Bailando: 3D Dance Generation by Actor-Critic GPT with Choreographic Memory
Supplementary File

Li Siyao[1] Weijiang Yu[2] Tianpei Gu[3,4] Chunze Lin[4]
Quan Wang[4] Chen Qian[4] Chen Change Loy[1] Ziwei Liu[1] ✉
[1]S-Lab, Nanyang Technological University
[2]Sun Yat-Sen University [3]UCLA [4]SenseTime Research
siyao002@e.ntu.edu.sg weijiangyu8@gmail.com gutianpei@ucla.edu
{linchunze, wangquan, qianchen}@sensetime.com {ccloy, ziwei.liu}@ntu.edu.sg

Abstract

In this supplementary file, we present the detailed structures of the pose encoder and decoders, derive the actor-critic loss function from the reinforcement learning objective, and give an analysis of the user study results on different types of music. A demo video is available at https://www.youtube.com/watch?v=YbXOcuMTzD8.

1. Structure of Pose Encoder and Decoders

The encoder and decoders in Section 3.1 of the main paper are complied as 1D temporal CNNs. The detailed structures are shown in Table 1. The encoder is designed to downsample the 3D joint sequence 8 times into the encoding features, and therefore the learned quantized features are aware of contextual poses in the dance sequence.

Separate VQVAEs are trained for the compositional upper-and-lower half bodies, respectively. The joint number J of the upper half body is 15 and that of the lower is 9, while $J = 1$ in the global-velocity branch.

2. Derivation to Actor-Critic Loss

Here we derive the actor-critic loss $\mathcal{L}_{AC}$ from the initial reinforcement learning objective. The derivation is mainly based on the instruction of [1].

First, the learning objective can be reframed as

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[R(\tau)] = \int \pi_\theta(\tau)R(\tau)d\tau, \qquad (1)$$

where θ denotes the weights of the policy making network, τ represents a string of actions and $\pi_\theta(\tau)$ is the probability that the policy network predicts to take such actions.

One approach to maximize J is to optimize the network weight θ along the gradient $\nabla_\theta J$ as $\theta \leftarrow \theta + \alpha\nabla_\theta J$. Since

$$\nabla_\theta \pi_\theta = \pi_\theta \frac{\nabla_\theta \pi_\theta}{\pi_\theta} = \pi_\theta \nabla_\theta \log \pi_\theta, \qquad (2)$$

Table 1: **Architectures of pose encoder and decoders. (Section 3.1)**. "Conv" and "TransConv" represent 1D temporal the convolution and transpose-convolution operations, respectively, and their arguments represent the input channel number, the output channel number, the kernel size, the convolution stride, the padding size on both ends of input data, and the dilation number in turn. **RB** denotes Residual Block. J denotes the 3D joint number.

	Residual Block
	Input: 0; **Argument:** p, d
1	ReLU, Conv(512, 512, 3, 1, p, d)
2	ReLU, Conv(512, 512, 1, 1, 0, 1)
	Output: 0 + 2

	Encoder
	Input: 0, **Argument:** J
1	Conv($J \times 3$, 512, 4, 2, 1, 1)
2	**RB**($p = 1$, $d = 1$)
3	Conv(512, 512, 4, 2, 1, 1)
4	**RB**($p = 3$, $d = 3$)
5	Conv(512, 512, 4, 2, 1, 1)
6	**RB**($p = 9$, $d = 9$)
	Output: 6

	Decoder
	Input: 0, **Argument:** J
1	**RB**($p = 9$, $d = 9$)
2	TransConv(512, 512, 4, 2, 1, 1)
3	**RB**($p = 3$, $d = 3$)
4	TransConv(512, 512, 4, 2, 1, 1)
5	**RB**($p = 3$, $d = 3$)
6	TransConv(512, 512, 4, 2, 1, 1)
7	Conv(512, $J \times 3$, 3, 1, 1, 1)
	Output: 7

