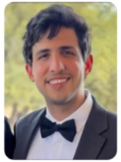
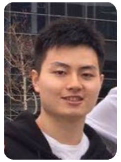
VaryNote: A Method to Automatically Vary the Number of Notes in Symbolic Music

https://varynote.github.io

Juan Huerta, Bo Liu, and Peter Stone

CMMR 13 - November 14, 2023

Learning Agents Research Group
The University of Texas at Austin


Overview

VaryNote can increase or decrease the number of notes of any midi piece by any desired multiple. To achieve this we only need a corpus of chord labeled examples

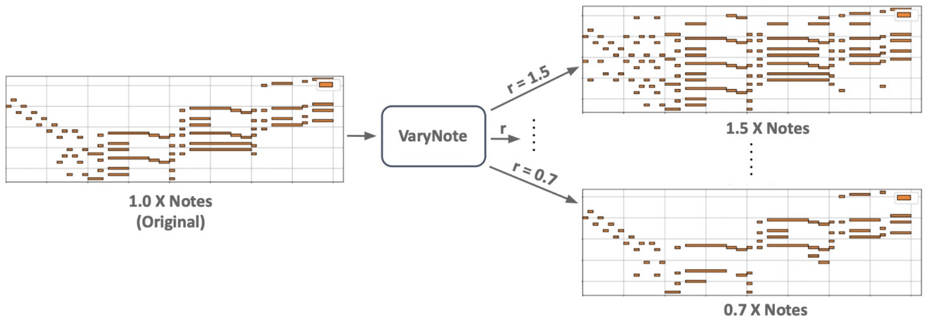


Fig 1: VaryNote example usage: given a piece of MIDI music we varying the number of notes according to a desired input-output ratio: r.

Applications

- Enhance, or ornament, compositions
- Simplifying music for learning
- Data augmentations for MIR applications

Contributions

- Formulate the task of varying the number of notes in music as an optimization problem.
- Introduce a new way to think about training pitch autoencoders with multiple music objectives

Agenda

Part 1: Algorithm Breakdown (5 min)

- ❑ Method
  - ❑ VaryNote Architecture
  - ❑ VaryNote Training
  - ❑ Masking Mechanism
  - ❑ A Simple Music Theory Baseline

Part 2: Experimental Design and Results (5 min)

- ❑ Experiment
  - ❑ Evaluating Harmonic Similarity
  - ❑ Comparing output with KKL
  - ❑ Human Survey results
  - ❑ Listening Examples

AutoEncoder Architecture

Part 1: Algorithm Breakdown

The goal is to learn to reconstruct a pitch vector x_t at time t using the encoder with $d = 32 : E_\phi : \mathbb{R}^{\mathcal{X}} \to \mathbb{R}^d$ and decoder $D_\theta : \mathbb{R}^d \to \mathbb{R}^{\mathcal{X}}$, parameterized by ϕ and θ respectively.

Why an autoencoder?

- Simplicity, and speed
- Unsupervised training method

Activations functions:

- ReLU - Standard
- K-WTA - sparsity constraints in the bottleneck
- Lifetime - sparsity constraints in the batch level

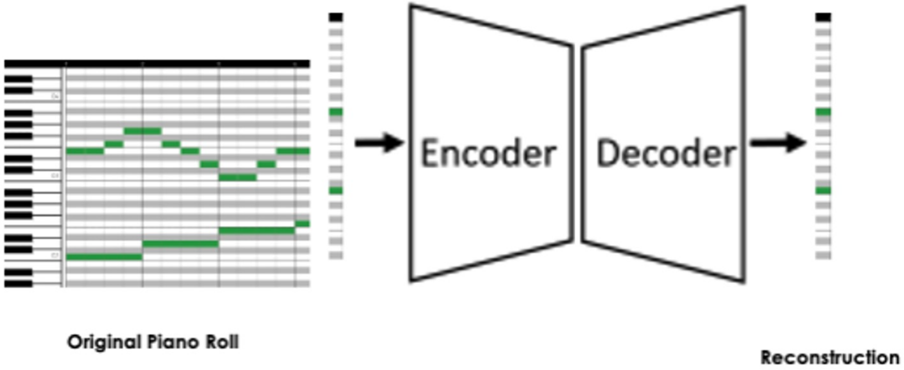


Fig 2: Diagram of a basic pitch autoencoder. The objective is to learn a compressed representation of the data.

