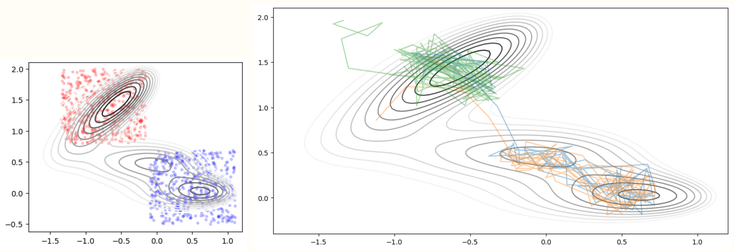
Modern autoregressive architectures for the collective variable problem (T6)

Andrej Leban

Department of Statistics, University of Michigan

Background

- Evaluating *trajectories* from a particle subject to the Müller-Brown Potential that undergoes Langevin dynamics.
- Synthetic example characteristic of many real-life molecular systems.
- Can the sequential nature of the problem be amenable to Transformer architectures?
- Train and test set generated from random starting positions with the same simulation parameters.



(a) Starting positions

(b) Example generated trajectories

Architecture

- **BERT-style encoder + projection heads**: used, e.g., for predicting protein backbone angles as part of a diffusion model [Wu et al., 2024].
- **Built manually** using the *Huggingface* BERT implementation as base [Huggingface, 2024].
- **Two projection heads**: coordinate prediction f and state classification g, both take the encodings as input.
- **Bi-directional attention**.
- **Trajectory Loss:**

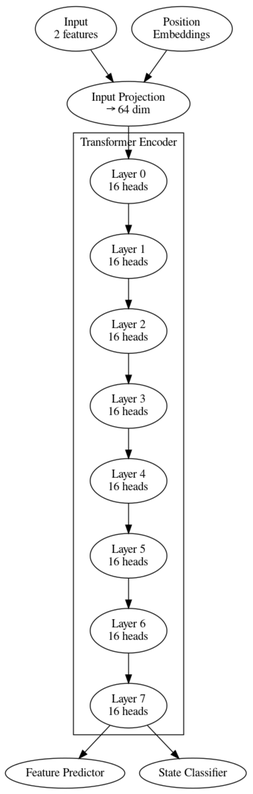
$$\mathcal{L}_{\text{traj}} = \frac{1}{N-1}\sum_{t=1}^{N-1} \|f(x_{1:N-1})_t - x_{t+1}\|^2$$

- **Total Loss:**

$$\mathcal{L} = \gamma\, \mathcal{L}_{\text{traj}} + \lambda\, \text{CrossEntropy}(g(x_{1:N}), y)$$

- **Three model sizes** evaluated:

	Small Model	Medium Model	Large Model
`inputs_to_hidden_dim`	48	192	360
`embeddings`	16.4 K	65.7 K	123 K
`encoder`	21.6 K	399 K	2.2 M
`feature_predictor`	338	4.4 K	15.0 K
`state_classifier`	338	4.4 K	15.0 K
Trainable Params	**38.7 K**	**474 K**	**2.3 M**



References

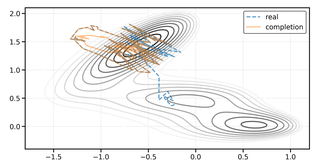
Huggingface. Transformers: State-of-the-art machine learning for NLP, CV, and more, 2024. URL `https://github.com/huggingface/transformers`. Accessed: 2024-12-04.

K. E. Wu, K. K. Yang, R. van den Berg, S. Alamdari, J. Y. Zou, A. X. Lu, and A. P. Amini. Protein structure generation via folding diffusion. *Nature Communications*, 2024.

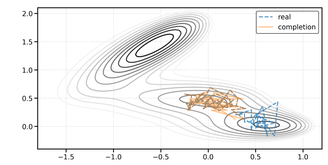
Trajectory Generation

The generation procedure:

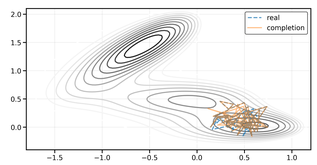
1. The model standardizes the data internally.
2. Predict all next steps for the current sequence.
3. Predicted frames are added and beginning frames dropped to maintain a sliding window of the original sequence length throughout.



