

# RecCoT: 通过链式思考提升推荐效果

Shuo Yang

Nanjing University of Science  
and Technology  
Nanjing, China  
yangshuo1115@gmail.com

Jiangxia Cao<sup>\*</sup>

Kuaishou Technology  
Beijing, China  
jiangxiacao@gmail.com

Haipeng Li

Nanjing University of Science  
and Technology  
Nanjing, China  
lihaipeng@njjust.edu.cn

Yuqi Mao

Nanjing University of Science  
and Technology  
Nanjing, China  
yuqimao@njjust.edu.cn

Shuchao Pang<sup>\*</sup>

Nanjing University of Science  
and Technology  
Nanjing, China  
pangshuchao@njjust.edu.cn

## 摘要

在实际应用中,用户总是从多个方面与项目进行互动,例如通过隐式二进制反馈(如点击、不喜欢、长时间观看)和显式反馈(如评论、评价)。

现代推荐系统(RecSys)从这些隐式反馈信号中学到用户-项目的协作信号作为一个大规模的二进制数据流,并基于用户的个性化历史交互来推荐其他高度相似的项目。然而,从这种协作连接的角度来看,RecSys并不关注项目本身的实际内容,而是优先考虑项目之间的行为共现的高概率信号。因此,在这种二进制学习范式下,RecSys难以理解用户为什么喜欢或不喜欢某些项目。为了缓解这一问题,一些工作试图利用基于内容的评论来捕捉语义知识以增强推荐模型。然而,这些方法大多数专注于预测评价的评分,但没有提供人类可理解的解释。为了更好地识别用户的评论,我们探索了一个级联推理框架RecCoT,生成一个人类可解读的思维链来提升推荐效果。受成功推理大语言模型(如DeepSeek R1)的启发,我们首先提出了一种频率感知奖励策略来监督更大的LLM适应长尾评分分布。在更大LLM收敛后,我们可以为每条评论生成解释性的推理CoT,以支持微调较小的LLM来生产语义嵌入以提升推荐准确性。我们的工作研究了推

理能力对个性化推荐的影响,在推荐性能上取得了显著改进<sup>1</sup>。

## CCS Concepts

• Information systems → Recommender systems.

## 1 介绍

**背景。**如今,随着互联网技术和数字内容的蓬勃发展,信息爆炸成为越来越严重的问题,影响了用户体验。为了帮助用户找到最相关的信息,在许多平台上(例如快手(短视频和直播)、淘宝(购物)等),推荐系统变得越来越重要。据我们所知,在业界中,这些平台上的推荐系统大多基于协同过滤的思想(如YouTube [4]),通过大量的用户行为的多分类任务学习用户的偏好(例如,是否点击/不喜欢/长视频观看/购买该商品?)。总的来说,CF在捕捉用户的积极行为信号方面极其简单、高效和准确。然而,它主要关注概率计算,而不是真正理解项目背后的意义,这使得解释消极行为(如不喜欢)变得困难。尤其重要的是,除了隐式二进制反馈(例如点击、不喜欢、长时间观看),用户还会留下一些显式反馈(例如评论、评价),来进一步描述他们对这种互动的感受,这为理解为什么用户喜欢或不

<sup>\*</sup>Corresponding authors.

<sup>1</sup><https://github.com/shuoyang2/RecCoT>

喜欢某些项目提供了空间。因此，一个热门话题是利用基于内容的评价提供语义知识以增强推荐系统。

**相关工作。**在此研究方向中，主要任务是评分预测，其定义如下：给定用户的过往评论和其他用户对项目的评论，预测一个用户-项目组合的评分。早期的工作如 DeepCoNN [16] 和 D-attn [9] 遵循了分离式双塔范式。它们分别生成用户和项目的表示，并通过点积计算最终得分 (DeepCoNN 和 D-attn 分别使用 CNNs 和 Transformers [13] 作为其塔)。在此范式中，ANR [3] 扩展了 D-attn 的功能，为用户/项目生成多个表示，并利用用户/项目感知权重来融合它们。上述方法依赖于从头开始训练的小型模型。然而，随着大型语言模型 (LLMs) 的兴起，近期工作倾向于利用预训练的 LLMs 来增强模型能力，因为语义知识提供了泛化的信息。例如，基于 LLM 的方法如 RGCL [11] 使用 BERT [5] 来编码评论，而 LoID [15] 采用 LoRA [7] 进行微调，提供更强大的语义表示。

