**Enhancing Recommender Systems with Large Language Model Reasoning Graphs**

**摘要**

这段话介绍了一种新的推荐系统方法，名为LLMRG（利用大型语言模型的推理图），旨在通过创建个性化的推理图来改善推荐系统的效果。这些推理图通过因果和逻辑推理将**用户的个人资料和行为序列**相连，以一种易于理解的方式表示用户的兴趣。**LLMRG包含四个主要部分**：链式图推理(chained graph reasoning)、发散式扩展(divergent extension)、自验证打分(self-verification and scoring)和知识库自改进(knowledge base self improvement)。通过使用**图神经网络**编码这些推理图，它们可以作为额外的输入来增强传统推荐系统，而无需额外的用户或物品信息。这种方法展示了如何利用大型语言模型来使推荐系统更加逻辑化和可解释，并且通过个性化的推理图来改进推荐效果。LLMRG的效果在基准测试和现实世界场景中得到了证明，能够增强基础推荐模型。简而言之，这是一种结合了工程化推荐系统和基于语言模型推理图的高级推荐方法，旨在提供更准确、个性化且易于解释的推荐。

# **引言**

传统推荐系统主要依赖机器学习技术来识别用户行为数据中的模式，但这些系统通常不能深入理解内容的真实含义或背后的语义关系，因此在捕捉用户多样化兴趣和行为模式之间的全面联系方面存在困难。

近期的研究尝试通过结合图形结构信息来增强推荐系统，这种信息提供了标准用户-项目交互之外的有价值上下文数据。即便如此，即使是这些基于知识图谱的更先进推荐系统也缺乏执行复杂推理或推断的能力，仅仅叠加事实关系还不足以让系统深刻理解用户的兴趣，从而生成真正有洞察力的推荐。

此外，随着GPT-3、GPT-4等大型语言模型的发展，这些模型展示了在没有明确训练的情况下进行推理、推断和逻辑处理的强大能力，为开发能够捕获细微语义关系的优越知识表示提供了新机会。

基于此，文章提出了一种使用LLM构建个性化推理图的方法，用于推荐系统。这种方法通过LLM处理用户的个人资料和行为序列，输出一个通过因果和逻辑推理链接概念的图形表示。这个推理图编码了用户兴趣和行为之间的高级语义关系，然后使用图神经网络学习这个图的结构和语义的密集特征表示，这个图嵌入作为额外输入提供给传统推荐模型，允许推荐考虑通过推理得到的概念关系，同时仍然受益于传统模型的推荐能力。该方法不仅提高了推荐性能，而且通过表面化推荐背后的明确推理提高了可解释性。

# **背景**

## **基于图的推荐系统**

基于图的推荐系统是通过引入额外的关系信息来提高推荐质量。例如，知识图谱被用来表示实体之间的关系，以捕获复杂的实体互动，这是一种强大的表示方法。除了预定义的知识图谱，一些方法还从用户-项目互动中学习构建信息图。知识图谱提供外部信息，而图学习方法可以提取潜在结构。将两者结合起来，可以同时利用知识图谱和互动图，让推荐模型编码更丰富的连接模式。

然而，与利用大型语言模型（LLMs）构建推理图相比，基于图的推荐系统存在一些**缺点**：

1. 构建和维护知识图谱需要大量的人类专业知识，而LLMs可以自动从大量文本中提取关系知识；
2. 预定义的知识图谱可能在某些实体或领域有覆盖空白，LLMs则可以对文中提及的任何实体进行推理；
3. 从互动中构建图的图学习方法仅限于可观察的用户-项目连接，LLMs可以通过**推理推断**更抽象和潜在的关系；
4. 知识图谱和图在构建后是**静态**的，而LLMs可以随着训练数据的增加，不断扩展其知识和推理能力。

## **选择LLM的理由**

**理解能力**：LLMs能够从文本中解析意义和关系。通过自注意力和上下文嵌入技术，模型可以识别**实体、行动和因果链**。例如，BERT通过掩码语言建模来学习双向表示，这种表示能够整合上下文信息。

**常识积累**：LLMs通过训练，从互联网上获取了大量的词汇，吸收了关于物理、心理学和推理的隐含知识。像PaLM这样的模型经过约束调整，以更好地融入常识，使其能够补充缺失的前提并进行推断。

**转换器架构（transformer architectures）：**赋予了LLMs组合泛化和符号推理能力。自注意力层允许LLMs串联思想，逐步跟随论点，并做出连贯的推断。例如，通过链式思维提示，GPT-3可以解释其推理过程以增强其健壮性。

# **LLMRG**

## **问题重述**

推荐系统的目的是根据用户的历史互动来预测用户的兴趣。顺序推荐将用户的互动历史视为一个有序序列，并试图模拟用户动态变化的兴趣。

让表示用户集合，是items集合，列表按照时间顺序表示用户*u∈U*的交互顺序，其中是与时间t交互的items，是时间序列的长度。

使用相对的时间索引，而不是绝对的时间戳。解释：相对时间索引指的是以某个事件或时间点为基准，描述其他时间点与此基准的时间差，例如，“两天后”、“三小时前”这样的表达，而不是使用具体的日期和时间（如2023年1月1日13:00）

此外，让表示用于个性化模型的用户属性，其中是属性的数量。给定用户的交互历史，顺序推荐任务是预测用户在下一个时间步骤中将与之交互的项目。这可以建模成：用户在时间步骤上的所有可能items的概率分布：

(1)

由此提出了一种新的推荐系统框架——利用大型语言模型（LLMs）构建的推理图（LLMRG）。该方法首先使用LLM根据用户的个人资料和行为序列，通过因果和逻辑推断来构建个性化的推理图，这些图能够解释用户的兴趣并嵌入丰富的语义关系。

紧接着提出了一种**自适应推理架构**，基于LLM的能力进行自我验证，包含四个部分：链式图推理、发散扩展、自验证打分和自改进知识库。通过使用图神经网络编码这些概念推理图，可以将其作为额外输入加入到传统的推荐系统中。这样，推荐系统不仅能从经过工程优化的推荐算法中受益，还能利用从LLM图推理过程中得到的解释性知识。

## **自适应推理架构**

### **Chained Graph Reasoning(链式图推理)**

这个部分描述了如何结合用户行为序列（）和相关用户属性（）来构建推理链（），这些推理链旨在为每个item链接到现有的链条上，如果逻辑上相连的话；或者如果没有适用的链接到现有推理链的话，就在项目本身的基础上开始全新的链条。这个过程旨在进一步定制推理链，以便推荐项目，通过逐步沿着用户的行为序列进行迭代推理链的构建，直到最后一个项目为止。

