arXiv:2601.08764v1 [cs.IR] 13 Jan 2026

FusID：用于生成式音乐推荐的模态融合语义 ID

Haven Kim Yupeng Hou JulianMcAuley加
利福尼亚大学圣迭戈分校

摘要

生成式推荐系统通过利用语义 ID 表征物品，取得了显著进展。然而，现有的、对每种模态分别进行词元化的方法面临两个关键局限：（1）模态之间存在冗余，降低了效率；（2）无法捕捉模态间交互，限制了物品表征能力。我们提出 FusID，这是一种模态融合语义 ID 框架，通过三个关键组件解决上述局限：（i）多模态融合，通过联合编码不同模态的信息学习统一表征；（ii）表征学习，在保持差异性并防止特征冗余的同时，使频繁共现的物品嵌入彼此更接近；以及（iii）乘积量化，将融合后的连续嵌入转换为多个离散词元，以缓解 ID 冲突。在多模态下一首歌曲推荐（即播放列表续播）基准数据集上的评估表明，FusID 实现了零 ID 冲突——确保每个词元序列恰好映射到一首歌曲——并缓解了码本利用不足，同时在平均倒数排名和 Recall@k (k=1,5,10,20)方面优于基线方法。

1 引言

推荐系统日益采用生成式范式，将推荐视为序列生成任务 [19, 24, 5, 11, 9, 10]。这一范式转变的核心是使用语义 *ID*（SID）——这类离散词元序列通过将连续表征（通常由物品编码器生成）映射到离散码本中，为自回归语言模型编码物品。在音乐领域，语义 ID 成功利用与单曲相关的物品模态提取物品词元序列，并将其用于设计生成式下一首歌曲推荐（即播放列表续播） [16]和对话智能体 [6, 7]。

尽管前景广阔，但在多模态场景中实现基于 SID 的生成式推荐系统仍面临重大挑战。首先，现有方法 [6, 7] 为每种模态（例如音频、标签、歌词）分别训练分词器，独立编码相似的语义信息，导致同一信息被多次编码。这使学得的码本中存在大量冗余，造成词汇表规模不必要地增大，不仅加重语言模型的负担，还增加了训练和推理过程中的计算开销。其次，这些方法将各种模态孤立处理，未能捕捉同一物品不同视图之间丰富的模态间交互 [2, 23] 与互补信息；例如，音频特征传达的情感基调可能与文本中的歌词主题相契合，但独立编码无法利用这类协同组合 [20] 来构建更全面、更具判别性的物品表征。这些局限表明，我们需要一种统一框架，能够联合编码多模态信息，同时消除冗余并捕捉不同模态之间的协同关系。

