Adversarial regularizer for improving autoencoders' latent space with human 3d-poses

Vincent Naayem - Supervisor: Dr. Andrey Davydov - Professor: Professor Pascal Fua
(sciper: 257676)

I. Introduction

One goal of unsupervised learning is to uncover the underlying structure of a dataset without using explicit labels. A common architecture used for this purpose is the autoencoder, which learns to map datapoints to a latent code from which the data can be recovered with minimal information loss. Typically, the latent code is lower dimensional than the data, which indicates that autoencoders can perform some form of dimensionality reduction. For certain architectures, the latent codes have been shown to disentangle important factors of variation in the dataset which makes such models useful for representation learning. More recently, it was shown that imposing a prior on the latent space allows autoencoders to be used for probabilistic or generative modeling.

In some cases, autoencoders have shown the ability to interpolate. Specifically, by mixing codes in latent space and decoding the result, the autoencoder can produce a semantically meaningful combination of the corresponding datapoints. This behavior can be useful in its own right e.g. for creative applications. However, [1] also argue that it demonstrates an ability to "generalize" in a loose sense – it implies that the autoencoder has not simply memorized how to reconstruct a small collection of datapoints. From another point of view, it also indicates that the autoencoder has uncovered some structure about the data and has captured it in its latent space. These characteristics have led interpolations to be a commonly reported experimental result in studies about autoencoders and latent-variable generative models in general. The connection between interpolation and a "flat" data manifold has also been explored in the context of unsupervised representation learning and regularization.

In [1], researchers propose a regularization procedure which encourages interpolated outputs to appear more realistic by fooling a critic network which has been trained to recover the mixing coefficient from interpolated data. They then develop a simple benchmark task where they can quantitatively measure the extent to which various autoencoders can interpolate and show that their regularizer dramatically improves interpolation in this setting. They also demonstrate empirically that their regularizer produces latent codes which are more effective on downstream tasks, suggesting a possible link between interpolation abilities and learning useful representations.

Some models like SMPL are now widely used to parameterize body poses and shapes. However, they offers no guarantee to produce realistic human bodies when random values are passed as its inputs. This complicates its usage within an optimization, regression, or generative frameworks, where it is desirable that any sample drawn be plausible.

To mitigate this issue, we explore the effects of ACAI [1] on 3d human poses interpolations. We chose simple autoencoder architectures and compared interpolation results with and without ACAI. We also explore the effect of the regularization term hyperparameter of ACAI on the latent space. We used a spherical distribution in the input space of the learned prior, which facilitates plausible sample generation as shown in [2]. 3d Human poses are drawn from the h36m dataset. Different metrics are used to measure the "interpolation quality" and the "dataset coverage". Both concepts are explained in section IV.

II. Implementation

A. Dataset and Autoencoders

The dataset used consist of 3d-poses from the h36m dataset. A 3d-pose consist of 17 3d-joints (17×3 tensors). Augmentation such as rotation up to 30° and horizontal flip are used on the training set.

Autoencoders consist of the following structure: First, an input $x \in R^{d_x}$ is passed trough an "encoder" $z = f_\theta(x)$ parametrized by θ to obtain a latent code $z \in \mathbb{R}^{d_z}$. The latent code is then passed through a "decoder" $\hat{x} = g_\phi(z)$ parametrized by ϕ to produce an approximate reconstruction $\hat{x} \in \mathbb{R}^{d_x}$ of the input x.

We consider the case where f_θ and g_ϕ are implemented as multi-layer neural networks. The encoder and decoder are trained simultaneously to minimize the squared L2 distance $||x-\hat{x}||^2$ between the input x and the output $\hat{x}$.

As a baseline, we trained two models of "vanilla" autoencoders consisting of symmetrical bottleneck fully connected layers. For the two autoencoders we simply chose different latent-space size giving networks of size $51 \times 256 \times 256 \times 16 \quad 16 \times 256 \times 256 \times 51$ nodes and of size $51 \times 256 \times 256 \times 8 \quad 8 \times 256 \times 256 \times 51$ nodes.

A spherical distribution is used to draw latent vectors from a set $\mathbb{P}_z \subseteq \mathbb{R}^{d_z}$

- $z_N \sim \mathbb{P}_z = \mathcal{N}(0, \mathcal{I}) \quad (Normal)$
- $z_S \sim \mathbb{P}_z = \mathcal{S} \subset \mathbb{R}^{d_z} \quad (Spherical)$

where the spherical vectors are sampled by drawing vectors z_N from a normal distribution and computing $z_S := \frac{z_N}{||Zn||^2}$. The desirable properties of the latent space, such as continuity and boundedness are all inherent to the Spherical distribution.

