5k字，预计阅读15分钟。

1. 研究背景

1、对于工业界的推荐系统来说，一般由匹配/召回（matching）阶段和排序/精排（ranking）阶段组成。**「匹配主要是检索出Top-K个候选物品」**，而**「精排则是通过更精确的分数来对候选物品进行排序」**。本篇文章，主要**「聚焦于匹配问题」**，提高匹配的性能。

2、可以将推荐系统看作是一个**「序列推荐问题」**，即想要预测用户下一个可能交互的物品。

3、目前推荐系统的**「准确性」**并不是唯一的问题，人们更有可能被推荐一些新的或**「多样化」**的东西【长尾问题】。

本篇文章主要是基于上述三个研究背景进行展开。

2. 研究现状

1、传统的推荐方法主要是用协同过滤来预测分数。后来演变成用神经网络---对建立用户与物品的表示，并比传统方法的性能更为优越。但对于大规模的电子商务系统，很难直接使用神经网络给出用户对目标物品的CTR预估。现如今工业界使用快速KNN（如Faiss[1]）来生成

一个候选物品集，然后使用神经网络模型（如xDeepFM[2]）来整合用户物品属性，以优化诸如CTR之类的业务指标。

2、当然最近也有使用**「图嵌入」**（Graph Embedding）来表示用户和物品。PinSage[3]建立在GraphSAGE[4]的基础上，将基于图卷积网络的方法应用于具有数十亿节点和边的数据。GATNE[5]考虑了不同的用户行为类型，并利用一种异构的图嵌入方法来学习用户和物品的表示。但是这些方法忽略了用户行为中的序列信息，无法捕捉相邻用户行为之间的相关性。【引入序列推荐】

3、推荐也可以看作一个序列化问题，通过用户的历史，来估计下一个用户感兴趣的物品。很多模型（DIN、DIEN）都从用户的行为序列给出一个整体的Embedding。但是，统一的用户Embedding不能反映用户在一段时间内的**「多种兴趣」**。如下图所示：

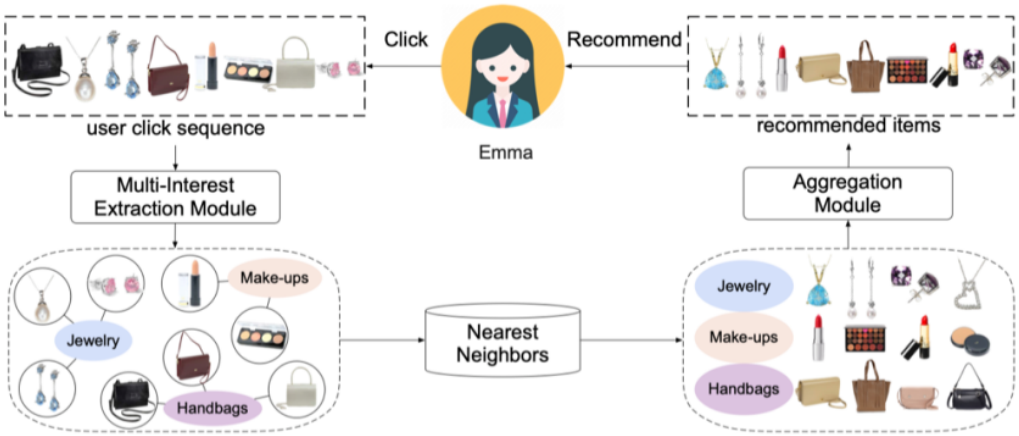

Figure 1: A motivating example of our proposed framework. An e-commerce platform user, Emma, has multiple interests including jewelry, handbags, and make-ups. Our multi-interest extraction module can capture these three interests from her click sequence. Each interest retrieves items from the large-scale item pool based on the interest embedding independently. An aggregation module combines items from different interests and outputs the overall top-N recommended items for Emma.

