arXiv:2305.01051v1 [cs.SD] 1 May 2023

LooPy：面向电子舞曲音乐信息检索的便于研究的混音框架

Xinyu Li 纽约
大学 美国纽约
xl3133@nyu.edu

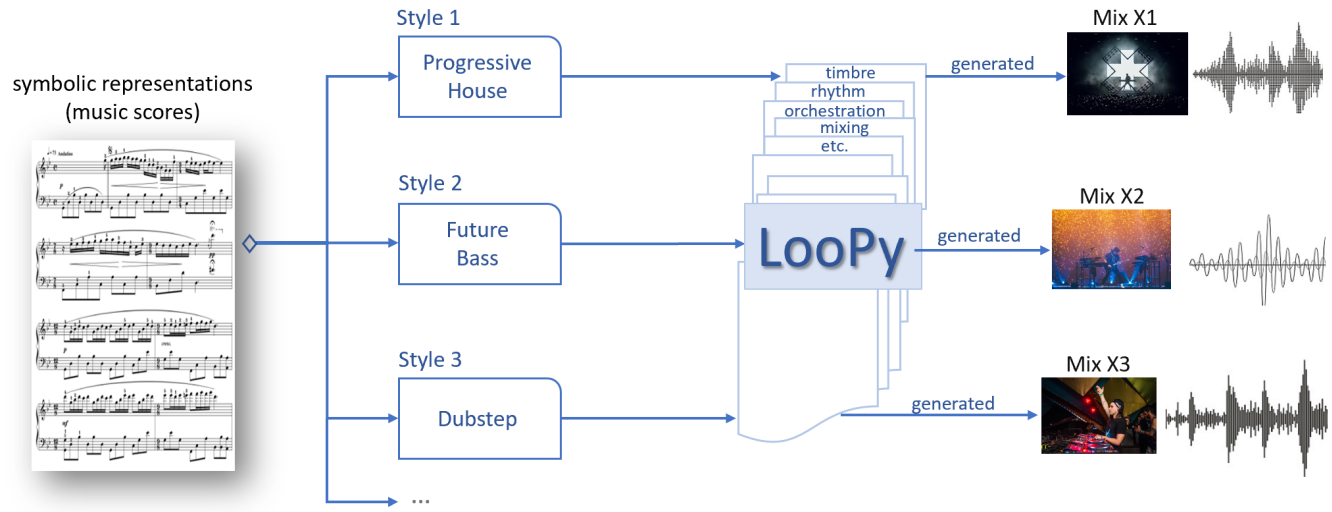


图 1：我们的软件包“LooPy”的概念示意图。

摘要

近年来，随着深度学习的发展，音乐信息检索（MIR）取得了飞速进展。然而，与其他音乐类型相比，电子舞曲（EDM）一直较少受到研究。鉴于其广泛的应用场景，我们推出了一个用于自动生成电子舞曲音频的 Python 软件包，作为电子舞曲歌曲音乐信息检索的基础设施，以降低获取带标签数据的难度。它是一款便捷的工具，可轻松接入许多符号音乐生成流程的末端。在该软件包中，我们提供了一个框架，可根据指定的旋律和和弦构建专业级模板并生成制作精良的音轨；也可通过概率符号旋律生成器，仅给定特定调性便生成大量音轨。实验表明，无论从主观还是客观标准来看，我们生成的混音都能达到世界知名艺人发行的原版参考歌曲的同等品质。我们的代码可在此代码库中获取[1]，项目官方网站也已上线[2]。

CCS 概念

• **信息系统** → **音乐检索**； • **应用计算** →**声音与音乐计算**。

关键词

合成数据生成器，音乐信息检索

1 引言

人工智能驱动的音乐生成一直是热门研究课题。符号音乐生成经历了迅猛发展，如今仍在持续发展，相关研究（如 [11, 12, 24]）既提出了一些新方法，也探索了其部分应用。从 LSTM 等简单的序列模型 [16],、变换器 [25, 27]，到更复杂的模型，包括 Music VAE [22],、生成对抗网络（GAN） [8] 和扩散模型 [19],，这些模型或以监督方式训练，或通过强化学习训练[14],，能够生成达到人类作曲家水平的符号乐谱。然而，音乐音频生成的发展相对滞后，因为鲜有研究成功构建出可扩展的框架，用于编曲和混音，并达到商业制作的水准。因此，要自动为音乐音频生成下游音乐信息检索任务所需的标签相对困难。我们认为，符号音乐生成的发展使得利用预先设定的编曲与混音模板合成数据成为一种颇具前景的选择。只要格式适当对齐，研究人员便可借助此类模板轻松合成大量专业或准专业的电子舞曲歌曲。由于符号音乐表示可以直接获取，这些歌曲天然带有标注。

[1]代码：https://github.com/Gariscat/loopy[2]网站：https://loopy4edm.com

我们的软件包旨在构建一个基于 Python 的电子舞曲数据生成器模板框架，将生成的符号音乐转换为制作精良的音频文件。鉴于 House 音乐在电台和音乐节中广受欢迎，我们目前专注于 House 音乐模板。我们提供的演示模板能够生成达到专业水准的电子舞曲曲目，同时兼顾客观标准（如响度等）和人类听众的主观评价。需要指出的是，本研究面向的是判别式音乐信息检索任务，而非生成用途，~~不过也可~~能有助于人类音乐家创作歌曲。我们将在下一节详细讨论研究动机。简而言之，我们的主要贡献有以下三点：

(1) 我们推出了一个开源的、基于 Python 的电子舞曲生成框架，其中明确包含乐谱和混音器通道信息。该框架既可作为用于下游音乐信息检索任务（如音源分离和转录等）的数据合成器，也可作为音乐生成及其应用的潜在工具。(2) 我们以该框架为例，构建了一个可直接使用的模板，用于专业水准的电子舞曲生成；通过客观和主观标准，达到了专家商业制作的水平。(3) 我们提供了一个简单的（基于规则的）节奏和旋律生成器，只需给定调性和和弦进行，即可生成海量符号音乐乐谱，从而验证了我们关于大规模数据合成的假设。