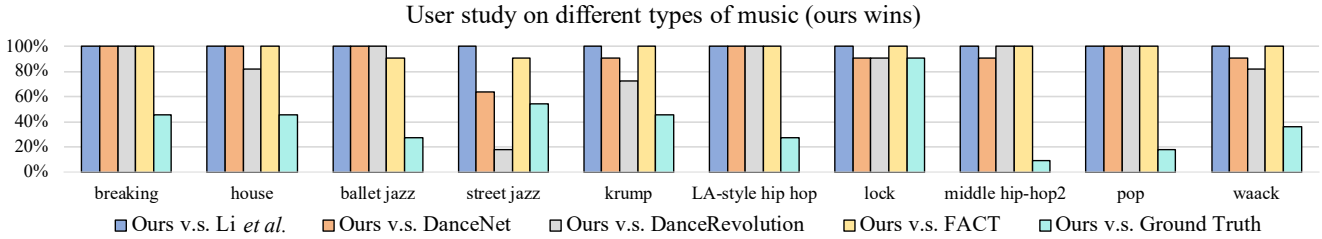

Figure 1: **The distribution of our method wins on different music types compared to state of the art.** Each bar indicates the percentage that *Bailando* wins in the comparison to the corresponding method.

we can rewrite $\nabla_\theta J$ into

$$\begin{aligned}\nabla_\theta J &= \int \nabla_\theta \pi_\theta(\tau) R(\tau) d\tau \\ &= \int \pi_\theta \nabla_\theta(\tau) \log \pi_\theta(\tau) R(\tau) d\tau \\ &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)} \left[\nabla_\theta \log \pi_\theta(\tau) R(\tau)\right].\end{aligned} \tag{3}$$

Note that $\tau = \{\mathbf{a}_t\}_{t=0}^{T'-1}$ is the trajectories of actions predicted on states $\{\mathbf{s}_t\}_{t=0}^{T'-1}$, the probability of trajectory $\pi_\theta(\tau)$ predicted by the policy making network is expanded to be $\prod_{t=0}^{T'-1} \pi_\theta(\mathbf{a}_t, \mathbf{s}_t)$, where $\pi_\theta(\mathbf{a}_t, \mathbf{s}_t)$ is the probability of action $\mathbf{a}_t$ under state $\mathbf{s}_t$. Hence,

$$\nabla_\theta \log \pi_\theta(\tau) = \sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t, \mathbf{s}_t) \tag{4}$$

and we have

$$\begin{aligned}&\nabla_\theta J = \mathbb{E}_{\tau\sim\pi_\theta(\tau)} \left[\nabla_\theta \log \pi_\theta(\tau) R(\tau)\right] \\ &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)} \left[\left(\sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t, \mathbf{s}_t)\right)\left(\sum_{t=0}^{T'-1} R(\mathbf{a}_t, \mathbf{s}_t)\right)\right] \\ &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)} \left[\sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t, \mathbf{s}_t)\left(\sum_{t'=0}^{T'-1} R(\mathbf{a}_{t'}, \mathbf{s}_{t'})\right)\right].\end{aligned} \tag{5}$$

For on-policy reinforcement learning, $\nabla_\theta J$ is estimated on simultaneously sampled sectional trajectories $\{(\mathbf{a}_t^m, \mathbf{s}_t^m)\}_{m=0}^{M-1}$, where M denotes the sampling batch size, such that the equation above is approximated to be

$$\nabla_\theta J \approx \frac{1}{M} \sum_{m=0}^{M-1} \sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t^m, \mathbf{s}_t^m) \left(\sum_{t'=0}^{T'-1} R(\mathbf{a}_{t'}^m, \mathbf{s}_{t'}^m)\right). \tag{6}$$