**动机。**尽管这些方法提高了模型的内容信息能力，但它们忽略了以下问题：

- 无法解释的语义压缩：当前基于 LLM 的推荐系统存在决策路径不透明的问题：它们将文本语义压缩成无法解释的潜在表示，从而模糊了预测过程中的逻辑联系。
- 高计算成本：之前的工作总是以端到端的方式利用大型语言模型 (LLMs)，这导致了巨大的计算需求，主要是因为这些十亿参数规模的模型 (例如 GPT [1])，导致训练时间过长和推理延迟高，使得它们在许多现实场景中不切实际<sup>2</sup>。

**我们的工作。**本文提出了 RecCoT，一个具有两项关键技术贡献的新型推荐框架：

- (1) 缓慢可解释的 CoT 生成通过更大的 LLM：受 NLP 中推理能力的巨大成功启发，我们考虑利用这种训练范式来增强推荐系统，例如 GRPO [10]。然而，在推荐系统中存在一个严重的长尾问题 (例如，在许多数据集中评分为 5 的数据超过 90%)，因此简单的 GRPO 训练策略会遇到奖励操纵问题 [14]。为

<sup>2</sup>使用 NVIDIA T4 GPU 在一个包含 1 亿数据集上训练 LoID 一个 epoch 需要 24 小时。

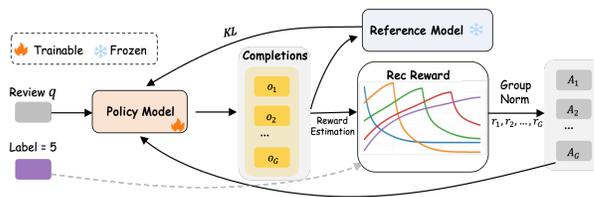


图 1: RecCoT 中使用的 GRPO 整体框架。

此，我们首先提出了一种频率感知的奖励政策来监督更大的 LLM 以适应长尾评分分布。它引入了一个准确性奖励机制以确保可解释的推理生成。

- (2) 快速语义嵌入对齐通过较小的 LLM：由于推理模型始终需要十亿级的参数规模，因此端到端训练范式对于下游推荐系统来说是不可能实现的。因此，在本文中，我们应用缓存范式为下游推荐系统提供语义知识以避免大量计算。为了确保生成的语义嵌入能够与下游任务对齐，我们进一步微调了一个较小的 LLM 来为每个评论生成语义嵌入。得益于我们的慢-快 LLM 设计，我们的推荐系统模型可以大大简化，从而实现轻量级和高效的预测。

**贡献。**我们的贡献总结如下：

- 我们提出了一种解耦推理-执行框架：更大的大语言模型负责可解释的 CoT 生成，而更小的大语言模型则进行高速语义对齐。
- 我们设计了一种奖励策略来适应长尾分布，以确保 CoT 的质量。
- 我们使用最先进的方法进行了详细分析。

## 2 初步的

### 2.1 问题陈述

在这项工作中，我们将评分预测任务视为主要的研究目标。具体地说，给定用户-项目对 ( $u, i$ ) 的观测到的评价-评分 ( $t_{u,i}, r_{u,i}$ ) 作为训练模型的语料库，我们的目标是根据用户的历史评论和其他用户的项目评论来预测其他未观测到的用户-项目的评分分数  $r_{u,i}$ 。

## 2.2 群组

GRPO 是一种强化学习方法，它直接使用同一组多个采样输出的平均奖励作为基线，这显著减少了训练资源的使用并简化了训练过程。具体来说，对于每个问题  $q$ ，GRPO 从旧策略  $\pi_{\theta_{old}}$  中采样一组 CoTs  $o_1, o_2, \dots, o_G$ ，然后通过最大化以下目标函数来优化策略模型：

$$\mathcal{J}_{GRPO}(\theta) = \min \left( \frac{\pi_{\theta}(o_i | q)}{\pi_{\theta_{old}}(o_i | q)} A_i, \text{clip} \left( \frac{\pi_{\theta}(o_i | q)}{\pi_{\theta_{old}}(o_i | q)}, 1 - \epsilon, 1 + \epsilon \right) A_i \right) - \beta \mathbb{D}_{KL}(\pi_{\theta} \| \pi_{\theta_{ref}}), \quad (1)$$

$$\mathbb{D}_{KL}(\pi_{\theta} \| \pi_{\theta_{ref}}) = \frac{\pi_{\theta_{ref}}(o_i | q)}{\pi_{\theta}(o_i | q)} - \log \frac{\pi_{\theta_{ref}}(o_i | q)}{\pi_{\theta}(o_i | q)} - 1, \quad (2)$$

