dates; however, practical deployment requires $t_f < 0$ to offset latency, which drastically degrades quality. Consistent with prior work, training on well-composed datasets is insufficient, since such datasets rarely contain errors, corrective maneuvers, or co-adaptive behavior. These results motivate developing training objectives that explicitly encode anticipation and coordination, laying a foundation for future research on real-time audio accompaniment models.

7. REFERENCES

[1] Peter Keller, "Joint action in music performance," in *Enacting Intersubjectivity: A Cognitive and Social Perspective to the Study of Interactions*. 2008.

[2] William J Wrigley and Stephen B Emmerson, "The experience of the flow state in live music performance," *Psychology of Music*, 2013.

[3] Wiebke Trost, Caitlyn Trevor, Natalia Fernandez, et al., "Live music stimulates the affective brain and emotionally entrains listeners in real time," *Proceedings of the National Academy of Sciences*, 2024.

[4] Julian D Parker, Janne Spijkervet, Katerina Kosta, et al., "Stemgen: A music generation model that listens," in *ICASSP*. IEEE, 2024.

[5] Chris Donahue, Antoine Caillon, Adam Roberts, et al., "Singsong: Generating musical accompaniments from singing," *ArXiv*, 2023.

[6] Simon Rouard, Robin San Roman, Yossi Adi, et al., "Musicgen-stem: Multi-stem music generation and edition through autoregressive modeling," in *ICASSP*, 2025.

[7] Yusong Wu, Tim Cooijmans, Kyle Kastner, et al., "Adaptive accompaniment with realchords," in *ICML*, 2024.

[8] Christodoulos Benetatos, Joseph VanderStel, and Zhiyao Duan, "Bachduet: A deep learning system for human-machine counterpoint improvisation," in *NIME*, 2020.

[9] Zihao Wang, Kejun Zhang, Yuxing Wang, et al., "Songdriver: Real-time music accompaniment generation without logical latency nor exposure bias," in *ACM MM*, 2022.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al., "Attention is all you need," *NeurIPS*, 2017.

[11] Ethan Manilow, Gordon Wichern, Prem Seetharaman, et al., "Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity," in *WASPAA*. IEEE, 2019.

[12] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, et al., "High-fidelity audio compression with improved rvqgan," *NeurIPS*, 2023.

[13] Roger B Dannenberg, "An on-line algorithm for real-time accompaniment," in *ICMC*, 1984.

[14] Christopher Raphael, "Music plus one and machine learning.," in *ICML*, 2010.

[15] Arshia Cont, "Antescofo: Anticipatory synchronization and control of interactive parameters in computer music.," in *ICMC*, 2008.

[16] George E Lewis, "Too many notes: Computers, complexity, and culture in voyager," in *New Media*. Routledge, 2003.

[17] Gérard Assayag, Georges Bloch, Marc Chemillier, et al., "Omax brothers: a dynamic yopology of agents for improvization learning," in *ACM workshop on Audio and music computing multimedia*, 2006.

[18] Jérôme Nika and Marc Chemillier, "Improtek: integrating harmonic controls into improvisation in the filiation of omax," in *ICMC*, 2012.

[19] Jérôme Nika, Marc Chemillier, and Gérard Assayag, "Improtek: introducing scenarios into human-computer music improvisation," *Computers in Entertainment (CIE)*, 2017.

[20] Nan Jiang, Sheng Jin, Zhiyao Duan, et al., "RL-duet: Online music accompaniment generation using deep reinforcement learning," in *AAAI*, 2020.

[21] Alexander Scarlatos, Yusong Wu, Ian Simon, et al., "Realjam: Real-time human-ai music jamming with reinforcement learning-tuned transformers," in *CHI EA*, 2025.

[22] Lyria Team, Antoine Caillon, Brian McWilliams, et al., "Live music models," *ArXiv*, 2025.

[23] O. J. M. Smith, "A controller to overcome dead time," *ISA Journal*, 1959.

[24] Maximilian Schwenzer, Muzaffer Ay, Thomas Bergs, et al., "Review on model predictive control: an engineering perspective," *The International Journal of Advanced Manufacturing Technology*, 2021.

[25] Kevin Black, Manuel Y Galliker, and Sergey Levine, "Real-time execution of action chunking flow policies," *ArXiv*, 2025.

[26] Ke Chen, Yusong Wu, Haohe Liu, et al., "Musicldm: Enhancing novelty in text-to-music generation using beat-synchronous mixup strategies," in *ICASSP*. IEEE, 2024.