Note that the optimization of the policy making network for policy $(\mathbf{a}_t^m, \mathbf{s}_t^m)$ is not expected to be influenced by the past trajectories, *i.e.*, the rewards before t. Therefore, Equation (6) is reframed as

$$\nabla_\theta J \approx \frac{1}{M} \sum_{m=0}^{M-1} \sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t^m, \mathbf{s}_t^m) \left(\sum_{t'=t}^{T'-1} R(\mathbf{a}_{t'}^m, \mathbf{s}_{t'}^m)\right), \tag{7}$$

where $\sum_{t'=t}^{T'-1} R(\mathbf{a}_{t'}, \mathbf{s}_{t'})$ is the expected "reward to go" under policy $(\mathbf{a}_t, \mathbf{s}_t)$, which is formally named as the Q-value $Q(\mathbf{a}_t, \mathbf{s}_t)$.

To avoid the bias of rewards, *e.g.*, all rewards are positive, the Q-value item in Equation (7) is normalized by an expected "reward to go" on state $\mathbf{s}_t$, *i.e.*, the critic value $\mathbf{v}_t = \mathbb{E}_{\mathbf{a}_t}[Q(\mathbf{a}_t, \mathbf{s}_t)]$, such that

$$\begin{aligned}\nabla_\theta J &\approx \frac{1}{M} \sum_{m=0}^{M-1} \sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t^m, \mathbf{s}_t^m) \left(Q(\mathbf{a}_t, \mathbf{s}_t) - \mathbf{v}_t\right) \\ &= \frac{1}{M} \sum_{m=0}^{M-1} \sum_{t=0}^{T'-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t^m, \mathbf{s}_t^m) \left(R(\mathbf{a}_t, \mathbf{s}_t) + \mathbf{v}_{t+1} - \mathbf{v}_t\right).\end{aligned} \tag{8}$$

Looking closely to Equation (8), if the normalized "reward to go" is positive, to increase J, the policy making network will be optimized to enhance the probability of action $\mathbf{a}_t^m$ under state $\mathbf{s}_t^m$. In our proposed framework, the actions are within finite selections of the choreographic memory. Hence, the optimization along with Equation (8) is equivalent to the the optimization via an weighted cross-entropy loss on the in-time self predictions of the policy making network:

$$\mathcal{L}_{AC} = \frac{1}{T'-1} \sum_{t=0}^{T'-2} \left(\sum_{h=u,l} \text{CrossEntropy}\left(\mathbf{a}_t^h, \hat{p}_{t+1}^h\right)\right) \cdot \text{sg}[\boldsymbol{\varepsilon}_t], \tag{9}$$

where $\hat{p}_{t+1}^h = \arg\max_i \mathbf{a}_{t,i}^h$ is the pose code number predicted by the policy making network. $\boldsymbol{\varepsilon} \in \mathbb{R}^{(T'-1)\times 1}$ denotes the so-called TD-error calculated as

$$\boldsymbol{\varepsilon}_{0:T'-2} = \mathbf{r}_{0:T'-2} + \text{sg}[\mathbf{v}_{1:T'-1}] - \mathbf{v}_{0:T'-2}, \tag{10}$$

where $\mathbf{r}_t = R(t)$.

Note the time length in Equation (9) is $T'-1$ instead of T' because the length of critical values is T' and the TD-error need to be calculated by subtracting the neighboring items.

3. Detailed Distribution of User Study Result

we conduct a user study among the dance sequences generated by each compared method and the ground truth data in AIST++ test set. The experiment is conducted with 11 participants separately. For each participant, we randomly play 50 pairs of comparison videos with a length of around 10 seconds, where each pair contains our result and one competitor's in the same music, and ask the participant to indicate "*which one is dancing better to the music*".

The detailed distribution of the user study results on various music types are shown in Figure 1. Our method appears to perform balanced in different kinds of dances and achieves outstanding performance in some types. For example, for the breaking dance, *Bailando* wins all votes in comparisons to state of the art, while even over 40% participants are more favor of its generated dances than ground truth.

References

[1] Sergey Levine. https://rail.eecs.berkeley.edu/deeprlcourse/.