ISMIR 2021 Paper Summary

Junchuan Zhao

June 19, 2022

Abstract

A summary of music generation papers accepted by ISMIR 2021. (Publicly Available)

1 SURPRISENET: MELODY HARMONIZATION CONDITIONING ON USER-CONTROLLED SURPRISE CONTOURS

SupriseNet Traits
(1) relieves trade-off between coherence, which caused by the melody and surprisingness, which caused by the user-supplied condition.
(2) the chords are correponded to the melody and follow the surprise contour.

Transition matrix ($a \in \mathbb{R}^{N \times N}$): encode the distribution of the chord at this time given the previous element.
$p(\text{chord}_t|\text{chord}_{t-1})$

Surprisingness: equivalent to the definition of the information content in information theory.
Surprisingness $= -\log p(\text{chord}_t|\text{chord}_{t-1})$.
The higher the surprise value at a certain time, the gerater the amount of information at that time.

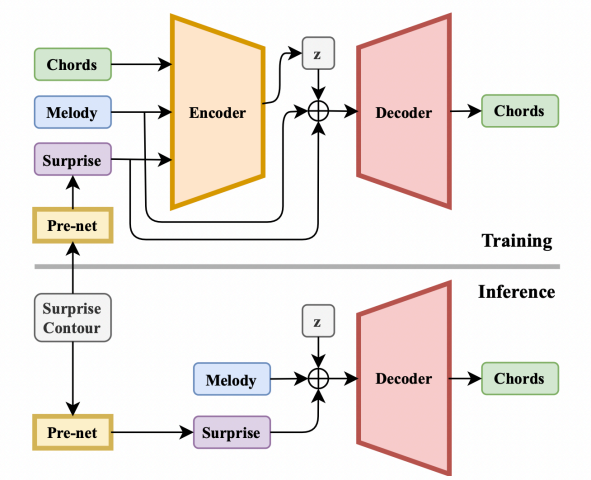

Figure 1: The structure of SURPRISENET.

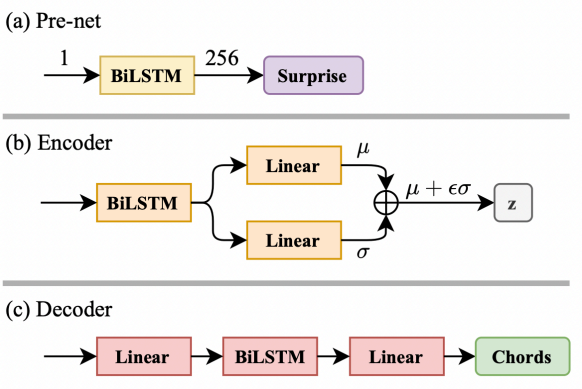

Figure 2: Three main components of SURPRISENET.

Pre-net: extend the feature from a scaler to a 256-dimensional vector.
CVAE: the features of the chord, melody, and extended surprise contour are concatenated and fed into the encoder to generate a latent code z subject to standard normal distribution.

Future work
(1) the meaning of surprise: chord, melody?
(2) the surprise contours, relationship between different surprise contours?
(3) sample the surprise contours that generate melodious music?
(4) surprisingness – style classification.

2 CONTROLLABLE DEEP MELODY GENERATION VIA HIERARCHICAL MUSIC STRUCTURE REPRESENTATION

Problems
(1) model larger scale music structure and multiple levels of repetition
(2) controllability to create desired tempo, styles, and mood
(3) scarcity of training data

1. MusicFrameworks
(1) high-level representation: repeated sections and phrases
(2) low-level representation: rhythm structure and melodic contour
2. Generate melodies from music frameworks
3. Achieve controllability by editing the music frameworks at any level

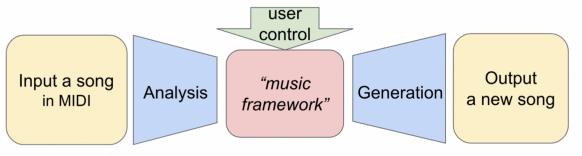

Figure 3: Architecture of MusicFrameworks.

Advantages of Music Frameworks A new song can be generated using the structure from song A, basic melody from song B, and basic rhythm form from song C.