[27] Hugo Flores Garcia, Prem Seetharaman, Rithesh Kumar, et al., "Vampnet: Music generation via masked acoustic token modeling," *arXiv preprint arXiv:2307.04686*, 2023.

[28] John Thickstun, David Leo Wright Hall, Chris Donahue, and Percy Liang, "Anticipatory music transformer," *Transactions on Machine Learning Research*, 2024.

[29] MIDI Manufacturers Association, "Complete MIDI 1.0 detailed specification," http://www.midi.org/techspecs/gm.php, 1999, Updated 2008.

[30] Shih-Lun Wu, Aakash Lahoti, Arjun D Desai, et al., "Towards codec-LM co-design for neural codec language models," in *NAACL SRW*, 2025.

[31] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al., "The llama 3 herd of models," *ArXiv*, 2024.

[32] Jade Copet, Felix Kreuk, Itai Gat, et al., "Simple and controllable music generation," *NeurIPS*, 2024.

[33] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[34] Priya Goyal, Piotr Dollár, Ross Girshick, et al., "Accurate, large minibatch sgd: Training imagenet in 1 hour," *ArXiv*, 2017.

[35] Ilya Loshchilov and Frank Hutter, "Sgdr: Stochastic gradient descent with warm restarts," in *ICLR*, 2017.

[36] Ruben Ciranni, Giorgio Mariani, Michele Mancusi, et al., "Cocola: Coherence-oriented contrastive learning of musical audio representations," in *ICASSP*. IEEE, 2025.

[37] Francesco Foscarin, Jan Schlüter, and Gerhard Widmer, "Beat this! accurate beat tracking without dbn postprocessing," in *ISMIR*, 2024.

[38] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, et al., "madmom: a new Python Audio and Music Signal Processing Library," in *ACM MM*, 2016.

[39] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, et al., "Fr\'echet audio distance: A metric for evaluating music enhancement algorithms," *ArXiv*, 2018.

[40] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu, "On layer normalization in the transformer architecture," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, arXiv:2002.04745.

[41] Biao Zhang and Rico Sennrich, "Root mean square layer normalization," *arXiv preprint arXiv:1910.07467*, 2019.

[42] Zhen Qin, Dong Li, Weigao Sun, Weixuan Sun, Xuyang Shen, Xiaodong Han, Yunshen Wei, Baohong Lv, Xiao Luo, Yu Qiao, and Yiran Zhong, "Transnormerllm: A faster and better large language model with improved transnormer," *arXiv preprint arXiv:2307.14995*, 2023.

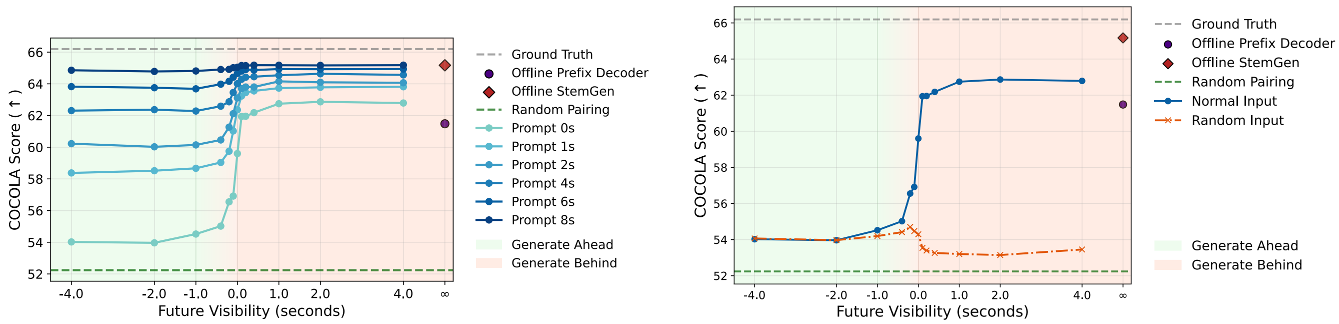
[43] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu, "Roformer: Enhanced transformer with rotary position embedding," *arXiv preprint arXiv:2104.09864*, 2021.

[44] Alex Henry, Prudhvi Raj Dachapally, Shubham Pawar, and Yuxuan Chen, "Query-key normalization for transformers," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, arXiv:2010.04245.