Emma对珠宝、包、化妆品感兴趣，她在这段时间内应该可能会点击三种类别的物品。

【注】本篇文章对推荐的分类：传统、基于神经网络（端到端、非端到端）、从基于神经网络的模型细化到基于图表示、再从用户历史行为中得到序列化推荐。基于图表示和序列化推荐是目前较为热门的研究方向。大家可以看一下文章的一些引用文献。

3. 创新与贡献

基于上述问题，作者提出一个新的序列化推荐模型：**「controllable multi-interest framework，ComiREC」**。模型分为**「多元兴趣模块」**（multi-interest module）和**「聚合模块」**（aggregation module）。多元兴趣模块从用户行为序列中捕捉多种兴趣，可以在大规模的物品池中检索候选的物品集。然后将这些物品输入到聚合模块以获得总体Top-K推荐。并且聚合模块利用**「可控因素」**（controllable factor）来**「平衡推荐的准确性和多样性」**。

4. 模型方法

4.1 问题定义

假设一个用户集合$u \in \mathcal{U}$和一个物品集合$i \in \mathcal{I}$，对于每一个用户，定义用户序列$(e_1^{(u)}, e_2^{(u)}, \ldots, e_n^{(u)})$，根据时间先后顺序排序。$e_t^{(u)}$记录了第$t$个物品与用户交互。文章所有的字母表示含义见下表：

Table 1: Notations.

Notation	Description
u	a user
i	an item
e	an interaction
$\mathcal{U}$	the set of users
$\mathcal{I}$	the set of items
$\mathcal{I}_u$	the set of testing items of user u
d	the dimension of user/item embeddings
K	the number of interest embeddings
N	the number of candidate items
$\mathbf{V}_u$	the matrix of interest embeddings of user u
$\delta(\cdot)$	indicator function

4.2 多元兴趣模型整体框架

在工业推荐系统的物品池中经常有上百万或亿级别的物品，因此匹配阶段在推荐系统中扮演一个非常重要的角色。

对于**「匹配阶段模型」**：

1. 首先通过用户历史行为序列计算出用户的**「embedding」**，然后基于embedding检索用户的候选物品集合。
2. 然后利用快速K近邻(KNN)算法从大规模物品池中为每个用户生成最接近的物品集。

匹配阶段的决定性因素是根据用户历史行为计算出的**「用户Embedding的质量」**。对于部分序列模型只是建立一个单一的用户Embedding，但是其缺乏表达能力，因为现实世界的用户通常同时对几种类型的物品感兴趣，这些物品通常用于不同的用途，并且在类别上有很大的差异。因此作者提出了用于推荐系统的多元兴趣框架（**「Multi-Interest Framework」**），主要分为多元兴趣提取模块和聚合模块。

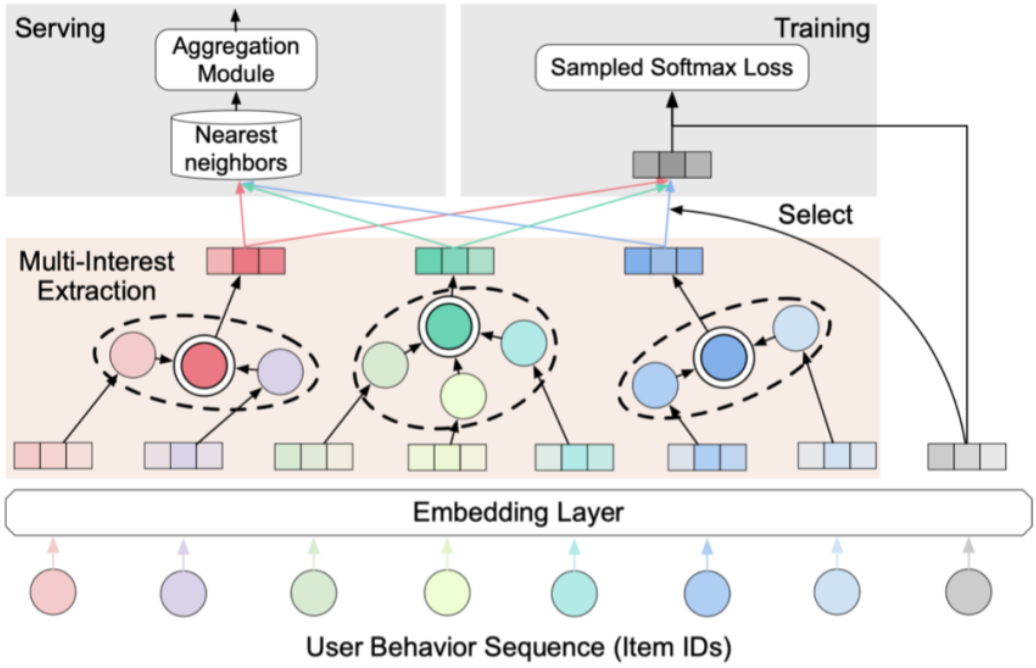

Figure 2: An overview of our model for the sequential recommendation. The input of our model is a user behavior sequence, which contains a list of item IDs. The item IDs are fed into the embedding layer and transformed into the item embeddings. Interest embeddings are generated through the multi-interest extraction module and can be then used for model training and serving. For model training, the nearest interest embedding to the target embedding will be chosen to compute the sampled softmax loss. For serving, each interest embedding will independently retrieve top-N nearest items, which are then fed into the aggregation module. The aggregation module generates the overall top-N items by a controllable procedure that balances the recommendation accuracy and diversity.

