Loops That Listen: A Voice-Controlled Dynamic Drum Looper with AI Variation

Paolo Sandejas[1]
paolosandejas@students.calarts.edu

Zhaohan Cheng[1]
zcheng@calarts.edu

Ajay Kapur[1,2]
akapur@calarts.edu

[1]California Institute of the Arts, Valencia, CA, USA [2]NYU Shanghai, Shanghai, China

Abstract

Live looping constrains musical dynamics. The same pattern cycles indefinitely, while a performance demands response and variation. CHULOOPA is a real-time drum looper that resolves this tension directly. The performer beatboxes patterns, and the system varies them in response to the live musical energy. A dBFS-mapped RMS measure, "spice", drives selection from a pre-generated bank of five variations. These variants are generated in parallel by a grid-based decoder-only transformer and sorted by deviation score into a predictable creativity spectrum for the performer. At each loop boundary, the variant matching the current spice level is selected without interrupting playback. CHULOOPA demonstrates that audio-responsive AI variation can give a loop a controllable range of complexity while keeping it anchored to the performer's original pattern.

Keywords: dynamic looping, audio-driven variation, spice detection, grid-quantized variation, co-creative AI, live performance

1 INTRODUCTION

The strength of a loop is also its main constraint: once recorded, it never changes. For drum loops in live performance, this constraint is especially pronounced. Where a live drummer would subtly add fills, vary dynamics, or shift the groove to match the moment, a drum loop plays back identically every cycle. While not always undesired, with hardware loop pedals like the BOSS RC-1 and software tools such as Ableton Live's Session View excelling at stacking static loops to provide rhythmic stability to a performance, an unchanging loop lacks the organic variation that makes live drumming compelling. With that limitation in mind, how can dynamic musicality be brought into drum loops without sacrificing their stability?

CHULOOPA (ChucK-based (Wang et al., 2015) Loop Operator for Performance Audio) is a voice-controlled dynamic drum looper that addresses this. Unlike prior intelligent loopers (Marchini et al., 2017; Burloiu, 2020) that work from guitar audio or pre-loaded MIDI drum scores, CHULOOPA uses voice input as its sole creation interface: a performer beatboxes a drum pattern, the system transcribes it in real-time via a user-trained KNN classifier, loops it, and immediately generates a bank of 5 AI variations that a post-generation deviation-sort arranges into a predictable conservative-to-adventurous spectrum. Simultaneously, the variant controller reads the live audio energy, *spice*, and at each loop boundary performs a weighted probabilistic selection from the variation bank to match the current musical moment, without manual intervention.

Voice is the only creation interface, chosen over the alternatives. Finger drumming on pads or tapping a surface both need a controller and occupy the hands. Loading a pre-made MIDI groove sacrifices immediacy. Beatboxing keeps the hands free for a second instrument and lets the performer sketch a rhythmic feel in the moment. Its one cost is transcription: vocal percussion is timbrally idiosyncratic from player to player, so CHULOOPA trains a classifier on the performer's own voice rather than a generic drum dataset.

2 RELATED WORK

CHULOOPA builds on three prior bodies of work: intelligent loopers, generative models for drum pattern variation, and personalized beatbox classifiers for real-time percussion transcription.

Intelligent Loopers. From Pachet's Continuator (Pachet, 2003) and OMax (Assayag et al., 2006), which pioneered real-time style learning and pattern recombination from live input, to the Reflexive Looper (Marchini et al., 2017), which established looping agents that adapt to a musician using constrained optimization to schedule material across upcoming beats, intelligent loopers have pursued the same goal: making loops responsive to a live performer. Rolypoly~ (Burloiu, 2020) takes a complementary approach, using RNNs pretrained on drum groove data to adapt the microtiming and velocity of a loaded drum score to a musician's timing, modulating how a pattern is played. The Living Looper (Shepardson and Magnusson, 2023; Shepardson et al., 2025) extends this lineage: rather than replaying recorded audio, each loop channel uses a pretrained RAVE autoencoder to project the recording into a latent space, where a lightweight linear model is fit per-recording, allowing the loop to drift and transform over time. Madaghiele et al. (2025) push this toward full autonomy: their Multi-Agent Autonomous Looper (MAAL) assigns a rule-based agent to each loop track; agents continuously evaluate the rhythmic and harmonic compatibility of incoming live segments and autonomously decide what to loop, removing manual record triggers entirely. *Recursive Speculation* (Madaghiele and Cotton, 2025) demonstrates this with vocal input, where MAAL autonomously samples and layers a performer's improvised extended techniques into evolving looped structures.

