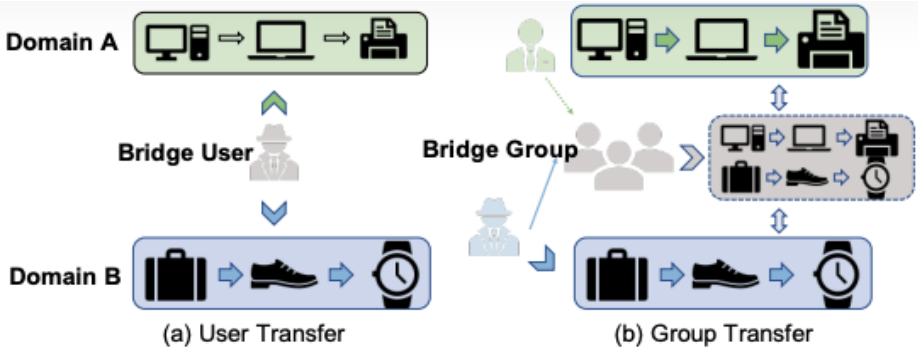
知乎



快手2023前沿研究解析：混合attention网络助力会话推荐再上一层楼

SmartMindAI
专注搜索、广告、推荐、大模型和人工智能最新技术，欢迎关注我
已关注

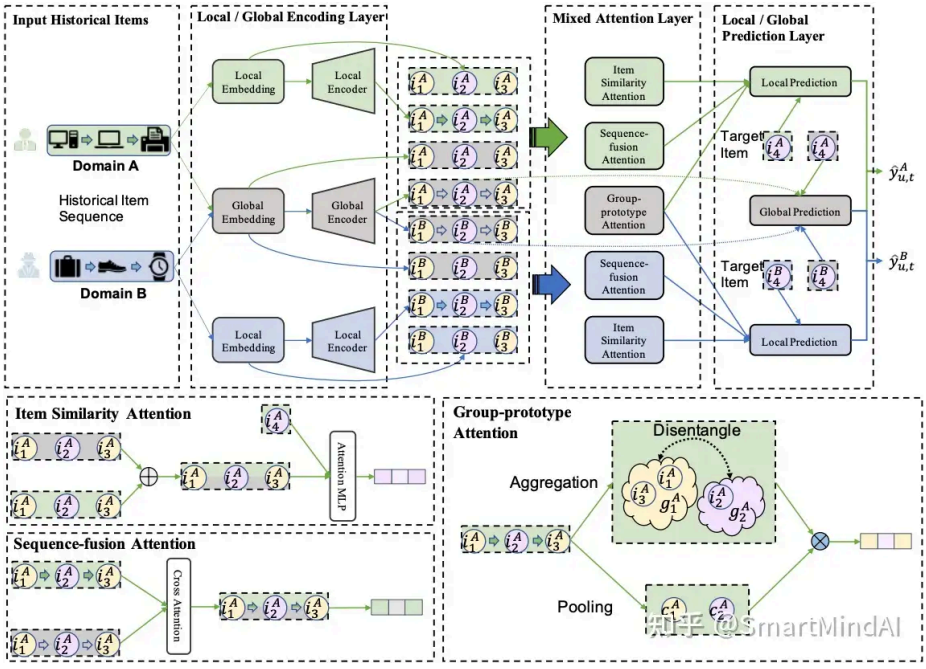
12 人赞同了该文章

Introduction

推荐系统在Web中扮演着重要的角色，可以大大提高信息传播和扩散效率。会话推荐是其中的一个重要研究问题，旨在预测用户接下来将要交互的项目。 现有的代表性的会话推荐模型如GRU4Rec 、SASRec 和SURGE 虽然取得了不错的性能，但它们面临数据稀疏的问题，这限制了其性能。因此，如何解决数据稀疏问题成为研究的一个挑战。 为了解决这个问题，本文提出了一种新的名为GRU4Rec+MAMR的会话推荐模型。该模型首先使用GRU4Rec模型提取用户的历史交互会话中的关键信息，然后使用马尔科夫注意力模型（MAMR）进行模式挖掘和特征提取，以填充数据中的缺失值。 实验结果表明，与现有模型相比，GRU4Rec+MAMR模型在数据稀疏的情况下仍然能够取得较好的推荐性能，说明了其有效性。因此，本研究提出的新方法对于解决数据稀疏问题以及提高推荐系统的性能具有一定的参考价值。