具体来说，该部分采用了一个基于提示的框架，利用大型语言模型创造性地生成可能的新推理链，这些新推理链逻辑上能激励用户与他们序列中的下一个已知item进行互动。提示输入包括已知的下一个item、到目前为止构建的现有推理链以及可用的用户属性。它输出一套可能的新推理链，解释用户可能为什么想要采取下一个item。这些动态生成的新链条被整合到不断发展的逻辑推理图中，以支持对用户不断演变的行为轨迹背后越来越复杂的相互依赖的动机和兴趣的建模。

### **Divergent Extension（发散式拓展）**

用“发散扩展模块”来进行创造性的思维延伸，预测用户可能会感兴趣的下一些项目。这个模块通过对每个推理链进行想象性的延续，挖掘用户的动机和思考过程，利用语言模型探索合理的推理轨迹的可能延续，来预测用户接下来可能感兴趣的相关项目。

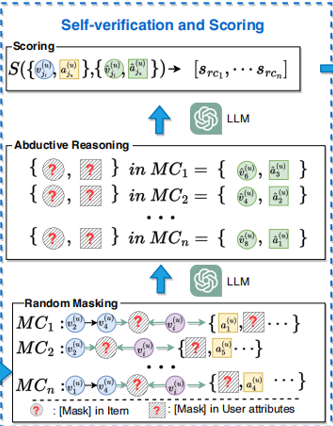
例如，如果一个推理链表明用户对带有复杂哲学主题的科幻电影感兴趣，该模块就可能生成新的序列，预测用户可能喜欢的具有相似主题和风格的更深层次的科幻电影。

这个发散思维的过程不仅限于根据用户已消费内容做出反应性推荐，而是主动预测并推荐新项目，以模拟用户的动机。该方法能够产生多个多样化的可能选项，更好地捕捉用户的多面兴趣和他们可能采取的不同路径。**生成的新项目预测可能不在原始推荐列表中，因此需要使用另一个小型语言模型**来计算生成项目与原始列表之间的相似度，以检索最相关的推荐项。通过这种方式，可以为用户量身定制一套全面的个性化推荐。

### **Self-verification and Scoring（自验证打分）**

自验证打分模块利用大型语言模型（LLM）的归纳推理（abductive reasoning）能力，来检查由**链式图推理**和**发散扩展模块**动态生成的推理链的**合理性和连贯性**。在将新的推理链添加到图中之前，模块会遮蔽（mask）推理链中关键项或用户属性，然后提示LLM填充被遮蔽的链中的空白（[Mask]），以最合理的预测完成这一链。(指的是掩码链)

如果预测的item或属性与最初被遮蔽的内容匹配，就说明这个推理链在逻辑上是通顺的，并且与用户的行为历史和属性一致。匹配得分越高，表明整个推理图越稳健。相反，低匹配分数则指出某些推理链在连贯性或合理性上可能存在问题。系统随后可以选择性地过滤或调整这些问题链条，再将它们整合入图中。因此，设置了一个**阈值分数**用于自我验证，以判断推理的合理性，从而提高链式图推理和发散扩展模块动态构建的推理链的整体健全性，确保推荐的可靠性与用户兴趣相符。具体而言，这个模块主要涉及三个步骤：随机遮蔽、归纳推理和评分，这在图1中有所示例:



### **Knowledge Base Self-improving.（知识库自改进）**

由于观察到**许多知识元素和推理过程在不同查询中被重复应用**，为了避免重复工作，系统引入了一个缓存已验证推理链的知识库，以便后续重用。

通过重用之前的推理结果而不是重新计算，系统显著减少了对语言模型的使用。系统采用了一种自我改进的方法来随时间维护知识库的质量。利用自我验证和评分模块来评估推理链的有效性，只有**得分高的链被保留在知识库**中，而低分的链则被丢弃，以过滤掉低质量或错误的推理结果。在进行新的推理之前，系统首先检查知识库中是否已经存在相关的链。如果存在，就直接检索并利用这个预先计算好的链，而不是再次调用语言模型。通过这个包含缓存的高质量推理链的知识库，显著减少了计算需求。实验表明，与从头开始进行推理相比，这种方法可以在经过3000次推理和验证步骤后，将语言模型的使用量减少大约30%。

## **LLMRG 框架**

顺序推荐方法通常将用户的交互历史视为一个有序的序列，并试图对用户动态变化的兴趣进行建模。如图1所示，LLMRG有两个组成部分，即一个具有自验证的自适应推理模块和一个基本的顺序推荐模型：

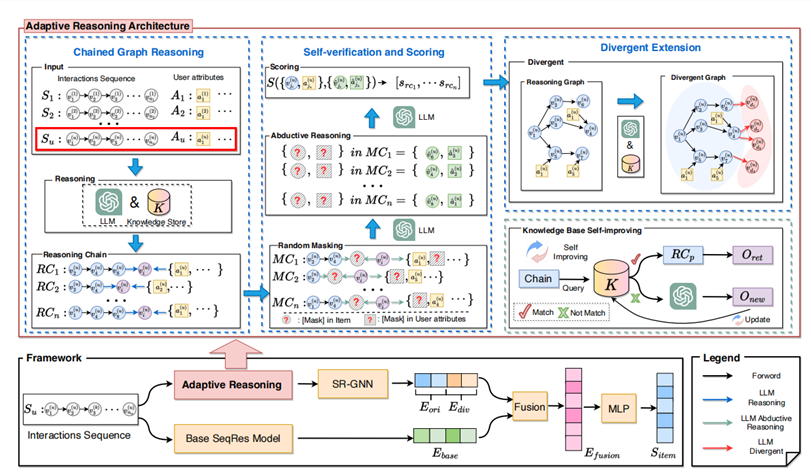
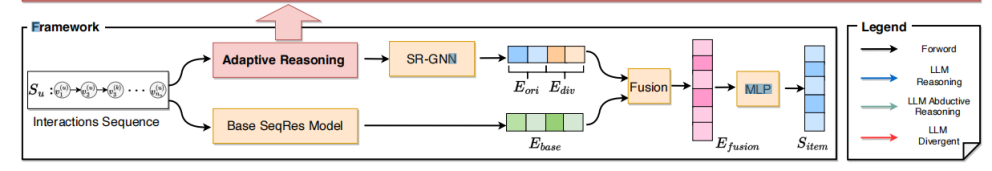