To impose a spherical distribution to the latent space, when training all models we normalize the latent space codes $\mathbf{z} = f_\theta(x)$ generated by the encoder to obtain $\mathbf{z}_s = \frac{f_\theta(x)}{||f_\theta(x)||2}$ and then feed it to generator to obtain the reconstruction $\hat{x} = f(\mathbf{z}_s)$.

B. Interpolations

Interpolating using an autoencoder describes the process of using the decoder g_ϕ to decode a mixture of two latent codes. To obtain points residing on a sphere of dimension $\mathbb{R}^{d_z}$ we will use a spherical interpolation to generate interpolated latent codes. Let x_1 and x_2 be two input images we are interpolating between and

$$\hat{x}_n = g_\phi\left(\frac{\sin(1-\frac{n-1}{N-1})\theta}{\sin\theta} z_1 + \frac{\sin\frac{n-1}{N-1}\theta}{\sin\theta} z_2\right) \quad (1)$$

be the decoded point corresponding to mixing x_1 and x_2's latent codes using coefficient $\alpha = \frac{n-1}{N-1}$. The images $\hat{x}_n, n \in \{1, ..., N\}$ then comprise a length-N interpolation.

Ideally, adjusting α from 0 to 1 will produce a sequence of realistic datapoints where each subsequent $\hat{x}_\alpha$ is progressively less semantically similar to x_1 and more semantically similar to x_2.

III. ACAI

As described in the original paper [1] aims for high quality interpolation. On one hand we hope that intermediate points are indistinguishable from real data. And on the other hand, we hope that intermediate points exhibit a semantically smooth morphing between endpoints. The latter is hard to enforce since these semantic characteristics are dependent on the use case. To measure these two qualities, we define metrics in section IV.

The regularizer focuses on encouraging interpolated datapoints, indistinguishable from reconstruction of real data.

To enforce this constraint ACAI introduce a critic network, as is done in Generative Adversarial Networks (GANs). The critic is fed interpolations of existing datapoints (i.e. $\hat{x}_\alpha$ as defined in Eq. 1). Its goal is to predict α from $\hat{x}_\alpha$, i.e. to predict the mixing coefficient used to generate its input. In contrast, the autoencoder is trained to fool the critic to think that α is always zero. This is achieved by adding an additional term to the autoencoder's loss to optimize its parameters to fool the critic.

Formally, let $d_\omega(x)$ be the critic network, which for a given input produces a scalar value. The critic is trained to minimize

$$\mathcal{L}_d = ||d_\omega((\hat{x}_\alpha)) - \alpha||^2 + ||d_\omega(\gamma x + (1-\gamma) g_\phi(f_\theta(x)))||^2 \quad (2)$$

The second term serves as a regularizer with two functions: First, it enforces that the critic consistently outputs 0 for non-interpolated inputs; and second, by interpolating between x and $g_\phi(f_\theta(x))$ in data space it ensures the critic is exposed to realistic data even when the autoencoder's reconstructions are poor. We found the second term was not crucial for our approach, but helped stabilize the adversarial learning process.

The autoencoder's loss function is modified by adding a regularization term:

$$\mathcal{L}_{f,g} = ||x - g_\phi(f_\theta(x))||^2 + \lambda ||d_\omega(\hat{x}_\alpha)||^2 \quad (3)$$

where λ is a scalar hyperparameter which controls the weight of the regularization term. Note that the regularization term is effectively trying to make the critic output 0 regardless of the value

of α, thereby "fooling" the critic into thinking that an interpolated input is non-interpolated (i.e., having $\alpha = 0$). As is standard in the GAN framework, the parameters θ and ϕ are optimized with respect to $\mathcal{L}_{f,g}$ (which gives the autoencoder access to the critic's gradients) and ω is optimized with respect to $\mathcal{L}_d$. The use of this regularizer is reffered as Adversarially Constrained Autoencoder Interpolation (ACAI). Assuming an effective critic, the autoencoder successfully "wins" this adversarial game by producing interpolated points which are indistinguishable from reconstructed data.

