AJAMT: REAL-TIME MULTI-AGENT MUSIC GENERATION WITH PREDICTIVE LOOKAHEAD AND FALLBACK SCHEDULING

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-time music accompaniment requires generative models to produce musically coherent responses under strict timing constraints, yet the variable inference latency of large language models (LLMs) makes their use in live interaction challenging. We present **AJAMT**, a real-time multi-agent framework for multi-instrument symbolic music accompaniment. AJAMT combines predictive generation with buffered scheduling and fallback mechanisms to decouple individual generation deadline misses from playback interruptions, while a lightweight symbolic evaluator reduces the overhead of iterative agentic refinement. Across currently available inference backends, the fastest evaluated configuration achieves a mean end-to-end latency of 1.21 seconds and maintains 100% playback continuity at 120 BPM. Under a multi-tempo stress test, AJAMT further maintains 100% playback continuity at 180 BPM despite 24.4% of generation requests missing their deadlines. Under a controlled shared-melody evaluation, AJAMT also remains competitive with the offline ComposerX baseline across the evaluated audio-aesthetic and symbolic-structure metrics.

1 INTRODUCTION

Symbolic music accompaniment aims to generate complementary musical parts conditioned on an ongoing melody, and has been increasingly explored for human–AI music interaction. Recent large language models (LLMs) have demonstrated strong capabilities in understanding and generating structured symbolic music, making them a promising foundation for interactive accompaniment. However, directly applying LLMs to live music performance introduces two key challenges. **1) Real-time generation.** Unlike offline music generation, live accompaniment imposes explicit playback deadlines: each accompaniment segment must be generated before the corresponding musical position is reached. The variable inference latency of LLMs makes satisfying such deadlines difficult, especially as the available generation window decreases at higher tempos. **2) Multi-instrument generation.** Realistic accompaniment requires multiple instruments to remain harmonically and rhythmically coordinated with the incoming melody. Although agentic workflows can decompose this task into specialized reasoning and generation stages, iterative communication and self-refinement introduce additional inference overhead. Therefore, an interactive accompaniment system must balance musical quality and multi-instrument coordination with the strict timing requirements of live performance.

Existing music generation approaches address different aspects of these challenges, but are not designed to satisfy them simultaneously. Offline symbolic generation models have demonstrated the ability to generate polyphonic or multi-track music and capture musical structure over extended contexts Hadjeres et al. (2017); Dong et al. (2018); Huang et al. (2019). More recent generative models further improve controllability through textual or musical conditioning Copet et al. (2023). However, these methods primarily focus on generation quality, controllability, or long-term structure without explicitly considering the playback deadlines imposed by live interaction. On the other hand, online accompaniment systems such as RL-Duet Jiang et al. (2020) formulate accompaniment as a sequential response to human performance, demonstrating the feasibility of real-time machine accompaniment. Such approaches, however, employ specialized generation policies and restricted musical representations, making it difficult to exploit the flexible reasoning and multi-instrument

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

generation capabilities of modern LLMs. Meanwhile, multi-agent LLM systems such as ComposerX (Deng et al., 2024) demonstrate the potential of specialized agents for coordinated symbolic music composition, but their iterative planning and refinement workflows are designed primarily for offline generation. Directly adapting such workflows to live accompaniment is problematic because multiple sequential LLM calls can accumulate latency before playable material becomes available. It is therefore desirable to design an LLM-based accompaniment framework that retains flexible multi-instrument generation while remaining responsive under the temporal constraints of live performance.

In this paper, we propose **AJAMT**, a real-time multi-agent framework for symbolic music accompaniment. Our key observation is that real-time accompaniment does not require LLM inference itself to be instantaneous: generation latency can be partially hidden by predicting accompaniment ahead of its scheduled playback time. Based on this observation, AJAMT organizes interaction at the musical bar level and maintains a sliding context of the recent human performance for causal accompaniment generation. A real-time buffer schedules accompaniment generation ahead of the target bar, allowing LLM inference to proceed while previously prepared music is being played. To support coordinated multi-instrument generation without the latency of iterative multi-agent refinement, AJAMT separates harmonic planning and accompaniment generation into two stages and uses a lightweight symbolic evaluator to validate generated outputs. When a newly generated accompaniment is unavailable at its playback deadline, previously buffered musical material can be used as a fallback to avoid interrupting the performance. In this way, AJAMT shifts the requirement from making every individual LLM request sufficiently fast to maintaining usable musical output at each playback deadline.

Our main contributions are summarized as follows:

- We formulate real-time LLM-based accompaniment as a deadline-constrained generation problem in which inference deadlines and playback continuity are treated as distinct system properties.
- We introduce AJAMT, which combines causal sliding-window inference, predictive generation, real-time buffering, and fallback scheduling to reduce the dependence of continuous playback on the latency of individual LLM requests.
- We develop a coordinated multi-agent generation pipeline with lightweight symbolic evaluation, enabling harmonic planning and multi-instrument accompaniment without iterative LLM-based self-reflection.
- We evaluate AJAMT across multiple inference backends and playback tempos using 160 bar-level observations per model–tempo condition. With the fastest currently evaluated backend, AJAMT achieves a mean end-to-end latency of 1.21 seconds. At 180 BPM, playback continuity remains at 100% despite a 24.4% generation deadline miss rate, demonstrating that individual inference misses need not propagate into playback interruptions.