图1：LLMRG框架有两个主要组成部分

即一个具有自验证的自适应推理模块和一个基本的顺序推荐模型。将来自自适应推理模块（Eori和Ediv）和基础模型（Ebase）的嵌入连接起来，以获得自我融合。这种融合的嵌入用于预测用户的下一个项目。关键优点是，自适应推理模块可以构建个性化的推理图，超越了用户兴趣的顺序建模。自我验证和得分有助于改进推理过程。将其与标准推荐模型相结合，可以在不访问额外信息的情况下结合互补的优势。

自适应推理模块采用用户的交互序列和属性作为输入。该输入通过链式图推理、自验证评分，以及反复的发散扩展模块来构造推理图和发散图。自适应推理模块表示为映射，其中和分别表示推理图和发散图。我们利用SR-GNN从图中自动提取嵌入，它可以在推荐任务中支持不同的邻接矩阵和有向图。这个过程为推理图产生两个嵌入图(降维):**和**，：和。同时，基本顺序推荐模型直接处理输入以产生嵌入的。最后，将自适应推理模块（和）中的嵌入图与基础模型（）连接起来，获得。通过，融合后的嵌入值可用于预测用户的下一个item。

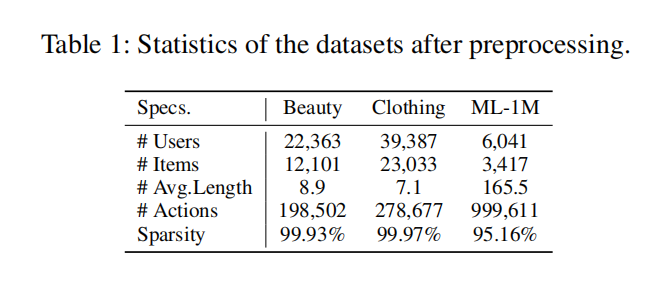
主要优点:自适应推理模块可以构建个性化的推理图，发散扩展采用发散思维超越反应性建议，主动根据进化的行为轨迹推荐新的项目。自验证评分也能提升推理过程的准确度。将其与标准的顺序推荐模型相结合，可以在不寻求额外信息的情况下有互补的优势。



# **实验**

## **4.1 实验设计**

### **4.1.1 数据库 & 评价指标**

选择三个基准数据集上进行实验：亚马逊美妆(the Amazon Beauty)、亚马逊服装(Amazon-Clothing)（Xue等人，2022年、2023年）和MovieLens-1M（ML-1M）数据集。亚马逊数据集最初由McAuley等人在2015年介绍，以其高稀疏性和短序列长度而闻名。选择了美妆和服装子分类，使用细粒度的产品类别和品牌作为项目属性。ML-1M数据集，来源于Harper和Konstan（2015年），是一个大型且密集的数据集，包含从电影推荐网站(MovieLens)收集的长项序列，以电影类型作为属性。在预处理后，这四个数据集的统计信息汇总在表1中：

将所有用户-项目互动视为隐式反馈(implicit feedback)。对于每个用户，我们去除重复的互动，并按时间顺序排序剩余的互动来构建序列化用户档案。

为了评估推荐系统的性能，采用了**留一法策略**，即我们反复从每个用户的互动序列中保留一个项目，这可以测试模型预测被保留项目的能力。在整个项目集上进行预测，不进行任何负采样。

选用了两个广泛使用的排名指标：Top-n指标**HR**@n（命中率）和**NDCG**@n（标准化折扣累积增益），其中n设置为5和10。**HR**@n测量被保留的项目是否出现在前n个推荐中，而**NDCG**@n则考虑了被保留项目的位置，为高排名的命中分配更高的分数。为了确保评估的稳健性，用不同的随机种子重复每次实验5次，并将平均性能作为最终指标。这考虑到可变性，并确保结果不依赖于特定的随机初始化。

### **4.1.2 基准测试**

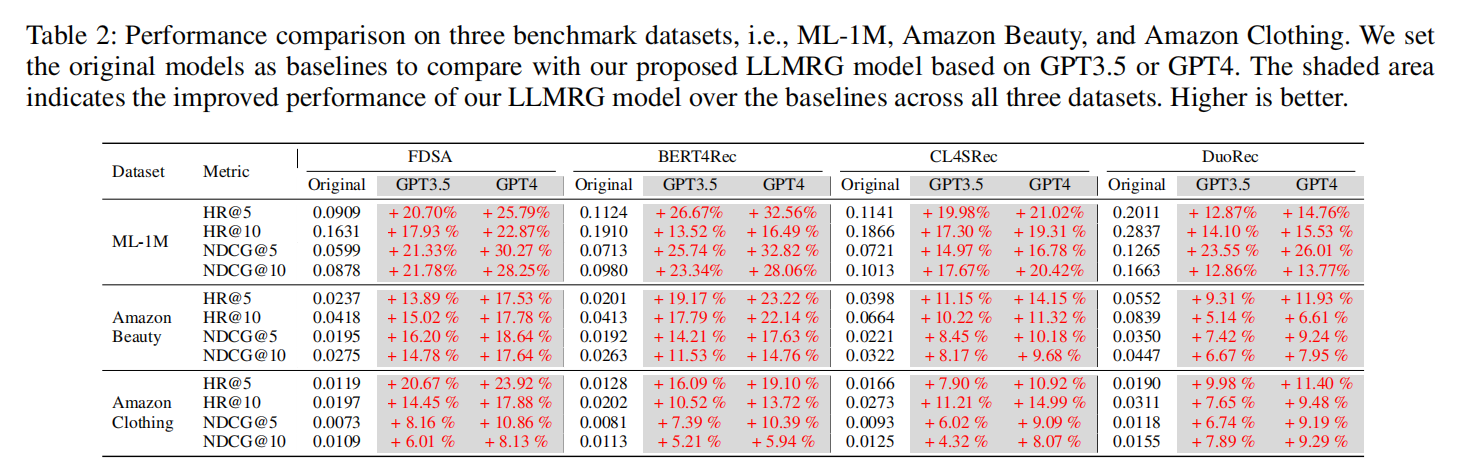
三组基线方法进行比较：

（1）通用序列方法使用序列编码器生成用户和项目的隐藏表示。例如，BERT4Rec（Sun等人，2019年）采用双向Transformer作为序列编码器；

（2）属性感知序列方法将属性信息融合到序列推荐中。例如，FDSA（Zhang等人，2019年）应用自注意力块来捕捉项目和属性的转换模式；

（3）对比序列方法基于通用序列方法设计辅助目标进行对比学习。例如，CL4SRec（Xie等人，2022b）为序列推荐中的对比学习提出了数据增强策略。DuoRec（Qiu等人，2022年）为序列推荐中的对比学习提出了监督和非监督采样策略。

