ARRANGE：用于叠加人声的智能体技能

Henry Tan[1]，Eidee Tan[2]

[1]哈佛大学，美国[2]*enrica.ai*

摘要

为现有的钢琴录音添加歌唱人声，需要对旋律、音节安排、发音、时序和人声音域作出决策。ARRANGE 将这项工作组织为一系列智能体技能，并提供易读的中间产物和作曲家审批检查点。大语言模型将简明英语指令、键盘演奏的音符以及人声参考转化为局部编辑；确定性脚本会在开放音频模型生成结果之前验证并应用这些编辑。乐谱画布让音符、音节和发音方面的决策可供检查和编辑。在一项基准测试中，针对 16 个修订请求，每个请求抽样三次，并使用五种大语言模型；类型化编辑的精确修改正确性达到 0.83–0.98，而不使用 ARRANGE 指令、通过重写整个计划的方式仅为 0.33–0.69。另一项乐谱编辑基准测试和一个第一人称实例探讨了中间表示与修正之间的相互作用。作者聆听评估还进一步考察了声音迁移和音域选择，其中包括对八种复制人声进行的 185 次聆听呈现。这些评估表明，受约束的编辑具有价值，同时也揭示了模型执行能力和聆听测量的局限。本文的贡献在于提出一种交互架构，通过可检查的中间表示保留音乐决策，而非开发新的歌声合成模型或证明其具有普遍可用性。

关键词

人机共创、Agent Skills、混合主动交互、用户能动性、歌声合成、乐谱记谱 ic

乐谱记谱

arXiv:2611.29283v1 [cs.SD] 22 Nov 2026

1. 引言

一位钢琴家录制了一段乐曲，并为其写下歌词。他们希望这些歌词能伴随着自己的演奏被唱出来。这样便形成了快速反馈循环。要实现这一效果，需要做出若干细致的决策：人声演唱哪些音符、哪些音节对应这些音符、每一行歌词相对于演奏时序从何处进入，以及每个音节如何发音。令人信服的音频效果并不意味着这些决策易于检查或修改。

作者最初使用 Suno 探索这项任务。在那次尝试中，生成的编曲取代了录制的钢琴声，而且很难通过提示词逐项修改音乐决策。这段经历促使作者提出 ARRANGE，但并不能证明当前商业产品普遍存在这一局限

工具。Suno 现在提供上传音频、多轨编辑、片段音高控制和时间拉伸设置的相关说明 [12]。开放系统也支持编辑和音频条件控制 [2, 3, 14, 15]。因此，我们的问题更加具体：基于大语言模型的界面如何在为录制演奏添加人声的同时，呈现并保留作曲家的决策？

ARRANGE 将这项任务视为通过智能体技能进行编译。作曲家提供简明英语、在录音上用键盘演奏的旋律，以及人声参考。每个阶段都会生成一个可供检查和修补的中间产物，然后再继续后续工作。当音节缺失或音符错误时，修正的目标就是代表该决策的制品。智能体负责转换请求；脚本约束其执行方式；作曲家则评判音乐效果。

编译器类比涉及明确的中间表示、验证和局部转换。自然语言请求仍然存在歧义，歌唱模型也仍具有随机性。ARRANGE 并不会让两者变得确定。它将能够明确表示的决策转移到共享乐谱和类型化编辑词汇中，同时将审批权留给作曲家。

本文的贡献如下：

预印本。已提交至 *HUMANE-LLM 2027*；目前正在审稿，尚未录用或发表。

henrytan@g.harvard.edu (H. Tan); eidee@enrica.ai(E. Tan)© 2026 本文作者版权所有。允许依据知识共享署名 4.0 国际 BY 4.0）使用。

1. 一个六阶段工作流程，为每项技能指定输入、输出和作曲器检查点，并将音频后端与交互契约分开。2. 乐谱画布，将乐谱记谱、音节、发音和回放联系起来，以支持局部修正。3. 一项综合评估，结合计划修订与乐谱编辑基准测试、一个第一人称实例，以及作者聆听评估，从而揭示该方法的优势与局限。