2 Related Work

2.1 Symbolic Music Generation

Symbolic music generation represents music as discrete structured events such as pitches, durations, instruments, and temporal positions. Earlier neural approaches explored polyphonic and multi-track generation using representations ranging from piano rolls to event sequences. DeepBach (Hadjeres et al., 2017) models four-part chorales through a dependency network and enables constrained generation through pseudo-Gibbs sampling, while MuseGAN (Dong et al., 2018) explicitly models interactions among multiple instrument tracks for symbolic music generation and accompaniment. Music Transformer (Huang et al., 2019) introduces relative self-attention for modeling long-range dependencies in event-based symbolic music, and Pop Music Transformer (Huang & Yang, 2020) incorporates beat-based structure for expressive piano generation. MuseFormer (Yu et al., 2022) further exploits musical bar structure through fine- and coarse-grained attention to efficiently model long multi-instrument sequences.

Text-compatible symbolic representations have become particularly relevant to language-model-based music generation. ChatMusician (Yuan et al., 2024) incorporates ABC notation into a large

language model for joint music understanding and generation, demonstrating that structured symbolic music can be represented directly within a language-model interface. MuseCoco (Lu et al., 2023) instead uses intermediate musical attributes to support controllable symbolic music generation from textual descriptions. Beyond symbolic generation, MusicGen (Copet et al., 2023) demonstrates high-quality audio generation conditioned on textual descriptions or melodic input, illustrating the broader role of language-based conditioning in modern music generation systems. These approaches substantially improve musical modeling and controllability, but are primarily designed for settings in which generation may complete before the resulting music is consumed. AJAMT focuses on the complementary problem in which symbolic generation must remain causal and produce usable accompaniment under explicit playback deadlines.

2.2 LLM-Based and Multi-Agent Music Generation

Large language models have also been used as reasoning and coordination components for music generation. MusicAgent (Yu et al., 2023) uses an LLM as a central agent that interacts with specialized music-processing tools, demonstrating the potential of agentic planning for music understanding and generation. More directly related to symbolic composition, ComposerX (Deng et al., 2024) decomposes composition among specialized agents responsible for musical roles such as melody, harmony, accompaniment, and evaluation, using iterative communication and refinement to produce multi-instrument symbolic music. CoComposer (Xing et al., 2025) similarly explores collaborative multi-agent composition with LLMs.

These systems demonstrate that agent specialization can support complex musical planning and multi-instrument coordination. However, their workflows are primarily designed for offline composition, where multiple rounds of planning, critique, and revision can execute without an explicit playback deadline. Such sequential agent interactions are difficult to apply directly to live accompaniment because every additional LLM call contributes to the time before playable material becomes available. AJAMT retains agent specialization for harmonic planning and accompaniment generation, but organizes these interactions as a bounded two-stage pipeline and replaces iterative LLM-based self-reflection with lightweight symbolic evaluation. The resulting architecture is designed around the timing constraints of continuous playback rather than unconstrained offline composition.

2.3 Interactive and Real-Time Music Generation

Interactive music generation conditions machine-generated music on human-provided musical context. Several symbolic generation systems support conditional accompaniment. MuseGAN (Dong et al., 2018), for example, considers a human–AI cooperative setting in which generated tracks are conditioned on a human-composed track. Such conditional generation systems demonstrate the feasibility of generating coherent musical parts from human-provided context, but they are not designed around the explicit playback deadlines imposed by live performance. Real-time accompaniment therefore introduces an additional systems requirement: generated musical material must not only be coherent with the human input, but must also be available before its scheduled playback position.

RL-Duet (Jiang et al., 2020) addresses a more explicitly online setting by formulating human–machine accompaniment as a sequential decision-making problem and generating responses at short musical intervals. This formulation enables low-latency interaction, but targets a comparatively restricted generation setting and does not address rich multi-track, multi-instrument symbolic arrangement with language-model-based reasoning. In contrast, LLM-based generation provides a flexible interface for harmonic reasoning, style control, and multi-instrument generation, but introduces substantially higher and variable inference latency.

AJAMT addresses a different operating regime from both offline generative composition and note-level online accompaniment. Rather than requiring every model response to complete immediately, AJAMT predicts multi-instrument accompaniment ahead of its scheduled playback position and retains previously generated future material as fallback. This formulation separates generation deadline satisfaction from playback continuity: a late LLM response constitutes a generation deadline miss, but does not necessarily cause an interruption if valid accompaniment has already been prepared. AJAMT therefore frames real-time LLM accompaniment as deadline-constrained generation under variable inference latency, complementing prior work that primarily emphasizes generation quality, controllability, or low-latency note-level response.

