Unsupervised machine learning of topological phase transitions from experimental data

N. Käming, **Anna Dawid**, K. Kottmann, M. Lewenstein,
K. Sengstock, A. Dauphin, and C. Weitenberg

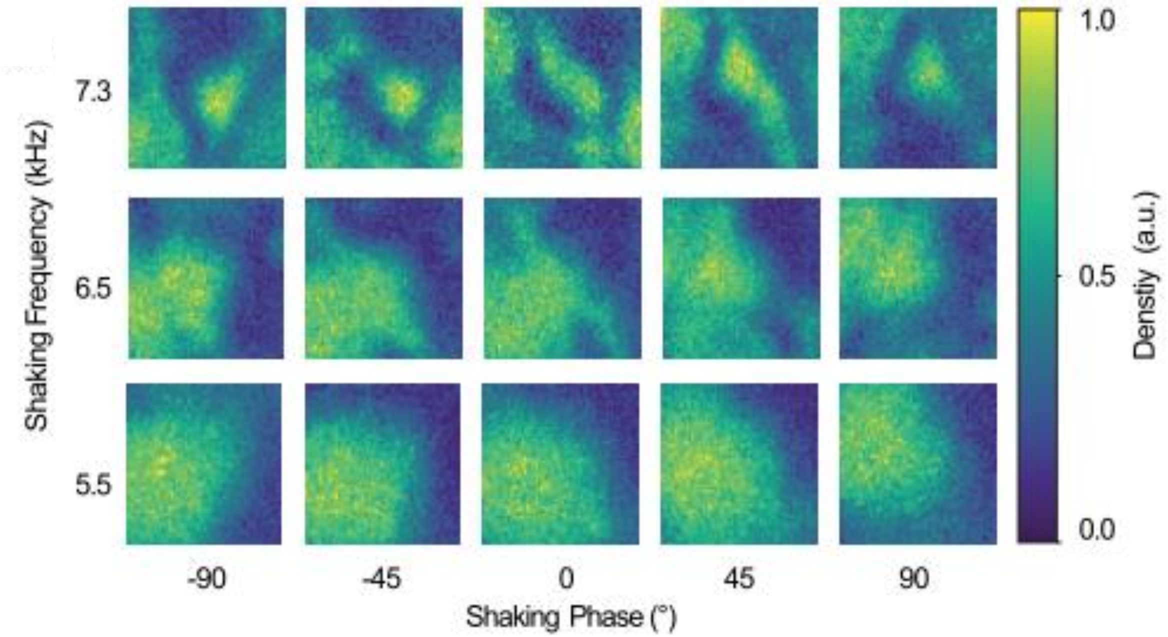
Mach. Learn.: Sci. Technol. 2 035037 (2021)

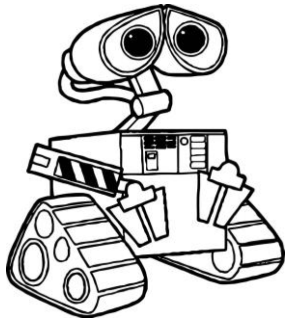
Universität Hamburg

ICFO

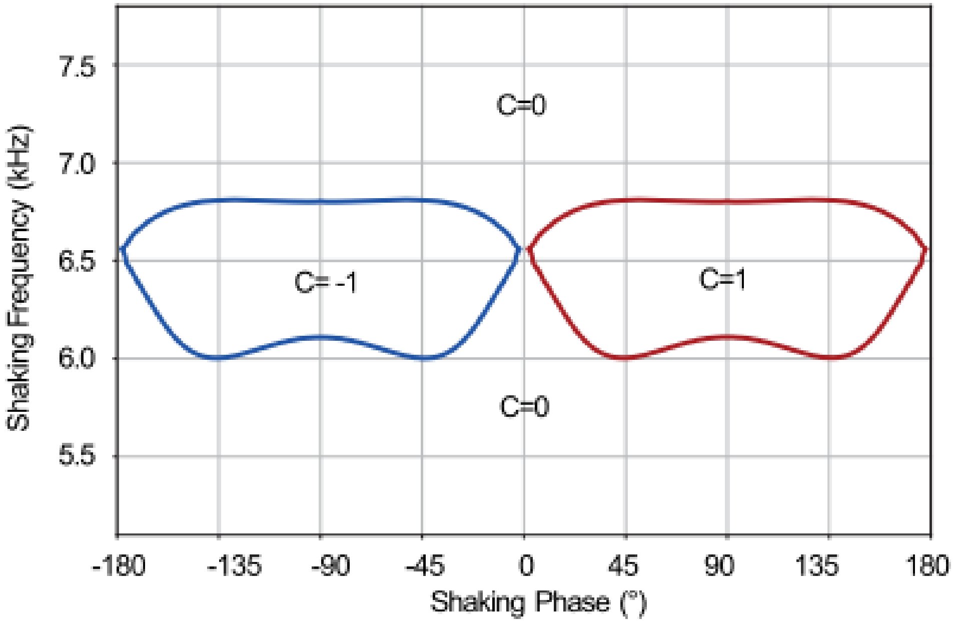
The Institute of Photonic Sciences

experimental data





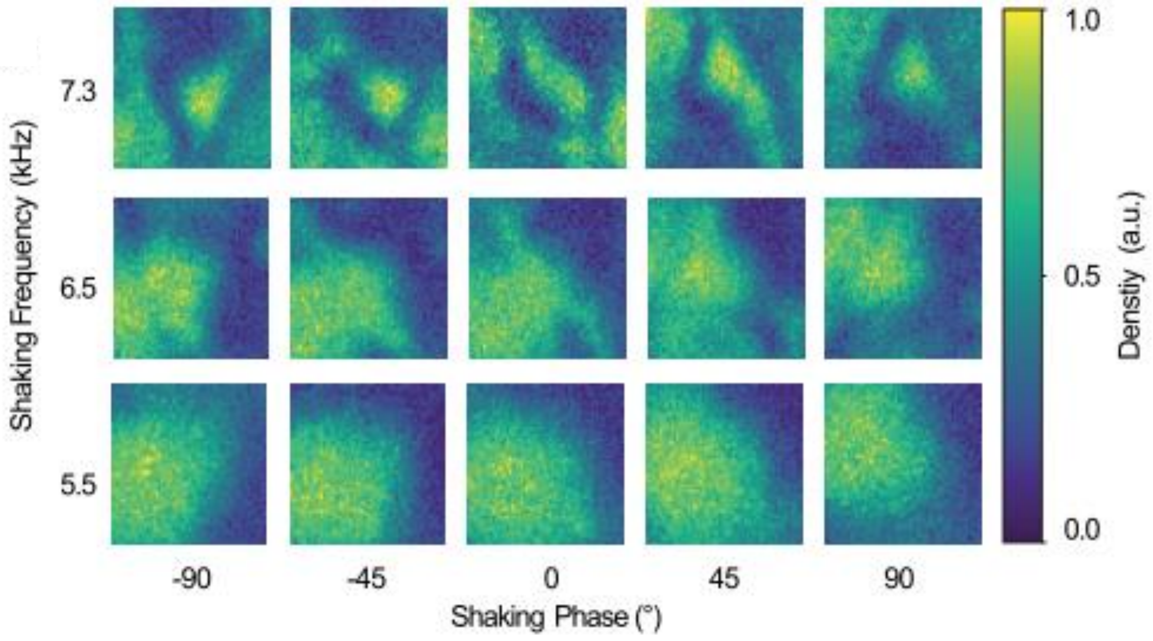
phase diagram

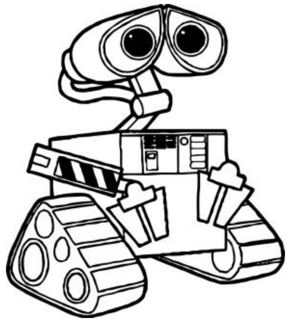


What we want to achieve

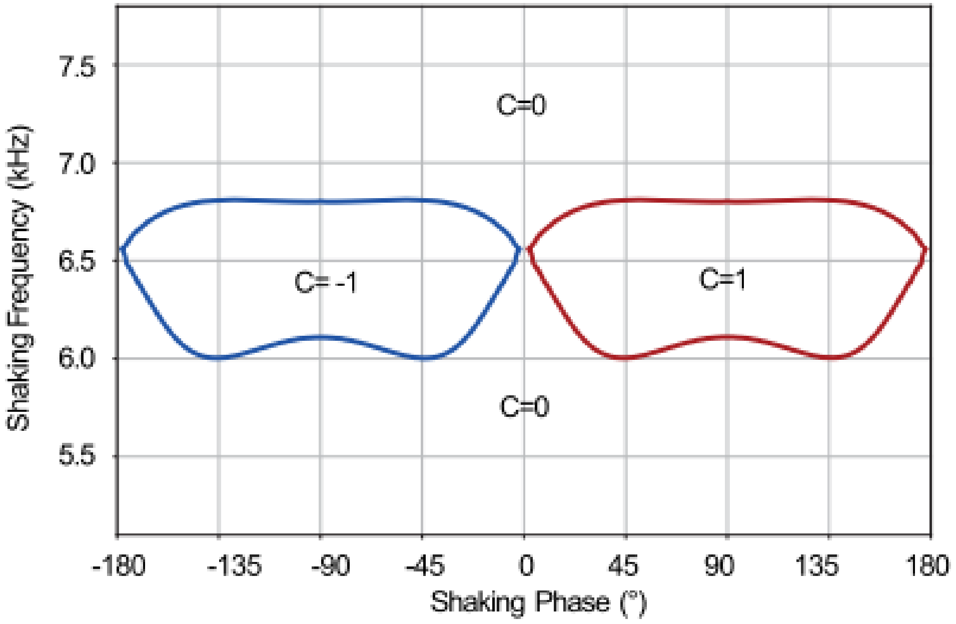
Why it's ambitious:

experimental data





phase diagram



- experimental data is noisy
- presence of additional effects of experimental implementation, which don't change the underlying physics, but may confuse the ML model
- topological models are characterized by global order parameters which are extremely challenging for ML

Outline

methods

- Ultracold system and experimental data
- ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions

results

- Unsupervised methods for raw data
- Micromotion phase removal (supervised ML)
- Unsupervised methods for postprocessed data
- Similarity analysis with influence functions (supervised ML)