本研究面向的用户是能够读懂乐谱记谱并在键盘上演奏旋律的作曲家。我们评估的是这一使用场景，而不声称适用于具有各种音乐背景的人群。

2. 相关工作

可见的大语言模型工作流。 AI Chains 展示了可编辑的中间结果如何提升多步大语言模型交互的透明度和可控性 [13]。Agent Skills 将智能体可为特定任务加载的指令、脚本和资源打包在一起 [1]。ARRANGE 采用了这种打包方式，并针对音乐决策专门设计其中间表示。其类型化操作直接作用于制品状态，而非要求模型重新生成整个制品。原子概念编辑相关研究探讨了对模型输入进行受控修改 [5]；而 ARRANGE 研究的是对作曲家的计划或乐谱所做修改的范围。

音乐生成与编辑。 Suno 可根据文本生成歌曲，并支持基于音频的制作与编辑功能 [12]。YuE 面向长篇歌词转歌曲生成 [14]；ACE-Step 和 ACE- Step 1.5 支持生成与编辑操作 [2, 3]。YuE2 将符号生成与音频生成相结合[15], ，因此在合成前进行符号规划并非 ARRANGE 独有。我们着重关注交互契约：作曲家可读的制品、明确的校正操作，以及各阶段之间的审批点。

音频与乐谱后端。 高分辨率钢琴转录可从录制演奏中提取音符和时序 [6]。SoulX-Singer 支持乐谱条件歌声合成 [9], ，而 Seed-VC 支持语音转换，包括通过音高条件控制进行演唱 [7]。光学音乐识别则提供了从现有乐谱进入的另一种可能途径 [10]。这些都是互补组件；ARRANGE 并未贡献新的转录、分离、合成或转换模型。

人机共创。 AI Song Contest 的参赛团队描述了如何将创作工作划分为不同部分，并协商应对模型的局限 [4]。关于文本生成音乐的研究指出，将生成素材融入音乐家的工作流存在困难 [11]。对协作式歌曲创作的详细记录也使人类的贡献清晰可见 [8]。ARRANGE 将这种分解方式构建为智能体工作流，并以乐谱记谱作为共享表示，让作曲家能够质疑并纠正决策。

3. 技能作为编译阶段

ARRANGE 技能是一个包含指令、脚本和参考资料的文件夹。六个概念阶段构成了工作流程；其中一些阶段共用同一个实现文件夹。表 1 将每个阶段的制品与作曲家针对该制品所做的决策分别列出。

智能体会选择技能、转换语言，并调用脚本和音频模型。我们将它结构化的修订命令称为类型化编辑：每条命令都会选择一个预定义操作，并提供带有指定类型的具名参数。例如，“把弦乐推迟两个小节进入”会转化为类型化编辑 `shift_entry(layer="strings", by_bars=2)`。其中，`shift_entry` 是操作， `layer`标识目标层， `by_bars` 指定以小节为单位的偏移量。模型会输出此命令；应用脚本则检查操作和参数，并将其应用到现有计划中。因此，模型指定的是局部更改，而不是重写整个计划。对于不受支持或含糊不清的请求，预期系统会提出问题，而不是推断编曲决策。

表 1ARRANGE 中的阶段契约。检查点是用于规范工作流程的指令。不应将其解读为一种形式化验证的保证，认为不受约束的智能体无法绕过这些检查点。

阶段	输入	输出	作曲器检查点
聆听	钢琴录音和敲击节拍	转录音符、调性估计，and 相对于演奏的 beat map	小节编号是否符合我对乐曲的计数方式?
播放旋律	录音 and beat map	包含音高和在键盘上演奏的节奏	这是预期的旋律吗， or 是否需要重播某个乐句?
Fit	旋律 take and 歌词	带音节的音符，以及每个音符的发音	Is 每个 预期的 音节以正确的音符和正确的声音呈现，吗?
Plan	已批准的音乐材-料和语言要-求	层、进入与选段 plan	Does the 拟议 更改是否符合请求并保留先前决定?
Sing	已批准 乐谱和人声参考	人声演绎与录音的时钟对齐，包括进入小节的测量	它是否唱出了预期的旋律线合适的音域?
Mix	录音 and ap-经过验证的人声演绎	混音、分轨和数字音频工作站工程	平衡和声像位置是否可-接受?