In the next page I will explain these activations →

AutoEncoder Architecture

Part 1: Algorithm Breakdown

k-WTA

k-WTA: the k-largest neurons in the autoencoder's hidden layer (or code) is kept and the rest, as well as their derivatives, are set to zero.

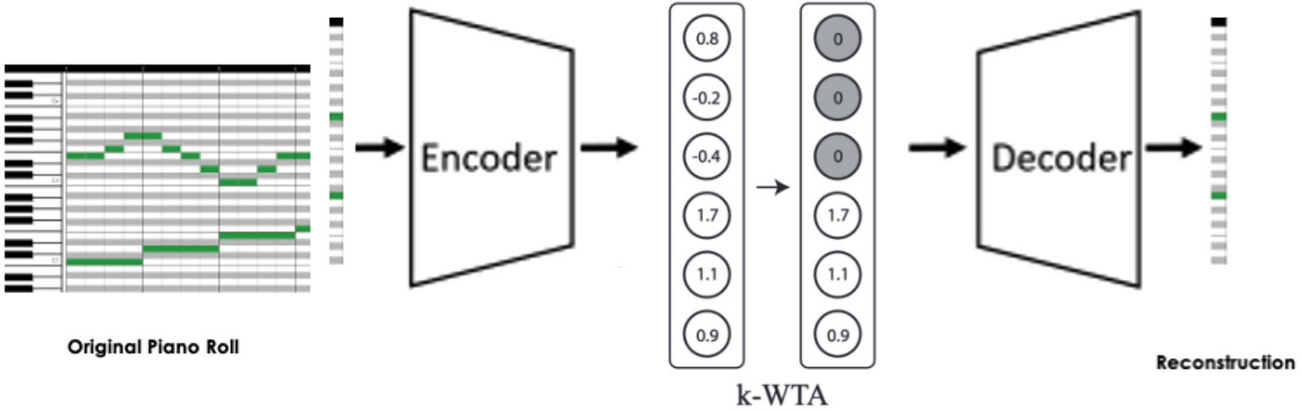


Fig 3: Diagram of a basic pitch autoencoder with k-WTA activation: only the k-top neurons in the bottleneck are kept during every pass.

Lifetime sparsity

Lifetime sparsity keeps the k largest activation of that hidden unit across the mini-batch samples and setting the rest of activations of that hidden unit to zero. This encourage a wider range of neurons to be active according to previous research.

The Autoencoder

Part 1: Algorithm Breakdown

Combined Auxiliary Loss

1) $$\mathcal{D} = L_{\text{total}} = L_{\text{MSE}} + cL_{\text{CE}}$$
$$= \frac{1}{P}\sum_{t=1}^{P}(x_t - \hat{x}_t)^2 - \frac{c}{N}\sum_{i=1}^{N}\log\frac{\exp\left(o_t[y_i]\right)}{\sum_{y=1}^{K}\exp\left(o_t[y]\right)}.$$

Finally, VaryNote trains with the presented L_{total}

2) $$\min_{\theta,\phi} L_{\text{total}}\left(F_{vc}(X \mid r),\ X\right) \quad \text{s.t.} \quad \frac{||F_{vc}(X \mid r)||_0}{||X||_0} = r.$$

- **C** is the number of chord classes
- **H** is time step
- $o_t \in \mathbb{R}^{C \times H}$ s the chord sequence Bi-LSTM output
- **c** is a constant to weight loss proportion
- **N** = 24 possible chords