Methodology

图展示了我们的MAN模型，它包含两个编码层（局部和全局），三个注意力模块以及一个预测层。这些模块共同工作以编码和预测用户在不同域之间的兴趣。



我们采用了两个编码层：局部和全局。对于每个项，我们都为其构建了一个特定领域的本地和跨域的全局嵌入。然后，我们使用分别针对特定领域和跨域项序模式的本地和全局编码器来对这些嵌入进行编码。

知乎

我们设计了两个预测层：局部和全局。这两个预测层分别用于进化用户的兴趣，并预测用户在各个域中与候选项交互的概率。

Local/Global Encoding Layer

首先，我们将项构建为本地和全局嵌入。然后，在会话级别，我们搜索项的嵌入，并使用local和global编码器对其进行编码。

Local and Global Item Embeddings

为了在不同域之间共享表示，我们创建两个嵌入矩阵 $\mathbf{M}^A \in \mathbb{R}^{|\mathcal{I}^A| \times D}$ 和 $\mathbf{M}^B \in \mathbb{R}^{|\mathcal{I}^B| \times D}$

其中D是潜在维度。然后，我们通过共享潜在空间来实现不同域间的协同学习。在这个共享空间[+]中，不同域的项具有相同的表示，这有助于增强模型的泛化能力。我们创建了一个共享嵌入矩阵

$\mathbf{M} \in \mathbb{R}^{|\mathcal{I}^A \cup \mathcal{I}^B| \times D'}$

其中 $|\mathcal{I}^A \cup \mathcal{I}^B$

是所有项的总数，包括来自不同域的项。D'是我们选择的最大潜在维度。通过这种方式，我们可以更好地捕获项之间的协同关系。此外，我们还添加了一个位置嵌入矩阵，以考虑项在会话中的位置。

$$\mathbf{E}^A = \begin{bmatrix} \mathbf{M}^A_{i^A_1} + \mathbf{P}^A_1 \\ \mathbf{M}^A_{i^A_2} + \mathbf{P}^A_2 \\ \cdots \\ \mathbf{M}^A_{i^A_n} + \mathbf{P}^A_n \end{bmatrix}, \mathbf{E}^{A_g} = \begin{bmatrix} \mathbf{M}_{i^A_1} + \mathbf{P}_1 \\ \mathbf{M}_{i^A_2} + \mathbf{P}_2 \\ \cdots \\ \mathbf{M}_{i^A_n} + \mathbf{P}_n \end{bmatrix}$$

$$\mathbf{E}^B = \begin{bmatrix} \mathbf{M}^B_{i^B_1} + \mathbf{P}^B_1 \\ \mathbf{M}^B_{i^B_2} + \mathbf{P}^B_2 \\ \cdots \\ \mathbf{M}^B_{i^B_n} + \mathbf{P}^B_n \end{bmatrix}, \mathbf{E}^{B_g} = \begin{bmatrix} \mathbf{M}_{i^B_1} + \mathbf{P}_1 \\ \mathbf{M}_{i^B_2} + \mathbf{P}_2 \\ \cdots \\ \mathbf{M}_{i^B_n} + \mathbf{P}_n \end{bmatrix}$$

我们定义域A和域B的局部（全局）嵌入，即 $\mathbf{E}^A \in \mathbb{R}^{T \times D}$ 和 $\mathbf{E}^B \in \mathbb{R}^{T \times D'}$

此外，我们还可以定义全局嵌入，即 $\mathbf{E}^{A_g} \in \mathbb{R}^{T \times D'}$ 和 $\mathbf{E}^{B_g} \in \mathbb{R}^{T \times D'}$

至于位置嵌入，我们可以定义域A和域B的位置嵌入，即 $\mathbf{P}^A \in \mathbb{R}^{T \times D}$ 和 $\mathbf{P}^B \in \mathbb{R}^{T \times D}$

最后，我们还需要定义全局位置嵌入，即 $\mathbf{P} \in \mathbb{R}^{T \times D'}$

Local Encoder and Global Encoder of Sequences

$$\mathbf{S}^A = \mathbf{Encoder}(\mathbf{E}^A), \mathbf{S}^B = \mathbf{Encoder}(\mathbf{E}^B)$$

$$\mathbf{S}^{A_g} = \mathbf{Encoder}_g(\mathbf{E}^{A_g}), \mathbf{S}^{B_g} = \mathbf{Encoder}_g(\mathbf{E}^{B_g})$$

