How Does it Sound?

Anonymous Author(s)
Affiliation
Address
email

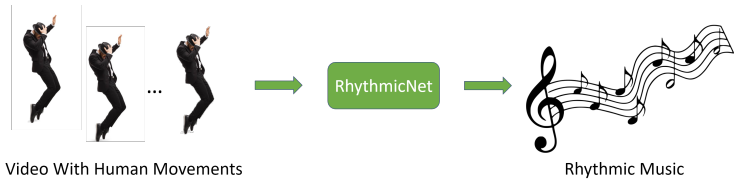

Figure 1: RhythmicNet: Given an input of a silent human movement video, RhythmicNet generates a soundtrack for it. Please see Supplementary Materials for samples of videos and generated soundtracks.

Abstract

One of the primary purposes of video is to capture people and their unique activities. It is often the case that the experience of watching the video can be enhanced by adding a musical soundtrack that is in-sync with the rhythmic features of these activities. How would this soundtrack sound? Such a problem is challenging since little is known about capturing the rhythmic nature of free body movements. In this work, we explore this problem and propose a novel system, called 'RhythmicNet', which takes as an input a video which includes human movements and generates a soundtrack for it. RhythmicNet works directly with human movements by extracting skeleton keypoints and implements a sequence of models which translate the keypoints to rhythmic sounds. RhythmicNet follows the natural process of music improvisation which includes the prescription of streams of the beat, the rhythm and the melody. In particular, RhythmicNet first infers the music beat and the style pattern from body keypoints per each frame to produce rhythm. Next, it implements a transformer-based model to generate the hits of drum instruments and implements a U-net based model to generate the velocity and the offsets of the instruments. Additional types of instruments are added to the soundtrack by further conditioning on the generated drum sounds. We evaluate RhythmicNet on large scale datasets of videos that include body movements with inherit sound association, such as dance, as well as 'in the wild' internet videos of various movements and actions. We show that the method can generate plausible music that aligns well with different types of human movements.

1 Introduction

Rhythmic sounds are everywhere, from raindrops falling on surfaces to birds chirping, to even machines capable to generate unique sound patterns. When sounds accompany visual scenes, they enhance the perception of the scene by complementing it with additional cues such as semantic association of events, means of communication, drawing attention to parts of the scene, and many more. For visual scenes that include activity of people, rhythmical music that is in-sync with the

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

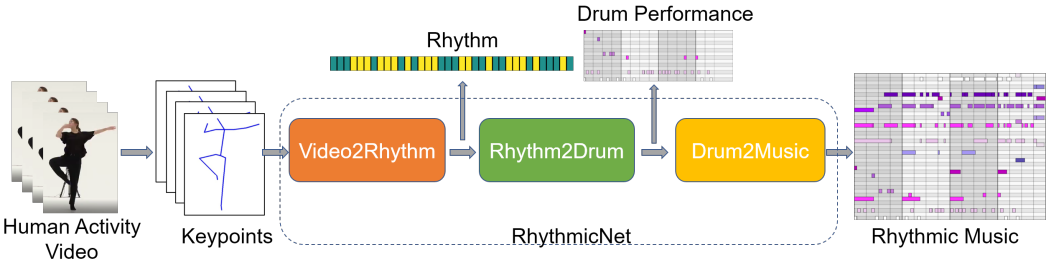

Figure 2: System Overview of RhythmicNet. Keypoints are extracted from human activity video and are processed through Video2Rhythm stage to generate the rhythm. Afterwards Rhythm2Drum converts the rhythm to drum performance. In the last step, Drum2Music component adds additional instrument tracks on top of the drum track.

rhythm of body movements can emphasize the actions of the person and enhance the perception of the activity. Furthermore, since rhythm is the foundation of music, association of music with human activity in a video often enhances the experience of watching the video. Indeed, it is a usual practice that a musical soundtrack is manually chosen in professionally edited videos.

Drum instruments serve as the fundamental part in music by generating the underlying leading rhythm patterns. While drum instruments vary in shape, form, and mechanics, their main purpose is to set the rhythm essential to any music. Indeed, drums are known to have existed from around 6000 BC, but even beforehand there were instruments based on principle of hitting two objects and generating sounds. On top of drum patterns, additional instruments add secondary patterns and melody, eventually creating rich multifaceted music. It is also the case in modern music composition and improvisation, where composers would start a new musical piece by designing the rhythm by prescribing the corresponding drum track. As the piece evolves, additional tracks of accompanying instruments are gradually superimposed on top of the drum track to produce the final music.

Inspired by the possibility of automatically associating rhythmic soundtracks to videos, we explore in this work the generation of rhythmic music correlated with human body movements. We follow similar music composition and improvisation steps as in music improvisation by first generating the rhythm of the music that is strongly correlated with the beat and patterns of the movements. Such rhythm can then be used to generate novel drums music that accompanies the body movements. With the rhythm being inferred, we follow the next steps of music improvisation and add new instruments (piano and guitar) tracks to enrich the music. In summary, we address the challenge of generating a rhythmic soundtrack for a human movement video by proposing a novel pipeline named 'RhythmicNet', which translates human movements from the domain of video to rhythmic music with three sequential components: Video2Rhythm, Rhythm2Drum, and Drum2Music.