Outline

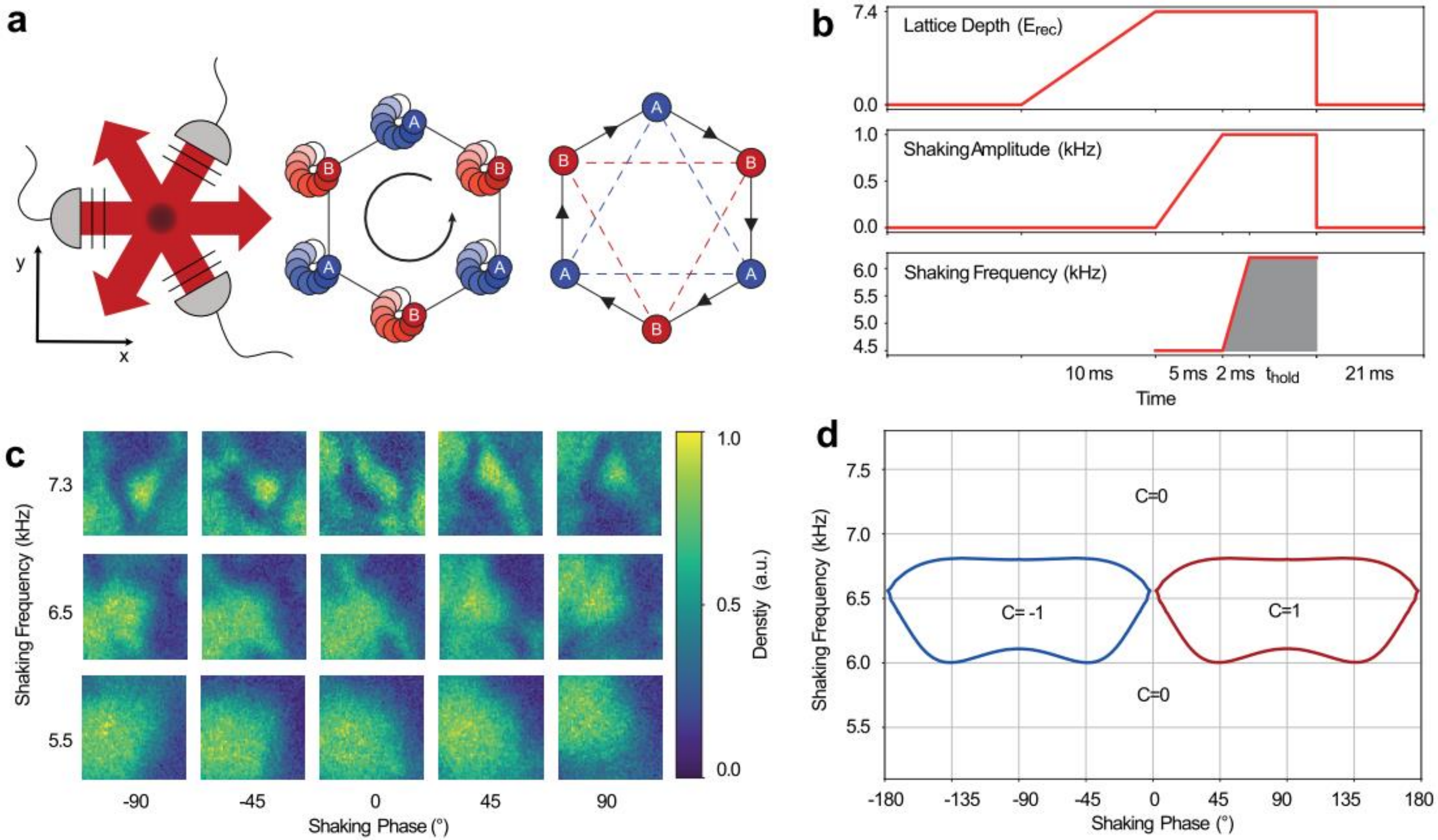
methods

results

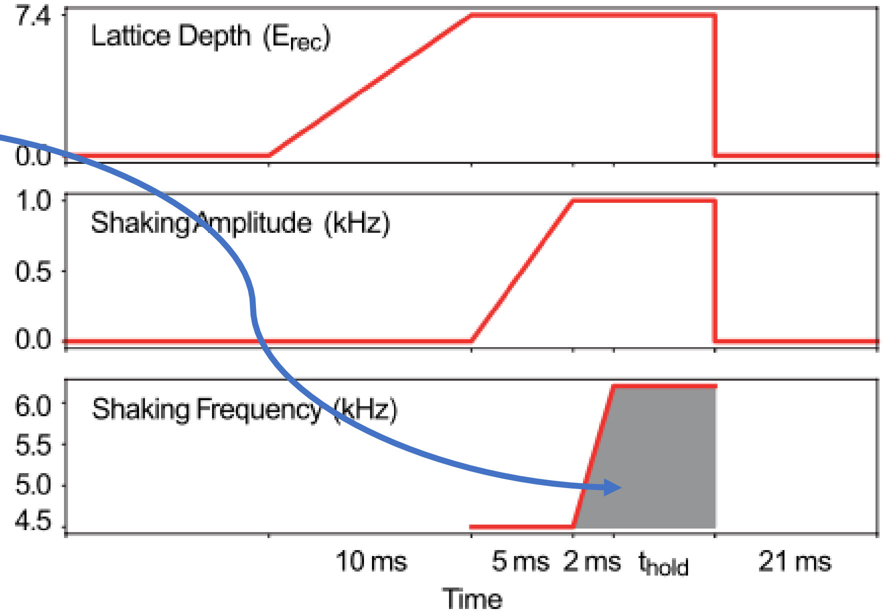
- Ultracold system and experimental data
- ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions
- Unsupervised methods for raw data
- Micromotion phase removal (supervised ML)
- Unsupervised methods for postprocessed data
- Similarity analysis with influence functions (supervised ML)

Topological Haldane model

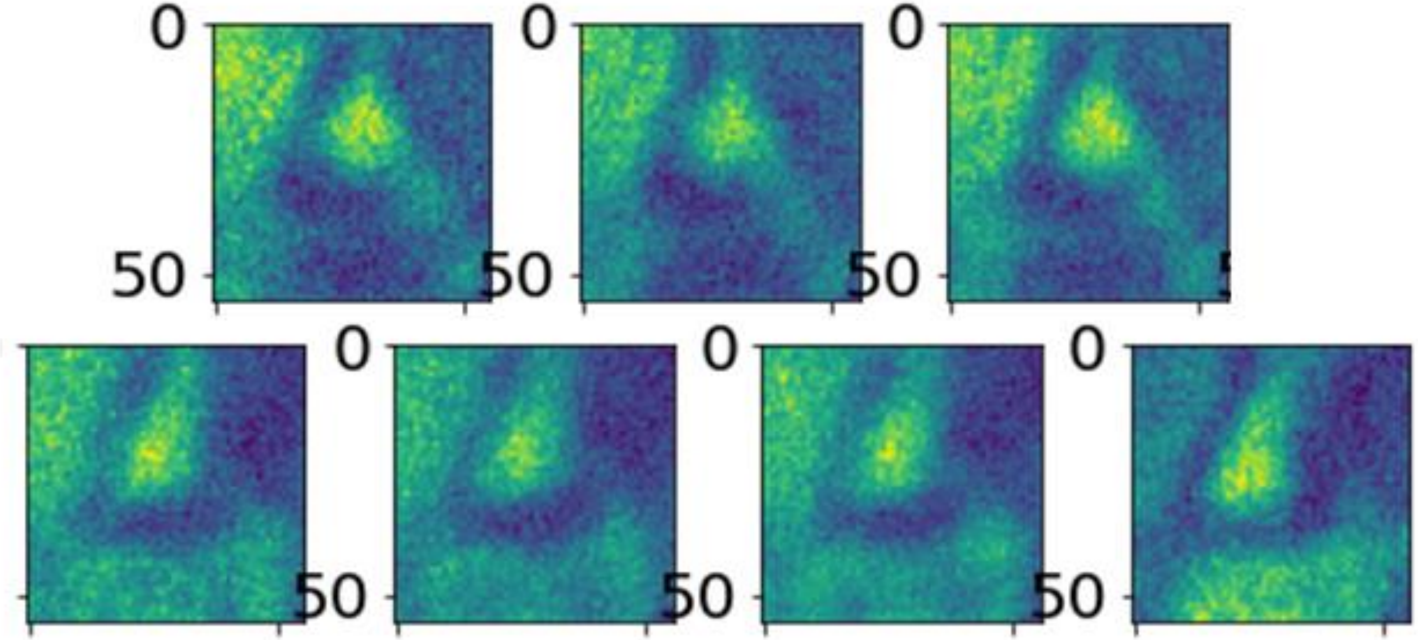
realized via Floquet-driving of ultracold fermions (^{40}K) in a honeycomb lattice



Micromotion phase



shaking frequency = 7.4 Hz
shaking phase = 90˚
different micromotion phases



Outline

methods

Ultracold system and experimental data

ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions

results

Unsupervised methods for raw data

Micromotion phase removal (supervised ML)

Unsupervised methods for postprocessed data

Similarity analysis with influence functions (supervised ML)

Machine learning

Machine learning (ML) studies algorithms

- whose performance **improves with data** ("learning from experience")
- which solve problems **without being programmed** how to solve them *explicitly*