AI Drum Variation. Nuttall et al. (2021) first trained a Transformer-XL on the Groove MIDI Dataset (GMD) for drum sequence generation; Haki et al. (2022) subsequently demonstrated real-time Transformer-based drum accompaniment in performance contexts. Evans et al. (2024) extended this line of work into the GrooveTransformer, a

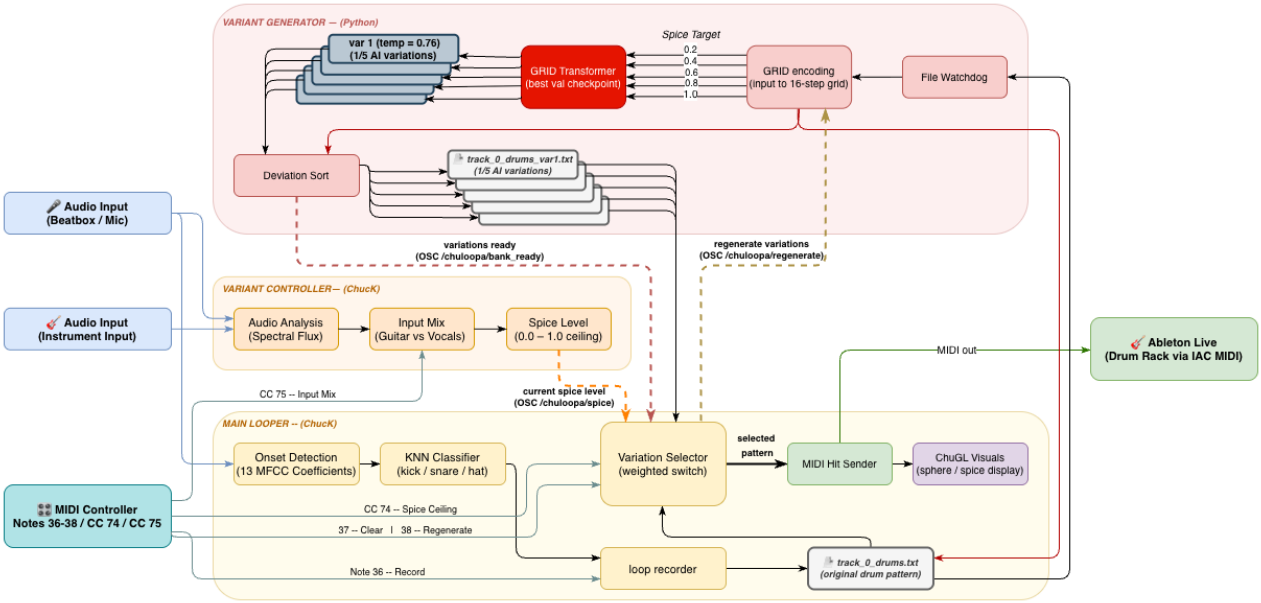

Figure 1: **CHULOOPA system architecture**. Three concurrent processes communicate via OSC: main looper, variant generator, and variant controller.

VAE-based Eurorack module for real-time generative drum sequencing. GrooveTransformer steers variation through two pre-selected latent anchors and live MIDI input; it does not derive selection from raw audio energy, and generation is bounded by those reference points rather than a single live-recorded loop. Bruford et al. (2020) present jaki, which generates 1-bar LSTM continuations of a seed drum pattern and tunes them to user-defined musical features (syncopation, density, repetition) via deep reinforcement learning. Alain et al. (2020) use active learning to converge on user drum-loop preferences from like/dislike ratings. Brosnan (2026) present DARC, a drum accompaniment model that conditions on musical context and a rhythm prompt (e.g. beatboxing), targeting music prototyping rather than live performance.

Beatbox Transcription. Mapping vocal percussion to drum classes is a cross-modal transcription task: Santos et al. (2025) formalize it as mapping arbitrary percussive inputs to kick/snare/hat labels. Delgado et al. (2019) showed that vocal percussion styles vary significantly across amateur performers, and Ramires et al. (2018) demonstrated that personalized, user-specific classifiers substantially outperform generic models because different users produce acoustically distinct vocal percussion for the same drum class. Martanto and Kartowisastro (2025) confirm that KNN with MFCC features achieves strong accuracy for voice-derived percussion. Under live performance latency constraints, Stowell and Plumbley (2010) demonstrated real-time beatbox classification using delayed decision-making, and Weber et al. (2024) showed that high-quality drum transcription is achievable with minimal training data. Together, this body of work prioritizes personalization and real-time latency over generalization.

What sets CHULOOPA apart from its closest neighbors is the *selection signal*. Where jaki (Bruford et al., 2020) and DeepDrummer (Alain et al., 2020) tune variation to explicit user targets or like/dislike ratings, GrooveTransformer (Evans et al., 2024) steers between two pre-set latent anchors via live MIDI, and DARC (Brosnan, 2026) conditions on a rhythm prompt for offline prototyping, CHULOOPA derives selection autonomously from live audio energy and requires no manual variation control during performance. MAAL (Madaghiele et al., 2025) is the closest in spirit: it autonomously decides *what* to loop at the track level. CHULOOPA instead takes a single recorded loop as given and autonomously decides *how* to vary it, driven by the performer's live audio energy rather than rule-based agent evaluation.

These three areas collectively motivate CHULOOPA's design: a personalized beatbox classifier trained on the performer's own voice, a Transformer-based variation bank for generation, and live audio energy as the autonomous selection signal, replacing manual variation control with continuous, performance-driven adaptation.

Human–AI Co-Creativity. Although variation selection is autonomous in mechanism, needing no manual toggle, the overall performance is collaborative. CHULOOPA acts less as an autonomous generator than as a co-creative partner (Lubart, 2005), with control over the evolving loop shared between performer and system. The performer sets the material (the recorded pattern) and the bounds of the system's latitude (the spice ceiling); the system decides, moment to moment, how far to depart from that material in response to the live energy. This places CHULOOPA within recent work on shared control in AI performance tools (Browne, 2025), where the musical interest lies in how control over the performance shifts between performer and system over time.

