# CompLex: Music Theory Lexicon Constructed by Autonomous Agents for Automatic Music Generation

**Zhejing Hu**[1], **Yan Liu**[1*], **Gong Chen**[1] and **Bruce X.B. Yu**[2]
[1] Department of Computing, The Hong Kong Polytechnic University
[2] Zhejiang University-University of Illinois Urbana-Champaign Institute
zhejing.hu@connect.polyu.hk, {yan.liu,gong-cg.chen}@polyu.edu.hk, xinboyu@intl.zju.edu.cn

## Abstract

Generative artificial intelligence in music has made significant strides, yet it still falls short of the substantial achievements seen in natural language processing, primarily due to the limited availability of music data. Knowledge-informed approaches have been shown to enhance the performance of music generation models, even when only a few pieces of musical knowledge are integrated. This paper seeks to leverage comprehensive music theory in AI-driven music generation tasks, such as algorithmic composition and style transfer, which traditionally require significant manual effort with existing techniques. We introduce a novel automatic music lexicon construction model that generates a lexicon, named CompLex, comprising 37,432 items derived from just 9 manually input category keywords and 5 sentence prompt templates. A new multi-agent algorithm is proposed to automatically detect and mitigate hallucinations. CompLex demonstrates impressive performance improvements across three state-of-the-art text-to-music generation models, encompassing both symbolic and audio-based methods. Furthermore, we evaluate CompLex in terms of completeness, accuracy, non-redundancy, and executability, confirming that it possesses the key characteristics of an effective lexicon.

## 1 Introduction

Generative artificial intelligence in music has seen remarkable progress in both academia [Copet *et al.*, 2024; Bhandari *et al.*, 2025; Hu *et al.*, 2025] and industry such as Suno [Suno, 2025] and Udio [Udio, 2025]. These advancements highlight the growing potential of artificial intelligence in the arts and creative industries [Wei *et al.*, 2023; Karystinaios *et al.*, 2023; Han *et al.*, 2024], offering transformative capabilities for music creation, production, and consumption.

Despite significant progress, the development of generative artificial intelligence in music still falls short of the significant achievements seen in natural language processing (NLP) [Wei *et al.*, 2022; Farquhar *et al.*, 2024]. This gap is mainly due to the limited availability of music data, especially compared to the super large-scale datasets used to train large language models (LLMs) [Briot *et al.*, 2017; Ji *et al.*, 2023]. For example, while LLMs can ensure grammatical accuracy through data-driven pre-training and prompt tuning, large music models (LMMs) face challenges in adhering to music theory using data alone. This discrepancy stems from the vast difference in data availability between language and music models. Billions of new text data points are generated daily, whereas only tens of thousands of new pieces of music, both original and adapted, are created. The limited quantity of music data remains a challenge, as the total number of musical pieces from various periods and styles throughout history amounts to only millions [1], starkly contrasting the trillion-level text corpora used to train LLMs [Achiam *et al.*, 2023]. This limited quantity of music data remains a key bottleneck in the progress of generative music models.

Data augmentation has been proposed as a way to expand music datasets and has achieved significant performance improvements [Shorten and Khoshgoftaar, 2019]. However, we observe a challenging cycle: effective data augmentation relies on high-performing models, but improving these models depends on high-quality data, which is difficult to generate. Specifically, high-quality music data requires efficient generative models, and if these models perform poorly, the augmented data may introduce noise, undermining further training [Shorten and Khoshgoftaar, 2019]. Additionally, human auditory perception is highly sensitive to small changes in timing, pitch, and volume [McDermott and Oxenham, 2008], leading to low tolerance for suboptimal augmented samples. This cycle affects not only music but also many other domains, such as medical imaging [Shamshad *et al.*, 2023] and finance [Cao, 2022], which also struggle with limited data availability.

In this context, knowledge-informed approaches are essential for enhancing the performance of music generation models. Music theory can significantly improve various aspects of music, including musicality [Akama, 2019], melody smoothness [Wu *et al.*, 2020], and overall structure [Dai *et al.*, 2020].

*Corresponding author.

[1] https://open.spotify.com/

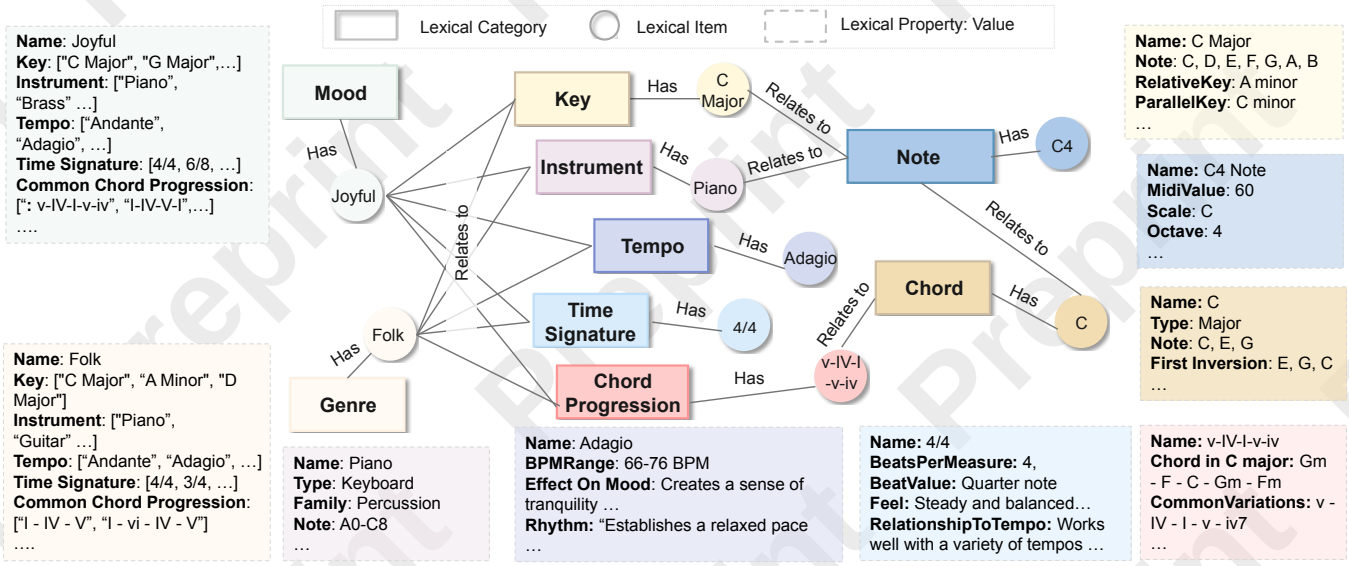



Figure 1: An illustration of CompLex, which contains 9 lexical categories and 37,432 lexical items, each associated with lexical properties and values. The figure displays one item from each category.