Mitchell, McGraw Hill, 1997

Types of machine learning

Supervised – machine learns on pairs of input and output data

Unsupervised – machine groups and interprets basing just on the input data

Reinforcement – algorithm learns to react to an environment

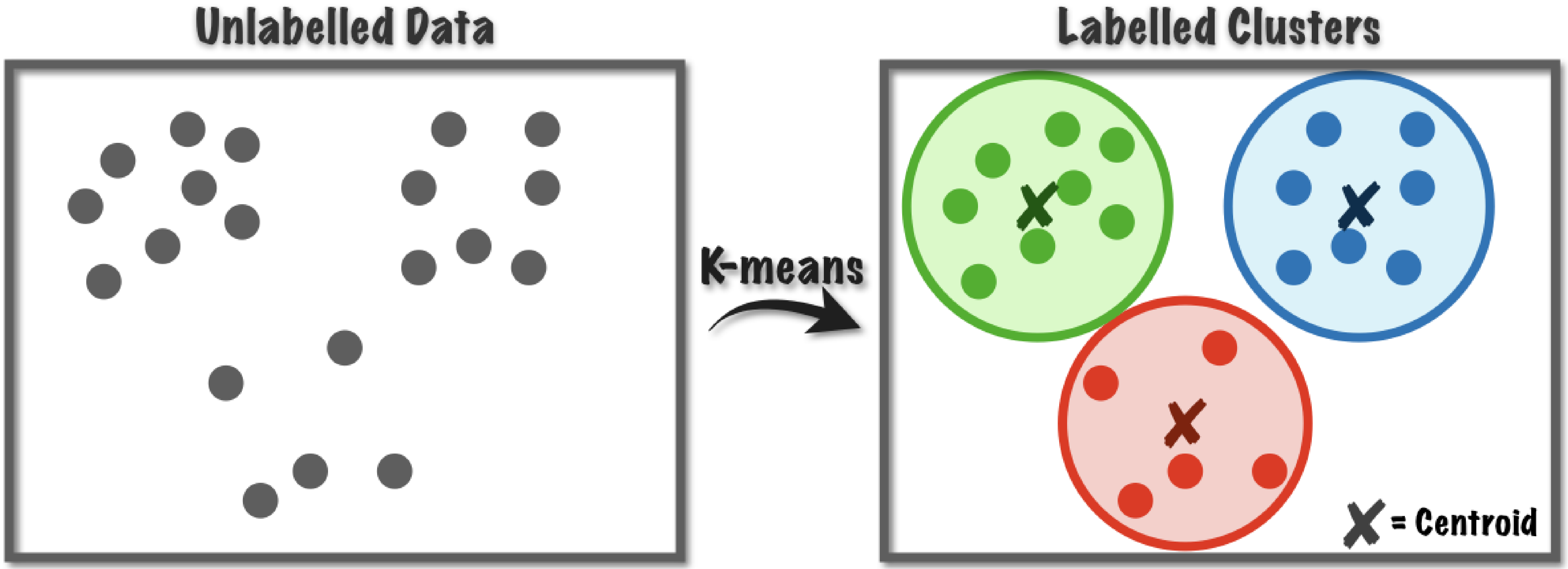
Types of machine learning

Supervised – machine learns on pairs of input and output data

Unsupervised – machine groups and interprets basing just on the input data

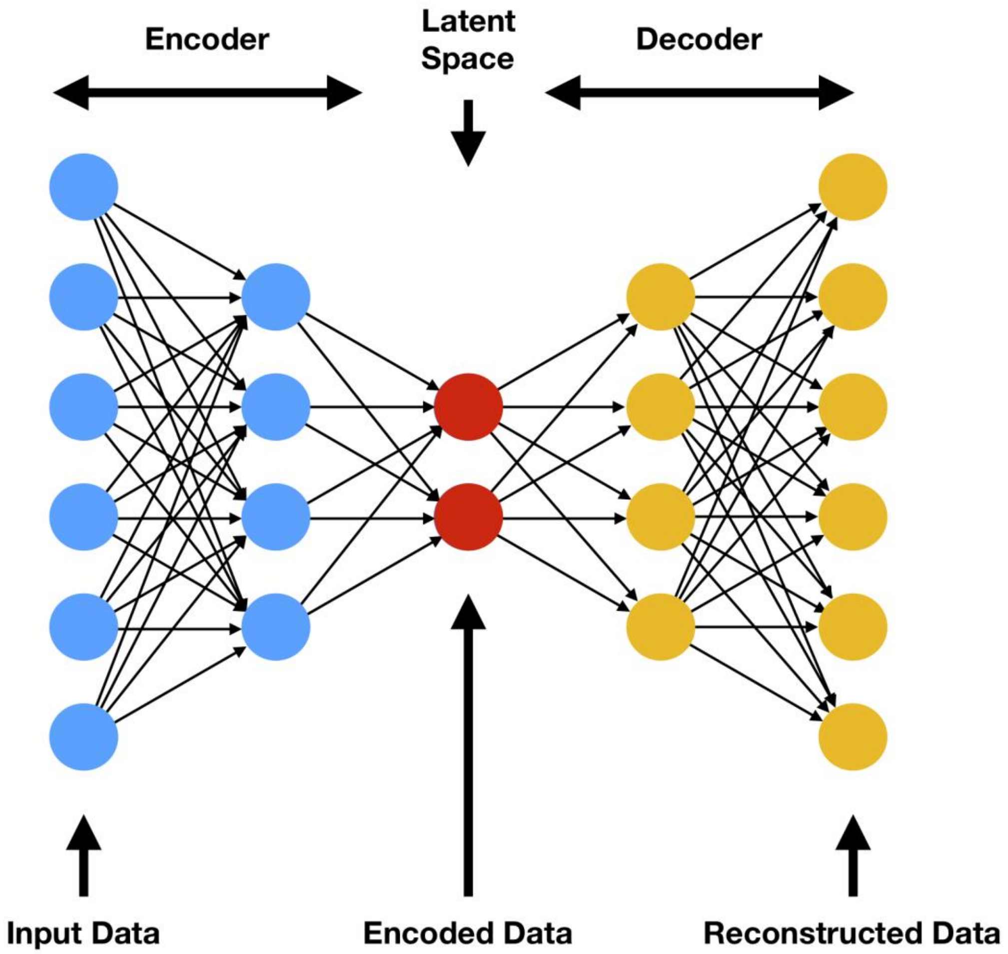
Reinforcement – algorithm learns to react to an environment

k-means clustering



https://towardsdatascience.com/k-means-a-complete-introduction-1702af9cd8c

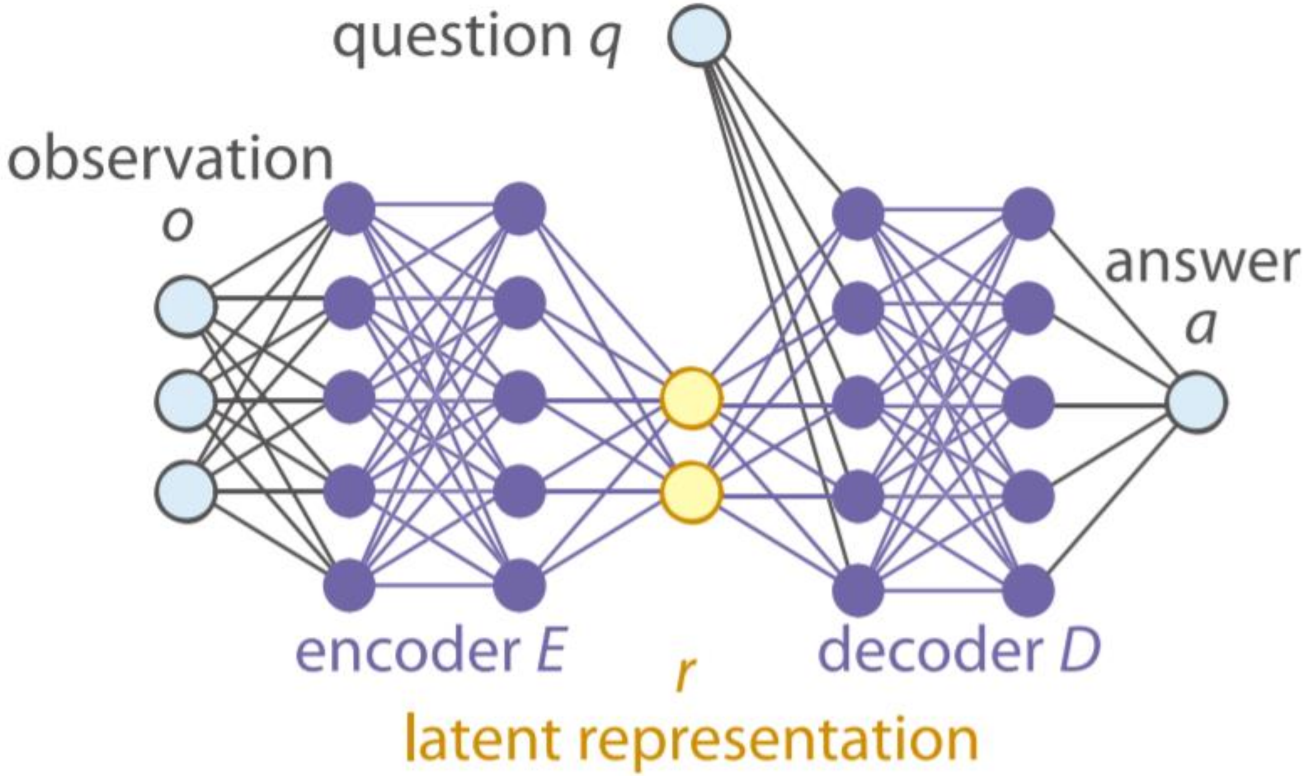
Autoencoder (AE)



„compression” algorithm

https://www.compthree.com/blog/autoencoder/

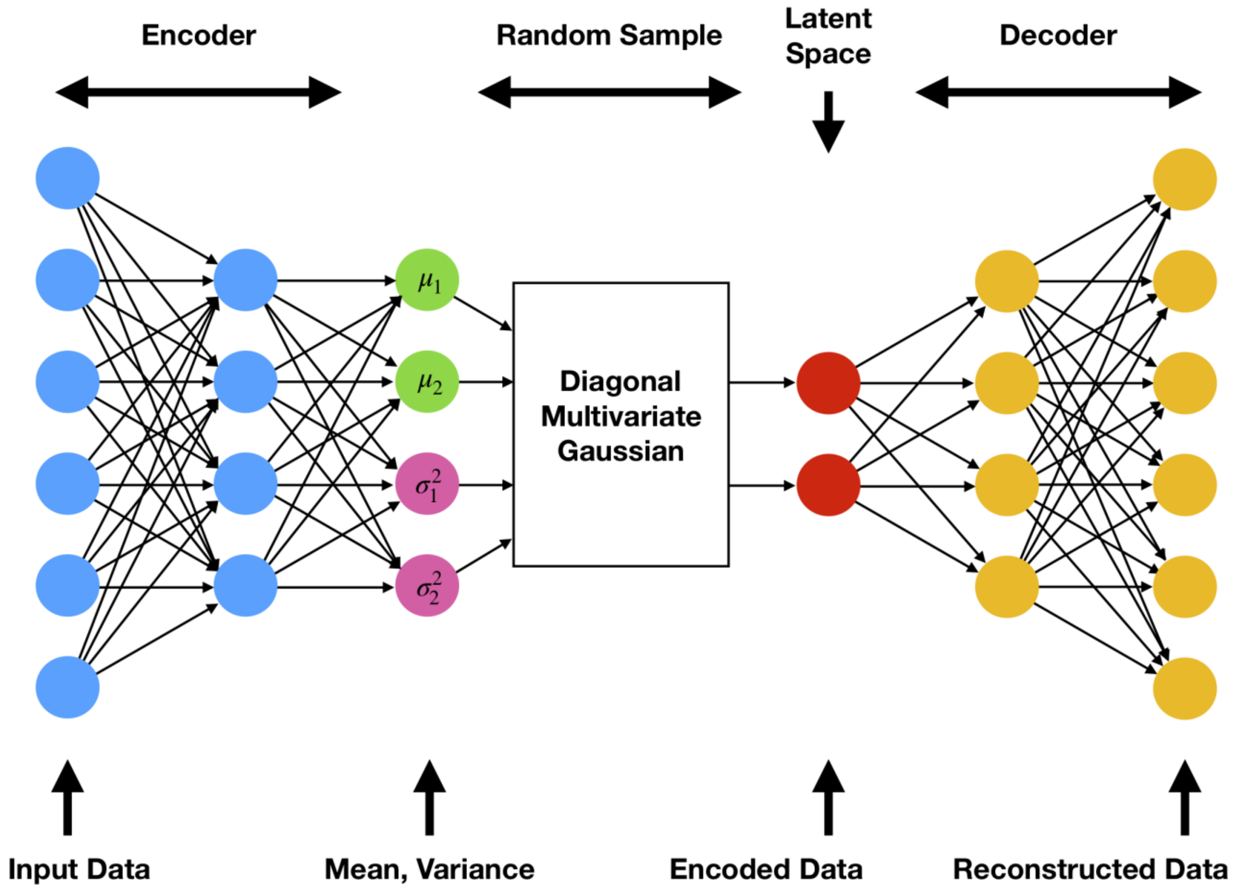
AE with a question neuron



we compress input data and can add something extra to the decoder

beautiful paper: Phys. Rev. Lett. **124**, 010508 (2020)