2 动机

在音乐信息检索（MIR）领域，为电子舞曲（EDM）这类技术复杂的音乐类型获取带标签数据一直难以实现。首先，几乎无法对电子舞曲歌曲开展数据标注。传统音乐通常使用钢琴和一些管弦乐器，摇滚音乐则使用吉他、贝斯和鼓；相比之下，专业电子舞曲歌曲使用的乐器和合成器数量要多出几个数量级。此外，电子舞曲曲目还使用许多自动化效果，例如滤波器自动化和混响自动化[5]等，这使得其混音器通道在技术上远比其他音乐类型复杂。因此，标注电子舞曲歌曲要困难得多。除此之外，从用于制作电子舞曲歌曲的项目中获取数据也并不可行。电子舞曲制作人社区使用 [1–3], 等数字音频工作站（DAW），每条音轨都由一个数字音频工作站项目描述，而不是由原始乐谱描述。然而，几乎没有任何数字音频工作站具备足够成熟的功能，使用户能够便捷地通过脚本将音符绘制到钢琴卷帘中。一个例外是 Reaper，它内置了名为 ReaScript 的脚本功能 [9]。不过，考虑到 Reaper 在电子舞曲社区中的相对不普及，研究人员和音乐人可能很难合作使用 Reaper 制作模板。因此，即使有先进深度学习算法生成的大量符号音乐，我们也几乎无法仅凭某个数字音频工作站项目模板批量制作电子舞曲曲目，更不用说构建高质量模板本身的难度了。总之，带有详细音乐学标注（如音符位置、时值等）的电子舞曲歌曲数据集或数据生成器仍然是一片空白。

[5]Brooks 大师课：https://www.youtube.com/watch?v=JTJwR125DGQ

面对这些困难，为什么还要费力检索电子舞曲的详细信息？我们提出了几个应用场景，以展示音乐信息检索对于电子舞曲歌曲的重要性。首先，通过检索更加细粒度的信息（音色、节奏等），音乐流媒体服务可以构建更准确的推荐系统，为用户提供更加个性化的推荐。其次，这可以提升音乐可视化系统的表现。先前的工作（如 [17] ）已经证明，在仅获得相对粗粒度的检索信息（子类型）的情况下，利用监督学习方法实现可视化自动化具有潜力。只有当系统能够检索更加细粒度的信息时，其全部能力才能得到释放。第三，电子舞曲歌曲的自动音乐转录（AMT）尚未得到研究；当前最先进的转录算法 [10]是在 Slakh2100 和 [18] 以及 MusicNet [26] 等数据集上训练的，而这些数据集只包含古典音乐，或由钢琴、吉他、贝斯和鼓演奏的其他音乐类型。最后但同样重要的是，音乐源分离（MSS）在电子舞曲歌曲上的表现尚不成熟，但它在电子舞曲节日中发挥着重要作用，使艺术家能够在演出中混合不同曲目的不同部分。当前的音乐源分离算法（如 [23]）使用鼓-贝斯-人声-其他的方案，在 Slakh2100 和 MUSDB18 [20], 等数据集上进行训练。这些算法在电子舞曲场景中能发挥一定作用，但无法在乐器之间分离出主旋律和和弦等细致声源。基于这些应用前景，我们提出了“LooPy”。其内部结构类似于 FL Studio（社区中最受欢迎的数字音频工作站之一），并搭建起从符号音乐到专业制作电子舞曲之间的桥梁，作为面向音乐信息检索研究的电子舞曲数据基础设施。

3 相关工作

尽管相关研究还相对有限，已有少数研究探讨了电子舞曲的音乐信息检索。Hsu 等人 [13] 提出了用于电子舞曲子流派分类的新型神经网络架构。Zehren 等人 [28]提出了一种“提示点”检测方法，以自动构建音轨之间的过渡，模拟专业 DJ 在舞台上的操作。尽管这些研究在各自的子任务上取得了成功，但所涉及的信息检索仍相对粗略。音乐转录和音乐源分离等更具挑战性的任务则几乎尚未得到研究。

现在，我们讨论几项与我们的工作相似的研究。还有一些用于音乐生成的 Python 代码库或软件包，它们输出音频波形，而非符号乐谱或 MIDI 文件。早期开发者构建了一个名为“Musicpy” [21] 的音乐编程框架，具有简洁易用的语法和良好的可读性。此外，较新的项目“DawDreamer” [7] 为基于 Python 的 DAW 开辟了道路，支持虚拟演播室技术（VST）乐器和音效等核心 DAW 功能。Spotify 开发的另一个 Python 软件包“pedalboard” [6], 提供快速 I/O 工具，并支持与 TensorFlow [5] 数据管道兼容的音效添加功能。尽管这些工作并未涵盖我们的构想，但确实展现了使用 Python 脚本处理音乐音频的良好前景。我们的代码库也将它们的若干音效封装为高级方法，用于音频混音。

我们的作品“LooPy”能够根据指令和符号音乐表示生成音频。换句话说，它在某种程度上也可以被视为一款数字音频工作站（DAW）。不过，我们的目标是

为电子舞曲提供高品质的模板框架。为简化流程，“LooPy”中的生成器（乐器）直接使用在 FL Studio 等 DAW 中预先渲染的音色 20 [1]（通过 Sylenth 等 VST 插件 1 [15], ），而不是从头合成波形。我们的首要目标是通过 Python API 提供可靠的电子舞曲音轨模板，并使其兼容符号音乐乐谱。如果一切顺利，我们计划加入更多通用 DAW 功能，例如加载 VST 插件，为用户提供更多选择。