[45] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, Rodolphe Jenatton, Lucas Beyer, Michael Tschannen, Anurag Arnab, Xiao Wang, Carlos Riquelme, Matthias Minderer, Joan Puigcerver, Utku Evci, Manoj Kumar, Sjoerd van Steenkiste, Gamaleldin F. Elsayed, Aravindh Mahendran, Fisher Yu, Avital Oliver, Fantine Huot, Jasmijn Bastings, Mark Patrick Collier, Alexey Gritsenko, Vighnesh Birodkar, Cristina Vasconcelos, Yi Tay, Thomas Mensink, Alexander Kolesnikov, Filip Paveti'c, Dustin Tran, Thomas Kipf, Mario Luči'c, Xiaohua Zhai, Daniel Keysers, Jeremiah Harmsen, and Neil Houlsby, "Scaling vision transformers to 22 billion parameters," *arXiv preprint arXiv:2302.05442*, 2023.

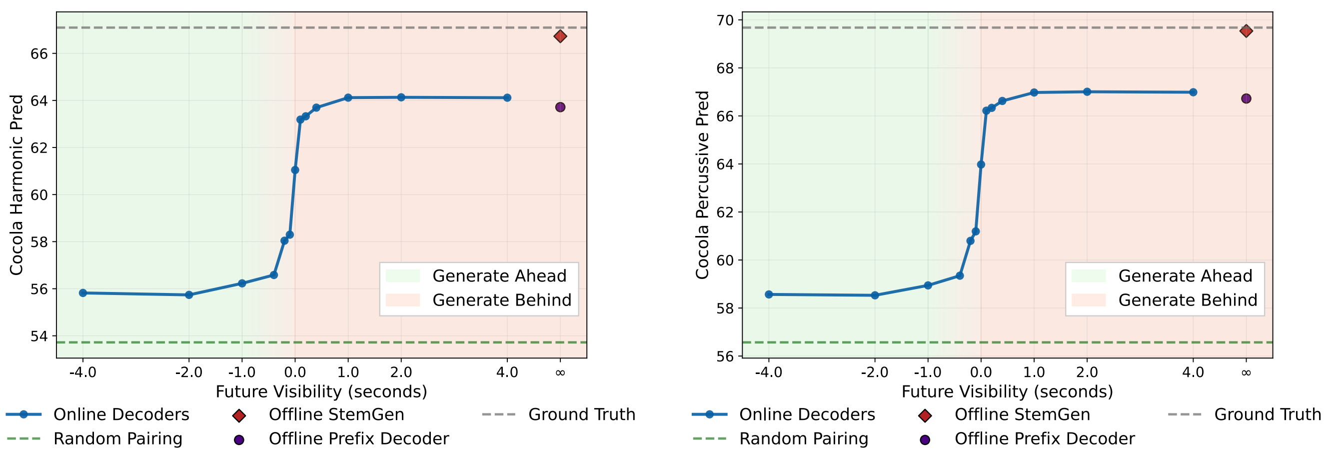
[46] Noam Shazeer, "Glu variants improve transformer," *arXiv preprint arXiv:2002.05202*, 2020.

[47] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai, "Gqa: Training generalized multi-query transformer models from multi-head checkpoints," in *Proceedings of EMNLP 2023*, 2023, arXiv:2305.13245.



(a) COCOLA Score Across Prompt Length (b) COCOLA Score Compared with Random Input

Fig. 5. Accompaniment performance for streaming models with $k = 1$ evaluated across different future visibility t_f. Left: performance under different prompt length. Right: performance when conditioning on the paired input compared with conditioning on a random input.



(a) COCOLA Harmonic (b) COCOLA Percussive

Fig. 6. COCOLA harmonic score and COCOLA percussive score of generated accompaniment for streaming models with $k = 1$ across different t_f.