In the first component of RhythmicNet, given a human movement video, we extract the keypoints from the video and use a spatio-temporal graph convolutional network [1] in conjunction with transformer encoder [2] to capture motion features and estimate music beats. Since music beats are periodic and there are various visual changes occurring in human movements, we propose an additional stream to the beats, called style stream, which captures fast movements and combine it with the music beats stream to construct the rhythm that will be used for music generation. The Rhythm2Drum includes an encoder-decoder transformer that given the rhythm, it generates the hits of drum performance and a U-net [3] to generate its velocities and offsets. We find that these two components are critical for generation of satisfactory drum music. In the last stage, the Drum2Music component, we adopt an encoder-decoder architecture using transformer-XL [4] to generate a music track of either piano or guitar conditioning on the generated drum performance to complete the music. An overview of the RhythmicNet is shown in Fig. 2. Our main contributions are: (i) To the best of our knowledge, we are the first to generate a novel musical soundtrack that is in-sync with human activities. (ii) We introduce an entire pipeline, named 'RhythmicNet', implementing three stages to complete the transformation. (iii) RhythmicNet is robust and generalizable, and we perform experiments that show the output music generated by RhythmicNet will be consistent with the human body movements in videos in large-scale dance dataset and 'in the wild' internet videos.

2 Related Work

Generation of sound for a video is a challenging problem since it aims to relate indirectly correlated signals. It belongs to the class of problems of *Audio-visual learning*, which deals with exploration and leveraging of the correlation of both audio and video for tasks such as audio-visual correspondence task [5, 6, 7, 8], video sound separation task [9, 10, 11, 12], audio-visual event localization task [13], transformations of audio to body movements [14, 15, 16], lips movements [17] and talking faces [18, 19, 20].

Several approaches were proposed for the relation of sounds for a video. A deep learning approach showed the potential of such application by proposing a recurrent neural network to predict the audio features of impact sounds from videos. The approach was able to produce a waveform from these features [21]. In a subsequent work, a conditional generative adversarial network was proposed to achieve cross-modal audio-visual generation of musical performances [22]. In both methods, single image is used as an input, and the network performs supervision on instrument classes to generate a low-resolution spectrogram. Concurrently, for natural sounds, a Sample RNN-based method [23] has been introduced to generate sounds such as baby crying, water flowing, given a visual scene. This approach was enhanced by an audio forwarding regularizer that considers the real sound as an input and outputs bottle-necked sound features provided stronger supervision for natural sound predictions only from visual features [24].

Compared to natural sounds with relatively simple structures, music usually contains more complex elements. For the generation of music, an interactive background music synthesis algorithm guided by visual content was introduced to synthesize dynamic background music for different scenarios [25]. The method, however, relies on reference music retrieval and cannot generate new music directly. Direct music generation approaches have been developed for videos capturing a musician playing an instrument. A ResNet-based method was proposed to predict the pitch and the onsets events, given video frames of top-view videos of pianists playing the piano [26]. Later, Audeo [27] demonstrated the possibility to transcribe video to high-quality music. While the results of such methods are promising, the generation is limited to a single instrument. Thereby, Foley Music [28] proposed a Graph-Transformer network to generate Midi events from body keypoints and achieved convincing synthesized music from Midi, and Multi-instrumentalist Net [29] showed the generation of music waveform of different instruments in an unsupervised way. While these approaches show the possibilities of generating music from videos, the videos have to contain solid visual cues such as instruments to indicate the types of music being generated. It is unclear whether it is still possible to generate music when such visual cues do not exist. Recently, a novel approach of dance beat tracking was proposed [30], aimed at detecting musical beats from a dance video by using its visual information only. Inspired by this work, we design methods to estimate the musical beats from movements more precisely and utilize the beats to improvise new music.

For music generation, symbolic representation, which uses the Musical Instrument Digital Interface (Midi), has been shown to be useful in modeling and generating music. Earlier works converted Midi into piano-roll representation and used generative adversarial networks [31] or variational autoencoder [32, 33] to generate new music. A limitation of piano-roll is that it may result in memory inefficiency when the length of music is too long. In order to address this limitation, event-based representation was proposed and was shown to be a useful and efficient representation in modeling music [34, 35, 36]. While the event-based representation enabled models to obtain convincing generated results, it lacks metrical structure, leading to unsteady beats in the generated samples. Thereby recently, a new representation called Remi was proposed to impose a metrical structure in the input data so that the models can include awareness of the beat-bar-phrase hierarchical structure in music [37]. In our work, we convert the Midi into the Remi representation in the Drum2Music stage. While the methods mentioned above generate unconditional music, music generation has also progressed with constraints. For example, it was proposed to constrain generative models to sample for predefined attributes [38]. Systems such as Jukebox [39] and MuseNet [40] showed the possibility of generating music based on user preferences, which correspond to network model specifically trained with labeled tokens as a conditioning input. Furthermore, a Transformer autoencoder has been proposed to aggregate encoding of Midi data across time to obtain a global representation of style from a given performance. Such a global representation can be used to control the style of the music [41]. In the Drum2music stage, we consider the drum track as the conditioning input and aim to generate the track of another musical instrument such as piano or guitar.

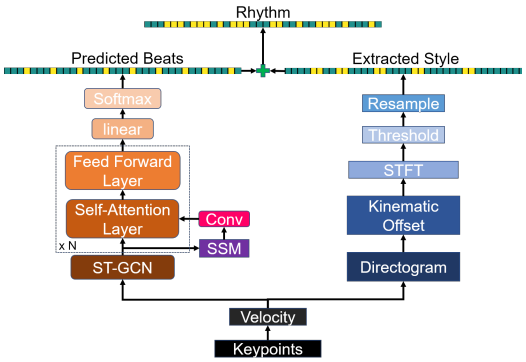

Figure 3: Detailed schematics of the components in the Video2Rhythm stage.