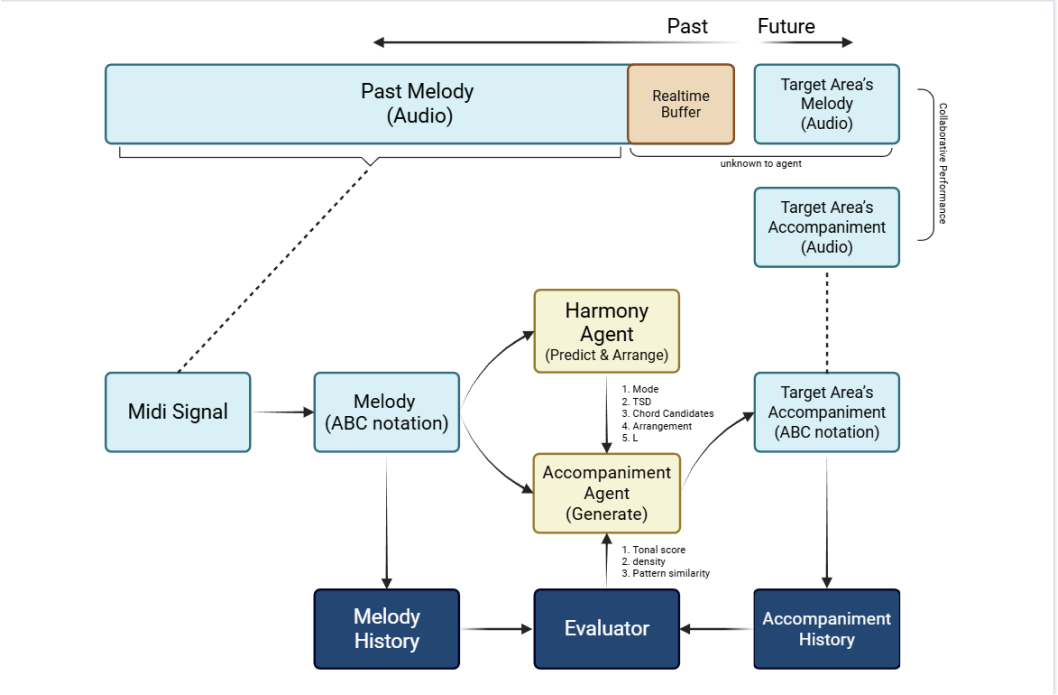


Figure 1: Overview of the AJAMT architecture. Incoming MIDI signals are converted into symbolic melody representations and incorporated into the observed musical context. Based only on past observations, the Harmony Agent performs harmonic planning and the Accompaniment Agent generates accompaniment for upcoming musical regions. A lightweight Symbolic Evaluator assesses the generated result using the melody and accompaniment histories. Predictive buffering allows accompaniment to be prepared ahead of playback, while previously generated future bars provide fallback material when a new request misses its deadline.

3 AJAMT

AJAMT is a real-time multi-agent framework for generating multi-instrument symbolic accompaniment from an ongoing human melody. The central idea is to decouple LLM inference from immediate playback by generating accompaniment ahead of its target musical position. As illustrated in Figure 1, AJAMT continuously observes incoming melody, maintains a bounded causal context through a Sliding Inference Window, and invokes a two-stage agentic pipeline to generate future accompaniment. Generated music is assessed by a lightweight Symbolic Evaluator and scheduled through a Real-Time Buffer. Each successful request additionally produces tentative future accompaniment that can be retained as fallback material. If a newly requested accompaniment is unavailable when its target playback position is reached, AJAMT uses a valid cached prediction instead of blocking playback. This design allows generation and playback to proceed asynchronously while preserving causal interaction with the human performer.

3.1 Real-Time Problem Formulation

We consider an online accompaniment setting in which a human performer provides a symbolic melody sequentially over musical bars. Let x_t denote the melody observed at bar t, and let $X_{\leq t} = \{x_1, \ldots, x_t\}$ denote all melody observations available up to that point. Given the observed melody, the system generates a multi-instrument accompaniment $Y_t = \{y_t^{(1)}, \ldots, y_t^{(M)}\}$, where M is the number of accompaniment instruments and $y_t^{(m)}$ contains the symbolic events generated for

instrument m. Generation is causal: accompaniment may depend only on melody that has already been observed rather than future human input.

Unlike offline conditional music generation, a musically valid output is useful in real-time accompaniment only if playable material is available before its scheduled playback position. Let b denote the tempo in beats per minute (BPM) and B the number of beats in a bar. The duration of one bar is

$$D_{\mathrm{bar}}(b) = \frac{60B}{b}. \tag{1}$$

For a $4/4$ piece, $B = 4$ and $D_{\mathrm{bar}}(b) = 240/b$ seconds. More generally, if a generation request is issued L bars before its target playback position, the available generation window is

$$D_{\mathrm{gen}}(b, L) = LD_{\mathrm{bar}}(b) = \frac{60BL}{b}. \tag{2}$$

Let τ_t denote the end-to-end latency of a generation request, measured from the inference trigger until the generated accompaniment has completed the generation and symbolic-processing pipeline. The request satisfies its generation deadline when

$$\tau_t \leq D_{\mathrm{gen}}(b, L). \tag{3}$$

A deadline miss, however, does not necessarily imply a playback interruption. AJAMT maintains previously generated future accompaniment that may be used when the newest request is late. We therefore distinguish whether a newly generated output meets its deadline from whether playable accompaniment remains available at the corresponding musical position.