4 特性

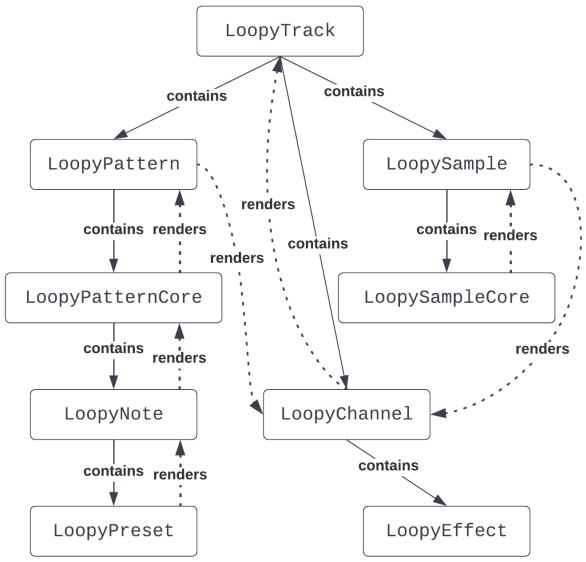


图 2：该软件包的核心类及其关系。

“LooPy”以 Python 软件包的形式封装，包含多个模块，可用于快速、便捷且专业地构建电子舞曲音轨模板。以下各小节将介绍该软件包支持的功能。

4.1 核心类

在“LooPy”中，音轨数据以字典形式分层存储，该软件包由若干核心类构建而成。这些类在图 2 中以矩形表示。接下来的几段将说明这些类如何协同工作，共同构成一个完整的电子舞曲项目文件。在附录 A 中，我们将详细展示本研究成果与 FL Studio [1]，之间的关联；如果用户熟悉 FL Studio 或其他面向电子舞曲制作的数字音频工作站（DAW），这将有助于他们更好地理解我们的配置。

每条音轨（歌曲）都由一个 Loopy 表示。由于电子舞曲歌曲包含由特定音乐结构（旋律、和弦进行等）构成的循环，因此每种重复出现的音乐段落类型都对应一个 LoopyPatternCo 落的每次出现，音轨都会存储一个 LoopyPatter ，一个模式核心定义一种模式，而一个模式则定义一个实际段落及其在音轨中的位置（小节索引）。类似地，音轨会存储所有采样的出现记录，每条记录都包含其采样核心，该核心定义了存储音频采样的路径。此外，音轨还会存储渲染过程中使用的所有混音器通道。

模式核心中存储着所有音符。每个音符都保存着生成该音符的生成器指针。在“LooPy”中，目前所有生成器都定义为 LoopyPreset 对象，而不是从头生成声 器。我们为每种预设预先渲染了钢琴键盘上 88 个音高对应的声音。因此，定义音符时，我们会直接从预渲染文件中加载音频。当然，我们也会考虑集成一些波形合成器，以提供更大的声音设计自由度。

渲染音轨的音频时，这些类的工作方式类似于反向传播。每个音符的音频由预设渲染。模式核心的音频由渲染后的音符组合而成。模式/采样的音频基本上就是模式/采样核心的音频。随后，模式和采样的声音会进入混音器通道，每个通道都定义为一个 LoopyChannel 实例。每个通道都包含多种效果，例 混响等。这些效果基于 [6] 封装为 LoopyEffect 类。最后，各通道的输出组合成音轨的音频。

4.2 风格

每位艺术家，有时甚至每首发行的音轨，在作曲和混音方面都会展现出一些独特风格。我们构建了 Loopy 为电子舞曲风格或子流派的基类。它包含两个字典：一个声音表和一个通道表，其中包括作曲和混音所需的所有信息。随后，通过修改这两个类变量——声音表和通道 LoopyStyleBase 扩展出不同的风格。

声音表和通道表的示例见清单 1 和 2。为便于保存和加载（为 JSON 文件），我们将这两个表都格式化为 Python 字典。对于声音表中的每个键值对，键表示音轨的一个组成部分（例如，贝斯、底鼓），值则是一个字典列表，每个字典都包含源文件路径以及加载声音的参数。请注意，对于填充音效和下滑音等效果采样（fx），我们会在目录中放入十几个采样，并随机挑选几个来编排每条音轨，以增加多样性。另一方面，通道表只是将音轨的各个组成部分映射到一个字典列表，每个字典都存储 LoopyEffect 对象的名称和参数。

列表 1：声音表

```
"低音": [{" 来源_路径 ": "Ultrasonic-BS-Home.wav", "
增益 ": -19.0}, .....],"底鼓": [{" 来源_路径 ": "KSHMR_
Top_Kick_03.wav", "增益": -21.1 ,
```

```
    " 空白_每个 ": 8},

], ……
“fx”: [
  {" type ": " 主要 - 填充 ",
   “ 方向 ”: “ .. / 采样 / 主 - 填充 ”,
   “ 高通 ”: 250,
   “ 增益 ”: -12,
   " 每个 ": 8},

], ……
```

……

列表 2：通道表

```
" 主音 " [
  {" type " " 高通 " ,
   " 频率 ": 300},
  {" 类型 ": " 侧链 ",
   " 衰减 ": 0.5,
   " 插值_顺序 ": 3,
   " 幅度 ": 0.66},
  {" 类型 ": " 混响 ",
   " 干声_电平 ": 0.5,
   " 湿声_电平 ": 0.8},
  {" 类型 ": " 平衡 ",
   " 增益 ": 1.5},
  {" 类型 ": " 限幅器 ",
   " 阈值 ": -6.0}
],
```

……

4.3 生成

为了生成音轨的音频，“LooPy”会根据声音表构建预设并将采样加载到音轨中，然后根据通道表构建混音器通道。输入乐谱（音符信息）后，会初始化一批 LoopyNote 实例，并将其放入 LoopyPa[illegible] 对象中。每个 LoopyNote 对象都包含一个 Loopy[illegible] 对象的地址，该对象负责“演奏”音符，以及音符的其他信息，例如位置和音符的值等。所有模式和采样都正确放置后，音轨即可进行渲染。