A. APPENDIX

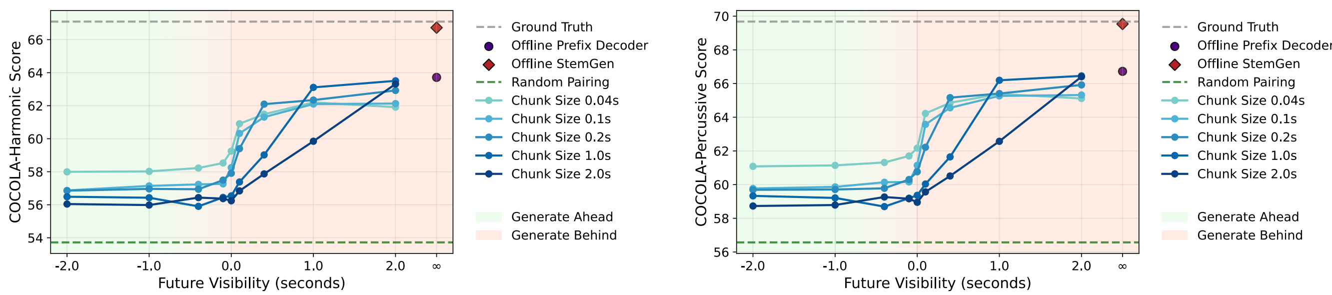
A.1. Acknowledgment

We would like to thank Ke Chen for the discussion on this project.

A.2. Additional Results

We further investigate how each model configuration attends to the input and output. We run streamed generation while pairing each target stem with a randomly chosen input from the test set, and we compare COCOLA against the true paired input. As shown in Fig. 5b, the scores with true and random inputs are almost indistinguishable when $t_f \leq 2$, which indicates that the decoder relies mainly on its own history and the instrument token under low or negative visibility. For small positive t_f, the gap increases, which implies that the model begins to exploit the input stream for both harmonic and rhythmic cues. This is consistent with the main results where coherence and beat alignment improve as lookahead increases. We warm start decoding by providing a ground truth prefix of both input and output with duration L seconds, then start streaming. Fig. 5a shows that gains from prompting are largest when $t_f \leq 0$, and decrease as t_f grows. This suggests that, in low future-visibility regimes, a short history reduces exposure bias and stabilizes local decisions, whereas with more lookahead the model already observes sufficient recent context and benefits less from a longer prefix.

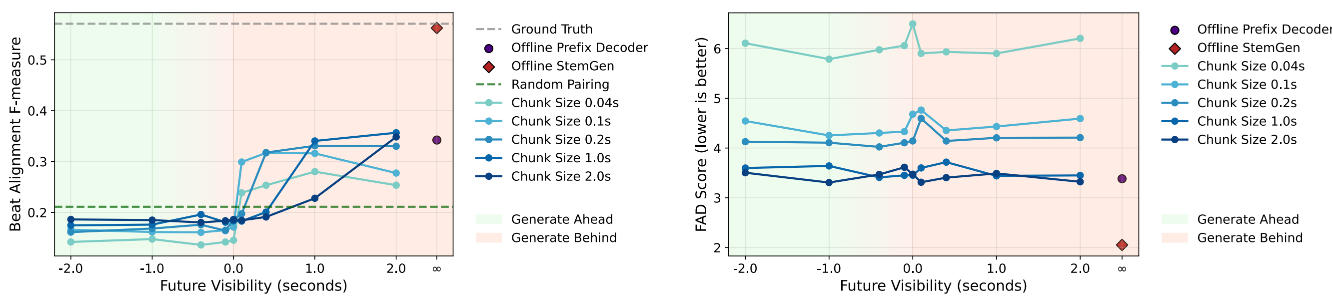
We include the COCOLA harmonic score and COCOLA percussive score of generated accompaniment for streaming models with $k = 1$ across different t_f in Fig. 6. For streaming models with combinations of $k > 1$ and t_f, we include COCOLA harmonic score and COCOLA percussive score in Fig. 7, and beat alignment and FAD score in Fig. 8.



(a) COCOLA Harmonic

(b) COCOLA Percussive

Fig. 7. COCOLA harmonic score and COCOLA percussive score of generated accompaniment for streaming models with combinations of $k > 1$ and t_f.



(a) Beat Alignment

(b) COCOLA Percussive

Fig. 8. Beat alignment and FAD score of generated accompaniment for streaming models with combinations of $k > 1$ and t_f.

A.3. Dataset Details

When selecting input-output track pairs and a start time for each $10\,\mathrm{s}$ window, we filter by silence and instrument class. We compute A-weighted short-time RMS at $50\,\mathrm{Hz}$ and label frames with level below $-60\,\mathrm{dB}$ as silent. We then reject a window if the input mixture and the target stem do not have at least 50% overlap of non-silent frames. When choosing the target stem, we exclude vocal classes.