Several components of music theory have already been incorporated into generative models, such as music themes [Shih *et al.*, 2022], motif-level repetitions [Hu *et al.*, 2023], and phrase-level call-and-response structures [Hu *et al.*, 2024]. These works highlight that music theory can indeed boost music performance, even when only a few pieces of musical knowledge are integrated.

In this paper, we aim to leverage comprehensive music theory in AI-driven music generation tasks, such as algorithmic composition and style transfer, which have traditionally demanded considerable manual effort using existing methods. Specifically, we introduce a novel automatic approach to music lexicon construction that generates the music compositional lexicon, CompLex (Figure 1). CompLex is a comprehensive lexicon consisting of 9 categories, 90 properties, and 37,432 lexical items, all derived from just 9 manually input lexical category keywords and 5 prompt templates, significantly reducing reliance on human labor. To facilitate this process, we present LexConstructor, a new multi-agent algorithm designed to automatically detect and mitigate hallucinations during the generation of property-value pairs in the lexicon. CompLex shows impressive performance improvements across three state-of-the-art text-to-music generation models, covering both symbolic and audio-based methods. We also evaluate CompLex in terms of completeness, accuracy, non-redundancy, and executability, confirming that it satisfies the essential characteristics of an effective lexicon.

## 2 Related Work

### 2.1 Automatic Music Generation

Automatic music generation has seen significant advancements in recent years, particularly with the development of systems like Suno [Suno, 2025], which generates high-quality audio-based music from user-provided text input. In academia, models such as MusicGen [Copet *et al.*, 2024] are capable of generating high-quality audio-based music, while Text2Midi [Bhandari *et al.*, 2025] specializes in creating symbolic-based music using textual descriptions. Despite these advances, the challenge of limited data in the music domain remains a significant obstacle. To address this, knowledge-informed approaches have proven to be highly beneficial, as they integrate specific aspects of music theory to enhance generation performance, even with limited data samples. Notable works such as Transformer VAE [Jiang *et al.*, 2020], Melons [Wu *et al.*, 2020], MusicFrameworks [Dai *et al.*, 2021], and MeloForm [Lu *et al.*, 2022] have demonstrated the impact of incorporating music knowledge into generative models. In this paper, we aim to further advance AI-driven music generation by leveraging comprehensive music theory, specifically through the construction of a detailed music theory lexicon.

### 2.2 Autonomous Agents

Recently, autonomous agents have garnered significant interest, with notable success achieved by LLMs [Wang *et al.*, 2024; Durante *et al.*, 2024]. The applications of autonomous agents have quickly diversified into various tasks, such as software development [Qian *et al.*, 2024], biomedical discoveries [Gao *et al.*, 2024] and science debates [Cobbe *et al.*, 2021], and fields such as music [Tatar and Pasquier, 2019], psychology [Kovač *et al.*, 2023], and finance [Mayo *et al.*, 2024]. The strengths of the autonomous agent approach—specialization, and robust collaboration—make it particularly suitable for scenarios that require the integration of diverse expertise and coordination of multiple processes. To enable effective collaboration among agents, researchers have developed various communication strategies. For example, a role-reversal strategy is employed to mitigate hallucinations during direct agent communication [Qian *et al.*, 2024]. To enhance performance in open-ended discussion scenarios,

