PINNERFORMER：在Pinterest的用户表现的序列建模

Nikil Pancha
npancha@pinterest.com
Pinterest, Inc.
美国旧金山

Andrew Zhai
andrew@pinterest.com
Pinterest, Inc.
美国旧金山

Jure Leskovec
jure@cs.stanford.edu
美国帕洛阿尔托的
斯坦福大学。

Charles Rosenberg
crosenberg@pinterest.com
Pinterest, Inc.
美国旧金山

ABSTRACT

在过去的几年里，序列模型在个性化推荐系统中越来越受欢迎。这些方法传统上是将用户在网站上的行动作为一个序列来预测用户的下一步行动。虽然理论上很简单，但这些模型在生产中的部署是相当具有挑战性的，通常需要流媒体基础设施来反映最新的用户活动，并有可能管理易变的数据来为用户的隐藏状态进行编码。在这里，我们介绍了PINNERFORMER，这是一个经过训练的用户表征，可以使用用户最近行为的顺序模型来预测用户未来的长期参与。与之前的方法不同，我们通过新的密集的所有行动损失来适应我们的建模，对长期的未来行动而不是下一个行动的预测进行建模。我们表明，通过这样做，我们大大缩小了每天产生一次的批处理用户嵌入和每当用户采取某种行动时产生的实时用户嵌入之间的差距。我们通过大量的内部实验和消减来描述我们的设计决策，并在A/B实验中验证了我们的方法的有效性，显示出PINNERFORMER与我们之前的用户表示法相比，在Pinterest的用户保留率和参与度上有了很大的改善。PINNERFORMER已于2021年秋季在生产中部署。

CCS概念

• **信息系统** → **推荐系统**; **个人化**.

关键字

表征学习, 多任务学习, 个性化, 识别识别系统

ACM参考格式：

Nikil Pancha, Andrew Zhai, Jure Leskovec, and Charles Rosenberg.2022.PINNERFORMER：Pinterest的用户代表的序列建模。在*第28届ACM SIGKDD知识发现和数据挖掘会议（KDD '22）论文集中，2022年8月14-18日，美国华盛顿特区。*ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539156

KDD '22，2022年8月14-18日，美国华盛顿特区。
Acm isbn 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539156

1 简介

每月有超过4亿用户使用Pinterest，从我们数十亿的图钉内容库中发现创意和灵感。一个图钉从一张图片开始，通常包括文字、一个网络链接，以及一个将个人图钉与用户策划的图钉集连接起来的板块。灵感是Pinterest的关键，主要通过我们的搜索和推荐系统来促进，允许用户通过（a）Homefeed，我们的个性化推荐产品来寻找内容、
(相关图钉，与查询图钉相关的推荐，以及
(c) 搜索，与用户文本查询相关的推荐。用户通过互动提供反馈，例如将图钉保存到板子上（Repin），点击进入基本链接，放大一个图钉（特写），隐藏不相关的内容，等等。为了实现我们的使命--为每个人带来创造他们所喜爱的生活的灵感，我们需要根据用户的兴趣和背景来个性化我们的内容，同时考虑到用户在他们的Pinterest旅程中给出的反馈；也就是说，我们需要对我们的用户有一个强有力的代表性。

学习用户嵌入（表征）已经成为一种越来越流行的改善推荐的方法。这样的嵌入已经被采用来推动行业中的排名和候选人生成，并被用来推动YouTube[6]、Google Play[26]、Airbnb搜索[8]、JD.com搜索[30]、阿里巴巴[12，18]等的个性化推荐。除了学习个性化嵌入的工作外，还有大量的工作专注于直接使用顺序信息建立排名模型[4, 18, 19, 31]，实现基于用户最近参与的个性化推荐。

用户在网站上的行为往往是有顺序的；行动可以按时间排序，这自然导致了顺序建模方法。各种方法已经被提出来，以预测基于用户历史互动序列的未来参与[3, 9, 17, 20, 21]。最近的工作已经应用了各种深度学习模型，包括递归神经网络（RNNs）和转化器来进行这种顺序推荐，并获得了有希望的结果[3, 7, 9, 10, 21, 25, 29]。顺序模型传统上专注于实时设置，旨在预测用户的下一步行动或从导致该点的所有行动中的参与。

在实践中，将现有的序列建模方法部署到大型网络规模的应用中有两个关键的挑战：（a）计算成本，和（b）基础设施的复杂性。现有的

序列建模方法大致分为两类：无状态模型和有状态模型。无状态模型可能有很高的计算成本，因为在用户的每一个动作之后都必须从头开始计算嵌入，而有状态的模型需要强大的可靠的流媒体基础设施来处理潜在的错误或数据损坏的模型对特定用户的状态[18]。

在这里，我们提出了PinnerFormer，这是一个端到端的学习型用户表征，已经在Pinter-est的生产中部署。与之前的顺序用户建模工作类似，Pinner- Former直接根据用户过去的Pin参与度来学习一个表示。我们提出了一个密集的所有行动损失，这使得我们的嵌入能够捕获用户的长期兴趣，而不是仅仅预测下一个行动。这使得我们的嵌入可以在一个离线的批量设置中计算，并大大简化了基础设施。

我们还解决了基础设施的复杂性问题，在Pinterest表现为：有几十个排名模型可以从个性化中受益，但为每个模型开发一个定制的解决方案是不可扩展的。与其为每个模型制作一个用户嵌入（这将增加复杂性），我们选择投资于开发一个高质量的用户嵌入，可以用于许多下游任务。尽管在某些情况下，特定任务的性能可能会被牺牲掉，但复杂性的权衡使得共享嵌入对大多数的使用情况都是有利的。

我们在离线以及在线A/B实验中评估了PinnerFormer。在离线实验中，我们发现这个训练目标几乎将日常推断的模型和实时推断的模型之间的性能差距减半，并且比其他方法更能反映用户的长期兴趣。然后，我们展示了PinnerFormer作为一个特征的效用，证明了它在不同领域的多个排名模型中作为一个特征使用时，能够带来显著的在线收益。

2 设计选择

我们首先讨论了PinnerFormer的关键设计选择。

设计选择1：单个用户的单一嵌入与多个嵌入。大多数生成用户表征的方法产生一个单一的嵌入[3, 7, 9, 10, 21, 25, 29]，但有些方法是学习固定或可变数量的用户嵌入[12, 13, 17, 24]。在我们之前的用户表征PinnerSage中，我们决定允许一个可变的、潜在的大数量的
的嵌入，使该模型能够明确地代表用户的不同兴趣[17]。

尽管使用多个嵌入可以使模型更明确地捕捉用户的兴趣，并且在检索时效果很好，但在下游模型中使用时，这会导致一些问题：在训练数据中存储20多个256d float16的嵌入并不能很好地扩展，特别是当数据集可能包含数十亿行时，就像它们用于排名模型那样。另外，这也增加了模型训练和推理的成本；处理5000多个浮点数字会带来不可忽视的延迟，特别是当它们在聚合之前被转换的时候。在训练时，大的例子也会增加加载数据的时间。为了避免这些问题，当在排名模型中使用PinnerSage时，我们通常会使用一个用户嵌入的加权聚合作为最终的用户表示。因为我们

我们希望PinnerFormer能够很容易地作为一个功能来使用，我们产生了一个单一的嵌入来捕捉用户的兴趣，允许在下游模型中无痛使用。在离线评估中，我们表明我们的单一嵌入能够比PinnerSage更好地反映用户的长期兴趣，而所需的存储量却只有一小部分。

