第 24 章 个性化网络参数

在工业级推荐系统中，不同用户群体的行为模式和内容偏好往往存在显著差异。例如，新用户缺少历史行为，可能更倾向于浏览短视频并尝试不同类型的内容；活跃用户拥有较丰富的交互历史，其兴趣通常更容易被模型识别；而沉默用户和回流用户则可能表现出不同于活跃用户的内容偏好与交互习惯。这些差异不仅影响用户的反馈分布，也会影响推荐模型在不同群体上的预测效果。

传统推荐模型通常使用统一的网络参数处理所有用户，并通过用户 ID、用户属性、历史行为等特征实现个性化。然而，提供个性化特征并不意味着模型一定能够充分利用这些信息。当不同用户群体的样本数量存在较大差异时，模型训练容易受到主流用户群体的主导，使少数群体的特征难以对模型参数产生足够的影响。即使模型能够识别某些用户群体的特征，这些特征也可能在训练过程中被其他更容易优化的特征所替代，导致模型无法充分学习不同群体的行为规律。

个性化网络参数（Personalized Network Parameters）旨在从模型结构出发，增强网络对不同用户及用户群体的适应能力。其核心思想如下：

定义 24.1

个性化网络参数的方法在保留共享模型主体的基础上，通过引入由个性化特征控制的参数、子模块或者激活门控，使得不同用户能够以不同方式使用网络中的计算单元。与仅依赖输入特征表达用户差异的方法相比，这类方法进一步允许不同用户对模型内部的计算过程产生不同影响，从而提升模型对长尾用户群体和特殊场景的建模能力。



需要注意的是，个性化网络参数并不等同于为每个用户单独训练一套模型，也不意味着必须为不同用户维护完全独立的参数。考虑到工业级推荐系统中的用户规模、模型容量和在线推理成本，实际方案通常采用共享参数与个性化模块相结合的方式，在控制额外计算开销的同时增强模型的表达能力。

本章围绕个性化网络参数的典型设计展开。需要说明的是，LHUC（Learning Hidden Unit Contributions）同样可以归入个性化网络参数的范畴，其通过调节网络隐藏单元的激活贡献实现个性化建模，相关内容已在第15章《特征工程》中介绍，因此本章不再赘述。在此基础上，本章首先介绍 POSO（Personalized COld Start MOdules）方法，分析其如何通过个性化模块缓解少数用户群体的特征在模型训练中被主流用户群体淹没的问题；随后介绍 APG、PEPNet 和 APPNet 等代表性方法，从动态参数生成、个性化门控以及自动特征划分等角度，进一步讨论个性化网络参数的设计思路及其在推荐系统中的应用。

24.1 POSO 模型

POSO 模型源自于快手科技发表在 2021 年的论文《POSO: Personalized Cold Start Modules for Large-scale Recommender Systems》[2]。该方法最初面向大规模推荐系统中的用户冷启动问题，但其核心思想并不局限于新用户，也可以扩展到活跃用户、回流用户以及新物品等场景。

24.1.1 问题背景与核心思想

用户冷启动通常被认为是由历史行为不足导致的建模问题。新用户刚进入推荐系统时，系统尚未积累足够的点击、观看、点赞等交互记录，因此难以准确判断其兴趣。常见的解决思路包括利用用户属性和上下文特征进行预测，或者借助迁移学习等方法，将其他用户积累的知识迁移到新用户。

然而，POSO 模型指出，冷启动问题不仅来自行为数据不足，还与不同用户群体之间的样本分布不均衡有关。在大规模推荐系统中，新用户通常只占全部训练样本的一小部分，而大部分样本来自已有较多交互记录的常规用户。当模型使用统一的网络参数进行训练时，主流用户群体产生的梯度往往占据主导地位。即使输入中

包含能够区分新老用户的个性化特征，模型也可能逐渐依赖其他更容易优化的特征，而没有充分利用这些个性化信息。

POSO 模型将这种现象称为**个性化信息的淹没（Submergence of Personalization）**。其关键在于：模型虽然具备表达用户差异的输入特征，但这些特征未必能够有效影响网络内部的计算过程。因此，仅增加用户标识或冷启动标记，并不能保证模型充分学习少数群体的行为模式。

图24.1展示了在工业实践中观察到的新用户与常规用户的行为差异。图中的各项指标以常规用户的平均行为为参照，展示新用户相对于常规用户的行为变化。可以看到，新用户的视频播放量（Video View，VV）相对较低，但完播率（Finish-View Rate）较高、单次播放时长相对较短，同时点赞行为也更加频繁。这些差异表明，新用户并不是常规用户在行为数据较少时的简单版本，而可能具有不同的内容消费模式和反馈分布。

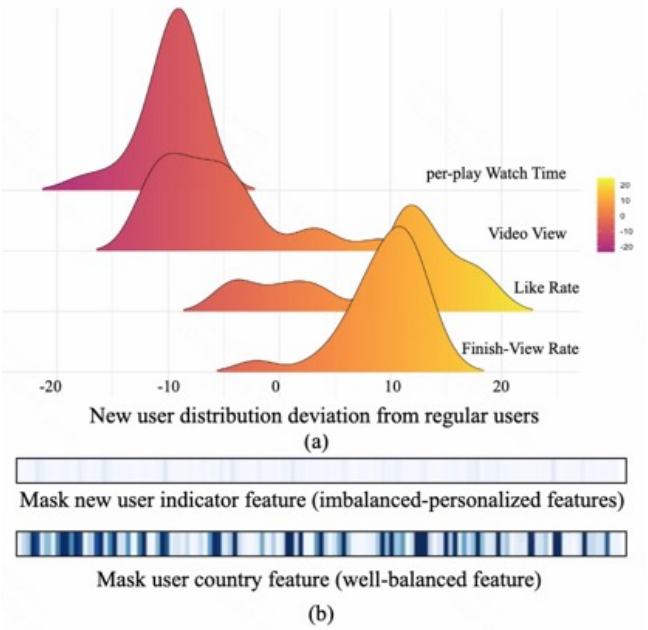


图 24.1: 新用户与常规用户的行为分布差异，以及个性化特征对模型激活的影响。

图24.1(b) 进一步展示了个性化信息被淹没的证据。作者在训练完成的模型中分别屏蔽不同输入特征，并观察模型中间激活的变化。结果发现，屏蔽新用户标记等分布不均衡的特征，对模型激活的影响可能很小。相比之下，屏蔽分布相对均衡的用户国家等特征，却可能引起更明显的激活变化。这说明模型对不同特征的实际依赖程度并不一致。由于新用户样本较少，即使模型接收到了相应的个性化信息，训练过程也可能没有充分建立这些特征与预测结果之间的有效联系。

针对这一问题，一种直接的个性化方案是为每个用户单独维护一套预测模型。假设用户 u 的输入为 $\mathbf{x}_u$，其预测函数为 f_u，则可以写为

$$y_u = f_u(\mathbf{x}_u) \tag{24.1}$$

这种方案能够为不同用户提供充分独立的参数，但用户规模通常达到千万甚至亿级，为每个用户维护独立模型不仅会带来巨大的参数存储和训练成本，也会使在线服务难以部署和维护。

另一种方案是为不同用户群体建立多个子模型，再根据用户的个性化特征融合这些模型的预测结果。假设系统包含 N 个子模型，用户 u 对应的门控权重为 $g_i(\mathbf{x}_{pc})$，则其预测可以表示为

$$y_u = \sum_{i=1}^{N} g_i(\mathbf{x}_{pc}) f^{(i)}(\mathbf{x}_u) \tag{24.2}$$

其中，$\mathbf{x}_{pc}$ 表示个性化编码（Personalization Code），即用于区分用户群体或个体状态的关键特征，$f^{(i)}$ 表示第 i 个子模型。这种设计能够让不同用户以不同权重使用各个子模型，但如果为完整预测网络复制多套参数，仍然会产生较高的计算开销。POSO 模型的关键改进是，不必复制整个预测网络，而是将个性化机制嵌入现有网络的

部分中间模块，仅对这些模块进行个性化扩展，其余结构继续共享。

对于一个输入为 $\mathbf{x}$、输出为 $\hat{\mathbf{x}}$ 的中间模块，POSO 模型将原始计算替换为

$$\hat{\mathbf{x}} = C \sum_{i=1}^{N} g_i(\mathbf{x}_{pc}) f^{(i)}(\mathbf{x}) \tag{24.3}$$

其中，$f^{(i)}$ 表示第 i 个个性化子模块，$g_i(\mathbf{x}_{pc})$ 是由个性化特征产生的门控权重，C 为用于控制输出整体尺度的校正因子。与为每个用户维护独立模型不同，POSO 模型只需要在指定的网络层中引入多个子模块，并通过个性化门控融合它们的输出。

POSO 模型的设计主要体现为以下三个特点。第一，**显式强化个性化信息**。门控网络直接使用原始个性化特征，而不是仅依赖主干网络已经提取的中间表示。这样，即使主干网络在训练过程中没有充分利用某些少数群体特征，门控网络仍然可以根据这些特征控制不同子模块的参与程度。第二，**能够灵活嵌入已有网络**。POSO 模型并不是一种独立的完整推荐模型，而是一种对现有网络模块进行个性化改造的方法。它可以应用于全连接层、多头注意力和多任务专家网络等结构，在保留原有网络主体的基础上增强模型的个性化表达能力。第三，**尽量保持模块之间的结构独立性**。POSO 模型的多个子模块接收相同的输入，其输出经过门控融合后形成统一表示，再传递给后续网络。它不需要为不同用户建立相互依赖的完整网络分支，因此能够较为自然地集成到已有推荐模型中。

24.1.2 POSO 的模块设计

POSO 模型以快手的大规模推荐模型为应用背景。该模型采用经典的 Embedding & MLP 范式，并结合多头注意力（Multi-Head Attention，MHA）建模用户历史行为，使用多门控混合专家网络（Multi-gate Mixture-of-Experts，MMoE）处理多个预测目标。