Music Frameworks Analysis

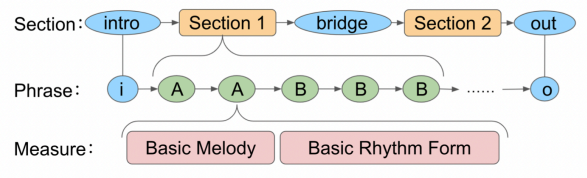

Figure 4: An example music framework.

(1) section and phrase-level structure analysis results
(2) basic melody and basic rhythm form within each phrase

Basic Melody: a sequence of half notes representing the most common pitch in each 2-beat segment of the original phrase

Basic rhythm form: consists of a per-measure descriptor with two components (a. pattern label; b. rhythmic complexity)

Generation Using Music Frameworks
1. user or the library provides the section and phrase structure.
2. generate a basic melody, generate rhythm using the basic rhythm form.
3. generate a new melody given the basic melody.

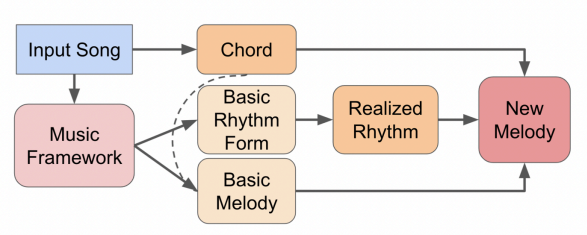

Figure 5: Generation process from music frameworks within each phrase.

Basic Melody Generation
input: $x_i = (\text{pos}_i, c_i, ...)$
sampling with Dynamic Time Warping: melody contour rating function to estimate the contour similarity between two basic melodies.

Realized Rhythm Generation
rhythm pattern: r_i (256 possible rhythm patterns with a 2-beat length)
input: $x_i = (r_{i-1}, \text{brf}_i, \text{pos}_i)$
beam search

Realized Melody Generation
input: x_i includes the current note's duration, the basic melody pitch, the current chord (c_i), three positional features (pos_i).

Network Architecture

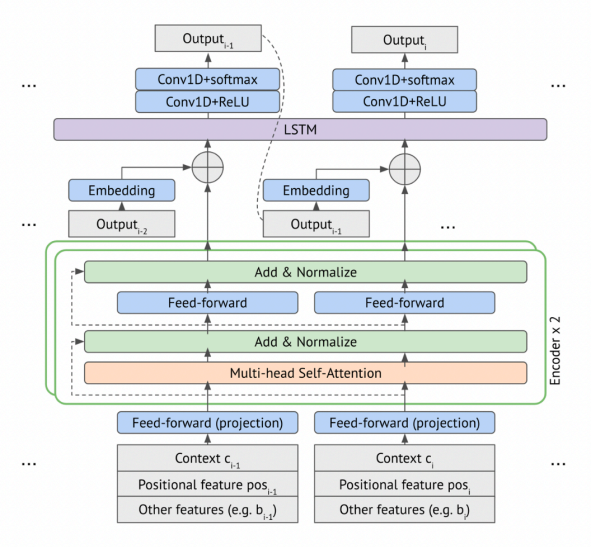

Figure 6: Transformer-LSTM architecture for melody, basic melody and rhythmic pattern generation.

Conclusion
(1) The key idea: adopt an abstract representation – music frameworks (long term repetitive structures; phrase-level basic melodies and basic rhythm forms)
(2) Controllability: manipulation of music frameworks, can be edited and combined to guide compositions.

3 MINGUS: MELODIC IMPROVISATION NEURAL GENERATOR USING SEQ2SEQ

MINGUS relies on two dedicated embedding models (pitch and duration) and exploits in prediction features such as chords (current and following), bass line, position inside the measure.

Data representation
P: $[0-128]$, D: $[0-12]$, C (Chord): $[0-128\times 4]$, NC (Next Chord): $[0-128\times 4]$, B: [0-128], BE: [0-3], O: [0-95]

Model Architecture

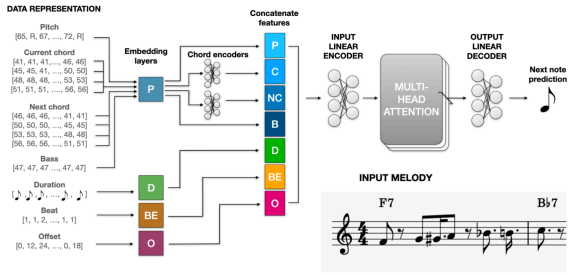

Figure 7: MINGUS model architecture and data representation.