3 SYSTEM DESIGN

CHULOOPA runs as three concurrent processes, color-coded in Figure 1: the main looper (yellow) handles beatbox recording, drum classification, loop playback, and variation selection; the variant generator (red) generates a bank of five AI variations in the background the moment a new pattern is recorded; and the variant controller (orange) continuously monitors the live audio environment,

streaming a single *spice* value every 500 ms. Separating spice detection into its own process was a deliberate design choice. Both the main looper and variant controller process live audio, but for different purposes: drum onset classification in the former and dBFS energy measurement in the latter.

3.1 Beatboxing Classifier

To initialize CHULOOPA, the performer first teaches the system their own beatboxing vocabulary. Rather than relying on a generic drum-sound dataset, CHULOOPA asks the performer to record 10 examples each of kick, snare, and hi-hat, creating a small personalized classifier that adapts to the performer's vocal timbre and articulation in under two minutes. We restrict the vocabulary to three classes. Kick, snare, and hi-hat span the low, mid, and high spectral bands the onset features already discriminate, they align with prior beatbox-transcription work (Stowell and Plumbley, 2010; Ramires et al., 2018; Martanto and Kartowisastro, 2025), and they keep per-user training brief. It leaves out the toms, claps, and cymbal colors that many loop-based genres rely on, a limitation we revisit in the conclusion.

The system extracts 13 MFCC coefficients per onset, informed by prior beatbox-classification work using MFCC-based features (Martanto and Kartowisastro, 2025). A KNN classifier (Cover and Hart, 1967) (k=3), implemented via ChAI's `KNN2` object (Li and Wang, 2024), trains automatically at startup in under one second. KNN was chosen deliberately: it performs well with as few as 30 samples, trains instantly, and requires no hyperparameter tuning. User-specific training constrains the problem space enough that a simple classifier suffices.

3.2 Real-Time Transcription

Immediate audio feedback during recording is central to CHULOOPA's performance model: the performer hears classified drum samples in real-time while beatboxing, allowing technique correction before committing a pattern to loop. This requires reliable onset detection and sub-30 ms classification latency.

Onset Detection: Spectral flux with adaptive thresholding (1.5× running mean) in 512-sample frames, 128-sample hop (75% overlap). The 512-sample frame provides 86 Hz frequency resolution at 44.1 kHz, necessary to discriminate kick fundamentals (∼60–100 Hz) from mid-frequency snare energy.

Classification & Playback: Each onset is classified (kick/snare/hat) via MFCC-13 KNN and the corresponding drum sample plays immediately. Simultaneously, a MIDI note-on is sent to Ableton Live via the macOS Inter-Application Communication (IAC) driver, routing hits to a Drum Rack and giving the performer full control over drum sound design independently of the classification engine. Total input-to-playback latency is ∼25 ms. Each hit is stored with delta-time encoding, the interval to the next hit or to loop end for the final hit, enabling drift-free loop playback. For AI variation, this timestamped sequence is quantized to a 16-step grid and written back as the active playback loop; the quantized version is then passed to the GRID model, ensuring the original and all variations share the same temporal reference.

3.3 Spice Detection

To autonomously select from the variation bank, we define *spice* as a normalized measure of live audio energy that controls the selection of generated variations. The variant controller continuously measures the energy of the live performer and maps it to a spice value from 0.0 to 1.0, allowing CHULOOPA to autonomously respond to changes in musical intensity. RMS energy was chosen as the primary signal because it captures the cumulative acoustic energy of the room, with no additional sensing required. This intensity naturally invites more adventurous drumming, making it a reliable proxy for when variation is musically appropriate. Quieter passages favor conservative patterns; building energy pulls the system toward more adventurous ones.

Spice rests on an explicit assumption: that rising acoustic energy signals rising musical intensity, and that intensity is where denser, more adventurous drumming is welcome. This is a heuristic rather than a rule. A loud passage may want a steady backbeat, and a quiet one a busy fill, so spice is a proxy the performer can always override through the spice ceiling (CC 74). We adopt amplitude as a first, robust signal because it is always available and needs no additional sensing. Enriching it with spectral and rhythmic-density cues, so the system reasons about intensity from more than loudness alone, is an open direction we return to in the conclusion.

The detector averages the RMS energy over a 500 ms window and converts to a dBFS scale:

$$\text{spice} = \text{clamp}\left(\frac{\text{dBFS} - F}{P - F},\ 0.0,\ 1.0\right) \tag{1}$$

where F is the noise-floor threshold (–75 dBFS) and P is the expected peak performance level (–50 dBFS). The dBFS mapping was chosen deliberately: a linear RMS scale would compress most of the musically-useful dynamic range near zero, whereas loudness perception is logarithmic.

In performance mode, with a two-channel audio interface, the input mix knob blends the instrument input (an electric guitar in the reported sessions) and the vocal input before analysis, so spice reflects the collective energy of the full performance rather than the beatboxer's voice alone. The 500 ms window smooths over transient spikes while remaining responsive to sustained shifts in musical energy. Spice is streamed to the main looper process via OSC every 500 ms.