这种分离限制了模型必须重新生成的状态量。验证会检查类型化编辑是否遵循操作语法和参数规则，但不会确认模型是否正确理解了音乐请求。例如，有效的 `shift_entry` 命令仍可能针对错误的层，或指定错误的偏移量。因此，修订基准既衡量预期更改，也衡量连带更改。它还会记录预期编辑未涉及的作曲家所有字段的修改。

乐谱画布和计划是彼此相关的中间表示，而不是可以互换的单一文件。歌词技能负责演唱旋律和音符级贴合；计划技能负责安排各层并记录编曲选择。它们的评估任务分别报告。聆听功能使用钢琴转录[6]；演唱功能使用 SoulX-Singer [9] 或 Seed-VC 等声音转换路线 [7]。替换后端需要使用遵循乐谱与时序接口的适配器。这并不意味着任意模型无需工程工作就能互换。

伴奏仍采用源录音，而非新生成的演奏。非八度的调性变化可能需要对该录音进行移调；因此，保留原演奏并不意味着最终波形始终不变。

4. 乐谱画布

乐谱画布以雕刻式乐谱的形式呈现匹配结果，并在音符下方标注歌词。小节编号遵循作曲家敲击的节拍，每个音节都有清晰可见的索引，发音视图则显示提供给演唱者的发音。点击音符即可回放，将符号表示与钢琴和人声联系起来。这些参考资料让作曲家和智能体能够指向同一个音乐事件。

4.1. 演奏并识别旋律线

作曲家一边聆听录音，一边用键盘弹奏预期的声乐旋律。这样可以同时捕捉音高和节奏，而不必将旋律线作为文本输入，也不必修正自动转录的演唱内容。作者在实际使用中发现，这种方式优于在时间线上拖动音节、修正参考转录或输入旋律。这是第一人称的设计观察，并非比较可用性研究的结果。

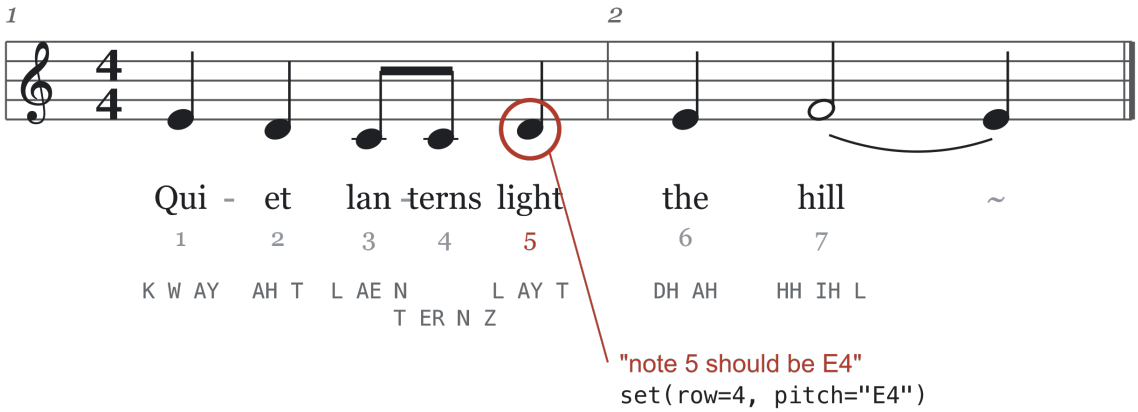

图1： ARRANGE 项目页面中的示意乐谱示例。音符共享音节标签、显示索引和逐音符声音；标记的音符是局部音高请求的目标。此插图并非经过评估的15小节节选的屏幕截图。

请求可以通过小节和拍、音节索引、单词、音高或乐谱图像来标识目标。相应的编辑词汇包括设置音域、移动或调整音符、插入或删除行、更改发音、提问和回答问题。较短的显示索引可以转换为内部行索引；显示内容与文件表示必须保持一致。

4.2. 定位并纠正错误

这个示例中的三个错误说明了中间表示为何重要。首先，将辅音拆分到单独的音符上，导致“morning”中出现了不想要的类元音音。其次，一行中的音节数超过可用音符数，导致有些词没有唱出来。第三，将“Footsteps”拆分到重复音符上，使塞辅音落在衔接处，听起来像是把这个词截断了。每个错误都在音符或发音层级上被识别，并通过局部调整予以纠正。