设计选择2：实时与离线推理。大多数先前关于顺序用户建模的工作都集中在实时或接近实时运行的模型上。在实践中，这至少会导致以下情况之一：

- **计算成本高：**对于用户的每一个行动，系统必须获取用户历史上的所有事件，并快速推断出一个潜在的复杂模型。
- **基础设施复杂度高：**用户的隐藏状态或嵌入可以逐步更新，但这需要一个强大的系统在任何数据损坏的情况下恢复和预热模型的状态[18]。

在Pinterest上，一个用户可能在一天内采取几十或几百次行动，因此一个每天最多更新一次用户嵌入的模型只需要相当规模的实时模型的一小部分计算资源。在离线评估中，我们证明了我们的损失表述大大减少了实时模型和日常推断模型之间的差距，在A/B体验中，我们显示PinnerFormer大大改善了下游排名模型的性能。

3 我们的方法：Pinnerformer

在这一节中，我们介绍了PinnerFormer，它从2021年秋季开始在Pinterest的生产中使用，同时描述了我们的模型（如图1所示）和它是如何部署的。

我们从一个品的语料库P = {□1, P_2, ...p_N}，其中N很大，大约有几十亿，还有一组用户U = {U_1, U_2, . .}，其中|U| > 500M。对于语料库中的每个Pin，我们有一个Pin- Sage [28] 嵌入 $_P$□∈R256，它是一个Pin $_P$□的vi-sual、文本和参与信息的集合。对于每个用户，我们有一个用户在网站上的行动序列，A_U = {□1, A_2, ., A_S}，按时间戳升序排列。在这项工作中，我们将这一行动序列限定为用户对图钉的参与，包括过去一年中图钉的保存、点击、反应和评论。基于这个假设，一个行动可以由PinSage嵌入来代表，以及一些关于该行动的元数据。在实践中，S对一个给定的用户来说可能非常大，对一些用户来说是几千或几万的数量级，所以我们用用户的M最近的行动计算其嵌入。

鉴于这些定义，我们的目标是学习一个用户代表 f: U↦→$\mathbf{R}^d$，这与一些Pin表示兼容 g: P ↦→R⊠在余弦相似性下。我们联合学习f和g，使用行动序列 A_U作为模型的唯一输入特征，并且只限制在最新的□。

在一个用户的完整行动序列中，可能有许多类型的行动，其中一些是好的（如长点击），而另一些是中性或负面的（如隐藏或短点击）。在这项工作中，我们重点学习预测*积极的*参与，我们将其定义为一个图钉的保存（"Repin"），一个图钉的特写超过10秒（"Closeup"），或一个长的点击（>10秒）到图钉的链接（"Click"）。我们只处理Homefeed上的参与

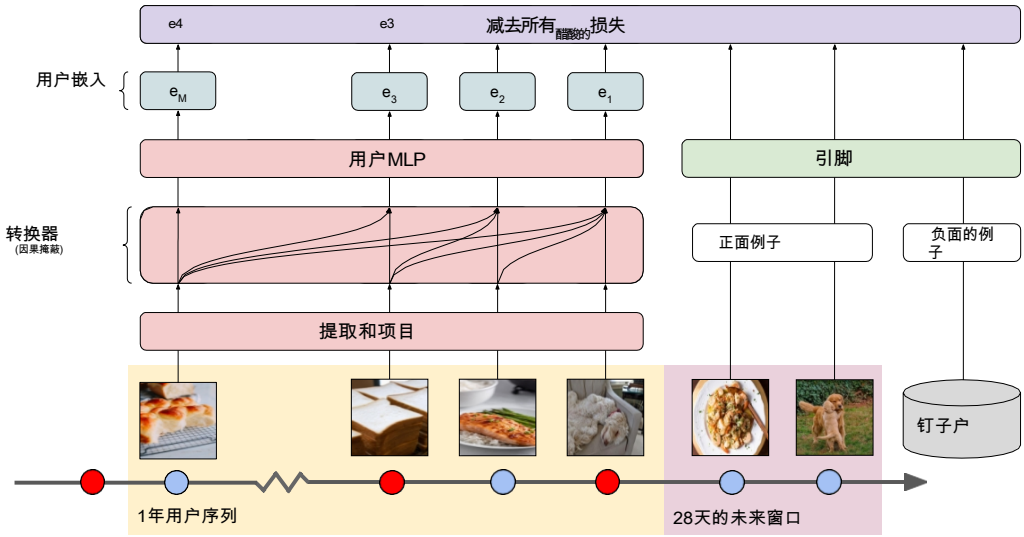

图1：PinnerFormer架构概述。特征通过一个带有因果掩码的转化器，嵌入在每个时间步骤中被返回。请注意，训练窗口（上面的28d）超过了我们未来的评估目标窗口（14d）。

在 "搜索 "或 "相关引脚 "等表面上，查询提供了大量的上下文，而在Homefeed上，用户提供了主要的上下文。

我们的主要目标是学习一个模型，该模型能够预测用户在生成其嵌入后的14天时间窗口内的*积极的*未来参与，而不是传统的序列建模任务，即嵌入将只预测下一个采取的行动。换句话说，我们的目标是学习嵌入$□_i$和$□_i$，以便如果$d\ (_{uk},\ _{pi})$ <$d\ (_{uk},\ p_j)$，那么在$_{uk}$产生后的14天内，$□_i$比p_j更可能被$_{uk}$所代表的用户积极地参与。我们选择这个14天的范围是为了可操作性，并假设用户在两周内的行为足以代表一个用户的长期兴趣。图1展示了PINNERFORMER的架构，下面我们将更详细地阐述每个组件。

3.1 特征编码

对于用户序列中的每个动作，我们有一个PinSage嵌入（256维）[28]和元数据特征：动作类型、表面、时间戳和动作时间。我们使用小型的、可学习的嵌入表来编码动作类型和表面，这是我们的两个分类特征，并放弃这两个特征中的任何一个的词汇的序列元素。我们用一个标量值来编码动作持续时间，即log(*duration*)。

为了表示一个行动发生的时间，除了原始的绝对时间戳之外，我们还使用了2个派生值：自用户最近一次行动以来的时间，以及行动之间的时间间隔。对于这些时间特征中的每一个，我们都遵循使用不同周期的正弦和余弦变换对时间进行编码的常见做法，其方式类似于Time2vec[11]，但有□固定周期，而不是学习周期，并且有对数变换的时间，而不是一个线性的。这产生了2□+1个特征（2□来自时间戳的周期性变化）。

所有的特征都串联成一个向量，形成一个维度为$□_{in}$的输入向量。与行动$_A$□相对应的表征被表示为$_a$□∈^{R}⊠in。

3.2 模型结构

在PINNERFORMER中，我们使用转化器模型架构[22]对用户行为的序列进行建模。我们选择使用PreNorm残差连接，在每个区块前应用层归一化，因为这种方法已经被证明可以提高训练的稳定性[16, 23]。我们首先构建输入矩阵$A = (_{aT}$- - $□_{T-M+1})^⊤$ ∈^{R}⊠×□，使用导致行动$□_{T+1}$的M行动作为用户的序列。然后，我们将这些投射到变压器的隐藏维度，添加一个完全可学习的位置编码，并应用一个由交替前馈网络（FFN）和多头自我注意（MHSA）块组成的标准变压器。变换器在每个位置的输出通过一个小的MLP和$□_2$归一化，从而得到一组嵌入物
$E = (e_1$ - - $_{eM})^⊤$∈^{S}x⊠，其中D是最终嵌入的二维空间。