3.4 AI Variation Bank

Variation model: CHULOOPA's generation engine is a grid-based decoder-only transformer (GRID), a 6-layer model with 256-dim embeddings, 8 attention heads, feed-forward dimension 1024, 42-token vocabulary, and 4.8M parameters, trained on the Expanded Groove MIDI Dataset (Callender et al., 2020). Bar pairs are constructed from performances containing at least two bars; pairs with empty bars or adjacent duplicates are discarded, yielding 12,085 training pairs. The model is conditioned on bar *pairs*: given a context bar, it generates the immediately following bar. A masked loss is applied only to the continuation bar, training the model to predict the next bar rather than reconstruct the context. Patterns are quantized to a 16-step grid and encoded as alternating P{step}/N{pitch}

tokens; generation terminates at a learned end-of-sequence marker.

When a loop is recorded, the variant generator immediately launches five parallel generation threads, one per spice target (0.2, 0.4, 0.6, 0.8, 1.0), chosen to maximise coverage of the spice spectrum while keeping generation overhead tractable; finer granularity would produce negligible perceptual difference between adjacent variants. Following the same encoding process, each thread feeds the recorded pattern as a GRID token sequence to the model, generating a new bar conditioned on the input. Each thread is assigned a sampling temperature that rises with its spice target ($T = 0.9 + 0.2 \times \text{spice}$, giving $T = 0.94$ for the lowest-target thread and $T = 1.10$ for the highest). In practice temperature has limited effect. Because the model is strongly conditioned on the input bar, the five threads stay close to the recorded pattern regardless of temperature: in an ablation that swept temperature over a range wider than production (0.76–1.40), the pairwise difference between threads changed by under two percent of grid cells. This strong conditioning is what preserves the performer's pattern, but it also means temperature cannot be relied on to spread the variants. The conservative-to-adventurous ordering is instead established downstream, by deviation sorting.

To ensure the spice axis is musically reliable, the five generated variants are sorted ascending by the tuple $(\max(0, \Delta_{\text{hits}}), V_{\text{new}})$, where Δ_{hits} is the signed hit count difference from the original and V_{new} is the count of drum voice classes absent from the original (Gillick et al., 2019). Density increase is the primary criterion; new voice classes break ties within the same density bucket. Sparser variations are not penalised. After sorting, slot 1 always holds the most conservative result and slot 5 the most adventurous, independent of which thread generated it. This is the same conservative-to-adventurous range the performer sees on stage as the ECHO–VAR meter (Figure 3).

We take hit density as the primary variation axis because it is both perceptually salient (a groove audibly thickens or thins) and cheaply measurable from the symbolic grid, which makes it a dependable control signal for the spice mapping. This backgrounds other axes of variation, such as timbral substitution, swing, and syncopation, which the 16-step grid and three-voice vocabulary already constrain. A single candidate per spice target is sufficient because performance-time variety comes from selection across the whole bank rather than from any one slot: the weighted distribution (Table 1) spans several slots at every spice level, and the no-repeat-more-than-twice rule keeps any single variant from dominating. Maintaining a rolling pool of candidates per slot and sampling among them over successive cycles is a natural extension we leave to future work. In practice, then, spice governs which point on the sorted spectrum plays at each loop boundary; the variants themselves are generated in advance and are unaffected by the live spice value. The variant generator signals completion via OSC. Total generation time: 3–5 seconds on consumer CPU hardware, entirely offline. Figure 2 shows the result: the original pattern alongside Var 1 (Low Spice), Var 3 (Medium Spice), and Var 5 (High Spice) as 16-step drum grids, with sparse additions at low spice giving way to denser fills and new drum voices at high spice.

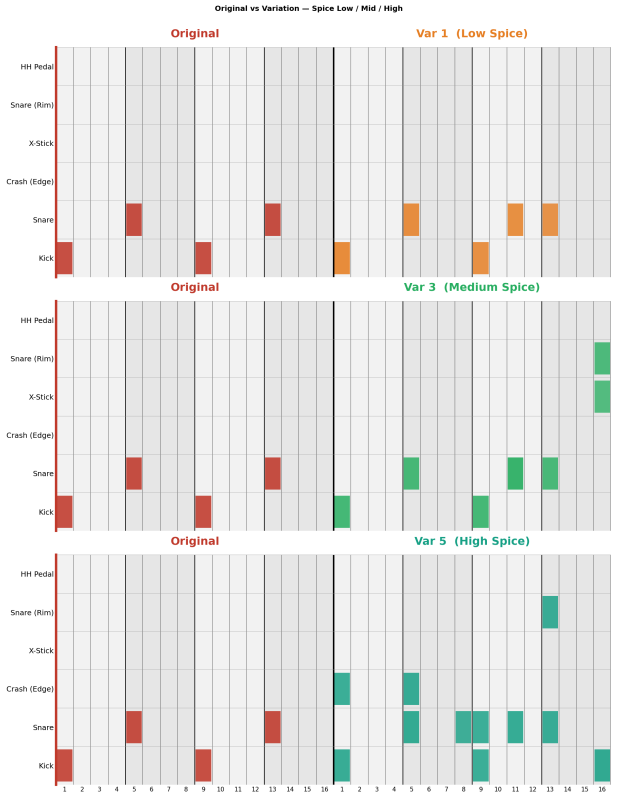

Figure 2: Drum grid comparison showing the original recorded pattern alongside AI-generated variations at low (Var 1), medium (Var 3), and high (Var 5) spice levels. Higher spice produces denser fills and introduces new drum voices (e.g. Crash (Edge) at high spice).

