论文题目：A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations [Recsys20]

背景

在推荐场景中，多任务学习是非常常见的做法。例如在做视频推荐的时候，我们不仅要考虑一个视频的CTR，还要考虑其他的指标(engagement, satisfaction)。最后的打分函数则是很多指标的加权求和。例如下面这个打分公式综合考虑了VTR(View-Through Rate), VCR(View-Completion ratio), CMR (comment rate), SHR(share rate), 和视频长度(video-len).

$$score = {p_{VTR}}^{w_{VTR}} \times {p_{VCR}}^{w_{VCR}} \times {p_{SHR}}^{w_{SHR}} \times \cdots \times {p_{CMR}}^{w_{CMR}} \times f(video\_len), \quad (1)$$

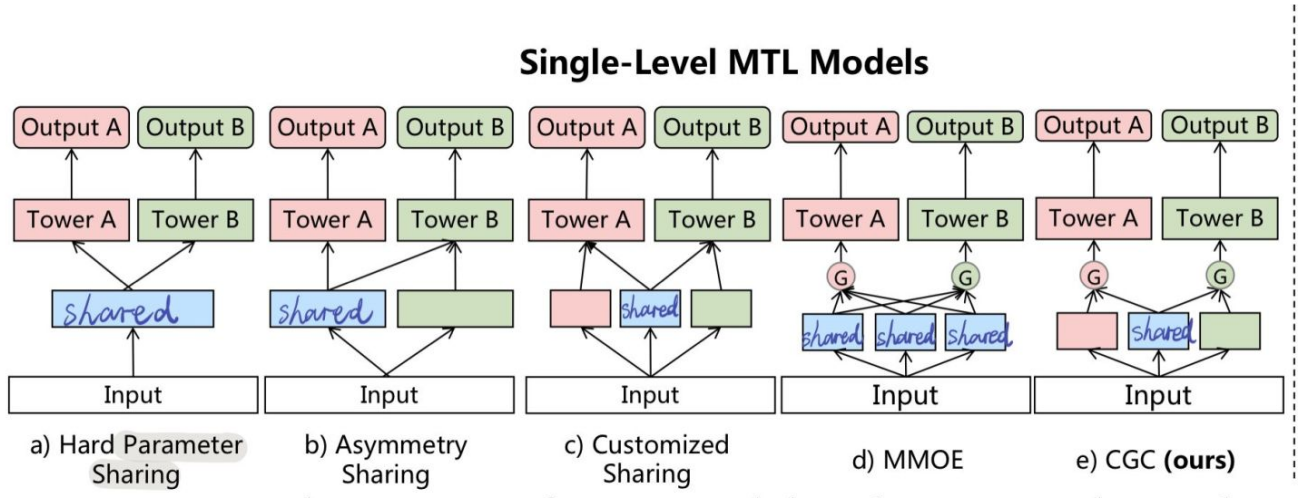
文章首先提出多任务学习中不可避免的两个缺点：