MINGUS: two parallel transformer models with the same structure, predicting pitch and duration, composed by the sole encoder module with a forward mask and a pad mask.
The structure is the same for pitch and duration models.
Pitch model: D, C, B, BE, O
Duration model: B, BE, O

4 SINTRA: LEARNING AN INSPIRATION MODEL FROM A SINGLE MULTI-TRACK MUSIC SEGMENT

SinTra
(1) A novel pitch-group representation
(2) A novel inspiration model
(3) Three modules based on Transformer-XL are designed for each stage of multi-scale training to process multi-track music.

Data Representation
Pitch-group representation

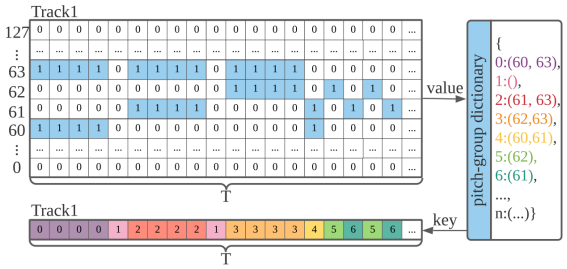

Figure 8: Illustration of the pitch-group representation.

Pyramid of Transformer-XL Model

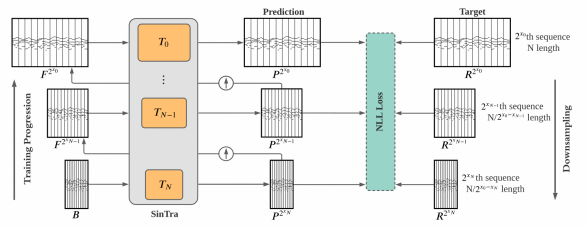

Figure 9: SinTra's multi-scale pipeline.

R: real music $\{R^{2^{x_0}}, ..., R^{2^{x_N}}\}$, $R^{2^{x_n}}$ is a downsampled version of $R^{2^{x_0}}$.
P: produced music P^{2^n} is the corresponding scale of $R^{2^{x_n}}$
The generation of a music sample starts at the coarsest scale and sequentially passes through all models up to the finest scale.

Network Structure

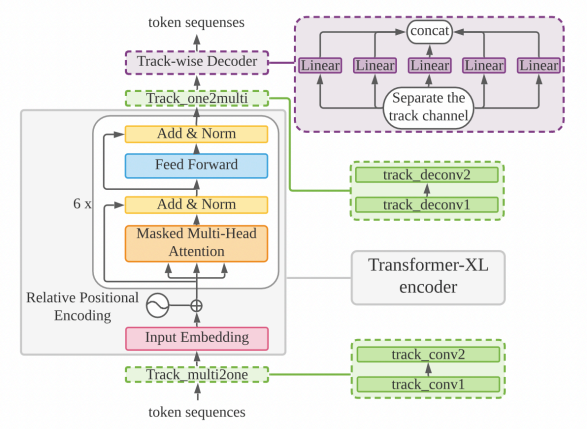

Figure 10: Network structure of each scale T_n.

Training Process

$$
\begin{aligned}
1st: P_{t+1}^4 &= Model_{1^{st}}(R_t^4) \\
loss_{1^{st}} &= NLL(P_{t+1}^4, R_{t+1}^4) \\
2nd: F_{t+1}^8 &= Upsample(P_{t+1}^4) \\
P_{t+1}^8 &= Model_{2^{nd}}(F_{t+1}^8) \\
loss_{2^{nd}} &= NLL(P_{t+1}^8, R_{t+1}^8) \\
3rd: F_{t+1}^{16} &= Upsample(P_{t+1}^8) \\
P_{t+1}^{16} &= Model_{3^{rd}}(F_{t+1}^{16}) \\
loss_{3^{rd}} &= NLL(P_{t+1}^{16}, R_{t+1}^{16})
\end{aligned}
$$

Figure 11: Training process.

5 COLLAGENET: FUSING ARBITRARY MELODY AND ACCOMPANIMENT INTO A COHERENT SONG

CollageNet: given a piece of melody and an irrelevant accompanient with the same length, fuse them into harmonic two-track music. Rhythm and pitch of the melody, and chord and texture of the accompaniment.