Symbolic representation. AJAMT represents melody and accompaniment using ABC notation, a compact text-based symbolic music format that explicitly represents musical information such as meter, key, note duration, bar boundaries, and multiple voices. Its textual representation allows symbolic music to be incorporated directly into LLM prompts without introducing a separate music-specific tokenization pipeline. Prior work such as ChatMusician (Yuan et al., 2024) has demonstrated the compatibility of ABC notation with language-model-based music understanding and generation. AJAMT uses ABC both for the causal melody context supplied to the agents and for the generated multi-instrument accompaniment.

3.2 Sliding Inference Window

Providing the complete performance history to the LLM would cause the input context to grow continuously as a performance progresses. AJAMT therefore maintains a fixed-size Sliding Inference Window containing the most recent W bars of observed melody. At bar t, the context supplied to the generation pipeline is

$$C_t = \{x_{\max(1,t-W+1)}, \ldots, x_t\}. \tag{4}$$

After a new bar is observed, the window advances by incorporating the latest melody and discarding context outside the W-bar range. This keeps the amount of musical context supplied to the agents bounded regardless of performance length while retaining recent melodic and harmonic information relevant to upcoming accompaniment. Importantly, C_t contains only observations available at generation time, ensuring that AJAMT does not rely on future human melody. As illustrated in Figure 1, the inference window advances with the performance, providing a causal and bounded representation from which accompaniment for upcoming musical regions can be predicted.

3.3 Predictive Buffering and Fallback Scheduling

The core real-time mechanism of AJAMT separates accompaniment generation from immediate playback. Each inference step uses the currently observed melody window to prepare accompaniment for a future musical region. As the inference window advances with the incoming performance,

generation for subsequent bars overlaps temporally with playback of previously prepared accompaniment. The Real-Time Buffer therefore converts musical lookahead into inference time without exposing future human melody to the generation pipeline.

For a target bar $t+L$, AJAMT issues a generation request using the currently observed context C_t. If generation and symbolic processing finish before the target playback position, the newly generated accompaniment becomes available for playback. Increasing L increases the available generation window according to Equation 2, while the generated accompaniment remains conditioned only on melody observed at the time of the request.

Each successful request generates a K-bar accompaniment block. The first bar corresponds to the nearest target playback position, while the remaining $K-1$ bars are tentative predictions for subsequent positions. These future predictions are retained in a fallback cache and may later be replaced by newer predictions as additional melody observations become available. In the evaluated system, $K=4$, so each successful request prepares the target bar together with three tentative future bars.

A generation request may nevertheless miss its deadline because of variable LLM latency. When a target playback position is reached, the scheduler first checks whether the newly generated accompaniment is available. If so, it is used for playback. Otherwise, AJAMT retrieves the corresponding previously generated prediction from the fallback cache when one is available. Formally, if G_t denotes the newly generated accompaniment for playback position t and F_t denotes a valid cached fallback, the played accompaniment $\hat{Y}_t$ is selected as

$$\hat{Y}_t = \begin{cases} G_t, & \text{if } G_t \text{ is available before the deadline,} \\ F_t, & \text{otherwise, if a valid fallback is available.} \end{cases} \tag{5}$$

This distinction is central to AJAMT. In a synchronous generation pipeline, a missed generation deadline directly delays the availability of playable output. AJAMT instead introduces predictive generation and buffered scheduling between inference and playback, such that a generation deadline miss does not necessarily imply a playback interruption. Fallback material is nevertheless finite: persistent generation delays can exhaust valid cached predictions if new future accompaniment is not produced quickly enough. This motivates evaluating generation deadline satisfaction and playback continuity as distinct system properties.

3.4 Multi-Agent Accompaniment Generation

AJAMT uses a coordinated two-stage agentic architecture to generate multi-instrument accompaniment while avoiding the latency of iterative multi-agent refinement. Given the current sliding context C_t, the first stage is performed by the *Harmony Agent*, which analyzes the observed melody and produces harmonic information for the upcoming target region. This harmonic plan provides a shared musical foundation for subsequent accompaniment generation.

The second stage is performed by the *Accompaniment Agent*. Conditioned on the observed melody context, the harmonic plan, the requested style, and the specified instrumentation, the agent produces the coordinated multi-instrument accompaniment in ABC notation. Rather than generating each instrument through multiple sequential LLM calls, the Accompaniment Agent generates the instrumental parts within a shared generation stage, reducing the number of sequential inference operations required before accompaniment becomes available.

The Harmony Agent and Accompaniment Agent are each invoked once per generation request rather than repeatedly alternating between generation, critique, and revision. This differs from offline agentic composition workflows such as ComposerX (Deng et al., 2024), where multiple stages of agent interaction and refinement can be performed without an explicit playback deadline. AJAMT instead uses a bounded two-stage generation pipeline followed by lightweight symbolic evaluation, making the workflow more suitable for deadline-constrained interaction.

3.5 Symbolic Evaluation

Iterative LLM self-reflection introduces additional inference calls between generation and playback, which is undesirable under explicit real-time deadlines. AJAMT therefore employs a lightweight

Symbolic Evaluator that analyzes generated accompaniment directly in the symbolic domain without invoking another language model for critique. Rather than initiating an additional generate–criticize–revise loop, the evaluator computes structured musical feedback that can be supplied to subsequent generation requests.