A.4. Transformer Backbone

We use a transformer backbone similar to Llama 3. Following current large-model practice, the network uses pre layer normalization throughout [40], with Root Mean Square Normalization as the normalization operator [41]; we adopt the simplified RMSNorm variant reported to work well at large scale [42]. Positional information is injected by rotary position embedding in self-attention [43]. Inside attention, we apply query-key normalization by normalizing queries and keys along the head dimension before the similarity computation [44], together with a dimension-dependent scaling factor on the normalized scores as recommended by recent scaling results [45]. The feed-forward sublayers use the gated SwiGLU activation [46]. To improve decoding efficiency while preserving quality, key-value projections are shared across groups of query heads, i.e., grouped-query attention [47]. The decoder also supports cross-attention for conditioning on external context.

A.5. Model Implementation Details

For all prefix-decoder models, we do not use a bidirectional mask on the prefix. Instead, we apply a single causal mask over the entire sequence so that every position attends only to past positions.

For the StemGen masked language model, we found a VampNet-style [27] confidence ranking to outperform the original StemGen ranking. At each sampling iteration and for each RVQ level, we compute a confidence score for token $\hat{y}_t$ as

$$\mathrm{conf}(\hat{y}_t) \;=\; \log p(\hat{y}_t) \;+\; temp \cdot g_t, \tag{4}$$

where $p(\hat{y}_t)$ is the model probability of $\hat{y}_t$, $g_t \sim \mathrm{Gumbel}(0,1)$ is i.i.d., and $temp$ is linearly annealed to 0 over the sampling iterations.

A.6. Streaming Prefix Decoder Details

In the streaming setting with chunk size $k > 1$, we train a prefix decoder. For each minibatch, we sample a prefix length ℓ uniformly from $\{0,\, k,\, 2k,\, \ldots,\, T-k\}$. Given ℓ, we construct each example by aligning inputs and outputs up to step ℓ, then require the model to predict the next k steps $(\ell+1, \ldots, \ell+k)$ under a causal attention mask. The loss is computed only on these k target steps, and gradients are applied exclusively to those positions, while earlier tokens serve as context without direct supervision. This variable-prefix sampling exposes the model to a range of prefix boundaries and supports chunked next-k prediction during streaming inference.

A.7. Model Sampling Details

For all decoder-only Transformers, we sample with softmax temperature 1.0 and top-$k = 200$. For the StemGen model, we use per-level maximum noise temperatures $[8.0,\ 8.0,\ 4.0,\ 4.0]$ for RVQ levels $\ell = 1, \ldots, 4$, and $[128,\ 64,\ 32,\ 32]$ sampling steps per level. The StemGen model is trained with input dropout probability 20% (the input embedding is zeroed when dropped). At sampling time we use classifier-free guidance with scale 2.0.

For online streaming models, the total input it ever conditioned on is $t_f + T$. That is, if $t_f > 0$, the model see extra input stream, and vice versa.

A.8. Listening Study Details

We ran a listening study in which 24 participants evaluated the models in this study as well as ground truth and random pairing examples. Participants blindly evaluated the models by indicating their preference between pairs of accompaniments for a given input. A Kruskal-Wallis H test and confirmed that there are statistically significant pairs among the permutations. We evaluate significance with a post-hoc analysis using the Wilcoxon signed-rank test with Bonferroni correction (with p¡0.05/21 as there are 7 models evaluated).

To ease the participant's cognitive load, we select samples with six or fewer input tracks for inclusion in the listening test.

	Ground Truth	Offline StemGen	Offline Prefix Decoder	$t_f = 1$	$t_f = 0$	$t_f = -1$	Random Pairing
Ground Truth	N/A	!	*	*	*	*	*
Offline StemGen	!	N/A	!	!	!	*	*
Offline Prefix Decoder	*	!	N/A	!	!	*	!
$t_f = 1$	*	!	!	N/A	!	*	!
$t_f = 0$	*	!	!	!	N/A	*	!
$t_f = -1$	*	*	*	*	*	N/A	!
Random Pairing	*	*	!	!	!	!	N/A

Table 1. Pairwise statistical significance results for listening study. * indicates significant difference ($p < 0.05/21$), ! indicates non-significant ($p > 0.05/21$).

A.9. Audio Mixing

For all objective evaluations and the listening study, we use a fixed loudness pipeline. First, we loudness-normalize the predicted track and the target track to $-18\,\mathrm{dB}$. When forming the mix, all tracks in the input mixture are summed at equal loudness, and the predicted (or target) track is mixed $+5\,\mathrm{dB}$ relative to each input track. Finally, we normalize the resulting mixture to $-18\,\mathrm{dB}$.