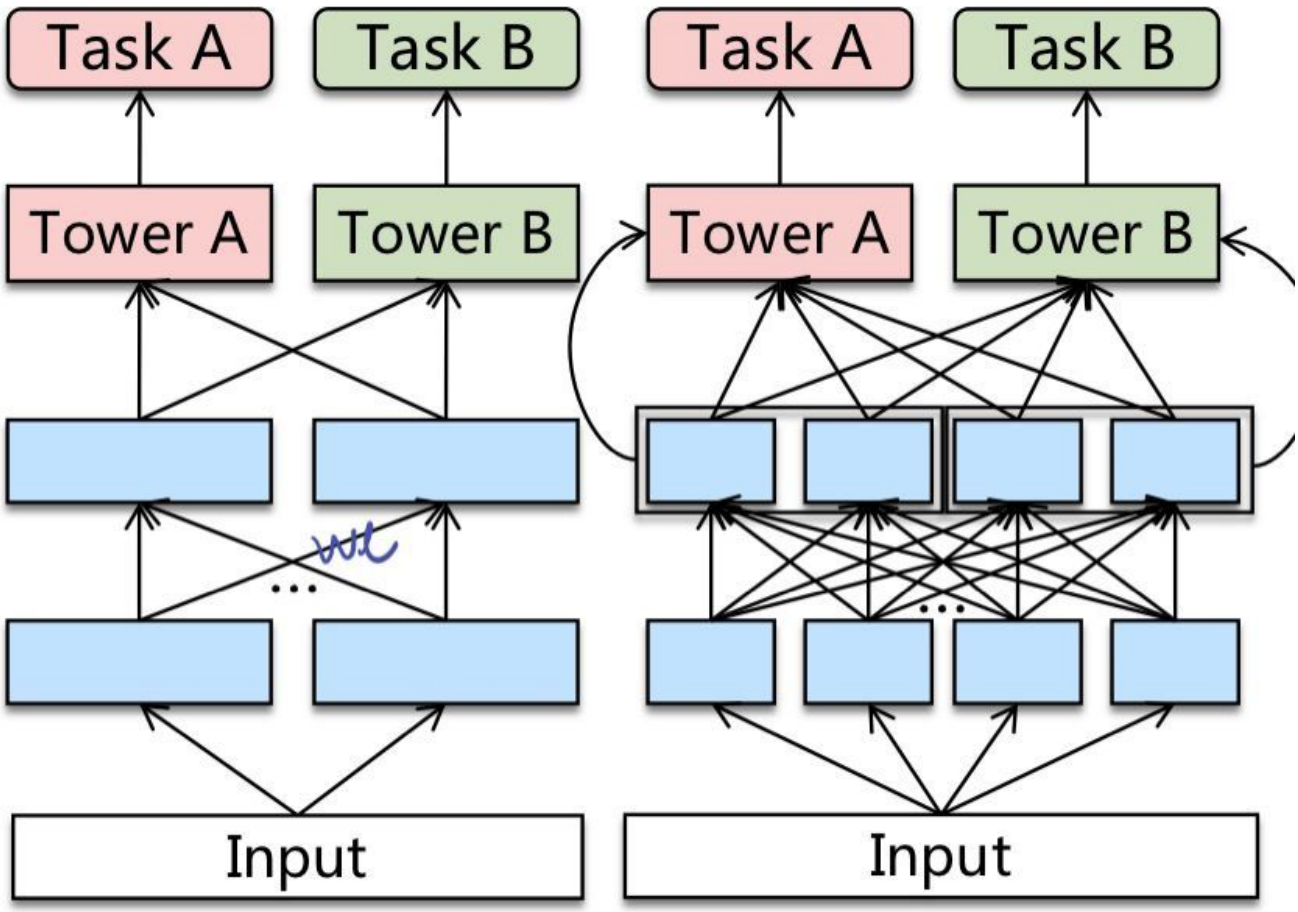
- **Negative Transfer**. 针对相关性较差的多任务来说，使用hard parameter sharing这种方式通常会出现negative transfer的现象，原因就是因为任务之间存在冲突的话，会导致模型无法有效进行参数的学习，学的不伦不类。也就是说，使用k个多目标优化的效果不如训k个单独的网络来得好。
- **跷跷板现象。**对于一些任务相关性比较复杂的场景，通常会出现跷跷板现象，即，如果想要提升一部分任务的效果，就必须牺牲其他任务的效果。（例如很多多任务学习的模型都面临一个问题：要想提升VTR准确率，VCR准确率就会下降；反之亦然）

为了解决“跷跷板”现象，PLE将共享的部分(shared components)和每个任务特定的部分(task-specific components)**显式地分开**建模，并使用多层的网络叠加来把握高阶的信息，实现”渐进的共享“。

> "adopts a progressive routing mechanism to extract and separate deeper semantic knowledge gradually".