Model Architecture

We use the encoders of two VAEs to encode the melody and accompaniment to two latent vectors.

melody: z_{mel}, $z_{mel} = z_p \oplus z_r$[2] (z_p: pitch vector, z_r: rhythm vector), EC^2-VAE.

polyphonic accompaniment: z_{acc}, $z_{acc} = z_c \oplus z_t$ (z_c: chord vector, z_t: texture vector).

Feed z_{mel} and z_{acc} to the actor model G with transforms them into another latent vector pair $\hat{z}_{mel}$ and $\hat{z}_{acc}$. The actor model G is trained under an adversarial framework together with a critic model D.

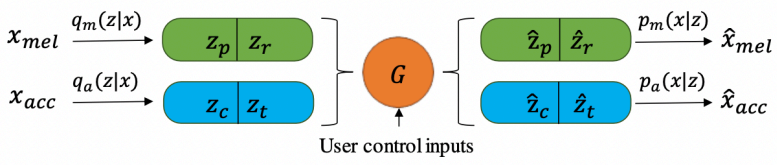

Figure 12: Inference diagram.

Training

1. pre-train the two VAEs for melodies and accompaniments.
2. train the G and D models in the latent space.

harmonic set (Ω_h): $\{x^{(i)}_{mel}, x^{(i)}_{acc}\}_{i=1}^{N}$

disharmonic set (Ω_{dh}): $\{x^{(i)}_{mel}, x^{(j)}_{acc}\}$ $(i \neq j)$

D model is trained to distinguish between positivie samples and negative samples.

positive samples

$\{z_{mel}, z_{acc}\} \sim \Omega^z_h$

negative samples

(1) latent vectors of the disharmonic pairs $\{z_{mel}, z_{acc}\} \sim \Omega^z_{dh}$

(2) latent vectors samples from prior $\{z_{mel}, z_{acc}\} \sim p(z)$

(3) latent vectors produced by the actor model $G(z_{mel}, z_{acc})$

G model is trained to transform disharmonic latent vector pairs into a more harmonic pair.

User Control

The input of the G model is extended with four scalars $c_{mp}, c_{mr}, c_{ac}, c_{at} \in [0, 1]$.

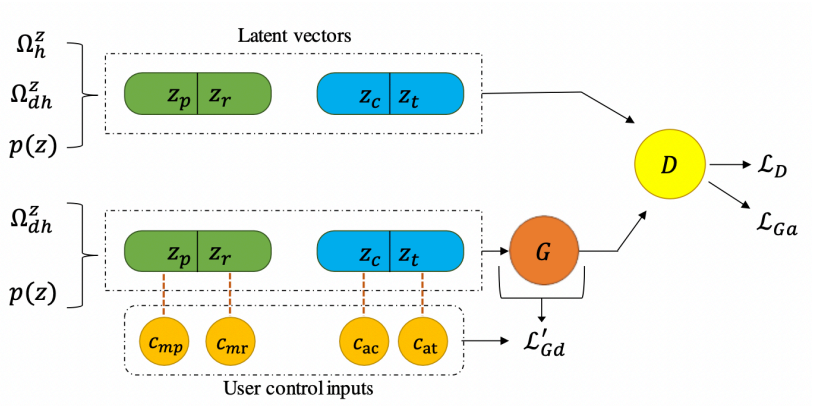

Figure 13: Training diagram.

6 DEEP MUSIC ANALOGY VIA LATENT REPRESENTATION DISENTANGLEMENT

Explicitly-constrained conditional variational autoencoder (EC^2-VAE)
(1) the disentanglement is explicitly coded.
(2) the disentanglement does not sacrifice much of the reconstruction.
(3) model is capable of making analogies in the inference phrase.

Analogy Algorithm
Any computational method capable of producing analogous versions of existing examples.
strict analogy algorithm: requires not only learning the representations but also disentangling them.
music representation learning: build a bi-directional mapping between the distributions of observation x and latent representation z.