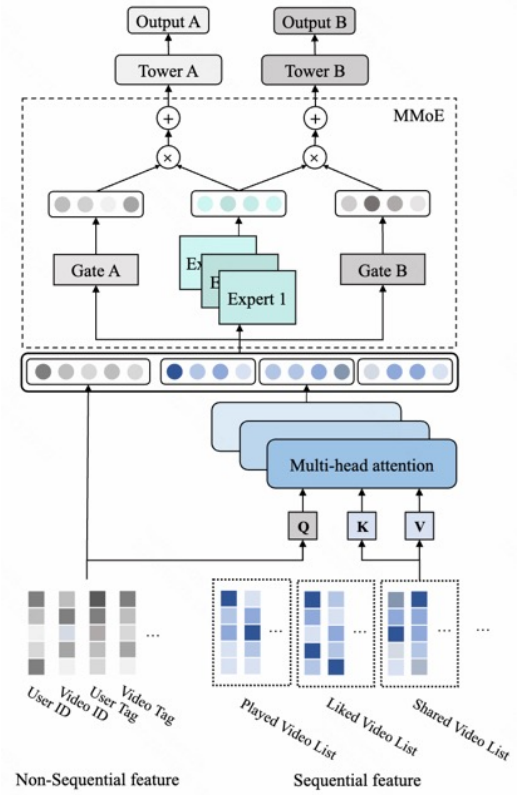


图 24.2: 快手大规模推荐系统中的排序模型。

如图24.2所示，模型整体可以划分为三个阶段：Embedding 生成、序列特征建模和多任务优化。首先，用户 ID、物品属性及上下文等非序列特征通过 Embedding 查表映射为低维向量。用户过去观看的视频等序列特征则

经过 Embedding 映射后输入 MHA，由当前用户的非序列特征生成 Query，由历史行为表示生成 Key 和 Value，从而提取与当前推荐请求相关的兴趣信息。随后，模型将非序列特征表示与经过注意力聚合的序列表示拼接，形成共享中间表示。最后，MMoE 模块根据不同预测任务的门控权重融合多个专家网络的输出，再由各任务独立的预测层生成点击率、长时观看率、点赞率等预测结果。

在这一结构中，不同模块承担着不同的建模职责，因此个性化改造也不必采用完全相同的方式。POSO 模型分别针对全连接网络、多头注意力和多任务专家网络设计了对应的个性化模块，其结构如图24.3所示。

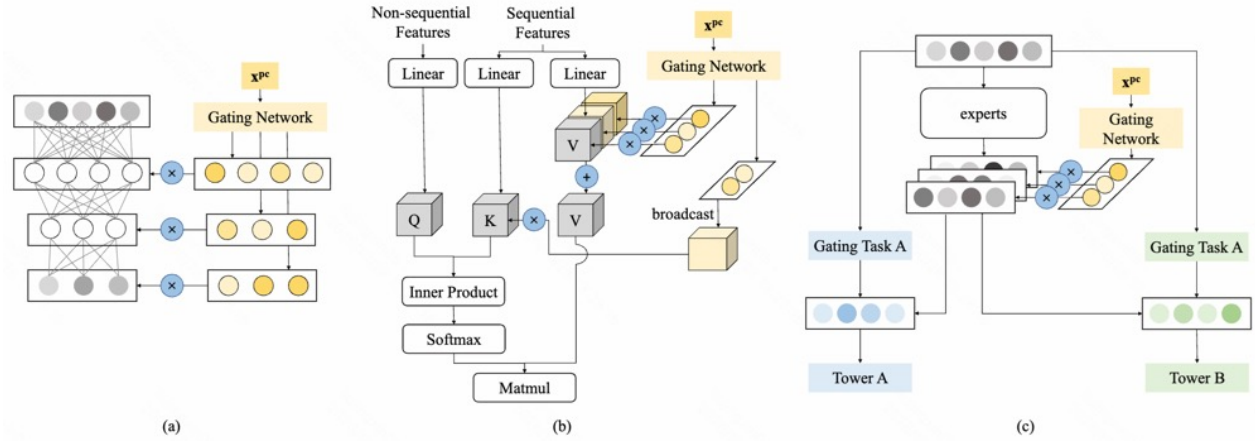


图 24.3: POSO 的三类个性化模块。

24.1.2.1 POSO (MLP) 模块

对于一个全连接层，其基本计算形式为

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x}) \tag{24.4}$$

其中，$\mathbf{x}$ 为输入表示，$\mathbf{W}$ 为可训练权重矩阵，σ 为激活函数。一种直接的个性化方式，是为不同用户群体建立多套全连接层参数，再通过门控网络融合各个子模块的输出。不过，这种设计会增加参数量和计算开销。POSO (MLP) 模型采用了更轻量的实现，即保留原有全连接层的权重矩阵，仅在其输出激活上施加由个性化特征生成的门控向量。具体形式如下：

$$\hat{\mathbf{h}} = C \cdot \mathbf{g}(\mathbf{x}_{pc}) \odot \sigma(\mathbf{W}\mathbf{x}) \tag{24.5}$$

其中，$\mathbf{g}(\mathbf{x}_{pc})$ 为个性化门控向量，$\odot$ 表示逐元素乘法，C 为尺度校正因子。

假设隐藏层的维度为 d，则门控网络为该层的每个激活单元生成一个对应的权重。对于同一输入表示，不同用户可以获得不同的门控向量，从而以不同程度激活各个隐藏单元。如果某些隐藏单元更适合描述新用户的行为模式，门控网络便可以提高这些单元在新用户样本上的贡献；对于常规用户，则可以采用另一种激活组合。

这种设计与传统的 MLP 共享同一组基础权重，个性化主要体现在隐藏单元的激活强度上。因此，它可以在不显著改变原有网络主体的情况下增强模型对不同用户群体的区分能力。如图24.3(a) 所示，POSO 模型可以逐层应用于 MLP，通过连续的个性化激活调制，使用户差异能够影响网络中多个层级的表示学习。

24.1.2.2 POSO (MHA) 模块

多头注意力常用于推荐系统中的用户历史行为建模。其核心是根据当前请求与历史行为之间的相关性，为不同历史行为分配不同的注意力权重。对于单个注意力头，计算形式为

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}}\right)\mathbf{V} \tag{24.6}$$

其中，$\mathbf{Q}$、$\mathbf{K}$ 和 $\mathbf{V}$ 分别为 Query、Key 和 Value，d_h 为注意力头的特征维度。

在工业级推荐系统中，Query 通常由当前用户及请求上下文等非序列特征生成，而 Key 和 Value 则主要由历史行为序列映射得到。三者在注意力计算中承担不同的职责，因此 POSO 模型并没有简单地对整个注意力模块复制多套参数，而是根据各部分对个性化信息的敏感程度采用不同的处理方式。

首先，Query 已经由当前用户的非序列特征生成，包含较强的用户个性化信息，因此 POSO 模型保留原有 Query，不额外引入个性化分支。其次，Key 决定历史行为与当前请求之间的匹配关系，直接为其引入多套完整参数会带来较多冗余。POSO 模型因而采用轻量门控，对 Key 的各维表示进行逐元素调制：

$$\hat{\mathbf{K}} = \mathbf{G}_K(\mathbf{x}_{pc}) \odot \mathbf{K} \tag{24.7}$$

这里的 $\mathbf{G}_K$ 根据个性化特征生成门控向量，用于调整不同维度在注意力匹配中的作用。

最后，Value 直接参与注意力输出的加权聚合，决定了聚合结果中包含哪些历史行为信息。为了提供更充分的个性化表达能力，POSO 模型为 Value 保留多个个性化子模块，并通过个性化门控融合它们的输出。将上述设计组合起来，可以得到

$$\begin{aligned}\hat{\mathbf{O}} = C \cdot \mathrm{softmax}\left(\frac{\mathbf{Q}\hat{\mathbf{K}}^\top}{\sqrt{d_h}}\right) \\ \cdot \sum_{i=1}^{N} g_i(\mathbf{x}_{pc})\mathbf{V}^{(i)}\end{aligned} \tag{24.8}$$

其中，$\mathbf{V}^{(i)}$ 表示第 i 个 Value 子模块的输出，$g_i(\mathbf{x}_{pc})$ 为对应的个性化门控权重。对于多头注意力，各个注意力头分别执行上述计算，最后拼接各头输出。

由此可见，POSO (MHA) 模型对 Query、Key 和 Value 采用了不同程度的个性化处理。其中 Query 保持不变，Key 通过轻量门控进行调整，Value 则使用多个个性化子模块。这种设计兼顾了注意力机制的计算特点与个性化表达需求，避免对整个注意力模块进行无差别扩展。如图24.3(b) 所示，个性化门控分别作用于 Key 和 Value，使不同用户能够以不同方式选择和聚合历史行为信息。

24.1.2.3 POSO (MMoE) 模块

如图24.3(c) 所示，POSO (MMoE) 在原有多任务专家结构中增加个性化门控，再与任务门控共同决定专家输出的贡献。由于主要改造集中在专家激活和输出融合环节，因此能够较为方便地集成到已有的多任务推荐模型中。传统的 MMoE 模型通过多个共享专家提取不同类型的特征表示，再由任务专属的门控网络决定各个专家对当前任务的贡献。对于任务 t，其计算过程可以表示为

$$\mathbf{g}_t(\mathbf{x}) = \mathrm{softmax}(\mathbf{W}_t\mathbf{x}) \tag{24.9}$$

$$\hat{\mathbf{x}}_t = \sum_{j=1}^{N_e}[\mathbf{g}_t(\mathbf{x})]_j e^{(j)}(\mathbf{x}) \tag{24.10}$$

其中，N_e 为专家数量，$e^{(j)}$ 表示第 j 个专家，$\mathbf{g}_t$ 为任务 t 对应的门控网络。最终，任务专属预测层根据 $\hat{\mathbf{x}}_t$ 输出相应的预测结果。

MMoE 模型主要通过任务门控建模不同预测目标之间的差异，但专家的激活方式并不直接受到用户群体的显式控制。而 POSO (MMoE) 模型在专家输出上进一步引入个性化门控，使模型能够同时根据用户特征和预测任务决定使用哪些专家。假设个性化门控为 $g_i(\mathbf{x}_{pc})$，则 POSO (MMoE) 模型可以写为

$$\begin{aligned}\hat{\mathbf{x}}_t = C\sum_{i=1}^{\hat{N}} g_i(\mathbf{x}_{pc})[\mathbf{g}_t(\mathbf{x})]_i \\ \cdot e^{(i)}(\mathbf{x})\end{aligned} \tag{24.11}$$

其中，$\hat{N}$ 表示最终参与融合的专家数量。在 POSO 模型的实现中，可以通过重新组织个性化子模块与任务专家的组合，将二者的作用整合到同一组专家门控中。为了控制额外开销，实际实现不必为每个用户群体和每个任务都复制完整的专家集合。

从建模的角度来看，个性化门控决定哪些专家更适合当前用户或用户群体，任务门控则决定这些专家对当

前预测目标的贡献。二者共同作用，使专家不仅能够学习任务之间的差异，还能够逐步形成对不同用户群体的专门化表示。例如，当新用户样本在某个个性化门控上的权重较高时，对应专家就会更多地参与这些样本的前向计算，并获得相应的训练信号。随着训练进行，该专家有机会学习更适合新用户的特征组合与行为模式。与单纯依靠 MMoE 模型从输入表示中自行发现用户差异相比，POSO 模型将用户群体信息直接引入专家的激活过程，从而增强了专家对特定群体的适应能力。