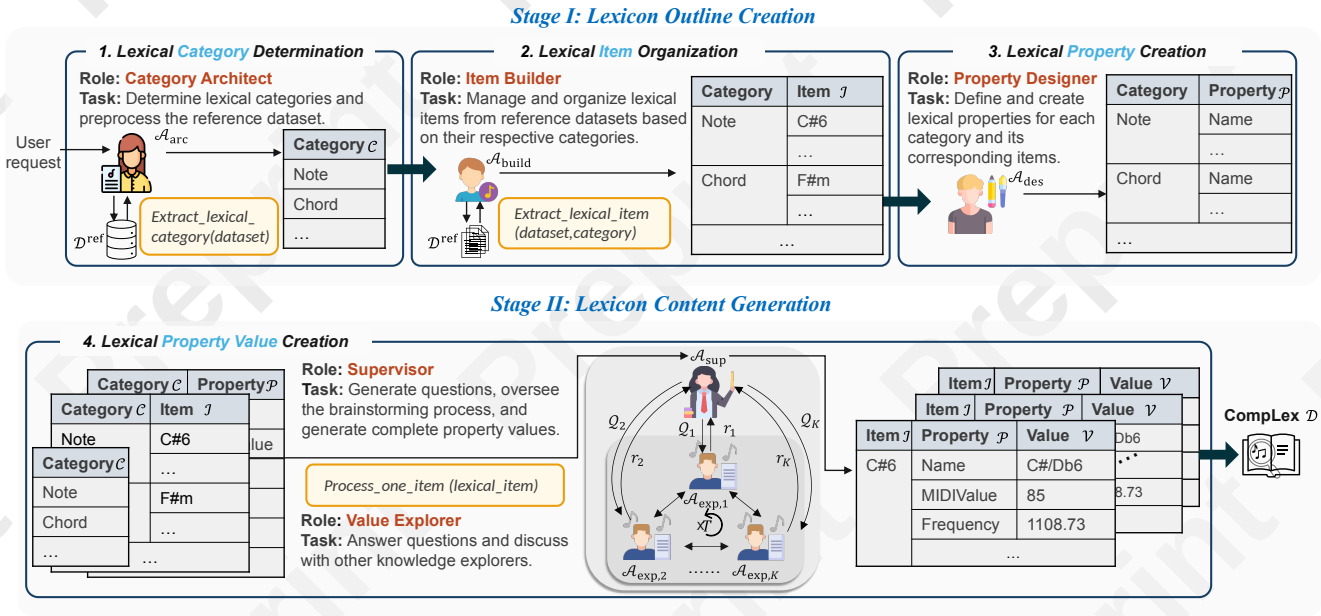



Figure 2: Overview of the LexConstructor algorithm, which includes two stages: Lexicon Outline Creation and Lexicon Content Generation. Stage I focuses on determining the structure of the lexicon, while Stage II populates it by assigning property-value pairs to items in all categories. The user provides the input request, and then five types of autonomous agents collaborate to create the lexicon.

a debating strategy allows agents to brainstorm and refine ideas through structured debate [Chan *et al.*, 2024]. The use of autonomous agents for automatically creating a comprehensive and compact lexicon remains underexplored. Traditional lexicon development relies on manual curation, which is time-consuming. To address this, we propose an automated music lexicon construction algorithm using multiple agents, reducing the need for costly human labor.

## 3 Problem Formulation

We introduce a structured lexicon to organize the vast, unstructured body of music-theory knowledge based on four key components: **Lexical Category**, **Lexical Item**, **Lexical Property**, and **Lexical Value**.

**Lexical Category.** A lexical category is a high-level concept in music theory that groups related items. Let $\mathcal{C}$ denote the set of all categories in our framework. Formally, we define:

$$\mathcal{C} = \{ c_1, c_2, \dots \}, \tag{1}$$

where each $c_j \in \mathcal{C}$ represents a distinct music-theory concept. Example categories include "Chord", "Key", and "ChordProgression", among others.

**Lexical Item.** A lexical item is a specific instance of a category. For each category $c \in \mathcal{C}$, let $\mathcal{I}(c)$ denote the set of items within that category. Formally, we define:

$$\mathcal{I}(c) = \{ i_1, i_2, \dots \}, \tag{2}$$

where each $i_j \in \mathcal{I}(c)$ represents a unique item under the concept $c$. For instance, under the "Chord" category, we might have "C Major", "A Minor", etc.

**Lexical Property.** A lexical property is a named attribute or feature that describes an item within a category. For each category $c \in \mathcal{C}$, let $\mathcal{P}(c)$ denote the set of properties associated with items in that category. Formally, we have:

$$\mathcal{P}(c) = \{ p_1, p_2, \dots \}, \tag{3}$$

where each $p_j \in \mathcal{P}(c)$ represents a distinct property under the concept $c$. For example, the "Chord" category might include properties such as "Name", "Type", and "Quality".

**Property-Value Pair.** Each lexical property $p_j$ of a lexical item can have a corresponding value, forming a property-value pair. For an item $i \in \mathcal{I}(c)$ within category $c$, we define the set of property-value pairs as:

$$\mathcal{V}(i, c) = \{ (p_1, v_1), (p_2, v_2), \dots \}, \tag{4}$$

where $v_j$ is the value associated with the property $p_j$. For example, for the item "C Major" in the "Chord" category, we might have:

$$\mathcal{V}(\text{C Major}, \text{Chord}) = \{ (\text{Name}, \text{C Major}), (\text{Type}, \text{Triad}), \dots \}.$$

**Objective.** Our objective is to design a model that can automatically identify and extract a comprehensive set of categories $\mathcal{C}$, the items of each category $\mathcal{I}(c)$, the properties of each category $\mathcal{P}(c)$, and generate the corresponding property-value pairs $\mathcal{V}(i, c)$ correctly from unstructured music-theory knowledge, and finally construct CompLex $\mathcal{D}$ based on a user request $u$.

## 4 LexConstructor

We propose LexConstructor, a multi-agent algorithm composed of several specialized LLM-based agents. Specifically, LexConstructor includes five distinct agent roles, with

each agent assigned specific roles and tasks, based on one prompt template. LexConstructor operates in two stages, as shown in Figure 2: Lexicon Outline Creation and Lexicon Content Generation. In the Lexicon Outline Creation stage, three types of agents collaboratively develop the lexicon's structural framework by determining the necessary categories, items, and properties. In the Lexicon Content Generation stage, two types of agents populate the lexicon by assigning property-value pairs to each item within all categories.

### 4.1 Stage I: Lexicon Outline Creation

Inspired by how humans construct a lexicon from scratch, we begin by creating an outline of the lexicon in Stage I. In this stage, LexConstructor focuses on establishing the foundational structure of the lexicon by completing three key subtasks: Lexical Category Determination, Lexical Item Organization, and Lexical Property Creation.

**Lexical Category Determination**

We employ a specialized agent, the **Category Architect** ($\mathcal{A}_{\text{arc}}$), to perform this subtask. The main responsibility of $\mathcal{A}_{\text{arc}}$ is to determine lexical categories based on user request $u$ and to pre-process the reference MIDI dataset $\mathcal{D}^{\text{ref}}$ into text-based structured information for future subtasks. To facilitate this process, $\mathcal{A}_{\text{arc}}$ utilizes a predefined function `Extract_lexical_category`($\mathcal{D}^{\text{ref}}$), which extracts lexical categories from the MIDI dataset based on the predefined categories. The agent analyzes the dataset, translating the MIDI information into a structured, text-based format, which will be used for the next subtask.