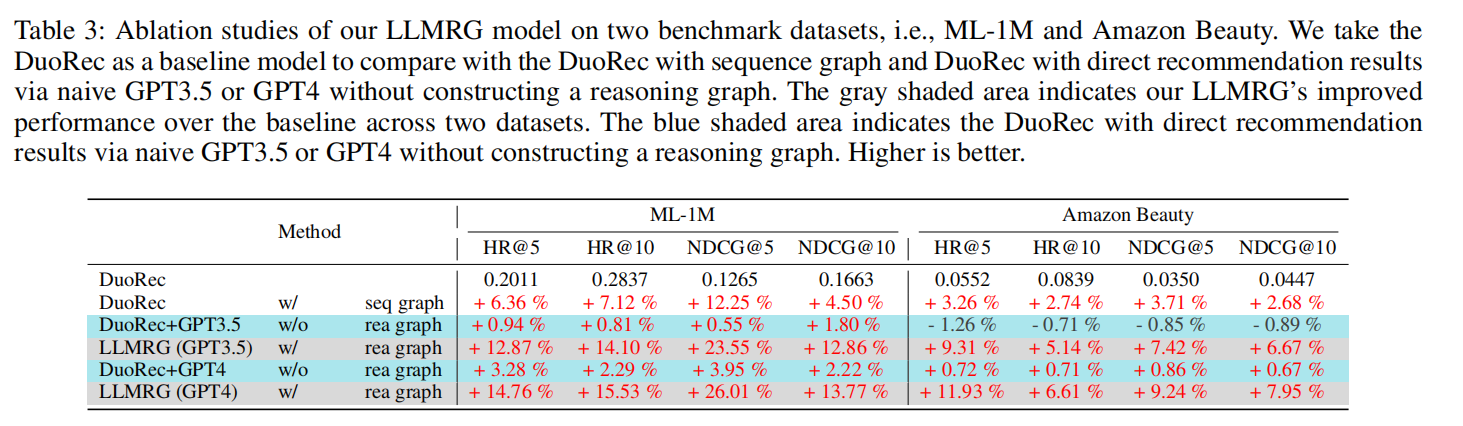
## **4.2 实验结果和分析**

如表2所示，在三个数据集进行了基准实验，将提出的建立在GPT3.5或GPT4之上的LLMRG模型与几种强基线方法进行了比较，包括FDSA（Zhang等人2019年）、BERT4Rec（Sun等人2019年）、CL4SREC（Xie等人2022年b）和DuoRec（Qiu等人2022年）：

表中的阴影区域突出显示了我们的LLMRG模型在三个数据集的所有基线上实现的性能改进。这些结果证明了LLMRG的即插即用特性，它可以有效地增强现有的多个推荐系统。更重要的是，我们观察到，与原始的基线模型相比，应用LLMRG后，在HR@5、HR@10、NDCG@5和NDCG@10上都有了显著的性能提高。这表明，传统的推荐系统难以对不同用户兴趣的概念关系和行为序列进行建模。相比之下，我们提出的LLMRG框架可以在不需要任何额外信息的情况下提高推荐性能。这些改进展示了大型语言模型如何为推荐系统带来逻辑推理和可解释性。此外，LLMRG的性能与底层LLM能力成正比——基于GPT4的LLMRG性能始终优于GPT3.5。此外，当比较ML-1M电影数据集与美容和服装产品数据集时，LLMRG方法在ML-1M的所有评估指标上取得了更大的改进。这是因为电影富含更复杂的语义信息，能够建立更多语义逻辑推理关系。相比之下，美容和服装产品的叙事复杂性较低，所以进行关系推理来改进推荐的机会较少。

## **4.3 消融实验**

为了证明我们提出的推理图的有效性，使用两个基准数据集：ML-1M和Amazon Beauty对我们的LLMRG模型进行了消融研究。我们将LLMRG与DuoRec基线模型以及Wu等人（2019）提出的用简单序列图增强的DuoRec进行了比较。序列图不需要推理就可以直接建模交互序列。我们还将DuoRec与不构造推理图的LLM- GPT-3.5和GPT-4 -相结合进行了比较。LLM只是简单地根据包含历史序列和用户配置文件的提示输出推荐的项，而不需要推理图。



如表3所示，与完整的LLMRG模型相比，序列图加强的DuoRec模型只有微小的改进。未集成推理图DuoRec+GPT3.5模型在ML-1M上未能显著提高DuoRec的性能，并且在亚马逊美妆数据集上甚至降低了性能。虽然DuoRec+GPT4在性能上超过了DuoRec+GPT3.5，但仍远远落后于我们的LLMRG模型。这些结果证明了构建推理图在提高性能中的关键作用，且简单的下一项预测还不够充分（DuoRec+GPT3.5和DuoRec+GPT4）。通过显式地建模用户配置文件和交互序列之间的推理过程，LLMRG能够做出准确的、可解释的建议。消融实验证实了推理图在有效利用大型语言模型对推荐系统的力量方面的必要性和价值。