3.5 Weighted Variation Selection

At each loop boundary, the main looper draws from a weighted distribution that shifts toward higher-numbered variants as spice rises (Table 1). The distribution includes the original pattern as a valid option: at low spice the system may simply continue playing the original, treating restraint as a musical choice. Selection uses a **rolling 4-bar spice average**, with no variant playing more than twice consecutively.

Table 1: Selection weights per spice tier. Cell shading indicates probability mass (darker = more likely). Values between tiers are linearly interpolated at loop boundaries.

Spice	Orig.	V1	V2	V3	V4	V5
0.00	0.50	0.40	0.10	0.00	0.00	0.00
0.25	0.20	0.40	0.30	0.10	0.00	0.00
0.50	0.05	0.20	0.40	0.30	0.05	0.00
0.75	0.00	0.10	0.30	0.40	0.20	0.00
1.00	0.00	0.00	0.20	0.40	0.30	0.10

The live performer retains creative control through a configurable spice *ceiling*, mapped to **CC 74**. Rather than selecting variations manually, the performer decides how much latitude to give the system to respond to the music accordingly.

3.6 Live Performance Controls

Inspired by hardware loopers, variation loading queues during the current cycle and executes at the next loop boundary, preventing mid-pattern interruptions. Each playback

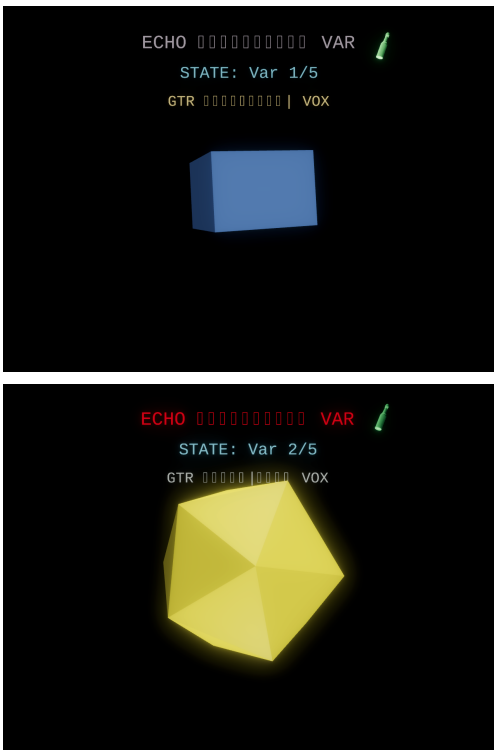

Figure 3: CHULOOPA performance UI. **Top:** Var 1 at low spice (blue cube). **Bottom:** Var 2 at high spice (yellow icosahedron). The central shape's color reflects the active variation slot (blue = Var 1 through red = Var 5) and its geometry tracks live spice independently (cube at low energy to icosahedron at peak). On-screen readouts: STATE (active slot, n/5), the ECHO–VAR meter (position on the conservative-to-adventurous spectrum), and the GTR–VOX meter (live guitar/vocal input mix, CC 75).

session holds a unique ID; scheduled drum hits validate their session before firing, eliminating overlap during transitions.

MIDI Controls: variation selection is fully automatic; no manual toggle required:

- **Note 36** (hold): record loop
- **Note 37**: clear loop
- **Note 38**: regenerate full variation bank
- **CC 74**: spice ceiling (caps audio-driven selection)
- **CC 75**: instrument/vocal input mix

ChuGL (Aday and Wang, 2024) visual feedback (Figure 3) closes the loop for the performer: the central shape shifts color with the active variation slot (blue = Var 1 through red = Var 5) and pulses on each hit, while a spice slider reflects the instantaneous spice level. The color is discrete, updating at each loop boundary; shape geometry tracks live spice, shifting from a cube at low energy to an icosahedron at peak. A pulse fires at the moment each beatbox onset is classified, letting the performer confirm each hit in real time. Three on-screen readouts orient the performer at a glance: STATE shows the active slot ($n/5$); the ECHO–VAR meter shows where that slot sits on the conservative-to-adventurous spectrum, from echoing the original pattern with minimal change (ECHO) to the most energetic, furthest-departing variation (VAR); and the GTR–VOX meter shows the live guitar/vocal input mix (CC 75).