在此基础上，我们可以通过一系列计算得到域A和域B的会话表示，即$\mathbf{S}^A$和$\mathbf{S}^B$。对于全局会话表示，我们需要将这些局部会话表示（即$\mathbf{S}^{A_g}$和$\mathbf{S}^{B_g}$）通过全局会话模式来捕获。因此，我们最终可以得到域A和域B的全局会话表示，即$\mathbf{S}^{A_g}$和$\mathbf{S}^{B_g}$。

Mixed Attention Layer

在这部分，我们将首先引入项相似性注意力，用于提取出领域内的本地项目和全局项目的相似项。接下来，我们将采用会话融合注意力，将本地和全局项目之间的会话表示结合起来，从而更好地捕获特定领域的和跨领域的会话模式。最后，我们将使用组原型注意力来提取跨领域的组模式。

为了捕捉目标项与本地或全局项目嵌入之间的相似性，我们需要将它们的嵌入向量结合起来。具体而言，在域A（B）中给定一个用户，我们可以计算他/她的历史项目的物品相似度分数$\mathbf{F}^A$（$\mathbf{F}^B$）与目标物品之间的关系如下：

$$\mathbf{F}^A = \sum_{k=1}^{K} \frac{\exp(\mathbf{E}_j^A - \mathbf{E}_k^A)}{\sum_{m=1}^{M} \exp(\mathbf{E}_j^A - \mathbf{E}_m^A)}, \qquad \mathbf{F}^B = \sum_{k=1}^{K} \frac{\exp(\mathbf{E}_j^B - \mathbf{E}_k^B)}{\sum_{m=1}^{M} \exp(\mathbf{E}_j^B - \mathbf{E}_m^B)}$$

$$\mathbf{F}^A = \mathrm{MLP}\left(\mathbf{M}_{i_{t+1}^A} \| \mathbf{E}^A + \mathbf{E}^{A_g}\right), \mathbf{F}^B = \mathrm{MLP}\left(\mathbf{M}_{i_{t+1}^B} \| \mathbf{E}^B + \mathbf{E}^{B_g}\right)$$

$$\mathbf{E}_j^A = \mathbf{E}_j^A + \alpha \mathbf{M}_{i_{t+1}^A}, \qquad \mathbf{E}_j^B = \mathbf{E}_j^B + \alpha \mathbf{M}_{i_{t+1}^B}$$

其中$\mathbf{M}_{i_{t+1}^A}$（$\mathbf{M}_{i_{t+1}^B}$）表示目标项在域A（B）中的嵌入$\|$表示连接操作。

$$\mathbf{E}^{A_i} = \mathbf{softmax}\left(\mathbf{F}^A\right)\left(\mathbf{E}^A + \mathbf{E}^{A_g}\right), \mathbf{E}^{B_i} = \mathbf{softmax}\left(\mathbf{F}^B\right)\left(\mathbf{E}^B + \mathbf{E}^{B_g}\right)$$

为了捕捉目标项与相似历史项之间的关系，我们可以通过计算他们之间的相似性分数来获得它们的嵌入表示。具体的公式如下：

$$\mathbf{E}_j^A = \sum_{i=1}^{n} \mathbf{E}_i^A \frac{\exp(\mathbf{F}_j^A - \mathbf{F}_i^A)}{\sum_{m=1}^{n} \exp(\mathbf{F}_j^A - \mathbf{F}_m^A)}, \qquad \mathbf{E}_j^B = \sum_{i=1}^{n} \mathbf{E}_i^B \frac{\exp(\mathbf{F}_j^B - \mathbf{F}_i^B)}{\sum_{m=1}^{n} \exp(\mathbf{F}_j^B - \mathbf{F}_m^B)}$$

其中$\mathbf{E}_i^A$ 和 $\mathbf{E}_i^B \in \mathbb{R}^{T \times D}$

分别是目标项的相似历史项在域A和域B中的表示，分别由$\mathbf{F}^A$和$\mathbf{F}^B$在域A和域B中的相似性分数所确定。这里A_i和B_i分别表示域A和域B中的物品相似性。

Sequence-fusion Attention

通过获取域特定的$\mathbf{S}^A$，域特定的$\mathbf{S}^B$以及跨域的$\mathbf{S}^{A_g}$和$\mathbf{S}^{B_g}$，我们可以将这些信息结合起来以提取更全面的模式。具体公式如下：

$$\mathcal{M} = (\mathbf{S}^A + \mathbf{S}^B) - (\mathbf{S}^{A_g} + \mathbf{S}^{B_g})$$