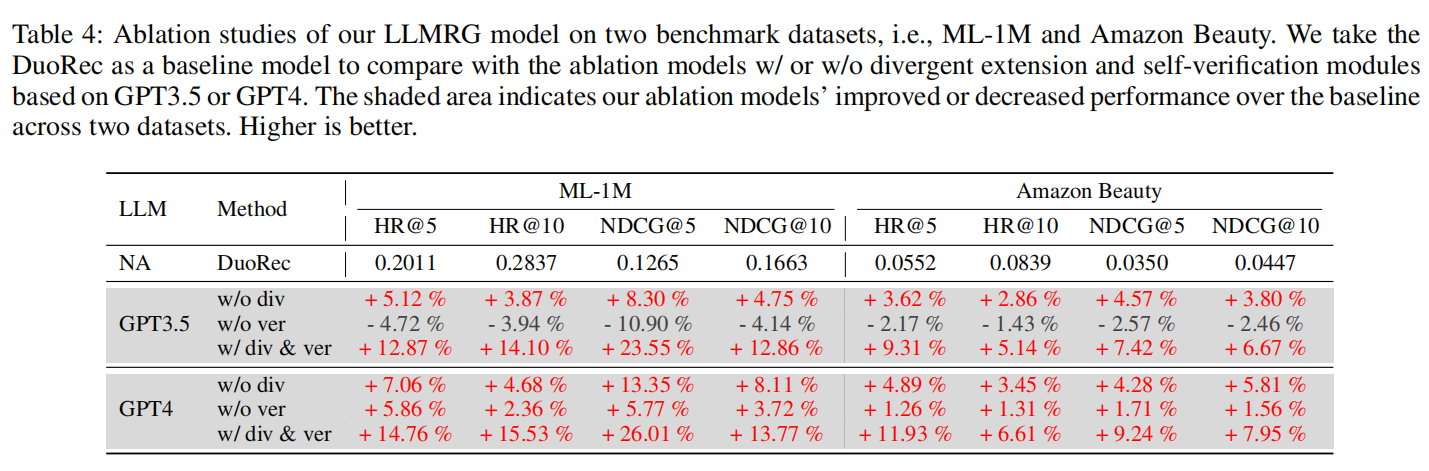
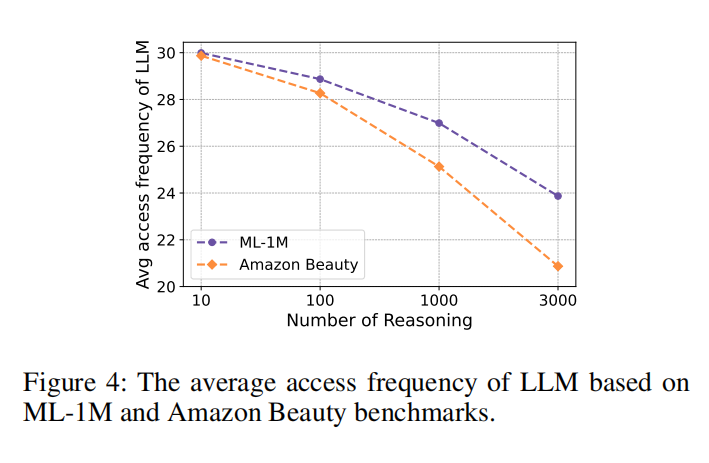
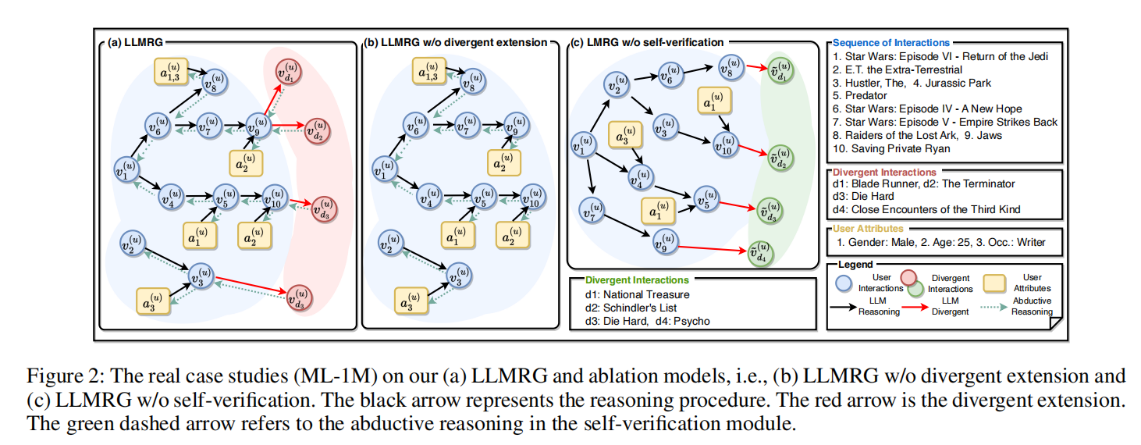
我们的其他消融研究进一步探索了在我们的LLMRG框架中每个模块的有效性。使用DuoRec作为基线模型，我们将其与基于或没有GPT3.5或GPT4的发散扩展和自我验证模块的LLMRG消融版本进行了比较。

表4中的结果显示，LLMRG(与GPT3.5或GPT4)没有发散性扩展模块，与完整的LLMRG相比，只提供了微小的改进。然而，从LLMRG（GPT3.5）中移除自验证模块实际上会降低性能。这证明了GPT3.5有限的推理能力——没有验证，不受控制的推理会引入噪声，降低整体性能。总的来说，这些消融实验清楚地证明了我们的发散扩展和自我验证模块在保持准确性的同时实现更高级的推理方面的价值。这些模块协同工作，扩展可能解决方案的搜索空间，同时过滤掉不准确或不相贯的推理线。