为了表示Pins，我们学习了一个MLP，它只接受PinSage作为输入，并$□_2$，将输出嵌入归一化。我们发现，使用$□_2$归一化的嵌入来表示用户和Pin，可以带来稳定的训练，而不会牺牲offiine的性能。

3.3 度量衡学习

为了训练我们的表征，我们需要一对{(□1, □1), . . ., $_{(}$□□, □□)}，由用户嵌入和目标引脚嵌入组成，其中用户和引脚都可以重复。在这项工作中，我们选择不使用明确的负面例子（也就是说，我们没有损失项为

[1] 详情见可重复性材料的A.1节。

[2] 重现性材料的A.2节中提供了更精确的方程式。

负面的参与，如隐藏）。在设计我们的模型时，有几个考虑因素：

(a) 我们如何选择这些配对？
(b) 对于一个给定的（□□, □□）对，我们如何选择负面的例子？
(c) 给定一个(□□, □□)对和一组负面的例子，我们如何计算损失？

我们首先描述（b）和（c），然后在第3.4节阐述（a）。

*3.3.1 负面选择。*我们考虑两个负面例子的来源：批内负面例子和随机负面例子。当为一个给定的用户选择批次内的负面例子时，我们选择批次内的所有正面例子作为负面例子，掩盖了对该用户有正面参与的图钉。这种方法既高效又简单，但如果天真地执行，可能会导致受欢迎的图钉被降级，因为有吸引力的图钉比没有吸引力的图钉更有可能出现在负面信息中。批量否定的另一个缺点是，否定例子的分布与用于检索的图钉的真实基础分布不同，导致训练和服务之间的差异。负面例子的第二个来源是那些从我们可能面对Homefeed的所有Pin的语料库中统一采样的例子，但是单独使用这些例子会导致模型崩溃，因为负面例子可能太容易了。我们考虑的第三个方案是将随机和批处理的底片结合起来，通过将批处理和随机底片库合并成一个单一的底片库，其中包含批处理和随机底片的组合，来利用两者的独特特性[26]。

在实践中，更大的负数池可以提高学习到的嵌入的质量，因此我们从训练中使用的所有GPU中收集负数例子，选择可以舒适地装入GPU内存的最大可能的池。

*3.3.2 损失函数。*在选择了负面样本的来源之后，我们可以针对给定的一对用户和正面嵌入（□□）产生一组负面嵌入{□1, . . .□□}，用于给定的一对用户和正面嵌入（□□, □□）。我们为每对用户计算 损失，然后计算加权平均数，这样，在给定的 GPU 上的每一个用户都被赋予相同的权重。

我们发现效果最好的损失函数是带有logQ修正的采样softmax[2, 27]，我们根据给定的负数在批次中出现的概率对每个logit进行修正。我们还学习了一个温度$\tau \in [0.01, \infty]$，对稳定性的下限进行约束。如果我们让$s(u, p) = \square\square, \square\rangle/\tau$，一个没有样本概率校正的采样软最大损失将被定义如下：

$$L(ui, pi) = -\log \frac{(\square\square, \square\square)}{\square\square(\square\square, \square\square) + \square=, \square\square)} \quad (1)$$

当底片不是均匀分布时，A修正项
□□ (□) = □ (Pin □ in batch | User □□ in batch) 应该被应用来修正采样偏差，其中□可能是一个正面或负面的例子。然后，单对样本概率校正的softmax损失定义如下：

$$L(ui, pi) = -\log \frac{es(ui, ui) - \log(Qi(pi))}{es(ui, ui) - \log(Qi(pi)) + \square= es(ui, nj) - \log(Qi(nj))} \quad (2)$$

为了简单起见，我们用一个计数-最小的草图[5]对□进行近似。

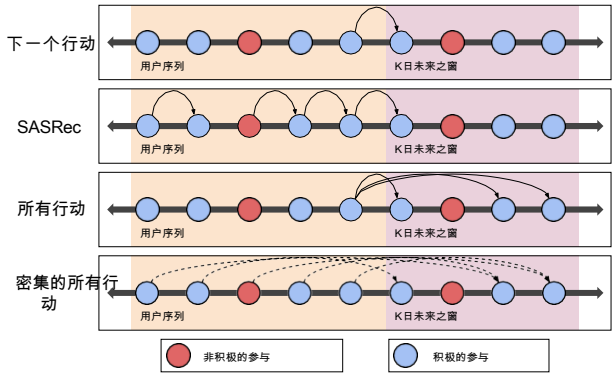

图2：四个探索的训练目标。蓝色圆圈代表对应于被认为是积极的行动的嵌入，而红色圆圈代表对应于被认为是非积极的行动的嵌入（但不一定是明确的消极的）。密集的所有行动损失中的确切配对被抽样，所以这只是一个潜在的具体化。请注意，我们并不试图预测非正面的例子。

3.4 培训目标

鉴于我们的损失函数，我们要解决如何选择配对（□□, □□）的问题。我们的模型应该能够预测三种形式的正面参与：转发，特写，和点击。每种行为都有价值，但我们没有像多任务学习文献[1, 14]中常见的那样学习特定的任务头，而是选择以多任务的方式训练一个嵌入，直接学习一个可以有效检索不同类型的积极参与的嵌入。在我们的损失计算函数中，我们没有明确地对不同的参与度进行不同的加权。我们考虑的四个训练目标描述如下，并在图2中描述。

*3.4.1 下一步行动预测。*序列建模任务的天真目标是下一个行动预测，在这个任务中，我们预测□$_{T+1}$，给定用户序列{□□, A_{T-1}, . ., $A_{T-\square+1}$}（如果$A_{\square+1}$是一个积极的约定）。这个目标对于实时序列模型来说是很直观的，因为在在线环境中，□□总是用户最近采取的行动。SASRec[10]扩展了这个简单的训练目标，旨在预测模型中每一步的下一个动作，而不是只预测最近的正向动作。行动。我们在实验中对此稍作修改，只允许

积极行动，为模型的损失做出贡献。

与这些传统的目标不同，我们的目的不是要预测一个

相反，我们每天推断用户嵌入，旨在捕捉用户的长期兴趣。为了做到这一点，我们引入了两个备用的训练目标。

*3.4.2 所有行动预测。*基于我们并不只希望预测用户的下一个行动，我们构建了一个天真的训练目标，预测用户将采取的*所有*行动
在接下来的□天里，使用□1，最终的用户嵌入。[3] 假设

用户有积极的参与行动□+3，□+8，和

[3] □可能不等于14：我们将评价目标固定为14天的窗口，但培训
如4.2.3所示，在同一窗口上的 "自动 "和 "自动 "可能无法优化性能。

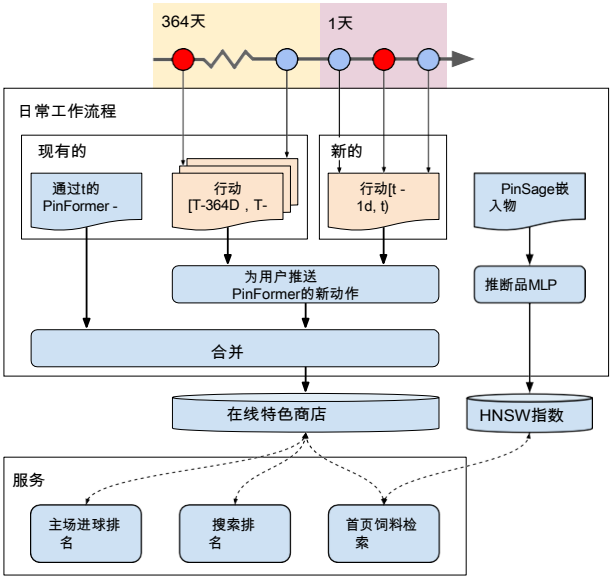

图3：PinnerFormer是逐步推断出来的。我们只计算过去一天在Pinter-est上参与活动的用户的嵌入，然后将新的嵌入与前一组嵌入合并，如果新的嵌入缺失，则退回到旧的嵌入。