The output of $\mathcal{A}_{\text{arc}}$ is a set of categories $\mathcal{C}$, formally defined as: $\mathcal{C} = \mathcal{A}_{\text{arc}}(\mathcal{D}^{\text{ref}}, u)$, where $u$ is the user request and includes 9 category key words in the work, which are mood, genre, key, instrument, tempo, time signature, chord, and note. In addition the reference dataset $\mathcal{D}^{\text{ref}}$ is translated into structured, text-based information.

**Lexical Item Organization**

The **Item Builder** agent ($\mathcal{A}_{\text{builder}}$) is responsible for organizing lexical items within each category identified by $\mathcal{A}_{\text{arc}}$. Due to the limitations of LLMs in generating large, unique item lists efficiently, $\mathcal{A}_{\text{builder}}$ leverages the pre-defined function `Extract_lexical_item`$(c, \mathcal{D}^{\text{ref}})$ to assist in item extraction. Given a reference dataset $\mathcal{D}^{\text{ref}}$ and a target category $c$, this function helps the agent identify and extract all relevant items for that category. Formally, this step is represented as: $\mathcal{I}(c) = \mathcal{A}_{\text{build}}(c, \mathcal{D}^{\text{ref}})$.

**Lexical Property Creation**

The **Property Designer** agent ($\mathcal{A}_{\text{des}}$) is tasked with defining and creating lexical properties for each category and its corresponding items. Taking the generated categories and items from earlier steps, $\mathcal{A}_{\text{des}}$ defines a set of properties $\mathcal{P}(c)$ for each category. The result is a comprehensive set of properties that describe each category's items, formally expressed as: $\mathcal{P}(c) = \mathcal{A}_{\text{des}}(c, \mathcal{I}(c))$.

To ensure that the lexicon outline is both compact and comprehensive, the aforementioned subtasks are executed iteratively until all agents determine that no further refinement is needed. This iterative process allows the agents to refine their outputs based on the results from subsequent stages.

### 4.2 Stage II: Lexicon Content Generation

While the outline of the lexicon can be determined through tool usage and by utilizing information directly from the reference dataset, property-value pairs require the exploration of complex knowledge relations that can't be easily extracted from the dataset or hard-coded into the function. This makes value generation more susceptible to errors, such as inaccurate values assigned to properties. Therefore, we use the LLM's expert knowledge to generate these values during this stage. To further mitigate hallucinations and ensure accurate results, we design a Question-Answering (QA) communication strategy among agents.

**Lexical Property Value Creation**

We design two different types of agents: the **Supervisor Agent** ($\mathcal{A}_{\text{sup}}$), which oversees the property value generation process, ensures consistency, and manages quality control, and the **Value Explorer Agents** ($\mathcal{A}_{\text{exp}}$), which provide detailed information and populate property values through collaborative exploration. To handle the large number of items to be populated into the lexicon, we define a function `Process_One_Item`$(\mathcal{I})$ that allows the agent to process one item at a time, thus avoiding memory constraints of LLMs.

Formally, for each item $i \in \mathcal{I}(c)$, the value $\mathcal{V}(i, c)$ is generated as follows: $\mathcal{V}(i, c) = \mathcal{A}_{\text{sup}}\left(i, \mathcal{P}(c)\right)$, where $\mathcal{A}_{\text{sup}}$ takes one item $i$ at a time and generates the value corresponding to the property $\mathcal{P}(c)$.

The entire lexicon population process involves iterating through all categories and items, which is represented as:

$$\mathcal{D} = \bigcup_{c \in \mathcal{C}} \bigcup_{i \in \mathcal{I}(c)} \mathcal{V}(i, c). \tag{5}$$

**Question Answering Communication**

The Supervisor Agent formulates targeted questions, denoted as $\mathcal{Q}$, and poses them to $K$ Value Explorer Agents. These agents then collaborate, brainstorming and providing answers to the questions. The Supervisor Agent evaluates the responses and generates the final property-value pair $\mathcal{V}(i, c)$. The process can be formalized as:

$$\begin{cases} \mathcal{A}_{\text{sup}} \xrightarrow{\{\mathcal{Q}_j\}_{j=1}^{K}} \{\mathcal{A}_{\text{exp},j}^{T}\}_{j=1}^{K} \xrightarrow{\{r_j\}_{j=1}^{K}} \mathcal{A}_{\text{sup}}, \\ \{\mathcal{A}_{\text{exp},j}^{0}\}_{j=1}^{K} \xrightarrow{\{\mathcal{Q}_j\}_{j=1}^{K}} \cdots \xrightarrow{\{r_j^{T-1}\}_{j=1}^{K}} \{\mathcal{A}_{\text{exp},j}^{T}\}_{j=1}^{K} \end{cases} \tag{6}$$

where:

- $\mathcal{Q} = \{q_1, q_2, \ldots\}$ is the set of diverse questions generated by the Supervisor Agent focus on item $i$.
- $\mathcal{Q}_j \subseteq \mathcal{Q}$ is the subset of questions assigned to $\mathcal{A}_{\text{exp},j}$.
- $r_j$ is the response from $\mathcal{A}_{\text{exp},j}^{T}$ to its assigned questions at iteration $T$.

### 4.3 Text-to-Music Generation Application

The lexicon can be applied in text-to-music generation by refining the user prompt. Given a user prompt $u$, the system queries the lexicon $\mathcal{D}$ to retrieve related items $\mathcal{I}(c_u)$, where $c_u$ is the category corresponding to $u$. For each item $i_j$ in $\mathcal{I}(c_u)$, we extract its properties $\mathcal{P}(i_j)$, which are then used to find additional related items $\mathcal{I}(c_k)$ in other categories, expanding the context. The music is generated by using the related lexical items $\mathcal{I}(c_u) \cup \mathcal{I}(c_k)$. For example, if $u =$ "happy mood", items from the "Mood" category can be quickly retrieved and then related information such as tempo (120 bpm), key (C major), and chord progression ("I-IV-V-V"), among others, can be further retrieved.

