Object Recognition and Computer Vision
Topic C: Text-to-Motion Retrieval - Project Report

January 2024

Borachhun YOU
École Normale Supérieure Paris-Saclay
borachhun.you@ens-paris-saclay.fr

Anshuman SINHA
Institut Polytechnique de Paris
anshuman.sinha@ip-paris.fr

Abstract

This report explores a model named TMR which was designed for text to 3D human motion retrieval task. The evaluations of the pretrained models are performed and discussed. In addition, different data augmentation methods are presented with the objective of improving the performance of the model.

1. Introduction

Multimodal learning has recently captured much attention in the field of machine learning. In this project, our team chose a topic that revolves around 2 different types of data: text and motion. Specifically, we focus on text-to-motion retrieval task, i.e. given a natural language query, we would like to search for the most relevant 3D human motions from the database.

In order to achieve this task, Petrovich et al. presented a transformer-based model named **TMR** [6], and it is the primary focus in this project.

In this report, we start by exploring architecture of the TMR model (section 2), before taking a look at the result of the evaluation of the pretrained models (section 3). After that, different data augmentation methods are presented in an attempt to improve the performance of the model (section 4).

2. Model

TMR is an extension of a transformer-based model named TEMOS [5], which was created for text-to-motion synthesis task. The model is composed of 2 independent encoders whose inputs are text and motion, used to generate the encodings of text prompts and to learn representations of 3D human motions. It also features an additional decoder to reconstruct 3D motions from motion representations and corresponding positional encodings.

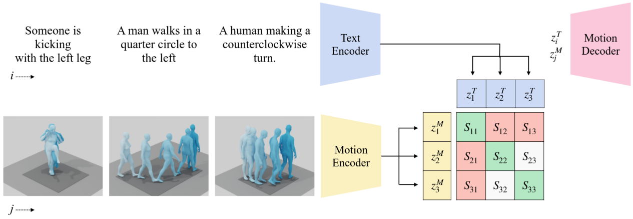

Figure 1. Simplified architecture of TMR (Source: TMR: Text-to-Motion Retrieval Using Contrastive 3D Human Motion Synthesis by Petrovich et al. [6])

The cross-modal retrieval task is achieved with the help of a similarity function between the two modalities. The function compares a given natural language query with each motion in the database, and the best match is indicated by the maximum value. In the case of TMR, cosine similarity is calculated between the embeddings outputted by the encoders of both modalities.

Both the motion and text encoders are transformer-based, which output parameters of a Gaussian distribution, allowing a sampling of a latent vector. Text features from a pre-trained DistilBERT [10] model are used as inputs of the text encoder while the motion encoder takes motion sequences directly as input.

Training is performed on two main losses. The first loss is the same as the one used for TEMOS [5] and the second one, a novel contribution of the paper, is a contrastive loss that also includes negative samples in its calculations, along with a temperature parameter.

The TEMOS loss consists of 3 major weighted components: namely, an L1 reconstruction loss representing the decoders performance, an L1 cross-modal similarity loss between the motion and text embeddings, and a KL divergence loss. The divergence loss is designed to regularise both the modality's embeddings to fit a normal distribution and enforces similarity between the two distributions.

3. Experiments

3.1. Datasets

In order to train and evaluate the TMR model, 2 different types of datasets are required: motion and text datasets.

AMASS [4] is a large human motion dataset that unifies multiple motion capture datasets by representing them within a common framework and parameterization. It was used as the motion dataset for both training and the evaluation of the model.

As for the text dataset, 2 datasets were utilized: HumanML3D and KIT-ML, both of which went through the same preprocessing procedure as presented in the original paper of TMR. **HumanML3D** [3] assigns text labels describing the motions in AMASS as well as HumanAct12 [1]. For training, validation and testing, the dataset was divided into 3 subsets of 23384, 1460 and 4384 motions, respectively. For each motion, there were 3 different texts on average describing the motion. As for the **KIT-ML** dataset [7], it was divided into splits of 4688, 292 and 786 motions for training, validation and testing, respectively. Each motion was annotated 2.1 times on average. During training, one text is chosen at random for the motion whereas the first text is selected for the motion during evaluation.

3.2. Evaluation protocols

The performance of the model was evaluated for both motion-to-text and text-to-motion tasks, with recall at different ranks (1,2,3,5,10) and median rank as measures. In addition, there were 4 different protocols of evaluation:

- Normal: use the whole test set without any modification.
- Threshold: use the whole test set but consider a retrieved motion to be correct if the similarity between its text label and the query text is above a threshold (set to 0.95)
- Nsim: use a subset by sampling 100 motion-text pairs with the texts being maximally dissimilar from each other.
- Guo: evaluation protocol described by Guo et al. in [2] which involves picking small batches of 32 motion-text pairs at random and computing the average performance.