□□+12，都在□天的窗口内□，我们的目标是预测所有3个行动：□□ +3, □□ +8, □□ +12，我们的目标是预测所有3个行动：从用户序列中找出
$\{_aT, _aT_{-1}, \ldots, a_{T-\square+1}\}$。这个目标迫使模型学习更长期的兴趣，而不是只关注用户的下一个行动，这应该会减少日常推理带来的呆板的影响。为了便于计算，我们在这个K天的时间窗口中对每个用户的32个行动进行随机抽样。

*3.4.3 密集的所有行动预测。*为了进一步改善每个批次提供的信号，我们从SASRec[10]中获得灵感，修改了所有行动预测的目标。[1]我们不是只用K天的窗口来预测行动，而是选择一组随机指数{□□}，对于每个$_{es}$□，旨在从接下来K天的所有积极行动的集合中随机选择一个积极行动。然后，在推理时，我们使用□$_1$作为我们的最终用户表示。为了确保这种方法能从数据的排序中学习，我们对转化器的自我注意区块应用了因果屏蔽，因此每个行动只能注意到过去或现在的行动，而不是未来的行动。我们观察到这种掩蔽大大改善了模型在这项任务上的表现。为了减少内存的使用，我们的目标不是预测所有的积极行动，而是只预测每个$_{es}$□的一个积极行动。

3.5 数据集设计

我们利用压缩的格式来存储训练序列。我们的一个观察是，给定一个用户的时间线、

[4] 观察一下，在没有任何子采样的情况下，假设批量大小为128，32个采样阳性，最大序列长度M=256，这可能产生多达128 - 256 - 32 = 1048576对，用于每个GPU的softmax计算。

许多独立的用户序列和正面例子可以被构筑。给定一个完整的用户序列$_A$□=$\{$□1, . $A_S\}$，以及一些最大的序列长度□，我们可以构建长度正好等于S-1的训练例子（假设所有的行为都是正数）。例如，序列$_{\{}$□5, . $A5, A_{5+M-1}\}$与正的约定$\{$□5$_{+M}, A_{7+M}\}$可以从完整的时间线$_A$□中提取。存储这些数据的一个潜在方法是提前将所有长度为M（或更小）的相关序列具体化，并为每个序列提供一套相应的未来积极约定。在体验不同的采样策略时，这就遇到了问题，因为调整参数需要训练数据的再生，这是一个缓慢的过程。为了提高生产率，我们将每个用户的序列作为单一行存储在我们的数据集中，并在训练期间对例子进行采样。这有一个明显的好处，即允许在训练期间进行自定义采样，但代价是减少训练数据的洗牌。

具体来说，有几个参数我们已经使用了调整
这个策略，所有这些都会对模型的整体性能产生重大影响：

- 最大序列长度
- 从一个用户的时间轴上取样的可能的用户序列的百分比
- 每个用户的最大取样序列数
- 作为每个序列的标签的最大数量的阳性例子样本

3.6 服务模式

由于我们专注于推断Pinner- FORMER的离线、批量设置，我们在一个每日递增的工作流程中推断模型，如图3所示。这个工作流程为过去一天中参与过任何Pin的用户生成新的嵌入，将它们与前一天的嵌入合并，然后将它们上传到一个键值特征存储，供在线使用。因为我们只为过去一天参与活动的用户生成新的嵌入，并离线运行推理（没有延迟限制），我们能够使用比其他方式更大的模型，这增加了我们的嵌入可以捕获的信息。在输入特征出现任何损坏的情况下（例如，由于记录错误），我们可以很容易地对所有嵌入在损坏后被更新的用户进行推理，第二天的数据将是正确的，假设上游的数据已经被修复。

针式嵌入的计算成本很低，只需要对现有的特征进行一个小的MLP转换，所以我们每天从头开始生成，然后编译一个HNSW[15]图，可以使用保存在特征库中的用户嵌入进行在线查询。

4 实验和结果

在这里，我们首先将PINNERFORMER与基线进行比较，进行消融，并通过offiine实验探索实时推理和日常推理之间的性能差距。然后，我们在A/B实验中展示了比PinnerSage的相当大的改进。

表 1 ： PinnerSage （ PS ）与 PinnerFormer 的对比。PinnerFormer在我们的参与度评估中优于PinnerSage，即使PinnerSage是通过将用户嵌入设置为最接近真实的正面嵌入的集群来评估的。更高的兴趣熵表明每个用户检索到的结果更加多样化，而更高的覆盖率表明在所有用户中检索到的结果更加独特。

模型	R@10	兴趣熵@50	P90覆盖率@10
PS (5个群组)	0.026	1.69	0.130
PS (20个群组)	0.046	2.10	0.133
PINNERFORMER	0.229	1.97	0.042

表2：实时与离线批处理推理。从实时推理到批量推理，在SASRec目标上进行训练时，召回率@10下降了13.9%，但在使用密集的所有行动目标（PinnerFormer）时，召回率仅下降了8.3%。

推断频率	模型	R@10	P90 覆盖率@10
	辽宁沈阳	0.198	0.048
一次	PINNERFORMER	0.229	0.042
每日	证券交易所	0.216	0.052
	PINNERFORMER	0.243	0.043
实时性	医学专家	0.251	0.057
	PINNERFORMER	0.264	0.045

4.1 离线评估指标

我们用于评估的主要指标是Recall@10。我们选择训练结束后的2周时间进行评估，并在与训练无关的用户集中进行评估。假设训练数据集在时间□结束，我们在时间□为评估集中的所有用户计算嵌入，然后衡量时间□的嵌入从时间□到□+14□，从100万个随机Pin的索引中检索出用户将参与的*所有*Pin的程度。假设我们有一组用户，U，每个用户有一组积极参与的P□，和一个100万个Pin的随机语料库N，我们计算Recall@k（R@k）如下：

$$\text{回顾@k}(U) = \frac{1}{|P_U|} \sum_{P \in P_U} 1\{|\{\square \in N \mid d(U,P) \geq d(U,N)\}| < k\}$$

$$\text{召回@k} = \frac{1}{|U|} \sum_{U \in U} \square \text{回顾@k}(\square)$$

在这里，用户和引脚之间的距离是由用户的嵌入和引脚的嵌入之间的欧几里得距离定义的。我们还观察到两种多样性的衡量标准：(a)与100万个图钉的索引中检索到的前50个结果相关的兴趣（大约350个独特的图钉主题）的熵（"Inter- est Entropy@50"），以及(b)在一组用户中，100万个图钉的索引占前10个检索结果的90%的比例（"P90 Coverage@10"）。前者衡量的是某一特定用户的检索结果的多样性，而后者则代表了所有用户的检索结果的全球多样性。两者都是有用的观察方法；一个简单的基线可以记忆与用户无关的流行度，通过指标（a）可以有良好的表现、但（b）会显示一个非常接近0.0的数值。