混音任务已经解决，那么我们该如何衍生出大量的符号音乐表示？符号生成是一个经过深入研究的领域，目前仍在进行密集研究。由于时间有限，我们实现了一个确定性的节奏旋律生成器，可根据给定的和弦进行生成旋律，并配合前面小节中提到的主要模块。

在旋律写作实践中，音符的音高和时值不应彼此割裂。然而，由于我们的工作主要旨在为判别任务提供基础设施，我们将这两个特征分开，以简化符号生成过程，代价是作曲质量有所下降。我们首先按照算法 1 中的步骤生成节奏，得到一个由音符“占位符”组成的列表。

每个占位符表示一个音符的位置和时值。然后，我们在指定的大调/小调音阶中随机选择音高并填入占位符，从而得到一个音符列表。列表中的每个元素都是（k， v， st），其中 k 表示音高， v 表示时值（数值）， st 表示位置。

部分生成的节奏如图 3 所示。每个蓝色片段表示一个待填入音符的占位符，每条红色竖直虚线表示一拍。我们为 4 小节生成节奏，因为大多数旋律会每隔 n 小节重复一次，其中 n 不大于 4。此外，我们提供参数 rep_bar，即旋律节奏重复所需的最少小节数，供用户模仿参考曲目的风格。

算法 1 一种基于规则的算法，用于为 N 小节生成拍号为 4/4 的节奏

输入： vs， r， λ ▷ 音符时值列表，例如 $[\frac{1}{16}, \frac{1}{8}, \frac{3}{16}, \frac{1}{4}]$,- 分辨率（最短音符的时值），例如 116，……的参数泊松分布 Pois(λ)

初始化 st， $ed \leftarrow -1$， 0 ▷ 占位符的起始和结束位置

初始化 $ps \leftarrow []$

当为真时 **执行**

采样 $t \sim$ Pois(λ)

更新 $st \leftarrow ed + t \cdot r$

示例 $v \sim$ 均匀分布(vs) ▷ 选择一个音符时值（均匀地作为默认）

更新 $ed \leftarrow st + v$

如果 $ed > N$ **则**

中断

结束 if

更新 $ps \leftarrow ps + [(v, st, ed)]$ ▷ 附加此占位符

结束循环

输出： ps ▷ 占位符列表

算法2 一种为 N 小节生成音符的基于规则的算法拍号为 4/4，来自一些节奏占位符

输入： ps， ks， ws ▷ 占位符列表，音高列表，例如例如 [C5、D5、E5、G5、A5、B5、C6], 概率权重（默认为均匀）

初始化 $ns \leftarrow []$

用于 v， st， $ed \in ps$ **执行**

样本 $k \leftarrow np.random.choice(ks, p = ws)$

更新 $ns \leftarrow ns + [(k, v, st)]$

结束于

输出： ns ▷ 音符列表

5 实验

我们的实验分为两部分。第一部分侧重于重制多首流行电子舞曲，并比较我们的重制作品与原曲。第二部分展示了我们针对特定电子舞曲风格进行大规模并行生成实验的评估结果。

表 1：响度比较

类别 / 标准	风格 1（本文方法）	风格 1（原始版本）	风格 2 我们的	风格 2 原曲	通用参考
综合 *LUFS*（*dB*）	**-5.4**	**-5.3**	**-4.9**	**-5.6**	**-8.0**
短时 *LUFS*（*dB*）	−5.3 ± 0.2	−5.3 ± 0.1	−5.0 ± 0.2	−5.6 ± 0.2	\

a 风格 1 指旋律渐进浩室，以 Tobu[3] 为代表。风格 2 指情感渐进浩室，以另外两位世界知名制作人[4] 为代表。b “我们的”指我们生成的数据，“原曲”指我们打算重制的原曲。所有读数均通过 8小节高潮段（电子舞曲曲目的高潮部分）测量。

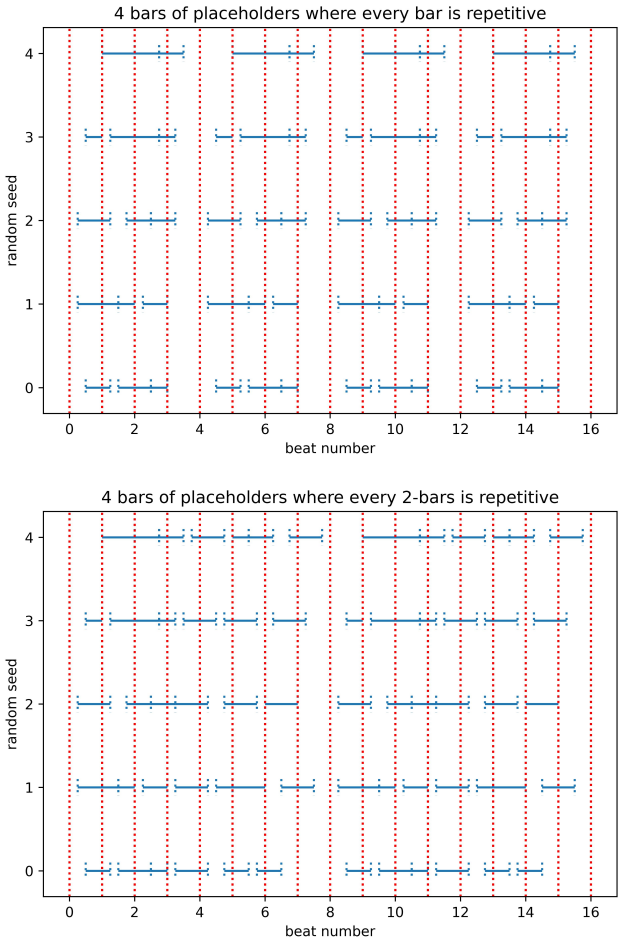


图 3：节奏生成结果示例

5.1 重制

在本小节中，我们重制了两首热门电子舞曲曲目[3][4]的高潮段落。对电子舞曲音轨质量的评估缺乏一种……

[3]Tobu - Life: https://soundcloud.com/7obu/tobu-life [4]Starlight (Keep Me Afloat): https://soundcloud.com/martingarrix/martin-garrix-dubvision-and-shaun-farrugia-starlight-keep-me-afloat