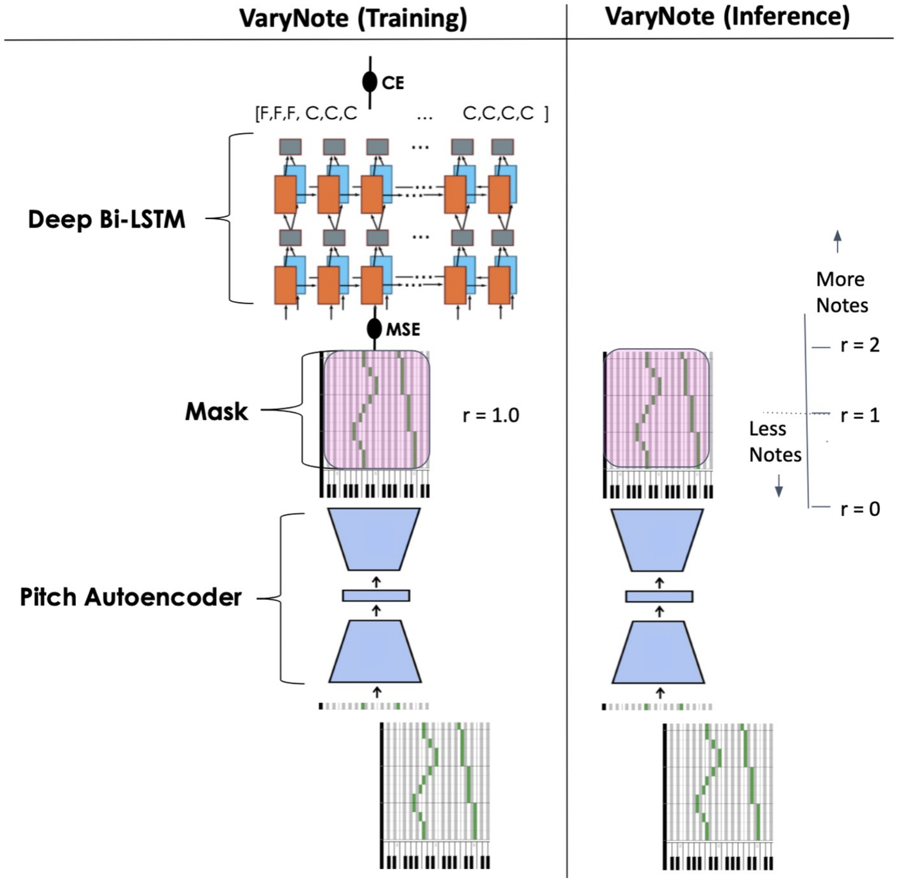


Fig 4: During training VaryNote combines MSE loss and softmax cross entropy loss. The mask requires an output-input ratio r. During training we can fix r, and apply the mask during inference.

The Masking Mechanism

Part 1: Algorithm Breakdown

To increase the number of notes, r > 1

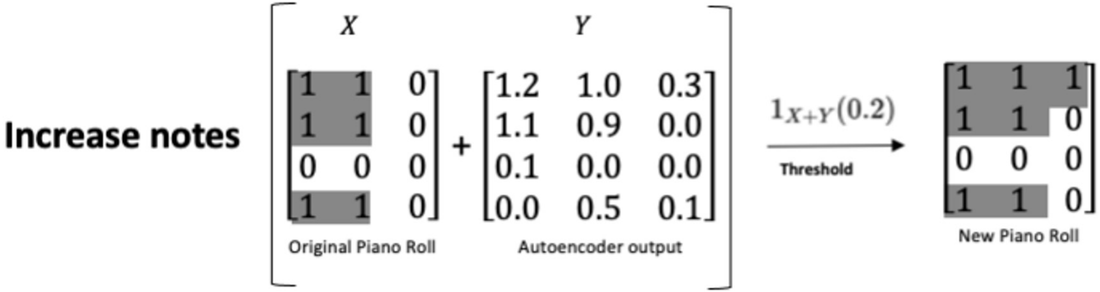


Fig 5. Increasing Number of Notes: to apply a relative increase in number of notes (output- input ratio r ≥ 1), we add the pitch autoencoder output with the original music and apply the mask in Eq. (8) that assures we meet the desired output-input ratio constraints.