24.1.3 POSO 在冷启动场景中的应用

POSO 模型的个性化门控需要依赖能够区分用户群体或物品状态的特征。对于不同的冷启动场景，个性化编码的设计也有所区别。

首先是用户冷启动场景。对于新用户，仅使用新老用户二值标记通常不够精细，因为不同新用户的行为积累速度和兴趣探索状态可能存在差异。POSO 模型使用累计观看次数（Accumulated View Count，AVC）等特征，并对其进行分桶，以更细致地描述用户所处的冷启动阶段。该特征作为个性化编码输入门控网络，使模型能够根据用户当前的行为积累程度调节不同子模块的贡献。例如，刚进入系统、尚未产生多少有效交互的用户，与已经观看过一定数量视频但仍处于冷启动窗口内的用户，可能需要不同的特征组合。通过分桶后的累计观看次数，门控网络可以学习这些阶段之间的差异，而不必为每一种用户状态单独训练完整模型。

此外，还有物品冷启动场景。对于新物品，仅使用新老物品二值标记通常也不够精细，因为不同新物品的曝光积累速度和用户反馈情况也可能存在差异。因此，物品冷启动与用户冷启动具有相似的问题，但个性化信息来自物品侧。以短视频推荐为例，我们可以结合视频发布时间和累计曝光量来识别新视频。POSO 模型会根据视频发布时间是否处于指定时间窗口内，以及视频累计曝光是否低于设定阈值来定义新视频，并使用视频年龄等特征作为个性化编码。这样，模型可以根据物品所处的生命周期调整不同子模块的贡献，增强对缺乏历史交互数据的新物品的建模能力。

在具体实现方面，POSO 模型的门控网络采用轻量级两层 MLP，并通过 Sigmoid 函数生成各子模块的门控权重。与 Softmax 不同，Sigmoid 不要求不同门控权重之和等于 1，因此多个子模块可以同时获得较高权重，也可以同时被抑制。由于不同子模块的输出可能存在尺度差异，POSO 模型还可以通过校正因子控制融合结果的整体尺度。

总体而言，POSO 模型通过将个性化特征直接引入网络内部的计算过程，缓解了统一模型对主流用户群体过度拟合、而对少数群体建模不足的问题。其设计并不局限于冷启动场景，只要不同用户群体或物品群体存在相对稳定的行为差异，并且可以通过可用特征进行识别，就可以考虑采用类似的个性化模块。例如，对于长期不活跃用户、回流用户或具有特殊消费习惯的用户群体，也可以使用相应的个性化编码控制网络内部的激活方式。

24.2 APG 模型

APG 模型源自阿里巴巴发表于 NeurIPS 2022 会议的论文《APG: Adaptive Parameter Generation Network for Click-Through Rate Prediction》[4]。传统点击率预测模型通常使用一套固定的网络参数处理所有样本，难以充分适应不同用户群体、物品类型及上下文条件下的行为差异。APG（Adaptive Parameter Generation Network）模型的核心思想是根据当前样本的特征动态生成网络参数，使预测函数能够随样本变化，而不再完全依赖一套固定的共享参数。然而，如果直接为每个样本生成完整的权重矩阵，参数生成本身就可能带来较大的计算与存储开销。

因此，APG 模型进一步引入低秩参数化、分解式前向计算、参数共享和过参数化等设计，在增强模型表达能力的同时控制计算成本。图 24.4展示了 APG 模型从基本模型逐步演进到高效结构的过程，其中图 24.4（a）为常规网络模型，图 24.4（b）为原始 APG 模型，图 24.4（c）为低秩参数化，图 24.4（d）为引入共享参数，图 24.4（e）为过参数化。

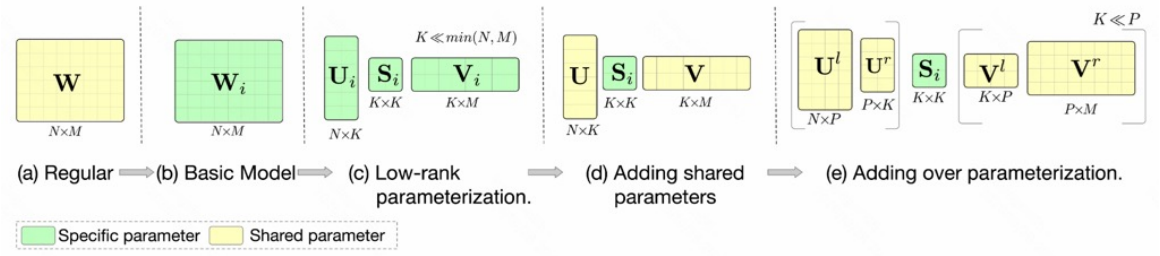


图 24.4: APG 模型不同版本的结构对比。

24.2.1 原始 APG 与条件设计

假设第 i 个样本的输入特征为 $\mathbf{x}_i \in \mathbb{R}^M$，预测网络的权重矩阵为 $\mathbf{W}_i \in \mathbb{R}^{N\times M}$。常规预测网络对所有样本使用相同的权重矩阵 $\mathbf{W}$，而 APG 模型根据样本对应的条件表示 $\mathbf{z}_i \in \mathbb{R}^D$，通过参数生成网络 $\mathcal{G}$ 动态生成权重：

$$\mathbf{W}_i = \mathcal{G}(\mathbf{z}_i) = \text{reshape}\left(\text{MLP}(\mathbf{z}_i)\right) \tag{24.12}$$

其中，$\text{reshape}(\cdot)$ 将 MLP 输出的一维向量转换为对应的权重矩阵。随后预测网络使用生成的参数完成前向计算：

$$\mathbf{y}_i = \sigma(\mathbf{W}_i\mathbf{x}_i) \tag{24.13}$$

其中，$\sigma(\cdot)$ 表示相应的激活函数。上述公式以单层网络为例，实际模型可以将相同的设计扩展到多个网络层。参数生成网络本身仍然可以共享，但其输出的参数随输入条件变化，因此不同样本能够使用不同的预测参数。

由于条件表示 $\mathbf{z}_i$ 决定了参数生成网络依据什么信息调整预测模型，因此原始的 APG 模型提出了三种条件设计方式，即 **Group-wise**、**Mix-wise** 和 **Self-wise**。其中，**Group-wise** 表示根据样本所属的用户群体或其他预先定义的分组构造条件表示。例如，可以使用用户 Embedding 作为条件，为不同用户生成具有个性化特征的网络参数。这种方式适合能够获得明确分组信息、且组间行为模式存在差异的场景。

Mix-wise 表示同时利用多个条件信息生成参数。假设样本包含多个条件表示，可以先将这些表示进行拼接、平均或注意力聚合，再送入参数生成网络；也可以分别根据不同条件生成参数，再对生成结果进行聚合。前者在输入侧融合条件，后者则在参数侧融合信息，从而提高参数生成方式的灵活性。

Self-wise 表示直接使用预测网络自身的输入作为条件，不依赖额外的分组或先验信息。例如，对于第一个隐藏层，可以将原始输入特征作为条件；对于后续隐藏层，则可以使用前一层的输入表示生成当前层参数。这使网络能够根据当前样本的中间表示动态调整后续计算。

上述三种方式分别利用分组信息、多种条件信息和网络自身的输入生成参数，可以根据具体任务和可用特征进行选择。无论采用哪种条件设计，原始的 APG 模型都需要将条件表示映射为完整的权重矩阵。当网络层较宽时，生成和存储这些动态参数的开销可能很大，因此还需要进一步改进参数化方式。

24.2.2 高效 APG 结构与复杂度优化

为了降低动态参数生成的开销，APG 模型对样本相关的权重矩阵进行低秩分解，并进一步将矩阵乘法拆解为连续的向量计算。随后，通过共享部分参数减少需要动态生成的参数规模，最后利用过参数化增强共享部分的表达能力。图 24.4展示了这些设计之间的关系。

第一个高效 APG 模型的设计是**低秩参数化**。对于 $\mathbf{W}_i \in \mathbb{R}^{N\times M}$，APG 假设其具有较低的内在秩，并将其表示为三个较小矩阵的乘积：

$$\mathbf{W}_i = \mathbf{U}_i\mathbf{S}_i\mathbf{V}_i, \tag{24.14}$$

其中，

$$\begin{aligned} &\mathbf{U}_i \in \mathbb{R}^{N\times K}, \\ &\mathbf{S}_i \in \mathbb{R}^{K\times K}, \qquad K \ll \min(N, M) \\ &\mathbf{V}_i \in \mathbb{R}^{K\times M}, \end{aligned} \tag{24.15}$$

此时，参数生成网络只需要根据条件表示生成三个子矩阵，其输出维度由原来的 NM 缩小为 $NK + K^2 + KM$。通过选择较小的秩 K，可以减少参数生成阶段的计算量与存储开销。

不过，仅进行低秩分解还不够。如果先将三个子矩阵相乘，显式重构出完整的 $\mathbf{W}_i$，仍然需要额外的矩阵乘法开销。因此，APG 进一步采用了分解式前向计算的方法。**分解式前向计算**是高效 APG 模型的第二个设计。分解式前向计算利用矩阵乘法的结合律，可以将原始计算改写为：

$$\begin{aligned}\mathbf{y}_i &= \sigma(\mathbf{U}_i\mathbf{S}_i\mathbf{V}_i\mathbf{x}_i) \\ &= \sigma\left(\mathbf{U}_i\left(\mathbf{S}_i\left(\mathbf{V}_i\mathbf{x}_i\right)\right)\right)\end{aligned} \tag{24.16}$$

按照从右向左的顺序，模型会依次计算 $\mathbf{V}_i\mathbf{x}_i$、$\mathbf{S}_i(\mathbf{V}_i\mathbf{x}_i)$ 和最终的 $\mathbf{U}_i$ 变换，而无须显式构造完整的权重矩阵。所以，原先重构矩阵所需的 $O(NK^2 + NMK)$ 复杂度的计算就可以避免，单层前向计算的计算开销就可以降低为 $O(MK + K^2 + NK)$ 的复杂度。因此，低秩分解与分解式前向计算需要配合起来使用，前者缩小参数表示的规模，后者进一步避免完整矩阵重构带来的计算成本。