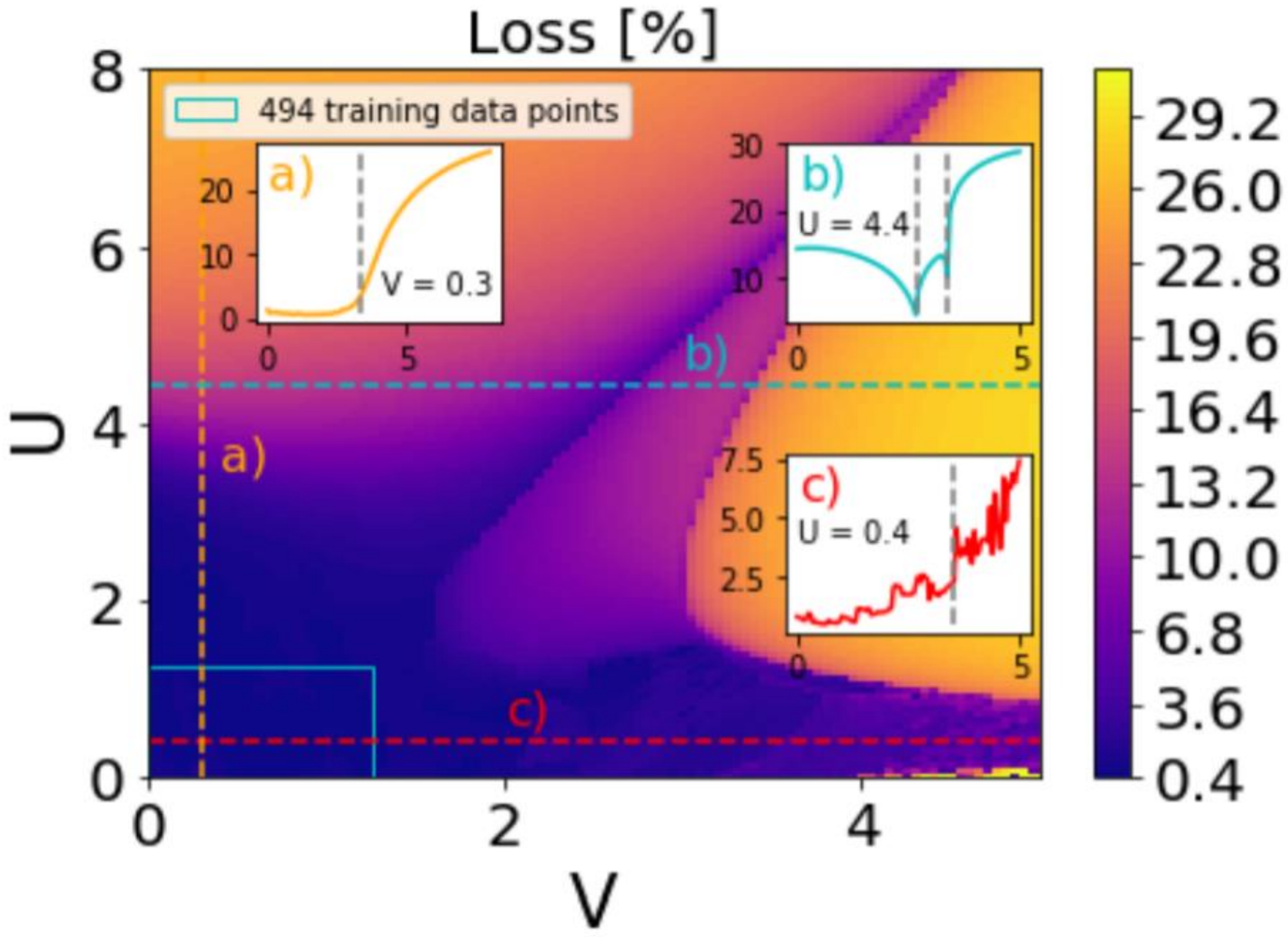
Variational autoencoder (VAE)



it was shown that encoding input data into a probability distribution rather than single features in latent space increases stability of data transformation

https://www.compthree.com/blog/autoencoder/

Anomaly detection with AE

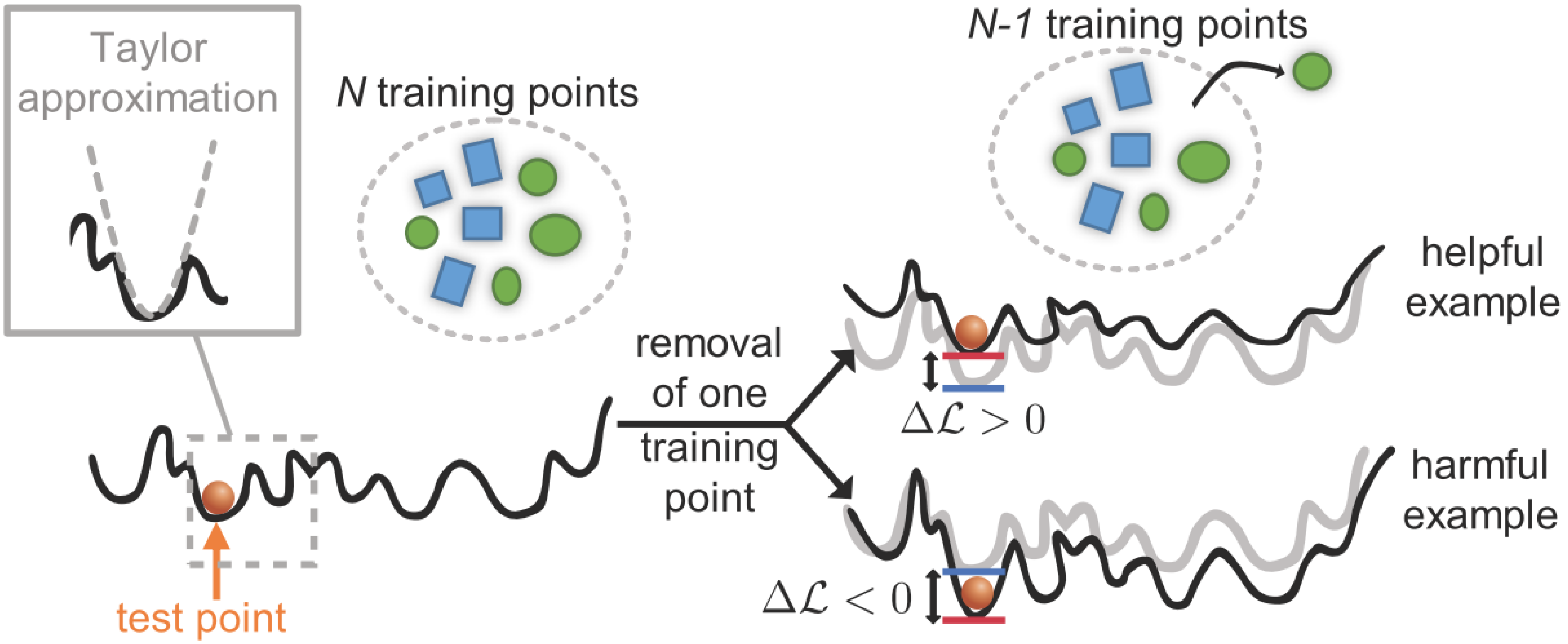


we train in one phase

-> we see input data are from other phase as AE fails badly

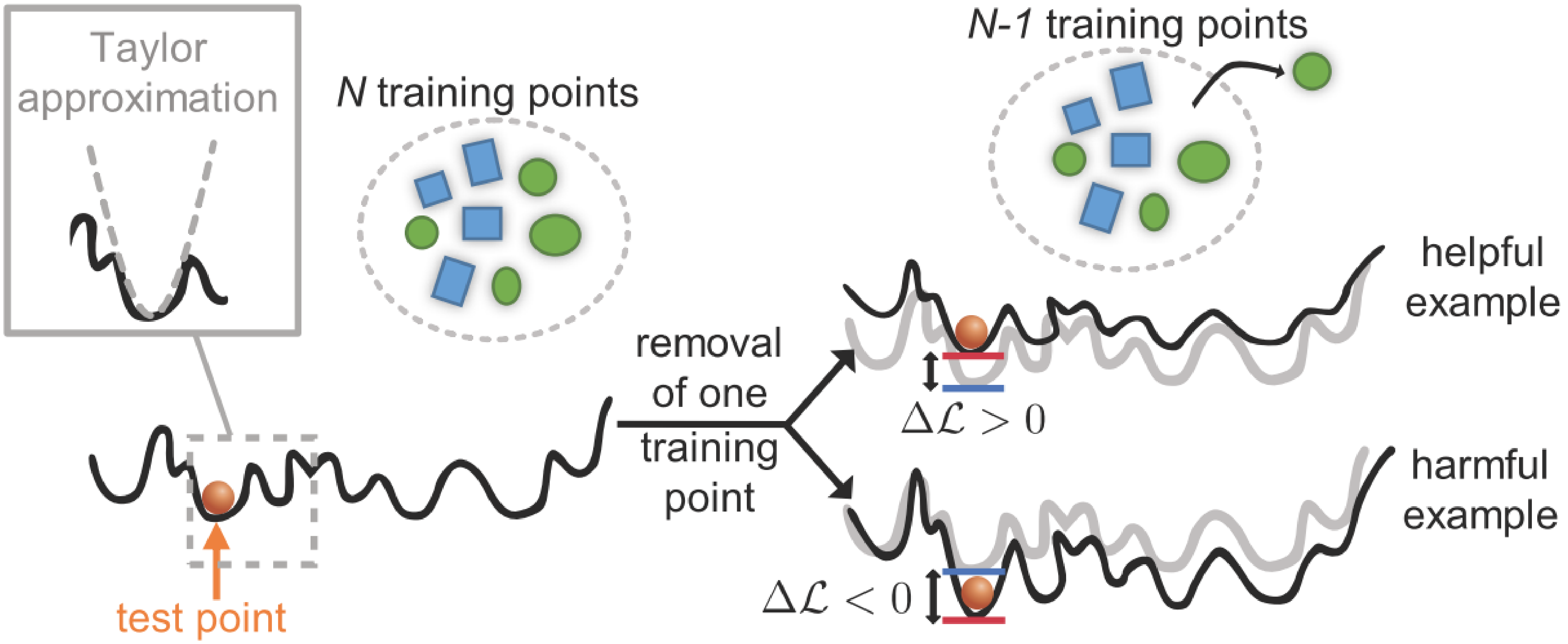
Phys. Rev. Lett. **125**, 170603 (2020)

Influence functions = approximation of leave-one-out training



New J. Phys. 22, 115001 (2020)

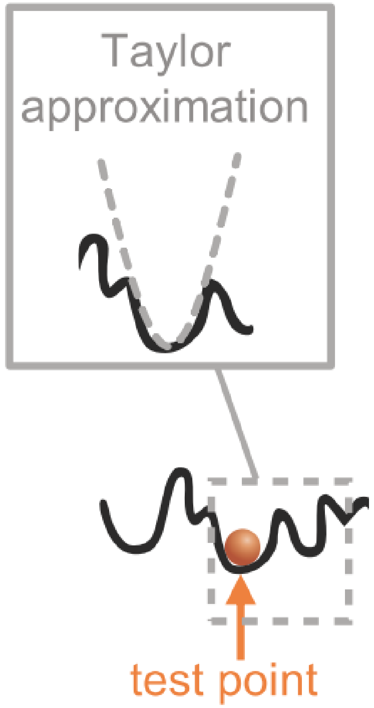
Influence functions = approximation of leave-one-out training



prohibitively expensive!