4.2 离线结果

在这一节中，我们首先将PINNERFORMER与基线进行比较，然后研究模型的哪些方面导致了良好的性能。

*4.2.1 与基线的比较。*在我们的离线评估中，我们与PinnerSage[17]的基线进行了比较，我们之前的多嵌入用户表示，基于神谕评估来测量召回率，以获得一个上限。具体来说，给定一个固定的截止点c，和一个给定的正数，我们选择用户的表述作为最接近正数的PinnerSage嵌入中的一个。我们相信这为那些c嵌入物预测参与度的能力建立了一个近似的上限。

为了计算多样性指标，我们不采用神谕的方法，而是用轮流混合的方式对结果进行排序：给定一组用户嵌入（按权重排序），每个嵌入都有一些检索到的结果，我们从第一个用户嵌入中提取第一个结果，从第二个用户嵌入中提取第二个结果，以此类推，在每个嵌入检索到一个结果后返回到第一个嵌入。

在表1中，我们显示了PINNERFORMER和PinnerSage之间的比较，如上所述的评估。即使在评估PinnerSage与甲骨文的前5个或20个集群时，我们看到单一的PINNERFORMER嵌入在检索用户在14天内可能参与的内容方面优于PinnerSage。增加用于检索结果的聚类的数量会导致在特定的情况下检索到的结果更加多样化。

当使用足够多的集群时。我们还看到，PinnerSage从索引中检索出更多独特的候选者，而PINNERFORMER的某些变体达到了相当的水平。

用它的嵌入来检索PinnerSage优于PINNERFORMER的一个方面。

独特的候选人，同时保持较高的参与度评价指标，如表4所示。

*4.2.2 日常推理与实时推理。*为了量化推理频率降低后的性能下降，我们比较了两个仅在训练目标上有差异的模型：

- 使用SASRec训练目标训练的模型，它直接预测用户将采取的下一个行动[10]。我们用采样的softmax损失代替二元交叉熵，并将$\square_1$上的损失与其他位置上的损失分开加权，因为我们发现这样可以提高性能。
- PINNERFORMER，使用密集的所有行动预测目标进行训练，窗口为28天。

然后，我们在评估窗口(□, □ + 14□)的三个不同频率上评估这些数据：

- **一次：**我们使用在时间t为用户预测的单一嵌入来预测用户在14天窗口（□, t + 14□）内的所有积极行动。
- **每天：**我们每天更新用户的嵌入，根据在x-1□计算的嵌入，预测他们在(□, x+1□)的行动，其中，$x \in \{\square, t + 1d, .., t + 13\square\}$。这一天的差距说明了一个动作在离线日志中可用，与它被上传到特征库之间的延迟。

[5] 见可重复性材料的A.4.4节，以进一步说明这一变化的理由。

表3：各种训练目标的比较。密集的所有行动目标使召回率最大化@10，28天的未来窗口比14天的窗口表现明显更好。

培训目标	召回@10	P90覆盖率@10
下一步行动	0.186	0.050
SASRec (Softmax)	0.198	0.048
所有行动 (28d)	0.224	0.028
密集的所有行动 (14d)	0.223	0.043
密集的所有行动 (28d)	0.229	0.042

- **实时性：**我们在每次行动后更新嵌入；也就是说，我们使用用户在正面行动之前的M行动序列来预测该正面行动（对于$(\square, t + 14\square]$中的所有正面行动）。

注意每日和实时设置与我们的主要评估不同。在这里，鉴于用户在某一时间点的嵌入，我们衡量我们预测用户将采取的*具体*行动的能力，而我们的主要评估衡量嵌入捕捉用户长期兴趣的能力。

一个实时模型在生产中是不实用的，因为它将大大增加推理成本，而不是批量模型：一些用户可能每天采取几十或几百个行动，这相当于离线模型的许多倍的成本，即使使用较短的序列。我们希望这个实时基线能比一个离线的、每天计算的模型表现得更好，但它有助于量化避免实时设置的机会成本。

在表2中，我们还注意到，PinnerFormer的性能随着推理频率的增加而增加，从评估开始时的一次，到每天一次，再到实时。令人惊讶的是，即使在实时情况下，PinnerFormer的性能也超过了只预测下一个参与项目的训练模型。

这个实验还提供了证据，证明密集的所有行动预测目标具有降低模型对短期变化的敏感性的预期效果，反而能学习到更稳定的用户兴趣：当从实时推理到每日推理，以及每日推理到只推理一次时，在密集的所有行动目标上训练模型时，其性能损失（-8.3%）比下一个行动预测任务（-13.9%）小。

在实时性能和日常推理性能之间仍有不小的差距，但考虑到比我们的PinnerSage基线的改进，以及在实时推理PinnerFormer的高成本和低复杂性，我们认为这是一个可以接受的权衡。

*4.2.3 训练目标的选择。*在表3中，我们观察到，在训练中只有预测下一个动作的训练会导致较低的Re-call@10，但检索到的索引覆盖率较高。较低的Recall@10可以解释为所有的行动预测任务都比下一个行动预测更符合评估目标。我们认为，我们观察到下一个行动预测的索引覆盖率较高，是因为在较长的时间范围内预测行动比只预测下一个行动更难，所以学到的Pin嵌入可能更偏向于检索更普遍参与的内容，而不是检索与最近行动直接相关的内容。我们还观察到，在28天的未来窗口进行所有行动预测的训练，比14天的窗口产生更好的结果、

表4：负数池和样本概率校正（SPC）的影响。SPC明显提高了召回率@10，但代价是降低了全局结果的多样性。

SPC	负数来源	P90覆盖率@10	召回@10
N	随机	0.002	0.136
N	批量生产	0.163	0.071
N	混合的	0.083	0.138
Y	随机	0.001	0.139
Y	批量生产	0.119	0.167
Y	混合的	0.042	0.229

甚至当评估被固定在14天的窗口时也是如此。我们认为这可以解释为每个用户序列有更多标签的结果，这可以提高训练效率。

密集的所有行动损失在Recall@10和全局多样性方面优于所有行动预测。这两种损失的关键区别在于，在所有行动损失中，来自一个用户的所有正面例子的梯度将通过同一个用户嵌入进行反向传播，从而产生更大的平均效应，相比之下，密集所有行动损失中的梯度都通过不同的转化器输出，在进入转化器后才被平均到一起。

我们还尝试将基于不同训练目标计算的损失加在一起，但这样的配置并没有超过任何单目标模型。

*4.2.4 采样的Softmax。*在表4中，我们比较了我们的softmax损失的不同设置的性能。在所有情况下，我们都看到，批量内否定词的存在增加了检索结果的多样性，但导致召回率@10低于混合否定词。当我们使用随机否定词训练模型时，该模型似乎对所有用户的检索结果都非常相似；当检索100000个用户的100万个Pin中的10个时，只有1000个Pin可以占到检索结果的90%。这似乎表明该模型未能学习到有关用户兴趣的细节，因为它为大多数用户检索了非常相似的内容。

总的来说，我们看到样本概率校正并没有在随机底片上增加召回率@10，这是预期的：在这种情况下，所有底片在批次中的概率应该是相等的，因为采样是无偏的。当包括批次内底片时（无论是单独，还是与随机底片结合），启用样本概率校正会增加召回率，同时减少全局多样性。鉴于批内否定和混合否定在Recall@10方面的巨大差异，我们选择使用带有样本概率修正的混合否定作为我们的损失函数，尽管混合否定带来的复杂性略高。

