# Self-Correcting Non-Chronological Autoregressive Music Generation

Wayne Chi*[1], Prachi Kumar*[1], Suri Yaddanapudi[1], Rahul Suresh[1], and Umut Isik[1]

*Equal Contributors, [1]Amazon Web Services

## Introduction

**Motivation**: Music composition is often non-chronological by nature, with motifs being added and inserted throughout the composition process. In addition, autoregressive techniques are prone to accumulation of errors.

**Our Work**: We generate music using a non-chronological, autoregressive model that is able to self-correct by adding or removing notes—even notes previously generated by the model.

**Human AI Collaboration**: In our use case, users collaborate with the model to enhance input melodies. Since we generate notes one-by-one and non-chronologically, users have a finer degree of control during the human AI collaboration process.

## Music Generation Background

Previous approaches to music generation treat music as image generation or as a time series problem analogous to autoregressive language modeling.

**Our Approach**: Takes elements from both image-based and time series generation.

**Coconet**[1]—the model behind Google's Bach Doodle—is another non-chronological autoregressive music generation model. Rather than directly modeling addition and removal of notes, Coconet uses Gibbs Sampling to prevent accumulation of error.

We compare against a Gibbs sampling approach using Coconet,

## What is a Piano Roll

**Piano Roll**: A 2D discrete representation of music as an image matrix across time and pitch.

We can map musical pieces to piano rolls with the following definitions:

- $T$: Number of time steps
- $P$: Number of note pitches
- $x$: A point in $\{0,1\}^{T\times P}$ which represents a piano roll. $x \in X$
- $p^{\text{PR}}(x)$: A probability density function on $\{0,1\}^{T\times P}$

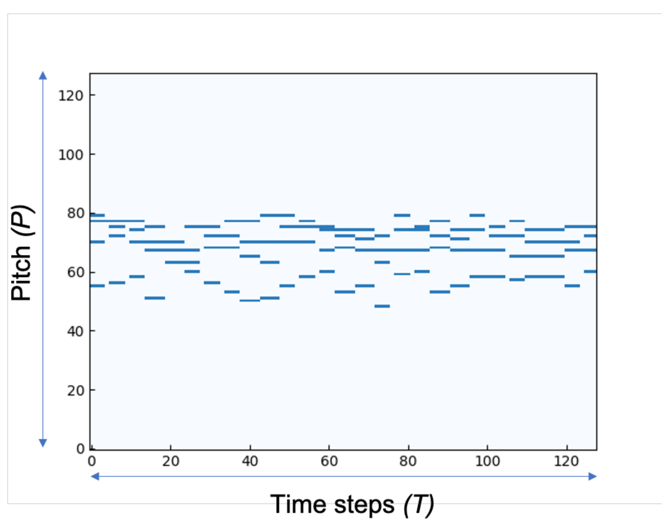



Figure 1: *A piano roll*

## Generating Music Using Edit Sequences

**Edit Sequence**: A tuple of **edit events** that can be mapped to a piano roll

**Edit Event**: A one-hot matrix $e^{(t,p)} \in \{0,1\}^{T\times P}$

We model $p^{\text{PR}}(x)$ as $p^{\text{ES}}(s)$ on the set of **edit sequences** (ES)

**Mapping Edit Sequences to Piano Rolls**

$$\pi : \bigcup_{M=1}^{\infty} \mathcal{E}^M \to \{0,1\}^{T\times P} \tag{1}$$

$$\pi(e_1, \ldots, e_M) = \sum_{i=1}^{M} e_i \pmod 2. \tag{2}$$

$\mathcal{E}$: set of all edit events. $\mathcal{E}^M$: edit sequences of length $M$.

**Mapping Between Joint Probability Distributions**

$$\begin{aligned} p^{\text{PR}}(x) &= p^{\text{PR}}(\{(t_1,p_1),\ldots,(t_N,p_N)\}) \\ &= \sum_{s\in\pi^{-1}(x)}^{\infty} p^{\text{ES}}(s) \end{aligned} \tag{3}$$

$N$: number of notes in the piano roll.
$\pi^{-1}(x)$: inverse image of $\pi(x)$.
$(t_i, p_i)$: time and pitch of a note or edit event.
$s$: sequence of edit events $(t_1,p_1)\ldots(t_M,p_M)$ where $M \geq N$.