IV. Metrics

A. Mean Distance

Finding the closest image among all possible 3d human poses is infeasible ; instead we use a size-D collection of human poses $\mathcal{D}$ which correspond to the whole dataset used. Then, we match each pose in the interpolation to a real datapoint by finding

$$q_n^* = arg\min_q D_{n,q} \tag{4}$$

for $n \in \{1, ..., N\}$, where $D_{n,q}$ is the euclidean distance between $\hat{x}_n$ and the q^{th} entry of $\mathcal{D}$. To capture the notion of "intermediate points looks realistic" in Table I, we compute

$$MeanDistance(\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_N\}) = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{n=1}^{N} D_{n,q_n^*} \tag{5}$$

where we average over an interpolation of N steps and over a batch of B interpolation sequences where endpoints has been randomly selected.

B. Interpolation in Latent Space and smoothness

To evaluate each model, we randomly select $N = 10k$ samples $\{\mathcal{P}_1^t, ..., \mathcal{P}_N^t\}$ from the test set, and, for each pair, we construct the corresponding interpolation sequence $\{z_0^r, ..., z_T^r\}$ and $\{\mathcal{P}_1, ..., \mathcal{P}_N\}$. We use the mean per-vertex position error (Eq. 8) between human poses $\mathcal{P}_i$ and $\mathcal{P}_j$ and compute pair distances between their human poses by $\Delta(\mathcal{P}_i, \mathcal{P}_j)$, which we represent by Δ_{ij}.

The smoothest minimal transformation Δ_{ij} between every consecutive bodies is equal to $\Delta_{0T} = \frac{\Delta(\mathcal{P}_0^i, \mathcal{P}_T^i)}{T}$. For different initial pairs, this value can be drastically different, as the corresponding bodies might be very close to or very far from each other. Hence, we normalize the transformation Δ_{ij} of every sequence by the expected average transformation Δ_{0T} , yielding $\bar{\Delta}_{ij}$, which should be 1 in the smoothest case. Note, however, that such an ideal case can typically only be achieved by going through physically impossible poses, for instance by shrinking the arms to go from a body with arms up to one with arms down. Hence, actual transformations will typically obtain values higher than 1, but a good latent space should nonetheless yield values as constant as possible throughout the entire interpolation steps, indicating a smooth gradual transition.

We illustrate the behavior of different models in Figure 1 and 2, where we average the consecutive interpolation distances $\bar{\Delta}_{ij}$ across all pairs to obtain a sequence $\bar{\bar{\Delta}}_{ij}$. The closer the curve is to being horizontal, the smoother is the interpolation.

1) Smoothness Ratios (R_{ij}): To measure the smoothness of an interpolation, we compute the ratio between the maximal and minimal transitions in a sequence $\bar{\Delta}_{ij}$, i.e.,

$$R_{ij} = \frac{max(\bar{\Delta}_{ij})}{min(\bar{\Delta}_{ij})} \tag{6}$$

We report the mean, variance and median of the ratios R_{ij} in Table I

2) Average Normalised Change over Interpolations (ANCI): We also measure the quantity of change over interpolations of a model by averaging the sequence $\bar{\bar{\Delta}}_{ij}$.

$$ANCI = mean(\bar{\bar{\Delta}}_{ij}) \tag{7}$$

C. Dataset Coverage

An ideal latent representation should cover the whole space of realistic human poses and nothing else. In other words, it should have good *Recall* and *Precision*. By recall, we mean that all samples in the training set should be well approximated by poses our model generates. By precision, we mean that these generated poses should never deviate too far from the training set. While recall indicates how well the generated samples cover the dataset distribution, precision indicates how realistic the generated samples are. We define these metrics as follows.

1) Recall Distance: To evaluate recall, we use our pose generator to produce H36M poses. Hence, for all models we produce a body pose $\mathcal{P} = g_\phi(\mathbf{z})$ given a sampled latent vector. We first generate $180k$ samples from the pose generator of each

model. Then, given a ground-truth pose $\mathcal{P}_t$ from training and test set, we select the generated pose $\mathcal{P}_r = g_\phi(z_r)$ with minimum vertex-to-vertex distance

$$d(\mathcal{P}^r, \mathcal{P}^t) = \frac{1}{N_{verts}} \sum_{v=1}^{N_{vert}} ||\mathcal{P}_v^r, \mathcal{P}_v^t||_2 \qquad (8)$$

where v sums over the vertices.

We then repeat this operation for 10k bodies randomly sampled from the training and test set. We report the mean, variance and median of the resulting distances in the row 'Recall' of Table I.