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \dots, r_G\})}{\text{std}(\{r_1, r_2, \dots, r_G\})}. \quad (3)$$

其中  $q$  是输入标记， $o_i$  是 CoT 结果， $A_i$  是在每个组  $\{o_1, \dots, o_G\}$  内的 CoT $_i$  的标准化奖励。通过这种方式，新政策模型可以增加获得更高回报的 CoT 的概率。图 1 说明了本文中 GRPO 的框架。

## 3 方法论

我们的方法 RecCoT 的架构如图 2 所示。在部分 (a) 中，我们利用 LoRA 对更大的 Qwen 进行微调，使其适应特定数据集，并以给定格式输出。在部分 (b) 中，我们将来自部分 (a) 的可解释推理内容与原始评论结合作为输入来微调一个较小的 BERT。生成的结果 CLS 表示随后被缓存到存储器中以支持下游推荐系统 (RecSys)。在部分 (c) 中，这些堆叠的 CLS 向量通过注意力层进行处理，并输入对比学习模块以生成最终的推荐分数。

### 3.1 假设生成。

为了确保在 RecSys 场景下，Qwen 模型在长尾分布中生成高质量的 CoTs。我们首先引入一个多奖励系统，该系统使用三个奖励函数来约束 CoT 遵循 Deepseek R1 [6]。

**格式奖励。**我们实现了基于正则表达式的模式匹配，以验证输出的 CoT 是否符合预定义的 JSON 格式。如果符合预定义的 JSON 格式，则为  $R_{\text{format}} = 1$ ；否则，分配值为 0。

**频率感知准确度奖励。**为了缓解长尾问题，我们设计了一种新的奖励策略，该策略：高频类别的分数会降低，低频类别的分数会增加，并且我们进一步鼓励模型输出浮点数结果以进行相对值比较。具体而言，我们的分段奖励函数定义如下：

$$R_{\text{predict}} = f \cdot [\text{Base}(e) + \text{Penalty}(e)], \quad e = \hat{r} - r, \\ \text{Base}(e) = \exp(-e^2), \\ \text{Penalty}(e) = \begin{cases} -\lambda \cdot (\exp(\mu \cdot |e|) - 1) & e < 0 \\ -\gamma \cdot \left(1 - \frac{\exp(-\kappa \cdot e)}{1 + e}\right) & e \geq 0 \end{cases} \quad (4)$$

其中  $f$  是评分类别的出现频率， $r$  是真实评分， $\hat{r}$  是预测评分， $\lambda$  和  $\mu$  是低估惩罚因子， $\gamma$  和  $\kappa$  是高估惩罚因子。我们的奖励曲线显示在图 1 中。

**分析质量奖励。**通过语义完整性指标量化 CoT 的逻辑深度。

$$R_{\text{quality}} = \frac{\ell_{\text{CoT}} - \ell_{\min}}{\ell_{\max} - \ell_{\min}} \quad (5)$$

其中  $\ell_{\text{CoT}}$  是当前 CoT 文本的长度， $\ell_{\min} = 100$  和  $\ell_{\max} = 200$  分别是预设的最小和最大有效长度阈值。此函数确保 CoT 避免信息冗余并具有足够的分析深度。

最后，我们可以利用它们作为奖励系统来执行 GRPO (在方程 (1-3) 中) 以微调 LLM (例如，Qwen)，为每个评论生成 CoT，并采用 2 倍设置。

### 3.2 语义嵌入对齐

**BERT 软件工具箱。**在获得每篇评论的 CoT 之后，我们下一步将 CoT 和评论融合为一个语义嵌入，并将其缓存到存储器中，以减轻 RecSys 模型的计算压力。具体来说，我们首先对 BERT 进行微调如下：

$$CLS_{u,i} = \text{BERT}(o_{u,i} \oplus t_{u,i}) \quad (6)$$

其中  $o_{u,i}$  和  $t_{u,i}$  是 CoT 和原始评论文本， $CLS_{u,i}$  是压缩后的语义嵌入。之后，我们使用均方误差损失来微调 BERT：

$$\mathcal{L}_{\text{BERT}} = (\hat{r}_{u,i} - r_{u,i})^2, \quad \hat{r}_{u,i} = \text{MLP}(CLS_{u,i}) \quad (7)$$