$$p^{\text{ES}}(s) = p^{\text{ES}}\big((t_1,p_1),\ldots,(t_M,p_M)\big) = \prod_{i=1}^{M} p^{\text{ES}}\big((t_i,p_i)|(t_1,p_1),\ldots,(t_{i-1},p_{i-1})\big) \tag{4}$$

**Assumption**: $p^{\text{ES}}((t_i,p_i)|(t_1,p_1),\ldots,(t_{i-1},p_{i-1}))$ is ordering invariant.

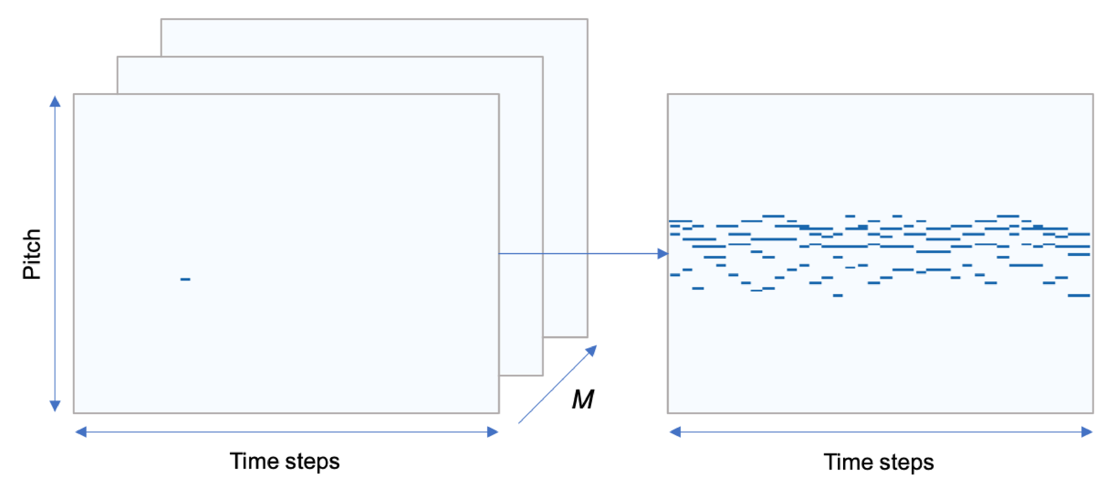



Figure 2: *Mapping from an edit sequence (left) of length M to a piano roll (right). Each slice in an edit sequence is the addition or removal of a note.*

## References

[1] Cheng-Zhi Anna Huang et al. "Counterpoint by convolution". In: *arXiv preprint arXiv:1903.07227* (2019).

[2] Cheng-Zhi Anna Huang et al. "The Bach Doodle: Approachable music composition with machine learning at scale". In: *International Society for Music Information Retrieval (ISMIR)*. 2019. URL: `https://goo.gl/magenta/bach-doodle-paper`.

[3] Benigno Uria, Iain Murray, and Hugo Larochelle. "A deep and tractable density estimator". In: *International Conference on Machine Learning*. 2014, pp. 467–475.