3.3. Evaluation of pretrained models

The code base of the TMR model is available at https://github.com/Mathux/TMR and it was primarily used for training and evaluations of the model in this project.

The code base provides 2 pretrained models, one was pretrained with HumanML3D and the other with KIT-ML. Both pretrained models were evaluated on the test set of their respective dataset. Furthermore, cross-dataset evaluations were also done with the models, i.e. the model that was pretrained with HumanML3D was evaluated on the KIT-ML dataset and vice versa for the other pretrained model. The evaluation results are given in table 1-4 in appendix A.

4. Improvement

This section demonstrates an attempt to improve the performance of the TMR model with data augmentation on both types of dataset. Methods of text augmentation are discussed in section 4.1 while methods of motion augmentation are explored in section 4.2.

4.1. Text augmentation

The text datasets were augmented through a process called **back translation**. The idea was to use large language models (LLMs) to paraphrase the texts by translating them from the original language to another language and then back to the original language in order to get a copy of the texts with different words but still have the same meaning. In our case, the texts were translated from English to German and then back to English with pretrained LLMs:

- English to German: T5-Base [8], a LLM that unifies NLP tasks, including machine translation, document summarization, question answering and classification tasks, into text-to-text-format.
- German to English: Bert2Bert [9], which was initialized on the Bert-Large checkpoints and fine-tuned on German to English translation on the WMT dataset.

After new texts were added to the existing datasets, further processing was done on the new datasets, specifically removing stop words. Stop words are words that do not contribute meaning to the sentence. Articles (“a”, “the”) and verb “to be” (“is”, “are”) are some examples of stop words.

The augmentation and processing of the texts were performed by accessing the existing annotation files of the 2 text datasets in the code base of the model. The annotation files were previously generated using codes in the AMASS Annotation Unifier repository, which unifies different motion-text datasets into a common representation.

The re-training of the model with the new text datasets used the same optimizer and hyperparameters as in [6]. After the re-training, the model was revaluated both on the same dataset and cross-dataset. The results of the evaluations are given in table 5-8 in appendix B.

Based on the results, when testing on the same dataset as the training, there was no improvement in terms of the measures that were used. In contrast, when evaluating with different dataset from the one used for training, we can observe some improvements for certain measures and protocols, which are written in bold in the result tables.

The code for the text augmentation is made available through this link: https://bit.ly/tmrtextaug.

4.2. Motion augmentation

The motion datasets underwent several augmentation techniques. Some of them had tunable hyperparameters. Displayed results refer to the specific values mentioned in each section.

4.2.1 Dropping/Repeating Frames

This technique involves replacing each frame (joint coordinates) with the previous one with a probability ϵ. This is a tunable hyperparameter and experiments were run on the following values: $\epsilon \in \{0.05,\ \mathbf{0.1},\ 0.2,\ 0.5\}$

Thereafter, best performances were noticed with $\epsilon = 0.1$. Being a very simple augmentation task, it is computationally easy to implement and integrate, and can subsequently increase the dataset by multiple folds.

4.2.2 Adding Random Noise

This technique involves adding random Gaussian noise to the 3D motion data to achieve augmentation. As a result, the joint coordinates get noisy, thus creating small and effective perturbations in the motions, globally, while conserving the motions that it actually represents.

4.2.3 Varying Frame Rate

Adjusting the fps (frames per second) of the video motions also acts as an augmentation mechanism. The tested values are 12.5 (default) and 25, respectively.

4.2.4 Adding Latent Noise in Autoencoder

We design a lightweight auto-encoder on KIT to reconstruct the motions (joint positions) from lower dimensional latent representations. Herein, augmentation is achieved by adding random Gaussian noise to the bottleneck / latent variables and performing a self-supervised learning task by trying to reconstruct the data with a noisy auto-encoder.

The results of training and testing on original and cross-datasets are given in table 9-12 in appendix C.

5. Conclusion

Throughout this report, we have explored TMR, a model designed specifically for text-to-motion retrieval task. Particularly, we have studied the architecture of the model, the datasets that were used and the evaluation protocols that were carried out. We have also performed the evaluation of the pretrained models on the same datasets as well as cross-datasets. Moreover, different data augmentation techniques on both text and motion datasets have been presented with the goal of improving the performance of the model. With the obtained results, some improvements can be observed with cross-dataset evaluations of the models re-trained on the augmented datasets.