*4.2.5 多任务学习。*在这里，我们衡量了单任务和多任务学习之间的性能差异。对于3种积极的行动类型中的每一种，我们训练一个模型来预测一个单一的行动类型（10s特写，10s点击，Repin），然后训练一个模型来预测这3种行动类型中的任何一种。在表5中，我们看到了结果：当我们对一个特定的动作类型进行训练时，当我们只把该动作类型作为一个正面标签时，我们的召回率@10最大化。当我们对所有3种动作类型进行训练时，我们最大限度地提高了所有的Recall@10，但在每个单独的任务上表现稍差。

表5：不同动作类型与训练目标动作类型的表现。每个任务中表现最好的目标是粗体字，第二好的是下划线。尽管多任务的表现不如单任务学习，但它在整体的RecallI@10上的表现比任何单独的模型都好，在所有其他任务上的表现是第二好的。

培训目标	评价任务			
	10s特写	10s Click	报道	全部
10秒特写	**0.27**	0.02	0.09	0.17
10s Click	0.01	**0.49**	0.01	0.12
报道	0.15	0.03	**0.17**	0.13
多任务	0.23	0.28	0.13	**0.23**

表6：特征消减，包括所有的，但每次只有一个特征。删除任何特征都会导致召回率@10的下降。

遗漏的特征	P90覆盖率@10	召回@10
品圣	0.0005	0.142
时间戳	0.050	0.210
表面	0.040	0.224
行动类型	0.042	0.226
时间	0.042	0.226
位置编码	0.041	0.228
无	0.042	0.229

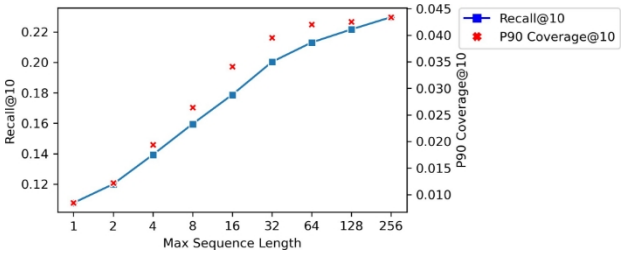

图4：最大序列长度与召回率和覆盖率。较长的序列会带来较好的性能，但回报率却很低。

比起单任务设置，多任务性能是次要的。对于每个特定任务的评估，多任务性能是次要的，所以我们选择多任务训练目标作为每个目标之间的权衡，确保最终的嵌入不会强烈地偏向一个特定的任务。

*4.2.6 特征消减。*在表6中，我们看到每个特征对最终模型性能的影响。对最终嵌入有明显贡献的两个特征是时间戳和PinSage嵌入。如果没有PinSage，模型就无法理解用户行为背后的内容，这体现在低召回率@10和非常低的全局多样性上，表明我们为所有用户检索到了一组高度相似的结果。我们看到删除每个特征的负面影响，所以我们选择在PINNERFORMER中包含所有特征。

表7：在线A/B实验结果，用PinnerFormer代替PinnerSage作为我们Homefeed排名模型的一个特征。我们看到整个网站的指标都有改善。

公制	电梯	公制	电梯
花费的时间	+1%	家政服务转帖	+7.5%
DAU	+0.4%	首页的点击率	+1%
WAU	+0.12%	家居用品特写	+6%

*4.2./ 序列长度。*图4显示了序列长度对模型性能的影响。当序列长度翻倍到32左右时，我们看到召回率@10和全局多样性都有近乎恒定的增加，但是随着序列长度的增加，我们看到回报率在减少。在这项工作中，我们没有研究长度超过256的序列，因为这样的模型需要在批量大小或训练资源方面做出牺牲。较小的批次规模使得与较短的序列模型的比较变得不可能，因为用于学习嵌入的负数池发生了变化，需要更长的时间来训练。使用更多的机器（512序列长度的16个GPU/2个机器，1024序列的32个GPU/4个机器）可以训练更长的序列模型，但减少了可能的并行训练运行的数量。由于能够并行训练更少的模型，调整建模的决定变得更慢，所以对于PINNERFORMER，我们在最终模型中选择256的序列长度。

4.3 A/B实验的排名

我们进行了几个A/B实验，将其作为排名模型的一个特征，以更好地了解PINNERFORMER的在线表现。

*4.3.1 Homefeed。*我们的第一个比较是在Pinterest的Homefeed排名模型中，该模型帮助决定内容在Homefeed上显示给用户的顺序。此前，这个模型使用用户的顶级kPinnerSage嵌入物的加权平均值作为特征。在实验的启用组中，我们用单一的PINNERFORMER嵌入来取代PinnerSage的这种加权。为了进行公平的比较，控制组和启用组的排名模型都是在同一日期范围的数据上训练的。

表7显示了这个实验的主要结果。PINNERFORMER极大地提高了Homefeed的参与度，并导致了每日活跃用户（DAU）和每周活跃用户（WAU）的增加。在实验结束后的几个月内，我们没有看到改进的退步。

*4.3.2 广告。*为了验证这种嵌入在它被明确训练的用例之外作为一个特征是否有用，我们还进行了一个A/B实验，将PINNERFORMER加入到广告排名模型中（不取代PinnerSage）。每一个主要的表面（Homefeed、Related Pins和Search）都有一个单独的模型，专门用于确定我们向用户展示广告的顺序，所以我们对它们中的每一个进行了独立的实验。总的来说，我们看到每个表面上的广告参与度都有明显的提高，从点击率（CTR）和长点击率（gCTR）来看，如表8所示。

表8：在线A/B实验结果，将Pinner- Former作为一个特征加入到Ads排名模型。每个表面都从PinnerFormer中明显受益。

度量衡相关引脚	搜索主页		
CTR	+7.1%	+7.3%	+10.0%
gCTR	+6.9%	+5.2%	+10.1%

5 结论

在这项工作中，我们提出了PINNERFORMER，这是一个单一的、端到端的学习型嵌入，旨在推断出一个offiine环境，并在多天的时间范围内捕捉用户的兴趣。

与其他专注于根据用户过去的行为进行建模的工作相比，我们并不直接专注于预测用户的下一次参与，而是应用一个新的损失函数来捕捉用户在数天内的兴趣。我们表明这种训练目标减少了实时推断的模型和每天推断一次的模型之间的性能差距。我们还提出了一些去尾巴化的实验，以显示我们模型的每个组成部分对整体性能的贡献，证明了多任务学习和抽样softmax的有效性。

在未来，我们计划更彻底地调查PINNERFORMER作为候选生成器的表现，并将Pin参与以外的行动作为用户行动序列的元素，帮助建立一个更全面的用户代表。

鸣谢

作者要感谢Jay Adams, Dhruvil Badani, Kofi Boakye, Haoyu Chen, Yi-ping Hsu, Haomiao Li, Yang Liu, Cosmin Negruseri, Yan Sun and Jiajing Xu，他们在整个项目中为我们提供了帮助或支持。

参考文献

[1] Sean Bell, Yiqun Liu, Sami Alsheikh, Yina Tang, Edward Pizzi, M. Henning, Karun Singh, Omkar Parkhi, and Fedor Borisyuk.2020.*GrokNet：统一计算机视觉模型主干和嵌入的商业*。

[2] Yoshua Bengio和Jean-Sébastien Senécal。2008.自适应重要性抽样以加速神经概率语言模型的训练。*IEEE Transactions on Neural Networks* 19, 4 (2008).