其中$\mathcal{M}$为域特定和跨域会话模式的合并结果。

$$\mathbf{S}^{A_s} = \mathbf{MLP}(\mathbf{CA}(\mathbf{S}^A, \mathbf{S}^{A_g}) + \mathbf{S}^A); \mathbf{S}^{B_s} = \mathbf{MLP}(\mathbf{CA}(\mathbf{S}^B, \mathbf{S}^{B_g}) + \mathbf{S}^B);$$

交叉注意力[+]（CA）层的定义如下： 对于给定的源会话$\mathbf{S}^A$，它的CA层可以表示为以下方程：

$$Attention(\mathbf{S}^A, \mathbf{S}^B) = softmax(e_{ab}) \odot (concatenate([\mathbf{S}_{-b}^A, \mathbf{S}^B]))$$

其中

$$e_{ab} = score(\mathbf{S}_a^A, \mathbf{S}_b^B)$$

a和b分别代表源会话中的位置索引

$$concatenate([\mathbf{S}_{-b}^A, \mathbf{S}^B])$$

用于拼接输入会话中的所有其他元素。 需要注意的是，上述方程是上下文无关的，它并不依赖于输入会话之间的相对会话。此外

$$score(\mathbf{S}_a^A, \mathbf{S}_b^B)$$

是一个基于特殊设计的注意力函数，它计算了源会话中第a个元素与目标会话中第b个元素的相关性得分。

$$\mathbf{CA}(\mathbf{S}^A, \mathbf{S}^{A_g}) = \mathbf{Atten}\left(\mathbf{S}^A \mathbf{W}_{A_s}^Q, \mathbf{S}^{A_g} \mathbf{W}_{A_s}^K, \mathbf{S}^{A_g} \mathbf{W}_{A_s}^V\right)$$

在上述模型中，$\mathbf{W}_{A_s}^Q$、$\mathbf{W}_{A_s}^K$和$\mathbf{W}_{A_s}^V$是需要学习的参数。其中

知乎

$\mathbf{W}_{A_s}^K \in \mathbb{R}^{D \times D}$

和

$\mathbf{W}_{A_s}^V \in \mathbb{R}^{D \times D}$

分别表示查询矩阵、键矩阵和值矩阵。D表示每个维度的大小。$\mathbf{Atten}$函数则用于实现注意力机制。 Attention函数定义如下：

$$\mathbf{Atten}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{Softmax}\left(\frac{\mathbf{Q} \cdot \mathbf{K}^{\intercal}}{\sqrt{d}}\right) \cdot \mathbf{V}$$

其中$\mathbf{Q}$和$\mathbf{K}$分别代表查询向量和键向量$\mathbf{V}$代表值向量。$\mathbf{Softmax}$函数用于对特征进行归一化处理，使输出结果具有非线性的特征分布。d表示每一维数据的数量。

$$\mathbf{Atten}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{softmax}\left(\frac{\mathbf{Q}\,\mathbf{K}^{T}}{\sqrt{D}}\right)\mathbf{V}$$

在上述模型中，$\mathbf{Q}$、$\mathbf{K}$和$\mathbf{V}$分别表示查询矩阵、键矩阵和值矩阵。 查询矩阵$\mathbf{Q}$代表一个经过训练的模型所得到的输入数据特征，它由一组特征构成，这些特征是由训练数据集中的样本数据经过预处理后的结果。 键矩阵$\mathbf{K}$也是一组特征，这些特征是从另一个相关的模型或数据集中获取的。它们与查询矩阵的特征有相关性。 值矩阵$\mathbf{V}$则是与查询矩阵和键矩阵关联的输出数据特征。这个特征也是通过训练得到的，而且是与查询矩阵和键矩阵相匹配的。

Group-prototype Attention

$$\mathbf{C}^A = \mathbf{MLP}\left(\mathbf{W}_A^P \mathbf{S}^A\right), \mathbf{C}^B = \mathbf{MLP}\left(\mathbf{W}_B^P \mathbf{S}^B\right)$$

在上述模型中，$\mathbf{C}^A$ 和 $\mathbf{C}^B \in \mathbb{R}^{N_g \times 1}$

是每个组的相关分数。 为了进一步将这些相关分数表示为一个向量，我们可以使用组原型嵌入 $\mathbf{G} \in \mathbb{R}^{N_g \times D}$