4.3 多元兴趣提取模块

本篇文章，作者主要使用两种方法：**「动态路由法」**[^6]（dynamic routing method）和**「自注意力机制法」**（self-attentive method）来作为兴趣提取方法，分别对应的模型称为**「ComiRec-DR」**和**「ComiRec-SA」**。

Dynamic Routing Method

使用**「动态路由方法」**作为用户行为序列的多兴趣提取模块。将用户行为序列的物品embedding列表看作是**「最初的胶囊（primary capsules）」**，多元用户兴趣看作是**「兴趣胶囊（interest capsules）」**。

> “

> 胶囊网络的概念是在2011年首先提出的，并在动态路由方法提出后广为人知。**「胶囊是一组神经元」**，其活动向量代表一个特定类型实体的实例化参数。胶囊的输出向量的长度表示由胶囊所表示的实体在当前输入中的概率。
>
> ”

令$\mathbf{e}_i$表示最初层的胶囊i，可以给出基于最初胶囊的下一层的兴趣胶囊j的计算：

$$\hat{\mathbf{e}}_{j|i} = \mathbf{W}_{ij}\mathbf{e}_i$$

其中$\mathbf{W}_{ij}$表示一个转换矩阵，那么胶囊j的**「总输入」**是所有预测向量$\hat{\mathbf{e}}_{j|i}$的加权和：

$$\mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{e}}_{j|i}$$

其中c_{ij}为**「耦合系数」**(coupling coefficients)，由迭代动态路由过程确定。胶囊i与下一层所有胶囊的耦合系数之和应为1。因此使用“**「routing softmax」**”来计算偶和系数：

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})}$$

其中b_{ij}表示i和j耦合的**「对数先验概」**率。

应用一个非线性的**「“压缩”（squashing）函数」**，以确保短向量缩小到几乎零长度和长向量缩小到略低于1的长度。因此，胶囊j的**「输出」**向量表示为：

$$\mathbf{v}_j = squash(\mathbf{s}_j) = \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2}\frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$$

$\mathbf{s}_j$是胶囊j的总输入，为了计算输出胶囊$\mathbf{v}_j$，我们需要计算基于$\mathbf{v}_j$和$\mathbf{e}_i$的内积的概率分布。$\mathbf{v}_j$的计算依赖于自身;为此，提出了一种**「动态路由方法」**来解决这一问题。其中用户u的输出兴趣胶囊为矩阵$\mathbf{V}_u = [\mathbf{v}_1, \ldots, \mathbf{v}_K] \in \mathbb{R}^{d\times K}$。**「$K$为兴趣胶囊的维度。」**

具体算法如下所示：

Algorithm 1: Dynamic Routing

Input: primary capsules $\mathbf{e}_i$, iteration times r, number of interest capsules K

Output: interest capsules $\{\mathbf{v}_j, j = 1, ..., K\}$

for each primary capsule i and interest capsule j: initialize $b_{ij} = 0$.

for $iter = 1, \cdots, r$ **do**

  for each primary capsule i: $\mathbf{c}_i = \text{softmax}(\mathbf{b}_i)$.

  for each interest capsule j: $\mathbf{s}_j = \sum_i c_{ij} \mathbf{W}_{ij} \mathbf{e}_i$.

  for each interest capsule j: $\mathbf{v}_j = \text{squash}(\mathbf{s}_j)$.

  for each primary capsule i and interest capsule j: $b_{ij} = b_{ij} + \mathbf{v}_j^\top \mathbf{W}_{ij} \mathbf{e}_i$.

return $\{\mathbf{v}_j, j = 1, ..., K\}$

Self-Attentive Method

给出用户行为序列的embeddding矩阵$\mathbf{H}^{d \times n}$，n为用户行为序列的长度，d为用户/物品的embedding的维度。使用**「自注意力机制」**来获得权重向量$\mathbf{a} \in \mathbb{R}^n$:

$$\mathbf{a} = softmax(\mathbf{w}_2^T tanh(\mathbf{W}_1 \mathbf{H}))^T$$

其中$\mathbf{w}_2 \in \mathbb{R}^{d_a}$和$\mathbf{W}_1 \in \mathbb{R}^{d_a \times d}$是可训练化参数。