The Masking Mechanism

Part 1: Algorithm Breakdown

To decrease the number of notes, r < 1

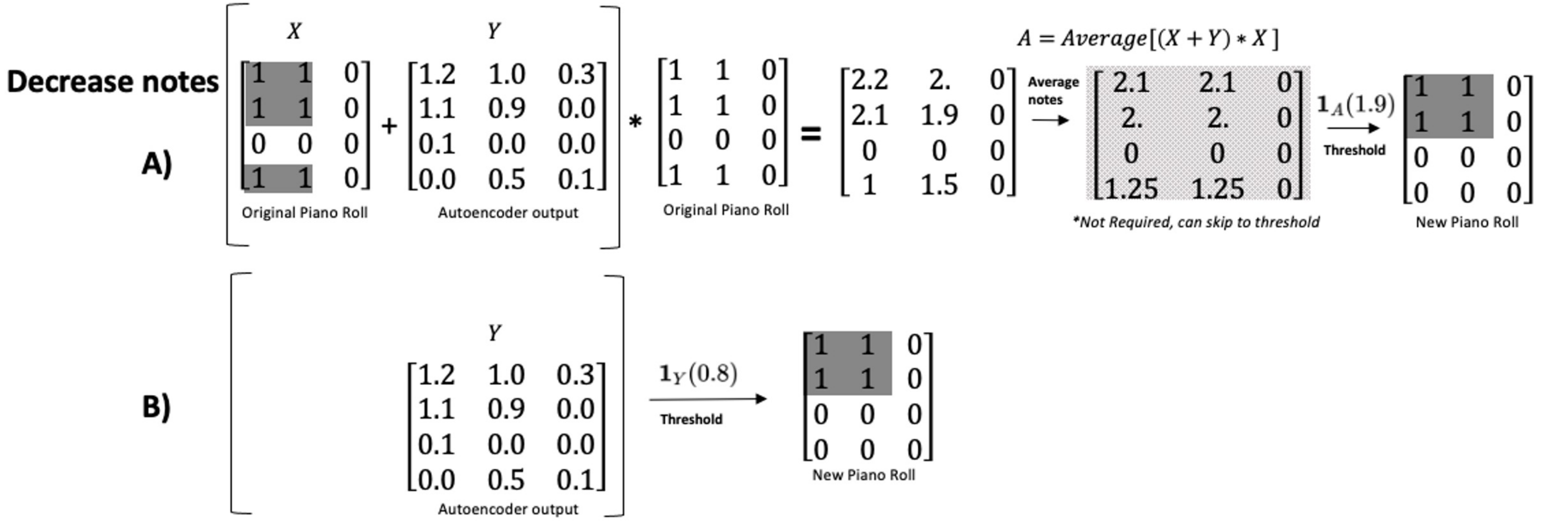


Fig 6. Decreasing Number of Notes: to apply a relative decrease in number of notes (output- input ratio r < 1), we multiply, element-wise, the pitch autoencoder output with the original music and apply the mask in Eq. (8) that assures we meet the desired output- input ratio constraints