New J. Phys. 22, 115001 (2020)

Influence functions



Analytical approximation for leave-one-out training

$$\mathcal{I}(z_{\mathrm{r}}, z_{\mathrm{test}}) = \frac{1}{n}\nabla_\theta \mathcal{L}(z_{\mathrm{test}}, \hat{\theta})^T \underbrace{H_\theta^{-1}(\hat{\theta})\nabla_\theta \mathcal{L}(z_{\mathrm{r}}, \hat{\theta})}$$

approximated change in parameters due to removal of z_{r}

Assumption: Hessian is positive-definite.
Generalization to non-convex models was done by Koh & Liang: arXiv:1703.04730, ICML 2017's best paper

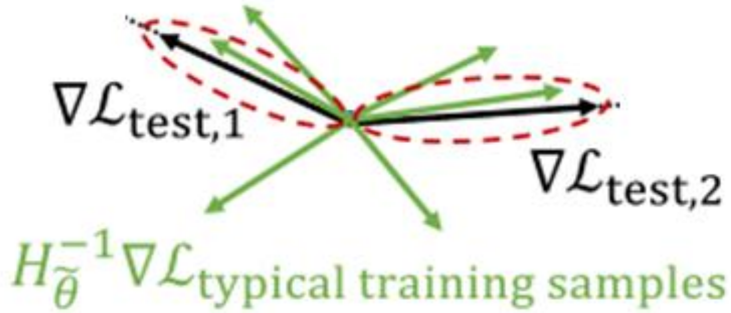
Geometrical interpretation of influence functions

$$\mathcal{I}(z_{\mathrm{r}}, z_{\mathrm{test}}) = \frac{1}{n}\nabla_\theta \mathcal{L}(z_{\mathrm{test}}, \hat{\theta})^T H_\theta^{-1}(\hat{\theta})\nabla_\theta \mathcal{L}(z_{\mathrm{r}}, \hat{\theta})$$

it is a scalar product of two gradients, corrected by local curvature described by the Hessian

Geometrical interpretation of influence functions

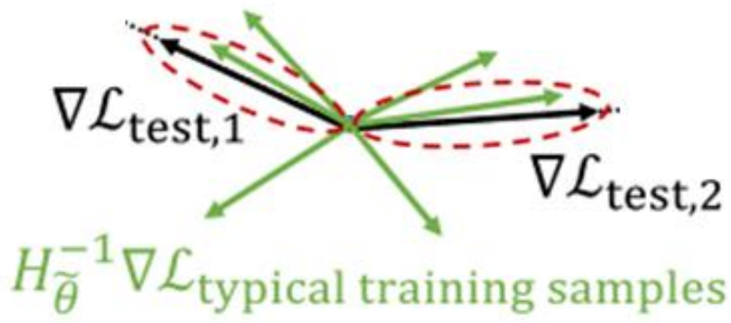
$$\mathcal{I}(z_{\mathrm{r}}, z_{\mathrm{test}}) = \frac{1}{n} \nabla_\theta \mathcal{L}(z_{\mathrm{test}}, \hat{\theta})^T H_\theta^{-1}(\hat{\theta}) \nabla_\theta \mathcal{L}(z_{\mathrm{r}}, \hat{\theta})$$



it is a scalar product of two gradients, corrected by local curvature described by the Hessian

Geometrical interpretation of influence functions

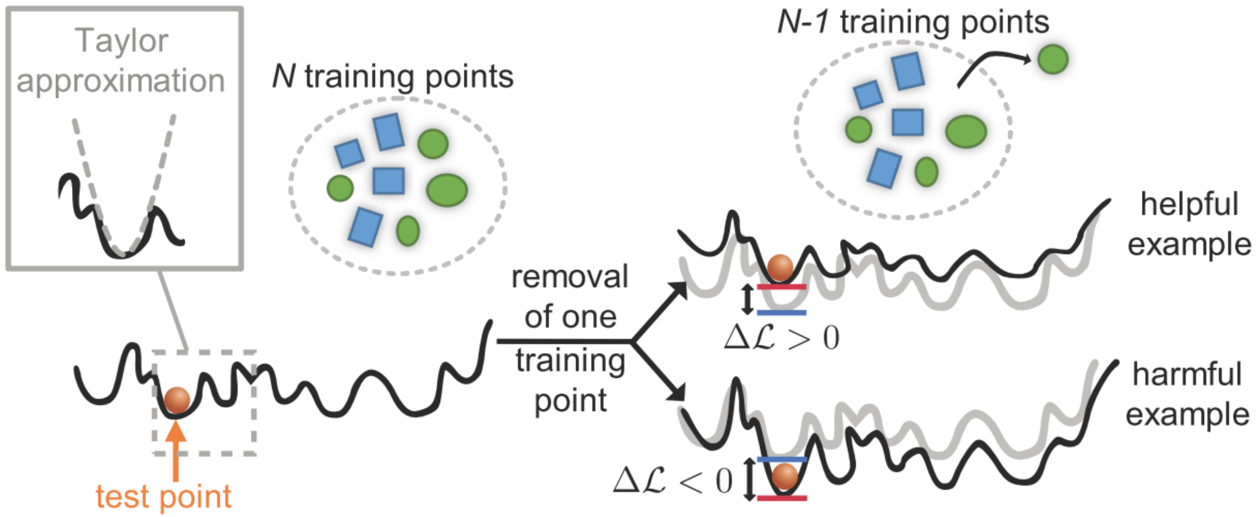
$$\mathcal{I}(z_{\mathrm{r}}, z_{\mathrm{test}}) = \frac{1}{n}\nabla_\theta \mathcal{L}(z_{\mathrm{test}}, \hat{\theta})^T H_\theta^{-1}(\hat{\theta})\nabla_\theta \mathcal{L}(z_{\mathrm{r}}, \hat{\theta})$$



it is a scalar product of two gradients, corrected by local curvature described by the Hessian

notion of similarity in the model internal representation!

Influence functions = approximation of leave-one-out training



which data characteristics are influential?

similarity analysis

training points which are similarly influential (= have similar influence functions' values) are similar from the model's point of view

New J. Phys. 22, 115001 (2020)