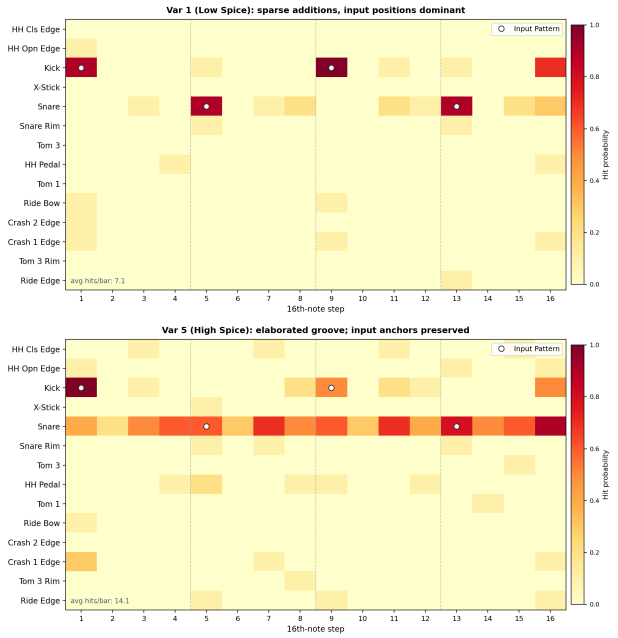

Figure 4: Hit-probability heatmaps for a simple kick–snare input, aggregated over N=10 independent banks. White circles mark input pattern positions. **Top:** Var 1 (Low Spice, 0.2): sparse additions, input positions dominant (92% activation). **Bottom:** Var 5 (High Spice, 1.0): elaborated groove with snare fills, hi-hats, toms, and cymbal voices; the kick input anchors at steps 1 and 9 remain preserved.

A hot-sauce icon in the corner tracks variation-bank readiness, turning green once all five variants are generated and staying grey while generation is still running in the background.

4 EVALUATION

The central evaluation question is whether CHULOOPA's AI-generated variations stay coherent with the performer's recorded pattern while offering a reliable, controllable range of complexity.

4.1 Variation Analysis

As an illustrative case, we first run 10 independent generation banks on a simple two-kick, two-snare pattern spanning 2.6 seconds. Each bank contains five variants at spice levels 0.2–1.0, deviation-sorted so slot 1 is always the most conservative result and slot 5 the most adventurous. Hit-probability heatmaps (Figure 4) aggregate binary grids across all 10 runs; darker cells indicate more consistent activation.

At low spice, Var 1 produces an average of 7.1 hits/bar, with 92% of input cells activating consistently: the model holds onto what the performer played. Additional voices appear only at low probability (1.1% averaged across all empty grid cells), acting as occasional ornamentation rather than structural additions. At high spice, Var 5 reaches 14.1 hits/bar: the snare row spreads broadly, the kick keeps its input anchors, and hi-hat, tom, and cymbal voices emerge. Even here additions stay targeted. Mean activation across empty cells is 3.7%, and input positions still fire 72.5% of the time, so the model elaborates the groove rather than filling the grid at random.

Table 2: Variation statistics across seven input patterns (N=10 banks each). V1/V5 density in hits/bar (the bank is density-sorted, so V5 $\geq$ V1 by construction); V1 overlap is the mean activation probability at input positions in the most conservative slot. The quantities of interest are the size of the V1-to-V5 spread and the degree of input preservation.

Input pattern	Hits	V1 dens.	V5 dens.	V1 overlap
Simple kick–snare	4	7.1	14.1	92%
Syncopated kicks	4	8.1	13.2	45%
Half-time sparse	6	6.2	13.6	60%
Four-on-the-floor	10	10.9	17.3	74%
Boom-bap	14	14.3	18.8	86%
Syncopated break	14	14.1	19.2	81%
16th-note hats	20	19.2	23.2	88%

To test that this behaviour generalises beyond one simple input, we repeat the evaluation across seven input patterns spanning 4–20 hits and distinct rhythmic feels: a half-time sparse groove, four-on-the-floor, boom-bap, a syncopated break, and a dense sixteenth-note hat pattern, alongside the two simple kick–snare inputs (Table 2).[1] The slot-1-to-slot-5 ordering is monotonic by construction, since the bank is sorted by density; what the seven patterns test is whether that ordering carries a usable spread and whether the input survives it. Both hold. Mean density climbs from 11.4 to 17.1 hits/bar (Figure 5), a spread wide enough to be musically distinct at every input level, and each pattern's variations stay stratified around its own input density rather than collapsing toward a common output. Input preservation at slot 1 averages 75.1% across patterns; it is strongest for denser grooves (86–92%) and weakest for the two sparsest inputs: syncopated kicks (4 hits, 45%) and half-time sparse (6 hits, 60%). Syncopation alone does not explain the drop: syncopated break (14 hits) is equally syncopated but holds 81% overlap, in line with the other mid-to-dense patterns. What the two weak patterns share instead is a low hit count, so each hit is a larger share of the total and a single reinterpreted hit swings the percentage further, leaving the model fewer anchors to hold onto. Because generation is stochastic, banks vary run to run: per-slot density has a standard deviation of roughly 1 to 2 hits/bar across the ten banks (mean 1.1 at slot 1, 2.0 at slot 5), so the more adventurous slots are also the less predictable ones.

4.2 Aesthetic Observations

As a preliminary, single-practitioner complement to the statistics above (pending the controlled listener study we scope in the Conclusion), the first author performed with CHULOOPA across multiple solo sessions of electric guitar and beatboxing, each lasting 10–20 minutes.[2] Across these sessions, generated variations were consistently judged coherent musical ideas by the performer, with occasional exceptions traceable to input-side quantization artifacts rather than output-side incoherence. The strict 16-step grid eliminates beat-drift entirely.