The evaluator considers three complementary properties of the generated accompaniment: *tonal alignment*, which measures compatibility between melody and accompaniment pitch classes; *density ratio*, which characterizes the relative note density of the accompaniment with respect to the melody; and *pattern similarity*, which measures rhythmic similarity between the generated accompaniment and the preceding accompaniment context. Together, these features provide lightweight feedback about harmonic compatibility, textural density, and short-term rhythmic consistency without requiring an additional LLM inference step.

The resulting feedback is incorporated into subsequent Harmony Agent context rather than triggering iterative regeneration of the current bar. Consequently, symbolic evaluation remains outside the critical path of repeated LLM self-reflection while still allowing recent accompaniment quality to influence future harmonic planning. This preserves a fixed number of LLM calls per generation cycle and makes the agentic workflow compatible with predictive real-time scheduling.

Together, the Sliding Inference Window, predictive buffering, fallback scheduling, two-stage agent architecture, and Symbolic Evaluator form an asynchronous generation pipeline organized around musical playback deadlines. The sliding window bounds the causal input context, the Harmony and Accompaniment Agents provide harmonic reasoning and coordinated multi-instrument generation, and symbolic evaluation provides lightweight feedback without iterative LLM reflection. Most importantly, predictive generation and the Real-Time Buffer reduce the dependence of continuous accompaniment on the latency of any individual LLM request.

4 EXPERIMENTS

We evaluate AJAMT from two complementary perspectives. First, we examine whether the system can maintain continuous accompaniment under the latency constraints of real-time playback across multiple inference backends and playback tempos. Second, we compare the musical characteristics of AJAMT with ComposerX (Deng et al., 2024), an offline multi-agent symbolic music composition system. The first evaluation focuses on deadline satisfaction, fallback usage, and playback continuity, while the second examines audio-level aesthetic quality and symbolic musical structure.

4.1 EXPERIMENTAL SETUP

Melody selection and causal evaluation. We construct the real-time evaluation set from the ComposerX (Deng et al., 2024) evaluation prompt set and its corresponding symbolic melodies. We retain melodies in $4/4$ meter with at least 12 bars, ensuring a consistent four-beat bar duration across all timing experiments. Twenty melodies satisfying these criteria are selected for evaluation. For each melody, the first four bars are used as a warm-up period to establish musical context, while the following eight bars are used for measurement. This results in 160 bar-level observations per model–tempo condition, grouped into 20 melody-level clusters. Each model–tempo condition is executed independently rather than obtained by retrospectively applying different deadline thresholds to a shared set of latency measurements.

AJAMT is evaluated causally. When accompaniment for target bar t is generated, the system receives only melody observations preceding the target bar; melody events at and after t remain unavailable to the agents. This prevents future-information leakage and reproduces the information constraints of live accompaniment.

Sliding context and predictive generation. The Sliding Inference Window contains the four most recently observed bars ($W = 4$). Thus, when preparing accompaniment for target playback bar t, the most recently observed bar is $t-1$, and the LLM receives melody context from bars $[t-4, t-1]$. Under the evaluated prompts, this bounds the input context to approximately 250–350 tokens.

AJAMT schedules generation one bar before the target playback position. Let $B = 4$ denote the number of beats per bar and b the playback tempo in BPM. With a one-bar lookahead ($L = 1$), the available generation deadline is

$$D(b) = \frac{60BL}{b} = \frac{240}{b}. \tag{6}$$

Each successful Accompaniment Agent request produces a four-bar prediction ($K = 4$), covering the current target bar and the following three bars. The target bar is used for its scheduled playback, while the remaining predicted bars are stored in the fallback cache. If a subsequent request misses its playback deadline, a corresponding previously generated bar can therefore be retrieved from this cache when available.

Inference backends. We evaluate AJAMT using multiple hosted and local inference configurations. The hosted configurations include GPT-oss 20B, GPT-oss 120B, and Qwen-3.8 27B served through Groq, and GPT-4o-mini served through the OpenAI API. We additionally report a historical Groq Llama-3.3 70B configuration whose hosted endpoint was subsequently discontinued, as well as a local GPT-oss 20B configuration executed through Ollama on an NVIDIA RTX 5080 GPU.

Latency measurement. End-to-end generation latency is measured from the moment a generation request is triggered until the resulting accompaniment is ready after both LLM stages and symbolic parsing. For request i, we define

$$\tau_i = T_{\text{ready},i} - T_{\text{trigger},i}. \tag{7}$$

Thus, the reported latency includes Harmony Agent inference, Accompaniment Agent generation, and ABC parsing rather than model inference alone. For each backend, we report mean latency and P95 latency. We additionally report the mean-latency operating boundary

$$\text{BPM}_{\max} = \frac{60B}{\bar{\tau}} = \frac{240}{\bar{\tau}}, \tag{8}$$

where $\bar{\tau}$ is the measured mean end-to-end latency. This quantity should be interpreted as an average-latency operating indicator rather than a guaranteed maximum supported tempo, since individual requests may exhibit substantially higher latency.