其中多层感知器是一个投影器， $\hat{r}_{u,i}$  表示预测评分。

**嵌入缓存。**BERT 收敛后，我们接下来缓存每条评论的语义嵌入如下：

$$\text{Cache\_Store}_u \leftarrow (CLS_u^1, CLS_u^2, \dots, CLS_u^m) \\ \text{Cache\_Store}_i \leftarrow (CLS_i^1, CLS_i^2, \dots, CLS_i^n) \quad (8)$$

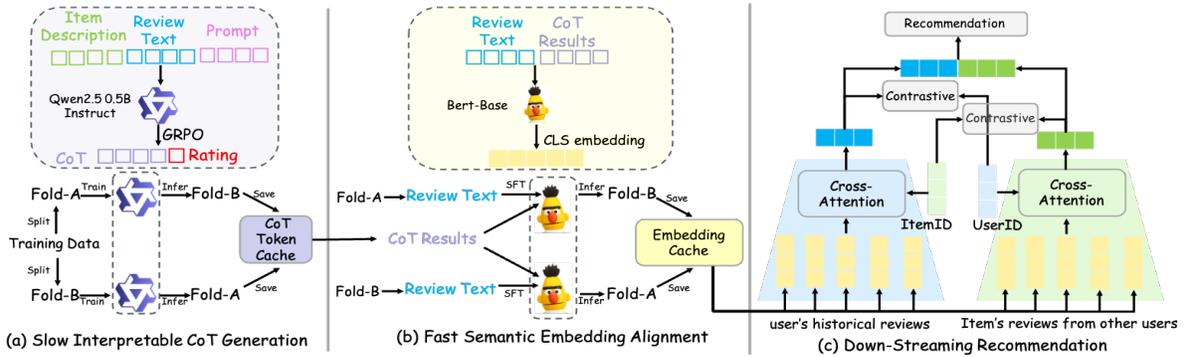


图 2: 所提出的 RecCoT 的级联 workflow。

其中, 缓存\_存储保存所有评论的嵌入,  $CLS_u^*/CLS_i^*$  是相应用户/项目的语义嵌入, 而  $m, n$  表示来自用户  $u$  和项目  $i$  的评论数量。

### 3.3 下游推荐系统

在 RecSys 模型中, 除了缓存\_存储外, 我们可以直接选择语义嵌入而无需重新计算, 如下所示:

$$\begin{aligned} H_u &= (CACHE_u^1, \dots, CACHE_u^m) \leftarrow \text{Cache\_Store}_u \\ H_i &= (CACHE_i^1, \dots, CACHE_i^n) \leftarrow \text{Cache\_Store}_i \end{aligned} \quad (9)$$

其中  $H_u \in \mathbb{R}^{m \times d}$  和  $H_i \in \mathbb{R}^{n \times d}$  是用户/项目的语义嵌入结果,  $d = 768$ 。根据历史语义序列, 我们接下来进行交叉注意力层, 将语义信息  $H_u$  整合到项目 ID 表示  $p_i$  中, 反之亦然用于用户 ID 表示  $p_u$ 。

$$v_i = \text{Attention}(p_i, H_u, H_u), \quad v_u = \text{Attention}(p_u, H_i, H_i) \quad (10)$$

其中  $v_i$  和  $v_u$  是更新后的项目和用户表示。为了建模被注意力忽略的用户-项目关系, 我们应用对比学习。对于每个更新的表示  $v_i/v_u$  (锚点), 其相互作用的用户/项目表示  $p^+$  被视为正例, 而一个随机批内表示  $p^-$  作为负例。我们最小化:

$$\mathcal{L}_{cl} = \max(0, \Delta + D(v, p^+) - D(v, p^-)) \quad (11)$$

其中  $D = \|\cdot\|_2^2$ , 且边界  $\Delta$  强制执行分离鲁棒性。最后, 我们采用均方误差损失来优化下游推荐系统模型, 如下所示:

$$\hat{r}_{u,i} = \text{MLP}(v_u \oplus v_i), \quad \mathcal{L}_{rec} = (\hat{r}_{u,i} - r_{u,i})^2, \quad (12)$$