2) Precision Distance: Our approach to computing precision mirrors the one we used for recall. We randomly generate $10k$ latent points $\mathbf{z}^r$ from every model, and for each sample $\mathcal{P}^r = g_\phi(\mathbf{z})$, look in the training and test datasets for the nearest neighbor in terms of the distance given in Eq. 8. If the latent representation only produces poses similar to those seen in training, this distance should be consistently small.

V. Results

A. ANCI

The ANCI can be interpreted as a measure of average latent space gradient between real poses codes. This measure may increase if the latent space quality decreases and becomes more chaotic. In some cases where other quality metrics of latent space are good and beat the baseline, arises the question that this also means that the generator can generate more diverse and meaningful poses. With both a latent space dimension of 8 and 16, we can observe that this metric consistently increases relative to λ.

B. Smoothness

ACAI is expected to derive latent space generating smoother interpolation. But in this application ACAI seems to accentuate "High gradient zone" in latent space. The cumulated effect is visualized in fig 1 and 2 where a higher lambda seems to accentuate "high gradient" zones in the latent space. Individual interpolation smoothness graphs show very different behaviors and type of "high gradient" zones that appear as peak or plateau in different location of the interpolation. When averaged together , the aggregated smoothness graphs shows that these higher gradient zone' centers and width follow a distribution close to normal visually.

C. Mean Distance

Mean Distance is a relatively stable metric with ACAI. The primary objective of ACAI is expected to improve this metric. In parallel to the drastic change in latent space structure, when λ increases Mean Distance seems to decrease up to a certain value beyond which it increases I.

D. Dataset coverage

Precision and recall distances show the degree of similarity between latent space and images in the dataset. In table I, they seem correlated and no trend can be observed relative to the hyperparameter λ. Nonetheless, these metrics along with other metrics can help infer a good latent space behavior. In general if these distance are drastically higher, a chaotic latent space can be observed.

E. Possible explanations and further work

If we project real datapoints and fake datapoints generated by the model in a 2D plane with T-SNE like in fig3 of [2] one can expect to observe real points more concentrated in some clusters. When we increase hyperparameter of ACAI, points in between where real points are rarer are more subject to "contributions" from more distant points. There may be a balance between contributions of points intra-cluster and extra-cluster. In the smoothness graphs obtained 1and 2 we can observe that when the λ are close to the maximum value experimented graphs starts to look more "sinusoidal". Having tested a range of values it would be interesting to test with a wider range of values to observe to what extent the patterns of these graphs can change.

We also experienced the limits of simple metrics to express the semantic quality of the latent space. It is hard to tell if rougher interpolation behavior can exhibit better latent space qualities. More visualisations can be useful and spending resources to develop powerful visualisation tools can be useful for this purpose.

During this research, we also observed that a latent space dimension of 16 produced more chaotic latent space than of 8. Furthermore, increasing the weight of ACAI (λ) with a latent space dimension of 16 in our example showed to degrade the latent space drastically.

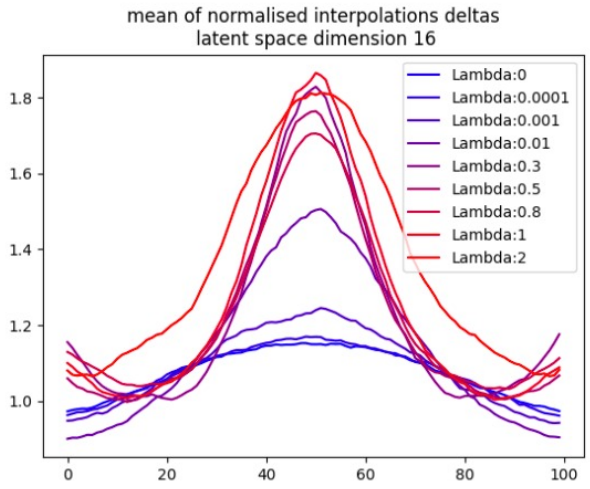


Fig. 1: Average normalized pose transformation of consecutive generated samples in interpolations between source and target examples. The effect of ACAI is depicted by plotting the results of each model with latent dimension of 16. In abscissa, Normalised per-vertex error. In ordinate, interpolation step t in $\alpha = \frac{t}{T}$ where $T = 100$ steps