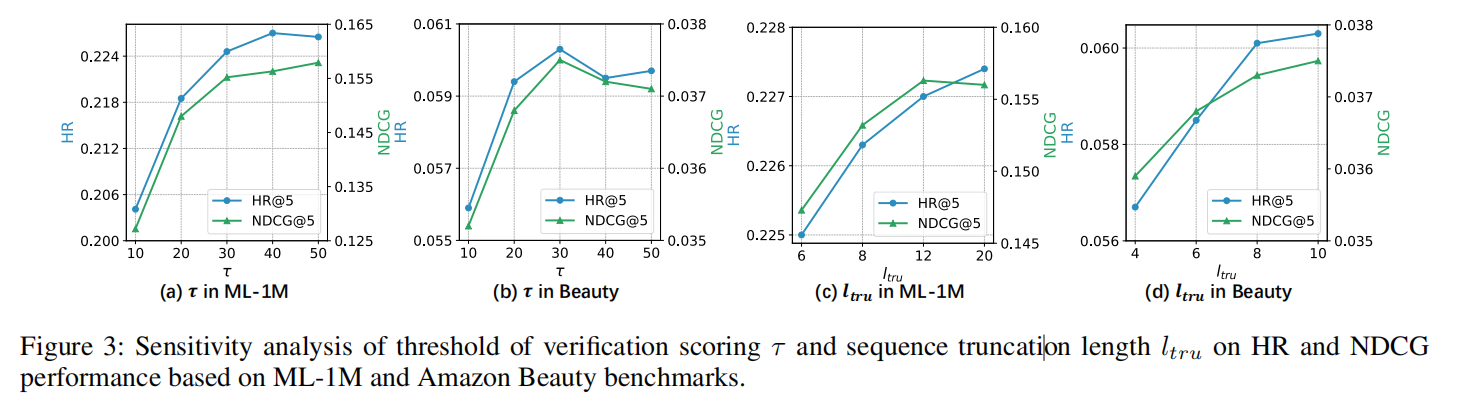


如图4所示，分析知识库自我改进的有效性。基于LLMRG（GPT3.5），我们计算了在两个基准数据集上的模型调用到LLM的平均访问频率。实验结果表明，随着推理步数的增加，平均访问频率显著降低。经过3000次的推理和验证，与不使用该模块相比，平均访问频率降低了约30%，证明了该知识库包含了可重用的高质量推理链。此外，亚马逊Beauty中高质量推理链的重用率高于ML-1M，而亚马逊数据集的长尾分布是造成这种差异的原因之一。

为了提供直观的例子来证实定量结果，检查了来自ML-1M的真实案例。研究数据集使用(a)我们的完整LLMRG模型，(b) LLMRG没有发散扩展模块，(c) LLMRG没有自验证模块。图2和附录中的案例研究说明了两种模型之间在推理方面的差异。LLMRG生成具有合理理由的一致建议，利用分歧思维来扩大可能性，并利用自我验证来过滤掉糟糕的选项。如果没有不同的扩展，LLMRG将努力超越明显的选择。如果没有自我验证的话，LLMRG的建议就会变得更加投机，有时甚至更荒谬，因为这个模型缺乏检查自己思维的能力。这些定性分析反映了我们数值结果中的模式，作为进一步验证每个推理模块与我们完整的LLMRG框架协同工作所增加的价值。案例研究提供了直观的例子，说明我们的方法如何结合创造性思维和批判性评价来产生合乎逻辑的建议。

## **4.4 灵敏度分析**

我们评估了LLMRG对HR和NDCG上的两个最关键参数和序列长度ltru的敏感性，它们分别控制了验证评分的阈值和序列截断长度。图3(a)和(b)显示，较大的值会产生更鲁棒的推理，并过滤掉较差的选项，从而提高了模型在ML-1M数据集上的性能。



然而，在Beauty数据集上，性能从开始下降，可能是因为更高的验证评分阈值过滤掉了更多的推理链，增加了图的稀疏性。图3(c)和(d)表明，一般来说，较长的序列通过合并更多的信息可以带来更好的推荐结果。总之，较大的和较长的序列都倾向于提高性能。显示出一个峰值，超过这个峰值，稀疏推理图效果变差，特别是对于逻辑较差的序列，比如亚马逊产品。

# **结论**

作者提出的LLMRG是一种利用LLM构建个性化推理图的新方法。这种方法展示了LLM如何在不需要任何额外信息的情况下为推荐系统带来逻辑推理和可解释性。利用现实世界的数据集验证了LLMRG方法的有效性并证明即插即用方法可以有效增强现有的多种推荐器。但是需要指出的是，虽然设计了知识库自改进功能，但LLMRM基本上受限于对于较长的交互序列，LLMRM的访问频率基本上是有限的。

# **补充说明**

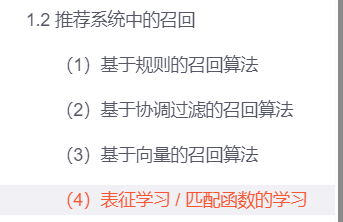
首先，这篇文章应该是属于推荐系统这个比较大类下的对SR-GNN的改动 那首先得先理解什么是推荐系统这类算法：

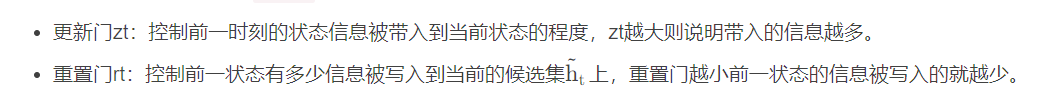
<https://zhuanlan.zhihu.com/p/558136597>

SR-GNN：

[SR-GNN-CSDN博客](https://blog.csdn.net/qq_46281780/article/details/127981104)

[【序列推荐、图神经网络】SR-GNN：Session-based Recommendation with Graph Neural Networks-CSDN博客](https://blog.csdn.net/CRW__DREAM/article/details/128609712?utm_medium=distribute.pc_relevant.none-task-blog-2~default~baidujs_baidulandingword~default-0-128609712-blog-127981104.235%5ev43%5epc_blog_bottom_relevance_base8&spm=1001.2101.3001.4242.1&utm_relevant_index=3)





[【序列召回推荐】(task2)序列召回GRU4Rec模型-CSDN博客](https://blog.csdn.net/qq_35812205/article/details/127780974?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522170817246316800215051735%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=170817246316800215051735&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-1-127780974-null-null.142%5ev99%5epc_search_result_base1&utm_term=%E5%BA%8F%E5%88%97%E5%8F%AC%E5%9B%9E&spm=1018.2226.3001.4187)

[图神经网络GNN在推荐系统的应用：综述\_为了概括推荐系统的历史,可以一般地分为三个阶段:浅层模型[85, 148, 149],神经网-CSDN博客](https://blog.csdn.net/weixin_42327752/article/details/123921701?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522170817486116800185854920%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=170817486116800185854920&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-2-123921701-null-null.142%5ev99%5epc_search_result_base1&utm_term=GNN%E5%8F%AC%E5%9B%9E&spm=1018.2226.3001.4187)

Attention机制来获取序列中每一个item对于序列中最后一个item的attention score:

[深度学习：注意力机制Attention\_注意力机制是谁提出的-CSDN博客](https://blog.csdn.net/pipisorry/article/details/78338085?ops_request_misc=&request_id=&biz_id=102&utm_term=Attention%E6%9C%BA%E5%88%B6&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-4-78338085.nonecase&spm=1018.2226.3001.4187)