3 Methods

RhythmicNet includes three sequential components: 1) Rhythm association for human movements (Video2Rhythm), 2) Generation of drum from rhythm (Rhythm2Drum), 3) Addition of instruments to the drum track (Drum2Music). We describe the details of each stage below.

Video2Rhythm. We decompose the rhythm into beats and style streams. We propose a novel model to predict music beats and a classical approach to extract style patterns from human movements.

Music Beats Prediction. To predict regular music beats from human body movements, we extract 2D skeleton keypoints via the OpenPose framework [42] and perform the first-order difference to obtain the velocity for each video. Motion sequences can be considered as three dimensional tensor $X \in \mathbb{R}^{V \times T \times 2}$ where V is the number of keypoints, T is the number of frames, and the last dimension indicates the 2D coordinates. We formulate the prediction of music beats as a temporal binary classification problem. Given the skeleton keypoints X, we aim to generate the output with the same length $Y \in \mathbb{R}^T$, where each frame is classified into 'beat' ($y = 1$) or 'non-beat' ($y = 0$). We encode the keypoints using a spatio-temporal graph convolutional neural network (ST-GCN) [1]. Such encoding represents the skeleton sequence as an undirected graph $G = (V, E)$, where each node $v_i \in V$ corresponds to a key point of the human body and edges reflect the natural connectivity of body keypoints. The sequence passes through a spatial GCN to obtain the features at each frame independently, and then a temporal convolution is applied to the features to aggregate the temporal cues. The encoded motion features are then represented as $P = AXW_SW_T \in \mathbb{R}^{V \times T_v \times C_v}$, where X is the input, $A \in \mathbb{R}^{V \times V}$ is the adjacency of matrix of the graph defined based on the keypoints connections of the human body, W_S and W_T are the weight matrices of spatial graph convolution and temporal convolution respectively, and T_v and C_v indicate the number of temporal dimension and feature channels. We obtain the final motion features $P \in \mathbb{R}^{T_v \times C_v}$ by averaging the node features. Given the motion feature P, we use a transformer encoder that contains a stack of multi-head self-attention layers to learn the correlation between different frames. Due to the periodicity of music beats, we introduce two components to allow the model to capture the music beats more accurately: 1) Instead of using common positional sinusoids to represent timing information, we adopt a relative position encoding [43] to allow attention to explicitly resolve the distance between two tokens in a sequence, which is critical for modeling the timing in music where relative differences arguably matter more than their absolute values [35]. 2) We use temporal self-similarity matrix (SSM) of motion features, which has been shown effective in human action recognition [44, 45] in regularization of the transformer. The SSM S can be constructed by computing all pair wise similarities $S_{ij} = f(P_i, P_j)$ between pairs of frame-level motion features P_i and P_j, where $f(\cdot)$ is the similarity function. We use the negative of the squared euclidean distance as the similarity function, $f(a, b) = -||a - b||^2$, followed by taking softmax over the time axis. As the SSM has only one channel, it is passed to a convolution layer to get $\hat{S} = \text{Conv}(S)$ before adding to every attention head. The complete self-attention mechanism is as follow: $\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T + \hat{S} + R}{\sqrt{D_k}})V$, where Q, K, V are the standard query, key and value respectively, and R is the ordered relative position encoding for each possible pairwise distance among pairs of query and key on each head. We train the model using weighted binary cross-entropy loss that puts more weights on the beat category to address label

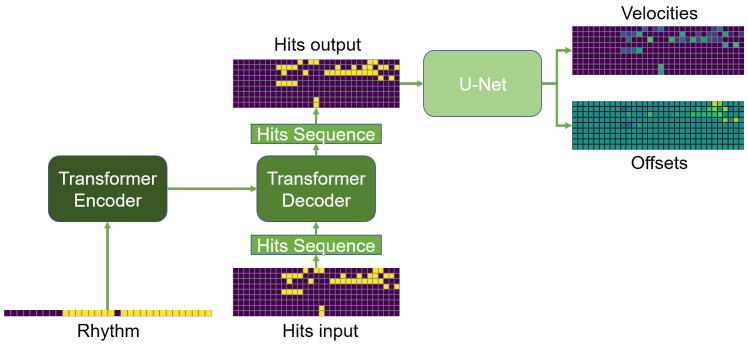

Figure 4: Detailed schematics of the components in the Rhythm2Drum stage.

imbalance.

Compared to previous work [30], the combination of graph representation, relative self-attention and SSM components enables the model to better capture the spatial-temporal structures in body dynamics, allowing for more accurate beat estimation. The output of the network is the beat activation function; i.e., for each video frame, the model predicts its probability of being a 'beat' frame. To obtain beat positions, we apply an algorithm based on HMM decoding proposed in [46].

Style Extraction. Beats represent the monotonic periodic pattern occurring at every fixed time interval. Between two music beats, there are typically various irregular movements that could be captured into a rhythm. However, since these patterns are inconsistent, it is unclear how to extract such style patterns from visual information alone. We, therefore, propose to capture rapid movements and call this stream style. We infer changing patterns of motion strength by kinematic offsets [47] & [48]. To obtain kinematic offsets, we calculate a motion directogram by factoring the motion into different angles. Given $F_t(j,t)$ as the velocity magnitude of joint j at time t, we could formulate the directogram $D(t,\theta)$ as:

$$D(t,\theta) = \sum_j F_t(j,t)\mathbb{1}_\theta(\angle F_t(j,t)), \text{ where } \mathbb{1}_\theta(\phi) = \begin{cases} 1 & |\theta - \phi| \leq 2\pi/N_{\text{bins}} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The indicator function $\mathbb{1}_\theta(\phi)$ is used to distribute the motion of all joints into N_{bins} angular intervals. Then the first-order difference of the directogram is calculated to obtain the acceleration of motion across different angles. The mean acceleration in the positive direction only is extracted as kinematic offset which measures motion strength (i.e., the larger the value, the more remarkable in motion strength). Once the kinematic offsets are obtained, we detect fast changes in the motion strength within fixed intervals by taking the Short-Time-Fourier-Transform (STFT). We use the highest frequency bin to extract possible style patterns from motion. We choose frames with the top 10% frequency magnitude as the style pattern to music beats for every second. The output signal is re-sampled to have the same sampling rate as the music beats. Combining both music beats and style patterns obtained from the above process, we get the rhythm that captures the tempo of the music and is consistent with body movements.