[3] 陈敏敏，Alex Beutel，Paul Covington，Sagar Jain，Francois Belletti，和Ed Chi。2021. arXiv:1812.02353 [cs.LG]，REINFORCE推荐系统的Top-K非政策修正。

[4] 陈奇伟，赵欢，李伟，黄培培，和欧文武。2019.阿里巴巴电子商务推荐的行为序列转化器.In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional 稀疏的数据*。

[5] Graham Cormode 和 Shan Muthukrishnan.2005.一个改进的数据流摘要：count-min草图及其应用。*Journal of Algorithms* 55, 1 (2005).

[6] Paul Covington, Jay Adams, and Emre Sargin.2016.用于YouTube推荐的深度神经网络。在*ACM国际推荐者会议上系统（RecSys '16）*。https://doi.org/10.1145/2959100.2959190

[7] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler.2017.基于用户的顺序性递归神经网络推荐。在*第十一届ACM推荐系统会议（RecSys '17）上*。

[8] Mihajlo Grbovic and Haibin Cheng.2018.在Airbnb使用嵌入技术进行搜索排名的实时个性化。在*第24届ACM SIGKDD知识发现与数据挖掘国际会议（KDD '18）论文集*。

[9] Balázs Hidasi和Alexandros Karatzoglou。2018.基于会话的推荐的具有Top-k收益的循环神经网络。在*第27届ACM信息和知识管理国际会议论文集*。

[10] Wang-Cheng Kang and Julian McAuley.2018.自我关注的顺序推荐。In *IEEE International Conference on Data Mining (ICDM)*.

[11] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker.2019.Time2vec：Learning a vector representation of time. *arXiv preprint arXiv:1907.05321* (2019).

[12] 李超，刘志远，吴萌萌，徐玉琪，赵欢，黄皮皮，康国良，陈奇伟，李伟和李迪伦。2019.带有动态路由的多兴趣网络在天猫的推荐。In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*.

[13] 刘宁浩，谭巧玉，李跃宁，杨红霞，周敬仁，和胡霞。2019.Is a single Vector Enough?探索网络嵌入的节点多义性。In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

[14] 刘晓东，高建峰，何晓东，邓丽，杜凯文，和王亦一。2015.使用多任务深度神经网络的表征学习用于语义分类和信息检索。(2015).

[15] Yu A Malkov和Dmitry A Yashunin。2018.使用分层可导航小世界图的高效和稳健的近似近邻搜索。*IEEE模式分析和机器智能交易* 42 (2018).

[16] Toan Q. Nguyen和Julian Salazar。2019.Transformers without Tears：改进，自我关注的正常化。abs/1910.05895（2019）。arXiv:1910.05895

[17] Aditya Pal, Chantat Eksombatchai, Yitong Zhou, Bo Zhao, Charles Rosenberg, and Jure Leskovec.2020.PinnerSage.*第26届ACM SIGKDD知识发现与数据挖掘国际会议（2020）论文集*。

[18] 皮琦，边伟杰，周国瑞，朱小强，盖坤。2019.关于长序列用户行为建模预测点击率的实践。In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

[19] 皮琦，朱小强，周国瑞，张玉静，王哲，任乐建，范莹，盖坤。2020.基于搜索的用户兴趣模型与终生潜在行为数据的点击率预测.*CoRR* abs/2006.05639 (2020). arXiv:2006.05639

[20] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme.2010.用于下一篮子推荐的个性化马尔科夫链的因子化。在 *ings of the 19th International Conference on World Wide Web (WWW '10)*.

[21] 孙飞，刘军，吴健，裴长华，肖林，欧文武，姜鹏。2019.BERT4Rec：带有双向编码器Rep- resentations from Transformer的顺序推荐。在*第28届ACM国际信息和知识管理会议（CIKM '19）论文集*。

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin.2017.注意力是你需要的所有。*神经信息处理系统的进展》*（2017）。

[23] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao.2019.Learning Deep Transformer Models for Machine Translation. arXiv:1906.01787 [cs.CL]

[24] Jason Weston, Ron Weiss, and Hector Yee.2013.通过嵌入多个用户兴趣的非线性潜因分析。在*ACM国际会议上，Recommender Systems（RecSys）*。

[25] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing.2017.递归推荐网络。在*第十届ACM国际网络搜索和数据挖掘会议（WSDM '17）论文集*。

[26] 杨吉，易新阳，程志远，洪立昌，李阳，王小明，徐太白，和池爱华。2020.在推荐中学习双塔神经网络的混合负采样。In *Companion Proceedings of the Web Conference*.

[27] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Ajit Kumthekar, Zhe Zhao, Li Wei, and Ed Chi（编者）.2019.*Sampling-Bias- Corrected Neural Modeling for Large Corpus Item Recommendations*.

[28] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec.2018.用于网络规模推荐系统的图卷积神经网络。*在第24届ACM SIGKDD国际知识发现与数据挖掘会议上*。

[29] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He.2019.一个用于下一个项目推荐的简单卷积生成网络。在*第十二届ACM国际会议上，网络搜索和数据挖掘（WSDM '19）*。

[30] 张晗，王松林，张康，唐志玲，蒋云江，肖云，闫伟鹏，杨文云。2020.迈向个性化和语义检索：通过嵌入学习为电子商务搜索提供端到端的解决方案。In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.美国纽约，NY。

[31] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai.2019.Deep interest evolution network for click-through rate prediction.In *Proceedings of the AAAI conference on artificial intelligence*.

A 可重复性的信息

A.1 时间戳编码

除了原始时间戳之外，我们还使用两个派生值来代表时间：序列中最新的时间戳与一个动作的时间戳之间的差值，以及序列中每两个连续动作之间的时间间隔，将最后一个动作设置为零。为了对时间戳进行编码，我们修改了Time2vec，使其使用固定的周期，并对原始时间值进行了对数变换。 具体来说，给定一个时间戳t和P周期，{□1, p_2, . . .$_{pP}$ }，我们得到2□ + 1个特征□1, . . ., $_2P$+1，方法是

$$r\ (t)\ _{2\square\text{-}1} = \cos\left(\frac{2\pi\square}{\square} + \ 2\square 1\right)$$

$$\square\ (\square)_{2\square} = \sin\left(\frac{2\square\square}{\square} + \square 2\square、\quad \square = 1, \ldots,\right.$$

$$\square\ (\square)\ _{21} = \log(\square)$$

其中□是一个学习向量。我们手动选择周期，选择使用现实生活中重要的□$_{abs}$周期和其零头：0.25小时，0.5小时，0.75小时，1小时，2小时，4小时，8小时，16小时，1d，7d，28d，和365d。

我们使用□$_{rel}$=32个对数尺度的均匀间隔的时期来编码相对时间差特征，范围从一秒钟到四周。这假定模型能够区分短持续时间，如10秒与1分钟，与长持续时间，如10天与11天相比，更为重要。

A.2 模型结构

Here we describe in more detail the transformer architecture we use. We first construct the input matrix $A = A_T \cdots A_{T-M+1}{}^\top \in \mathbf{R}^{M\times D_{\text{in}}}$ using the vector representations of the M actions leading up to action a_{T+1} as the sequence. We first project this to the model's hidden dimension H using a learnable matrix $W \in \mathbf{R}^{D_{\text{in}}\times H}$, then add a positional encoding PE $\in \mathbf{R}^{M\times H}$. This generates an input $V^{(0)} = \square W + \text{PE} \in$ R□×^{H}为变压器。