发音视图至关重要，因为仅更改文本未必会改变发送给合成器的声音。对于“Footsteps”，作曲家调整了跨音符的声音拆分方式，使一个塞辅音延续到第二个音符。匹配和发音文件保留了这一设置，以供后续渲染使用。

这些示例说明，应该在具体失败层面提高可见性，而不应仅总结智能体的意图。但这并不能证明所有可听见的缺陷都能通过乐谱记谱诊断。歌声合成模型仍可能无法良好呈现正确的乐谱，因此每次修改后都必须重新聆听相应的乐句。

5. 评估

我们提出四个问题：类型化操作能否保留预期的计划变更；以乐谱为导向的表征是否有助于执行修正；该工作流程在一位作曲家的实践中能发挥什么作用；以及作者聆听评估能揭示有关声音迁移和音域选择的哪些信息。模型输出基准测试并非用户研究。在整个音乐评估过程中，作者同时也是作曲家和聆听者。

5.1. 五个模型的计划修订

该基准测试包含 16 条固定的修订请求：七条针对作者的真实编曲计划，九条针对合成计划。每条请求都对应一项预期变更。每个模型在四种条件下分别回答三次，因此每个模型在每种条件下有 48 次抽样，五个模型的比较总计 960 次抽样。

这些条件分别是：由脚本应用的类型化 ARRANGE 操作；路径和值补丁；依据 ARRANGE 计划文档重写整个计划；以及仅依据一行指令、没有 ARRANGE 文档来重写整个计划。前三种条件都提供了计划结构信息。因此，这项比较评估的是一整套指令、表示方式和执行流程；它并未将词汇效果与所有验证行为区分开来。

正确性要求变更路径与预期完全一致；如果请求含糊或不受支持，则应在不进行编辑的情况下提出问题。另行统计无关变更路径、对受保护的作曲家所有字段所做的更改、无效输出以及遗漏的重新渲染依赖项。派生的时序和所有权记账不计入变更数。

表 2五模型计划基准测试中的精确修改正确性。每项比率基于 16 次请求 × 3 抽样。最后一列比较了使用类型化操作与不使用 ARRANGE 文档进行改写的结果。Wilcoxon 检验对每项请求的平均正确性进行配对比较；p 值仅供探索，未针对多重比较进行调整。

模型	类型化操作	Path 补丁	使用文档重写	重写不使用文档重写	差异与 p
			文档 tion	档	
Claude Sonnet 5.5	0.98	0.81	0.38	0.33	+0.65；p = 0.001
Claude Opus 5.5	0.90	0.73	0.71	0.69	+0.21；p = 0.031
Claude Haiku 5.5	0.83	0.75	0.73	0.62	+0.21；p = 0.047
Gemma 4 31B	0.90	0.67	0.62	0.46	+0.44；p = 0.009
Qwen 3.5 27B	0.83	0.69	0.71	0.50	+0.33；p = 0.078

对于每个模型，类型化操作的正确性最高。对于 Sonnet，采用文档和不采用文档进行重写时，每次抽样分别平均更改 19.9 个和 22.1 个无关字段；有文档的重写条件在 48 次抽样中产生了 11 个无效输出。在五种类型化条件中，记录到两次受保护字段更改，且均发生在 Qwen 上。Haiku 的类型化条件仍然更改了无关字段，表明类型化操作并不能消除语义错误。

较早的三条件 Sonnet 运行在采用类型化操作且没有附带更改的情况下达到了 0.96 的正确性；表 2 中较晚的四条件运行达到了 0.98。这是两次独立的运行，并不是对同一组输出的两次估计。

后续测试考察了前沿模型是否会使 ARRANGE 的任务结构变得多余。使用 ARRANGE 的类型化操作时，Claude Fable 5.1 的精确修改正确性达到 0.94；不使用 ARRANGE 指令、改为重写时则为 0.69（每种条件抽样 48 次；配对 Wilcoxon 检验 p = 0.062）。观察到的差距小于 Sonnet 的差距，但仅凭前沿能力，仍无法在这些请求上达到使用 ARRANGE 时的表现。