Rhythm2Drum. The rhythm is a binary sequence that indicates where it should sound. The stage of Rhythm2Drum introduces drum performance that can fully interpret the provided rhythm. Following the GrooveVAE setup [33], each drum performance can be represented by three matrices: hits, velocities, and offsets. The hits represent the presence of drum onsets and is a binary matrix $H \in \mathbb{R}^{N\times T}$, where N is the number of drum instruments and T is the number of time steps (one per 16th note). The velocities V is a continuous matrix to reflect how hard drums are struck with ranges between $[0,1]$. The offsets O is also continuous and stores the timing offsets, taking values in $[-0.5, 0.5)$ that indicates how far and in which direction each note's timing lies relative to the nearest 16th note. Both V and O have the same shape as H. Given the input rhythm sequence $Y \in \mathbb{R}^{1\times T}$, we aim to generate the H, V and O. Unlike the GrooveVAE [33], which models all three matrices simultaneously with multiple losses, we model the H, V, and O smoothly in two steps using the combination of an encoder-decoder transformer [2] and a Unet [3]. In the first step, the binary rhythm sequence is passed as input to transformer encoder. In decoder, we convert the H matrix into word sequence defined by a small vocabulary set of all possible combinations of hits,

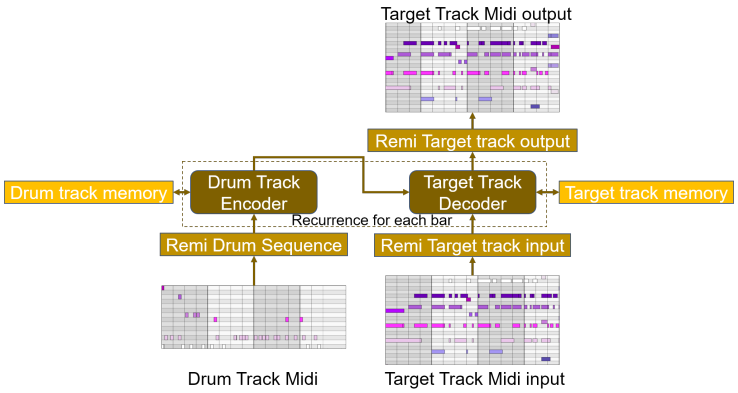

Figure 5: Detailed schematics of the components in the Drum2Music stage.

and map back to binary matrix in the final output. We observe that by learning the hits H as word sequence autoregressively, the transformer can generate more natural and diverse drum onsets. We train the transformer using cross-entropy loss. In the second step, we add style patterns (velocity and offsets) to the onsets. Since the hits H has the same shape as the V and O, we can consider it as a transformation between 2 images of same shape. To achieve such transformation, we adopt a U-net [3] to take onset matrix H as input and generate V and O. We use Mean-Square Error (MSE) loss to optimize the U-net. Finally, given the generated matrices H, V and O, we convert them to the Midi representation to produce drum track.

Drum2Music. Drum performance contains rhythmic music, however, it lacks pitches and melody that play dominant roles in music. Therefore, we propose to add further instruments tracks conditioning on the generated drum track. We propose an encoder-decoder architecture for this task, with the encoder taking the drum track as input, and the decoder generating the track of another instrument. We consider the piano or guitar as the other instrument, as these are dominant instruments in modern music. We use the Remi representation [37] to represent multitrack music, since it has been shown to be effective in composing music with better rhythmic structure. Compared to the commonly-used Midi-like event representation [35], the Remi representation includes information such as Tempo changes, Chord, Position, and Bar, which allow our model to learn the dependency of note events occurring at the same position across bars. For both the encoder and decoder, we adopt the transformer-XL network model [4] which extends the transformer by including the recurrence mechanism. The recurrence mechanism enables the model to leverage the information of past tokens beyond the current training segment, thereby looking further into the history. In the training process, we split the music piece into segments with a total number of bars. In the encoder, for recurrent step i, we provide the i-th bar of drum performance x_i to the transformer-XL.

The encoder contains stack of multi-head self-attention layers. Its output E_i can be represented as: $E_i = \text{Enc}(x_i, M_i^E)$, where M_i^E is the encoder memory used for the i-th bar input and the encoder hidden state sequence computed in previous recurrent steps. Similarly, in the decoder, the prediction of j-th token of the i-th bar $y_{i,j}$ is formulated as $y_{i,j} = \text{Dec}(y_{i,t<j}, M_i^D, E_i)$, where $y_{i,t<j}$ are the previously generated tokens in the same bar, M_i^D is the decoder memory used for i-th bar, and E_i is the corresponding encoder output of the same bar. The decoder consists of a stack of layers with casual self-attention, cross-attention to the encoder output, and feed-forward network. We adopt a teacher forcing strategy in the training process and feed the ground truth tokens into the decoder to generate the next tokens. We minimize the negative log-likelihood (NLL) between generated tokens and ground truth tokens to optimize the model. In the inference process, we provide the drum track to the encoder for each bar and generate the tokens in the decoder one by one. Finally, we use the temperature-controlled stochastic top-k sampling method [49] to randomly generate a new music track.