其中,  $\mathcal{L}_{rec} + \mathcal{L}_{cl}$  用于训练推荐系统模型。

## 4 实验

### 4.1 实验设置

**4.1.1 数据集.** 我们从 Amazon 数据集中选择了 7 类代表性的数据集 [12]. 为了提高数据质量, 将过滤评分少于 5 次的用户/项目, 并清理评论中的噪声字符和短文本。其中, 最小的数据集 Amazon Fashion 包括了 2,035,490 名用户, 874,297 个项目, 以及 2,500,939 个评分 (原始数据) —— 过滤后为 15,520 名用户, 86,316 个项目和 115,464 个评分。最大的数据集 CDs and Vinyl 包含 1,754,118 名用户, 701,706 个项目, 以及 4,827,273 个评分 (原始数据), 过滤后为 183,733 名用户, 534,796 个项目和 2,518,472 个评分。

**4.1.2 参数设置.** 在我们的方法中, 嵌入大小  $d$  是 768, 固定丢弃率为 0.5。在 (a) 部分, 批量大小设置为 4, 学习率设置为  $2e-4$ 。在 (b) 部分, 嵌入大小  $d$ 、丢弃率、学习率、批量大小和周期分别固定为 768、0.5、 $1e-5$ 、4 和 4。在 (c) 部分, 注意力层数量、学习率、批量大小和周期分别固定为 4、 $1e-4$ 、128 和 5。我们使用 Adam [8] 算法更新参数。

### 4.2 性能比较

表 1 显示了 RecCoT 在七个不同数据集上的性能结果。我们可以看到, 我们的模型与 SOTA 模型相比有了显著的改进。可以发现 RGCL 的表现优于 BiGI[2], 这是因为 RGCL 考虑到了评论的因素, 并从用户那里学习更多的表示。此外, LoID 在各个方面都优于 RGCL 和

**表 1: 亚马逊数据集上的实验结果 (均方误差, 越低越好。)**

Datasets	BiGI	RGCL	LoID	RecCoT(w/o CoT)	RecCoT(w/o CoT)+item	RecCoT
Fashion	1.48	1.44	0.68	0.633	0.769	0.570
Baby	1.36	1.38	1.01	0.946	0.947	0.941
CDs	0.95	0.8	0.67	0.620	0.65	0.600
Scientific	1.18	1.4	0.83	0.850	0.826	0.817
Musical	1.19	1.17	0.86	0.892	0.872	0.831
Software	1.74	1.47	1.35	1.206	1.217	1.180
Games	1.46	1.43	1.18	0.949	0.971	0.939

BiGI, 因为它考虑了预训练机制和 ID 对齐, 并在特定领域中微调 BERT。

我们的模型有三种类型, 即不使用任何额外信息、添加项目侧信息特征以及添加 CoT 信息。可以观察到, 在给定项目侧信息特征的情况下, 模型的性能得到了提升。这种性能的提升主要是由于项目元数据在特征表示上的增强: 通过引入诸如项目属性和内容描述等结构化信息, 模型能够构建多维度的项目画像以提供更好的模型性能。

CoT 推理机制的引入显著增强了模型对用户兴趣偏好和项目特征属性进行建模的能力。通过 CoT, 模型可以克服传统协同过滤中固有的静态表示限制。这使得能够从表面行为深入到潜在动机的深层次语义解析, 从而解决了不可解释的语义压缩问题。

### 4.3 奖励政策的讨论

**表 2: 奖励策略性能比较**

数据集/奖励	思考链	预训练	对比学习
Amazon Fashion			
reward	0.8	0.401	0.570
reward-liner	1.1	0.421	0.789
Musical Instruments			
reward	0.61	0.304	0.831
reward-liner	0.66	0.314	0.854

在本节中, 我们研究了不同奖励策略对模型的影响。我们在三个数据集上训练模型, 并且只有奖励策略在各模型间有所不同。(将两折的 MAE 结果取平均作为 (a) 部分的最终结果) 表 2 显示了两个奖励策略