链式会话在一个真实夹具和一个合成夹具上测试了连续八个请求，每种夹具重复三次。使用类型化条件时，每个模型在六次会话中有三次最终计划与预期计划完全一致。排除一个所需时序未包含在计划中的一拍映射请求后，每个模型的六次会话都与预期状态一致。这项排除揭示了输入限制，不应被视为在完整会话任务上取得成功。

在单次完成基准测试中，Claude 模型通过 Claude Code 命令行界面调用，且未启用工具。开放模型则通过 vLLM 在本地运行，使用 FP8 检查点和服务器默认采样，完成长度上限为 32,000 个令牌。代码仓库列出了检查点和调用命令；耗尽的 Qwen 推理调用被判为无效。

5.2. 在乐谱上执行修正

一项重放研究使用了作曲家本人交互中记录的 15 条评分更正。其中大多数描述的是可听见的问题，而不是完整明确的编辑要求。借助歌词技能的结构化表示，三种 Claude 模型在 0.42–0.47 的抽样中提出了可接受的修正；若将有理有据的问题也计入，则这一比例为 0.53–0.69，而基于原始行的抽样比例为 0.18–0.31。对结构化抽取条件与

原始行条件进行配对比较，得到 $p = 0.064$。这提供了提示性证据，但尚不足以在统计学上确认效果有所改善。结构化抽取的 36 次抽样中有 26 次成功进行了发音修正，而原始行抽取的 36 次均未成功；结构化条件呈现了原始条件中缺失的发音信息。

公开发布版本包含其中九个修正请求。在该子集中，结构化修正的修正接受率为 0.56–0.59，原始行的修正接受率为 0.26–0.41。完整研究结果与已发布子集结果不应混为一谈。

另一个精选基准测试涵盖三首歌曲中的 62 个请求，其中包括九个有意不完整的请求。请求通过乐谱引用或屏幕截图标明音符，涵盖局部编辑、模式、乐段、症状、修订和问题。每个模型对每个请求抽样三次。成功的标准是音高和文本符合预期、时序处于指定容差范围内，或给出恰当的澄清或回答。

表3精选乐谱编辑的正确性。“类型化提示”在一个提示中提供结构化指令。“已安装技能”则允许 Claude Code 智能体加载技能并使用脚本；这属于单独的执行条件。每个模型-执行条件包含 186 次抽样。

模型	类型化提示	原始行与重写	已安装技能
Claude Haiku 5.5	0.80	0.60	0.84
Claude Opus 5.5	0.92	0.83	0.89
Claude Sonnet 5.5	0.91	0.74	0.82
Gemma 4 31B	0.55	0.49	—
Qwen 3.5 27B	0.68	0.58	—

类型化提示提高了每个模型的报告通过率。请求级符号检验显示，Haiku 的 $p < 0.001$，Sonnet 的 $p = 0.004$。已安装技能的结果表明，受限的单响应基准测试不能替代对智能体执行的评估：加载指令和使用工具可能引入额外故障。

在这 62 个请求中，Fable 使用类型化提示的得分也达到 0.90，而使用原始行的得分为 0.79（每种条件下抽样 186 次；请求级符号检验 $p = 0.115$）；其已安装技能得分为 0.88。

5.3. 作者录音中的一段示例诗句

该示例旨在测试作曲家能否在自己的伴奏上听辨声乐旋律线，并通过乐谱加以纠正。示例使用了作者钢琴录音中主题曲《紫色》的 15 小节；该主题曲由昆西·琼斯、罗德·坦珀顿和杰里米·卢博克创作，歌词由作者撰写。作者在键盘上演奏了预定的声乐旋律，并敲击了 88 拍，使其与演奏对齐。

聆听时发现了第 4 节所述的三处歌词和发音错误。作曲家在乐谱中找到了这些问题，进行了局部修正，并在重新渲染（耗时约一分钟）后再次聆听。在五次渲染结果中，乐句起始点与乐谱标记位置的偏差均在 0.1 秒以内。这些时序和延迟观察仅适用于一个主歌段落及其设置。下一项评估将考察所选人声和调能否使指定乐句听起来令人信服。