A Music Theory Baseline

Part 1: Algorithm Breakdown

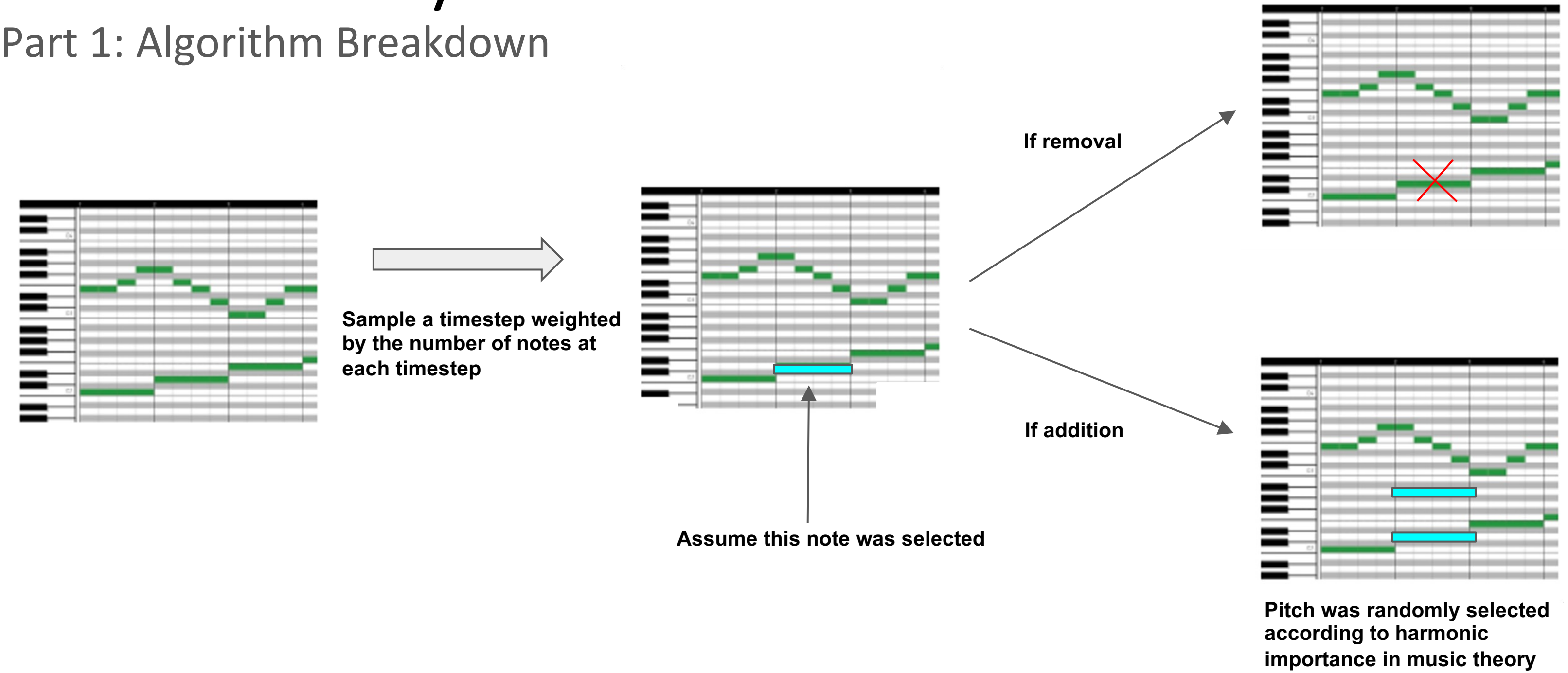


Fig 7. We designed a method that can automatically generate harmonic intervals and automatically remove notes. To add notes, the algorithm requires two steps. First we sample harmonic intervals from a probability distribution computed from aggregating music theory rules used in prior work.

Experimental Design

Part 2: Experimental Design and Results

We run a set of 3 different experiments to evaluate our results:

1. Recovering Chord Information

- To verify that the added or reduced notes do not significantly affect the harmonic structure of music we test if we can recover ground truth chords from the original piano roll (Fig. 3).

2. Music Similarity with Kullback-Leibler Divergence

- To get a sense of the music similarity without using a human analyst, we apply Lerch e.t. multi-criteria evaluation metrics based on probabilistic measures of musical features.

3. Human Evaluations

- In order to evaluate the practical use of this method, we conduct a small survey designed to understand how human listeners, musically trained and untrained, judge reduced/added note transformations.

Recovering Chord Information

Part 2: Results

Recovering Chord Information

To accomplish this, we follow the following procedure

For each method.

- Transform the validation data using note multiples: r ∈ [0.3, 0.5, 0.7, 1, 1.3, 1.5, 1.9]
- Using a separate and isolated Bi-LSTM model trained on the original data, we predict symbolic chords for each note multiple.

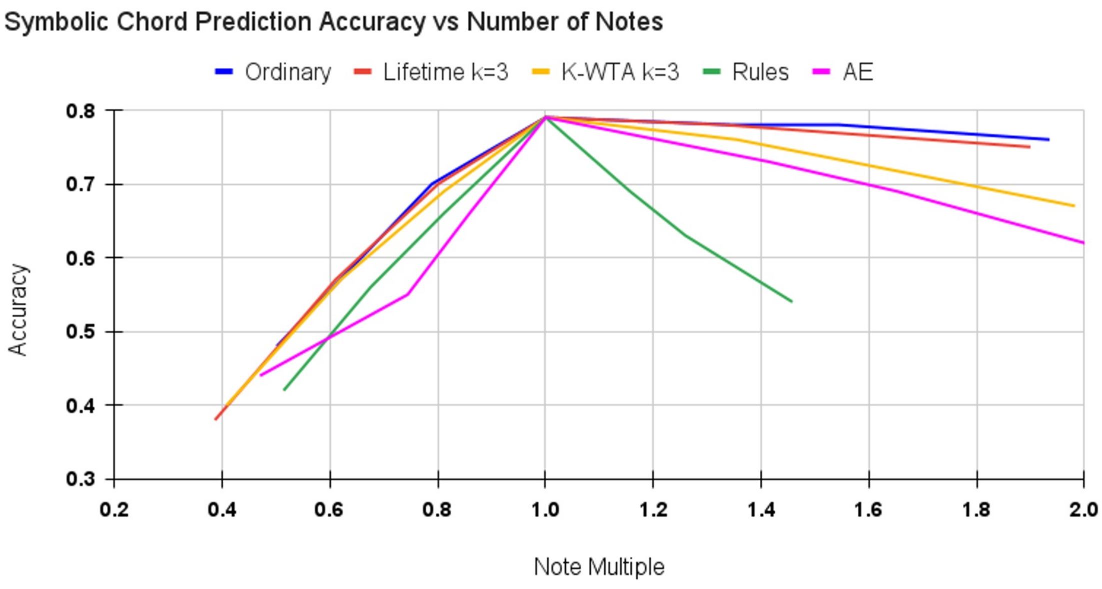


Fig. 8. Symbolic chord prediction accuracy using a Bi-LSTM model trained on the original data as we transform our validation data using VaryNote

