Diffusion model inbetweening

Aditya Kumar

Implementation Details

Introduction

Motion in-betweening is used for generating motion sequences to interpolate user-provided keyframe constraints. For decades it had been a challenging and time consuming process. The authors have tried to use diffusion models in generating diverse human motions based on text constraints and keyframes. They have named this modedl Conditional Motion Diffusion In-betweening (CondMDI)

Motion Representation

Each motion sequence can be divided into 2 parts:

- **Local Motion** - Motion of the pose of the skeleton relative to the root
- **Global Motion** - Motion of the root of the skeleton

Root postions are usually represented relative to the previos frame and hence to have spatial sparsity in these keyframes becomes a challenging task.

My implementation

The paper has 3 modes of training and evaluations :

1. Conditionally trained on randomly sampled frames and joints (CondMDI)
2. Conditionally trained on randomly sampled frames
3. Unconditionally (no keyframes) trained

I have trained and attached the results for the unconditionally trained model as it requires the least amount of space and time to train. The other modes as are described in the readme file work fine and you are free to run them and see the results.

During the course of implementing this paper the major problem faced was that of AMASS dataset that is used to generate HumanML3D dataset that is used to train the models in this paper. I first downloaded this dataset on my computer and then used the HumanML3D repository to generate the dataset. Then uploaded this dataset on kaggle where I used it to train my model.

The authors have used 1200000 epochs (or number of steps) to train the model with each epoch taking 6 minutes. Kaggle (where I was trying to run my code) has a limit of 12 hours of runtime. This was a major bottleneck in my implementation. So I limited the number of epochs to 100 inorder to be able to generate results in the given time frame. Also Kaggle only allows 20GB of space and the dataset itself consumed 12GB and the model and evaluators consumed another 4 GB thus there was not enough space left to save checkpoints or the logs that were getting generated.

The model worked fine and was able to generate a test sequence of frames of man jumping and exercising.

Dataset Description

The model is evaluated on the HumanML3D dataset, which contains 14,646 text-annotated human motion sequences taken from the AMASS and HumanAct12 datasets. Motion sequences have variable lengths with an average of 7.1 seconds and are padded with 0's to have a fixed length of 196 frames with 20 fps. Motion in every frame is represented by a 263-dimensional feature vector, capturing the relative root joint translations, rotations and local pose with respect to the root joint.

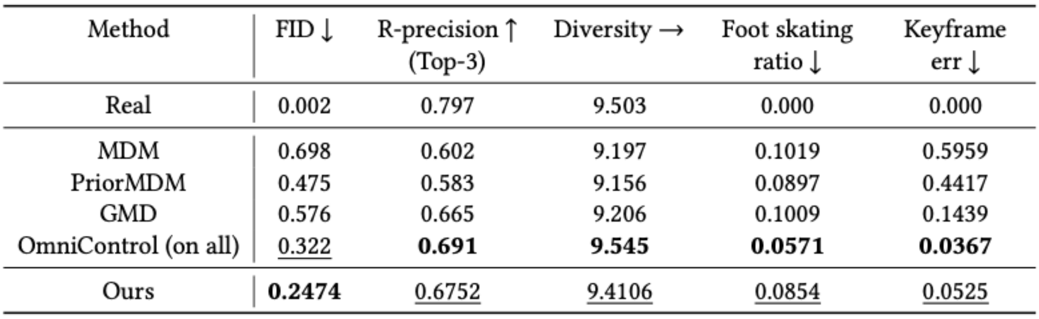
The AMASS dataset is freely available on the internet but it cannot be accessed from any VM and needs to be downloaded from the official website (requires you to sign in to the website). The representation of human joints used in GMD has been changed from relative to absolute in the HumanML3D dataset. Hence the paper uses absolute joint positions for training but for evaluations it still uses the old relative joint positions. This implies that the size of the dataset grows from 6GB to 12GB. The link of HumanML3D with absolute joint positions is provided in the google form while the link of the other dataset is given here.

Results

The model uses 5 metrics to evaluate the performance of the model:

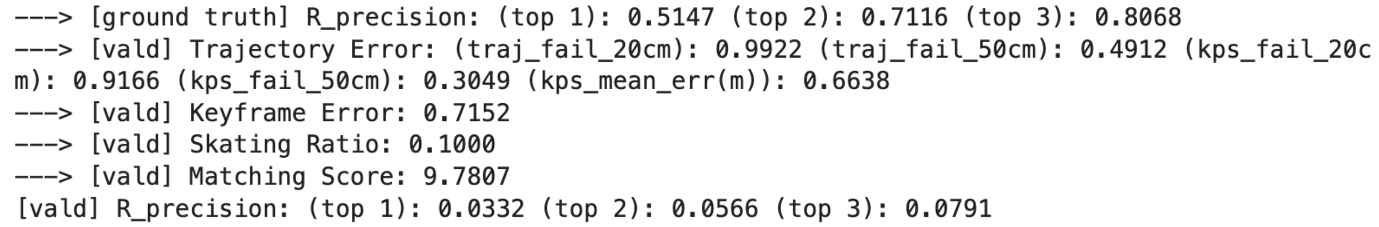
- **FID** - Frechet Inception Distance is the distance between the generated and the real motions in the latent space of pretrained encoders.
- **R-Precision** - The proximity of the motion to the text it was conditioned on.
- **Diversity or Matching Score** - Measures the variability in the generated motion.
- **Foot Skating Ratio** - Measures the proportion of frames in which either the foot skids more than a certain distance
- **Keyframe Error** - Mean distance between the generated motion root locations and the keyframe root locations at teh keyframe motion steps.

Results given in the Paper



Method	FID ↓	R-precision ↑ (Top-3)	Diversity →	Foot skating ratio ↓	Keyframe err ↓
Real	0.002	0.797	9.503	0.000	0.000
MDM	0.698	0.602	9.197	0.1019	0.5959
PriorMDM	0.475	0.583	9.156	0.0897	0.4417
GMD	0.576	0.665	9.206	0.1009	0.1439
OmniControl (on all)	0.322	**0.691**	**9.545**	**0.0571**	**0.0367**
Ours	**0.2474**	0.6752	9.4106	0.0854	0.0525

Results obtained by me



```
---> [ground truth] R_precision: (top 1): 0.5147 (top 2): 0.7116 (top 3): 0.8068
---> [vald] Trajectory Error: (traj_fail_20cm): 0.9922 (traj_fail_50cm): 0.4912 (kps_fail_20c
m): 0.9166 (kps_fail_50cm): 0.3049 (kps_mean_err(m)): 0.6638
---> [vald] Keyframe Error: 0.7152
---> [vald] Skating Ratio: 0.1000
---> [vald] Matching Score: 9.7807
[vald] R_precision: (top 1): 0.0332 (top 2): 0.0566 (top 3): 0.0791
```

The FID obtained by me was 0.562