5.4. 将采样人声与歌曲匹配

采样人声是否实用，取决于它与歌曲及调的匹配程度。在作者的试验中，当旋律音高过高时，有些渲染结果听起来单薄或吃力，人声也变得不那么容易辨认。因此，聆听实验考察了哪种开放音频后端能生成实用的人声，以及选择合适的调如何改善同一采样人声的表现。

作者首先比较了 SoulX-Singer 和 Seed-VC，评判二者与采样人声的相似度以及演唱质量。在 35 次盲编码片段试听中，报告的平均评分分别为 SoulX-Singer 95.9/100、Seed-VC 88.5/100。这些是单位听者的判断，未在听者之间进行校准。作者还观察到，在测试配置下，人声音色的迁移效果比强力唱法、花腔和颤音等歌手特有技巧更令人信服。

选择合适的调可以让一些渲染结果更加饱满、更易辨认，也更令作者满意。为探究这一选择的实用起点，八个复制的人声演唱了同一段 15 秒片段 *All the ThingsYou Are* 的不同调版本。作者在一次试点实验、随机打乱的盲听、复核和排序比较中，共听取了 185 次呈现。最后，他通过并排比较选出的版本被用作参考调号，以比较各项建议。

测试的规则将旋律的最高音与参考人声已观测音高范围的上部相匹配，以其第90百分位数的演唱音高表示。该百分位数在试点实验中选定，并在盲测轮次开始前固定。在八种人声中，该规则给出的建议与作者选择的平均相差1.50个半音，而按中位音高匹配的平均差为6.00个半音（图 2）；个别误差达到四个半音。在用于匹配大语言模型比较的六种人声中，该规则的平均误差为1.33个半音，而各模型条件均值则高出4.0–7.4个半音。

聆听条件也很重要。对于分别聆听以及按音调序列聆听的 52 个相同的音频文件，后者设置下的平均紧张度评分约低七分。因此，数值评分需要一致的聆听条件才能具有可解释性。该规则提出一个用于试听的调；作曲家仍需判断人声与歌曲的配合效果。

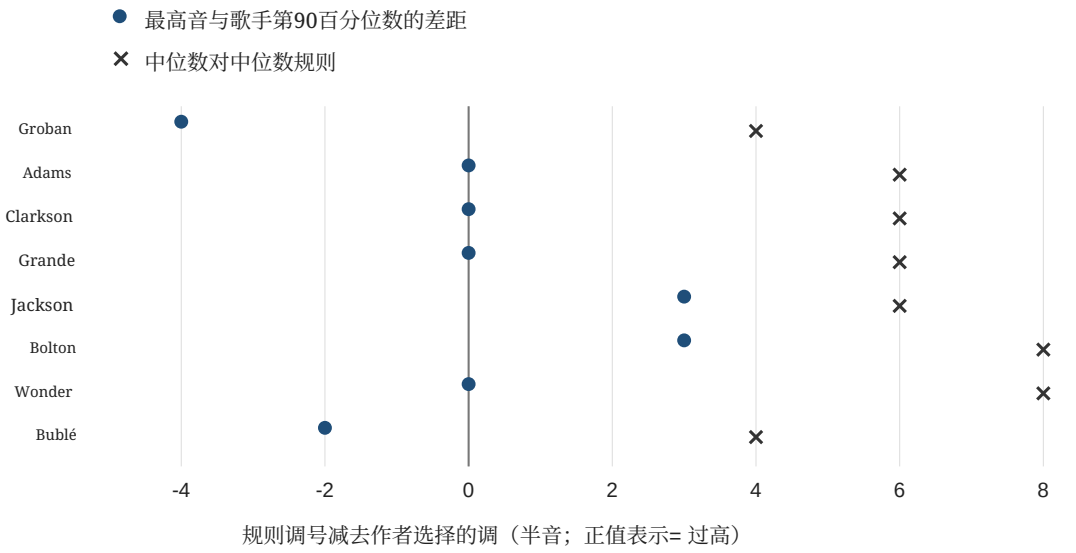

图 2： 相对于作者对 15 秒片段中八个复制的人声所做最终并排选择的音域规则误差。正误差表示音高过高。最高音规则的平均绝对误差为 1.50 个半音，而中位数匹配为 6.00 个半音。这些是作者内部的判断，并非总体估计。