在本文中，我们提出了 **FusID** （模态**融**合语义**ID**），这是一种新颖的框架，能够学习模态融合语义 ID，以应对上述挑战。如图1所示，

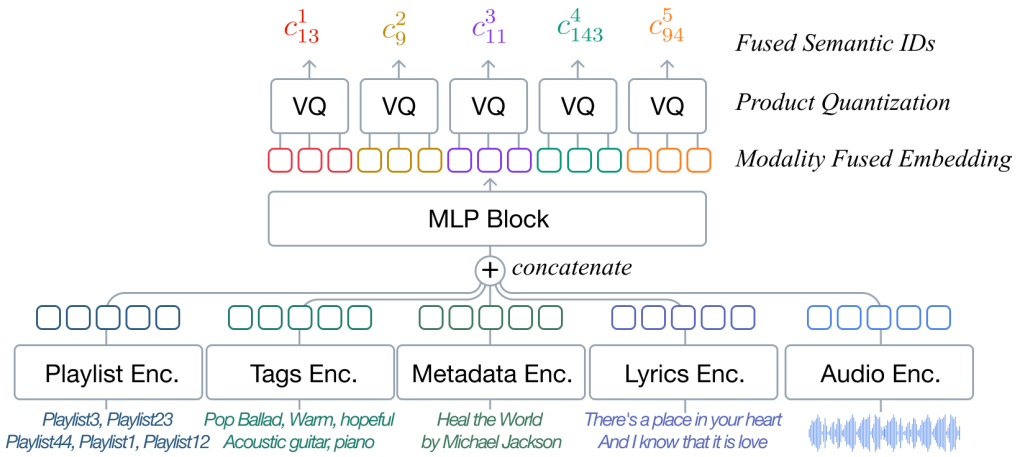


图1：模态融合语义 ID 概览。

FusID 包含三个阶段：（*1*）多模态融合模块：我们设计了一种简单的融合网络，对多种模态进行联合处理，以学习模态间交互。（*2*）表征学习：我们采用表征学习目标，使不同上下文中频繁共现的物品的嵌入彼此接近，同时拉远不频繁共现的物品的嵌入。此外，我们引入正则化损失，确保所有嵌入元素都承载重要信息，并防止词元之间出现特征重叠，从而保持所学习表征的 distinctiveness 并避免冗余。（*3*）乘积量化：我们应用乘积量化，将融合嵌入分解到多个子空间中，生成具有完美唯一性的复合语义 ID，实现零 ID 冲突——确保每个词元序列都恰好映射到一个物品。

2 FusID：模态融合语义 ID

给定一个具有 m 种模态的物品（例如播放列表上下文、标签、元数据、歌词和音频），我们首先使用预训练编码器提取模态特定特征 $M_1, \ldots, M_m$。随后，FusID 使用单个融合网络，同时接收拼接后的 $(M_1, \ldots, M_m)$，并生成融合嵌入矩阵 $E \in \mathbb{R}^{n \times d}$，其中每一行 $e_i \in \mathbb{R}^d$ 对应于该物品的第 i 个语义子嵌入。之后，这些 n 子嵌入通过乘积量化（第2.2节）转换为 n 语义 ID，从而构成每个物品的最终词元序列。

2.1 模态融合模块与表征学习

为整合不同模态的信息，我们采用单个融合网络 f_θ。该网络以 m 模态特征 $(M_1, \ldots, M_m)$ 为输入，其中每个 M_i 代表物品的第 i 个模态，并输出融合嵌入 $E = f_\theta([M_1; \ldots; M_m])$。我们将 E 视为 n 个不同的嵌入 $\{e_1, \ldots, e_n\}$，其中每个 $e_i \in \mathbb{R}^d$ 之后都会进行量化。融合网络通过两种类型的损失函数进行训练：对比损失促使频繁共现的物品在不同上下文中的嵌入彼此接近，同时使很少共现的物品的嵌入彼此分离；正则化损失则用于防止 n 子嵌入之间出现冗余和坍塌。

对比损失：我们采用对比损失，最小化正样本的能量，同时最大化负样本的能量 [15],，以促使频繁共现物品的嵌入彼此接近，并拉远不常共现物品的嵌入。给定一对物品（i，j）及其融合嵌入 E_i 和 E_j，我们计算它们之间的欧几里得距离 $D(E_i, E_j)$。令 $y \in \{0, 1\}$ 表示（i，j）是正样本对（$y = 1$）还是负样本对（$y = 0$）。对比损失定义如下：

$$\mathcal{L}_{\text{cont}} = \text{MSE}\big(|1 - y|,\ D(E_i, E_j)\big) \tag{1}$$

在进行对比对挖掘时，我们使用归一化共现，将得分较高的配对视为正样本，将得分较低的配对视为负样本。

正则化损失： 为避免冗余并确保所有子嵌入都包含有用信息，我们采用受 VICReg [3] 启发的正则化损失 $\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{cov}} + \mathcal{L}_{\text{var}}$。协方差损失 $\mathcal{L}_{\text{cov}}$ 通过惩罚子嵌入 $\{e_1, \ldots, e_n\}$ 之间的相关性，鼓励 n 子嵌入捕捉互补信息。由于我们的目标是避免每个嵌入内部的特征重叠（$E_i = e_1$，……，e_n），而非采用 Bardes 等人 [3], 使用的逐元素惩罚，因此我们使用 (n) 个分块之间的分块交叉协方差项（即考虑 $\binom{n}{2}$ 对）：

$$\mathcal{L}_{\text{cov}} = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i<j}}^{n} \frac{1}{d} \left\| \text{Cov}(e_i, e_j) \right\|_F^2 \tag{2}$$