在不同数据集上的各个部分之间的差距。可以发现, 在 (a) 部分, 我们设计的奖励策略显著优于线性奖励机制。这是由于之前提到的奖励作弊问题, 验证了传统线性机制中存在的奖励作弊问题——模型倾向于通过局部最优路径获得短期回报而非全局最优策略。

### 4.4 跨域讨论

**表 3: 跨域性能比较**

数据集	CD	源域
Amazon Fashion	0.580	0.570
Industrial and Scientific	0.85	0.824
Musical Instruments	0.84	0.831

在本节中, 我们探讨了第二部分预训练模型的即插即用功能。我们使用在 CDs 和 Vinyl 领域训练的预训练模型来获取以下三个不同领域的 CLS, 并执行最终的下游任务。表 3 显示基于 CD 领域训练的预训练模型, 在目标领域中无需微调就与源领域基准仅存在极小的性能差异, 这充分验证了该模型在跨领域场景下的强大即插即用特性。

### 4.5 用户评论数量的讨论

本节考察用户交互频率与模型性能之间的关系。Amazon Fashion 数据集显示每个用户的平均互动次数为 7.44 次, 而 Musical Instruments 的平均互动次数更高, 达到 9.44 次。用户参与度分布揭示了不同的模式: Amazon Fashion 有 86.47% 低参与度用户 (<10 次互动), 10.86% 中等 (10-20 次互动), 1.46% 高 (20-30 次互动) 和 1.21% 超级用户 (>30 次互动)。Musical Instruments 显示出更高的参与度多样性, 其中 72.47% 为低参与度用户, 20.97% 中等, 4.08% 高, 2.48% 超级用户。

模型性能与用户参与模式直接相关。Amazon Fashion 在 10 次交互时达到最低 MSE (0.570), 但在 20 次交互时上升至 0.718, 在 30 次交互时则为 0.702。这是因为数据集主要由低参与度用户主导, 超过 10 次交

表 4: 评审计数性能比较

数据集/长度	RecCoT(无 CoT)	RecCoT(无 CoT)+项目	回溯一致性测试
Amazon Fashion			
10	0.633	0.769	0.570
20	0.765	0.779	0.718
30	0.776	0.753	0.702
Musical Instruments			
10	0.892	0.872	0.831
20	0.922	0.94	0.83
30	0.903	0.927	0.79

互的序列通过过度填充引入了噪声。相反, Musical Instruments 展现出了明显的长序列优势: MSE 从 20 次交互时的 0.83 显著下降到 30 次交互时的 0.79。这一改进对应于其大量中等参与度 (20.97%) 和高参与度 (4.08%) 用户基础, 证实了延长序列能增强能力。因此, 最佳性能需要针对每个数据集的参与度定制序列长度。

## 4.6 用户评论长度的讨论

评论长度对模型性能有显著非线性影响。我们选择了具有代表性的 Musical Instruments 数据集来研究这一现象。在最短的评论范围 (0-127 个字符) 内, 模型的表现 MSE 0.2825。随着评论长度增加到 128-255 字符范围, MSE 下降到 0.2464, 在 256-511 字符范围内达到最低点 0.2297。值得注意的是, 当评论扩展到长文本范围 (512-1024 个字符) 时, MSE 显示出显著反弹, 飙升至峰值 0.3501。这一波动趋势表明中等长度的评论 (256-511 字符) 最有利于优化模型性能, 而过长的评论可能会引入冗余信息或增加噪声, 反而降低了建模效率。

## 5 结论

本文介绍了一种简单而有效的名为 RecCoT 的方法, 它包括两个主要组成部分。1) 通过较大语言模型进行缓慢可解释的 CoT 生成, 我们提出了一种奖励约束推理范式。2) 通过较小的语言模型进行快速语义嵌入对齐, 我们使用 BERT 来获取语义嵌入, 并使用对比学习进行对齐。广泛的实验表明 RecCoT 超过了现有的