6. 讨论与局限

访问开放音频工具。 ARRANGE 始于一位作曲家希望听到自己的伴奏上方有令人信服的人声。Suno 在作者的探索中作为音乐质量参考。ARRANGE 致力于构建一条围绕开放音频工具和已发布智能体技能的路径，旨在降低人工智能辅助作曲的门槛，同时让作曲家能够便捷地参与音乐决策。

技能与创作控制。 现有模型仍需协调：音符、歌词、发音和时序必须在工具之间传递，同时不能丢失作曲家的选择。技能将这项工作封装为音乐任务，并明确规定输入、输出和审批节点。乐谱和计划为作曲家提供了可供检查的决策；类型化编辑则为智能体提供了一种有界的方式来应用修正。基准测试支持局部修改正确性有所提升；观察到的 Fable 提升表明，前沿能力并不能让特定任务指令和类型化编辑变得多余。这种结构究竟能否减轻音乐人的工作负担，或增强其主观能动性，仍有待检验。

Huma判断与校准。 仅有人声样本并不能保证演唱效果令人信服 n

这些试验。找到合适的调可以改善同一人声的表现，作者还发现，有些人声更适合特定的歌曲风格。后一项观察需要在更广泛的曲目中进行验证。评估时应区分音色相似度与演唱质量，并使用共同的示例来锚定高、中、低评分。正如调的比较所显示的，保持试听条件一致也很重要。目前的单听众评估无法证明不同听众之间意见一致。

当前局限。 音乐示例涵盖两首乐曲和一位经验丰富的作曲家。模型基准测试采用小规模、由项目团队筛选的请求集，并进行重复抽样；其统计检验仅属探索性分析。乐谱读写能力和模型配置仍是使用门槛，评估的工作流程既包含商业大语言模型，也包含开放音频模型。我们尚未衡量配置所需的精力、成本、认知负荷或相对修订时间。更广泛的研究应考察这些门槛，并检验智能体是否始终遵守作曲家检查点。让作曲更易于参与仍是我们的设计目标。

权利与发布。 使用或分发音乐录音和采样人声的许可，与软件访问权限彼此独立。仅有同意记录并不能证明已获许可；未同意的歌手，其音频仍须保密。下文将说明发布限制及其对复现的影响。

7. 结论

ARRANGE 围绕易读的音乐中间表示和有界变换，组织为录制演奏添加歌唱人声的流程。计划修订结果表明，在经过测试的大语言模型中，类型化操作能够提高精确修改正确性。乐谱编辑结果也显示，理解和执行方面仍存在失败情况，包括提示操作与已安装技能的使用之间的差异。第一人称音乐评估表明，可以采用这样的工作流程：由智能体提出方案并生成渲染结果，再由作曲家检查、纠正并批准。还需与作曲家开展更广泛的评估，以确定这种结构是否能提升其自主性和创作实践。

数据与代码可用性

代码、存储的评估记录和复现说明可在github.com/htansetiawan/arrange获取。项目页面为htansetiawan.github.io/arrange。许多改写后的模型回答已截短，受版权保护的拟合数据、乐谱截图、参考录音以及部分歌曲未予发布。存储的汇总结果可以重新计算，但有些实验无法根据发布内容完整地重新评分或再现。

人声比较包含 35 次呈现，其中 30 次已公开发布，包括九组首次聆听配对、其他系统片段和隐藏重复项。已发布子集的平均分为 SoulX-Singer 95.4、Seed-VC 90.0；在九组配对中，SoulX 有七组得分更高（配对 Wilcoxon 检验 $p = 0.098$）。这些结果与第5.4节中的完整研究数值不同。音域实验的 185 次呈现包括 48 个试验片段、60 个随机排序的盲测片段、14 次盲测复查和 63 次排序比较；这些是呈现次数，而非独立听众人数。第5.2节区分了纠错基准测试的完整研究结果与已发布子集结果。