其中，Cov(e_i， e_j) 是子嵌入 e_i 和 e_j 之间的交叉协方差矩阵。

方差损失 $\mathcal{L}_{\text{var}}$ 确保每个特征维度都保持足够的方差，并防止嵌入坍缩为平凡解（例如，所有向量都变为零或常数），否则协方差损失也会平凡地变为零 [3]:

$$\mathcal{L}_{\text{var}} = \frac{1}{n \times d} \sum_{c=1}^{n \times d} \max\left(0,\ \gamma - \sqrt{\text{Var}(e_{\cdot,c}) + \epsilon}\right) \tag{3}$$

其中，$\gamma = 1$ 是目标标准差，ϵ 是用于保证数值稳定性的小常数。总训练目标结合了这两种损失：$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cont}} + \alpha\, \mathcal{L}_{\text{reg}}$，其中 α 控制正则化强度。我们在所有实验中都将 $\alpha = 0.2$设为该值。

2.2 量化模块

训练完融合网络后，我们使用乘积量化将连续的融合嵌入转换为离散的语义 ID。对于每个项目，我们取融合嵌入矩阵 $E \in \mathbb{R}^{n \times d}$，并将其 n 行视为子嵌入（e_1、……、 e_n），每个子嵌入位于 $\mathbb{R}^d$ 中（其中 $d = 128$）。利用训练集中包含的歌曲，在相应子空间中通过 k-均值聚类，将每个子嵌入e_i 分配到 $K = 1024$ 个簇之一。当 e_i 被分配到第 x_i个簇（$x_i \in \{1, \ldots, 1024\}$）时，它会被标记为$c_{i,\ x_i}$。

因此，单首歌曲表示为由 n 词元组成的序列（$c_{1,\ x_1}$、……、 $c_{n,\ x_n}$），其中每个位置i 都拥有一个大小为 1024 的码本。这种乘积量化设计产生了巨大的组合语义 ID 空间，同时确保每个词元序列恰好映射到一个项目，从而消除 ID 冲突，并支持基于融合多模态表示的可扩展生成式推荐。

3 实验

数据集： 我们选择的数据集是 Million Playlist 数据集（MPD） [4]。尽管原始 MPD 包含 2,262,292 首唯一歌曲，但我们仅为其中的 537,042 首歌曲获取了完整的特征嵌入（标签、元数据、歌词和音频）。因此，我们过滤掉包含少于六首有效歌曲的播放列表，因为我们的下游任务需要足够的上下文来生成序列推荐。经过这一处理，我们得到 926,689 个播放列表。在这个经过筛选的数据集中，每个播放列表平均包含 36.81 首歌曲。最后，我们将数据划分为训练集（80%）、验证集（10%）和测试集（10%）。

评估： 我们从两个角度评估所提出的方法：（1）语义 ID 质量；（2）生成式推荐性能。对于语义 ID 质量，我们评估三项关键指标：（1）码本利用不足率（*CUR*），用于衡量未使用的码本条目所占的比例；（2）基数，用于统计语义 ID 可表示的不同物品数量；（3）冲突率，用于衡量与其他物品共享相同语义 ID 的物品所占的比例。对于生成式推荐性能，我们评估使用这些语义 ID 训练的语言模型在生成式推荐任务上的表现。遵循先前的研究 [13, 8],，我们使用平均倒数排名和 Recall@k（其中 $k = 1$、5、10、20）衡量性能，以评估模型预测下一首歌曲的能力。具体而言，对于长度为 l 的播放列表，当 $l \leq 30$ 时，我们使用前 $l - 1$ 首曲目作为上下文；否则使用前 30 首曲目。播放列表中的下一首曲目被视为相关的真实物品。