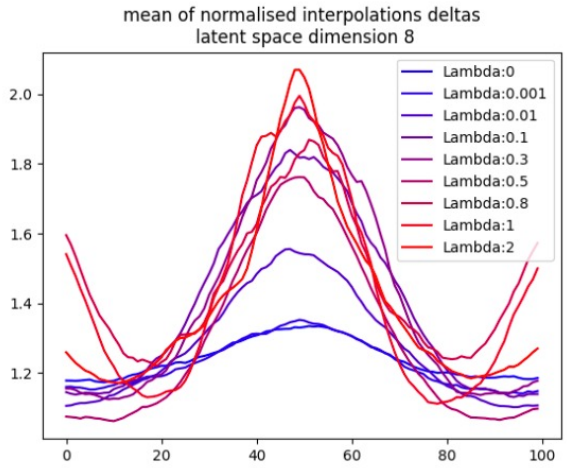


Fig. 2: Average normalized pose transformation of consecutive generated samples in interpolations between source and target examples. The effect of ACAI is depicted by plotting the results of each model with latent dimension of 8. In abscissa, Normalised per-vertex error. In ordinate, interpolation step t in $\alpha = \frac{t}{T}$ where $T = 100$ steps

VI. Conclusion

In conclusion, we experimented ACAI with different "vanilla" autoencoders and a semantically complex application of generating a good quality latent space of 3d human poses. We showed some effects of the hyperparameter on the latent space structure and obtained latent space closer to the dataset but with more diversity than the baseline. We show that doing further research with ACAI can help understand how to improve a latent space and its qualities and find better metrics.

References

[1] D. Berthelot, C. Raffel, A. Roy, and I. Goodfellow, "Understanding and improving interpolation in autoencoders via an adversarial regularizer," 2018.

[2] Anonymous, "Adversarial parametric pose prior," 2021.