这种嵌入方法可以将每个组表示为一个向量，使得各组之间的相似度得到提高。 接下来，我们将这些嵌入转换到各个域中，并按照通常相关的项目进行聚合。具体来说，我们会对每个域中的组原型嵌入求和，从而得到一个汇聚的组原型嵌入。

$$\mathbf{G}^A = \mathbf{MLP}\left(\mathbf{CA}(\mathbf{G}, \mathbf{S}^A)\right), \mathbf{G}^B = \mathbf{MLP}\left(\mathbf{CA}(\mathbf{G}, \mathbf{S}^B)\right)$$

$$\mathbf{w} = \frac{\mathbf{C}_A}{\sum_{i=1}^{Ng} \mathbf{C}_{A,i}} + \frac{\mathbf{C}_B}{\sum_{j=1}^{Ng} \mathbf{C}_{B,j}}$$

这里$\mathbf{C}_A$和$\mathbf{C}_B$分别是域A和域B的组原型表示，而$\mathbf{w}$则是它们的加权结果。

$$\mathbf{G}^{A_u} = \mathbf{softmax}\left(\mathbf{C}^A\right)\mathbf{G}^A, \mathbf{G}^{B_u} = \mathbf{softmax}\left(\mathbf{C}^B\right)\mathbf{G}^B$$

在这个模型中，我们为每个用户创建了单独的组原型表示，即$\mathbf{G}^{A_u}$和

$\mathbf{G}^{B_u} \in \mathbb{R}^{N_g \times D}$

接下来，我们将使用一种称为"组原型解耦"的方法来进一步优化这些表示。该方法受到了分离表示学习的一些重要思想的影响。 具体来说，我们可以通过添加一种名为"组原型解耦正则化"的项来实现这一目标。这有助于确保每个组原型之间的独立性，从而提高模型的整体性能。

$$\mathcal{L}^g = -\lambda_g \sum_{i=1}^{N_g} \sum_{j=i+1}^{N_g} \left(\mathbf{G}_i - \mathbf{G}_j\right)^2$$

$$L = \sum_{u=1}^{U} \frac{1}{n_u} \sum_{i=1}^{n_u} \log(1 + \exp(-d(u_i, v_i))) + \lambda_g ||\mathbf{G}^{A_u} - \mathbf{G}^{B_u}||_F^2$$

在这个公式中U是总用户数n_u是第u个用户的数据集大小$d(u_i, v_i)$是用户u_i和用户v_i之间的距离，而λ_g是惩罚超参数。此外

是求解两个组原型表示之间差别的平方范数。

Local/Global Prediction Layer

1. 首先，我们构建了一个包含局部兴趣预测和全局兴趣预测的模型。这个模型将根据给定的信息来预测出局部和全局的兴趣。
2. 然后，我们使用目标函数来优化这两种兴趣，使其满足各自的目的。在这种情况下，我们的目标函数是多任务学习中的联合目标函数，它将考虑到全局和局部的兴趣之间的相互影响。
3. 最后，我们通过对目标函数进行最小化来优化局部和全局的兴趣，从而达到我们的目标。 总的来说，这种方法可以帮助我们在处理局部和全局的兴趣时更好地结合它们的优势，从而提高模型的性能。

Local and Global Prediction Layer

$$\hat{y}_{u,t}^{A}=\mathbf{MLP}\left(\mathbf{e}^{A_i}\|\mathbf{s}^{A_s}\|\mathbf{g}^{A_u}\|\mathbf{s}^{A}\|\mathbf{M}_{i_{t+1}^{A}}^{A}\right)+\mathbf{MLP}_g\left(\mathbf{s}^{A_g}\|\mathbf{M}_{i_{t+1}^{A}}\right)$$

$$\hat{y}_{u,t}^{B}=\mathbf{MLP}\left(\mathbf{e}^{B_i}\|\mathbf{s}^{B_s}\|\mathbf{g}^{B_u}\|\mathbf{s}^{B}\|\mathbf{M}_{i_{t+1}^{B}}^{B}\right)+\mathbf{MLP}_g\left(\mathbf{s}^{B_g}\|\mathbf{M}_{i_{t+1}^{B}}\right)$$