表 1：语义 ID 评估。

	$\mathcal{L}_{\text{cont}}$	$\mathcal{L}_{\text{reg}}$	码本利用不足率（↓） All	码本利用不足率（↓） Test	基数（↑） All	基数（↑） Test	冲突率（↓） All	冲突率（↓） Test
基线 [6]	-	-	0.43	11.78	536080	192836	0.31	0.27
FusID（消融实验）	✓	✗	0.00	0.00	514929	178661	6.23	11.02
FusID（本文方法）	✓	✓	0.00	0.02	537042	193124	0.00	0.00

基数上限：*537042*（全部），*193124*（测试集）

实现细节： 遵循先前的研究 [7],，我们提取了四种模态的嵌入：标签（$M_{\text{tag}} \in \mathbb{R}^{1024}$）、元数据（$M_{\text{metadata}} \in \mathbb{R}^{1024}$）、歌词（$M_{\text{lyric}} \in \mathbb{R}^{1024}$）和音频（$M_{\text{audio}} \in \mathbb{R}^{512}$）。其中，$M_{\text{tag}}$、$M_{\text{metadata}}$和 M_{lyric} 使用预训练文本编码器 [22], 提取，而M_{audio} 则使用预训练音频编码器 [21]提取。第五种模态——播放列表共现（$M_{\text{playlist}} \in \mathbb{R}^{128}$）——则按照先前研究 [6]的方法获得：使用 word2vec [17] 风格的嵌入模型，并基于训练集，将每首曲目视为一个“词”、将播放列表视为“文档”，从而捕捉歌曲之间的隐式关系。

输入向量 $M_{\text{concat}} \in \mathbb{R}^{3072}$由五个不同的向量拼接而成，经由投影头处理后生成最终的融合嵌入 $E \in \mathbb{R}^{n \times 128}$。该投影头由一个将维度从 3072 映射到 2048 的线性层、批归一化 [12] 和 ReLU 激活函数组成。随后，另一个线性层将这个 2048 维表示投影为 $n \times 128$输出，并经过层归一化 [1]。在量化过程中，我们为每个 n 子维度使用 1024 个簇（$n = 5$）。FusID 使用批次大小为 128、学习率为 5e-4 进行训练。对于生成式推荐任务，我们在语义 ID 序列上训练了一个 GPT-2 [18] 风格的仅解码器结构，并通过根据给定的前置上下文对剩余曲目进行排序，评估其对播放列表续播的建模能力。

4 结果

多模态语义 ID： 表 1 展示了我们提出的 FusID 方法与基线方法在码本质量上的比较结果。结果表明，同时采用对比损失和正则化损失的 FusID 在所有指标上都取得了近乎完美的表现。具体而言，它在完整数据集上的码本利用不足率（CUR）为 0%，在测试集上的 CUR 为 0.02%，这意味着仅有一个码本条目未被使用。此外，该方法成功地为数据集中的全部 537,042 首歌曲编码（即基数完美），且冲突数为零。这表明每个项目都获得了唯一的语义 ID，从而消除了生成式推荐过程中的歧义。

相比之下，消融实验凸显了正则化损失的关键作用。仅采用对比损失时（FusID 消融），模型在完整数据集和测试集上的码本利用不足率（CUR）均为零；然而，它在完整数据集的 537,042 首歌曲中仅为 514,929 首编码，在测试集的 193,124 首歌曲中仅为 178,661 首编码，完整数据集和测试集的冲突率分别为 6.23% 和 11.02%。
基线方法 TalkPlay 在码本利用不足率（CUR）方面的性能显著下降，在完整数据集和测试集上的数值分别为 0.43% 和 11.78%。此外，它还表现出语义 ID 冲突：在完整数据集和测试集中，冲突率分别为 0.31% 和 0.27%，基数分别为 537,042 首歌曲中的 536,080 首，以及 193,124 首歌曲中的 192,836 首。