统一的量化标准。在制作方面，一个常见的衡量标准是响度。目前，主流电子舞曲的综合响度通常不低于 −8.0 dB [4]。此外，如今许多主流音乐在最响的部分都接近 −5.0LUFS，有时甚至更高，其中包括我们的两首参考曲目[3][4]。实验表明，与参考曲目相比，我们生成的结果在没有明显失真的情况下，也能达到极具竞争力的响度，如表 1 所示。此外，从图 5 可以看出，从梅尔频谱图来看，参考音轨的旋律和低音线略为突出。不过，从听感上来说，我们生成的音轨同样具有明亮的旋律和饱满的低音线。事实上，从梅尔频谱图来看，我们的音轨混音更为饱满，这也与我们的音量稍大的结果相符。

诚然，制作质量还有其他标准，例如立体声宽度等。然而，我们的工作定位是用于信息检索的合成数据生成器，而非数字音频工作站。混音是否响亮且干净，是电子舞曲音乐制作质量的重要指标。加入其他客观标准将作为未来的首要工作。

5.2 批量制作

我们还进行了一系列大规模生成实验[6]，使用我们为渐进浩室设计的主要模板。随后，我们针对生成结果示例和一些商业发行作品开展了一项双盲调查，评估以下 3 项标准：创造力、自然度和音乐性。参与评估的六段作品见此处，其中第 1、3、5 段为生成示例，第 2、4、6 段为商业发行作品的片段（参与者并不知道这一顺序）。为避免使用音乐识别软件识别这些片段，我们对商业发行作品片段随机进行了升调或降调处理，且未引入失真。

评估结果见图 4，结果表明，商业发行作品在音乐性和创造力方面仍显著优于我们使用简单算法生成的乐曲。不过，混音模板使我们的音轨在自然度上接近商业发行作品。鉴于“LooPy”定位为面向判别式音乐信息检索任务的基础设施，自然度是这里最重要的评判标准，因为它与音轨的制作质量密切相关，而创造力和音乐性主要与作曲有关。使用 [12, 27]等最先进的符号音乐生成算法，可以弥补我们在这两项指标上的不足。

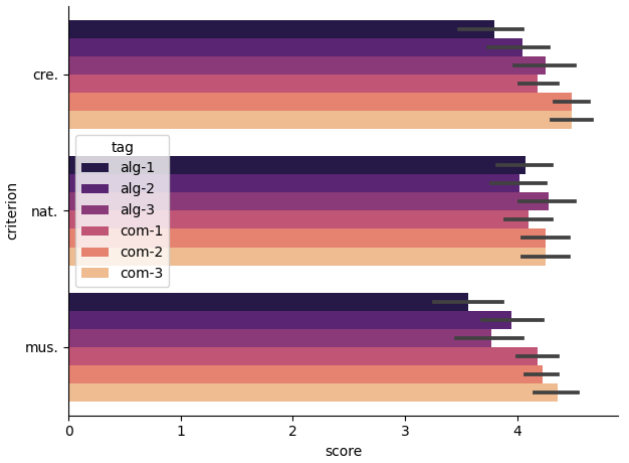


图 4：主观评价结果

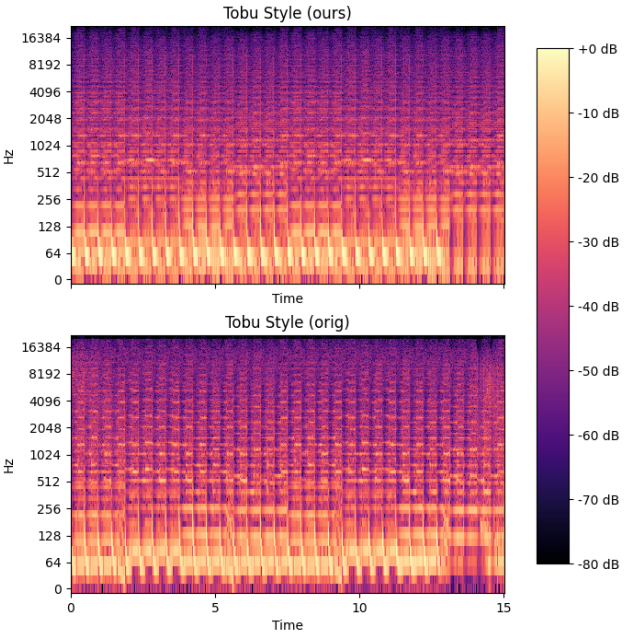


图 5：我们生成的音轨与原曲的梅尔频谱图

6 结论

总而言之，我们提出了一个便于研究的数据框架，以及一个专注于电子舞曲（EDM）音乐信息检索（MIR）的音频生成器。用户只需不到 10 行代码，即可根据指定的符号化旋律线和和弦进行生成制作精良的音乐音频，或指定一个调性来生成大量音轨。实验表明，与主流电子舞曲曲目相比，生成器能够生成达到专业水准的音乐。以下段落将介绍未来需要努力的几个方面。

首先，我们评估中采用的标准并未涵盖所有方面。尽管目前的生成结果听起来相当不错，但我们无法保证它们现阶段符合所有行业标准。为了让生成数据更适用于下游音乐信息检索任务，我们还应考虑立体声宽度等其他量化标准，以确保使用这些数据训练的模型能够良好地泛化到现实行业中的商业音乐。

其次，我们的模板应涵盖更多音乐流派。毫无疑问，未来贝斯、电音浩室、回响贝斯等其他流派也占据了市场的很大份额。即便是渐进浩室，我们的模板也只涵盖了其中一种流行风格。为这些流派和风格构建模板，是让该套件在社区中得到广泛应用的关键一步。诚然，要我们独自为每一种流行的电子舞曲流派构建模板并不现实。这也是我们竭尽全力将该套件打造得高水准、专业化的原因，以便专业艺术家和音乐信息检索研究人员能够轻松协作。我们计划邀请其他专业人士参与构建新模板，将其作为下一步工作之一。