SOTA 方法, 深入分析突显了我们模型组件的有效性。在未来, 我们将找到更多策略以防止奖励操纵。

## 6 假设案例

**评审:** 对于这个价格来说它只是一个普通的背带, 你可以在任何登山包上看到这种背带。看起来一点也不像吉他背带……也太细了。

**假设分析:** 这篇评论对产品的材料质量和价格持负面和批评态度。它强调这不是一个物有所值的产品, 并且将其与其它类似产品做了不利的比较。评论者似乎对这款产品感到失望。

**用户评分:** 1.0

**预测:** 1.8

**审阅:** 价格不错, 质量一般。尺寸稍微偏小。

**假设链:** 基于给定的用户反馈, 该产品收到了褒贬不一的评价。客户表示价格不错, 但质量似乎不尽如人意。此外, 他们还提到尺寸略小。虽然由于价格合理, 最初的印象可能不会特别负面, 但对尺寸的小批评表明了一些不满。

**用户评分:** 3.0

**预测:** 3.7

**评审:** 连一个储物口袋都没有, 拉链位置也不好。颈套对于标准古典调音键来说太紧了。看起来是为网球拍设计的。

**链式思考:** 用户对产品的设计提供了一些负面反馈, 特别是关于储物口袋和颈部袖套。他们提到这个关键看起来是为网球拍设计的, 并建议可能不适合他们的需求。

**用户评分:** 1.0

**预测:** 2.5

**评审:** 更新 4.12.16br 嗯, 自从我开始断断续续地练习到现在已经快一年了! 天啊! 我还很喜欢这只名为 34bacon34 的鲨鱼。花了大约一周或两周的时间让这把尤克里里保持调音, 但现在, 即使几周不弹它也大部分情况下还能在调上。最多只需要稍微调整一下一两个调音器。br 我在这里读了几条评论说它的木材是木制的。不是木制的。指板可能是... 可能是琴颈。但琴身是一个树脂塑料模具。这可以解释为什么声音如此明亮。现在, 我有了可以比较的东西——我的兰尼凯名为“king”的尤克里里。我发现了一些区别因素。其中一个就是声音。br 就像我说的, 它的声音非常 34 明亮 34, 如果你愿意这么说的话。它发出的声音非常好。br 弦对我来说很棒, 话说回来, 我的两把尤克里里的弦都是 Aquila 品牌的, 我对这个很满意。br 它有各种各样的颜色。我喜欢它们有趣的颜色和设计。br 质量, 而不仅仅是价格实惠——在我选择这把尤克里里之前做了关于价格点和质量的研究。由于我是即兴开始的, 我不想在这把可能在一个月左右就失去兴趣的尤克里里上花太多钱。我对这把尤克里里的质量仍然感到惊喜。br 我相信经验丰富的尤克里里手可能会不同意, 但作为一个成年初学者, 我认为这是我做的最好的购买决定之一。我后来换过了这把尤克里里, 但总是会回到它身边。尤其是在学习一首对我来说较难的新歌时。似乎我在“bacon”上更容易找到曲调。br br 无论如何, 在将近一年后, 我还是强烈推荐这款, 并认为这是开始使用的好质量尤克里里。你不必更换新的弦, 因为它已经配备了 Aquila 的弦。记得, 就像任何新乐器一样, 需要一些时间来磨合并适应你的演奏方式。br br 更新 br 我现在大概一个月断断续续地练习这把尤克里里。我认为在连续调音几天后, 它基本上保持了调音。我只需要偶尔做些微调。对我来说, 声音很棒, 但我没有可以比较的。它摸起来不廉价 34 塑料感。34 绝对是所有人的绝佳入门尤克里里。br br 到目前为止这仅仅是基于第一印象。在我回家之前实际上无法试用... 但是 它的声音非常美丽 深沉且宽广 不像玩具。弦是白色的 Aquila 弦, 所以我不需要购买任何弦来更换。尤克里里的绿色比在线上显示的 34 海泡绿 34 更深。这个特

**审阅:** 性价比很高, 表现非常好。

**假设过程:** 这是一篇关于一个产品的好评, 该产品效果不错且物有所值。评论者认为它非常适合预算, 这非常积极。然而, 由于一些小的批评, 总体情感显得中等。

**用户评分:** 4.0

**预测:** 3.8

**评审:** 这不是一笔好买卖。在几天内, 这座桥被绷紧到正常张力时, 这座设计得非常薄且以一个容易断裂的角度设置的桥……嗯, 桥裂开了, 并从乐器上飞了出去。我惊慌失措, 因为除了这一点外, 这把乐器的表现完全符合我的期望, 于是我立刻订购了另一把。重复的历史, 在被绷紧到正常张力后的几个小时内, 这座桥已经开始出现裂缝。我是专业音乐人依赖我的乐器来表演相当不可接受, 以前从未遇到过尤克里里的失败。