除此之外，高效 APG 模型中还引入了第三个设计，即**参数共享**。在低秩表示的基础上，APG 模型进一步将三个子矩阵划分为样本相关参数和跨样本共享参数。模型会将规模较小的 $\mathbf{S}_i$ 作为样本相关参数，由参数生成网络根据条件表示动态生成。而 $\mathbf{U}$ 和 $\mathbf{V}$ 则设为所有样本共享的参数。在实际计算中，只有 $\mathbf{S}_i$ 需要为每个样本单独生成，其他矩阵可以在样本之间复用，模型因此可以写成：

$$\mathbf{S}_i = \text{reshape}\left(\text{MLP}(\mathbf{z}_i)\right) \tag{24.17}$$

$$\mathbf{y}_i = \sigma\left(\mathbf{U}\left(\mathbf{S}_i\left(\mathbf{V}\mathbf{x}_i\right)\right)\right) \tag{24.18}$$

这种设计只需要为每个样本生成一个 $K \times K$ 的矩阵，而不再动态生成三个子矩阵。由于 $\mathbf{U}$ 和 $\mathbf{V}$ 在样本之间共享，其计算和存储可以在模型中复用，从而进一步降低动态参数生成的成本。此时，单层的主要计算复杂度为 $O(K^2D + MK + K^2 + NK)$，内存复杂度为 $O(K^2D + NK + MK)$。

最后，高效 APG 模型还引入了**过参数化**的方法。参数共享在降低开销的同时，也限制了共享矩阵 $\mathbf{U}$ 和 $\mathbf{V}$ 的规模。所以为了兼顾计算效率与表达能力，APG 模型进一步采用过参数化设计，将两个共享矩阵写成更大矩阵的乘积：

$$\begin{aligned}\mathbf{U} &= \mathbf{U}_l\mathbf{U}_r \\ \mathbf{V} &= \mathbf{V}_l\mathbf{V}_r\end{aligned} \tag{24.19}$$

其中，

$$\begin{aligned}&\mathbf{U}_l \in \mathbb{R}^{N\times P}, \quad \mathbf{U}_r \in \mathbb{R}^{P\times K} \\ &\mathbf{V}_l \in \mathbb{R}^{K\times P}, \quad \mathbf{V}_r \in \mathbb{R}^{P\times M}\end{aligned} \qquad P \gg K. \tag{24.20}$$

通过引入中间维度为 P 的矩阵，模型能够在训练过程中学习更多共享参数，从而提升表示能力。同时，矩阵乘积的参数化形式也可能带来一定的隐式正则化效果。需要注意的是，过参数化主要增加了训练阶段的参数表达空间，但并不意味着在线推理时也必须执行额外的矩阵乘法。在训练完成后，可以预先计算并保存 $\mathbf{U}$ 与 $\mathbf{V}$，在线推理仍然使用公式 (24.18)中的共享矩阵。因此，相比起参数共享后的 APG 模型，过参数化不需要引入额外的在线推理计算和存储开销。

上述设计的整体框架如图 24.5所示。图中展示了条件表示、样本相关参数与共享参数之间的关系；为简洁起见，图中省略了公式 (24.16)所示的分解式前向计算过程。为了更直观地比较不同版本的效率，表 24.1汇总了单层网络的计算复杂度和内存开销。这里假设 N 和 M 分别为权重矩阵的输出维度和输入维度，D 为条件表示的维度，K 为低秩维度。表中的复杂度主要反映动态参数生成及单层前向计算的开销，具体实现中的常数项未予展开。

从表 24.1可以看出，APG 模型的效率优化并非单纯来自低秩分解。低秩参数化减小了需要生成的参数规模，分解式前向计算避免了完整权重矩阵的重构，参数共享则进一步减少了样本相关参数的数量。最终，单层的计算复杂度由常规网络的 $O(NM)$ 变成 $O(K^2D + NK + MK + K^2)$，内存复杂度变成 $O(K^2D + NK + MK)$。当 K 远小于 N 和 M，且条件维度 D 适中时，这种结构可以显著降低动态参数生成的成本。

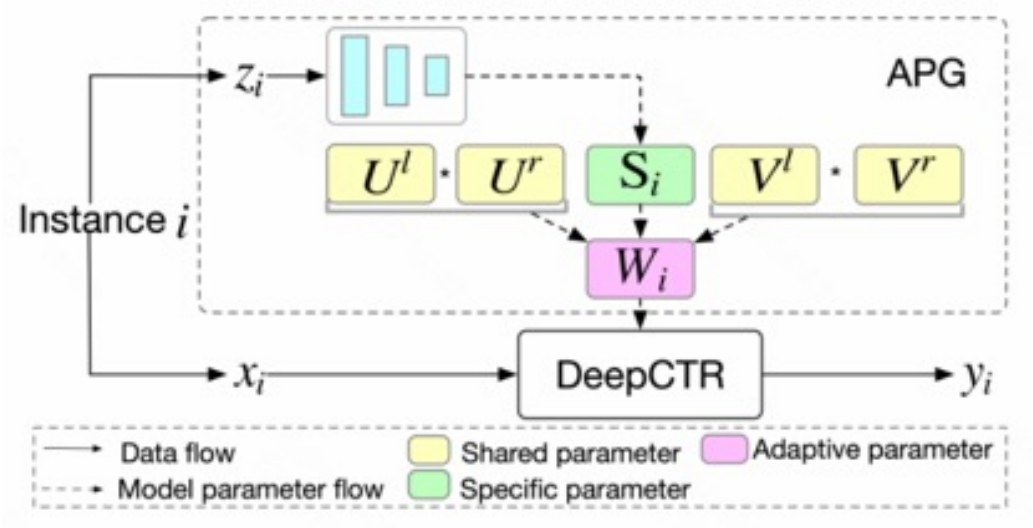


图 24.5: APG 模型的整体框架。

表 24.1: APG 不同版本的单层计算与内存复杂度

模型版本	计算复杂度	内存复杂度
常规深度 CTR 模型	$O(NM)$	$O(NM)$
直接生成完整权重矩阵	$O(NMD+NM)$	$O(NMD)$
低秩分解后重构权重矩阵	$O((NK+MK+K^2)D+NK^2+NMK+NM)$	$O((NK+MK+K^2)D)$
低秩分解与分解式前向计算	$O((NK+MK+K^2)(D+1))$	$O((NK+MK+K^2)D)$
引入参数共享后的 APG	$O(K^2D+NK+MK+K^2)$	$O(K^2D+NK+MK)$

综上所述，APG 模型通过动态生成与样本相关的参数，使深度 CTR 模型能够根据不同样本的行为模式进行自适应调整。同时，模型采用低秩参数化、分解式前向计算和参数共享等策略控制计算与存储开销，并通过过参数化增强共享参数的表达能力。其核心思想是将样本相关的个性化能力集中于规模较小的参数矩阵，同时复用跨样本共享的参数矩阵，从而在提升模型个性化建模能力的同时兼顾计算效率。

24.3 PEPNet 模型

PEPNet（Parameter and Embedding Personalized Network）模型源自快手发表于 KDD 2023 的论文《PEPNet: Parameter and Embedding Personalized Network for Infusing with Personalized Prior Information》[1]。该方法面向工业推荐系统中的多场景、多任务建模问题，通过个性化先验信息动态调节底层 Embedding 表示和顶层 DNN 隐藏单元，在保留共享模型主体的同时增强模型对不同用户、物品和场景的适应能力。

在工业级推荐系统中，同一用户可能在不同场景下表现出不同的行为偏好，而点击、观看时长、点赞等预测任务也具有不同的标签分布和样本稀疏程度。传统的多场景、多任务模型通常采用共享 Embedding 和共享 DNN 参数来学习不同场景与任务之间的共性，但这种参数共享也可能削弱模型对个体差异的刻画能力。一方面，不同场景中的特征重要性存在差异，共享 Embedding 难以始终为每个场景提供最合适的特征表示；另一方面，不同用户对各类行为的偏好不同，统一的任务塔参数也难以充分适应不同用户的需求。

这些问题可能导致模型出现不理想的**跷跷板现象（Seesaw Phenomenon）**，即模型在部分任务或场景上的效果提升时，其他任务或场景的效果却可能下降。尤其是在多任务推荐中，不同用户的行为反馈具有不同的稀疏程度，如果所有用户始终使用相同的 DNN 参数，模型就难以同时兼顾不同用户的多样化偏好。

因此，PEPNet 模型提出了一种可插拔的个性化网络结构，将个性化先验信息直接注入模型的两个关键位置。其中，Embedding Personalized Network（EPNet）模块根据场景相关的个性化信息调节底层 Embedding，使模型能够针对不同场景和用户状态选择更合适的特征表示。而 Parameter Personalized Network（PPNet）模块则根据用户、物品和作者等个性化信息调节顶层 DNN 的隐藏单元，使不同用户在不同任务中能够采用不同的参数激活方

式。两个模块均建立在门控神经单元（Gate Neural Unit，Gate NU）之上，通过轻量级门控机制来实现个性化。

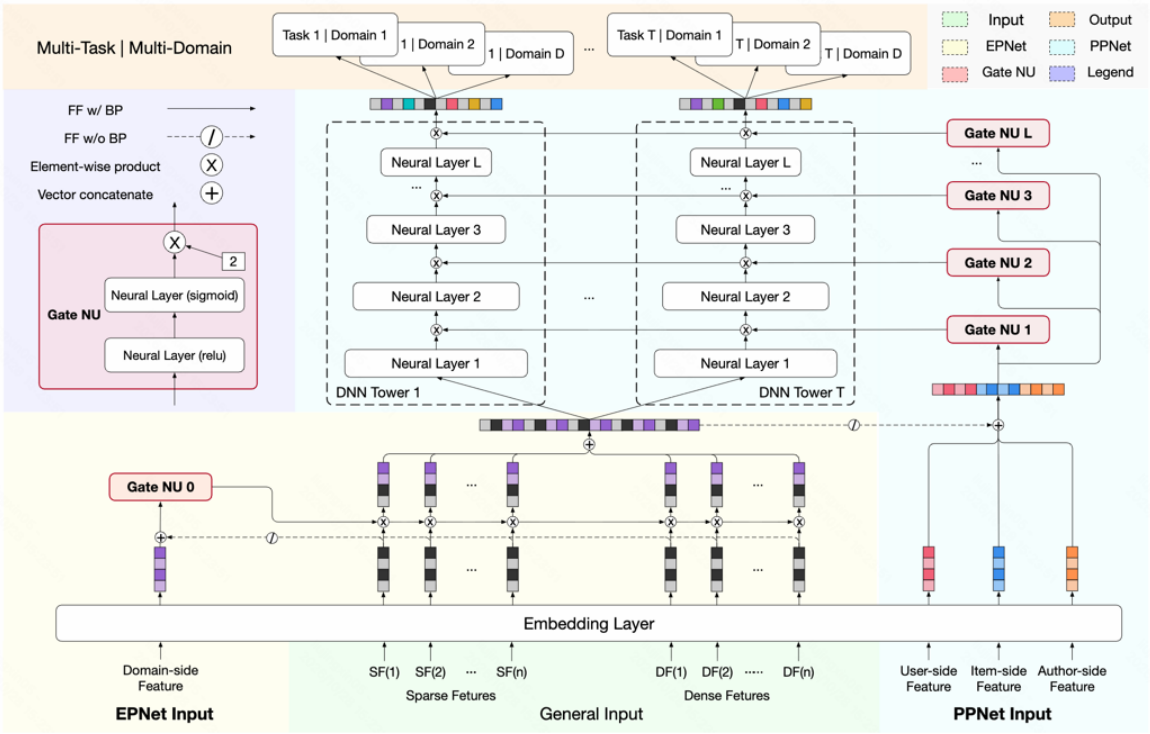


图 24.6: PEPNet 模型的整体架构。

如图24.6所示，PEPNet 模型并不是用一套独立模型替代原有的多场景、多任务推荐模型，而是在共享 Embedding 和任务塔的基础上增加个性化门控。EPNet 模块主要解决不同场景下特征重要性不同的问题，PPNet 模块则进一步解决不同用户在多个预测任务上的偏好差异问题。下面分别介绍其基础门控单元、Embedding 个性化机制和参数个性化机制。

