Advanced Time Series

Lecture 7:
Representation learning

Gleb Ivashkevich

Today

Transformers and t. s. ⟶ representation learning:

- **transformer** for t.s. forecasting model
- representation learning setup
- time-lagged autoencoder
- VAMPnets
- wrap-up

Transformer for t.s. forecasting

Transformers

Why?

- typical sequential models (RNNs) may still **not catch** temporal dynamics well
- **attention** layer may fix this
- but they are still **sequential**

Transformers are still **encoder-decoder**.

Transformers: forecasting

[Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting]()

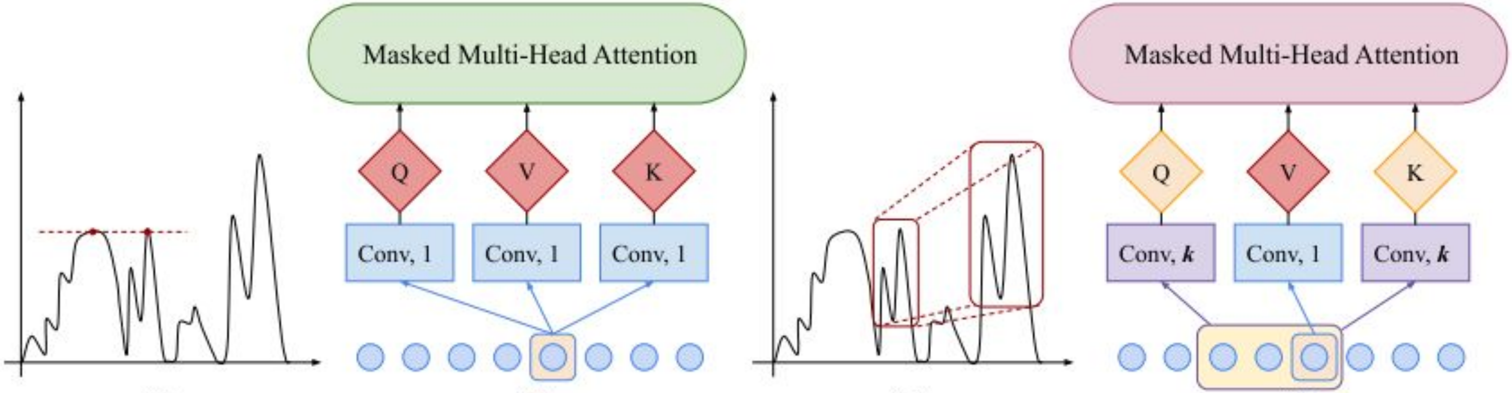
- innovation: **convolutional attention**
  (queries, keys and values are computed by conv layer)
- **attention memory bottleneck:** use smart masking
- **learnable positional encodings**
- **simpler attention**

Transformers: forecasting

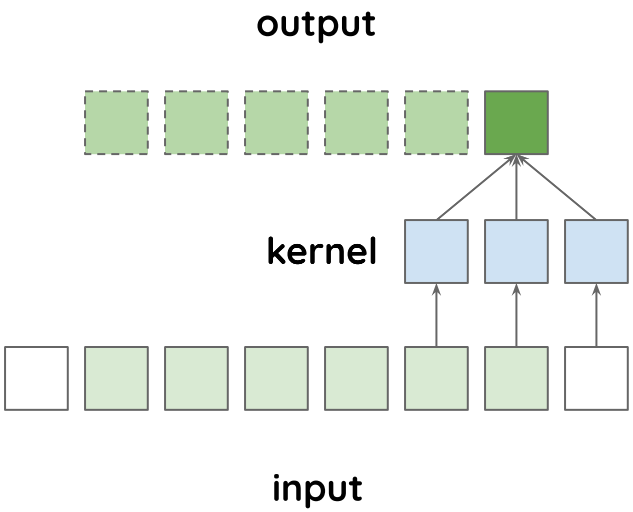
[Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting]

- decoder-only mode: **similar to DeepAR**
- **probabilistic forecasts** (DeepAR inspired)
- **causal convolutions**
- hourly power consumption dataset