4 Experiments & Results

Datasets. We use the AIST Dance Video Database [50], a large-scale collection of dance videos in 60 fps. This database includes 10 street dance genres, 35 dancers, 9 camera viewpoints, and 60 musical

Models\Metrics	CML_c (%)	CML_t (%)	Cem (%)	F (%)
TCN [30]	44.97	45.15	48.14	63.04
TF	16.07	16.24	32.85	46.90
ST-GCN	54.89	55.45	49.23	64.78
ST-GCN+TF	61.89	62.34	55.09	71.93
ST-GCN+TF+SSM	63.20	63.58	57.72	73.07
ST-GCN+TF+RelAttn	68.01	68.31	59.19	74.67
ST-GCN+TF+SSM+RelAttn	**71.43** (+26.46%)	**71.94** (+26.79%)	**61.59** (+13.45%)	**75.79** (+12.75%)

Table 1: Evaluation on music beat prediction. The abbreviation of each component: TF (transformer), ST-GCN (spatio-temporal graph convolutional network), SSM (Self-similarity Matrix), RelAttn (Relative Attention); F (F-score measure), Cem (Cemgil's score), CML_c (Correct metrical level continuous accuracy), CML_t (Correct metrical level total accuracy). Bold texts indicate the best values.

Model\Metric (lower better)	NDB	MSE Velocity	MSE Offsets
GrooveVAE [33]	46	0.0437	0.0402
TF multi-outputs w.o. hits sequence	44	0.0507	0.0348
TF multi-outputs w. hits sequence	39	0.0493	0.0369
TF w. hits sequence + Unet	**39** (↓15%)	**0.0267** (↓40%)	**0.0169** (↓58%)

Table 2: Evaluation on Rhythm2Drum performance. Bold texts indicate the best values. TF (encoder-decoder transformer), Multi-outputs (Predict the hits, velocities and offsets simultaneously), w./w.o. hits sequence (whether using word tokens to represent the hits).

pieces covering 12 types of tempi. For each genre, we use the 1080 basic dance videos, resulting $10,800$ videos in total used for training and testing of Video2Rhythm. We split the train/validate/test sets by 0.8/0.1/0.1 based on the dance genres, dancers, and camera ids. For Rhythm2Drum, we use the Groove Midi dataset [33] which contains 1150 Midi files and over 22,000 measures of drumming. We split the data with 0.8/0.1/0.1 split into the train/validate/test sets. For Drum2Music, we extract two subsets of Lakh Midi dataset [51] to train Drum2Piano and Drum2Guitar models separately. For Drum2Piano, we select the Midi files that contain both tracks of drums and acoustic piano with at least 16 bars, and we consider 16 bars to be one segment. This results in train/validate/test sets with 34991/1944/1944 segments. For drum2guitar, we perform a similar selection to obtain train/validate/test sets with 12904/717/717 segments.

Implementation Details. We use Pytorch [52] to implement all models in RhymicNet with two Titan X GPUs. For all videos, we extract 17 keypoints of body joints. In Video2Rhythm, the network contains a 10-layer ST-GCN and a 2-layer transformer encoder with 2-head attention. For the style extraction part, the motion sequence is down-sampled to 15 fps to calculate kinematic offsets. The number of bins used for the directogram is 12, and a 16-point FFT with hop-size 4 is applied to extract candidate styles. Each detected style is repeated 4 times to match the hop size, and then is up-sampled to original time resolution of 60 fps and re-indexed in unit of quarter note based on estimated tempo. In Rhythm2Drum, the Hits transformer includes 3 layers, 4 heads in both encoder and decoder. The vocabulary size of the decoder input is 152, consisting of all possible combinations of 9 types of drum hits in the dataset. The U-net for generation of velocity and offsets contains 4 down-sample blocks with channel size of 16, 32, 64, 128. In Drum2Music, the model consists of a recurrent transformer encoder and a recurrent transformer decoder. We set the number of encoder layers, decoder layers, encoder heads and decoder heads to 4, 8, 8, and 8 respectively. The length of training input tokens and the length of the memory are 256. We provide additional configuration details in supplementary materials and will share the Pytorch implementation of all components.

Video2Rhythm Evaluation. Following the rubrics proposed for musical beat tracking [53], we report the performance in terms of F-score measure, Cemgil's score (Cem), and continuity base scores: Correct Metrical Level continuity required/not required ($CML_{c/t}$) . For comparison, we implement the baseline model [30] based on a temporal convolutional network (TCN). The comparison and ablation results are shown in Table 1. Our best method significantly outperforms the baseline model in all metrics by a large margin. In particular, the continuity scores outperform the baseline model by more than 25%, indicating the estimated beat sequence is significantly more consistent.