之后，我们应用一个标准的变压器模型，由交替的2层前馈网络（FFN）块和多头自我注意（MHSA）块组成，其中前馈网络的隐藏维度是变压器隐藏维度的4倍。在每个MHSA块中，我们应用掩蔽，所以一个给定的输出可能只关注序列中的当前或之前的元素。

The model architecture can be described as follows:

$$U^{(l)} = V^{(l-1)} + \text{MHSA}\ \ \text{LayerNorm}\ \ V^{(l-1)}$$

$$\square\ ^{(\square)} = \square^{(\square)} + \text{FFN LayerNorm}\ \square\ ^{(\square)} \qquad , \quad \square = 1, \ldots, \tag{3}$$

$$\square$$

在对输入进行方程3所述的转换后，我们将最终的隐藏状态$V^{(L)}$ ∈R□×^{H}通过两层MLP，然后□$_2$将输出标准化。

$$\square^\top \quad \text{变换器之后的输出MLP被定义为}$$
$$_2\ \text{GELU}(W^\top\ \text{LayerNorm}(\square) + \square\ 2) + \square_2$$

其中□$_1$∈R□×^{4}□，W_2∈^{R4}□×□ ，其中H为变换器隐藏维度，D为嵌入维度，而e∈^{R256}是单一嵌入。

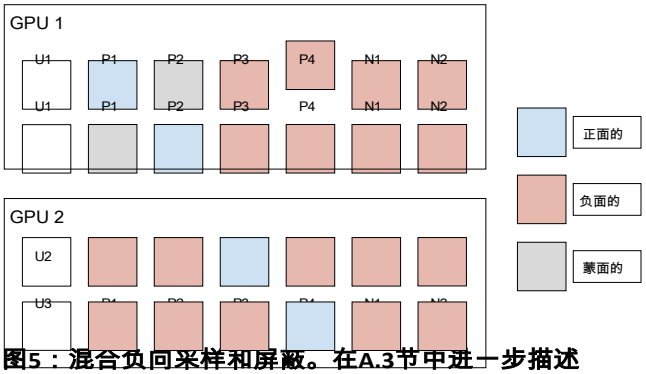

图5：混合负向采样和屏蔽。在A.3节中进一步描述

这就产生了一组嵌入 $E = \ \square^{1}$ - - - $^{\square\square^\top} \in$ R^S×□ ，其中D是最终嵌入维度。我们使用e1，即E的第一行和最近的输出，作为最终的用户嵌入。在用户没有M约定的情况下，输入序列可以被填充到长度为□，被填充的位置可以在注意力和损失计算中被掩盖，类似于语言建模任务中的处理方式。

A.3 混合阴性抽样掩码

在图5中，我们描述了带有掩码的混合负采样。在GPU 1上有两个用户□$_1$（可能是在不同的时间），在GPU 2上有一个□$_2$和□$_3$的嵌入。□$_1$与□$_1$和□$_2$接触，□2与□3接触，而□$_{3\text{则}}$与□4接触。□$_1$和□$_2$ 是随机的负数。在计算损失时，我们把每个阳性作为一个单独的行，但在第一行屏蔽□$_2$ ，在第二行屏蔽□ 。
在第二个过程中，□$_1$ ，因为它们都是□$_1$ 的正例。所有四个阳性例子都出现在两个过程中，因为它们在损失计算之前是跨GPU同步的。每个GPU上的每个用户在最终的损失计算中获得相同的权重，所以在这种情况下，用户□$_1$ 将被分配到两倍于□$_2$ 或□$_3$ 的权重。在实践中，一个批次将包含许多用户，因此权重将在所有GPU上几乎是统一的，即使它不是完全统一的。

在我们的实验中，我们将批量内负片的数量上限为5000，并将随机负片的数量固定为8192。

A.4 建筑烧蚀

在这里，我们展示了不同的模型超参数的影响：

*A.4.1 序列选择。*我们已经彻底探索了从用户的历史记录中删除较弱的参与，以便为经常在Pinterest上参与的用户生成一个更好的em- bedding，但没有看到对用户序列进行稀疏化的明显积极结果。

*A.4.2 嵌入尺寸。*在图6中，我们展示了不同的最终嵌入尺寸对整体性能的影响。我们看到，随着嵌入维度的增加，特别是超过128d的嵌入维度，召回率@10的改善越来越小。我们也可以看到，在较小的尺寸下，嵌入趋向于重新

对大多数用户来说都有类似的结果，这可能意味着一个水平的记忆的流行。因此，在小维度的情况下，每个用户的检索结果可以有更多的多样性，但由于牺牲了大量的Recall@10，这并不是一个好的方法。

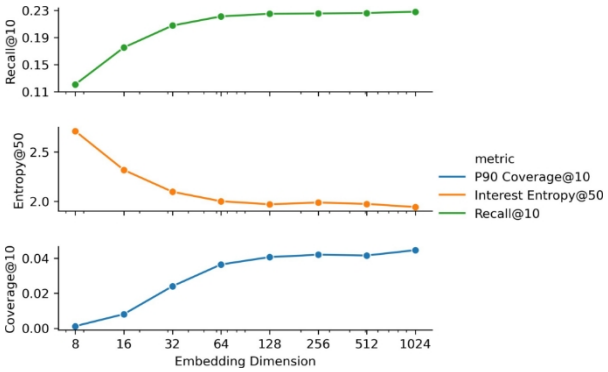

图6：召回率和多样性与嵌入维度。较小的嵌入表现更差，更有可能为许多用户检索到相同的结果。

表9：召回率与模型容量。较大的模型往往表现更好。

数层	隐秘的维度	P90覆盖率@10	召回@10
2	256	0.0359	0.2189
2	512	0.0388	0.2241
2	768	0.0397	0.2246
4	256	0.0366	0.2236
4	512	0.0412	0.2240
4	768	0.0426	0.2272
6	256	0.0383	0.2233
6	512	0.0400	0.2264
6	768	0.0417	0.2293

表10：SASRec，抽样softmax与二进制交叉熵损失对比。在我们的数据集上，采样softmax的表现明显优于二进制交叉熵。

SASRec损失	召回@10	P90覆盖率@10
二元交叉熵	0.138	0.111
采样的Softmax	0.181	0.056
采样的Softmax+相等的$\square_1$损失重量	0.198	0.048

权衡利弊。我们选择使用256d的嵌入，因为它提供了良好的offiine度量，并且与Pinterest排名模型中使用的大多数现有嵌入结构大小相同；将嵌入增加到1024d所带来的可以忽略不计的性能提升，对于大多数下游用例来说不值得将存储成本翻两番。

*A.4.3 变压器结构。*在表9中，我们显示了模型容量对最终性能的影响。较大的模型提高了召回率，无论是在层数方面，还是在隐藏规模方面。当改变用于多头自我关注的头数时，我们没有看到实质性的变化，所以我们将其保持在8头不变。

*A.4.4 对SASRec的修改。*在原论文中，SASRec[10]模型是基于二元交叉熵任务进行训练的，没有任何样本概率修正。我们做了两个修改：（a）我们对$\square_1$，最新的用户嵌入的损失给予同等的权重，以及
(b)我们用采样的softmax代替二元交叉熵。在表10中，我们显示我们的修改大大改善了召回率。