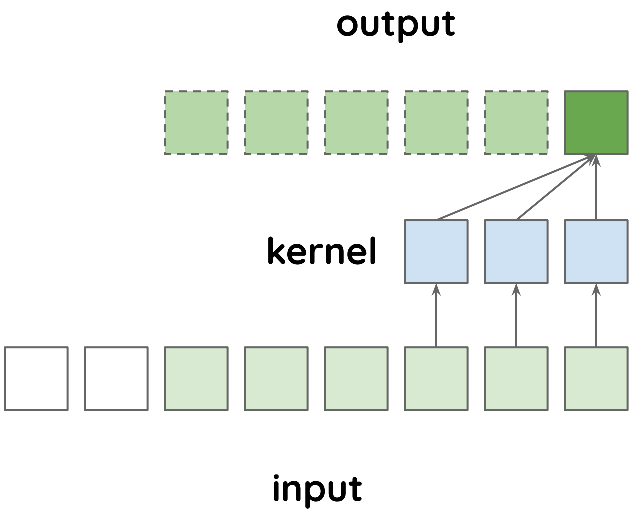
Recap


Masked Multi-Head Attention
Q
V
K
Conv, 1
Conv, 1
Conv, 1
Masked Multi-Head Attention
Q
V
K
Conv, k
Conv, 1
Conv, k
(a) (b) (c) (d)

Causal convolutions impl.

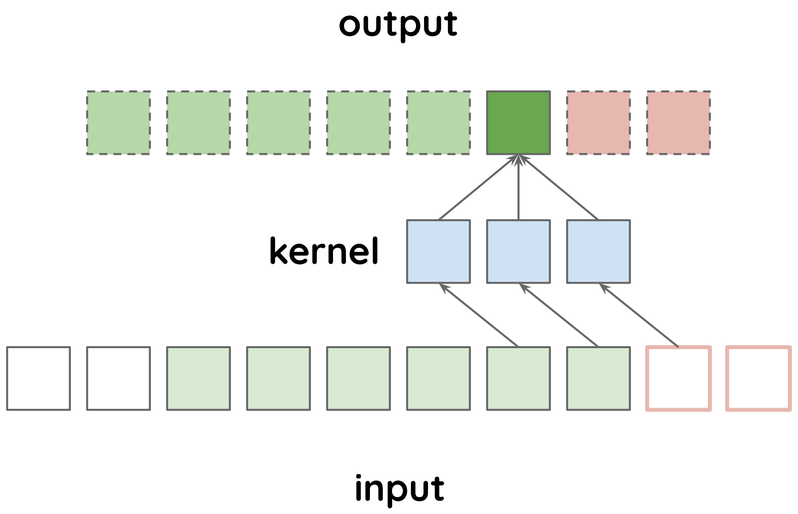


non-causal

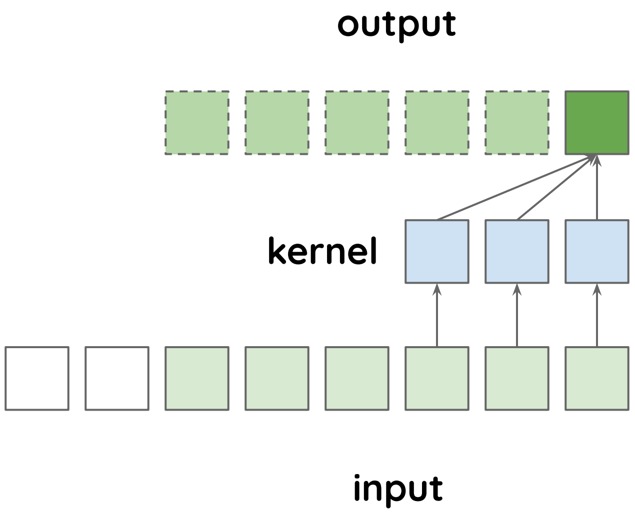


causal:
not available in PyTorch

Causal convolutions impl.