References

[1] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM International Conference on Multimedia*. ACM, 2020. 2

[2] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5142–5151, 2022. 2

[3] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5152–5161, 2022. 2

[4] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, 2019. 2

[5] Mathis Petrovich, Michael J. Black, and Gül Varol. Temos: Generating diverse human motions from textual descriptions, 2022. 1

[6] Mathis Petrovich, Michael J. Black, and Gül Varol. TMR: Text-to-motion retrieval using contrastive 3D human motion synthesis. In *International Conference on Computer Vision (ICCV)*, 2023. 1, 2

[7] Matthias Plappert, Christian Mandery, and Tamim Asfour. The KIT motion-language dataset. *Big Data*, 4(4):236–252, 2016. 2

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023. 2

[9] Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280, 2020. 2

[10] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020. 1

Appendix

Each table contains 2 sub-tables, the top gives the result of the text-to-motion task and the botton gives the result of the motion-to-text task. Bold numbers indicate improvements in the case of data augmentations.

A. Evaluation of pretrained models

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	5.82	10.95	14.32	20.51	31.32	27.00
Threshold	12.98	16.24	21.26	28.47	39.03	18.00
Nsim	46.00	64.00	71.00	82.00	88.00	2.00
Guo	67.52	81.18	86.31	91.47	95.39	1.03

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	9.22	11.68	16.74	22.54	32.57	25.50
Threshold	12.39	14.90	21.26	27.58	37.73	20.00
Nsim	49.00	69.00	78.00	83.00	87.00	2.00
Guo	68.09	81.84	86.86	91.20	95.42	1.02

Table 1. Pretrained and evaluated on HumanML3D

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	7.63	12.85	17.68	25.06	37.66	19.00
Threshold	22.39	29.52	36.90	45.93	59.92	6.50
Nsim	29.59	45.92	55.10	64.29	78.57	3.00
Guo	50.13	71.48	79.95	90.10	95.44	1.52

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	11.70	15.27	22.65	28.75	39.82	17.50
Threshold	19.21	23.16	32.57	40.59	51.15	10.25
Nsim	42.86	52.04	62.24	71.43	84.69	2.50
Guo	52.60	73.18	83.33	91.15	95.96	1.45

Table 2. Pretrained and evaluated on KIT-ML

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	2.32	4.49	6.05	8.81	15.35	87.00
Threshold	15.83	21.18	27.77	35.29	45.86	14.00
Nsim	29.59	51.02	56.12	70.41	85.71	2.00
Guo	55.19	73.21	82.20	90.49	96.25	1.30

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	3.69	4.79	7.15	10.41	15.63	91.00
Threshold	14.20	17.45	23.08	28.72	36.07	24.00
Nsim	41.84	55.10	63.27	71.43	77.55	2.50
Guo	55.24	73.23	82.53	90.92	96.27	1.31

Table 3. Pretrained on HumanML3D and evaluated on KIT-ML

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	1.03	1.78	2.40	4.11	6.55	413.0
Threshold	2.35	3.88	5.22	8.33	12.16	266.0
Nsim	14.00	19.00	25.00	35.00	44.00	14.50
Guo	25.94	39.46	47.9	58.14	72.76	4.13

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	1.37	1.76	2.67	3.95	6.25	456.5
Threshold	2.58	3.17	4.61	6.3	9.22	359.75
Nsim	16.00	22.00	26.00	33.00	42.00	15.00
Guo	25.98	38.48	46.08	56.93	72.74	4.38

Table 4. Pretrained on KIT-ML and evaluated on HumanML3D

B. Evaluation after text augmentation

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	3.35	5.54	8.05	11.72	19.25	54.00
Threshold	7.82	12.11	17.06	23.04	32.64	28.00
Nsim	38.00	54.00	65.00	75.00	84.00	2.00
Guo	56.34	73.43	80.52	88.18	94.73	1.22

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	5.47	6.66	10.04	13.32	20.32	53.00
Threshold	7.98	9.33	14.19	17.95	25.18	44.50
Nsim	38.00	52.00	64.00	73.00	82.00	2.00
Guo	58.12	74.36	81.39	88.14	94.98	1.17

Table 5. Trained and evaluated on HumanML3D

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	5.09	10.69	14.12	21.50	33.21	22.00
Threshold	19.34	28.63	35.11	44.40	55.98	8.00
Nsim	17.35	33.67	45.92	59.18	70.41	4.00
Guo	42.97	63.54	73.57	83.33	92.84	1.83