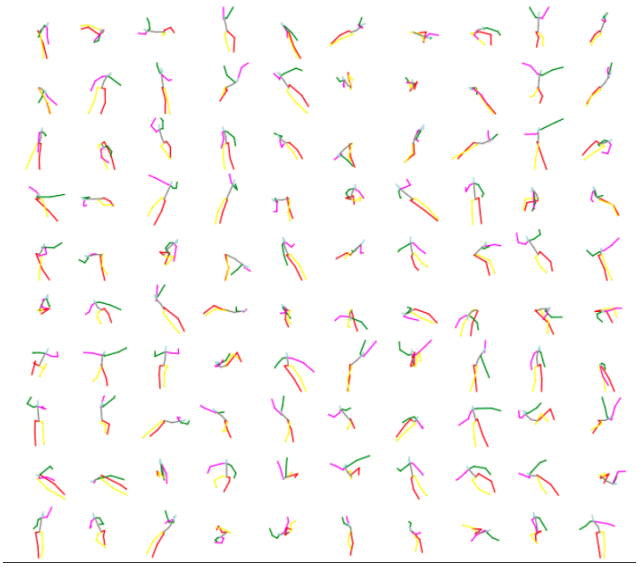
VII. Annex

A. Results Table

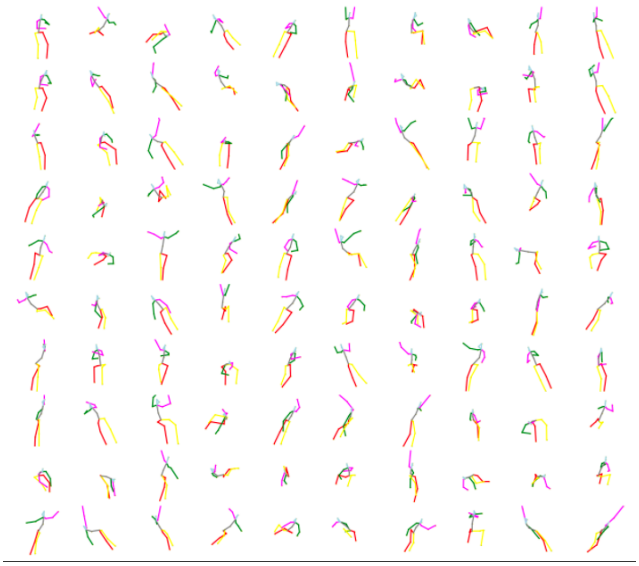
Latent space $d = 16$									
λ	0	0.0001	0.001	0.01	0.1	0.3	0.5	0.8	1
ANCI	**1.08 ± 0.00**	1.08 ± 0.01	1.09 ± 0.01	1.14 ± 0.04	1.20 ± 0.08	1.23 ± 0.07	1.22 ± 0.06	1.23 ± 0.05	1.25 ± 0.08
median	1.08	1.09	1.09	1.09	1.09	1.10	**1.07**	1.11	1.10
Ratios	**1.86 ± 0.19**	1.99 ± 0.25	2.34 ± 0.40	3.31 ± 3.07	4.65 ± 6.74	4.17 ± 5.25	3.39 ± 2.17	3.12 ± 2.26	4.05 ± 4.87
median	**1.75**	1.89	2.25	2.74	4.24	3.70	3.06	2.75	3.62
Mean Distance	0.45 ± 0.02	0.46 ± 0.02	0.44 ± 0.02	0.43 ± 0.02	**0.43 ± 0.02**	0.45 ± 0.02	0.48 ± 0.03	0.48 ± 0.02	0.49 ± 0.03
median	0.43	0.44	0.42	0.41	**0.40**	0.43	0.46	0.45	0.46
Precision	1.94 ± 0.12	**1.78 ± 0.10**	2.11 ± 0.15	1.86 ± 0.12	2.67 ± 0.15	3.76 ± 0.15	2.04 ± 0.14	2.75 ± 0.80	3.26 ± 0.89
distance	1.95	**1.79**	2.12	1.86	2.68	3.72	2.05	2.63	3.19
median									
Recall distance	1.52 ± 0.02	**1.44 ± 0.02**	1.53 ± 0.03	1.46 ± 0.02	2.19 ± 0.02	1.89 ± 0.03	1.71 ± 0.01	1.63 ± 0.03	1.77 ± 0.04
median	1.53	**1.45**	1.53	1.46	2.19	1.90	1.72	1.65	1.79
Latent space $d = 8$									
λ	0	0.001	0.01	0.1	0.3	0.5	0.8	1	2
ANCI	1.24 ± 0.00	**1.23 ± 0.01**	1.28 ± 0.02	1.41 ± 0.06	1.41 ± 0.09	1.30 ± 0.06	1.45 ± 0.04	1.42 ± 0.08	1.40 ± 0.07
median	1.22	1.21	1.24	1.31	1.27	**1.18**	1.40	1.34	1.30
Ratios	**1.96 ± 0.23**	2.23 ± 0.34	3.00 ± 0.20	4.98 ± 8.85	5.51 ± 12.0	3.78 ± 3.44	5.90 ± 13.9	5.90 ± 13.0	5.44 ± 15.6
median	**1.87**	2.14	2.62	4.43	4.90	3.38	5.47	5.90	4.18
Mean Distance	0.45 ± 0.03	0.43 ± 0.02	0.44 ± 0.03	0.43 ± 0.02	**0.43 ± 0.02**	0.46 ± 0.02	0.44 ± 0.02	0.46 ± 0.02	0.48 ± 0.03
median	0.42	0.41	0.41	0.41	**0.40**	0.44	0.41	0.43	0.45
Precision	1.05 ± 0.10	1.02 ± 0.10	0.90 ± 0.07	1.47 ± 0.23	1.73 ± 0.40	**0.86 ± 0.07**	4.89 ± 9.82	4.90 ± 9.81	0.97 ± 0.08
distance	1.05	1.00	0.89	1.45	1.68	**0.84**	4.23	4.31	0.94
median									
Recall distance	0.80 ± 0.02	0.81 ± 0.02	0.74 ± 0.02	0.97 ± 0.03	0.96 ± 0.03	**0.72 ± 0.02**	1.20 ± 0.06	1.17 ± 0.04	0.80 ± 0.02
median	0.80	0.80	0.72	0.97	0.96	**0.70**	1.22	1.16	0.78