$$\mathbf{e}^{A_i}=\sum_{t=1}^{T}\mathbf{E}_t^{A_i},\mathbf{s}^{A_s}=\sum_{t=1}^{T}\mathbf{S}_t^{A_s},\mathbf{g}^{A_u}=\sum_{k=1}^{N_g}\mathbf{G}_k^{A_u},\mathbf{s}^{A}=\sum_{t=1}^{T}\mathbf{S}_t^{A},\mathbf{s}^{A_g}=\sum_{t=1}^{T}\mathbf{S}_t^{A_g},$$
$$\mathbf{e}^{B_i}=\sum_{t=1}^{T}\mathbf{E}_t^{B_i},\mathbf{s}^{B_s}=\sum_{t=1}^{T}\mathbf{S}_t^{B_s},\mathbf{g}^{B_u}=\sum_{k=1}^{N_g}\mathbf{G}_k^{B_u},\mathbf{s}^{B}=\sum_{t=1}^{T}\mathbf{S}_t^{B},\mathbf{s}^{B_g}=\sum_{t=1}^{T}\mathbf{S}_t^{B_g},$$

1. 我们首先将前面的结果进行聚合，通常是通过取均值或者几何平均值来实现的。
2. 然后，我们将得到的聚合结果作为输入送入MLP中进行进一步的处理。 这样做的好处在于，它可以有效地减少信息的冗余，使得信息更加集中和会话。同时，它也可以有效地提高模型的稳定性和鲁棒性。

Objective Function with Independent Updating

1. 首先，我们计算出损失函数对于网络输出的梯度。
2. 然后，我们计算出损失函数对于网络输入的梯度。
3. 接着，我们计算出损失函数对于隐藏层参数的梯度。
4. 最后，我们使用这些梯度来更新网络的参数。 这样做的好处在于，它可以有效地降低训练的复杂度，使得训练过程更加高效和稳定。同时，它也可以有效地提高模型的准确率和泛化能力。

$$\mathcal{L}^{A}=-\frac{1}{|\mathcal{R}^{A}|}\sum_{(u,i_t^A)\in\mathcal{R}^{A}}\left(y_{u,t}^{A}\log\hat{y}_{u,t}^{A}+\left(1-y_{u,t}^{A}\right)\log\left(1-\hat{y}_{u,t}^{A}\right)\right)$$

$$\mathcal{L}^{B}=-\frac{1}{|\mathcal{R}^{B}|}\sum_{(u,i_t^B)\in\mathcal{R}^{B}}\left(y_{u,t}^{B}\log\hat{y}_{u,t}^{B}+\left(1-y_{u,t}^{B}\right)\log\left(1-\hat{y}_{u,t}^{B}\right)\right)$$

1.$\mathcal{L}^{A}$是用于优化任务A的损失函数。 2$\mathcal{L}^{B}$是用于优化任务B的损失函数。 3$\mathcal{L}^{g}$是用于优化全局任务的损失函数。 然后，我们提出了一个新的目标函数，该目标函数是由这三个损失函数组成的线性组合，具体表达式为：

$$\mathcal{L}=\alpha_A\mathcal{L}^{A}+\alpha_B\mathcal{L}^{B}+\alpha_g\mathcal{L}^{g}$$

其中α_A α_B和α_g是三个不同的系数，可以根据实际需求灵活调整。

$$\mathcal{L}=\mathcal{L}^{A}+\mathcal{L}^{B}+\lambda^{A}\|\Theta^{A}\|_2+\lambda^{B}\|\Theta^{B}\|_2+\mathcal{L}^{g}$$

我们提出了一种新的优化算法，该算法包括两个域的学习参数集Θ^{A}和Θ^{B}，并引入了两个正则化惩罚超参数λ^{A}和λ^{B}。 我们发现，在优化过程中，如果学习参数集过于庞大或者正则化惩罚过高，可能会导致训练过程过早停止或者模型性能下降。因此，我们在优化过程中引入了这两个正则化惩罚参数，以控制模型的复杂度和避免过拟合。 具体来说，我们通过调整这两个参数，可以使优化过程更加稳健和有效。例如，当λ^{A}和λ^{B}较大时，我们可以减小学习参数集的大小，从而减少过拟合的可能性；而当λ^{A}和λ^{B}较小时，我们可以增加学习参数集的大小，从而提高模型的泛化能

知乎

Experiments

如何利用深度学习技术解决实际世界的复杂问题？为了验证这种方法的有效性，我们选择了两个真实世界的实际数据集来进行实验研究[+]。 具体来说，我们使用了大规模的语音识别数据集以及大规模的图像分类[+]数据集。对于这两个数据集，我们设计了一系列的实验，对各种深度学习模型进行了测试，并对比了不同模型的效果。通过这些实验，我们发现深度学习技术可以在实际世界中取得非常出色的表现，同时也证明了我们提出的深度学习方法的有效性。