然后通过权重向量$\mathbf{a}$进行加权和得到向量$v_u = \mathbf{H}\mathbf{a} \in \mathbb{R}^d$。对于自注意方法利用用户序列的顺序，作者在输入Embedding中添加了可训练的**「位置嵌入」**（positional embeddings）。位置embedding与物品embedding的维数d相同，二者可以直接相加。

因此将$\mathbf{w}_2$扩展为$\mathbf{W}_2 \in \mathbb{R}^{d_a \times K}$，$K$为兴趣embedding的维度，attention向量$\mathbf{a}$变为$\mathbf{A}$:

$$\mathbf{A} = softmax(\mathbf{W}_2^T tanh(\mathbf{W}_1 \mathbf{H}))^T$$

用户最终的兴趣$\mathbf{V}_u \in \mathbb{R}^{d \times K}$:

$$\mathbf{V}_u = \mathbf{H}\mathbf{A}$$

模型训练

在多元兴趣提取模块来计算兴趣embedding：

$$\mathbf{v}_u = \mathbf{V}_u[:, argmax(\mathbf{V}_u^T \mathbf{e}_i)]$$

其中$\mathbf{e}_i$表示目标向量i的embedding，$\mathbf{V}_u$表示用户兴趣embedding矩阵。

给出一个训练样本(u, i)，用户embedding$\mathbf{v}_u$，和物品embedding$\mathbf{e}_i$，可以计算用户与物品交互的似然函数：

$$P_\theta(i|u) = \frac{exp(\mathbf{v}_u^T \mathbf{e}_i)}{\sum_{k \in \mathcal{I}} exp(\mathbf{v}_u^T \mathbf{e}_i))}$$

目标函数为最小化似然函数：

$$loss = -\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{I}_u} -logP_\theta(i|u)$$

「由于似然函数求和的计算复杂度大，因此使用采样的softmax技术来训练模型。」

4.4 聚合模块（Aggregation Module）

如何将这些不同兴趣的物品聚合起来，得到总的前n个物品？一种基本而直接的方法是根据**「物品与用户兴趣的内在相似性」**来合并和过滤物品，可以将**「相似性」**公式化为：

$$f(u, i) = \max_{1 \le k \le K} \left(\mathbf{e}_i^\top \mathbf{v}_u^{(k)} \right)$$

其中$\mathbf{v}_u^{(k)}$是用户u的第k个兴趣embedding。这是聚合过程中实现**「推荐精度最大化」**的有效方法。然而，**「目前推荐系统的准确性并不是唯一的问题。「人们更有可能被推荐一些」新的或多样化」**的东西。

给定从用户u的K个兴趣中检索到一个集合$\mathcal{M}$，有$K \cdot N$个物品，找到一个输出集合$\mathcal{S}$，有N个物品（Top-N），使预定义的**「值函数最大化」**。文章使用一个可控制的方法来解决这个问题。使用一个值函数$Q(u, S)$通过可控因子λ来**「平衡推荐的准确性和多样性」**：

$$Q(u, \mathcal{S}) = \sum_{i \in S} f(u, i) + \lambda \sum_{i \in S} \sum_{j \in \mathcal{S}} g(i, j)$$

其中$g(i, j)$是一个多样性或差异性的函数：

$$g(i, j) = \delta(\text{CATE}(i) \neq \text{CATE}(j))$$

其中$CATE(i)$表示物品i的类别，$\delta(\cdot)$是一个指示函数。

如果想要最大的准确率，那么可以令$\lambda = 0$，如果想要最大的多样性，可以令$\lambda = \infty$。文章提出了一个**「贪心推理算法」**来近似最大化值函数$Q(u, S)$：

Algorithm 2: Greedy Inference

Input: Candidate item set $\mathcal{M}$, number of output items N
Output: Output item set $\mathcal{S}$

$\mathcal{S} = \emptyset$
for $iter = 1, \cdots, N$ **do**
  $j = \text{argmax}_{i \in \mathcal{M} \setminus \mathcal{S}} \left(f(u, i) + \lambda \sum_{k \in \mathcal{S}} g(i, k) \right)$
  $\mathcal{S} = \mathcal{S} \cup \{j\}$
return $\mathcal{S}$