[1]Two patterns (simple kick–snare, syncopated kicks) are live-recorded beatbox takes; the remaining five are hand-authored directly on the 16-step grid. All seven align with the grid to within 0.2 steps after phase correction, so quantization is not approximating swung or rubato timing in this evaluation.

[2]Demo and supplementary materials: https://sites.google.com/view/chuloopa/home

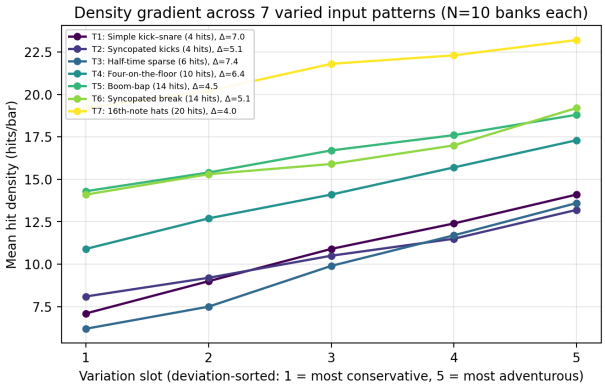

Figure 5: Mean hit density per deviation-sorted slot across seven varied input patterns (N=10 banks each). The bank is density-sorted, so each line rises by construction; what generalises is that the spread stays musically usable and the lines remain stratified by input density across every pattern, not a single test case.

Mid-range spice values (approximately 0.4–0.6) proved most musically satisfying, neither too sparse nor too elaborate. Hearing the system return to the original pattern at low spice served as a rhythmic anchor, reinforcing awareness of the source groove. Unlike static loops, which can grow stale when brought to the forefront of the performance, CHULOOPA invites attention when given the spotlight. When holding a chord or leaving space in the arrangement, the variations shine as a musical statement in their own right, with the performer remaining compositionally in control.

Beyond solo practice, two further informal signals point the same direction. CHULOOPA was performed live for roughly 70 people at a public showcase in Los Angeles, integrated directly into the first author's normal set (beatboxing $\rightarrow$ MIDI drum playback $\rightarrow$ AI variation, with no change to his usual flow); from the performer's perspective the variations added a distinct intrigue and drew audience attention to that segment. Separately, a demo video posted publicly to the first author's Instagram[3] drew unprompted positive comments from viewers with no connection to the project. The video is unrelated to the LA performance, and neither is controlled evidence, but both are informal, unblinded read-outs that extend the coherence judgment above from the performer alone to the performer and outside viewers, and suggest the system reads as coherent beyond the practice room.

5 CONCLUSION

Across seven varied input patterns, CHULOOPA's deviation sorting turns the variation bank into a usable, measurable complexity axis: mean density spans 11.4 to 17.1 hits/bar from slot 1 to slot 5, while the model preserves original hit positions at low spice (92% for a representative pattern, 75.1% on average across the seven), treating the performer's pattern as a structural anchor rather than a seed to discard. Spice drives autonomous selection at each loop boundary, matching the variation to the current musical energy without demanding performer intervention. An AI looper can elaborate a groove without displacing it.

[3]https://www.instagram.com/p/DXAUKppklBq/?hl=en

CHULOOPA's constraints are tradeoffs. The 16-step quantization grid is the source of its temporal reliability, but it excludes swing feel, non-4/4 meters, and 32nd-note microtiming. Variations are conditioned to continue from the input rather than replicate it, tied to the original but free to reinterpret it. The three-voice vocabulary (kick, snare, hi-hat) keeps training minimal and classification reliable but excludes the toms, claps, and cymbals many loop-based genres depend on; expanding it is plausible, though each added class demands more per-user training examples and sharpens the timbral confusability the personalized classifier must resolve. Spice tracks amplitude alone, a reliable proxy for musical energy but blind to harmonic or rhythmic context. Our evaluation demonstrates input preservation and density control objectively across varied patterns. Musical coherence has so far been judged only informally (by the performer across solo sessions and a public LA performance, and by audiences at that performance and online), while the felt sense of control remains assessed only by the first author. Performer-centred studies (measuring perceived controllability, learnability, and trust in the autonomous variation) and listener studies (assessing perceived coherence and whether variation tracks energy) remain the important next steps, this time under controlled, blinded conditions.

Natural extensions include multi-track looping with coherent cross-track variation, swing and compound-meter support, and melodic generation. An open question is whether spice can be enriched by incorporating spectral complexity and rhythmic density alongside amplitude, and how drum patterns are expected to adapt to those changes, giving the selection engine a fuller picture of when variation is musically called for. Together, the variation bank and autonomous selection keep the loop's repetition adjustable in real time, without asking the performer to manage it by hand.

AUTHOR DECLARATIONS

Conflicts of Interest: None.

Ethical Issues: No formal ethical review was required for this research. The GRID model was trained on the publicly available Expanded Groove MIDI Dataset (CC-BY 4.0).

Use of AI: Claude (Anthropic) assisted with technical implementation (system development, debugging) and paper writing (structuring, editing). All research design, conceptual decisions, and conclusions are the authors' own.