## 5 Experiment

In this section, we provide the experimental setup used to evaluate the performance of CompLex on text-to-music generation tasks and the performance of LexConstructor method.

### 5.1 Implementation Details

**Reference Dataset:** We select the **MidiCaps** dataset [Melechovsky *et al.*, 2024] as reference dataset, which is derived from the Lakh MIDI dataset [Raffel, 2016], one of the largest open-source collections of symbolic music data. MidiCaps contains 168,385 MIDI samples.

**CompLex Statistics:** The Genre category has 42 items and 13 properties; Mood has 48 items and 9 properties; Key has 24 items and 13 properties; Tempo has 13 items and 10 properties; Time Signature has 86 items and 12 properties; Instrument has 101 items and 8 properties; Chord has 193 items and 8 properties; Chord Progression has 36,797 items and 12 properties; and Note has 128 items and 5 properties.

**Language Models:** We test two versions of LLMs: **GPT-3.5-turbo-0125** and **GPT-4o**. For lexicon content generation, we utilize **GPT-4o**, selected for its superior efficiency and larger context window, supporting up to 16,384 tokens.

**Model Configurations:** All agents in the system are configured with a temperature setting of 0 [Chan *et al.*, 2024]. The agents are assigned specific roles and tasks, with 5 prompt templates where each agent role corresponds to one prompt template, as shown in the Appendix. The user inputs 9 specific lexical categories in the experiment for comparison. The loop in $T$ continues until the agents reach a consensus, at which point no further information is added. For the QA conversation strategy, we utilize $K = 3$ value explorers to facilitate brainstorming, enabling a thorough evaluation and refinement of the lexicon entries. For data analysis in the function $\texttt{Extract\_lexical\_category}(\mathcal{D}^{\text{ref}})$, we follow implementations in [Melechovsky *et al.*, 2024].

### 5.2 Baselines

For evaluating **CompLex** in text-to-music music generation tasks, we assessed its performance using three models:

- **Text2MIDI** [Bhandari *et al.*, 2025], a SOTA open-source text-to-symbolic music generation model trained on the MidiCaps dataset;
- **MusicGen** [Copet *et al.*, 2024], a SOTA open-source text-to-audio music generation model in academia (we used the large version);
- **Suno** [Suno, 2025], a SOTA black-box text-to-audio music generation model in industry.

These baselines represent the most advanced models in both symbolic and audio music generation.

To validate **LexConstructor**, we compared it against several baselines and state-of-the-art (SOTA) approaches [Islam *et al.*, 2024], including both single-agent and multi-agent models:

- **Direct Prompting**, where language models generate the music theory lexicon without explicit guidance.
- **Chain of Thought Prompting (CoT)** [Wei *et al.*, 2022] breaks down tasks into step-by-step subtasks, facilitating the handling of more complex problems.
- **Analogical Reasoning Prompting** [Yasunaga *et al.*, 2024] directs models to draw upon relevant past experiences to solve similar issues.
- **Multi-Agent Role Reverse** [Qian *et al.*, 2024], where agents reverse roles to collaborate on problem-solving.
- **Multi-Agent Debate** [Chan *et al.*, 2024], where agents engage in debate to find the optimal solution.

### 5.3 Evaluation Metrics

**Objective Metrics**

To assess the performance of **CompLex** on text-to-music generation tasks, we evaluate the following aspects: **Compression Ratio (CR):** Measures the structure of the music [Chuan and Herremans, 2018; Bhandari *et al.*, 2025]. **Contrastive Language-Audio Pre-training (CLAP):** Measures how closely the music aligns with the corresponding text caption [Bhandari *et al.*, 2025; Copet *et al.*, 2024]. **Mood Accuracy (MA):** Measures how effectively the generated music reflects the mood indicated in the textual description [Melechovsky *et al.*, 2024]. **Genre Accuracy (GA):** Measures how effectively the generated music reflects the genre in the textual description [Melechovsky *et al.*, 2024].

To assess the performance of **LexConstructor** and other baselines, we evaluate the effectiveness of the generated lexicon based on the following aspects: **Completeness:** Measures how thoroughly the generated lexical items in each category align with the actual items in music theory. The completeness score is averaged across all categories, with higher scores indicating a greater coverage of the relevant items. **Accuracy:** Measures the alignment between the generated MIDI pitch values and frequency values relative to their corresponding note names. Higher scores reflect better accuracy and adherence to established music theory. **Non-Redundancy:** Measures the level of non-duplication in the generated lexical items, calculated as the ratio of unique items to the total number of items. Higher scores indicate fewer redundant entries. **Executability:** Measures the lexicon's ability to be loaded error-free. Higher scores indicate greater reliability and functionality.

| Baseline | Method | Objective Metrics | | | | Subjective Metrics | |
|---|---|---|---|---|---|---|---|
| | | CR↑ | CLAP↑ | MA↑ | GA↑ | REL↑ | OVL↑ |
| Text2MIDI [Bhandari *et al.*, 2025] | W/o Enhancement | 2.02 | 0.17 | 0.37 | 0.66 | 70.13±7.22 | 73.54±5.14 |
| | LLM-Enhanced | 2.07 | 0.24 | 0.41 | 0.66 | 76.54±4.55 | 75.50±2.93 |
| | **CompLex-Enhanced** | **2.14** | **0.35** | **0.47** | **0.67** | **78.10±4.37** | **77.46±4.71** |
| MusicGen [Copet *et al.*, 2024] | W/o Enhancement | – | 0.22 | 0.19 | 0.46 | 75.83±4.43 | 78.17±3.57 |
| | LLM-Enhanced | – | 0.25 | 0.26 | 0.47 | 80.17±7.72 | 81.20±7.96 |
| | **CompLex-Enhanced** | – | **0.33** | **0.28** | **0.51** | **82.67±4.29** | **82.17±4.45** |
| Suno [Suno, 2025] | W/o Enhancement | – | 0.39 | 0.44 | 0.67 | 82.79±3.79 | 86.75±4.63 |
| | LLM-Enhanced | – | 0.39 | 0.43 | 0.52 | 83.88±6.05 | 88.67±3.93 |
| | **CompLex-Enhanced** | – | **0.46** | **0.57** | **0.79** | **91.38±3.84** | **93.58±4.13** |

Table 1: Performance of CompLex and other methods across different text-to-music generation models. CompLex-Enhanced method consistently outperforms other methods, indicating the effectiveness of CompLex. The Compression Ratio (CR) metric can only be applied to symbolic music representations (e.g., MIDI) and is not suitable for audio-based music.