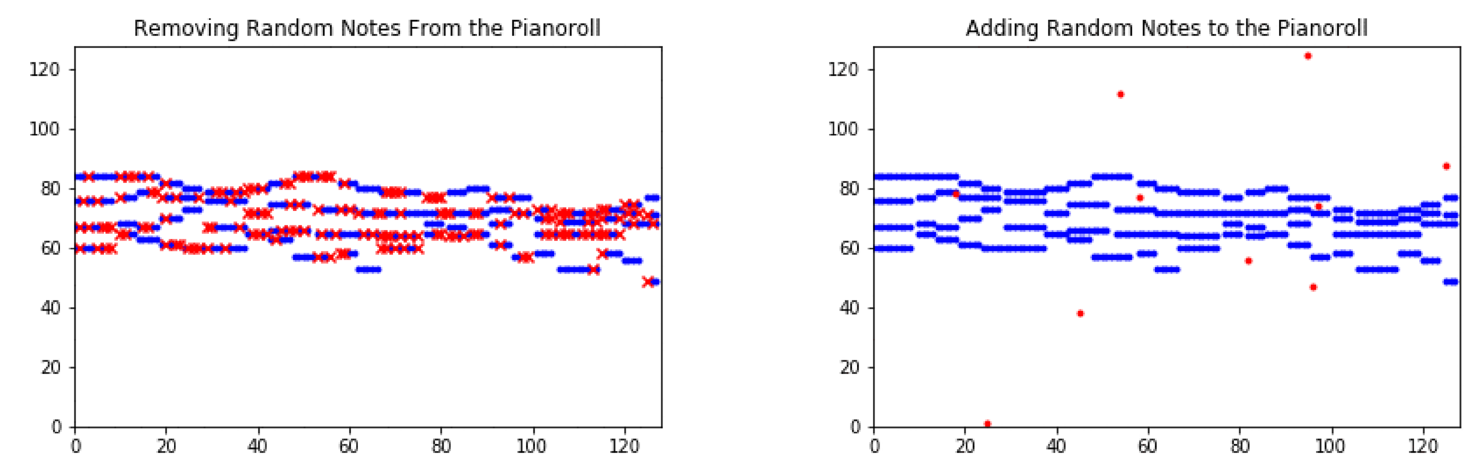
## Model Training and Inference

We train the model to add and remove notes by masking existing notes and adding random extraneous notes to each sample.

$\mathcal{T}$: Target piano roll; real piano rolls
$\mathcal{I}$: Input piano roll; piano rolls with masked and added notes
$D_{KL}$: Kullback-Leibler divergence
$P$: Softmax over the model's output for all times and pitches
$U$: Uniform distribution supported on $\mathcal{I}\Delta\mathcal{T}$.

**Loss Function**

$$\mathcal{L}(\mathcal{I},\mathcal{T},P) = D_{KL}(P \parallel U), \tag{5}$$




We sample from the model's output probabilities through **direct ancestral sampling**.

Steps in a single inference iteration

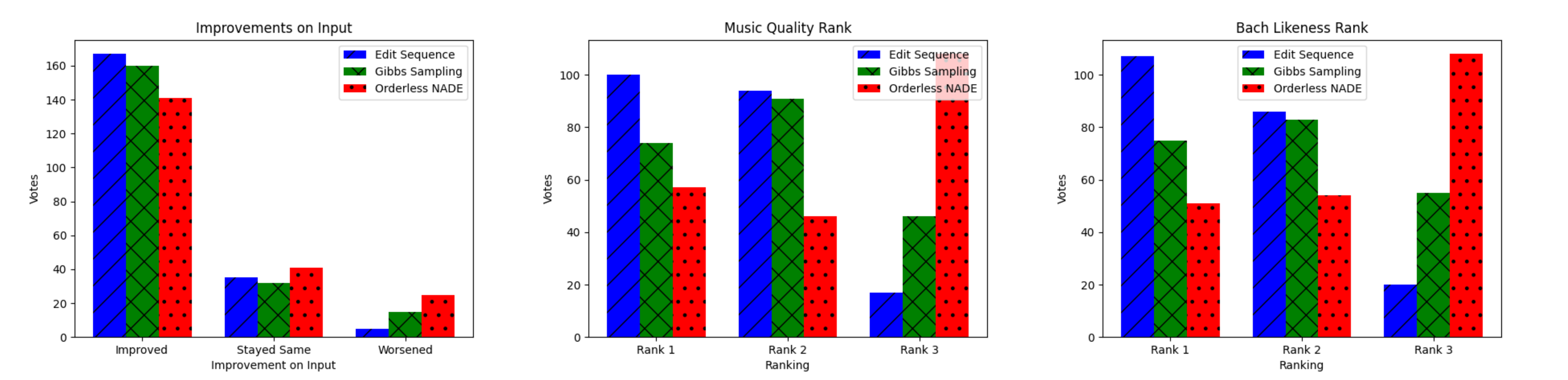
- Feed the input melody to the model.
- Sample the next edit event from the softmax applied over all times and pitches.
- Modify the input based on that edit event.
- Feed that modified melody back into the model.

## Empirical Evaluation

**Human Opinion Survey** We build an orderless NADE [3] model and use Coconet [1] to represent a Gibbs sampling approach

We trained using the JSB Chorales dataset [1], and used Bach Doodle dataset [2] for input melodies.

We see that **our approach outperforms both orderless NADE and Gibbs Sampling overall** in Figure 3.




a b c

Figure 3: Human Survey Evaluation Ratings. (a) describes whether users thought samples improved on the input. (b) describes user rankings for music quality. (c) describes user rankings for how similar a sample is to real Bach data.