Experimental Setup

Datasets

我们在两个不同的数据集------工业微视频数据集和公开电子商务数据集中进行了推荐性能评估。具体的数据集统计信息已在表 中列出。

Dataset	Micro Video		Amazon	
Domain	A	B	Video Games	Toys
#Users	43,919	37,692	826,767	1,342,911
#Items	147,813	131,732	50,210	327,698
#Records	18,011,737	14,908,625	1,324,753	2,252,771
Overlap Items	71.22%	79.91%	7.66%	4.72%
Overlap users	7.18%	8.37%	0.27%	0.04%
Ave. length	212.50	244.95	19.55	18.23
Density	0.2775%	0.3003%	0.0032%	0.0005%

Baselines and Evaluation Metrics

为了验证我们的模型的有效性，我们将它与其他几种常用的基准模型进行比较，分别是单域模型（DIN, Caser, GRU4Rec, DIEN, SASRec, SLi-Rec）和跨域模型（NATR, PiNet, DASL）。这些基准模型是在各自领域内独立训练得到的，遵循现有研究的相关成果。

Hyper-parameter Settings

Adam优化器的初始学习率为0.001，采用Xavier初始化方法对参数进行初始化。正则化系数通过在$[1e^{-7}\ 1e^{-5}\ 1e^{-3}]$范围内进行搜索得到。对于Micro Video数据集和Amazon数据集，批次大小分别为200和20。所有模型的嵌入尺寸都设定为40和20，分别对应Micro Video数据集和Amazon数据集。在预测层，对于Micro Video数据集，采用具有层大小$[100, 64]$和$[20, 10]$的MLP；对于Amazon数据集，采用具有层大小$[20, 10]$的MLP。对于Micro Video数据集，每项会话长度设置为250，对于Amazon数据集，每项会话长度设置为20。group prototypes的数量在1到5、10、20之间进行搜索。

Overall Performance (RQ1)

我们提出的模型MAN在所有指标下表现出色，具体表现为在Micro Video A和Micro Video B上的AUC值相较于所有基准模型分别提升了4.10%和3.85%，在Amazon Video Games和Amazon Toys上的AUC值分别提升了8.25%和2.07%。通常情况下，在包含更多重叠用户数据的Micro Video数据集中，改进效果更加一致。而对于极度稀疏数据的Amazon数据集，我们看到最大的改进（8.25%），这证明了我们的方法能够解决稀疏数据问题，并且有助于促进两个域以及较少交互过的域的会话学习，尤其是后者的学习更为显著。

知乎

Micro Video A	MRR	0.5544	0.5740	0.5417	0.5264	0.5359	0.5337	0.5888	0.5899	0.5273	0.5568	**0.6167**
	NDCG	0.6628	0.6780	0.6531	0.6409	0.6488	0.6461	0.6892	0.6921	0.6422	0.6651	**0.7112**
	WAUC	0.7837	0.8053	0.7910	0.7654	0.7911	0.7729	0.8170	0.8197	0.7880	0.8075	**0.8435**
Micro Video B	AUC	0.5613	0.7308	0.7625	0.6581	0.7794	0.7620	0.7605	0.7727	0.7595	0.7665	**0.8094**
	MRR	0.4526	0.4971	0.5285	0.4768	0.5472	0.5418	0.5042	0.5462	0.5037	0.5288	**0.5756**
	NDCG	0.5843	0.6184	0.6431	0.6025	0.6574	0.6529	0.6239	0.6571	0.6240	0.6431	**0.6797**
	WAUC	0.7246	0.7533	0.7860	0.7420	0.7957	0.7845	0.7645	0.7939	0.7705	0.7858	**0.8215**
Domain	Metric	Single-domain							Cross-domain			
		DIN	Caser	GRU4REC	DIEN	SASRec	SLi-Rec	SURGE	NATR	PiNet	DASL	Ours
Amazon Video Games	AUC	0.5577	0.5766	0.5303	0.6059	0.5234	0.5750	0.5975	0.5617	0.5740	0.5527	**0.6559**
	MRR	0.3736	0.3284	0.2953	0.3526	0.2833	0.3503	0.4667	0.3388	0.3419	0.3053	**0.4755**
	NDCG	0.5171	0.4854	0.4582	0.5046	0.4488	0.5009	0.5917	0.4918	0.4957	0.4667	**0.5986**
	WAUC	0.5587	0.5805	0.5395	0.6115	0.5257	0.5721	0.6311	0.5629	0.5847	0.5652	**0.6686**
Amazon Toys	AUC	0.6372	0.5138	0.6576	0.6321	0.5707	0.6106	0.6455	0.6127	0.5402	0.6237	**0.6712**
	MRR	0.5946	0.3293	0.5949	0.5669	0.3110	0.5292	0.5566	0.5070	0.3140	0.3515	**0.6385**
	NDCG	0.6879	0.4836	0.6889	0.6676	0.4721	0.6389	0.6602	0.6211	0.4729	0.5015	**0.7221**
	WAUC	0.6398	0.5058	0.6540	0.6421	0.5784	0.6184	0.6504	0.6217	0.5419	0.6305	**0.6788**