**Subjective Metrics**

Human evaluators assess the generated music from the music generation model based on **Relevance to Text Input (REL)** and **Overall Quality (OVL)** [Kreuk *et al.*, 2022; Copet *et al.*, 2024] from 1 to 100 where we emphasize musicality in OVL.

An online survey was distributed via social media, receiving 24 valid responses after excluding invalid ones (e.g., participants selecting the same option for all questions or submitting incomplete responses). All participants were fully informed about the purpose of the study and consented to the use of their data for research purposes. Detailed information on the design of the subjective experiment is provided in the Appendix.

## 6 Results and Discussion

This section presents the results and discussion of our experiments. First, we demonstrate the impact of CompLex on text-to-music generation tasks by measuring the quality of the output music. Next, we demonstrate the effectiveness of LexConstructor by evaluating the quality of the generated composition lexicon in terms of completeness, accuracy, non-redundancy, and executability. Additionally, we conduct an ablation analysis to validate the effectiveness of the multi-agent design to reduce redundancy and the QA communication to mitigate hallucinations.

### 6.1 Effectiveness of CompLex

We evaluate the performance of CompLex by refining the user text input on text-to-music generation models. We compare three different knowledge enhancement methods: **W/o Enhancement**, **LLM-Enhanced**, and **CompLex-Enhanced**, which use raw input without any refinement, LLMs, and CompLex to refine the user input. In these methods, the user specifies mood or genre requests.

Table 1 summarizes the performance of the three baseline models across these methods. The metrics include Compression Ratio (measuring structural compactness), CLAP (measuring alignment between text and audio), Mood Accuracy, Genre Accuracy, Relevance to Input, and Overall Musicality. Across all baseline models, the CompLex-Enhanced method consistently achieves higher performance in all metrics, highlighting the benefits of integrating a music theory lexicon for generating high-quality music. These improvements are primarily driven by the lexicon's ability to provide specific and structured guidance during the generation process.

The LLM-Enhanced method generally underperforms compared to the CompLex-Enhanced method, primarily due to the lack of domain-specific music theory integration. Additionally, Suno's performance shows improvements when the lexicon is incorporated, underscoring the importance of leveraging structured music knowledge to achieve improvement in both objective and subjective metrics. These findings demonstrate the critical role of CompLex in advancing AI-driven music generation.

### 6.2 Effectiveness of LexConstructor

Table 2 compares the performance of LexConstructor with other baseline models across two different backbones. The evaluation metrics include completeness, accuracy, non-redundancy, and executability. LexConstructor outperforms all baseline models in every metric, highlighting its superior efficiency in generating a music theory lexicon. In addition, the improvements demonstrate LexConstructor's ability to mitigate hallucinations. These gains are largely due to the multi-agent role-playing system, where each agent specializes in specific tasks, and the Question-Answering (QA) communication strategy, which effectively mitigates hallucinations during value generation. The multi-agent approach consistently outperforms single-agent models, underscoring the effectiveness of using multiple agents for complex tasks. Overall, LexConstructor shows strong potential for automatically constructing high-quality lexicons, with applications extending to other domains in the future.

### 6.3 Ablation Analysis

**Ablation Analysis of Multi-Agent:** Figure 3 illustrates our model's efficiency in terms of non-redundancy, showing its performance while generating 1000 lexical items with property values. Single-agent models begin to produce redundant outputs after generating about 50 unique items, with the issue being even more pronounced in the GPT-3.5-Turbo model. The superior performance of our approach can be attributed to its unique design, which incorporates specialized tool usage and task-specific subtasks. By focusing on generating one lexical item at a time, our method avoids referencing pre-

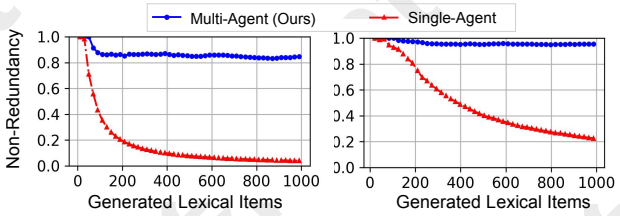
| Backbone | Method | Completeness↑ | Accuracy↑ | Non-Redundancy↑ | Executability↑ |
|---|---|---|---|---|---|
| GPT-3.5 -Turbo | *Single-Agent:* | | | | |
| | Direct Prompting | 0.54 | 0.22 | 0.05 | 0.53 |
| | CoT Prompting [Wei *et al.*, 2022] | 0.59 | 0.35 | 0.11 | 0.58 |
| | Analogical Prompting [Yasunaga *et al.*, 2024] | 0.59 | 0.38 | 0.13 | 0.60 |
| | *Multi-Agent:* | | | | |
| | Role Reverse [Qian *et al.*, 2024] | 0.65 | 0.64 | 0.19 | 0.66 |
| | Debate [Chan *et al.*, 2024] | 0.72 | 0.68 | 0.21 | 0.68 |
| | **LexConstructor (Ours)** | **0.80** | **0.78** | **0.84** | **0.75** |
| GPT-4o | *Single-Agent:* | | | | |
| | Direct Prompting | 0.60 | 0.32 | 0.21 | 0.55 |
| | CoT Prompting [Wei *et al.*, 2022] | 0.64 | 0.47 | 0.24 | 0.64 |
| | Analogical Prompting [Yasunaga *et al.*, 2024] | 0.65 | 0.48 | 0.28 | 0.63 |
| | *Multi-Agent:* | | | | |
| | Role Reverse [Qian *et al.*, 2024] | 0.72 | 0.72 | 0.31 | 0.72 |
| | Debate [Chan *et al.*, 2024] | 0.76 | 0.75 | 0.30 | 0.74 |
| | **LexConstructor (Ours)** | **0.83** | **0.88** | **0.95** | **0.82** |