Rhythm2Drum Evaluation. We use several metrics to evaluate Rhythm2Drum. To measure the diversity of the generated drum hits, we adopt the Number of Statistically-Different Bins (NDB)

Metrics	PC/bar	PI	IOI	PCH ↑	NLH ↑	NLL ↓
Dataset (Piano)	5.48	6.16	0.31	-	-	-
Drum2Piano w.o. memory	7.17	4.63	0.12	0.63	0.52	0.77
Drum2Piano	**6.82**	**5.86**	**0.14**	**0.63**	**0.54**	**0.53**
Dataset (Guitar)	5.33	5.51	0.22	-	-	-
Drum2Guitar w.o. memory	3.54	8.94	0.52	0.56	0.46	0.58
Drum2Guitar	**5.63**	**5.69**	**0.13**	**0.64**	**0.51**	**0.40**

Table 3: Drum2Music evaluation. For PC/bar, PI, IOI values, the closer to the dataset the better. For PCH and NLH values, the larger, the better.

metric proposed and used in [54, 55, 28]. We first cluster all training examples into $k = 50$ Voronoi cells by k-means. The generated examples are then assigned to the nearest cell. NDB is reported as the number of cells where the number of training examples is statistically significantly different from the number of generated examples by a two-sample Binomial test. For each model, we generate 9000 samples from the testing set and perform the comparison. For velocities and offsets, we report the Mean-Squared Error (MSE) in the test set. We compare our methods with the baseline model GrooveVAE [33]. The results are shown in Table 2. The results show that using hits sequence to predict the drum track enable more diverse samples such that the Unet component which generates velocities and offsets can generate more realistic drumming.

Drum2Music Evaluation. To evaluate the generated piano or guitar tracks, we use objective metrics suggested in [56]. For PC/bar (pitch count per bar), PI (average pitch interval), IOI (average inter-onset interval), we compare the statistics calculated on the test dataset and on the generated music. For PCH (pitch class histogram) and NLH (note length histogram), we calculate the overlapping area (OA) between the statistics on the test dataset and the generated music for each sample and report the average of them. We further compare the NLL loss in the validation set. The numerical results are shown in Table 3. We compare with the model without using memory to show the effectiveness of the recurrence mechanism. Our results show that for both Drum2Piano and Drum2Guitar using recurrent encoder-decoder transformer, the NLL loss is lower in the validation set and the statistics of the generated samples are much closer to the test dataset compared to the no memory counterparts.

In the wild Experiments and Qualitative Evaluation. In Fig. 6, we show a set of examples of generated soundtracks for video clips in AIST dataset and 'in the wild' clips downloaded from YouTube. For the AIST dataset, we generate and compare predicted beats track with the ground truth beats track. The predicted and GT tracks appear to be in close agreement. We then demonstrate the extracted style and its correspondence to frames which exhibit special movements. With the beat and style tracks we are able to generate the rhythm from which the waveform of drum music is generated. To test and demonstrate the generality of RhythmicNet we apply it to videos clips of various human activity. Examples of such activities are shown in Fig. 6 and include Ice Skating, Playing Football and Performing a Workout. We show additional examples and videos clips along with the soundtracks in Supplementary Materials. The generated rhythms are well synchronized with the videos and the drum track appears to be in-sync with the activities. We also demonstrate the effect of additional instruments (guitar and/or piano) and observe a richer music that accompanies the movements well. To evaluate the the effectiveness of generated soundtracks to perceptually follow the activities we also performed human perceptual tests through Amazon Mechanical Turk (See more details in supplementary materials) which demonstrate that the generated soundtrack music perceptually aligns more favorably with the activity than random soundtrack control.

5 Conclusion and Discussion

In this exploratory work we consider for the first time as far as we know the possibility to automatically generate novel rhythmic soundtracks consistent with human body movements captured in a video. Our results show that the RhythmicNet pipeline that we introduce is able to achieve this creative task. At its core, RhythmicNet defines and implements a systemic approach to automatically generate soundtracks by following the process of music improvisation in which a rhythm of movements is established and translated to drumming music with potentially additional accompanying instruments. We foresee future potential applications in video creation and editing which RhythmicNet can pave the way to unlock. In terms of limitations and future enhancements, the current setup of RhythmicNet allows the addition of a single instrument (piano or guitar) to the rhythmic drum track. Novel