24.3.1 门控神经单元

门控神经单元（Gate Neural Unit，Gate NU）是 EPNet 和 PPNet 模块的基础组件，其设计受到语音识别领域 LHUC（Learning Hidden Unit Contributions）方法的启发。LHUC 模型通过学习与说话人相关的隐藏单元贡献系数，对神经网络中的隐藏表示进行缩放，从而提高模型对不同说话人的适应能力。由于 LHUC 模型在本书第15章《特征工程》中进行过详细介绍，这里不再赘述。PEPNet 模型将 LHUC 的这一思想扩展到推荐系统，不再局限于用户 ID，而是综合利用用户属性、物品属性、作者信息和场景统计等多种个性化先验信息。

假设 Gate NU 的输入为 $\mathbf{x}$，其首先通过一个带有 ReLU 激活函数的全连接层，对不同先验特征进行交互：

$$\mathbf{x}' = \mathrm{ReLU}(\mathbf{x}\mathbf{W} + \mathbf{b}) \tag{24.21}$$

其中，$\mathbf{W}$ 和 $\mathbf{b}$ 分别为可训练权重矩阵和偏置项。随后，第二个全连接层通过 Sigmoid 函数生成门控向量：

$$\boldsymbol{\delta} = \gamma \cdot \mathrm{Sigmoid}(\mathbf{x}'\mathbf{W}' + \mathbf{b}'), \qquad \boldsymbol{\delta} \in [0, \gamma]^d \tag{24.22}$$

其中，$\mathbf{W}'$ 和 $\mathbf{b}'$ 为第二层的可训练参数，d 为门控向量的维度，γ 为缩放系数，原论文中设置为 2。

由公式24.21和公式24.22可知，Gate NU 根据输入的个性化先验生成与目标表示维度对应的门控权重。与 Softmax 门控不同，这里的各维权重通过 Sigmoid 独立生成，并不要求总和为 1。当某个维度的门控权重较大时，对应的隐藏单元或 Embedding 维度会得到更强的激活。当权重较小时，其贡献则会相应减弱。由于门控值的范围为 $[0, 2]$，该机制既能够抑制部分信号，也能够放大与当前用户或场景相关的信号。

需要注意的是，Gate NU 本身并不直接完成推荐预测，而是根据个性化先验生成调制系数。它的具体作用取决于所处的位置。在 EPNet 中，门控向量用于调整底层 Embedding；而在 PPNet 中，门控向量用于调整不同任

务塔中的隐藏单元。通过复用相同的基本设计，PEPNet 模型可以将个性化 Gate 机制统一为一种简单、灵活的网络组件。

24.3.2 Embedding 个性化网络（EPNet）

工业级推荐系统通常需要处理大量稀疏特征和稠密特征，其中用户 ID、物品 ID、作者 ID 等特征对应的 Embedding 表往往占据大量存储空间。为了控制模型规模，多场景推荐系统通常采用共享 Embedding 结构，让不同场景的样本共用底层特征表示。对于稀疏特征集合 $\mathcal{F}^S$ 和稠密特征集合 $\mathcal{F}^D$，其 Embedding 表示可以写为

$$\mathbf{E} = E(\mathcal{F}^S) \oplus E(\mathcal{F}^D) \tag{24.23}$$

其中，$E(\cdot)$ 表示特征映射与 Embedding 计算，$\oplus$ 表示向量拼接。

共享 Embedding 能够利用不同场景之间的共性，并减少重复参数，但也存在一定局限。例如，同一用户在不同场景下可能具有不同的行为目的，同一特征在不同场景中的预测价值也可能存在差异。如果所有场景始终使用完全相同的 Embedding 表示，模型就难以针对当前场景调整各类特征的重要性。

EPNet 模块在保留共享 Embedding 表的基础上，引入场景侧个性化先验信息，对生成的 Embedding 表示进行动态调整。其输入包括场景 ID，以及用户在不同场景中的行为统计、物品在不同场景中的曝光统计等特征。假设场景侧特征经过 Embedding 映射后得到 $\mathbf{E}_d$，EPNet 模块的门控计算可以表示为

$$\boldsymbol{\delta}_{\text{domain}} = \text{GateNU}_{\text{EP}}\left(\mathbf{E}_d \oplus \text{sg}(\mathbf{E})\right) \tag{24.24}$$

其中，$\text{GateNU}_{\text{EP}}$ 表示 EPNet 中的门控网络，$\text{sg}(\cdot)$ 表示停止梯度传播（stop-gradient），即前向计算保留输入值，但不将该分支的梯度传回原始 Embedding。

随后，EPNet 模块利用门控向量对共享 Embedding 进行逐元素调整：

$$\mathbf{O}_{\text{EP}} = \boldsymbol{\delta}_{\text{domain}} \odot \mathbf{E} \tag{24.25}$$

其中，$\odot$ 表示逐元素乘法，$\mathbf{O}_{\text{EP}}$ 为 EPNet 模块输出的个性化 Embedding 表示。实际实现中，也可以根据 Embedding 的规模和计算开销，在向量级或特征级进行调制，而不必始终对所有维度执行逐元素乘法。

这里需要区分两条计算路径，其中 Stop Gradient 操作仅作用于门控网络的输入分支，而公式24.25中的共享 Embedding 仍参与正常的前向计算和梯度更新。因此，EPNet 模块并不是冻结原有 Embedding，而是在不让门控分支反向改变其输入表示的前提下，学习如何根据场景先验生成调制系数。

通过这种方式，同一套共享 Embedding 可以根据不同场景获得不同的有效表示。例如，某个用户行为特征可能在视频消费场景中具有较强的预测价值，而在其他场景中作用有限，EPNet 模块可以通过不同的门控权重调节该特征的贡献。与为每个场景维护独立 Embedding 表相比，这种设计能够在保留参数共享优势的同时增强场景适应能力，也避免了直接复制大规模 Embedding 表所带来的额外存储成本。

24.3.3 参数个性化网络（PPNet）

EPNet 模块主要处理底层特征表示的个性化问题，但不同用户在各类行为上的偏好差异，还需要通过顶层预测网络进行建模。传统多任务推荐模型通常为每个任务配置独立的 DNN Tower，但同一任务塔中的参数仍然由所有用户共享。当不同用户对点击、观看时长、点赞等行为表现出不同的偏好时，统一的任务塔参数可能难以兼顾各类用户，进而导致部分任务或用户群体的预测效果受到影响。

PPNet 模块在 DNN Tower 中进一步引入个性化门控，根据用户、物品和作者等信息动态调整隐藏单元的贡献。与 EPNet 模块主要使用场景侧信息不同，PPNet 模块更侧重于描述具体用户与当前物品之间的个性化关系。其先验特征可以包括用户 ID、年龄、性别等用户侧信息，物品 ID、类别、热度等物品侧信息，以及短视频作者 ID 等作者侧信息。

首先，PPNet 模块将这些先验特征映射为 Embedding 并拼接，得到

$$\mathbf{O}_{\text{prior}} = E(\mathcal{F}_u) \oplus E(\mathcal{F}_i) \oplus E(\mathcal{F}_a) \tag{24.26}$$

其中，$\mathcal{F}_u$、$\mathcal{F}_i$ 和 $\mathcal{F}_a$ 分别表示用户、物品和作者侧特征集合。假设 $\mathbf{H} = [\mathbf{H}_1, \ldots, \mathbf{H}_T]$ 表示 T 个任务塔中对应层的隐藏表示，其中 $\mathbf{H}_t \in \mathbb{R}^h$，$h$ 为每个任务塔的隐藏层维度。PPNet 模块将个性化先验与 EPNet 模块输出的 Embedding 表示拼接，并通过 Gate NU 生成任务相关的门控向量：

$$\boldsymbol{\delta}_{\text{task}} = \text{GateNU}_{\text{PP}}\left(\mathbf{O}_{\text{prior}} \oplus \text{sg}(\mathbf{O}_{\text{EP}})\right) \tag{24.27}$$

其中，$\text{GateNU}_{\text{PP}}$ 表示 PPNet 模块中的门控网络。门控向量的维度为 Th，可以拆分为 T 个维度为 h 的子向量，分别用于调整不同任务塔的隐藏表示。

将门控向量拆分为 $\boldsymbol{\delta}_{\text{task}} = [\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_T]$，则第 t 个任务的个性化隐藏表示为

$$\mathbf{O}_{\text{PP},t} = \boldsymbol{\delta}_t \odot \mathbf{H}_t \tag{24.28}$$

对于包含多层 DNN 的任务塔，PPNet 模块可以在各层重复执行上述过程。假设第 l 层的隐藏表示为 $\mathbf{H}_t^{(l)}$，相应的个性化门控为 $\boldsymbol{\delta}_t^{(l)}$，则有

$$\mathbf{O}_{\text{PP},t}^{(l)} = \boldsymbol{\delta}_t^{(l)} \odot \mathbf{H}_t^{(l)} \tag{24.29}$$

$$\mathbf{H}_t^{(l+1)} = f\left(\mathbf{O}_{\text{PP},t}^{(l)} \mathbf{W}_t^{(l)} + \mathbf{b}_t^{(l)}\right) \tag{24.30}$$

其中，$\mathbf{W}_t^{(l)}$ 和 $\mathbf{b}_t^{(l)}$ 为对应任务塔的网络参数，f 为激活函数。隐藏层通常使用 ReLU，最终预测层使用 Sigmoid 输出二分类任务的预测概率。PPNet 模块的个性化门控主要作用于隐藏层，最终预测层不使用 Gate NU 中的放大系数 γ。