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	9.54	13.49	17.43	23.79	34.48	21.50
Threshold	18.70	**23.79**	29.01	34.48	45.42	13.50
Nsim	31.63	41.84	52.04	62.24	71.43	3.50
Guo	47.01	66.28	76.82	85.03	91.80	1.71

Table 6. Trained and evaluated on KIT-ML

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	1.34	2.58	3.75	6.09	11.12	109.0
Threshold	13.18	20.07	26.38	**36.07**	**48.02**	**12.00**
Nsim	**30.61**	48.98	**57.14**	69.39	81.63	3.00
Guo	52.31	72.36	**82.29**	**91.72**	**97.86**	1.42

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	2.24	3.33	5.45	8.00	12.35	101.5
Threshold	10.27	12.94	18.51	23.69	30.77	36.50
Nsim	37.76	51.02	63.27	**73.47**	**84.69**	2.50
Guo	53.91	72.57	82.03	90.38	**97.17**	1.33

Table 7. Pretrained on HumanML3D and evaluated on KIT-ML

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	0.73	1.14	1.64	2.94	5.04	485.0
Threshold	**2.69**	3.88	5.06	7.28	10.77	342.0
Nsim	**16.00**	**22.00**	**27.00**	**37.00**	**46.00**	**12.50**
Guo	23.77	36.72	44.50	55.95	72.17	4.54

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	1.09	1.55	2.44	3.26	4.88	493.0
Threshold	1.82	2.53	4.06	5.43	7.60	414.75
Nsim	13.00	22.00	**27.00**	**37.00**	**47.00**	**13.00**
Guo	24.82	37.27	45.14	55.47	**72.86**	4.48

Table 8. Pretrained on KIT-ML and evaluated on HumanML3D

C. Evaluation after motion augmentation

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	3.94	8.65	11.73	17.61	23.04	28.95
Threshold	7.53	13.15	17.54	22.58	30.51	29.14
Nsim	37.00	51.00	62.00	70.00	78.00	2.50
Guo	65.45	76.59	82.25	87.37	93.56	1.43

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	7.03	9.58	13.83	18.39	27.31	36.64
Threshold	10.58	12.16	17.33	22.17	29.98	29.47
Nsim	45.00	61.00	70.00	79.00	83.00	2.00
Guo	64.51	77.15	82.32	86.48	93.22	1.22

Table 9. Trained and evaluated on HumanML3D

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	6.47	10.92	16.34	22.89	33.40	21.00
Threshold	19.53	27.18	34.55	43.10	56.47	7.50
Nsim	22.69	39.45	48.39	58.74	72.31	3.50
Guo	45.61	67.28	74.89	86.31	93.52	1.69

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	10.32	14.12	20.48	26.53	37.75	17.75
Threshold	18.04	21.93	30.26	38.17	49.52	10.50
Nsim	40.68	50.73	61.05	69.92	82.44	2.50
Guo	52.35	**74.37**	83.19	**91.53**	95.62	1.54

Table 10. Trained and evaluated on KIT-ML

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	2.15	3.92	5.34	7.29	13.77	98.00
Threshold	**14.91**	20.83	**28.51**	**36.07**	43.54	**13.00**
Nsim	**30.12**	**52.12**	55.85	67.92	83.19	2.50
Guo	**57.62**	71.18	**83.48**	90.27	**96.39**	**1.20**

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	**4.37**	**5.21**	**8.07**	10.28	13.95	91.00
Threshold	13.45	17.28	21.59	**29.33**	34.86	**23.50**
Nsim	38.35	**56.24**	63.19	70.84	**78.03**	2.50
Guo	**56.12**	**74.53**	82.21	89.92	**96.72**	**1.20**

Table 11. Pretrained on HumanML3D and evaluated on KIT-ML

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	**1.17**	1.56	2.24	**4.38**	6.39	416.0
Threshold	2.10	3.58	**5.34**	8.10	12.02	269.0
Nsim	13.00	18.00	23.00	**37.00**	**46.00**	15.00
Guo	**26.13**	38.59	**48.25**	**57.88**	72.35	**3.86**

Protocol	R@1	R@2	R@3	R@5	R@10	MedR
Normal	1.22	1.58	2.49	3.65	**6.31**	457.75
Threshold	**2.59**	**3.24**	4.58	6.17	8.82	362.25
Nsim	**17.00**	**23.00**	**28.00**	**35.00**	**43.00**	**13.00**
Guo	**26.41**	**39.14**	**48.31**	**57.19**	**73.17**	**3.40**

Table 12. Pretrained on KIT-ML and evaluated on HumanML3D