Model Architecture

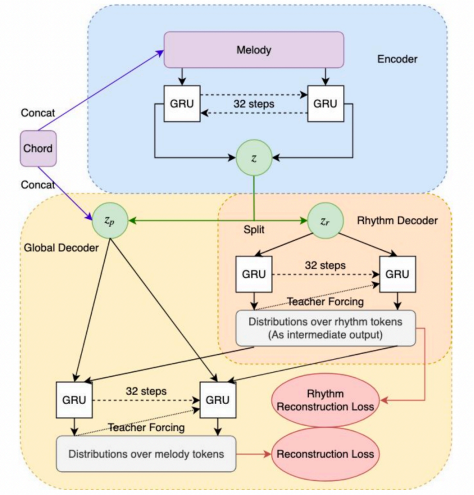

Figure 14: EC^2-VAE model.

To disentangle the latent rhythm representation z_r from the overall z, encourage the intermediate

output of z_r to match the rhythm feature of the melody.
z_p represents the other part of z, which is everything but rhythm.

7 SYMBOLIC MUSIC GENERATION WITH DIFFUSION MODELS

Denoising Diffusion Probabilistic Models (DDPMs)
forward process, reverse process.
Forward process: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$
$\Leftrightarrow x_t = \sqrt{1-\beta_t}x_{t-1} + \beta_t Z_{t-1}$ $(Z_t \sim \mathcal{N}(0, I))$
$q(x_{1:N}|x_0) = \prod_{t=1}^{N} q(x_t|x_{t-1})$
Reverse process: $p_\theta(x_{t-1}; \mu_\theta(x_t, t), \sigma_\theta(x_t, t))$
$p_\theta(x_{0:N}) = p(x_N) \prod_{t=1}^{N} p_\theta(x_{t-1}|x_t)$
Loss function
The training objective: $\max_\theta p_\theta(x_0) = \int p_\theta(x_0, ..., x_N) dx_{1:N}$
$L(\theta) = \mathbb{E}_{x_0,\epsilon,t}[|\epsilon - \epsilon_\theta(\sqrt{1-\beta_t}x_0 + \sqrt{\beta}\epsilon, t)|^2]$

Model

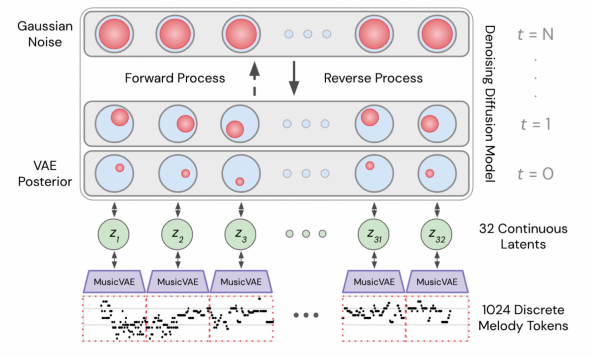

Figure 15: A diagram of the diffusion model.

MusicVAE embeddings
Transformer diffusion model: the network for $\epsilon_\theta(x_t, \sqrt{1-\beta_t})$ is a transformer.

Sampling

Algorithm 1 Sampling

$x_T \sim \mathbb{N}(0, I)$
for $t = N, ..., 1$ **do**
  $\epsilon \sim \mathbb{N}(0, I)$ if $t > 1$, else $\epsilon = 0$
  $x_{t-1} = \frac{1}{\sqrt{1-\beta_t}}(x_t - \frac{\beta}{\sqrt{\beta}}\epsilon_\theta(x_t, t)) + \sigma_t z$
end for
return x_0

Infilling

s represents the partially occluded sample.

Algorithm 2 Infilling

Input: mask m, sample s, N steps, $\beta_1, ..., \beta_N$

$x_N \sim \mathbb{N}(0, I)$

for $t = N, ..., 1$ **do**

$\epsilon_1, \epsilon_2 \sim \mathbb{N}(0, I)$ if $t > 1$, else $\epsilon_1 = \epsilon_2 = 0$

$y = \sqrt{1 - \beta_t} + \sqrt{\beta_t}\epsilon_1$, if $t > 1$, else s

$x_{t-1} = \frac{1}{\sqrt{1-\beta_t}}(x_t - \frac{\beta}{\sqrt{\beta}}\epsilon_z\theta(x_t, t)) + \sigma_t\epsilon_2$

$x_{t-1} = x_{t-1} \odot (1 - m) + y \odot m$

end for

return x_0