与直接为每个用户生成完整的 DNN 参数矩阵不同，PPNet 模块保留原有任务塔的权重矩阵，仅通过个性化门控调节隐藏单元的有效贡献。因此，不同用户可以在同一组共享任务参数下形成不同的计算路径，而不必为每个用户单独存储一套模型。例如，对于某个用户，观看时长相关的隐藏单元可能获得较高的门控权重，而点赞相关的隐藏单元可能获得较低的权重。对于另一用户，门控结果则可能相反。这里的个性化并不是直接人为指定各任务的重要性，而是由模型根据用户、物品和作者等先验信息学习得到。随着训练的进行，PPNet 模块可以逐步形成与不同用户偏好相适应的隐藏单元激活模式，从而缓解统一任务塔难以兼顾不同用户群体的问题。

24.3.4 工业部署与工程优化

PEPNet 模型的设计不仅考虑模型表达能力，也面向大规模推荐系统的训练与在线服务约束。模型在快手的工业级推荐系统中进行了部署，并引入了多项工程优化，包括 Embedding 存储管理、在线参数同步，以及 Embedding 与 DNN 参数的差异化训练策略。这些优化对于具有海量 ID 特征和频繁参数更新的短视频推荐场景尤为重要。

首先，Embedding 表的规模往往远大于 DNN 参数，尤其是用户 ID、物品 ID 和作者 ID 等高基数稀疏特征，会持续产生新的 Embedding，并带来较大的内存压力。因此，PEPNet 模型采用全局共享 Embedding 表管理机制（Global Shared Embedding Table，GSET），根据特征评分和资源限制管理 Embedding 的保留与淘汰，控制整体内存使用。与仅根据近期访问频率进行淘汰的传统缓存策略不同，这类机制还需要考虑低频特征反复进入和退出系统所带来的训练与服务稳定性问题。

其次，Embedding 和 DNN 参数的更新频率存在差异。用户和物品 ID 的 Embedding 会随着新样本持续更新，而 DNN 参数通常以相对稳定的节奏进行训练和同步。如果在每次训练更新后都完整同步所有 Embedding，将产生较高的存储、计算和通信开销。因此，PEPNet 模型采用按特征设置同步数量上限和更新有效期等策略，优先同步满足更新条件的 Embedding，减少不必要的参数传输。

最后，论文采用 Embedding 与 DNN 参数分开训练和更新的策略，以适应二者不同的变化速度。Embedding 使用 AdaGrad 优化器，学习率设置为 0.05；而 DNN 参数则使用 Adam 优化器，学习率设置为 5×10^{-6}。这种差异化配置旨在更及时地捕捉底层稀疏特征的变化，同时保持顶层网络参数更新的稳定性。

总体来看，PEPNet 模型将个性化建模从单纯的输入特征表达扩展到 Embedding 和 DNN 隐藏层两个关键环节。EPNet 模块根据场景侧先验调整底层特征表示，PPNet 模块根据用户、物品和作者等先验调整任务塔的隐藏单元，而 Gate NU 模块为两者提供统一的门控机制。通过这种分层个性化设计，PEPNet 模型在保留共享参数优势的同时，增强了模型对不同场景、用户及预测任务的适应能力。

24.4 APPNet 模型

APPNet 模型源自于阿里巴巴发表在 WSDM 2026 会议的论文《APPNet: Automatic Feature Partitioning-Based Parameter Personalized Network for Conversion Prediction in E-commerce》[3]。该模型面向电商推荐中的转化率预测问题，研究如何利用参数个性化机制刻画不同用户、物品及上下文条件下的行为差异。

传统 CVR（Conversion Rate，转化率）预测模型通常采用共享的 Embedding 层和 DNN 网络学习特征表示与预测函数。这种方式虽然能够充分利用整体训练数据，但也意味着不同样本需要共用同一套网络参数。当用户群体、物品类型或上下文条件之间存在显著差异时，共享网络可能难以同时适应不同的数据分布。参数个性化方法通过根据当前样本动态调整网络中的参数或中间表示，使得模型能够针对不同样本学习不同的预测模式。然而，个性化所依赖的特征如何选择并不简单。如果完全依靠人工指定先验特征，模型可能遗漏真正有助于区分行为模式的信息。如果将所有特征同时用于参数生成和主预测网络，则可能使两个分支对共享特征产生相互冲突的梯度。

为了解决这些问题，APPNet 模型提出自动特征划分模块（Automatic Feature Partitioning，AFP），并将其与参数个性化网络（Parameter Personalized Network，PPNet）相结合。AFP 模块根据训练数据自动学习特征的分配方式，控制不同特征对参数个性化和主预测任务的反向优化贡献，从而减少梯度冲突。PPNet 模块则利用个性化分支生成门控向量，对预测网络各层的隐向量进行逐元素调节。通过两者的配合，可以使 APPNet 模型能够在保留共享网络学习能力的同时，增强对不同样本行为差异的刻画能力。

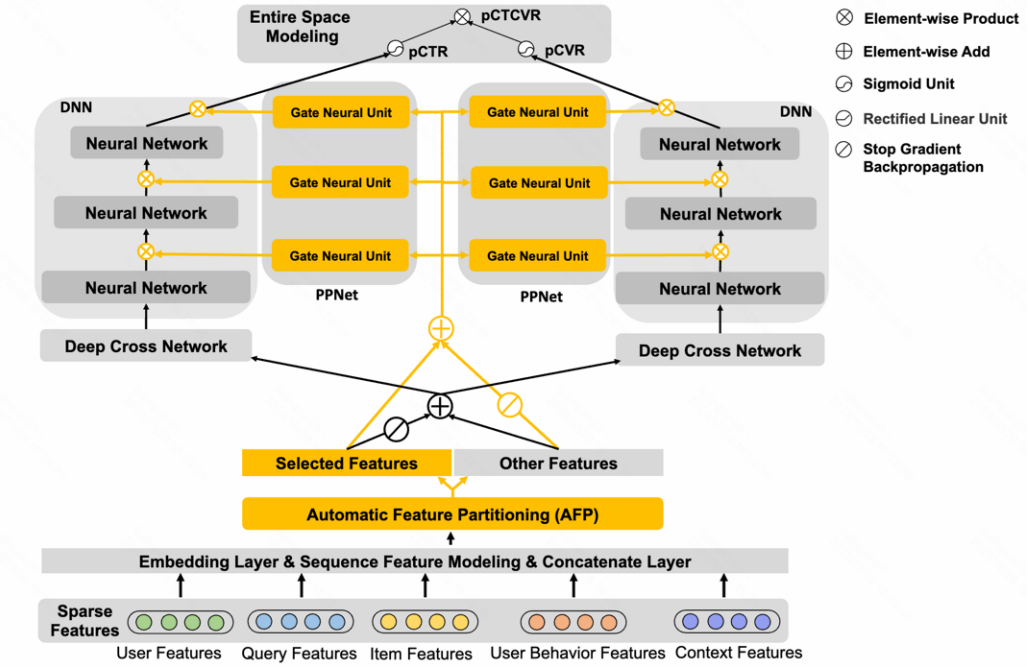


图 24.7: APPNet 模型的整体结构。

APPNet 模型的整体网络结构如图 24.7所示。模型由 Embedding 层、序列特征建模模块、特征拼接层、AFP 模块、PPNet 模块、Deep Cross Network v2（DCNv2）、多层 DNN 以及全空间多任务学习模块（Entire Space Multi-Task Model，ESMM）组成。其中，AFP 模块和 PPNet 模块是模型的核心组件，分别负责自动学习特征分配方式和实现参数个性化。

在前向计算过程中，离散特征首先经过 Embedding 层转换为稠密向量，序列特征经过相应的序列建模模块得到固定维度的表示。随后，模型将不同来源的特征表示进行拼接，形成统一的输入。AFP 模块在此基础上学习不同特征的划分权重，并通过特定的梯度处理方式区分参数个性化分支和主预测分支的优化路径。PPNet 模块根据个性化分支的特征表示生成门控信号，主预测分支则结合 DCNv2 和 DNN 网络完成特征交互与转化率预测。最后，模型通过 ESMM 模块组织点击率和点击后转化率之间的关系，构造联合的训练目标。

为了进一步理解 APPNet 模型的设计思路，可以将其与 APG 和 PEPNet 模型进行对比，如图 24.8所示。其中，APG（Adaptive Parameter Generation Network）模型根据当前样本动态生成预测网络参数，从而使模型能够适应不同样本的特征分布 [4]。PEPNet（Parameter and Embedding Personalized Network）模型利用个性化先验信息，通过门控机制调节 Embedding 表示和 DNN 隐层，使共享网络能够根据不同用户、场景或任务进行自适应调整 [1]。这两类方法分别从动态参数生成和先验信息驱动的参数、表示调节两个角度实现个性化。

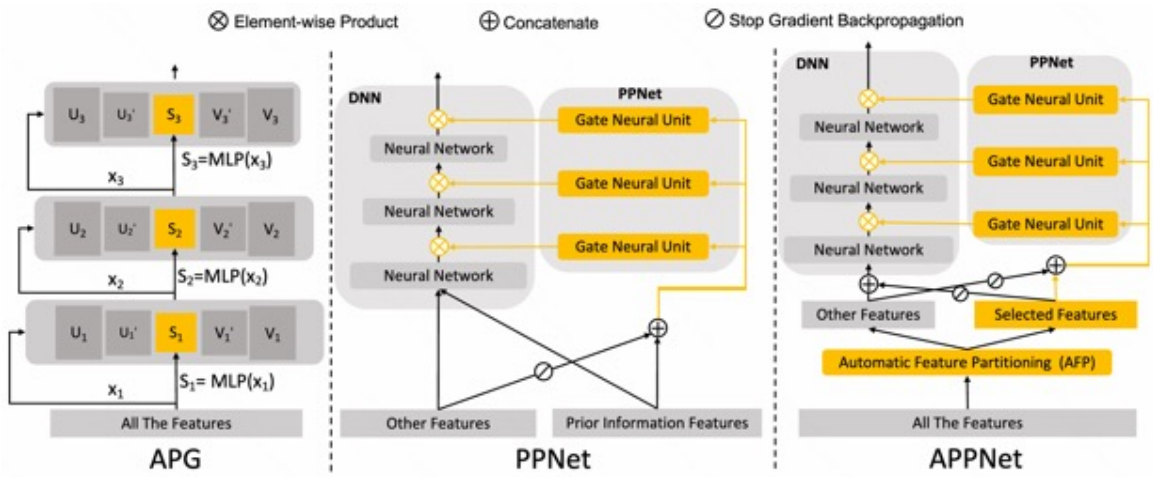


图 24.8: APPNet 与 APG、PEPNet 的结构对比。

APPNet 模型的主要改进并不是完全重新设计参数个性化网络，而是进一步研究哪些特征更适合驱动参数个性化，以及如何减少个性化分支与主预测分支之间的梯度冲突。与依赖人工指定先验特征的方式相比，AFP 模块能够在训练过程中自动学习特征分配。与直接让所有特征同时参与参数生成和主预测网络的方式相比，AFP 模块还通过停止梯度操作区分不同分支对特征表示的反向更新。基于这一设计，APPNet 模型能够在共享特征表示的基础上形成更适合不同任务目标的优化路径。下面分别介绍 AFP 模块、PPNet 模块及模型的多任务学习目标。