4.5 模型对比

文章选取了最近的两个序列化推荐模型进行对比：MIMN[^8]和MIND[^9]。

- MIMN：是**「推荐排序阶段」**的一个代表性工作，它使用记忆网络从长序列的行为数据中捕获用户的兴趣。MIMN和本文模型都针对用户的多种兴趣。对于非常长的连续行为，基于记忆的体系结构也可能不足以捕获用户的长期兴趣。与MIMN相比，我们的模型利用多兴趣提取模块来利用用户的多个兴趣。
- MIND：是最近**「推荐匹配阶段」**的代表性工作，提出了一种Behavior-to-Interest (B2I)动态路由，用于自适应地将用户的行为聚合为兴趣表示向量。与MIND相比，ComiRec-DR采用CapsNet使用的原始动态路由方法，可以捕获用户行为的序列信息。

5. 实验

5.1 实验配置

本文实验配置设计的**「很严格」**，所以详细说一下：

评估所有方法在强泛化下的性能：**「将所有用户」**按8:1:1的比例分成训练/验证/测试集。使用训练用户的整个点击序列来训练模型。为了进行评估，从验证和测试用户中提取前80%的用户行为，从训练过的模型中推断用户Embedding，并通过预测剩余20%的用户行为来计算指标。这种设置比弱泛化更困难，弱泛化在训练和评价过程中都使用了用户的行为序列。

【注】：**「在验证集和测试集中的用户是从未在训练集中出现过的」**。

5.2 数据集

- Amazon：使用Amazon数据集中的Book，每个训练样本被**「截断长度为20」**。【并不算一个长序列】
- Taobao：用户行为来自淘宝推荐系统。只使用点击行为，并按时间对一个用户的行为进行排序。每个训练样本**「被截断长度为50」**。

Table 2: Statistics of datasets.

Dataset	# users	# items	# interactions
Amazon Books	459,133	313,966	8,898,041
Taobao	976,779	1,708,530	85,384,110

5.3 方法比较

实验设置中，模型应该为验证集和测试集的未见用户提供预测。因此，基于因子矩阵分解的方法不适合此设置。

- MostPopular
- YouTube DNN【2016】
- GRU4Rec【2016】
- MIND【2019】

5.4 评估标准

- **「召回率（Recall）」**：使用每个用户的平均来代替全局

$$\text{Recall@N} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{\left|\hat{I}_{u,N}\cap\mathcal{I}_u\right|}{|\mathcal{I}_u|}$$

其中$\hat{I}_{u,N}$表示用户u的Top-N个推荐，$\mathcal{I}_u$表示用户通过模型选出的测试集物品；

- **「点击率（Hit Rate）」**：

$$\text{HR@N} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\delta\left(\left|\hat{I}_{u,N}\cap I_u\right|>0\right)$$

- 「NDCG(Normalized Discounted Cumulative Gain)」：

$$\text{NDCG@N} = \frac{1}{Z}\text{DCG@N} = \frac{1}{Z}\frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\sum_{k=1}^{N}\frac{\delta\left(\hat{i}_{u,k}\in\mathcal{I}_u\right)}{\log_2(k+1)}$$

该指标经常出现在信息检索中

5.5 实验结果

Table 3: Model performance on public datasets. Bolded numbers are the best performance of each column. All the numbers in the table are percentage numbers with '%' omitted.

	Amazon Books						Taobao					
	Metrics@20			Metrics@50			Metrics@20			Metrics@50		
	Recall	NDCG	Hit Rate	Recall	NDCG	Hit Rate	Recall	NDCG	Hit Rate	Recall	NDCG	Hit Rate
MostPopular	1.368	2.259	3.020	2.400	3.936	5.226	0.395	2.065	5.424	0.735	3.603	9.309
YouTube DNN	4.567	7.670	10.285	7.312	12.075	15.894	4.205	14.511	28.785	6.172	20.248	39.108
GRU4Rec	4.057	6.803	8.945	6.501	10.369	13.666	5.884	22.095	35.745	8.494	29.396	46.068
MIND	4.862	7.933	10.618	7.638	12.230	16.145	6.281	20.394	38.119	8.155	25.069	45.846
ComiRec-SA	**5.489**	8.991	11.402	**8.467**	**13.563**	17.202	**6.900**	**24.682**	41.549	9.462	31.278	51.064
ComiRec-DR	5.311	**9.185**	**12.005**	8.106	13.520	**17.583**	6.890	24.007	**41.746**	**9.818**	**31.365**	**52.418**