Music Similarity with Kullback-Leibler Divergence

Part 2: Results

Music Similarity with Kullback-Leibler Divergence

We compare the original MIDI music datasets against every method with 1.5 × notes by applying kernel density estimation (Gaussian kernel) to find a Probability Density Function (PDF) for the following features:

- **Pitch Count (PC)**: the number of different pitches within a sample
- **Pitch Range (PR)**: the difference of the highest and lowest used pitch
- **Average Pitch Interval (PI)**: the average value of the interval between two consecutive pitches.
- **Average Inter-Onset-Interval (IOI)**: the time between two consecutive notes.

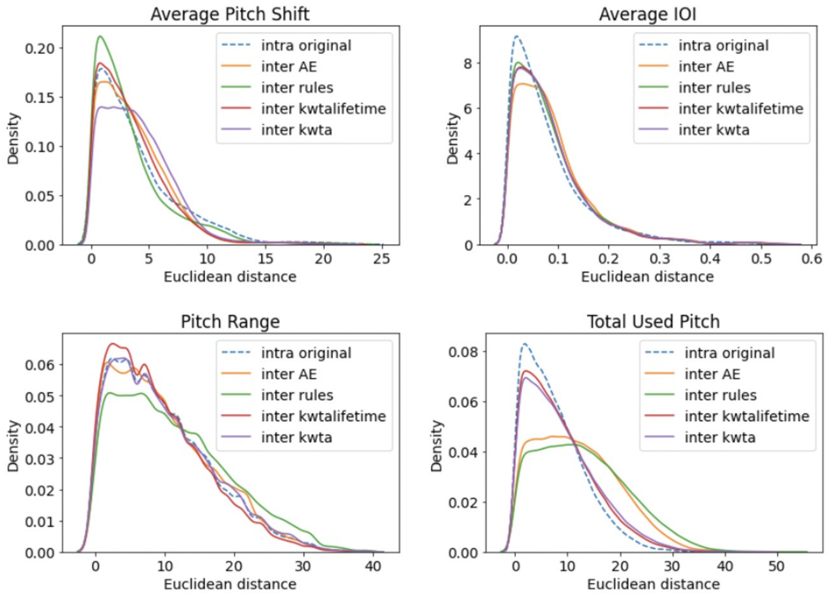


Fig. 9. We extract certain features and use kernel density estimation (Gaussian kernel) to find a probability density function for specific dataset generated by a model. ”Intra” refers to comparisons made within a single group of the original music. ”Inter,” on the other hand, refers to comparisons made between two different groups or categories, in this case comparisons made between the altered music and the original music.

Human Evaluations

Part 2: Results

Human Evaluations

There were 30 total participants; 11/30 participants self-reported knowing how to play an instrument. The survey has three sections.

- **Musical Preference**: the participants are asked to score VaryNote output from 1-5, 1 being the lowest appeal, and 5 being the highest appeal.
- **Perceived Musical Complexity:** the participants are asked to score VaryNote output from 1-5
- **Music Turing Tests (MTT)**: the participants are given two examples, VaryNote output, and the original music and are asked to identify the piece of music that was fully composed by a human
- **MTT - Multi Instrument:** To generate a multi-instrument output we simply isolate the notes from the VaryNote output and synthesize the MIDI with a new instrument.

Table 1. Human Evaluation Results for preference score, and complexity score. The highest mean for each question is shown in bold.

Experiment	Score Report				
	Original	×0.5 Notes	×0.7 Notes	×1.5 Notes	×1.9 Notes
Preference Mean	3.09	2.15	2.73	**3.62**	3.41
Std. Deviation	1.33	1.23	1.23	1.11	1.35
Complexity Mean	3.25	1.62	2.52	**3.92**	3.85
Std. Deviation	1.61	1.21	1.46	1.24	1.32

Table 2. MTT Results, **t**he piece the participant selects as being composed by a human receives a score of 1. We sum the total scores and divide by the total number of participants to get a proportion of times humans select the VaryNote output over the original music. The multi-instrument question uses string and woodwind MIDI instruments.