还有这个召回机制采用的是哪种模式

是i2i 还是u2i或者别的呢？



于是从上面的内容提出了对embedding方法的疑惑：

[推荐系统中常用的embedding方法\_item embedding-CSDN博客](https://blog.csdn.net/qq_40006058/article/details/123314370?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522170817471416800185876878%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request_id=170817471416800185876878&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~first_rank_ecpm_v1~rank_v31_ecpm-2-123314370-null-null.142%5ev99%5epc_search_result_base1&utm_term=Embedding%E6%96%B9%E6%B3%95%E5%AF%B9User%2FItem%E5%88%86%E5%88%AB%E5%AD%A6%E4%B9%A0%E5%88%B0%E4%B8%80%E4%B8%AA%E5%90%91%E9%87%8F%E8%A1%A8%E5%BE%81&spm=1018.2226.3001.4187)

[embedding\_embedding首次提出-CSDN博客](https://blog.csdn.net/mudan97/article/details/115549278?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522170817471416800185876878%2522%252C%2522scm%2522%253A%252220140713.130102334.pc%255Fall.%2522%257D&request_id=170817471416800185876878&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~first_rank_ecpm_v1~rank_v31_ecpm-17-115549278-null-null.142%5ev99%5epc_search_result_base1&utm_term=Embedding%E6%96%B9%E6%B3%95%E5%AF%B9User%2FItem%E5%88%86%E5%88%AB%E5%AD%A6%E4%B9%A0%E5%88%B0%E4%B8%80%E4%B8%AA%E5%90%91%E9%87%8F%E8%A1%A8%E5%BE%81&spm=1018.2226.3001.4187)

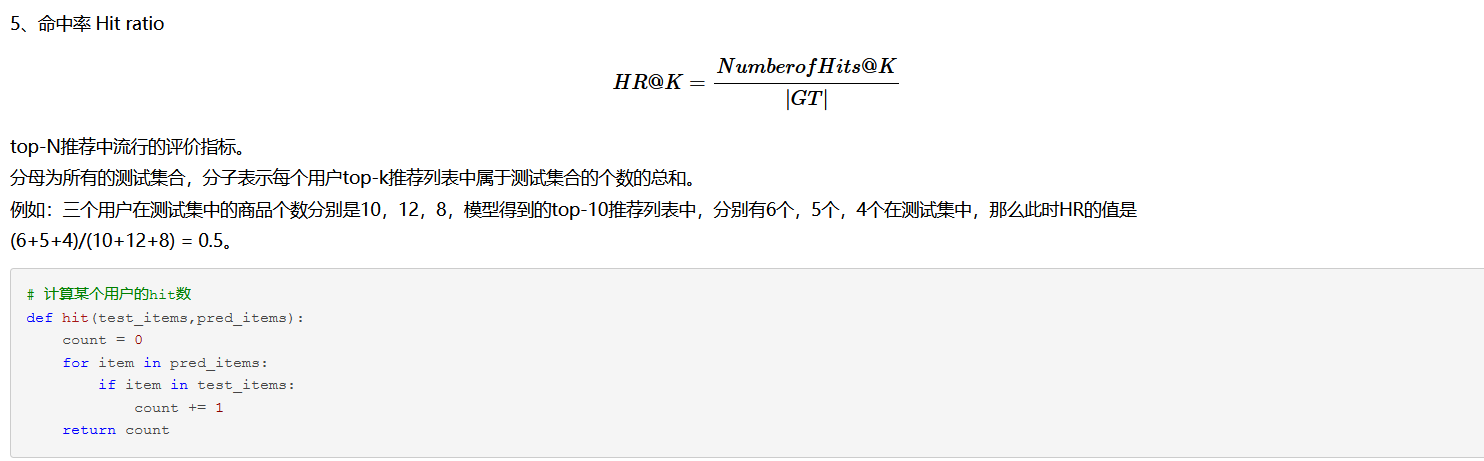
基本顺序推荐模型指的是什么

MLP:

[神经网络1：多层感知器-MLP - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/63184325)

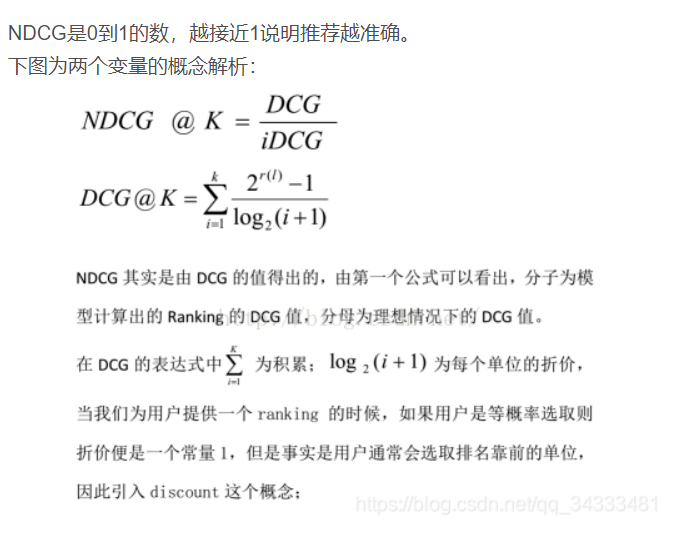
[深度学习02-神经网络(MLP多层感知器)-CSDN博客](https://blog.csdn.net/liaomin416100569/article/details/130572559)

**HR**@n（命中率）



**NDCG**@n

<https://blog.csdn.net/qq_34333481/article/details/102623753>



**BERT4Rec采用双向Transformer作为序列编码器：**

<https://blog.csdn.net/weixin_45104951/article/details/123486554>

**FDSA应用自注意力块来捕捉项目和属性的转换模式：**

<https://blog.csdn.net/qq_35148758/article/details/101453859>

**CL4SRec为序列推荐中的对比学习提出了数据增强策略：**

https://zhuanlan.zhihu.com/p/560301993

**DuoRec为序列推荐中的对比学习提出了监督和非监督采样策略：**

<https://blog.csdn.net/CRW__DREAM/article/details/123405727>

这应该就是四个原本就出现的推荐系统 加上LLMRG的改进的对比：

"从图中自动提取嵌入"是指在图神经网络（Graph Neural Networks, GNNs）的上下文中，自动从图数据（如网络、社交网络、推荐系统中的交互图等）中学习到表示节点、边或整个图结构的低维度、密集的向量表示，这些向量表示被称为“嵌入”（embeddings）。

具体来说，这个过程涉及到使用图神经网络对图结构中的节点进行编码，从而捕捉到节点的特征信息以及节点间的拓扑关系。通过这种方式，每个节点都被转换成一个固定长度的向量，这个向量能够有效地概括节点的属性以及它与其他节点的关系。