为了与其他模型进行公平比较，在聚合模块中设置了$\lambda = 0$。详细说明了「如何检索框架的top-N物品」：

1. 用户的每个兴趣都独立地检索top-N候选物品。因此，模型为每个用户总共检索$K \cdot N$物品。
2. 根据物品Embedding和相应兴趣Embedding的「内积」对物品进行排序。排序后，$K \cdot N$个物品中的top-N个物品被视为模型的最终候选物品。

如上图所示，ComiRec模型比所有的最先进的模型在所有的评估标准上有很大的差距。GRU4Rec比其他仅为每个用户输出单一Embedding的模型中获得了最好的性能。与MIND相比，由于「动态路由方法的不同」，ComiRec-DR获得了更好的性能。ComiRec-SA通过自我注意机制捕捉用户兴趣的能力较强，其结果与ComiRec-DR相当。

「参数敏感性」

Table 4: Model performance of parameter sensitivity. All the numbers are percentage numbers with '%' omitted.

	Amazon Books		Taobao	
Metric@50	Recall	NDCG	Recall	NDCG
ComiRec-SA (K=2)	8.835	**14.273**	9.935	32.873
ComiRec-SA (K=4)	8.467	13.563	9.462	31.278
ComiRec-SA (K=6)	**8.901**	14.167	9.378	31.020
ComiRec-SA (K=8)	8.547	13.631	9.493	31.196
ComiRec-DR (K=2)	7.081	12.068	9.293	30.735
ComiRec-DR (K=4)	8.106	13.520	9.818	31.365
ComiRec-DR (K=6)	7.904	13.219	10.836	**34.048**
ComiRec-DR (K=8)	7.760	12.900	**10.841**	33.895

调节兴趣K的敏感性。两个模型显示了这个超参数的不同性质。

5.6 可控制的学习

Table 5: Model performance of Amazon dataset for the controllable study. All the numbers are percentage numbers with '%' omitted.

	ComiRec-SA (K=4)		ComiRec-DR (K=4)	
Metric@50	Recall	Diversity	Recall	Diversity
$\lambda = 0.00$	**8.467**	23.237	**8.106**	19.036
$\lambda = 0.05$	8.347	38.808	7.931	42.915
$\lambda = 0.10$	8.229	46.731	7.850	46.258
$\lambda = 0.15$	8.142	51.135	7.820	46.912
$\lambda = 0.20$	8.086	53.671	7.783	47.581
$\lambda = 0.25$	8.034	**55.100**	7.764	**48.375**

为了获得每个用户最终的top-N候选物品，提出了一个新的模块来聚合根据每个用户的不同兴趣检索到的物品。除了对推荐达到较高的预测精度外，有研究认**「为推荐需要多样化，以避免单调，提高客户体验」**。

文章提出的聚合模块可以**「控制推荐准确度和多样性的平衡」**。使用以下基于物品类别的个体多样性定义:

$$\text{Diversity@N} = \frac{\sum_{j=1}^{N}\sum_{k=j+1}^{N}\delta\left(\text{CATE}\left(\hat{i}_{u,j}\right) \neq \text{CATE}\left(\hat{i}_{u,k}\right)\right)}{N \times (N-1)/2}$$

聚合模块可以通过选择一个合适的超参数，在准确性和多样性之间实现最优的平衡。

总结

本文对我的启发很大，不仅是模型，重要的是对整个目前序列化推荐发展的概述。所以想要清楚的了解自己的研究方向发展情况，除了综述，还可以从最新的文章中得到。

最近在Github上开源了一个项目：**「Recommender System with TF2.0」**，目前已有57个star，地址是：https://github.com/BlackSpaceGZY/Recommender-System-with-TF2.0。目前正准备把本文的模型使用TF2.0进行复现，希望大家可以star一下，感谢！

参考文献

[1]: Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. arXiv preprint arXiv:1702.08734 (2017).
[2]: Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining explicit and implicit feature interactions for recommender systems. In KDD' 18. ACM, 1754–1763.
[3]: Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In KDD' 18. ACM, 974–983.
[4]: Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In NIPS' 17. 1024–1034.
[5]: Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang.2019. Representation learning for attributed multiplex heterogeneous network. In KDD' 19. 1358–1368.
[6]: Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In NIPS' 17. 3856–3866.
[7]: Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. 2011. Transforming auto-encoders. In ICANN' 11. Springer, 44–51.
[8]: Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019.