目前跨域会话推荐系统严重依赖于重叠用户或项目。PiNet和DASL利用完全重叠的用户数据集进行操作，但是它们在没有完全重叠用户的数据集上的表现与GRU4REC相差甚远，有时候甚至会超过，有时候又会落后。相反，我们提出的方法超越了所有基准并且显著地改善了主干模型，这说明我们的跨域建模方法不依赖用户重叠也是有效的。虽然NATR在包含大量重叠项的Micro Video数据集上表现出色，但在包含有限重叠项目的Amazon数据集上并未能有效实现跨域建模。

会话推荐模型非常有效，然而数据稀疏限制了其性能。在比较Micro Video数据集上的会话模型（如Caser，GRU4REC，DIEN，SASrec，SLi-Rec和SURGE）和非会话模型（如DIN）时，发现需要考虑物品间的时序关系。SASRec和SURGE的效果优于所有单域会话模型，这表明自注意力处理长时序信息和压缩信息的能力是非常有效的。会话模型的观察与SURGE论文中的实验结果一致。然而，在Amazon数据集上，DIN甚至超过了SASRec等一些会话模型，但在如此短的会话场景下，也可能会大大降低。尽管会话模型能够捕捉物品间的时序关系，但它们却受到数据稀疏的影响。除此之外，已经尝试在同一域内同时训练两个模型，称为"共享"模型，但是结果表明一个域的优化会对另一个域产生负面影响，导致优化冲突。因此，设计跨域建模来避免优化冲突和负迁移是必要的。

Impact of Each Component (RQ2)

为了研究我们提出的方法对各种组件的影响，我们分别使用Micro Video和Amazon Video数据集，并对比了ISA模块、SFA模块和GPA模块的表现。我们发现在 Micro Video和Amazon Video数据集上，GPA模块是最有效的，这表明不同领域的兴趣小组是相似的。此外，当删除用于融合局部和全局会话模式的SFA模块时，性能有所下降，这说明确实存在跨越不同域的共同会话模式。最后，我们发现删除ISA模块后，由于表现下降，也存在类似的情况，即项跨越不同的域。总而言之，Group-Prototype Attention在这三种提出的关注力中最为重要。

Domain	Model	AUC	MRR	NDCG@10	WAUC
Micro Video A	w/o ISA	0.8136	0.5795	0.6826	0.8216
	w/o SFA	0.8133	0.5895	0.6904	0.8292
	w/o GPA	0.7983	0.5714	0.6762	0.8134
	w all	**0.8285**	**0.6167**	**0.7112**	**0.8435**
Micro Video B	w/o ISA	0.8059	0.5644	0.6711	0.8162
	w/o SFA	0.7939	0.5559	0.6643	0.8073
	w/o GPA	0.7996	0.5631	0.6701	0.8147
	w all	**0.8094**	**0.5756**	**0.6797**	**0.8215**
Amazon Video Games	w/o ISA	0.6195	0.4437	0.5743	0.6352
	w/o SFA	0.642	0.4426	0.5735	0.6523
	w/o GPA	0.6195	0.4198	0.5549	0.6288
	w all	**0.6559**	**0.4755**	**0.5986**	**0.6686**
Amazon Toys	w/o ISA	0.6499	0.5497	0.6548	0.6516
	w/o SFA	0.6502	0.6126	0.7021	0.6583
	w/o GPA	0.6546	0.6183	0.7074	0.6606
	w all	**0.6712**	**0.6385**	**0.7221**	**0.6788**

原文《Mixed Attention Network for Cross-domain Sequential Recommendation》