最后但同样重要的是，“LooPy”的一些功能还需要改进和扩展。实验过程中，我们发现有若干功能使用起来不够便捷。例如，受当前代码库的限制，将一个通道的输出路由到另一个通道的输入并不方便。这样的路由功能可以大幅简化混音器通道，因为用户只需创建一个侧链通道供多个乐器通道使用，而不必在每个通道上都添加侧链效果。此外，还需要构建高级功能，例如针对音量、滤波器和混响等的自动化控制，以提升音轨的专业性并增加动态变化。我们还计划在未来支持第三方 VST 插件与效果模块兼容。这些高级功能将使“LooPy”成为音乐信息检索领域更标准、更具通用性的基础设施。

致谢

我们感谢 Sylenth1 音色库的匿名制作者。我们使用这些音色库渲染预设；若没有它们，我们就无法轻松构建生成器的音色库。我们还感谢“pedalboard”的作者将其代码开源，这让我们能够更轻松地构建效果模块。

参考文献

[1] 2020. *FL Studio* 在线手册. https://www.image-line.com/fl-studio-learning/fl-studio-online-manual/ 计算机软件。[2] 2022. *Ableton* 参考手册第*11*版. https://www.ableton.com/en/manual/ credits/ 计算机软件。[3] 2022. *Logic Pro* 用户指南. https://support.apple.com/guide/logicpro/welcome/ Mac 计算机软件。[4] 2022. 舞曲母带处理。https://www.sageaudio.com/blog/mastering/mastering-for-dance-music.php[5] Martín Abadi、Ashish Agarwal、Paul Barham、Eugene Brevdo、Zhifeng Chen、Craig Citro、Greg S. Corrado、Andy Davis、Jeffrey Dean、Matthieu Devin、Sanjay Ghemawat、Ian Goodfellow、Andrew Harp、Geoffrey Irving、Michael Isard、Yangqing Jia、Rafal Jozefowicz、Lukasz Kaiser、Manjunath Kudlur、Josh Levenberg、Dandelion Mané、Rajat Monga、Sherry Moore、Derek Murray、Chris Olah、Mike Schuster、Jonathon Shlens、Benoit Steiner、Ilya Sutskever、Kunal Talwar、Paul Tucker、Vincent Vanhoucke、Vijay Vasudevan、Fernanda Viégas、Oriol Vinyals、Pete Warden、Martin Wattenberg、Martin Wicke、Yuan Yu 和 Xiaoqiang Zheng。

[6]所有渲染文件和数据均位于此文件夹中：https://drive.google.com/drive/folders/1X- jArl _6DsBxZdXGL7wzgaVI4m6f8wiy?usp=sharing

2015年。《TensorFlow：异构系统上的大规模机器学习》。https://www.tensorflow.org/ 软件可从 tensorflow.org 获取。[6] Alexandrawei。2022年。介绍 pedalboard：Spotify 面向 Python 的音频效果库。https://engineering.atspotify.com/2021/09/introducing-pedalboard-spotifys-audio-effects-library-for-python/ [7] David Braun。2021年。DawDreamer：弥合数字音频工作站与 Python 接口之间的鸿沟。https://archives.ismir.net/ismir2021/latebreaking/ 000001.pdf [8] Hao-Wen Dong、Wen-Yi Hsiao、Li-Chia Yang 和 Yi-Hsuan Yang。2017年。MuseGAN：用于符号音乐生成和伴奏的多轨序列生成对抗网络。arXiv: 1709 06298 [ ] . eess.AS Justin [9] 2023 Frankel。Reaper：ReaScript。https://www.cockos.com/reaper/sdk/reascript/reascript.php [10] Joshua P Gardner、Ian Simon、Ethan Manilow、Curtis Hawthorne 和 Jesse Engel。2022年。MT3：多任务多轨音乐转录。载于《国际学习表征会议》。https://openreview.net/forum?id=iMSjopcOn0p [11] Z. Guo、J. Kang 和 D. Herremans。2022年。受领域知识启发的音乐嵌入空间及用于符号音乐建模的新型注意力机制。arXiv: 2212 00973 [ ] . cs.SD Sangjun Han、Hyeongrae Ihm、[12] Moontae Lee 和 Woohyung Lim。2022年。利用神经离散表示生成符号音乐循环。arXiv: 2208 05605 [ ] . cs.SD Wei-Han Hsu、Bo-Yu Chen 和 Yi-Hsuan Yang。2021 [13] 年。基于梅尔频谱图和节拍图特征的电子舞曲子流派深度学习分类。arXiv: 2110 08862 [ ] . eess.AS Natasha Jaques、Shixiang Gu、Richard E. [14] Turner 和 Douglas Eck。2016年。通过使用强化学习微调循环神经网络来生成音乐。载于《深度强化学习研讨会》，NIPS。[15] LennarDigital。2023年。Sylenth1 用户手册。https://www.lennardigital.com/ sylenth1/manual.php [16] Xinyu Li。2021年。用于连续和弦与旋律生成的轻量级深度序列模型。https://studylib.net/doc/25597577/a-lightweight-deep-sequential-model-for-continuous-chord-... [17] Xinyu Li。2022年。HouseX：细粒度浩室音乐数据集及其在音乐产业中的潜力。载于《2022年亚太信号与信息处理协会年度峰会暨会议（APSIPA ASC）》。335–341。https://doi. org/ 10 23919 55919 2022 9980316 . /APSIPAASC . . Ethan Manilow、Gordon [18] Wichern、Prem Seetharaman 和 Jonathan Le Roux。2019年。削减音乐源分离领域中的 Slakh：用于研究训练数据质量与数量影响的数据集。载于《IEEE 音频与声学信号处理应用研讨会论文集（WASPAA）》。IEEE。[19] Gautam Mittal、Jesse Engel、Curtis Hawthorne 和 Ian Simon。2021年。利用扩散模型生成符号音乐。https://doi.org/10.48550/ARXIV.2103. 16091[20] Zafar Rafii、Antoine Liutkus、Fabian-Robert Stöter、Stylianos Ioannis Mimilakis 和 Rachel Bittner。2017年。用于音乐分离的 MUSDB18 语料库。https: //doi.org/ 10 5281 1117372 . / [21] 2021 zenodo. Rainbow-Dreamer。Rainbow-dreamer/musicpy：musicpy 是一种 Python 音乐编程语言，旨在通过音乐理论和算法，以非常便捷的语法创作音乐。https://github.com/Rainbow- Dreamer/musicpy [22] Adam Roberts、Jesse Engel、Colin Raffel、Curtis Hawthorne 和 Douglas Eck。2018年。用于学习音乐长期结构的分层潜向量模型。载于《国际机器学习会议》。[23] Simon Rouard、Francisco Massa 和 Alexandre Défossez。2022年。用于音乐源分离的混合变换器。arXiv: 2211 08553 [ ] . eess.AS Serkan Sulun、[24] Matthew E. P. Davies 和 Paula Viana。基于连续值情绪的符号音乐生成。《IEEE Access》10（2022），44617 44626 10 1109 2022 3169744 –。[25] 2022 https://doi.org/ . /access. . Masahiro Suzuki。Score Transformer：从音符级表示生成乐谱。载于 ACM 亚洲多媒体会议（澳大利亚黄金海岸）（MMAsia '21 31, 7 10 1145 3469877）。美国计算机协会，纽约州纽约市，美国，文章页码。https://doi.org/ . / . 3490612 [26] John Thickstun、Zaid Harchaoui 和 Sham Kakade。2017年。从头学习音乐特征。arXiv:1611.09827 [stat.ML][27] Botao Yu、Peiling Lu、Rui Wang、Wei Hu、Xu Tan、Wei Ye、Shikun Zhang、Tao Qin 和 Tie-Yan Liu。2022年。Museformer：用于音乐生成的细粒度与粗粒度注意力变换器。载于《神经信息处理系统进展》，Alice H. Oh、Alekh Agarwal、Danielle Belgrave 和 Kyunghyun Cho（编）。https://openreview.net/forum?id=GFiqdZOm-Ei [28] Mickaël Zehren、Marco Alunno 和 Paolo Bientinesi。2020年。DJ 混音提示点的自动检测。arXiv:2007.08411 [cs.SD]