For additional intuition, we express the mean latency as the equivalent number of beats elapsed at 120 BPM:

$$\text{LatencyBeats}_{120} = 2\bar{\tau}. \tag{9}$$

Playback metrics. Let D denote the generation deadline and let F_i indicate whether valid fallback material is available when request i misses its deadline. The generation deadline hit rate is

$$R_{\text{hit}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(\tau_i \le D), \tag{10}$$

and the corresponding deadline miss rate is

$$R_{\text{miss}} = 1 - R_{\text{hit}}. \tag{11}$$

We define the fallback rate as the fraction of playback positions for which the newest generation misses its deadline but valid cached accompaniment is available:

$$R_{\text{fb}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(\tau_i > D \wedge F_i = 1). \tag{12}$$

Table 1: Cross-backend real-time performance at 120 BPM. AVG and P95 denote mean and 95th-percentile end-to-end latency in seconds. $\mathrm{BPM}_{\max}$ is the mean-latency operating boundary, LB_{120} is the equivalent mean latency in beats at 120 BPM, HIT is the generation deadline hit rate, and CONT is the Playback Continuity Rate.

Model	AVG	P95	$\mathrm{BPM}_{\max}$	LB_{120}	HIT	CONT
GPT-oss 20B	1.21	1.77	198.2	2.42	98.8	100.0
GPT-oss 120B	3.71	6.28	64.7	7.41	11.2	86.2
Qwen-3.8 27B	4.43	10.41	54.2	8.86	13.1	90.6
GPT-4o-mini	7.33	9.94	32.7	14.66	0.0	51.9
Llama-3.3 70B†	1.03	1.15	232.0	2.06	100.0	100.0
Local GPT-oss 20B‡	15.00	18.50	16.0	30.00	0.0	0.0

Because newly generated on-time accompaniment and fallback accompaniment are mutually exclusive at a playback position, the Playback Continuity Rate is

$$R_{\mathrm{cont}} = R_{\mathrm{hit}} + R_{\mathrm{fb}}. \tag{13}$$

This metric directly measures the fraction of target playback positions for which AJAMT has playable accompaniment available, regardless of whether that material comes from the newest request or the predictive fallback cache.

4.2 Cross-Backend Real-Time Performance

We first compare inference backends at 120 BPM. Under $4/4$ meter and one-bar lookahead, the corresponding generation deadline is $D = 2.0$ seconds. Table 1 reports the measured end-to-end latency and real-time performance of each backend.

†Historical configuration; the hosted endpoint used during the original evaluation was subsequently discontinued. ‡Local inference configuration included for reference.

Among the currently available backends, GPT-oss 20B provides the strongest real-time performance. Its mean end-to-end latency is 1.21 seconds, with a P95 latency of 1.77 seconds. The corresponding mean-latency operating boundary is 198.2 BPM. At 120 BPM, 98.8% of requests complete within the 2.0-second deadline. The remaining late requests are covered by cached accompaniment, resulting in 100% playback continuity.

The slower backends illustrate why raw deadline satisfaction alone is insufficient for characterizing an interactive generation system. GPT-oss 120B completes only 11.2% of requests within the 120 BPM deadline, yet AJAMT maintains 86.2% playback continuity. Similarly, Qwen-3.8 27B achieves a 13.1% hit rate while maintaining 90.6% continuity. GPT-4o-mini misses every measured 120 BPM deadline, but previously generated fallback material still preserves playback for 51.9% of target positions. These results demonstrate that generation deadline satisfaction and playback continuity capture different aspects of real-time system behavior.

4.3 Multi-Tempo Stress Test

We next evaluate how AJAMT behaves as the playback deadline becomes increasingly strict. Each backend is independently evaluated at 60, 72, 120, 180, and 240 BPM, with 160 bar-level observations collected for every model–tempo condition. The corresponding one-bar deadlines are 4.00, 3.33, 2.00, 1.33, and 1.00 seconds, respectively. Table 2 reports deadline miss rate, fallback rate, and Playback Continuity Rate.

The stress test provides direct evidence for the distinction between generation deadline misses and playback interruptions. GPT-oss 20B maintains 100% playback continuity from 60 through 180 BPM. At 180 BPM, 24.4% of generation requests exceed the 1.33-second deadline; however, fallback accompaniment is available for all of these late requests, yielding a 24.4% fallback rate and preserving 100% playback continuity.

Table 2: Multi-tempo real-time stress test. Each cell reports generation deadline miss rate / fallback rate / Playback Continuity Rate (%). Each model–tempo condition contains 160 bar-level observations.

Model	60 BPM	72 BPM	120 BPM	180 BPM	240 BPM
GPT-oss 20B	0.0/0.0/100.0	0.0/0.0/100.0	1.2/1.2/100.0	24.4/24.4/100.0	76.9/68.1/91.2
GPT-oss 120B	33.1/33.1/100.0	53.1/45.6/92.5	88.8/75.0/86.2	100.0/70.0/70.0	100.0/52.5/52.5
Qwen-3.8 27B	47.5/44.4/96.9	60.6/55.0/94.4	86.9/77.5/90.6	90.0/65.0/75.0	100.0/51.2/51.2
GPT-4o-mini	100.0/82.5/82.5	100.0/80.0/80.0	100.0/51.9/51.9	100.0/4.4/4.4	100.0/0.0/0.0

Table 3: Audio-level aesthetic comparison between AJAMT and ComposerX over 127 paired outputs. Higher values indicate higher evaluator scores.