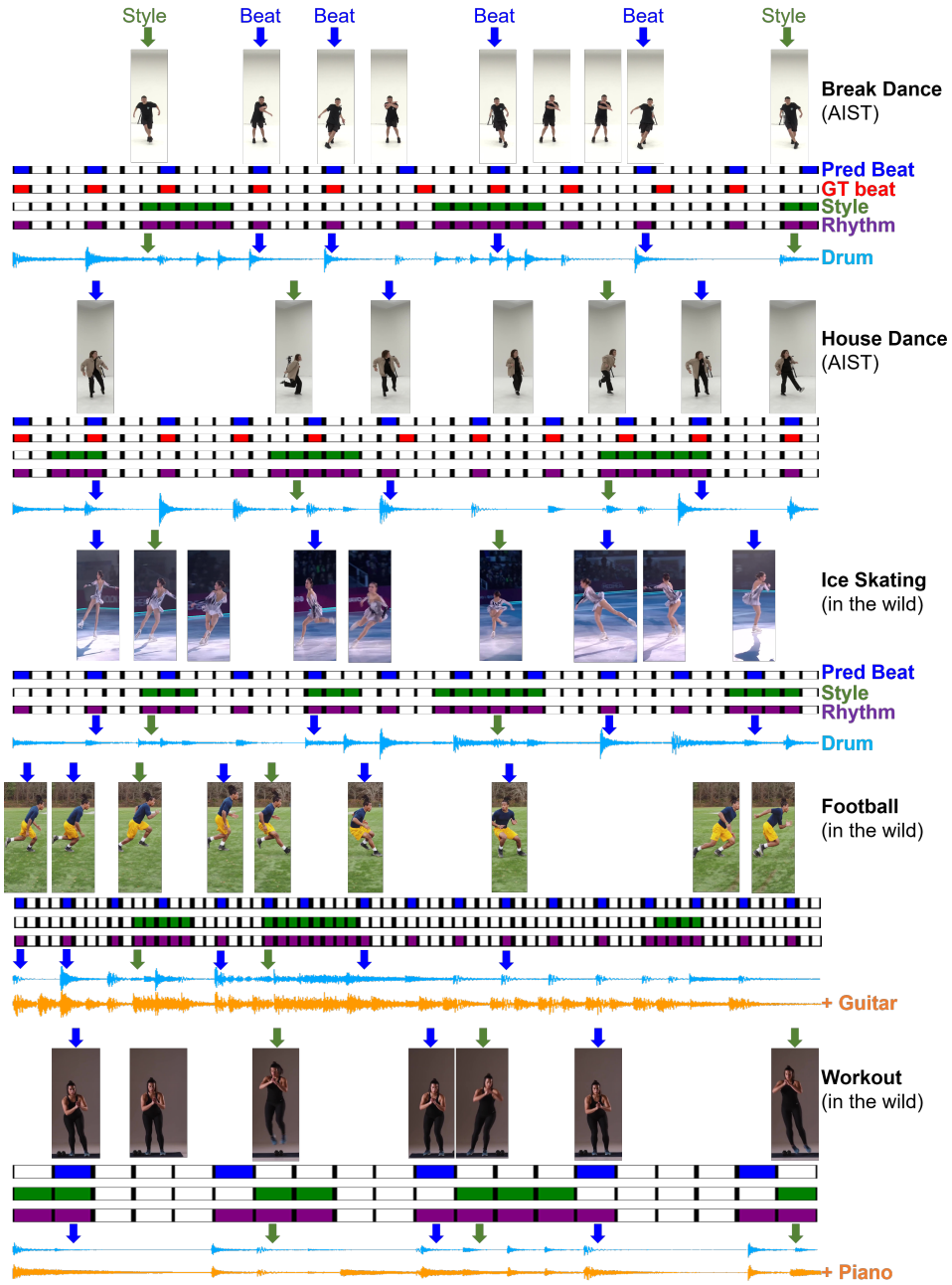

Figure 6: Examples of music generation on dance dataset and 'in the wild' videos. Dark blue: predicted beats, red: ground truth beats, Green: extracted style, Purple: rhythm, light blue: audio waveform of generated drum. Supplementary Materials include sounded video clips of these and additional samples.

components will need to be considered to address the transition from rhythmic drum track to a full bodied music with a symphony of instruments. Furthermore, enabling RhythmicNet to operate in real-time would allow music to be interactive with people and their movements. Due to the fact that the main cue for the generated music is human body movements, one possible concern could be that the soundtrack generated with RhythmicNet could be used to manipulate the original sounds of the video, to create a fake impression of people activity. This is a common concern in the application of any generative model. Failure in RhythmicNet may bring up an unsatisfying soundtrack but we do not expect serious consequences from this.

References

[1] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. *arXiv preprint arXiv:1801.07455*, 2018.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[4] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[5] Relja Arandjelovic and Andrew Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617, 2017.

[6] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in neural information processing systems*, pages 892–900, 2016.

[7] David Harwath, Antonio Torralba, and James Glass. Unsupervised learning of spoken language with visual context. In *Advances in Neural Information Processing Systems*, pages 1858–1866, 2016.

[8] Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. Ambient sound provides supervision for visual learning. In *European conference on computer vision*, pages 801–816. Springer, 2016.

[9] Ruohan Gao, Rogerio Feris, and Kristen Grauman. Learning to separate object sounds by watching unlabeled video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 35–53, 2018.

[10] Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. The sound of pixels. In *Proceedings of the European conference on computer vision (ECCV)*, pages 570–586, 2018.

[11] Hang Zhao, Chuang Gan, Wei-Chiu Ma, and Antonio Torralba. The sound of motions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1735–1744, 2019.

[12] Chuang Gan, Deng Huang, Hang Zhao, Joshua B Tenenbaum, and Antonio Torralba. Music gesture for visual sound separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10478–10487, 2020.

[13] Yapeng Tian, Jing Shi, Bochen Li, Zhiyao Duan, and Chenliang Xu. Audio-visual event localization in unconstrained videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 247–263, 2018.

[14] Eli Shlizerman, Lucio Dery, Hayden Schoen, and Ira Kemelmacher-Shlizerman. Audio to body dynamics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7574–7583, 2018.

[15] Shiry Ginosar, Amir Bar, Gefen Kohavi, Caroline Chan, Andrew Owens, and Jitendra Malik. Learning individual styles of conversational gesture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3497–3506, 2019.

[16] Hsin-Ying Lee, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. Dancing to music. In *Advances in Neural Information Processing Systems*, pages 3586–3596, 2019.

[17] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017.

[18] Amir Jamaludin, Joon Son Chung, and Andrew Zisserman. You said that?: Synthesising talking faces from audio. *International Journal of Computer Vision*, 127(11-12):1767–1779, 2019.

[19] Tae-Hyun Oh, Tali Dekel, Changil Kim, Inbar Mosseri, William T Freeman, Michael Rubinstein, and Wojciech Matusik. Speech2face: Learning the face behind a voice. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7539–7548, 2019.

[20] Yang Zhou, Xintong Han, Eli Shechtman, Jose Echevarria, Evangelos Kalogerakis, and Dingzeyu Li. Makelttalk: speaker-aware talking-head animation. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020.

[21] Andrew Owens, Phillip Isola, Josh McDermott, Antonio Torralba, Edward H Adelson, and William T Freeman. Visually indicated sounds. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2405–2413, 2016.

[22] Lele Chen, Sudhanshu Srivastava, Zhiyao Duan, and Chenliang Xu. Deep cross-modal audio-visual generation. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 349–357, 2017.