**链式思考:** 该评论表达了对产品的强烈不满, 称其为“糟糕的购买”, 原因是存在裂纹、爆炸和结构失效等问题。评论者提到他们之前依赖这款设备进行表演, 表明他们对该产品有很高的信任度。

**用户评分:** 5.0

**预测:** 3.8

**评审:** 是一种单层的 12 密耳薄膜, 相比传统的 10 密耳单层膜头, 它在低频响应、冲击力和功率方面有显著提升。适合重击者追求扎实音质和更长久耐用的膜头, 但仍然偏好单层的感觉。

**链式思考:** 这篇评论似乎对产品的性能持中立态度。评审者提到, 与传统的 10 密耳单层鼓头相比, 它提供了更强的低端、冲击力和功率。他们还推荐它作为适合重击手的理想选择, 这些重击手更看重坚实的声音和更多的生命力。然而, 他们并没有表现出任何具体的积极或消极情绪。

**用户评分:** 5.0

**预测:** 3.5

**评审:** 是一笔浪费的钱。

**连贯文本:** 这是一条负面评价, 提到产品毫无价值且性价比极低。用户似乎对自己的体验感到失望, 并没有分享任何积极的方面。**用户评分:** 1.0

**预测:** 1.8

**评审:** 需要一些 nl 电缆进行安装。这些正好符合要求。制作质量很好。电缆看起来质量不错。请注意, 这些只是带有 nl4 连接器的两芯电缆。这些可以在被动扬声器上使用, 但不能在双功放扬声器上使用。推荐购买。

**链式思考:** 本评论提供了一些关于产品特点和用户满意度的基本信息。评论者提到, 需要一些 NL 线缆进行安装, 并且这些线缆似乎满足了被动扬声器的需求。然而, 它也指出, 由于导体数量有限, 这些线缆可能不适合双功放的扬声器。

**用户评分:** 5.0

**预测:** 3.8

**评审:** 首先, 包本身制作得很好且非常有用。然而, 一个很小但非常重要细节彻底毁了它——廉价的拉链在角落处卡得很厉害。第一次拉开包时, 一个拉链弯折并掉了下来。真的吗? 一百元的包, 这就是你们能做到的最好的拉链吗? 不推荐购买。

**假设分析:** 该评论表达了对产品拉链质量的不满, 特别提到了其设计不佳。顾客表示由于拉链的质量差使用包非常困难, 完全破坏了他们的体验。他们还提到在拉开其中一个特定的拉链时拉链脱落的小问题。

**用户评分:** 2.0

**预测:** 1.5

## References

- [1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).
- [2] Jiangxia Cao, Xixun Lin, Shu Guo, Luchen Liu, Tingwen Liu, and Bin Wang. 2021. Bipartite graph embedding via mutual information maximization. In *Proceedings of the 14th ACM international conference on web search and data mining*.
- [3] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based neural recommender. In *ACM International Conference on Information and Knowledge Management (CIKM)*.
- [4] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv* (2018).
- [6] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).
- [7] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *ArXiv* (2021).
- [8] P. Diederik Kingma and Lei Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*.
- [9] Sungyong Seo, Jing Huang, Hao Yang, and Yan Liu. 2017. Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In *ACM Conference on Recommender Systems*

- (RecSys).
- [10] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300* (2024).
- [11] Jie Shuai, Kun Zhang, Le Wu, Peijie Sun, Richang Hong, Meng Wang, and Yong Li. 2022. A review-aware graph contrastive learning framework for recommendation. In *ACM International Conference on Research on Development in Information Retrieval (SIGIR)*.
- [12] Brent Smith and Greg Linden. 2017. Two decades of recommender systems at Amazon. com. *IEEE International Computing* (2017).
- [13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances In Neural Information Processing Systems* (2017).
- [14] Lilian Weng. 2025. Why We Think. *lilianweng.github.io* (May 2025). <https://lilianweng.github.io/posts/2025-05-01-thinking/>
- [15] Wentao Xu, Qianqian Xie, Shuo Yang, Jiangxia Cao, and Shuchao Pang. 2024. Enhancing content-based recommendation via large language model. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*.
- [16] L. Zheng, V. Noroozi, and P. S. Yu. 2017. Joint Deep Modeling of Users and Items Using Reviews for Recommendation. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM)*. 425–434.