24.4.1 自动特征划分模块（AFP）

AFP 模块的核心任务是学习特征在参数个性化分支和主预测分支中的分配方式。前者利用特征生成门控信号，决定预测网络应当如何针对当前样本进行调整；后者直接学习点击率或转化率预测函数。由于哪些特征更适合用于个性化建模通常难以通过人工规则准确判断，AFP 模块为输入特征引入可学习的权重，并在训练过程中持续调整特征的分配方式。

假设模型包含 N 个特征字段，第 i 个特征经过 Embedding 层后得到向量 $\mathbf{e}_i \in \mathbb{R}^d$，其中 d 为 Embedding 维度。将全部特征向量拼接，得到整体特征表示：

$$\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N] \in \mathbb{R}^{1 \times Nd} \tag{24.31}$$

记经过层归一化后的特征表示为：

$$\mathbf{X} = \mathrm{LN}(\mathbf{E}) \tag{24.32}$$

AFP 模块提供了两种实现方式：Feature-wise AFP 和 Bit-wise AFP。前者以完整特征字段为单位学习分配权重，后者则将划分粒度进一步细化到 Embedding 中的每个维度。两种方法的基本计算流程相似，主要区别在于可学习权重和二值掩码的维度。

24.4.1.1 Feature-wise AFP 模块

Feature-wise AFP 模块以特征字段为单位学习划分权重。首先，为每个特征字段设置一个可学习的权重，构成权重矩阵 $\mathbf{Z} \in \mathbb{R}^{1 \times N}$。权重采用 Xavier 正态分布初始化，并通过 Sigmoid 函数映射到 $[0, 1]$ 区间：

$$\mathbf{Z}' = \sigma(\mathbf{Z}), \qquad \mathbf{Z}' \in \mathbb{R}^{1 \times N} \tag{24.33}$$

随后，给定阈值超参数 λ，将连续权重转换为二值掩码：

$$\mathbf{M} = \mathbb{I}(\mathbf{Z}' \geq \lambda), \qquad \mathbf{M} \in \{0,1\}^{1\times N} \tag{24.34}$$

其中，$\mathbb{I}(\cdot)$ 为指示函数。当某个特征的权重大于或等于阈值时，对应掩码为 1；否则为 0。掩码用于决定不同特征在两条分支中的梯度归属。掩码为 1 的特征主要通过个性化分支获得梯度，而掩码为 0 的特征则主要通过主预测分支获得梯度。

阈值 λ 用来控制两类特征的相对分配比例。当 λ 增大时，通常会有更少的特征满足阈值条件，分配给个性化分支的特征随之减少，参数个性化能力可能减弱，而主预测分支保留的特征优化范围相应扩大。当 λ 减小时，则有更多特征参与个性化分支的优化。因此，阈值需要结合模型效果进行选择，从而平衡个性化建模能力与主预测能力。

但是，直接使用二值掩码进行特征划分存在一个问题，即指示函数不可导。如果仅根据二值掩码进行更新，权重参数将无法获得充分的梯度信号以跨越划分阈值，模型也就难以在训练过程中持续探索新的特征分配方式。因此，APPNet 模型采用直通估计器（Straight-Through Estimator，STE），在前向计算时使用二值掩码，而在反向传播时则保留连续权重的梯度路径。

具体来说，模型先将二值掩码与连续权重组合，再将其扩展到与拼接后的 Embedding 表示相同的维度：

$$\mathbf{Z}'' = \mathrm{Tile}\left(\mathrm{sg}(\mathbf{M} - \mathbf{Z}') + \mathbf{Z}'\right) \tag{24.35}$$

其中，$\mathrm{Tile}(\cdot)$ 表示沿 Embedding 维度扩展特征权重，$\mathrm{sg}(\cdot)$ 表示 Stop Gradient 操作。由于 Stop Gradient 操作只阻断反向梯度而不改变前向数值，上式在前向计算中等价于扩展后的二值掩码。在反向传播中，梯度则通过连续权重 $\mathbf{Z}'$ 传递，从而使特征权重能够持续更新。

利用上述掩码，AFP 模块构造出的个性化分支的输入 $\mathbf{S}$ 和主预测分支的输入 $\mathbf{O}$ 如下所示：

$$\begin{aligned} \mathbf{S} &= \mathbf{X} \odot \mathbf{Z}'' + \mathrm{sg}\left(\mathbf{X} \odot (1 - \mathbf{Z}'')\right) \\ \mathbf{O} &= \mathrm{sg}\left(\mathbf{X} \odot \mathbf{Z}''\right) + \mathbf{X} \odot (1 - \mathbf{Z}'') \end{aligned} \tag{24.36}$$

其中，$\odot$ 表示逐元素乘法，1 表示与掩码同形状的全 1 向量。在理解这组公式时，需要区分前向数值与反向梯度。其中，Stop Gradient 操作并不会将对应项的前向数值清零。因此，两路表示在前向计算中仍保留完整特征信息。这里二值掩码的关键作用是控制梯度流向，使得两条分支对共享特征表示的更新具有不同的归属。

因此，个性化分支通过 $\mathbf{X} \odot \mathbf{Z}''$ 获得掩码选中部分的梯度，主预测分支则通过 $\mathbf{X} \odot (1 - \mathbf{Z}'')$ 获得互补部分的梯度。两个分支虽然都可以利用完整的前向特征，但同一特征维度不会同时接收来自两条分支的梯度更新。该机制能够缓解参数个性化目标与主预测目标之间的梯度冲突，同时保留特征权重通过 STE 进行迭代优化的能力。

24.4.1.2 Bit-wise AFP 模块

Feature-wise AFP 模块以整个特征字段为单位划分特征，同一字段中的所有 Embedding 维度共享一个掩码。当同一特征向量中的不同维度分别适用于参数个性化和主预测任务时，字段级划分无法表达这种差异。因此，APPNet 模型进一步提出了 Bit-wise AFP 模块，将划分粒度从特征字段细化到 Embedding 维度。

假设可学习权重矩阵为 $\mathbf{Z}_{\mathrm{bit}} \in \mathbb{R}^{1\times Nd}$。与 Feature-wise AFP 模块不同的是，Bit-wise AFP 模块为拼接后 Embedding 表示中的每个维度独立学习划分权重。通过 Sigmoid 函数得到连续权重，并根据阈值生成二值掩码：

$$\begin{aligned} \mathbf{Z}'_{\mathrm{bit}} &= \sigma(\mathbf{Z}_{\mathrm{bit}}) \\ \mathbf{M}_{\mathrm{bit}} &= \mathbb{I}(\mathbf{Z}'_{\mathrm{bit}} \geq \lambda) \end{aligned} \tag{24.37}$$

由于权重矩阵与拼接特征表示具有相同的维度，因此不需要再进行字段级扩展。此外，Bit-wise AFP 模块也通过 STE 保留可学习权重的梯度路径：

$$\mathbf{Z}''_{\mathrm{bit}} = \mathrm{sg}\left(\mathbf{M}_{\mathrm{bit}} - \mathbf{Z}'_{\mathrm{bit}}\right) + \mathbf{Z}'_{\mathrm{bit}} \tag{24.38}$$

随后，利用逐维掩码分别构造个性化分支与主预测分支的表示：

$$\begin{aligned} \mathbf{S}_{\mathrm{bit}} &= \mathbf{X} \odot \mathbf{Z}''_{\mathrm{bit}} + \mathrm{sg}\left(\mathbf{X} \odot (1 - \mathbf{Z}''_{\mathrm{bit}})\right) \\ \mathbf{O}_{\mathrm{bit}} &= \mathrm{sg}\left(\mathbf{X} \odot \mathbf{Z}''_{\mathrm{bit}}\right) + \mathbf{X} \odot (1 - \mathbf{Z}''_{\mathrm{bit}}) \end{aligned} \tag{24.39}$$

Bit-wise AFP 模块与 Feature-wise AFP 模块都遵循相同的梯度划分原则，但前者能够为每个 Embedding 维度单独确定梯度归属。因此，同一特征中的不同维度可以分别参与个性化分支和主预测分支的优化，从而提供更细粒度的特征分配能力。相应地，Bit-wise AFP 模块需要为更多维度维护可学习权重，参数规模也会随之增大。

24.4.2 参数个性化网络（PPNet）

在完成了自动特征划分之后，APPNet 模型使用 PPNet 模块根据个性化分支的特征表示生成门控向量，并利用这些门控向量调节主预测网络的隐层表示。需要注意的是，这里的 PPNet 模块依然沿用了 PEPNet 中的门控神经单元（Gate Neural Unit，GNU）设计 [1]。其核心思路是根据当前样本的特征动态生成门控信号，对共享预测网络的中间表示进行逐元素缩放，而不是为每个样本单独维护一套完整的网络参数。

PPNet 模块由三个门控神经单元组成，分别对应 DNN 预测网络中的三个计算层。对于第 l 层，$l \in \{1,2,3\}$，门控网络首先通过一个带 ReLU 激活函数的全连接层提取个性化分支的特征：

$$\mathbf{S}^{(l)} = \mathrm{ReLU}\left(\mathbf{S}\mathbf{W}_g^{(l)} + \mathbf{b}_g^{(l)}\right) \tag{24.40}$$

随后，通过第二个全连接层和 Sigmoid 函数生成门控向量：

$$\boldsymbol{\delta}^{(l)} = \gamma \cdot \sigma\left(\mathbf{S}^{(l)}\mathbf{W}_g'^{(l)} + \mathbf{b}_g'^{(l)}\right), \qquad l \in \{1,2,3\} \tag{24.41}$$

其中，$\mathbf{W}_g^{(l)}$、$\mathbf{b}_g^{(l)}$、$\mathbf{W}_g'^{(l)}$ 和 $\mathbf{b}_g'^{(l)}$ 均为可学习参数，γ 为缩放系数，论文中设置为 2。由于 Sigmoid 函数的输出位于 $[0,1]$ 区间，门控向量的取值位于 $[0,2]$ 区间。因此，门控机制不仅能够抑制部分隐层维度，也能够增强对当前样本更有用的维度。