TABLE I: Results for each model

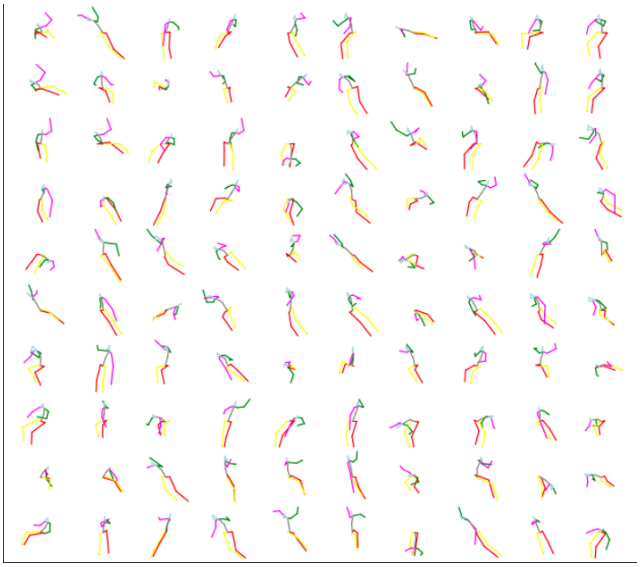
B. Reconstructed random samples from latent space



(a) of AE with latent space dimension 8 and a λ of 0



(b) of AE with latent space dimension 16 and a λ of 0.5



(c) of AE with latent space dimension 16 and a λ of 2

Fig. 3: Reconstructions of different models from randomly sampled latent codes

C. Individual Interpolation

Fig. 4: Example of interpolation from AE with latent space dimension 16 and a λ of 0, 10 intermediate interpolants. Corresponding transition distances (100-step interpolations) are provided in Fig5.

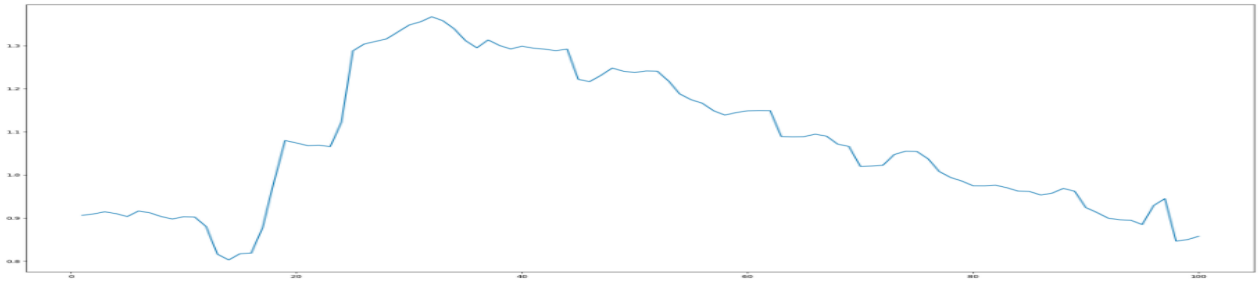


Fig. 5: Transition distances for the sequence in Fig. 4 from AE with latent space dimension 16 and a λ of 0. In abscissa, Normalised per-vertex error. In ordinate, interpolation step t in $\alpha = \frac{t}{T}$ where $T = 100$ steps

Fig. 6: Example of interpolation from AE with latent space dimension 16 and a λ of 0.1, 10 intermediate interpolants. Corresponding transition distances (100-step interpolations) are provided in Fig. 7.

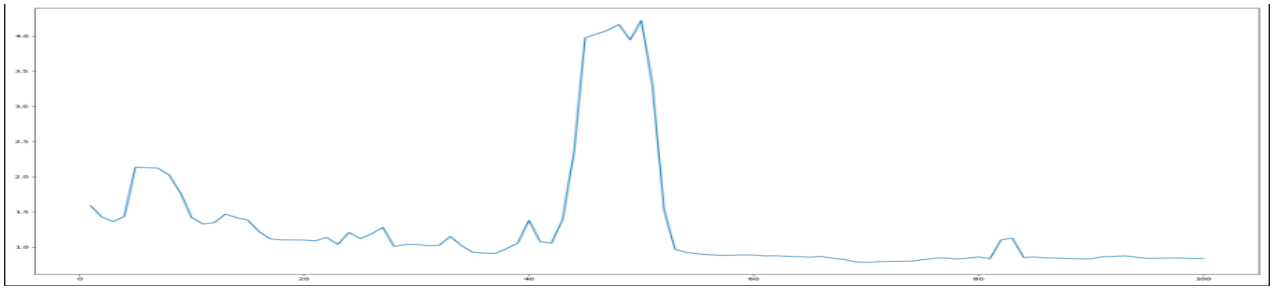


Fig. 7: Transition distances for the sequence in Fig. 6 from AE with latent space dimension 16 and a λ of 0.1. In abscissa, Normalised per-vertex error. In ordinate, interpolation step t in $\alpha = \frac{t}{T}$ where $T = 100$ steps