[23] Yipin Zhou, Zhaowen Wang, Chen Fang, Trung Bui, and Tamara L Berg. Visual to sound: Generating natural sound for videos in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3550–3558, 2018.

[24] Peihao Chen, Yang Zhang, Mingkui Tan, Hongdong Xiao, Deng Huang, and Chuang Gan. Generating visually aligned sound from videos. *IEEE Transactions on Image Processing*, 29:8292–8302, 2020.

[25] Yujia Wang, Wei Liang, Wanwan Li, Dingzeyu Li, and Lap-Fai Yu. Scene-aware background music synthesis. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1162–1170, 2020.

[26] A Sophia Koepke, Olivia Wiles, Yael Moses, and Andrew Zisserman. Sight to sound: An end-to-end approach for visual piano transcription. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1838–1842. IEEE, 2020.

[27] Kun Su, Xiulong Liu, and Eli Shlizerman. Audeo: Audio generation for a silent performance video. *arXiv preprint arXiv:2006.14348*, 2020.

[28] Chuang Gan, Deng Huang, Peihao Chen, Joshua B Tenenbaum, and Antonio Torralba. Foley music: Learning to generate music from videos. *arXiv preprint arXiv:2007.10984*, 2020.

[29] Kun Su, Xiulong Liu, and Eli Shlizerman. Multi-instrumentalist net: Unsupervised generation of music from body movements. *arXiv preprint arXiv:2012.03478*, 2020.

[30] Fabrizio Pedersoli and Masataka Goto. Dance beat tracking from visual information alone. *ISMIR*, 2020.

[31] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[32] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *International Conference on Machine Learning*, pages 4364–4373. PMLR, 2018.

[33] Jon Gillick, Adam Roberts, Jesse Engel, Douglas Eck, and David Bamman. Learning to groove with inverse sequence transformations. In *International Conference on Machine Learning (ICML)*, 2019.

[34] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications*, 32(4):955–967, 2020.

[35] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2018.

[36] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the maestro dataset. *arXiv preprint arXiv:1810.12247*, 2018.

[37] Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1180–1188, 2020.

[38] Jesse Engel, Matthew Hoffman, and Adam Roberts. Latent constraints: Learning to generate conditionally from unconditional generative models. *arXiv preprint arXiv:1711.05772*, 2017.

[39] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[40] Christine Payne. Musenet. *OpenAI Blog*, 2019.

[41] Kristy Choi, Curtis Hawthorne, Ian Simon, Monica Dinculescu, and Jesse Engel. Encoding musical style with transformer autoencoders. *arXiv preprint arXiv:1912.05537*, 2019.

[42] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*, 2018.

[43] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[44] Marco Körner and Joachim Denzler. Temporal self-similarity for appearance-based action recognition in multi-view setups. In *International Conference on Computer Analysis of Images and Patterns*, pages 163–171. Springer, 2013.

[45] Chuan Sun, Imran Nazir Junejo, Marshall Tappen, and Hassan Foroosh. Exploring sparseness and self-similarity for action recognition. *IEEE Transactions on Image Processing*, 24(8):2488–2501, 2015.

[46] Florian Krebs, Sebastian Böck, and Gerhard Widmer. An efficient state-space model for joint tempo and meter tracking. In *ISMIR*, pages 72–78, 2015.

[47] Peter Grosche, Meinard Müller, and Frank Kurth. Cyclic tempogram—a mid-level tempo representation for musicsignals. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5522–5525. IEEE, 2010.

[48] Abe Davis and Maneesh Agrawala. Visual rhythm and beat. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2532–2535, 2018.

[49] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[50] Shuhei Tsuchida, Satoru Fukayama, Masahiro Hamasaki, and Masataka Goto. Aist dance video database: Multi-genre, multi-dancer, and multi-camera database for dance information processing. In *ISMIR*, pages 501–510, 2019.

[51] Colin Raffel. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. PhD thesis, Columbia University, 2016.

[52] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[53] Matthew EP Davies, Norberto Degara, and Mark D Plumbley. Evaluation methods for musical audio beat tracking algorithms. *Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06*, 2009.

[54] Eitan Richardson and Yair Weiss. On gans and gmms. In *Advances in Neural Information Processing Systems*, pages 5847–5858, 2018.

[55] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. Gansynth: Adversarial neural audio synthesis. *arXiv preprint arXiv:1902.08710*, 2019.

[56] Li-Chia Yang and Alexander Lerch. On the evaluation of generative models in music. *Neural Computing and Applications*, 32(9):4773–4784, 2020.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Yes. The sections of Methods and Experiments & Results clearly describe the claims we made.
   (b) Did you describe the limitations of your work? [Yes] Yes. We describe the limitations of our work in the discussion section.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Yes. We discuss such impacts in the discussion section.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] Yes.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] N/A
   (b) Did you include complete proofs of all theoretical results? [N/A] N/A
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] No. However, the code will be available in the Github after the review process as we mention in the main paper.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Yes. We describe the training details in the implementation details section and more details can be found in the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] No. The error bars are not reported because it would be too computationally expensive.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Yes. We describe the number and type of GPUs in the Implementation Details section.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes] Yes. We cite all the existing assets used in our work.

   (b) Did you mention the license of the assets? [Yes] Yes. We mention the license of the assets in the supplementary material.

   (c) Did you include any new assets either in the supplementary material or as a URL? [No] No. We haven't included any new assets.

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] Yes. We discuss it in the supplementary material.

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] Yes. The Human Perceptual Test is fully described in the supplementary material.

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [No] No. There is no potential risk in our Human Perceptual Test.

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] Yes. We include these material in the supplementary material.