在主预测分支中，输入特征 $\mathbf{O}$ 首先经过 DCNv2 网络，以建模不同输入特征之间的交互关系：

$$\mathbf{O}' = \mathrm{DCNv2}(\mathbf{O}) \tag{24.42}$$

随后，DNN 逐层计算隐向量，并利用 PPNet 生成的门控向量对中间表示进行调节。对于前两层，其计算过程为：

$$\begin{aligned}
\mathbf{H}^{(1)} &= \mathrm{ReLU}\left(\mathrm{NN}^{(1)}(\mathbf{O}')\right) \\
\widetilde{\mathbf{H}}^{(1)} &= \mathbf{H}^{(1)} \odot \boldsymbol{\delta}^{(1)} \\
\mathbf{H}^{(2)} &= \mathrm{ReLU}\left(\mathrm{NN}^{(2)}(\widetilde{\mathbf{H}}^{(1)})\right) \\
\widetilde{\mathbf{H}}^{(2)} &= \mathbf{H}^{(2)} \odot \boldsymbol{\delta}^{(2)}
\end{aligned} \tag{24.43}$$

其中，$\mathrm{NN}^{(l)}(\cdot)$ 表示第 l 层的全连接网络，$\widetilde{\mathbf{H}}^{(l)}$ 表示经过门控调节后的隐向量。第三层先计算输出层的线性变换，再利用第三个门控向量调节输出，最终通过 Sigmoid 函数得到预测概率：

$$\begin{aligned}
\mathrm{logit} &= \mathrm{NN}^{(3)}(\widetilde{\mathbf{H}}^{(2)}) \odot \boldsymbol{\delta}^{(3)} \\
\hat{y} &= \sigma(\mathrm{logit})
\end{aligned} \tag{24.44}$$

以上公式描述了 CTR 或 CVR 预测分支中的计算过程。通过三个门控神经单元，PPNet 模块能够在不同网络层对隐向量进行逐元素调节，使得模型根据当前样本的特征动态改变有效的表示模式。相比起直接为每个样本生成完整的独立参数，门控方法保留了共享预测网络的主体结构，并通过较为轻量的门控计算实现样本级个性化。

从整体设计来看，AFP 模块和 PPNet 模块分别解决两个相互关联的问题。AFP 模块学习哪些特征维度应当通过个性化分支或主预测分支进行优化，PPNet 模块则将个性化分支的特征转换为门控信号，对主预测网络各层的隐表示进行动态调节。通过两者的结合，使得 APPNet 模型能够减少不同学习目标对共享特征的梯度冲突，同时保留共享网络对整体数据规律的学习能力。

24.4.3 全空间多任务学习目标

在电商推荐场景中，点击和转化通常存在明确的行为依赖关系。用户需要先点击物品，才可能进一步产生购买或其他转化行为。如果只使用点击后的样本来训练转化率模型，训练数据便会受到点击行为的选择影响，也难以充分利用未点击样本所提供的信息。因此，APPNet 模型采用 ESMM 组织点击率与点击后转化率之间的关

系，在曝光样本空间内联合优化相关预测目标。

假设模型分别输出点击率预测值 $\hat{y}_{\text{CTR}}$ 和点击后转化率预测值 $\hat{y}_{\text{CVR}}$。根据点击与转化之间的概率关系，用户从曝光到点击并最终完成转化的联合概率可以表示为：

$$\hat{y}_{\text{CTCVR}} = \hat{y}_{\text{CTR}} \cdot \hat{y}_{\text{CVR}} \tag{24.45}$$

其中，$\hat{y}_{\text{CTCVR}}$ 表示点击转化联合概率。对应的联合标签定义为：

$$y_{\text{CTCVR}} = \mathbb{I}(y_{\text{CTR}} = 1 \wedge y_{\text{CVR}} = 1) \tag{24.46}$$

也就是说，当样本发生点击且最终完成转化时，$y_{\text{CTCVR}} = 1$；否则取 0。对于未点击的曝光样本，即使无法观察其点击后的转化行为，也可以确定该样本没有产生从曝光到点击再到转化的完整行为链路，因此其联合标签为 0。这种标签定义使联合转化目标能够在整个曝光样本空间中进行训练。

基于上述关系，APPNet 模型采用点击率损失和点击转化联合损失构成最终训练目标：

$$\mathcal{L} = \mathcal{L}_{\text{CE}}\left(y_{\text{CTR}}, \hat{y}_{\text{CTR}}\right) + \alpha\, \mathcal{L}_{\text{CE}}\left(y_{\text{CTCVR}}, \hat{y}_{\text{CTR}}\hat{y}_{\text{CVR}}\right) \tag{24.47}$$

其中，$\mathcal{L}_{\text{CE}}(\cdot,\cdot)$ 表示交叉熵损失，α 为平衡点击率损失与点击转化联合损失的超参数。第一个损失直接约束点击率预测，第二个损失则通过点击率和点击后转化率预测值的乘积约束联合转化概率。两个目标共同训练，使得模型既能学习用户是否点击，也能学习曝光最终转化的概率。

综上所述，APPNet 模型在参数个性化网络的基础上引入自动特征划分机制，并通过 Stop Gradient 操作区分个性化分支和主预测分支对特征表示的反向更新。Feature-wise AFP 模块和 Bit-wise AFP 模块分别从字段级和 Embedding 维度级学习特征划分方式，PPNet 模块则将个性化特征转化为逐层门控信号，动态调节预测网络的隐层表示。结合 ESMM 框架对点击与转化关系的建模，APPNet 模型为复杂电商场景下的转化率预测提供了一种兼顾共享学习与样本个性化的建模方案。

24.5 本章小结

本章围绕工业推荐系统中的个性化网络参数，系统介绍了 POSO、APG、PEPNet 和 APPNet 等代表性方法。这些方法的共同动机在于：仅依赖输入特征表达用户差异并不充分，当不同用户群体的样本规模存在较大差异时，统一网络参数容易被主流群体主导，使新用户和长尾用户的行为模式难以被充分学习。个性化网络参数通过在共享模型主体上引入由个性化特征控制的门控、参数生成或隐藏单元调制机制，使不同用户能够以不同方式使用网络内部的计算过程，从而在不显著增加模型规模的前提下增强模型的个性化表达能力。

首先，本章介绍了 POSO 模型。POSO 模型认为冷启动问题不仅来自行为数据不足，还与个性化信息的淹没有关。即使输入中包含新用户标记等个性化特征，模型在训练过程中也可能逐渐依赖其他更容易优化的特征。针对这一问题，POSO 模型将个性化机制嵌入现有网络的部分中间模块，分别给出了针对 MLP 中间激活、MHA 注意力计算以及 MMoE 专家门控的个性化设计。在冷启动场景中，POSO 模型通过分桶后的累计观看次数刻画用户所处的冷启动阶段，并通过视频发布时间与累计曝光等特征刻画物品生命周期，使门控网络能够根据用户或物品状态动态调节各个子模块的贡献。

随后，本章介绍了 APG 模型。APG 模型从另一个角度切入，直接根据样本的条件表示动态生成网络参数，并提供了 Group-wise、Mix-wise 和 Self-wise 三种条件设计方式。考虑到为每个样本生成完整权重矩阵的开销，APG 模型进一步通过低秩参数化缩小需要生成的参数规模，通过分解式前向计算避免显式重构权重矩阵，通过参数共享将大部分参数固定为跨样本公共部分，并利用过参数化在训练阶段增强共享参数的表达能力。这一系列设计使 APG 在引入样本级参数个性化的同时，将单层计算的复杂度从 $O(NM)$ 左右降低到 $O(K^2D + NK + MK + K^2)$。

在此基础上，本章继续介绍了 PEPNet 模型。PEPNet 模型主要面向多场景、多任务推荐，其 Gate NU 组件借鉴了第十五章介绍的 LHUC 方法，并将个性化先验注入底层 Embedding 和顶层 DNN 两个关键位置：EPNet 模块根据场景侧信息调节共享 Embedding，使同一特征在不同场景中获得不同的有效表示；PPNet 模块根据用户、物品和作者信息调制各任务塔的隐藏单元，缓解统一任务塔难以兼顾不同用户偏好的问题。同时，PEPNet 模型还介绍了 GSET Embedding 管理、按特征设置同步上限以及 Embedding 与 DNN 差异化训练等工程优化，说明个性

化网络结构要在大规模系统中稳定运行，离不开配套的存储与训练机制。

最后，本章介绍了 APPNet 模型。APPNet 模型针对参数个性化中先验特征依赖人工指定、以及个性化分支与主预测分支可能存在梯度冲突的问题，提出自动特征划分模块 AFP。Feature-wise AFP 模块和 Bit-wise AFP 模块分别从特征字段级和 Embedding 维度级学习特征的分配方式，并通过停止梯度操作区分两条分支对共享特征表示的反向更新，从而缓解不同学习目标之间的相互干扰。在预测网络上，APPNet 模型复用 PPNet 模块的门控机制对各层隐表示进行逐元素调节，并结合 ESMM 框架在曝光空间内联合建模点击率与转化率。

总体而言，个性化网络参数的目标并不是为每个用户单独训练一套模型，而是让共享模型内部的计算过程能够随用户、场景或任务自适应变化。从 POSO 和 PEPNet 的门控调制，到 APG 的动态参数生成，再到 APPNet 的特征自动划分，这些方法各有侧重。其中门控方式实现简单、易于插入已有网络，参数生成能力更强但开销更高；而特征划分方式则着眼于避免分支之间的梯度冲突。在实际的工业级推荐系统中，选择哪一种方案需要综合考虑用户群体的行为差异、可用的个性化特征、模型结构以及工程资源，并在表达能力、参数开销、在线延迟和训练稳定性之间进行权衡。对于推荐算法初学者以及推荐算法工程师而言，理解这些方法背后的共同思想比记住某一种具体结构更重要。

24.6 参考文献

[1] Jianxin Chang et al. “Pepnet: Parameter and embedding personalized network for infusing with personalized prior information”. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023, pp. 3795–3804.

[2] Shangfeng Dai et al. “POSO: personalized cold start modules for large-scale recommender systems”. In: *arXiv preprint arXiv:2108.04690* (2021).

[3] Shui Liu et al. “APPNet: Automatic Feature Partitioning-Based Parameter Personalized Network for Conversion Prediction in E-commerce”. In: *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining*. 2026, pp. 416–425.

[4] Bencheng Yan et al. “Apg: Adaptive parameter generation network for click-through rate prediction”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24740–24752.