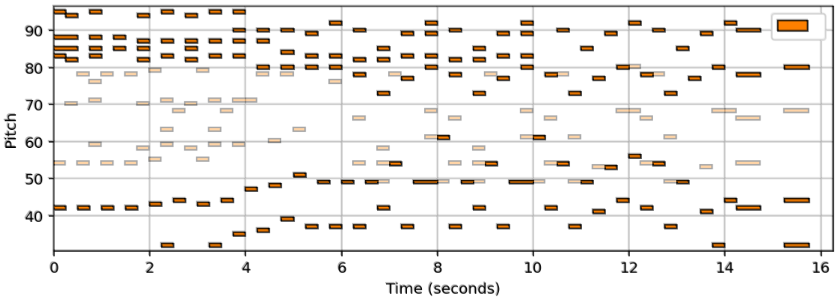
Experiment	×0.5 Notes	×1.5 Notes	×1.9 Notes
Music Turing Test (MTT) - Piano	0.22	**0.36**	0.17
MTT - Multi-Instrument	N/A	**0.57**	N/A

Examples

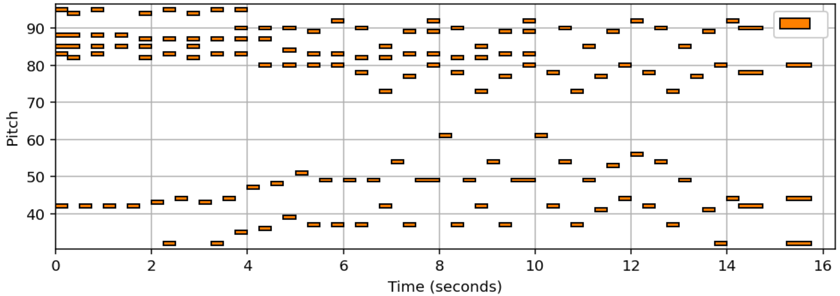
Part 2: Results

Isolated added notes

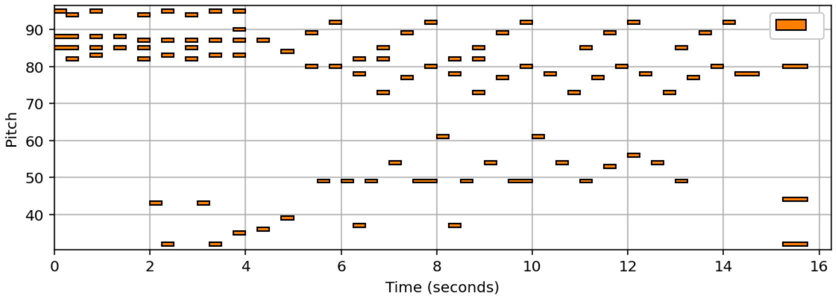
1.5 x Notes



Original



0.5 x Notes

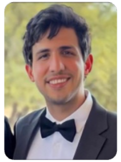


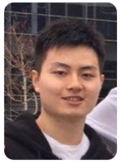
SCAN ME

Thank you !

https://varynote.github.io

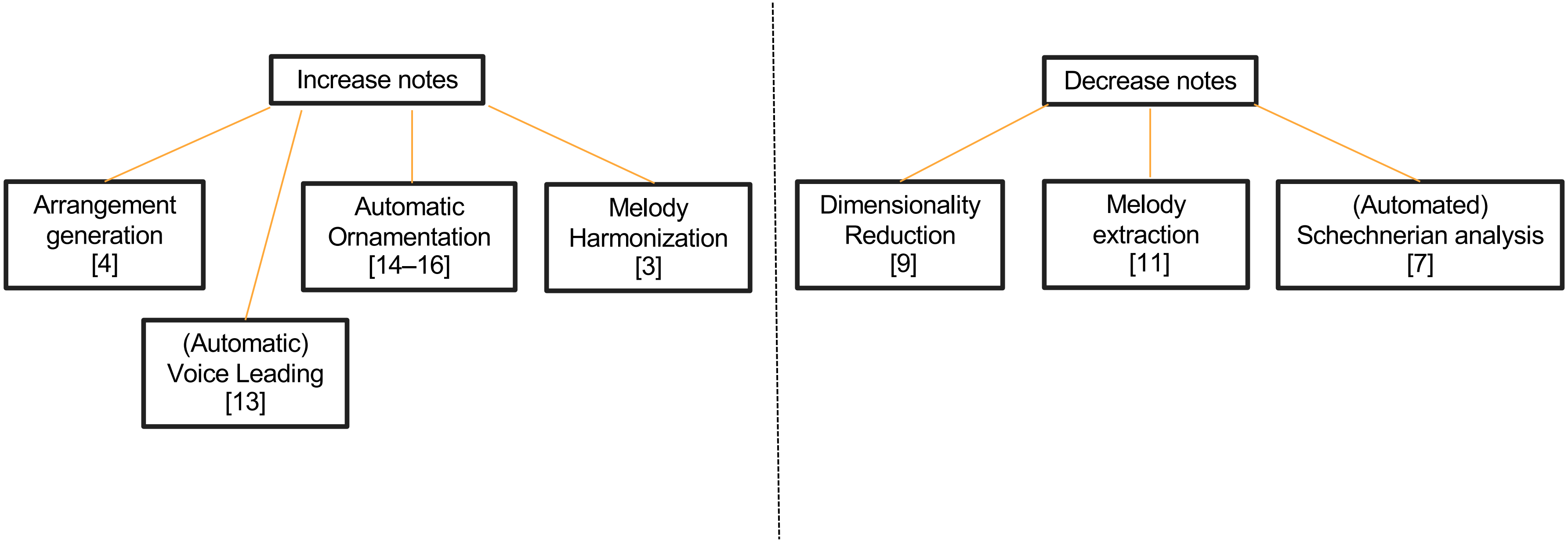
LARG
Learning Agents Research Group
The University of Texas at Austin