多任务模型对比





f) Cross-Stitch Network

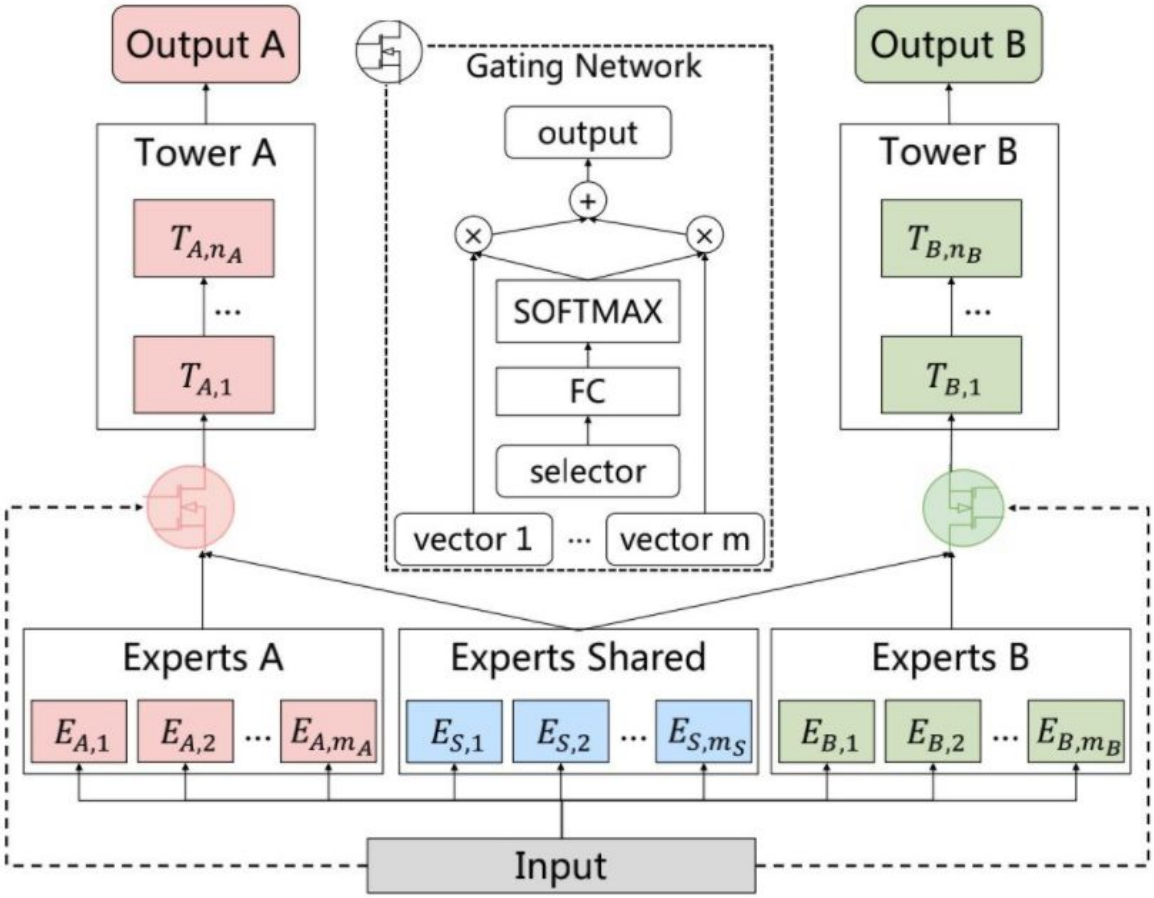
g) Sluice Network

- shared bottom方法：由于任务的冲突，会有negative transfer问题；
- cross-stitch network：通过学习层之间的**转换矩阵**来完成不同sub-module之间的联系，不同tower使用不同的连接参数，来把握不同task的差异。但是每个sub-module的参数都是共享的。
- MMOE的底层expert是共享的，即不同的任务都使用了这若干个expert的输出，只不过通过门控网络(就是注意力机制)，将若干个expert的输出进行了加权求和，每个任务的门控网络都是不同的，所以不同任务会对experts以不同的权重进行加权求和。相比之下，PLE则显式的分开了共享的expert（下图蓝色部分）和task-specific expert（下图红色和绿色部分），防止冲突的任务梯度同时更新共享的参数。虽然MMoE模型在理论上可以达到PLE的效果，但是在实际训练过程中很难收敛到这种情况。
- 还有之前我们讲过的SNR模型，通过**网络结构搜索(NAS)**的方式给不同的task以不同的网络结构模型，比MMOE更加灵活。

PLE 模型结构

1) CGC (custom gate control)