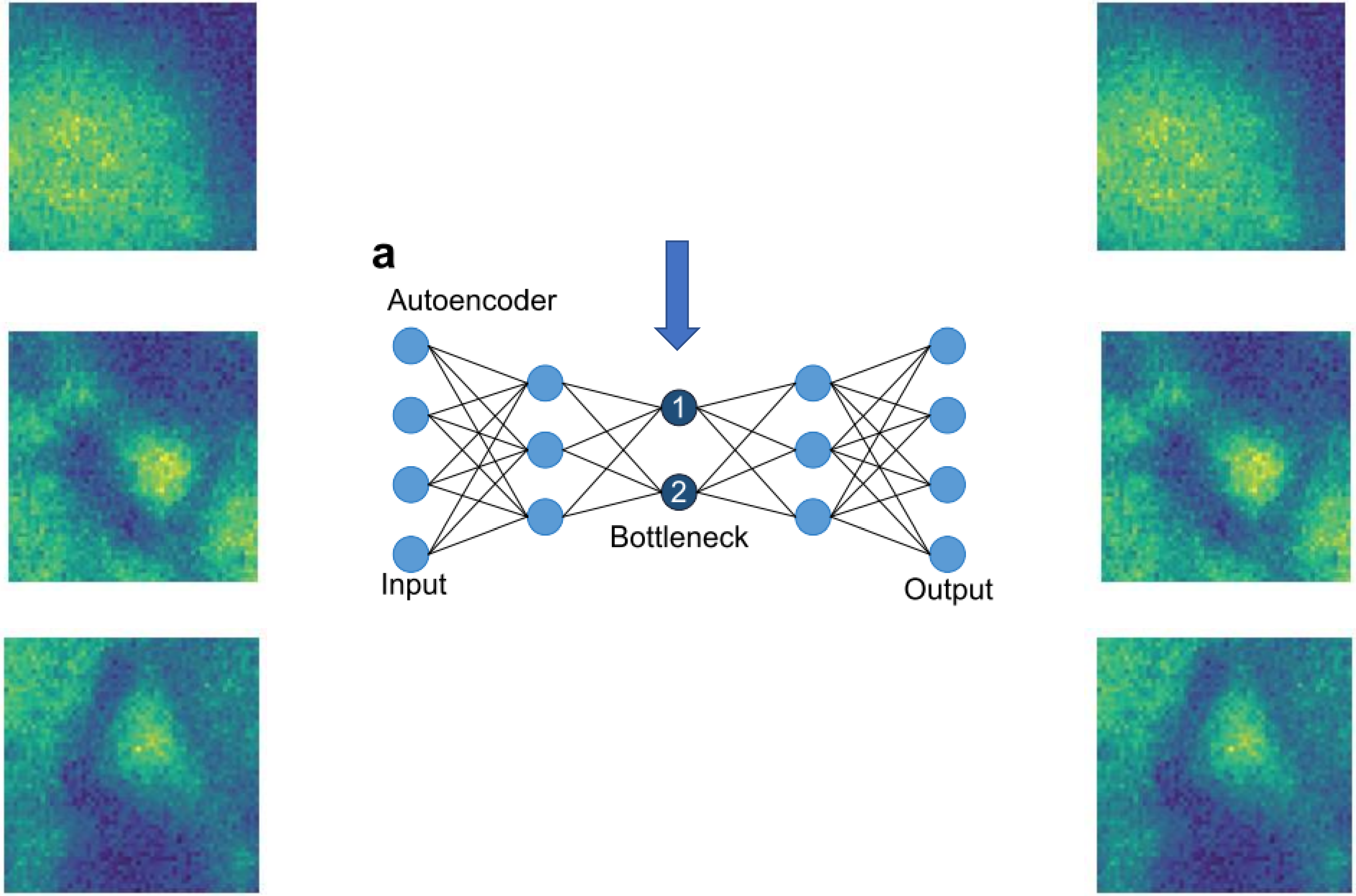
Outline

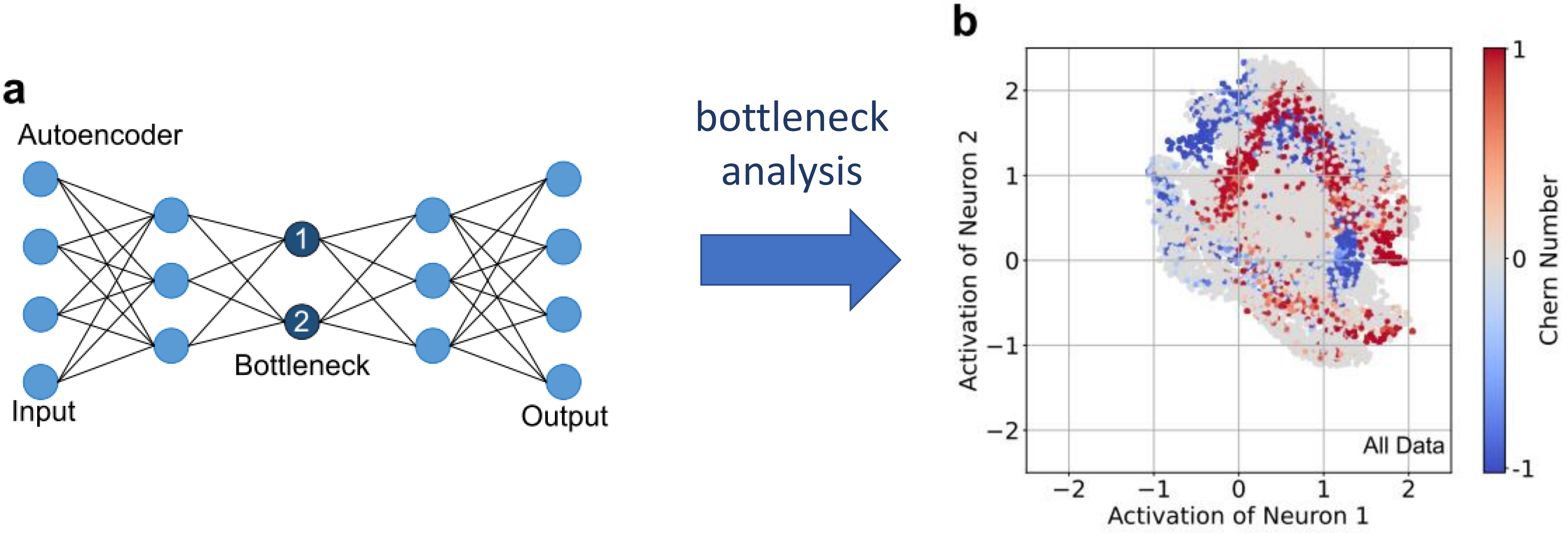
methods

- Ultracold system and experimental data
- ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions

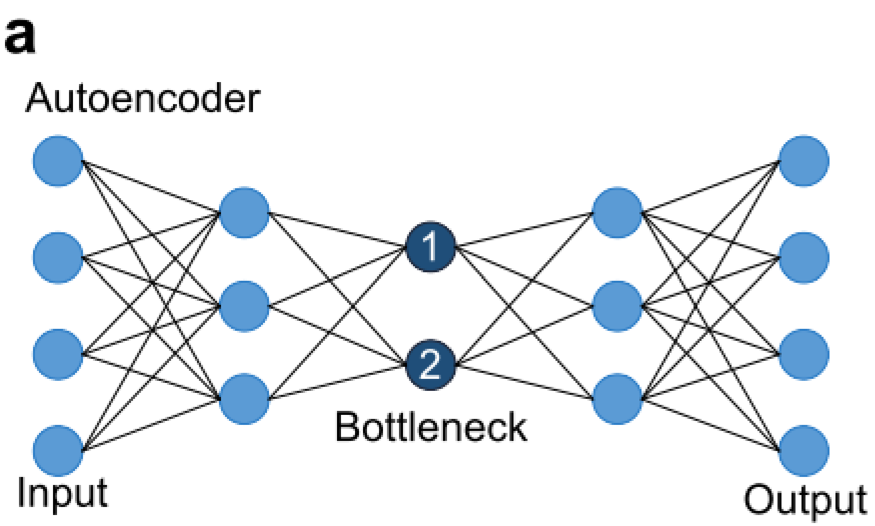
results

- **Unsupervised methods for raw data**
- Micromotion phase removal (supervised ML)
- Unsupervised methods for postprocessed data
- Similarity analysis with influence functions (supervised ML)

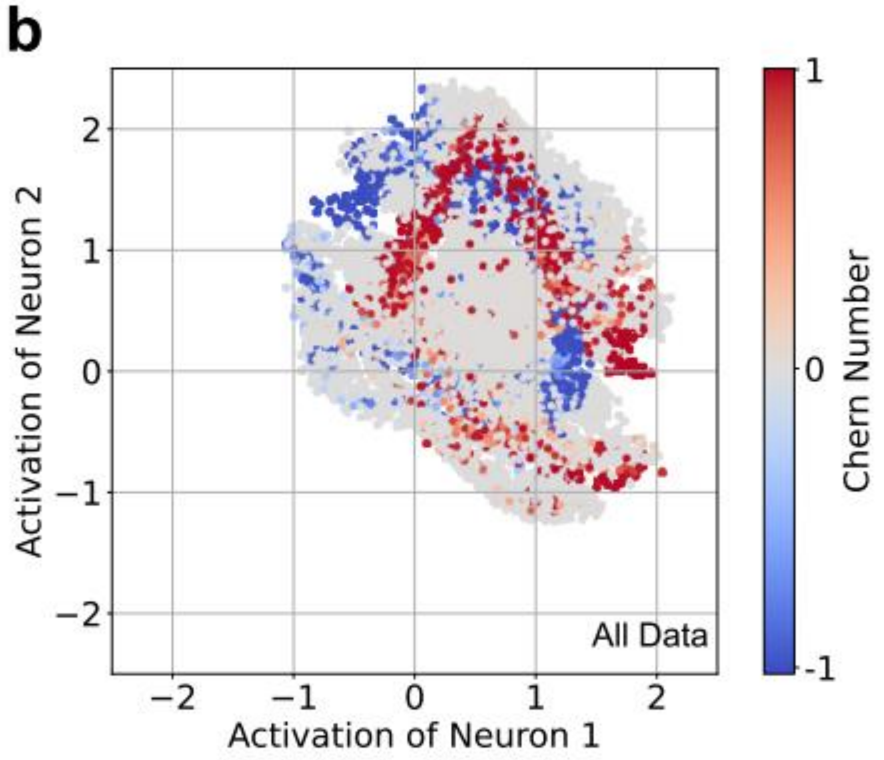

a
Autoencoder
1
2
Bottleneck
Input
Output

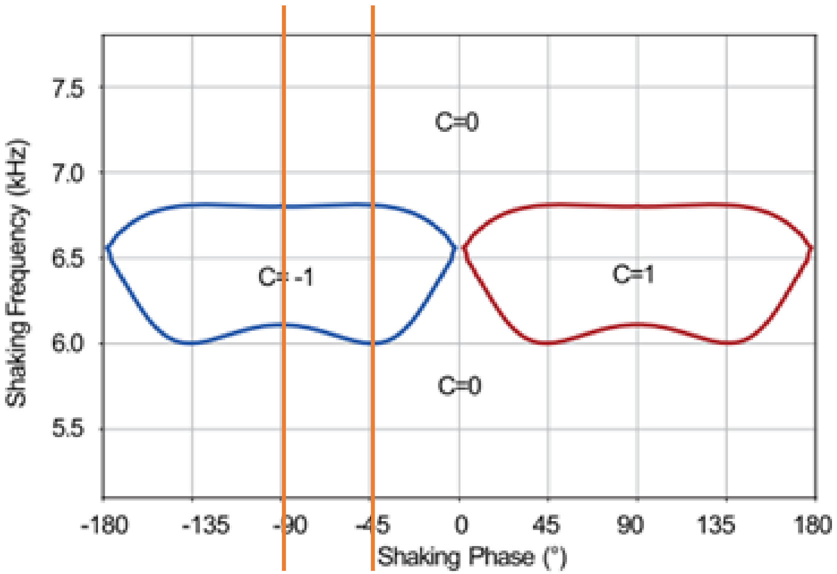


a
Autoencoder
Input
Bottleneck
Output
1
2
bottleneck analysis
b
Activation of Neuron 1
Activation of Neuron 2
All Data
Chern Number
1
0
-1