总体而言，这些结果表明，我们提出的方法能够有效学习高质量的语义 ID，并提升生成式推荐性能。

生成式推荐： 表 2 展示了我们提出的方法与基线方法之间的生成式推荐性能对比。我们选择了两种基线方法：SASRec [14], ，一种自回归序列推荐模型，以及 TalkPlay [6]。作为消融实验，我们还将 FusID 与一个未使用正则化损失进行训练的变体进行了比较。

结果表明，FusID 在所有指标上始终优于所有基线方法。具体而言，与表现最佳的基线方法 TalkPlay 相比，FusID 的性能有所提升，平均倒数排名提高了 6.21%，Recall@k 分别提高了 6.32%、6.46%、6.27% 和 6.09%。这些结果验证了 FusID 的多模态

表 2：生成式推荐评估。

	MRR	Recall@k			
		k=1	k=5	k=10	k=20
SASRec [14]	2.60	1.05	4.02	6.85	9.52
TalkPlay [6]	9.02	6.96	11.61	13.56	14.79
FusID（消融实验）	7.54	5.90	9.53	11.06	12.51
FusID（我们的方法）	**9.58**	**7.40**	**12.36**	**14.41**	**15.69**

融合方法生成的高质量语义 ID 能够实现更有效的生成式推荐，因为语言模型在通过无冲突、分布均衡的语义 ID 序列进行训练时，能够更好地捕捉物品之间的语义关系。

然而，在不使用正则化损失进行训练时，FusID 的表现不及 TalkPlay，但仍优于 SASRec。与完整模型相比，这一消融变体的性能显著下降：平均倒数排名降低了 21.29%，Recall@k 分别降低了 20.27%、22.90%、23.25% 和 20.27%。这些发现凸显了正则化损失在训练 FusID 模块中的关键作用。

5 结论

本文介绍了 FusID，这是一种多模态语义 ID 生成框架，旨在解决现有生成式推荐方法中的关键局限。FusID 通过对比学习目标和正则化损失融合多种模态——包括标签、元数据、歌词、音频和播放列表共现——生成高质量、无冲突且码本利用率近乎完美的语义 ID。我们在 Million Playlist 数据集上进行的实验表明，FusID 不仅能生成质量更高的语义 ID，还能显著提升下游生成式推荐的性能，并在所有评估指标上持续取得提升。这些结果凸显了多模态融合和无冲突语义表示对于构建有效的生成式推荐系统的重要性。

参考文献

[1] J. L. B a, J. R. Ki ros, an d G. E. Hi nton. L ayer norma lization. *arXi预印本i文章Xiv:1607.06450*, 2016年。

[2] T. Baltrušaitis、C. Ahuja 和 L.-P. Morency。多模态机器学习：综述与分类。*IEEE模式分析与机器智能汇刊*，2018年。[3] A. Bardes、J. Ponce 和 Y. LeCun。VICReg：用于自监督学习的方差-不变性-协方差正则化。*arXiv预印本 arXiv:2105.04906*，2021年。[4] C.-W. Chen、P. Lamere、M. Schedl 和 H. Zamani。RecSys挑战赛2018：自动音乐播放列表续播。载于第*12*528，2018年527届*ACM推荐系统会议*，第–页。[5] Y. Deldjoo、Z. He、J. McAuley、A. Korikov、S. Sanner、A. Ramisa、R. Vidal、M. Sathiamoorthy、A. Kasirzadeh 和 S. Milano。使用生成模型的现代推荐系统综述（Gen-RecSys）。载于第*30*届*ACM SIGKDD*知识发现与数据挖掘会议论文集，第6448–6458页，2024年。[6] S. Doh、K. Choi 和 J. Nam。TalkPlay：使用大语言模型的多模态音乐推荐。*arXiv*预印本 *arXiv:2502.13713*，2025年。[7] S. Doh、K. Choi 和 J. Nam。TalkPlay-Tools：通过大语言模型工具调用实现对话式音乐推荐。*arXiv*预印本 *arXiv:2510.01698*，2025年。[8] N. Hariri、B. Mobasher 和 R. Burke。基于潜在主题序列模式的上下文感知音乐推荐。载于第六届*ACM推荐系统会议论文集*，第131 138页，2012年 –。[9] Y. Hou、J. Ni、Z. He、N. Sachdeva、W.-C. Kang、E. H. Chi、J. McAuley 和 D. Z. Cheng。ActionPiece：对动作序列进行上下文分词以实现生成式推荐。国际机器学习大会，2025年。