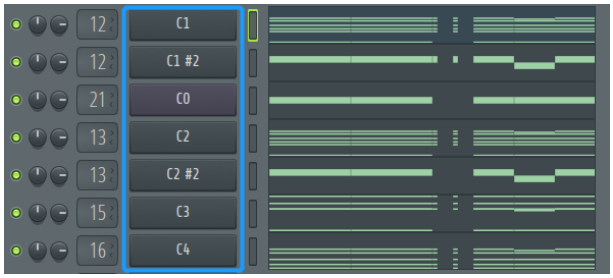


图6：生成器列表

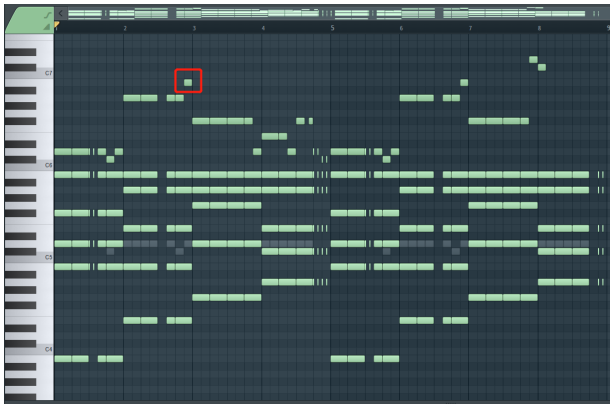


图7：模式的钢琴卷帘



图8：音轨中使用的采样列表

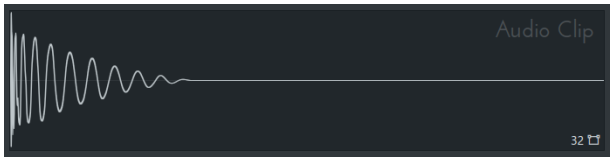


图9：采样的波形

A 设置

我们的工作深受 FL Studio 架构的启发，因此“LooPy”的设置与 FL Studio 类似。这附录部分展示了我们如何在软件包中实现 FL Studio 的主要功能。

如图 6 所示，乐器在 FL Studio 中被放置为生成器。为了简化操作，我们在“LooPy”中渲染了一些常见的电子舞曲

音色，这些音色来自 Sylenth 1 [15]，并被用作我们的生成器，称为 I

图 7 清晰地展示了音符和模式核心。在这个钢琴卷帘中，每个小矩形都是一个音符，并连接到一个生成器。例如，红色矩形框住的那个是一个八分音符。我们的模式核心可以看作是包含一串音符的钢琴卷帘。

图 8 中的采样列表对应于“LooPy”中的采样核心列表。每个采样核心都只是一个录制并处理后的波形，如图 9 所示。许多鼓组元素（底鼓、军鼓、踩镲等）和特殊音效（短促音、扫频、氛围音等）通常都会制作成采样，可以轻松拖放到音轨中。