Table 2: LexConstructor outperforms all baseline models across four key metrics—completeness, accuracy, non-redundancy, and executability—demonstrating its superior capability in generating a high-quality music theory lexicon.

| Strategy | Accuracy | |
|---|---|---|
| | Pitch Value↑ | Pitch Frequency↑ |
| w/o Communication | 0.28 | 0.34 |
| w/ Role Reverse | 0.72 | 0.72 |
| w/ Debating | 0.75 | 0.74 |
| **w/ QA** | **0.85** | **0.91** |

Table 3: Ablation Analysis of the QA Communication Strategy: It generates property values with higher accuracy, effectively mitigating hallucinations.

viously produced content. While a few duplicates still occur, this is due to the probabilistic nature of GPT models, which sometimes generate content similar to prior outputs.




(a)Non-Redundancy VS Generated Lexical Items (GPT-3.5-Turbo) (b) Non-Redundancy VS Generated Lexical Items (GPT-4o)

Figure 3: Ablation Analysis of the Multi-Agent Model: Our multi-agent approach generates lexical items with a higher non-redundancy rate, indicating a better performance.

**Ablation Analysis of QA Communication Strategy:** Table 3 shows different communication strategies among agents. It indicates that the QA strategy outperforms other communication strategies, achieving the best performance in mitigating hallucinations that occur during property-value creation. This is because the diverse questions cover different aspects of the concept, and the value explorers collaboratively exchange and refine their answers until reaching a consensus, leading to more accurate and coherent results.

### 6.4 Case Study on Hallucination Mitigation

Figure 4 presents a case study illustrating how LexConstructor effectively mitigates harmonic hallucinations. The note information in CompLex is correct and complete while generated from LLM-generated lexicon is incorrect and incomplete.

| Category | Item | Property | Value |
|---|---|---|---|
| Note | C#6 | MIDIValue | 73 |
| | | Frequency | 554.37 |

(a) Content of LLM-generated lexicon: Contains incomplete and incorrect entries. Red indicates incorrect values.

| Category | Item | Property | Value |
|---|---|---|---|
| Note | C#6 | MIDIValue | 85 |
| | | Frequency | 1108.73 |
| | | More Properties … | |

(b) Content of MusicLex: complete and accurate items with structured representation. Green indicates correct values.

Figure 4: A case study demonstrating LexConstructor mitigate hallucinations.

## 7 Conclusion

In this work, we introduce a novel automatic music lexicon construction model that generates CompLex, a comprehensive lexicon containing 9 lexical categories, 90 lexical properties, and 37,432 lexical items, all derived from just 9 manually input category keywords and 5 prompt templates. We also propose a new multi-agent algorithm, LexConstructor, that generates the lexicon and automatically detects and mitigates hallucinations during the generation of property-value pairs for each lexical item. CompLex demonstrates significant performance improvements across three state-of-the-art text-to-music generation models, encompassing both symbolic and audio-based methods. Furthermore, we evaluate CompLex in terms of completeness, accuracy, non-redundancy, and executability, confirming that it meets the key characteristics of an effective lexicon. CompLex can be applied to a broader range of tasks within the music domain, including algorithmic composition, and style transfer. Currently, CompLex focuses on the structure of music composition, in the future, we plan to expand its scope to encompass expressive elements of music performance.

## Acknowledgments

This work is supported by the project Towards Next-generation Artificial Auditory System with Brain-inspired Technologies (C5052-23GF).

## References

[Achiam *et al.*, 2023] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[Akama, 2019] Taketo Akama. Controlling symbolic music generation based on concept learning from domain knowledge. In *ISMIR*, pages 816–823, 2019.

[Bhandari *et al.*, 2025] Keshav Bhandari, Abhinaba Roy, Kyra Wang, Geeta Puri, Simon Colton, and Dorien Herremans. text2midi: Generating symbolic music from captions. In *AAAI*, 2025.

[Briot *et al.*, 2017] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*, 2017.

[Cao, 2022] Longbing Cao. Ai in finance: challenges, techniques, and opportunities. *ACM Computing Surveys (CSUR)*, 55(3):1–38, 2022.

[Chan *et al.*, 2024] Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate. In *ICLR*, 2024.

[Chuan and Herremans, 2018] Ching-Hua Chuan and Dorien Herremans. Modeling temporal tonal relations in polyphonic music through deep networks with a novel image-based representation. In *AAAI*, volume 32, 2018.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[Copet *et al.*, 2024] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. *NeurIPS*, 36, 2024.

[Dai *et al.*, 2020] Shuqi Dai, Huan Zhang, and Roger B Dannenberg. Automatic analysis and influence of hierarchical structure on melody, rhythm and harmony in popular music. *arXiv preprint arXiv:2010.07518*, 2020.

[Dai *et al.*, 2021] Shuqi Dai, Zeyu Jin, Celso Gomes, and Roger B Dannenberg. Controllable deep melody generation via hierarchical music structure representation. In *ISMIR*, 2021.

[Durante *et al.*, 2024] Zane Durante, Qiuyuan Huang, Naoki Wake, Ran Gong, Jae Sung Park, Bidipta Sarkar, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Yejin Choi, et al. Agent ai: Surveying the horizons of multimodal interaction. *arXiv preprint arXiv:2401.03568*, 2024.

[Farquhar *et al.*, 2024] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

[Gao *et al.*, 2024] Shanghua Gao, Ada Fang, Yepeng Huang, Valentina Giunchiglia, Ayush Noori, Jonathan Richard Schwarz, Yasha Ektefaie, Jovana Kondic, and Marinka Zitnik. Empowering biomedical discovery with ai agents. *Cell*, 187(22):6125–6151, 2024.