ACKNOWLEDGMENTS

This work received no external funding. The authors thank their peers in the Music Technology program at California Institute of the Arts for their feedback and recommendations.

REFERENCES

Aday, A. Z. and Wang, G. (2024). Chugl: Unified audiovisual programming in chuck. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Utrecht, The Netherlands.

Alain, G., Chevalier-Boisvert, M., Osterrath, F., and Piché-Taillefer, R. (2020). Deepdrummer: Generating drum loops using deep learning and a human in the loop. In *Proceedings of the Joint Conference on AI Music Creativity (CSMC + MuMe)*.

Assayag, G., Bloch, G., Chemillier, M., Cont, A., and Dubnov, S. (2006). OMax brothers: A dynamic topology of agents for improvisation learning. In *Proceedings of the ACM Workshop on Audio and Music Computing Multimedia*, pages 125–132.

Brosnan, T. (2026). DARC: Drum accompaniment generation with fine-grained rhythm control. *arXiv preprint arXiv:2601.02357*.

Browne, E. (2025). The shape of surprise: Structured uncertainty and co-creativity in AI music tools. In *Proceedings of the 6th Conference on AI Music Creativity (AIMC 2025)*, Brussels, Belgium.

Bruford, F., McDonald, S., and Sandler, M. (2020). jaki: User-controllable generation of drum patterns using an lstm encoder-decoder and deep reinforcement learning. In *Proceedings of the Joint Conference on AI Music Creativity (CSMC + MuMe)*.

Burloiu, G. (2020). Adaptive drum machine microtiming with transfer learning and RNNs. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Late Breaking Demo*.

Callender, L., Hawthorne, C., and Engel, J. (2020). Improving perceptual quality of drum transcription with the expanded groove MIDI dataset. *arXiv preprint arXiv:2004.00188*.

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Delgado, A., McDonald, S., Xu, N., and Sandler, M. B. (2019). A new dataset for amateur vocal percussion analysis. In *Audio Mostly 2019: A Journey in Sound*, pages 1–7, Nottingham, United Kingdom.

Evans, N., Haki, B., and Jordà, S. (2024). GrooveTransformer: A generative drum sequencer eurorack module. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Utrecht, The Netherlands.

Gillick, J., Roberts, A., Engel, J., Eck, D., and Bamman, D. (2019). Learning to groove with inverse sequence transformations. In *International Conference on Machine Learning*, pages 2269–2279. PMLR.

Haki, B., Nieto, M., Pelinski, T., and Jordà, S. (2022). Real-time drum accompaniment using transformer architecture. In *Proceedings of the 3rd Conference on AI Music Creativity (AIMC 2022)*, Saint-Étienne, France.

Li, Y. and Wang, G. (2024). ChAI → interactive AI tools in ChucK. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Utrecht, The Netherlands.

Lubart, T. (2005). How can computers be partners in the creative process? classification and commentary on the special issue. *International Journal of Human-Computer Studies*, 63(4-5):365–369.

Madaghiele, V. and Cotton, K. (2025). Recursive speculation. In *Proceedings of the 6th Conference on AI Music Creativity (AIMC 2025)*, Brussels, Belgium.

Madaghiele, V., Fasciani, S., Kelkar, T., and Erdem, Ç. (2025). MAAL: a multi-agent autonomous live looper for improvised co-creation of musical structures. In *Proceedings of the 6th Conference on AI Music Creativity (AIMC 2025)*, Brussels, Belgium.

Marchini, M., Pachet, F., and Carré, B. (2017). Rethinking reflexive looper for structured pop music. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, pages 139–144, Aalborg University Copenhagen, Denmark.

Martanto, J. and Kartowisastro, I. H. (2025). Beatbox classification to distinguish user experiences using machine learning approaches. *Journal of Computer Science*, 21(4):961–970.

Nuttall, T., Haki, B., and Jordà, S. (2021). Transformer neural networks for automated rhythm generation. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, Shanghai, China.

Pachet, F. (2003). The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):333–341.

Ramires, A., Penha, R., and Davies, M. E. (2018). User specific adaptation in automatic transcription of vocalised percussion. *arXiv preprint arXiv:1811.02406*.

Santos, A., Cardoso, A., Davies, M. E. P., and Dannenberg, R. B. (2025). Tapping into a new paradigm: A synthetic strategy for automatic drum TapScription. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*.

Shepardson, V. and Magnusson, T. (2023). The living looper: Rethinking the musical loop as a machine action-perception loop. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, pages 1–8, Mexico City, Mexico.

Shepardson, V., Stefánsdóttir, H. S., and Magnusson, T. (2025). Evolving the living looper: Artistic research, online learning, and tentacle pendula. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*.

Stowell, D. and Plumbley, M. D. (2010). Delayed decision-making in real-time beatbox percussion classification. *Journal of New Music Research*, 39(3):203–213.

Wang, G., Cook, P. R., and Salazar, S. (2015). Chuck: A strongly timed computer music language. *Computer Music Journal*, 39(4):10–29.

Weber, P., Uhle, C., Müller, M., and Lang, M. (2024). Real-time automatic drum transcription using dynamic few-shot learning. In *2024 IEEE 5th International Symposium on the Internet of Sounds*.