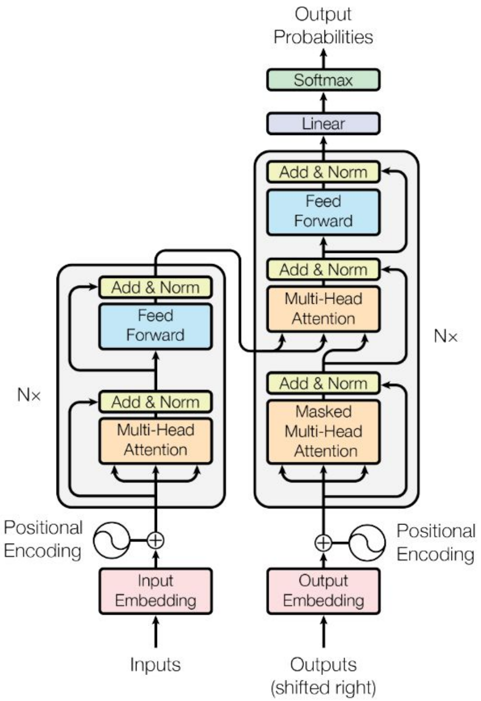


non-causal,
overpadded



causal

Positional encoding



- allows to attend both absolute and relative positions
- Additive!

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Image: [Attention Is All You Need]

Transformers

Benefits:

- allow for parallelization
- do not limit other architectural ideas
- add interpretability proxy
- much easier to reason about

Representation learning

Representations for t.s.

When:

- highly dimensional time series with complex patterns
- barely interpretable

Why:

- denser
- hopefully, provide some insights into structure
- simplify forecasting, classification and t.t.e.: substitute for pre-training

Representations for t.s.

Applications:

- manufacturing data
- molecular dynamics data
- various medical data

Naive: PCA

When:

- simple linear dependencies between[(pointwise!)] covariates

Why not:

- you never know if it’s linear or not
- temporal information is not used (neither short-term, nor long-term)

Reasonable: TICA

Time-lagged **I**ndependent **C**omponent **A**nalysis: temporal extension of PCA

How:

- instead of solving eigenvalue problem for covariance matrix, solve it for auto-covariance matrix:

$$C_{ij}(\tau) = \frac{1}{T-\tau-1} \sum_{t=1}^{t=T-\tau} x_i(t) x_j(t+\tau)$$

Reasonable: TICA

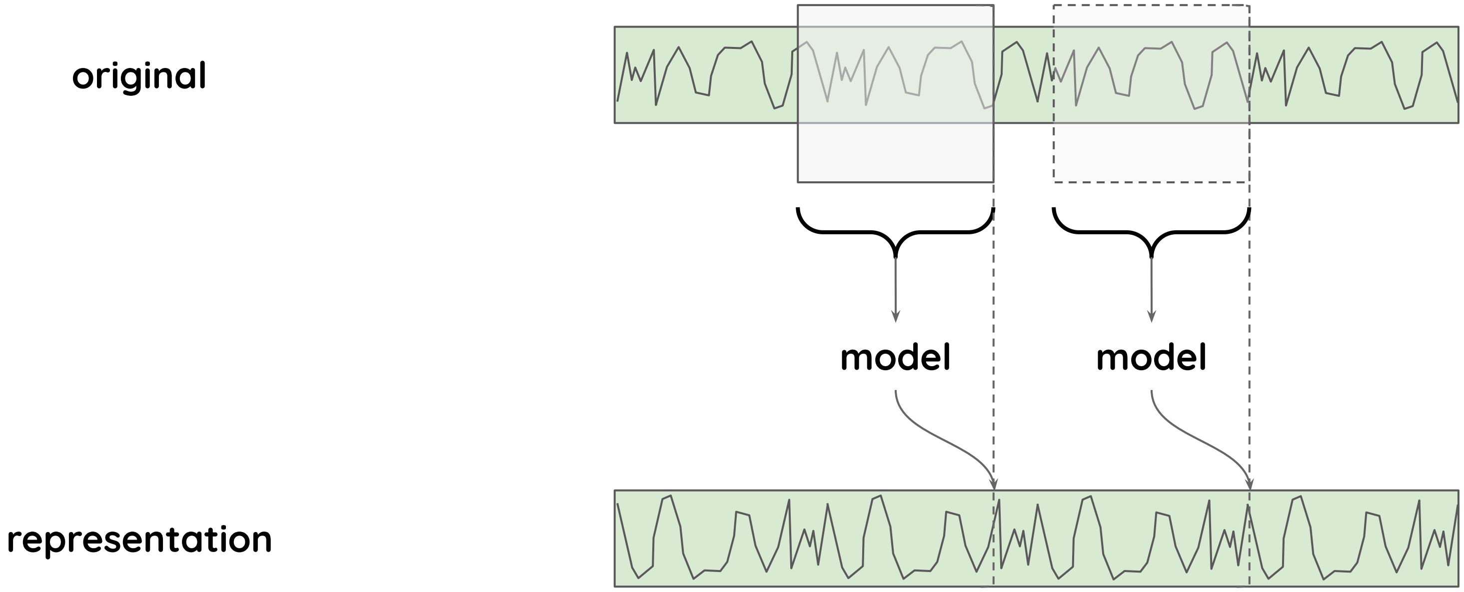
Pros:

- temporal information is partially included
- more meaningful components

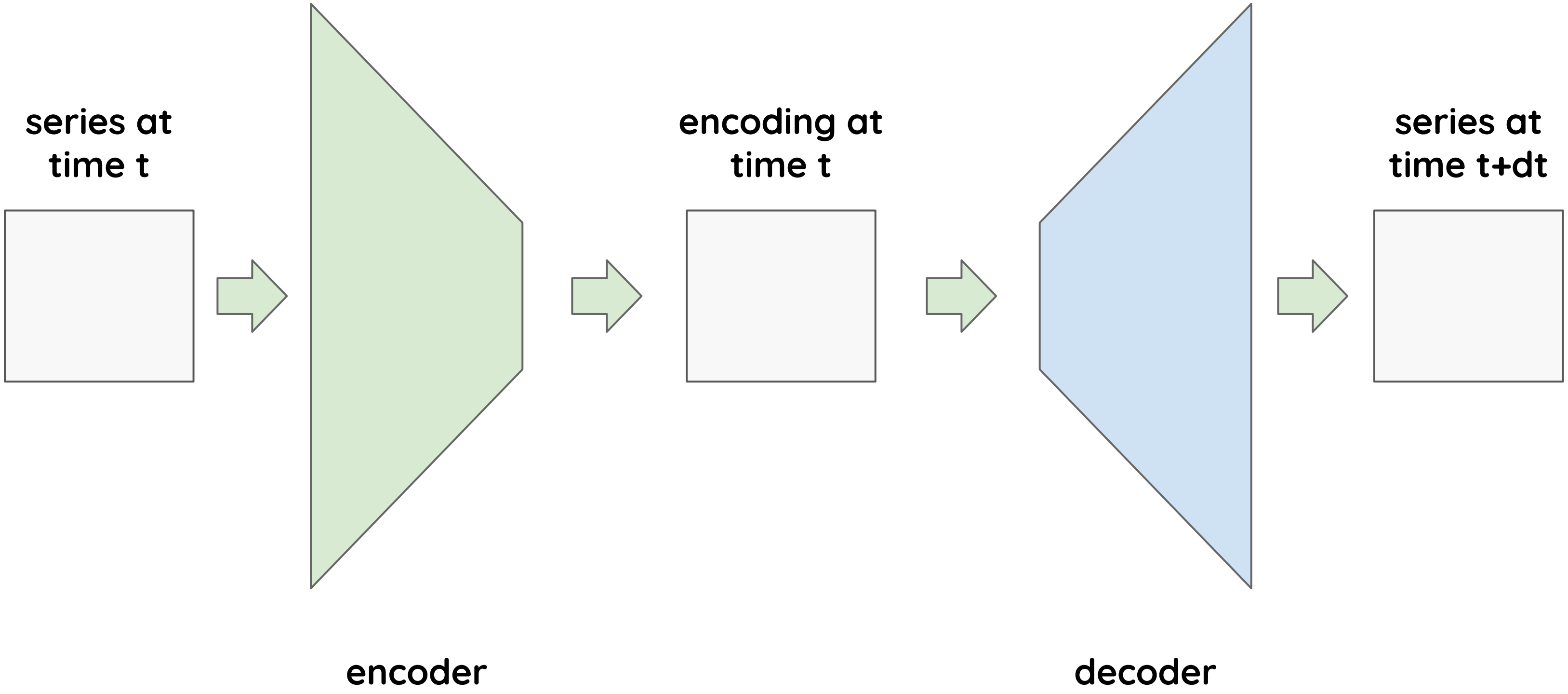
Cons:

- still linear
- no patterns are accounted for: it’s pointwise

Windowed representation



Time-lagged autoencoder



Time-lagged autoencoder

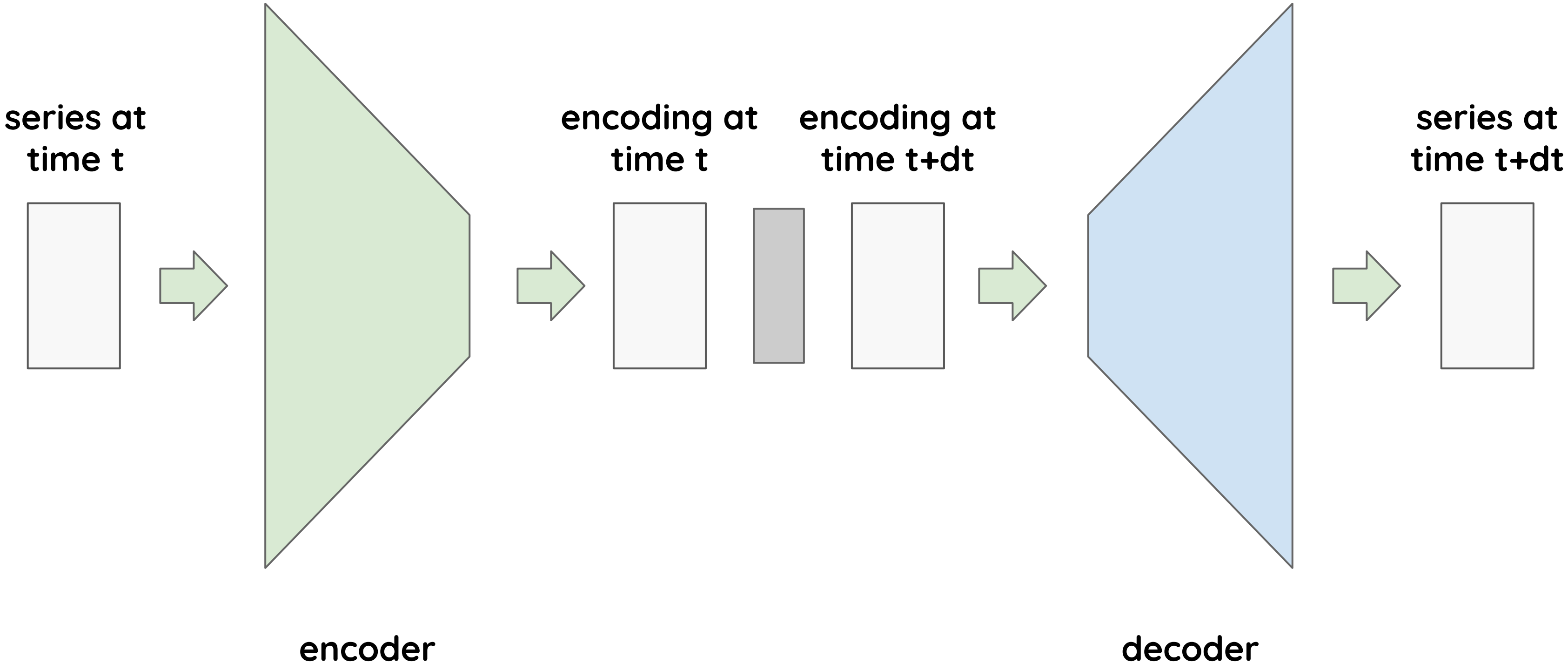
Pros:

- conceptually simple
- can contain any blocks needed (CNN, RNN, etc.)
- trained with MSE

Cons:

- may not find problem-specific representation
- no fundamental guarantees

TLA with propagator



TLA with propagator

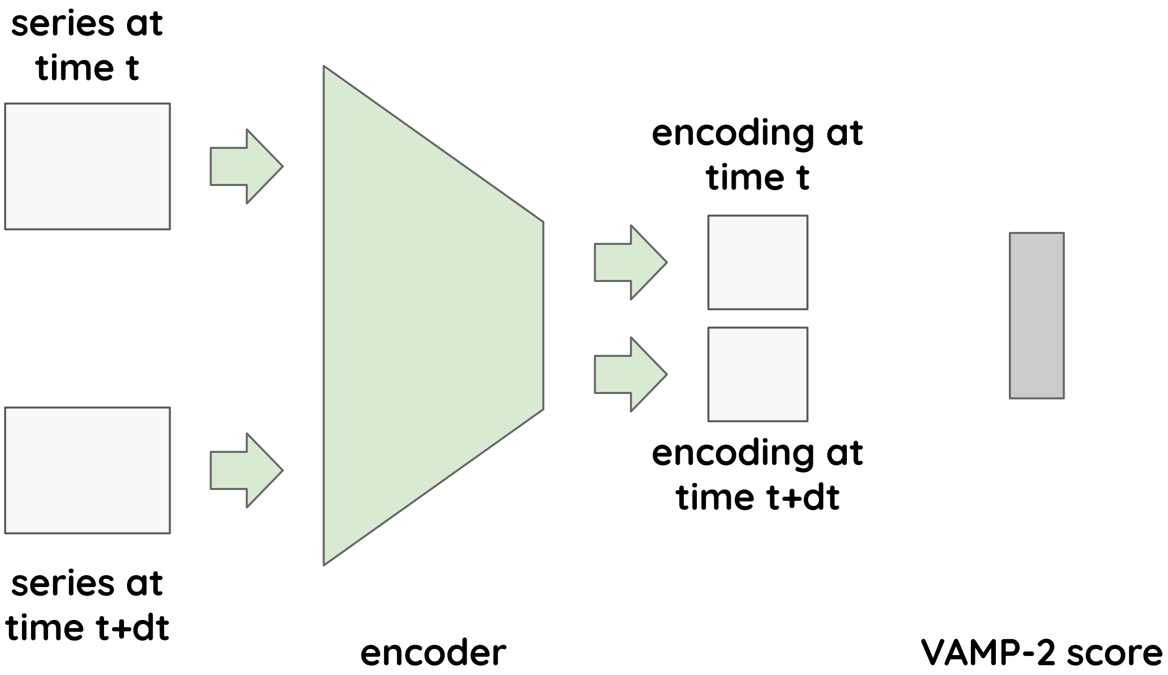
Pros:

- attempts to estimate (non-linear) dynamics

Cons:

- can be hard to train
- must be trained with several lags (as it's impossible to separate propagator from encoder and decoder)
- still ad hoc

VAMPnets


series at
time t
encoding at
time t
encoding at
time t+dt
series at
time t+dt
encoder
VAMP-2 score

VAMPnets

Pros:

- fundamentally validated (Koopman operator, etc.)
- provides hierarchy of relaxation times
- filters noisy, uncorrelated components

Cons:

- can be hard to train
- selection of parameters may be problematic

VAMPnets

VAMP-2 score:

- covariance matrices are calculated over transformed coordinates
- can be optimized directly

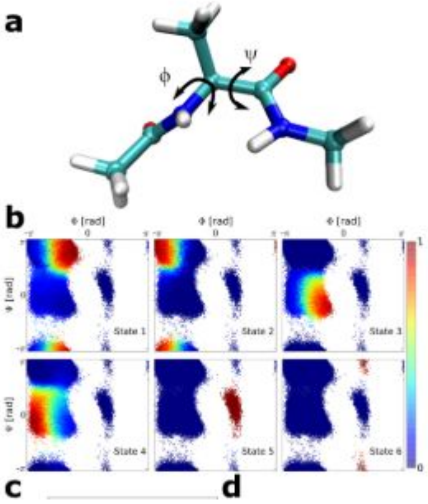
$$R = ||C_{00}^{-1/2} C_{0\tau} C_{\tau\tau}^{-1/2}||$$

[VAMPnets: Deep learning of molecular kinetics]

VAMPnets

Molecular dynamics:

- collective variables are important for modeling



Wrap-up

What we've learned

- **forecasting:** simple ED, probabilistic, transformers
- **classification**
- **survival analysis**
- **some representations learning**

Next:

- generative models, etc.

questions?