生成式人工智能声明

大语言模型智能体是所述系统的一个组成部分。Henry Tan 构思了 ARRANGE，设计了研究，做出了音乐和聆听方面的判断，并撰写了手稿初稿。Eidee Tan 开发并检查了生成的乐谱，并校对了手稿。在作者的指导下，Claude 协助完成了实现、实验执行、统计计算，以及对文本和参考资料的检查。ChatGPT 协助进行了文字编辑，并根据现有代码仓库和引用来源检查内容的一致性。人类作者负责核验最终手稿，并批准其提交。

参考资料

[1] Anthropic。2025。借助 Agent Skills 为智能体配备应对现实世界的能力。2025年10月16日。https: // www.anthropic.com/research/equipping-agents-for-the-real-world-with-agent-skills[2] Junmin Gong、Sean Zhao、Sen Wang、Shengyuan Xu 和 Joe Guo。2025。ACE-Step：迈向音乐生成基础模型。arXiv:2506.00045。https://doi.org/10.48550/arXiv.2506.00045[3] Junmin Gong、Yulin Song、Wenxiao Zhao、Sen Wang、Shengyuan Xu、Jing Guo 和 Xuerui Yang。2026。ACE-Step 1.5：推动开源音乐生成的边界。arXiv:2602.00744。https://doi.org/10.48550/arXiv.2602.00744[4] Cheng-Zhi Anna Huang、Hendrik Vincent Koops、Ed Newton-Rex、Monica Dinculescu 和 Carrie J. Cai。2020。AI Song Contest：歌曲创作中的人机协同创作。载于第21届国际音乐信息检索学会会议论文集。arXiv:2010.05388。https: //arxiv.org/abs/2010.05388[5] N. Kalibhat、Z. Wang、P. Bajpai、D. Proud、W. Zeng、B. Kim 和 M. Malek。2026。通过用于原子概念编辑的宪章解读和控制模型行为。载于第29届国际人工智能与统计学会议论文集。arXiv:2602.00092。https: //arxiv.org/abs/2602.00092[6] Qiuqiang Kong、Bochen Li、Xuchen Song、Yuan Wan 和 Yuxuan Wang。2021。通过回归起始与结束时间实现带踏板的高分辨率钢琴转录。IEEE/ACM音频、语音与语言处理汇刊 29，3707–3717。https://arxiv.org/abs/2010.01815[7] Songting Liu。2024。使用扩散Transformer进行零样本语音转换。arXiv:2411.09943。https://doi.org/10.48550/arXiv.2411.09943[8] Gianluca Micchi、Louis Bigo、Mathieu Giraud、Richard Groult 和 Florence Levé。2021。我一直在数拍：一项关于人机协同歌曲创作的实验。《国际音乐信息检索学会汇刊》4，1，263–275。https://doi.org/10.5334/tismir.93[9] Jiale Qian 等。2026。SoulX-Singer：迈向高质量零样本歌声合成。arXiv:2602.07803。https://doi.org/10.48550/arXiv.2602.07803[10] Antonio Ríos-Vila、Jorge Calvo-Zaragoza 和 Thierry Paquet。2024。乐谱Transformer：超越单声部转录的端到端光学音乐识别。载于《文档分析与识别——ICDAR 2024》。arXiv:2402.07596。https://arxiv.org/abs/2402.07596[11] Francesca Ronchini、Luca Comanducci、Simone Marcucci 和 Fabio Antonacci。2025。AI辅助音乐制作：一项关于文本生成音乐模型的用户研究。载于第17届计算机音乐多学科研究国际研讨会论文集。arXiv:2509.23364。https://arxiv.org/abs/ 2509.23364[12] Suno。2026。发行说明。访问于2026年11月8日。https://suno.com/release-notes[13] Tongshuang Wu、Michael Terry 和 Carrie J. Cai。2022。AI Chains：通过串联大语言模型提示词，实现透明且可控的人机交互。载于2022年CHI计算系统人因会议论文集。ACM。https://doi.org/10.1145/3491102.3517582[14] Ruibin Yuan 等。2025。YuE：扩展开放基础模型，实现长篇音乐生成。arXiv:2503.08638。https://doi.org/10.48550/arXiv.2503.08638[15] Ruibin Yuan 等。2026。YuE2：以业界前沿质量统一符号音乐与音频音乐生成。arXiv:2609.33757。https://arxiv.org/abs/2609.33757