(a) Small model

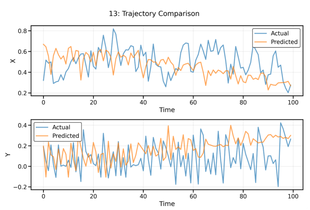


(b) Medium model

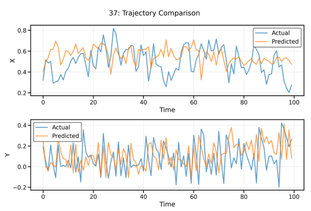


(c) Large model

- Predicting multiple frames at a time sometimes works better: pretend the last n steps of predicted output are actually predictions for the next n steps (as opposed to $[N-n+1, \ldots]$), append to the input, and repeat:



(a) Predicting 13 steps



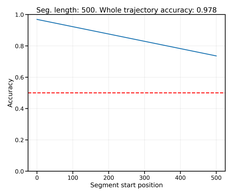
(b) Predicting 37 steps

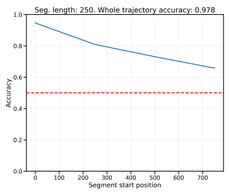
State Classification

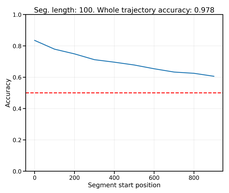
- States correspond to the two potential minima.
- The transformer encodings are shown to be a powerful base for a classifier:

	Accuracy	Precision	Recall	F1 Score
Small Model	0.9150	0.9234	0.9150	0.9154
Medium Model	0.9780	0.9788	0.9780	0.9780
Large Model	0.9910	0.9910	0.9910	0.9910

- Prediction deteriorates when prompted with a later trajectory slice:

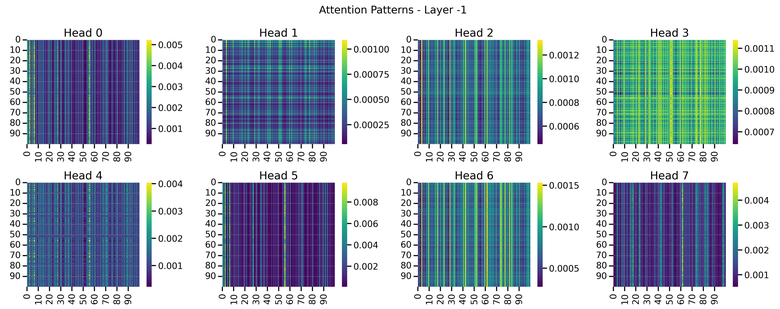




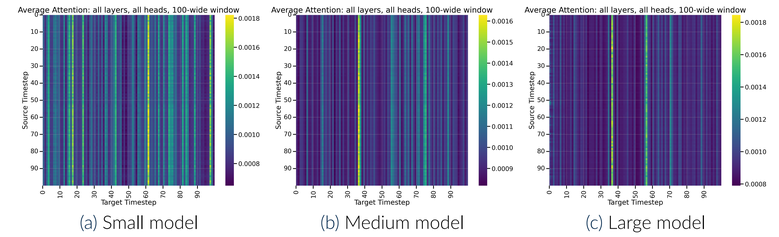


Attention Patterns

- Small model – last layer attention by head:

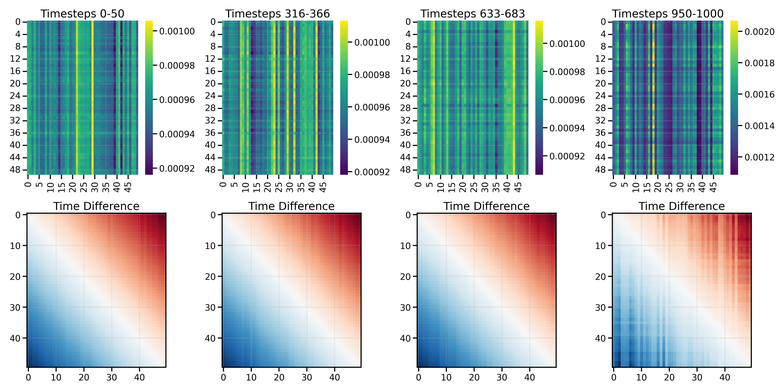


- Average attention across layers, heads with a 100-width window:



(a) Small model (b) Medium model (c) Large model

- When looking at sub-sections of a trajectory and visualizing the attention weighted by the time-difference within that section, we find that the largest model mostly attends to future and past steps *equally* throughout the trajectory:



Attention Pattern Takeaways

- The model seems to use a set number of key timesteps (*discretization*), which attend to all other timesteps equally. Perhaps this is why predicting next n steps sometimes works?
- Some heads also focus on a discrete number of target query timesteps.
- No local dependencies used, as evident by the absence of block structure.

leban@umich.edu