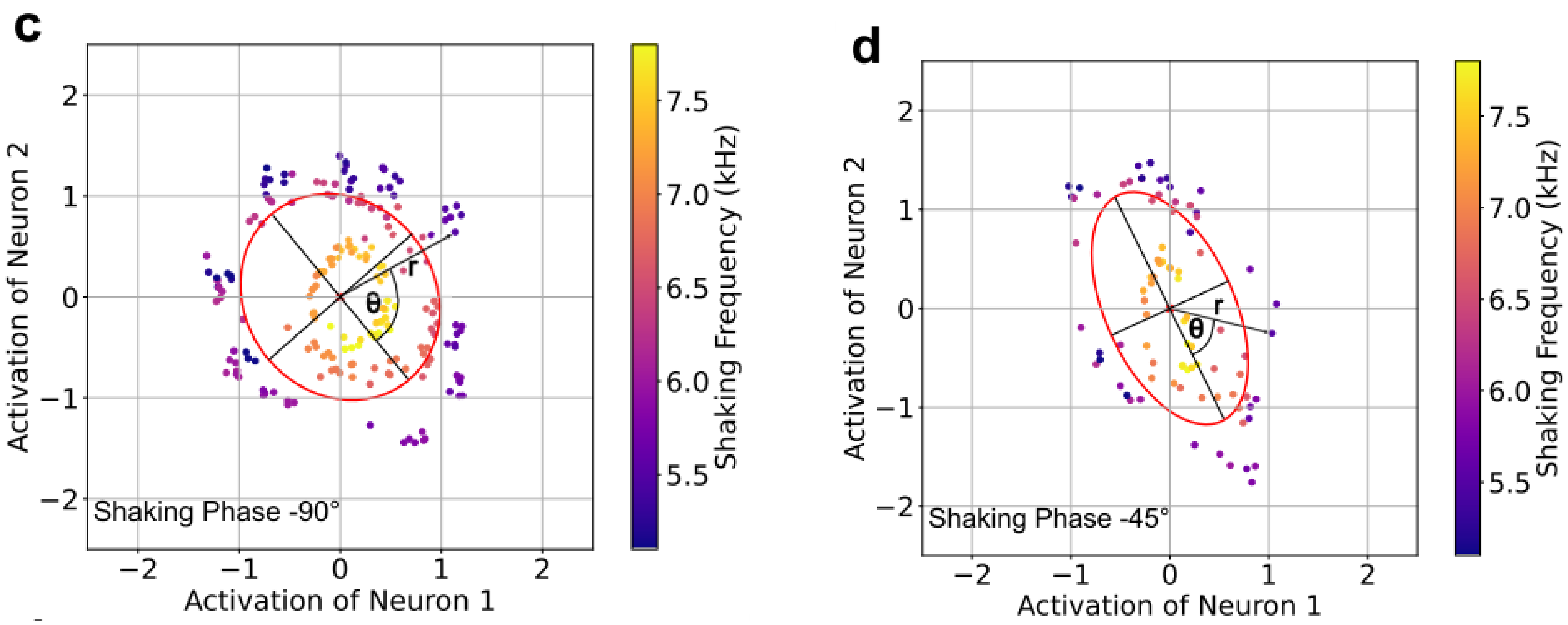


bottleneck
analysis

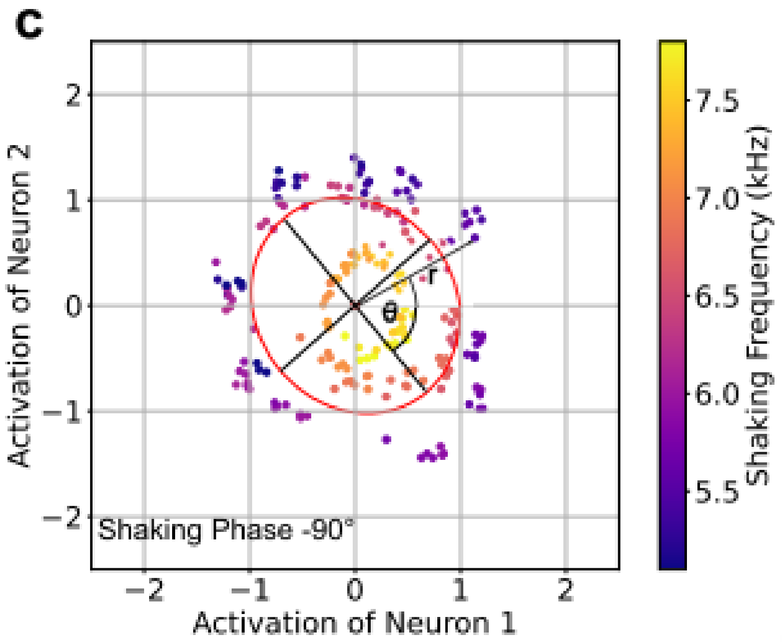


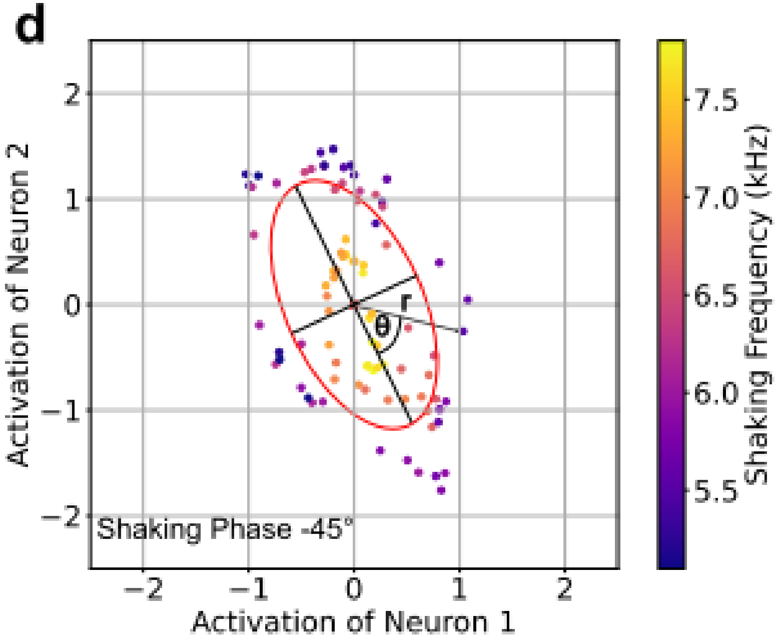


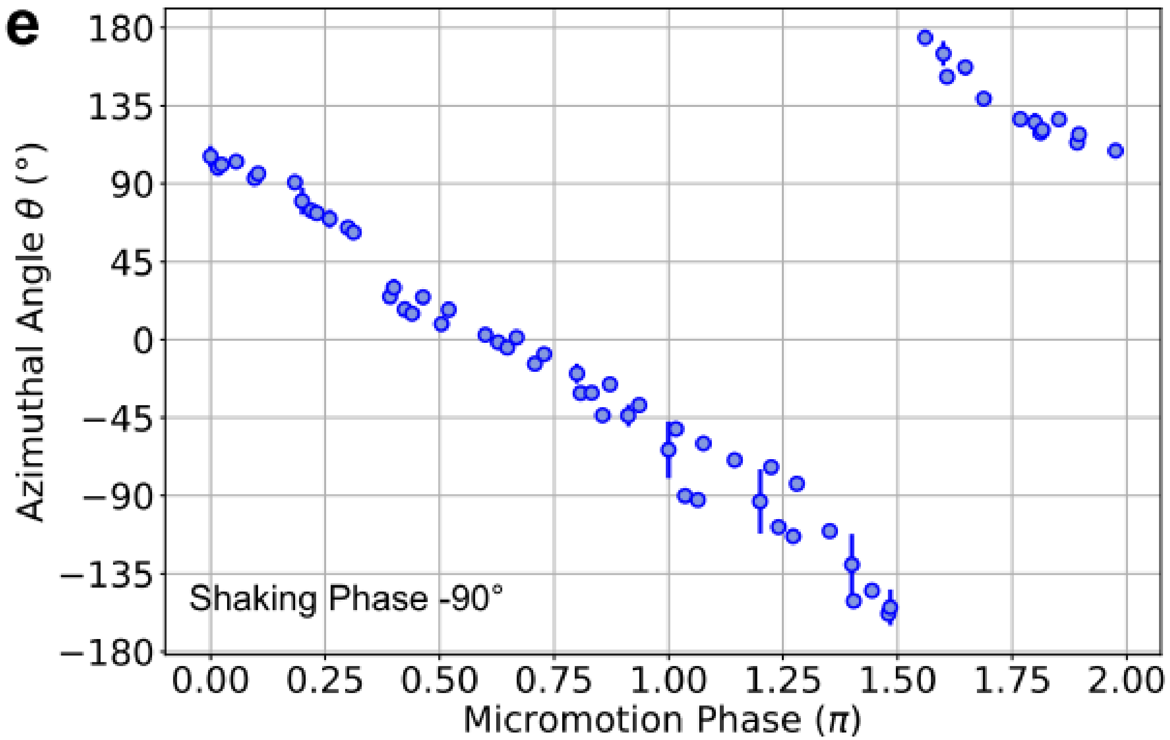
Let's take only single cuts
through the phase diagram!



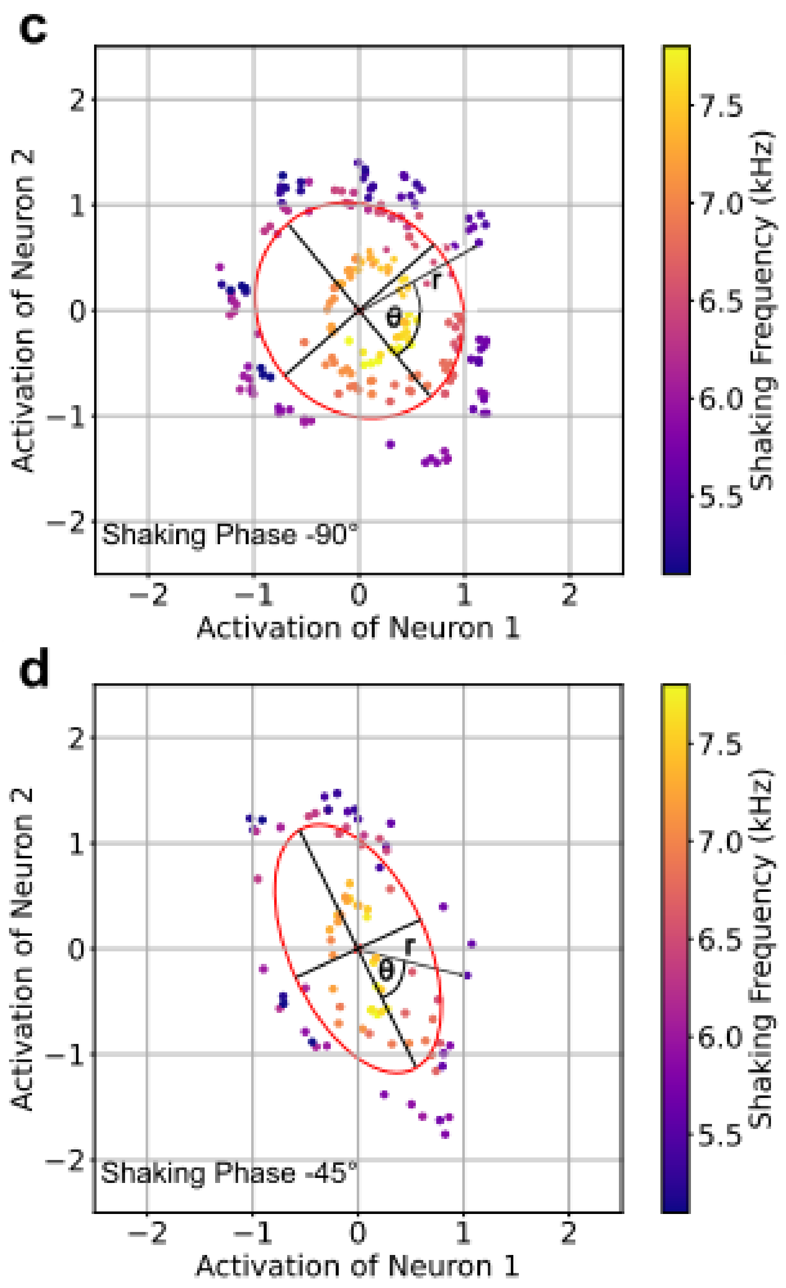
Data forms rings in latent space and can be fitted by an ellipse

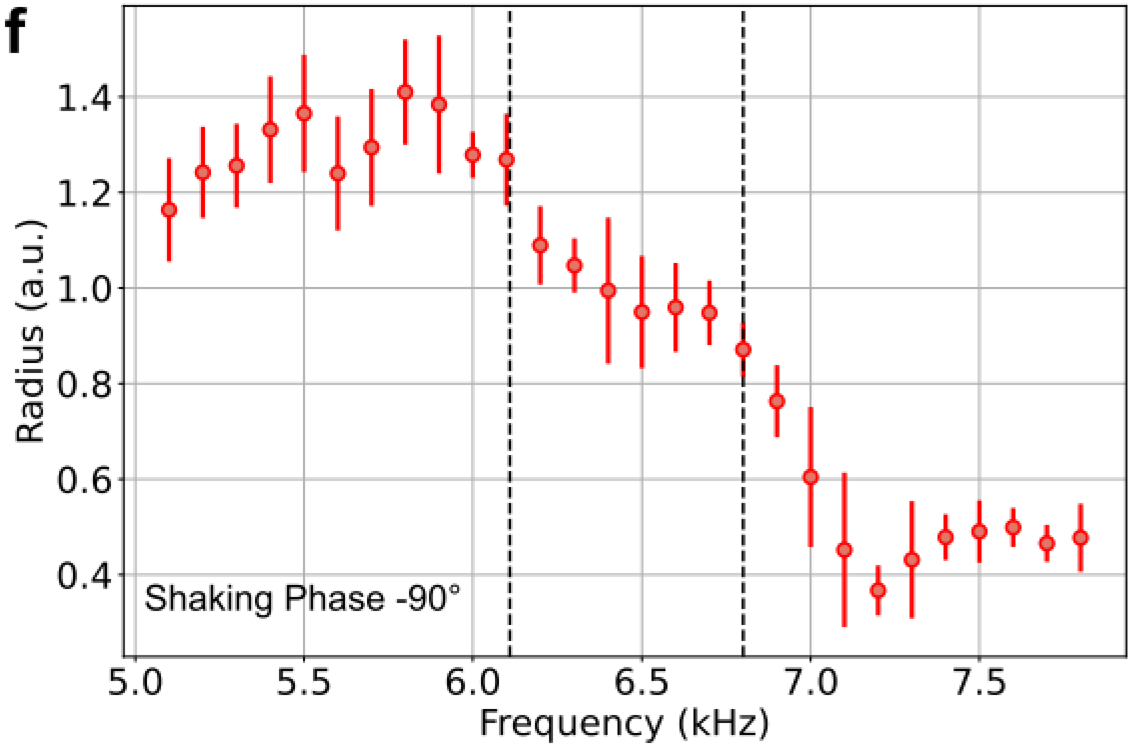






autoencoder learns micromotion phase!