[Han *et al.*, 2024] Bing Han, Junyu Dai, Weituo Hao, Xinyan He, Dong Guo, Jitong Chen, Yuxuan Wang, Yanmin Qian, and Xuchen Song. Instructme: an instruction guided music edit framework with latent diffusion models. In *IJCAI*, pages 5835–5843, 2024.

[Hu *et al.*, 2023] Zhejing Hu, Xiao Ma, Yan Liu, Gong Chen, Yongxu Liu, and Roger B Dannenberg. The beauty of repetition: an algorithmic composition model with motif-level repetition generator and outline-to-music generator in symbolic music generation. *IEEE Transactions on Multimedia*, 2023.

[Hu *et al.*, 2024] Zhejing Hu, Yan Liu, Gong Chen, Xiao Ma, Shenghua Zhong, and Qianwen Luo. Responding to the call: Exploring automatic music composition using a knowledge-enhanced model. In *AAAI*, volume 38, pages 521–529, 2024.

[Hu *et al.*, 2025] Zhejing Hu, Yan Liu, Gong Chen, and Bruce XB Yu. Compose with me: Collaborative music inpainter for symbolic music infilling. In *AAAI*, volume 39, pages 1327–1335, 2025.

[Islam *et al.*, 2024] Md. Ashraful Islam, Mohammed Eunus Ali, and Md. Rizwan Parvez. Mapcoder: Multi-agent code generation for competitive problem solving. In *ACL*, pages 4912–4944. Association for Computational Linguistics, 2024.

[Ji *et al.*, 2023] Shulei Ji, Xinyu Yang, and Jing Luo. A survey on deep learning for symbolic music generation: Representations, algorithms, evaluations, and challenges. *ACM Computing Surveys*, 56(1):1–39, 2023.

[Jiang *et al.*, 2020] Junyan Jiang, Gus G Xia, Dave B Carlton, Chris N Anderson, and Ryan H Miyakawa. Transformer vae: A hierarchical model for structure-aware and interpretable music representation learning. In *ICASSP*, pages 516–520, 2020.

[Karystinaios *et al.*, 2023] Emmanouil Karystinaios, Francesco Foscarin, and Gerhard Widmer. Musical voice separation as link prediction: modeling a musical perception task as a multi-trajectory tracking problem. In *IJCAI*, pages 3866–3874, 2023.

[Kovač *et al.*, 2023] Grgur Kovač, Rémy Portelas, Peter Ford Dominey, and Pierre-Yves Oudeyer. The socialai school: Insights from developmental psychology towards artificial socio-cultural agents. *arXiv preprint arXiv:2307.07871*, 2023.

[Kreuk *et al.*, 2022] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet,

Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*, 2022.

[Lu *et al.*, 2022] Peiling Lu, Xu Tan, Botao Yu, Tao Qin, Sheng Zhao, and Tie-Yan Liu. Meloform: Generating melody with musical form based on expert systems and neural networks. *arXiv preprint arXiv:2208.14345*, 2022.

[Mayo *et al.*, 2024] Katherine Mayo, Nicholas Grabill, and Michael P. Wellman. Fraud risk mitigation in real-time payments: A strategic agent-based analysis. In *IJCAI*, pages 157–165. ijcai.org, 2024.

[McDermott and Oxenham, 2008] Josh H McDermott and Andrew J Oxenham. Music perception, pitch, and the auditory system. *Current opinion in neurobiology*, 18(4):452–463, 2008.

[Melechovsky *et al.*, 2024] Jan Melechovsky, Abhinaba Roy, and Dorien Herremansn. Midicaps: A large-scale midi dataset with text captions. In *ISMIR*, 2024.

[Qian *et al.*, 2024] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. Chatdev: Communicative agents for software development. In *ACL*, pages 15174–15186, 2024.

[Raffel, 2016] C Raffel. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching. 331 Ph. D*. PhD thesis, thesis, Columbia University, 2016.

[Shamshad *et al.*, 2023] Fahad Shamshad, Salman Khan, Syed Waqas Zamir, Muhammad Haris Khan, Munawar Hayat, Fahad Shahbaz Khan, and Huazhu Fu. Transformers in medical imaging: A survey. *Medical Image Analysis*, 88:102802, 2023.

[Shih *et al.*, 2022] Yi-Jen Shih, Shih-Lun Wu, Frank Zalkow, Meinard Muller, and Yi-Hsuan Yang. Theme transformer: Symbolic music generation with theme-conditioned transformer. *IEEE Transactions on Multimedia*, 2022.

[Shorten and Khoshgoftaar, 2019] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[Suno, 2025] Suno. Official website. https://suno.com, 2025. Accessed: 2025-01-01.

[Tatar and Pasquier, 2019] Kıvanç Tatar and Philippe Pasquier. Musical agents: A typology and state of the art towards musical metacreation. *Journal of New Music Research*, 48(1):56–105, 2019.

[Udio, 2025] Udio. Official website. https://www.udio.com, 2025. Accessed: 2025-01-01.

[Wang *et al.*, 2024] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.

[Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 35:24824–24837, 2022.

[Wei *et al.*, 2023] Haojie Wei, Jun Yuan, Rui Zhang, Yueguo Chen, and Gang Wang. Jepoo: highly accurate joint estimation of pitch, onset and offset for music information retrieval. In *IJCAI*, pages 4892–4902, 2023.

[Wu *et al.*, 2020] Jian Wu, Xiaoguang Liu, Xiaolin Hu, and Jun Zhu. Popmnet: Generating structured pop music melodies using neural networks. *Artificial Intelligence*, 286:103303, 2020.

[Yasunaga *et al.*, 2024] Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H Chi, and Denny Zhou. Large language models as analogical reasoners. In *ICLR*, 2024.