Background
Appendix

Previous work on increasing / decreasing note information can be summarized as follows.



**None increase/decrease based on a desired multiple like VaryNote*

Human Survey Results - Statistical Analysis (Preference)

Appendix

Sums of squares, degrees of freedom, mean squares, and F and p-values, given the mean, standard deviation, and number of subjects in each group.

	Number of Subjects	Mean	Standard Deviation
Group 1:	30	3.09	1.33
Group 2:	30	2.15	1.23
Group 3:	30	2.73	1.23
Group 4:	30	3.62	1.11
Group 5:	30	3.41	1.35

	SS	df	MS	F	p
Between:	40.680	4	10.170	6.478	0.000
Within:	227.630	145	1.570		
Total:	268.310	149			

Human Survey Results - Statistical Analysis (Complexity)

Appendix

Sums of squares, degrees of freedom, mean squares, and F and p-values, given the mean, standard deviation, and number of subjects in each group.

	Number of Subjects	Mean	Standard Deviation
Group 1:	30	3.25	1.61
Group 2:	30	1.62	1.21
Group 3:	30	2.52	1.46
Group 4:	30	3.92	1.24
Group 5:	30	3.85	1.32

	SS	df	MS	F	p
Between:	112.832	4	28.208	14.897	0.000
Within:	274.566	145	1.894		
Total:	387.399	149			

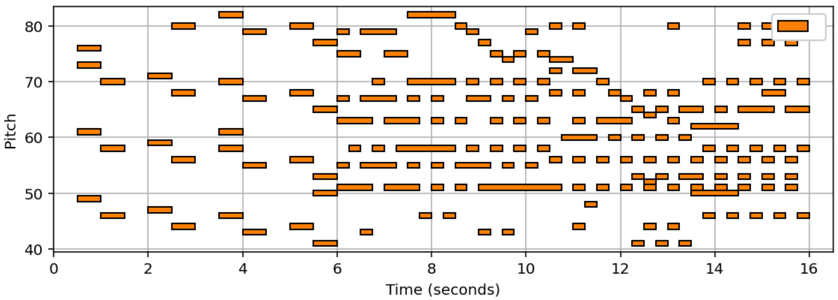
Examples

Part 2: Results

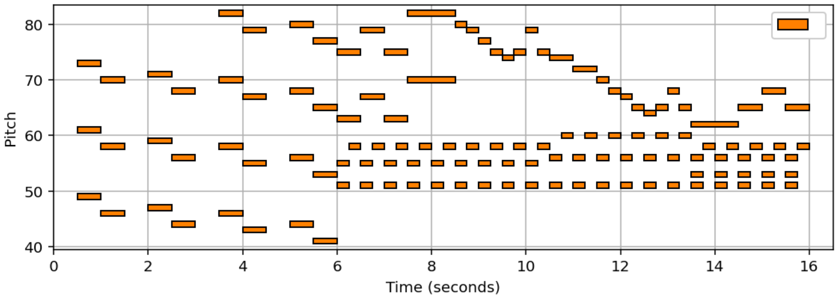
Only Isolated added notes

Added notes as strings

1.5 x Notes



Original



0.5 x Notes