Metric	AJAMT	ComposerX	Δ
CE	6.279	5.860	+0.419
CU	7.856	7.567	+0.289
PC	3.320	2.415	+0.905
PQ	8.027	7.788	+0.239

At 240 BPM, the generation deadline decreases to only 1.0 second. GPT-oss 20B consequently misses 76.9% of requests. Cached accompaniment covers 68.1% of all playback positions, allowing AJAMT to maintain 91.2% continuity despite the high deadline miss rate. This result also illustrates the limit of predictive fallback: when generation misses become sufficiently persistent, the cache can no longer cover every future playback position.

The same behavior is more pronounced for slower backends. At 120 BPM, GPT-oss 120B misses 88.8% of generation deadlines but maintains 86.2% playback continuity, while Qwen-3.8 27B misses 86.9% of deadlines but maintains 90.6% continuity. Fallback scheduling nevertheless cannot compensate indefinitely for insufficient generation throughput. GPT-4o-mini continuity decreases from 82.5% at 60 BPM to 51.9% at 120 BPM, 4.4% at 180 BPM, and 0% at 240 BPM as valid future accompaniment can no longer be replenished quickly enough.

4.4 Musical Quality Comparison

Real-time operation is useful only if the resulting accompaniment remains musically meaningful. We therefore compare AJAMT with ComposerX (Deng et al., 2024) under a controlled melody condition. The official ComposerX evaluation prompt set contains 151 natural-language prompts spanning diverse musical styles and instrumentation requirements. ComposerX produces 127 valid symbolic outputs from these prompts. For each valid output, we extract the corresponding monophonic melody and provide the same melody to AJAMT under causal evaluation, so that both systems are compared using a shared melodic source.

Baseline and rendering. We use ComposerX (Deng et al., 2024) as the primary musical-quality baseline because it represents an offline multi-agent composition workflow with iterative refinement. Musical-quality comparisons are conducted on the 127 valid ComposerX outputs and their corresponding AJAMT accompaniments. Symbolic scores from both systems are converted to MIDI and rendered to WAV using FluidSynth with the same GeneralUser GS SoundFont, controlling the rendering pipeline across systems.

Audio-level aesthetic metrics. We evaluate the rendered audio using Meta Audiobox Aesthetics (Tjandra et al., 2025). We report Content Enjoyment (CE), Content Usefulness (CU), Production Complexity (PC), and Production Quality (PQ). Table 3 reports the aggregate mean scores across the 127 paired outputs.

AJAMT obtains higher aggregate mean scores than ComposerX on all four evaluated Audiobox Aesthetics dimensions. The largest difference occurs in Production Complexity, where AJAMT scores

Table 4: Symbolic-structure comparison between AJAMT and ComposerX over 127 paired outputs.

Metric	AJAMT	ComposerX
Pitch Range	56.17	31.01
Polyphony	2.53	2.34

3.320 compared with 2.415 for ComposerX. Content Enjoyment, Content Usefulness, and Production Quality show smaller positive differences. Because these results report aggregate means without statistical significance testing, we interpret them as evidence that AJAMT remains competitive with the offline baseline under the controlled evaluation protocol rather than as evidence of statistically significant superiority.

Symbolic structure. We additionally compare pitch range and polyphony directly from the symbolic outputs. Pitch range measures the span between the lowest and highest generated pitches in semitones, while polyphony measures the average degree of simultaneous note activity. Results are shown in Table 4.

AJAMT exhibits a wider pitch range and slightly higher polyphony than ComposerX under the controlled melody condition. These measurements indicate that AJAMT's real-time operation is not achieved simply by restricting generation to a narrower pitch space or uniformly sparse accompaniment. However, the comparison should be interpreted cautiously because ComposerX performs offline full-song composition whereas AJAMT performs causal accompaniment generation. The results therefore characterize differences under a shared-melody protocol rather than establishing universal superiority over the offline system.

4.5 Discussion

The experiments reveal two complementary aspects of real-time LLM accompaniment. First, inference backend selection has a substantial effect on the raw timing margin available to the system. Among the currently evaluated configurations, GPT-oss 20B provides the strongest latency performance, with a mean end-to-end latency of 1.21 seconds and a P95 latency of 1.77 seconds. The previously evaluated Llama-3.3 70B endpoint achieved a lower mean latency of 1.03 seconds, but is retained only as a historical measurement because the hosted endpoint is no longer available.

Second, and more importantly, the multi-tempo stress test shows that raw inference latency alone does not determine whether accompaniment remains continuous. At 180 BPM, GPT-oss 20B misses 24.4% of generation deadlines while AJAMT nevertheless maintains 100% playback continuity. Similar behavior occurs with slower models, where previously generated fallback material preserves playback despite high deadline miss rates. At sufficiently high tempos or with sufficiently slow inference, however, persistent misses prevent the predictive cache from being replenished, causing continuity to decrease. The results therefore empirically support AJAMT's central distinction between generation deadline satisfaction and playback reliability.