[10] Y. Hou、A. Zhang、L. Sheng、J. Wu、X. Wang、T.-S. Chua 和 J. McAuley。迈向大型生成式推荐：分词视角。载于第*34*届*ACM*国际信息与知识管理会议论文集，CIKM '25，第6821–6824页，美国纽约州纽约市，2025年。计算机协会。[11] Y. Hou、A. Zhang、L. Sheng、Z. Yang、X. Wang、T.-S. Chua 和 J. McAuley。生成式推荐模型：进展与方向。载于*2025*年*ACM Web*大会会议论文集，第13–16页，2025年。[12] S. Ioffe 和 C. Szegedy。批归一化：通过减少内部协变量偏移加速深度网络训练。载于国际机器学习会议，第448–456页。pmlr，2015年。[13] D. Jannach、L. Lerche 和 I. Kamehkhosh。超越"热门歌曲"：使用恰当曲目生成连贯的音乐播放列表续播。载于第*9*届*ACM*推荐系统会议论文集，第187–194页，2015年。[14] W.-C. Kang 和 J. McAuley。自注意力序列推荐。载于*2018*年*IEEE*国际数据挖掘会议（*ICDM*），第197–206页。IEEE，2018年。[15] Y. LeCun。迈向自主机器智能：0.9版。2，2022-06-27。*Open Review*，62(1):1–62，2022年。[16] M. J. Mei、F. Henkel、S. E. Sandberg、O. Bembom 和 A. F. Ehmann。用于音乐推荐的语义 ID。载于第十九届*ACM*推荐系统会议论文集，第1070 1073页，2025年 –。[17] T. Mikolov、K. Chen、G. Corrado 和 J. Dean。在向量空间中高效估计词表示。*arXiv*预印本 *arXiv:1301.3781*，2013年。[18] A. Radford、J. Wu、R. Child、D. Luan、D. Amodei、I. Sutskever 等。语言模型是无监督的多任务学习者。*OpenAI*博客，1(8):9，2019年。[19] S. Rajput、N. Mehta、A. Singh、R. Hulikal Keshavan、T. Vu、L. Heldt、L. Hong、Y. Tay、V. Tran、J. Samost 等。采用生成式检索的推荐系统。神经信息处理系统进展，36:10299–10315，2023年。[20] H. Wang、J. Qin、W. Wen、Q. Li、S. Zhong 和 Z. Huang。用于多模态推荐的多模态表示解耦信息瓶颈，2025年。[21] Y. Wu、K. Chen、T. Zhang、Y. Hui、T. Berg-Kirkpatrick 和 S. Dubnov。通过特征融合与关键词到字幕增强进行大规模对比语言-音频预训练。载于*ICASSP 2023-2023 IEEE*国际声学、语音与信号处理会议（*ICASSP*），第1–5页。IEEE，2023年。[22] A. Yang、A. Li、B. Yang、B. Zhang、B. Hui、B. Zheng、B. Yu、C. Gao、C. Huang、C. Lv 等。通义千问3技术报告。*arXiv*预印本 *arXiv:2505.09388*，2025年。[23] F. Zhao、C. Zhang 和 B. Geng。深度多模态数据融合。*ACM*计算调查，2024年。[24] B. Zheng、Y. Hou、H. Lu、Y. Chen、W. X. Zhao、M. Chen 和 J.-R. Wen。通过整合协同语义使大语言模型适应推荐。载于*2024*年*IEEE*第*40*届国际数据工程会议（*ICDE*），第1435–1448页。IEEE，2024年。