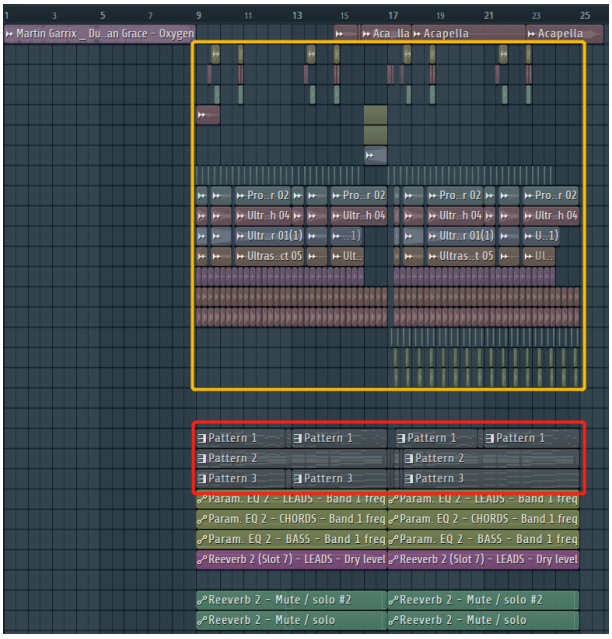


图 10：音轨的播放列表

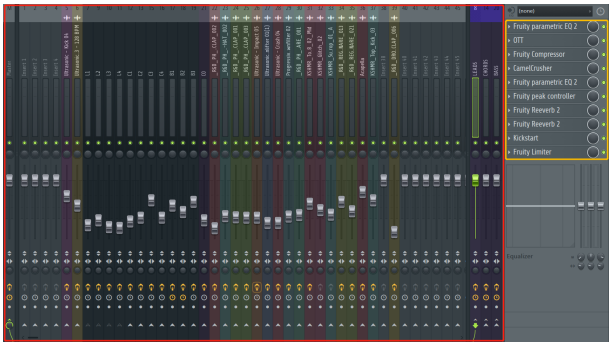


图 11：音轨的混音器通道和效果

在 FL Studio 中，模式（红框圈出）和采样（黄框圈出）在音轨中的放置方式如图 10 所示。对于 “LooPy”，当 Loo e 实例被指定在音轨中的位置（例如“第 8 小节的开始处”）时，我们会创建一个 Loop e 实例。

对于渲染，每个 Lo 对应图 11 中红框圈出的一个混音器通道，每个效果则对应黄框圈出的一个效果插件。请注意，“LooPy”和 FL Studio 的一个主要区别在于，我们将模式的音频路由到通道中，而后者将生成器的音频路由到通道中。目前，我们没有证据表明我们或 FL 的路由方式哪种更好，但未来我们会根据用户反馈调整设置。

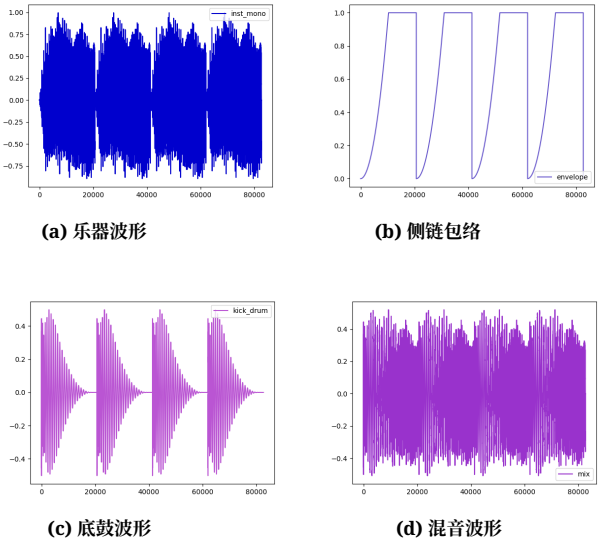
B 侧链压缩

值得注意的是，侧链压缩是电子舞曲混音中不可或缺的一部分。应用侧链包络是为了避免底鼓与其他乐器发生冲突，从而保持底鼓的冲击力和整体混音的清晰度。图 12 展示了将和弦合成器与底鼓混音时侧链效果的示例。为便于查看，这里只绘制单声道波形[6]。下一段将给出侧链压缩的严格定义。

给定 1 小节、采样率为 44100 Hz、每分钟节拍数为 128、拍号为 4/4，我们知道该小节有 4 拍。每拍的采样总数为

$$N_{beat} = int(\frac{60 \times 44100}{128}) \approx 20671$$

[6]此外，四个子图的 y轴并不完全一致，因为我们在混合波形之前调整了音量平衡。



(a) 乐器波形 **(b) 侧链包络**

(c) 底鼓波形 **(d) 混音波形**

图 12：侧链示意图。

，而每小节的采样总数为

$$N_{bar} = int(\frac{4 \times 60 \times 44100}{128}) \approx 82687$$

。首先，我们介绍朴素侧链公式。从数学上说，设包络为 e，乐器波形为 y_{inst}，底鼓波形为 y_{kick}。我们可以将 e 视为一个从 $\mathbb{N}$ 到 $[0,1]$, 的周期函数，在每个长度为 N_{beat}的周期内单调不减。在每个周期中，它达到 1 的位置以及达到 1 之前的增长顺序，都是用户可以根据期望风格进行调整的参数，因为侧链包络的形状会直接影响混音的律动。作为数组， e的形状为 $(N_{bar},\ )$。波形 y_{inst}和 y_{kick}的形状均为 $(N_{bar},\ 2)$，因为它们是立体声的。我们可以将混音波形 y_{mix} 写作

$$y_{mix} = y_{kick} + e \cdot y_{inst}$$

，其中 · 表示逐元素乘积[7]。

具体来说，我们并不希望所有声音都进行 100% 的侧链处理，因为那样会让混音的律动感过于摇摆。我们对所有乐器都应用侧链处理，并为每种乐器设置不同的混音电平，以确保呈现专业的律动。这里的混音电平定义为 $\alpha \in (0,1)$，其中

$$y_{mix} = y_{kick} + \alpha \cdot e \cdot y_{inst} + (1-\alpha) y_{inst}$$

。据专家建议， $alpha \approx 0.8$ [8]。

[7]在实现中，通过广播机制将二维（立体声）波形数组与包络相乘。[8] https://www.youtube.com/watch?v=awNv4-0uU5o