在推荐系统的上下文中，从图中自动提取嵌入特别有用，因为它允许系统理解和利用用户和项之间复杂的交互模式，从而提高推荐的准确性和个性化。例如：SR-GNN通过捕捉会话数据中的项与项之间的关系，自动提取有用的特征表示（即嵌入），进而用于更准确地预测用户的下一步可能感兴趣的项。

**对第一张图的补充：**

以一个简化的例子来说明如何将推理图中的节点转换为高维空间中的向量表示，即嵌入图的生成过程。假设我们有一个电影推荐系统，我们的目标是根据用户的观影历史来推荐电影。在这个例子中，推理图是由用户和他们观看过的电影构成的。推理图节点代表用户和电影。边代表用户观看过某部电影。

例如，假设我们有两个用户（用户A和用户B）和三部电影（电影1、电影2、电影3）。用户A观看了电影1和电影2，用户B观看了电影2和电影3。因此，推理图如下：

用户A节点连接到电影1和电影2的节点。

用户B节点连接到电影2和电影3的节点。

嵌入图的生成将推理图中的每个节点转换为高维空间中的向量表示的过程称为嵌入。每个向量的目的是捕捉该节点的重要特性和它与其他节点的关系。在我们的例子中，嵌入过程可能会考虑以下因素：用户的观影偏好（例如，偏好动作电影还是爱情电影）；电影的属性（如类型、导演、发行年份）；用户与电影之间的交互（如评分）。

假设我们将每个节点转换为一个3维向量。经过嵌入过程后，我们可能得到如下表示：

用户A的向量表示可能是[0.9, 0.1, 0.5]，其中每个数字反映了用户偏好和观影历史的不同方面。

电影1的向量表示可能是[0.8, 0.2, 0.0]，反映了该电影的类型、受众评分等特性。

**嵌入向量的使用**

嵌入向量可以被用来计算用户和电影之间的相似度，进而推荐给用户可能感兴趣的电影。例如，如果用户A的向量与电影3的向量在高维空间中很接近，系统可能会推荐电影3给用户A。通过这种方式，嵌入图不仅捕捉了用户和电影之间的直接关系（如观影历史），还能通过嵌入向量的相似度来推断用户可能的偏好，从而提高推荐系统的准确性和效率。  
 融合嵌入向量的过程涉及将来自不同来源的嵌入向量结合起来，以形成一个综合的、更能代表用户或项目特性的向量。这个过程旨在利用不同模型或数据源生成的嵌入信息，通过某种策略融合这些信息，以提高推荐系统的性能。下面是一个简化的例子来说明这个过程。

假设我们有两个不同的嵌入向量来源：

基于用户的观影历史生成的嵌入向量：这些向量基于用户过去的观影行为，如观看的电影类型、评分等。

基于推理图生成的嵌入向量：这些向量来自于一个更复杂的推理图，该图可能还包含了用户社交网络中的关系、用户对电影的评论内容分析等。

融合过程

为了预测用户的下一个可能感兴趣的电影，我们决定将这两个来源的嵌入向量融合起来。具体步骤如下：

向量对齐：首先，确保两组向量在维度上是一致的。如果它们的维度不同，可能需要通过某种维度缩放或变换方法（如PCA降维、向量填充等）来对齐它们。

融合策略选择：选择一个融合策略，如简单的向量加权平均、向量拼接、或使用更复杂的模型（如神经网络）来学习如何最好地结合这些向量。

实施融合：

假设我们选择了向量加权平均的方法，我们可能会为基于观影历史的嵌入向量赋予0.6的权重，为基于推理图的嵌入向量赋予0.4的权重。

假设用户的观影历史嵌入向量为[0.9, 0.1, 0.5]，推理图嵌入向量为[0.8, 0.2, 0.3]。

融合后的计算：

0.6×[0.9,0.1,0.5]+0.4×[0.8,0.2,0.3]=[0.86,0.14,0.44]

0.6×[0.9,0.1,0.5]+0.4×[0.8,0.2,0.3]=[0.86,0.14,0.44]。

**基本顺序推荐模型:** 这是一个直接处理输入（可能是用户的历史行为数据）以产生嵌入的模型。这意味着它不依赖于先前提到的推理图，而是直接从原始数据中学习如何表示信息为嵌入向量。这些嵌入向量代表了用户的兴趣和偏好。

预测：

**在推荐系统中，使用多层感知机（MLP）处理融合后的嵌入向量**，并最终输出一个预测（如，指的可能是"推荐项目"）的过程涉及到深度学习技术的应用。这个过程大致可以分为以下几个步骤：

1. 融合向量输入

首先，用户和项目的融合嵌入向量作为输入被送入MLP中。这些融合向量可能是通过前面讨论的方法得到的，它们综合了来自不同源的信息，比如用户的行为数据、社交网络数据、项目的属性等。

2. 多层感知机（MLP）处理

MLP架构：MLP是一种前馈神经网络，包含一个或多个隐藏层。每一层都包含若干神经元，这些神经元与前一层的每个神经元都有连接。每个连接都有一个权重，而每个神经元都有一个偏置项和一个激活函数（如ReLU）。

特征学习：通过MLP，融合向量经过一系列的线性变换和非线性激活函数处理，MLP能够学习到输入数据的高级特征表示。这个过程中，模型通过调整权重和偏置来最小化预测错误。

3. 输出层

MLP的最后一层是输出层，它的设计取决于具体的任务目标。在推荐系统中，输出层通常设计为输出一个向量，该向量的每个元素对应于用户可能感兴趣的一个项目的得分。

对于""的预测，MLP的输出层可能就是预测用户最有可能感兴趣的那个项目，或者是一个得分最高的项目列表。

4. 预测和推荐

得分计算：MLP输出的是一个或多个项目的得分。这些得分反映了用户对各个项目的偏好强度。

推荐生成：根据得分高低，系统可以选择一个或多个得分最高的项目作为推荐（）。例如，如果系统的任务是推荐一个最有可能被用户点击或购买的商品，那么它会选择得分最高的商品。

5. 训练和优化

在训练过程中，使用如交叉熵损失或均方误差损失函数来评估预测的准确性，并通过反向传播算法来更新MLP中的权重和偏置，以最小化预测误差。

优化算法，如SGD、Adam等，被用于在训练过程中调整模型参数。

通过MLP处理融合向量并输出推荐项目（），推荐系统能够利用深度学习模型的强大能力来捕捉复杂的非线性关系和交互效应，从而提供更精准、个性化的推荐。