Finally, the musical-quality evaluation suggests that imposing real-time constraints does not necessarily require reducing the generated accompaniment to trivially sparse or structurally narrow outputs. Under the controlled shared-melody protocol, AJAMT obtains competitive audio-aesthetic scores and exhibits a wider pitch range and slightly higher polyphony than ComposerX. At the same time, the two systems operate under different generation regimes, and the comparison should be interpreted as a controlled reference rather than a direct claim of overall superiority.

5 Conclusion

We presented AJAMT, a real-time multi-agent framework for generating multi-instrument symbolic accompaniment during human–AI jam sessions. AJAMT addresses the latency constraints of LLM-based interactive music generation through a Sliding Inference Window, predictive multi-bar generation, a Real-Time Buffer with fallback scheduling, and a bounded two-stage agentic architecture with lightweight symbolic evaluation. Together, these components decouple LLM inference from

immediate playback and allow accompaniment generation to proceed asynchronously while preserving causal interaction with the human performer.

Our experiments demonstrate that real-time performance cannot be characterized by mean inference latency alone. Among the currently evaluated backends, the fastest configuration achieves a mean end-to-end latency of 1.21 seconds and 98.8% on-time generation at 120 BPM, while maintaining 100% playback continuity. More importantly, at 180 BPM, AJAMT maintains 100% playback continuity despite a 24.4% generation deadline miss rate through buffered fallback material. At 240 BPM, continuity remains at 91.2% despite a 76.9% deadline miss rate, while experiments with slower backends further show that continuity eventually decreases when generation throughput is insufficient to replenish the predictive cache. These results empirically demonstrate that a generation deadline miss does not necessarily imply a playback interruption.

Under a controlled shared-melody evaluation, AJAMT also remains competitive with the offline ComposerX baseline across the evaluated audio-aesthetic metrics while exhibiting a wider pitch range and slightly higher polyphony. These results suggest that deadline-constrained operation need not be achieved simply by reducing accompaniment to uniformly sparse or structurally narrow outputs. Overall, AJAMT provides a practical framework for integrating comparatively high-latency LLM-based musical reasoning into continuous interactive accompaniment by treating generation deadlines and playback continuity as distinct system properties.

REFERENCES

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Qixin Deng, Qikai Yang, Ruibin Yuan, Yipeng Huang, Yi Wang, Xubo Liu, Zeyue Tian, Jiahao Pan, Ge Zhang, Hanfeng Lin, Yizhi Li, Yinghao Ma, Jie Fu, Chenghua Lin, Emmanouil Benetos, Wenwu Wang, Guangyu Xia, Wei Xue, and Yike Guo. ComposerX: Multi-agent symbolic music composition with LLMs. *arXiv preprint arXiv:2404.18081*, 2024.

Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: A steerable model for bach chorales generation. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1362–1371, 2017.

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2019.

Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 1180–1188, 2020. doi: 10.1145/3394171.3413671.

Nan Jiang, Sheng Jin, Zhiyao Duan, and Changshui Zhang. RL-Duet: Online music accompaniment generation using deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 710–718, 2020.

Peiling Lu, Xin Xu, Chenfei Kang, Botao Yu, Chengyi Xing, Xu Tan, and Jiang Bian. MuseCoco: Generating symbolic music from text. *arXiv preprint arXiv:2306.00110*, 2023.

Andros Tjandra, Yi-Chiao Wu, Baishan Guo, John Hoffman, Brian Ellis, Apoorv Vyas, Bowen Shi, Sanyuan Chen, Matt Le, Nick Zacharov, Carleigh Wood, Ann Lee, and Wei-Ning Hsu. Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound. *arXiv preprint arXiv:2502.05139*, 2025.

Peiyu Xing, Aske Plaat, and Niki van Stein. CoComposer: LLM multi-agent collaborative music composition. *arXiv preprint arXiv:2509.00132*, 2025.

Botao Yu, Peiling Lu, Rui Wang, Wei Hu, Xu Tan, Wei Ye, Shikun Zhang, Tao Qin, and Tie-Yan Liu. Museformer: Transformer with fine- and coarse-grained attention for music generation. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Dingyao Yu, Kaitao Song, Peiling Lu, Tianyu He, Xu Tan, Wei Ye, Shikun Zhang, and Jiang Bian. MusicAgent: An AI agent for music understanding and generation with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 246–255. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-demo.21.

Ruibin Yuan, Hanfeng Lin, Yi Wang, Zeyue Tian, Shangda Wu, Tianhao Shen, Ge Zhang, Yuhang Wu, Cong Liu, Ziya Zhou, Liumeng Xue, Ziyang Ma, Qin Liu, Tianyu Zheng, Yizhi Li, Yinghao Ma, Yiming Liang, Xiaowei Chi, Ruibo Liu, Zili Wang, Chenghua Lin, Qifeng Liu, Tao Jiang, Wenhao Huang, Wenhu Chen, Jie Fu, Emmanouil Benetos, Gus Xia, Roger Dannenberg, Wei Xue, Shiyin Kang, and Yike Guo. ChatMusician: Understanding and generating music intrinsically with LLM. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6252–6271, 2024. doi: 10.18653/v1/2024.findings-acl.373.