首先，只考虑一层的抽取网络，就是CGC。



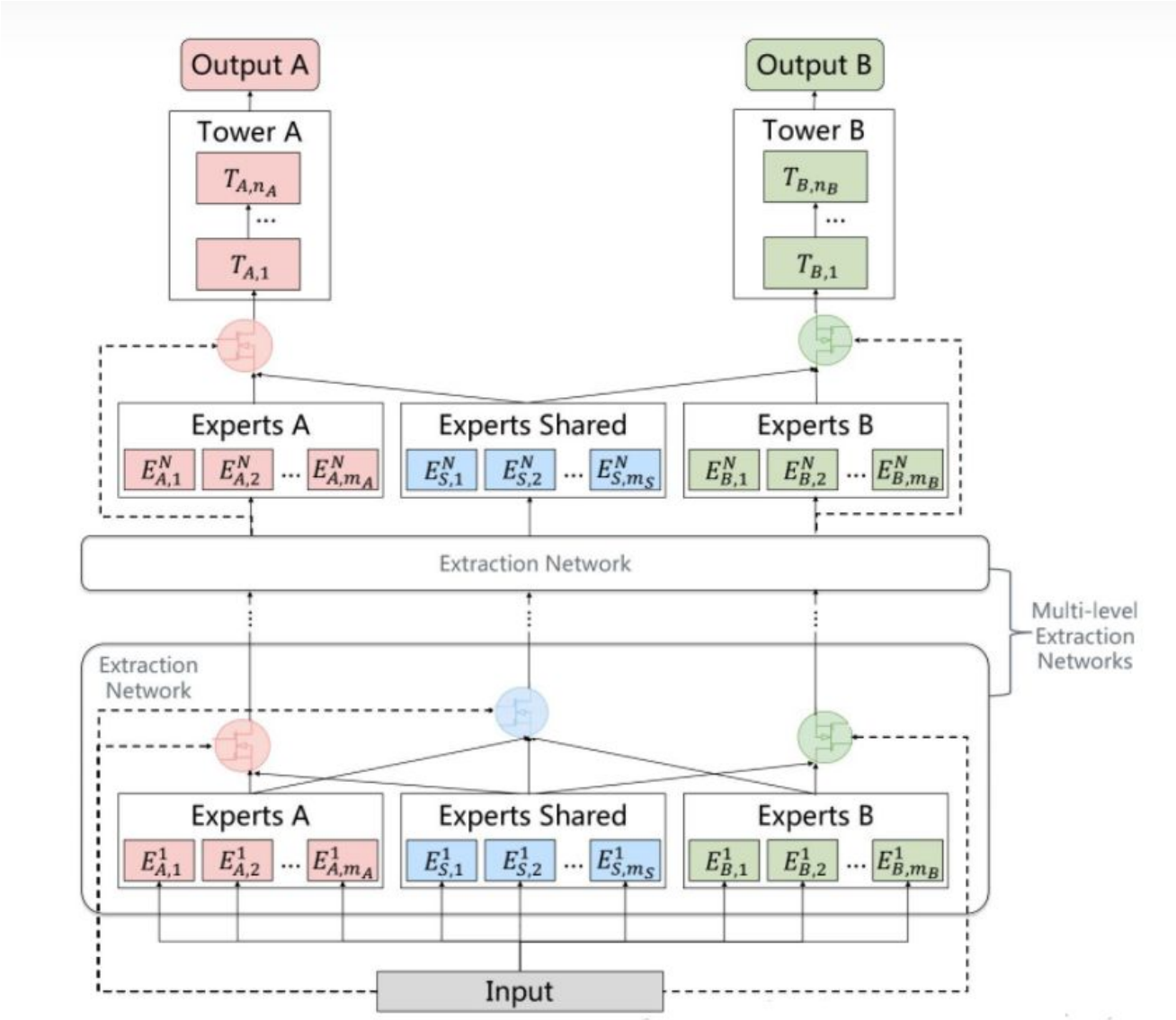
从图中的网络结构可以看出，CGC的底层网络主要包括shared experts和task-specific experts。其中对于任务A，有 m_A 个expert；对于任务B，有 m_B 个expert；还有 m_S 个任务A、B所共享的expert。每个子任务tower的输入是对task-specific和shared两部分的若干个expert vector进行加权求和（也是通过gate决定attention权重）。

这样对expert做一个显式的分割，可以让任务特定(task-specific)的expert只受自己任务梯度的影响，不会受到其他任务的干扰；而只有shared expert才受多个任务的混合梯度影响。

这里的gating network（其实就是attention network）是以**input**作为"裁判"（**selector**），通过单层全连接+softmax得到分配给不同expert的attention权重。

2) PLE (progressive layered extraction)

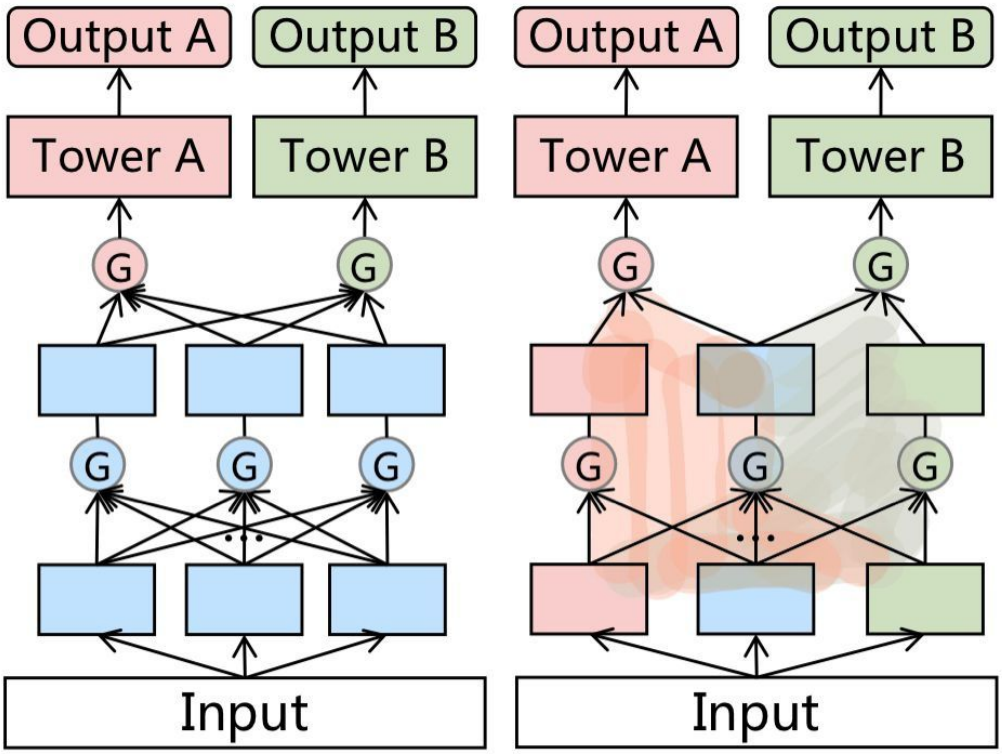
PLE就是上述CGC网络的多层叠加，以获得更加丰富的表征能力。具体网络结构如下图所示：



在底层的Extraction网络中，除了各个task-specific有门控网络外，shared experts也有门控网络。这样，task-specific expert的输出只融合了任务本身的expert和shared expert；而shared expert的输出是融合了所有expert的。

将任务A、任务B和shared expert的输出输入到下一层，下一层的gating network是以这三个**上一层输出的结果作为selector**，而不是用input作为selector，这样可以利用好更加抽象的高层信息。

我们再来直观地对比一下多层MMOE和PLE的联系和区别：



h) ML-MMOE　　i) PLE (ours)

可以看出，MMOE的每一层expert之间都是全部连接的，而PLE则是"局部连接"，只和自己任务的expert与shared expert连接。但是，从上图画的彩色部分可以看到，tower A的梯度也是会传播到Tower B对应的expert中去的，说明PLE中expert的分隔是渐进的(progressive). 文中把这种progressive separation比作化学中的提取过程。

另外，PLE明显是MMOE的子集，但是MMOE在现实中很难收敛成这个样子，所以PLE就显式的规定了这样的结构。

3. 实验

（1）线下准确率实验

下图中灰色的“zig-zag”说明很多多任务学习模型在VTR/VCR预估中都有跷跷板现象，即：想要提升VTR准确率就必须牺牲VCR准确率，反之亦然。

Table 1: Performance on VTR/VCR Task Group

Models	AUC VTR	MSE VCR	MTL Gain VTR	MTL Gain VCR
Single-Task	0.6787	0.1321	-	-
Hard Parameter Sharing	0.6740	0.1320	-0.0047	+1.8E-5
Asymmetric Sharing	0.6823	0.1346	+0.0036	-0.0025
Cross-Stitch	0.6740	0.1320	-0.0047	+1.6E-5
Sluice Network	0.6825	0.1329	+0.0038	-0.0008
Customized Sharing	0.6780	0.1318	-0.0007	+0.0002
MMOE	0.6803	0.1319	+0.0016	+0.0001
ML-MMOE	0.6815	0.1329	+0.0028	-0.0009
CGC	**0.6832**	0.1320	**+0.0045**	+3.5E-5
PLE	**0.6831**	**0.1307**	**+0.0044**	**+0.0013**

（2）线上AB测试

随机把用户进行分桶，每个桶使用一个多任务模型，以VTR、VCR作为评估标准，发现线上指标都有了明显提升：

Table 3: Improvement over Single-task Model on Online A/B Test

Models	Total View Count	Total Watch Time
Hard Parameter Sharing	-1.65%	-1.79%
Sluice Network	+0.75%	+1.29%
MMOE	+1.94%	+1.73%
ML-MMOE	+1.96%	+1.10%
CGC	+3.92%	+2.75%
PLE	**+4.17%**	**+3.57%**

参考：

被包养的程序猿丶：腾讯PCG RecSys2020最佳长论文——视频推荐场景下多任务PLE模型详解