radius doesn’t separate phases well…

Outline

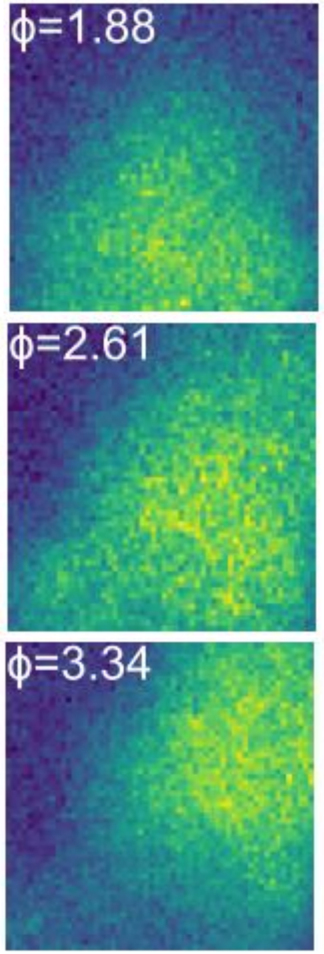
methods

- Ultracold system and experimental data
- ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions

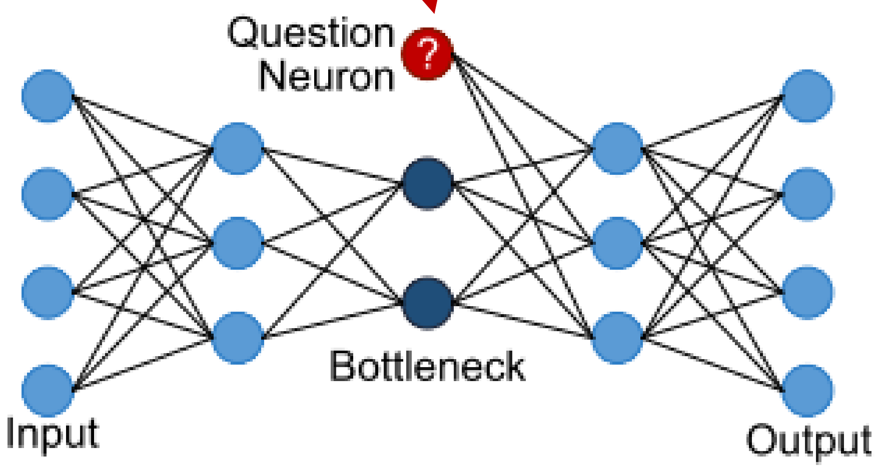
results

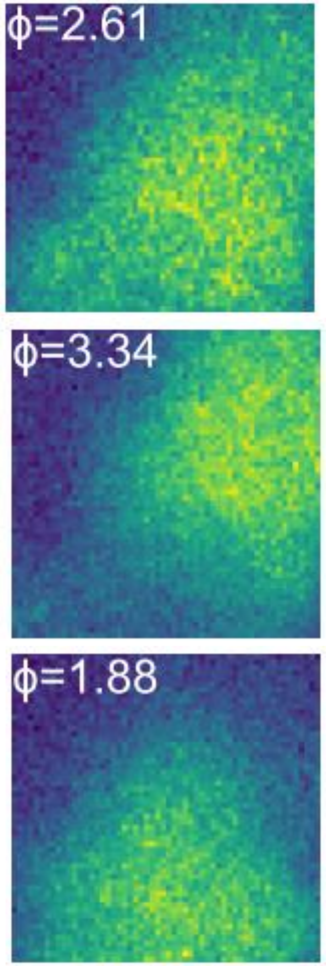
- Unsupervised methods for raw data
- Micromotion phase removal (supervised ML)
- Unsupervised methods for postprocessed data
- Similarity analysis with influence functions (supervised ML)

How to remove/fix the micromotion phase?



$$\Delta\phi = \phi_{\text{output}} - \phi_{\text{input}}$$





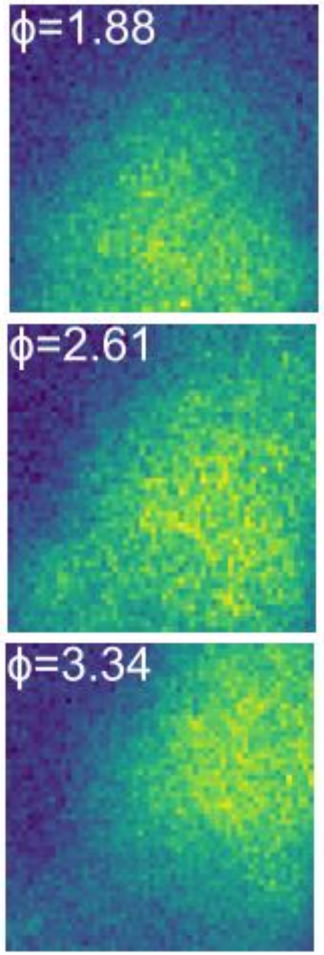
TRAINING

f_{sh} = 5.8 kHz

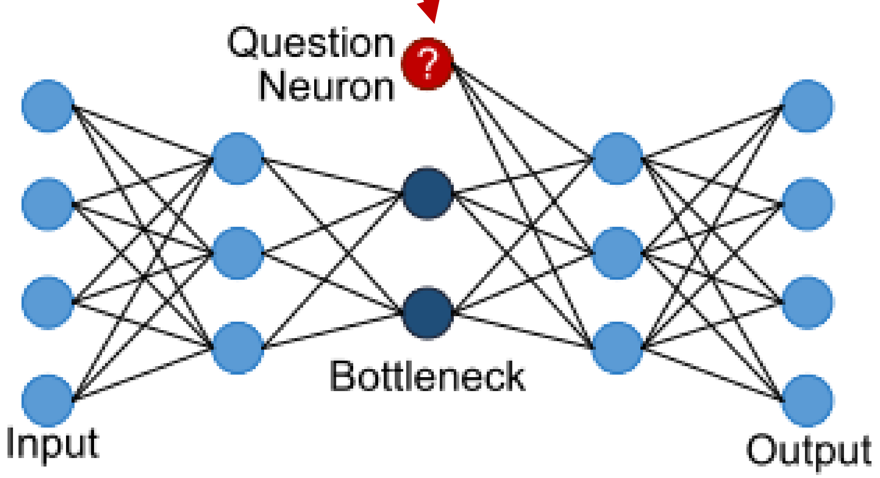
$\varphi = 90^\circ$

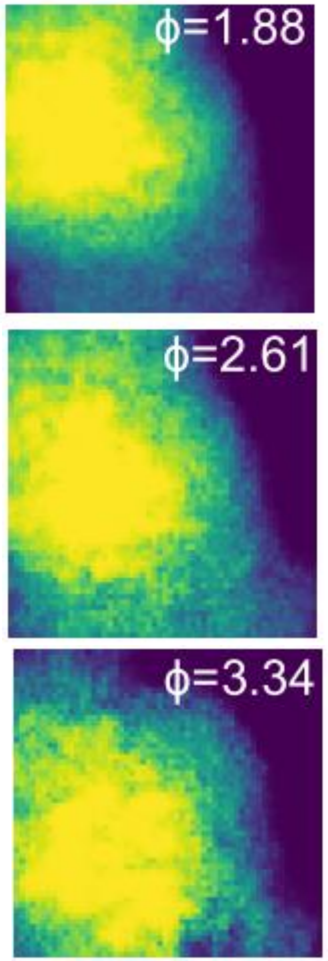
Autoencoder with a question neuron!

How to remove/fix the micromotion phase?



$$\Delta\phi = -\phi_{\mathrm{input}}$$





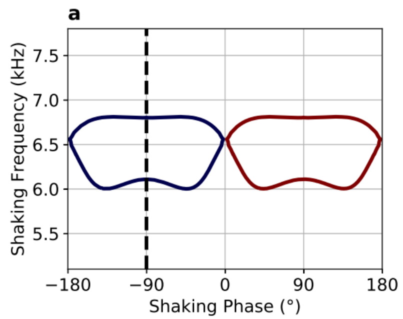
f_{sh} = 5.8 kHz

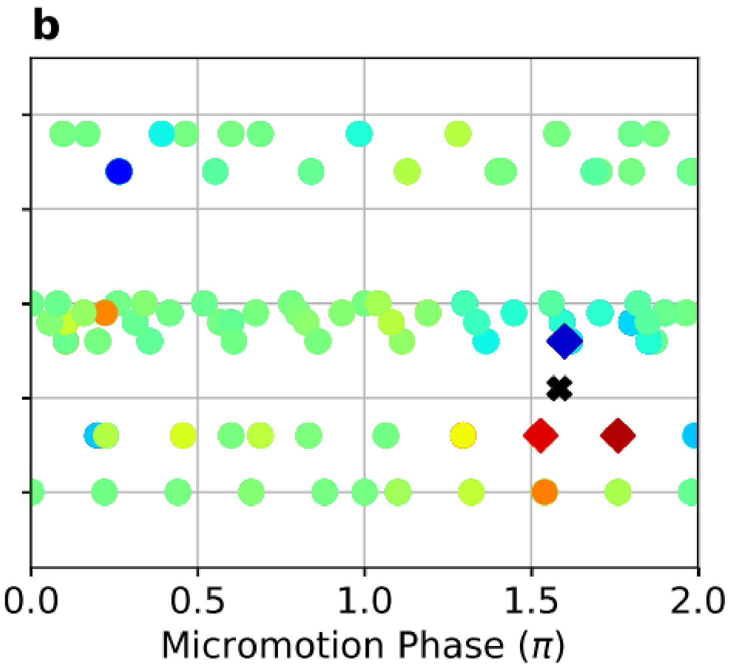
$\varphi = 90^\circ$

FIXING THE
MICROMOTION PHASE

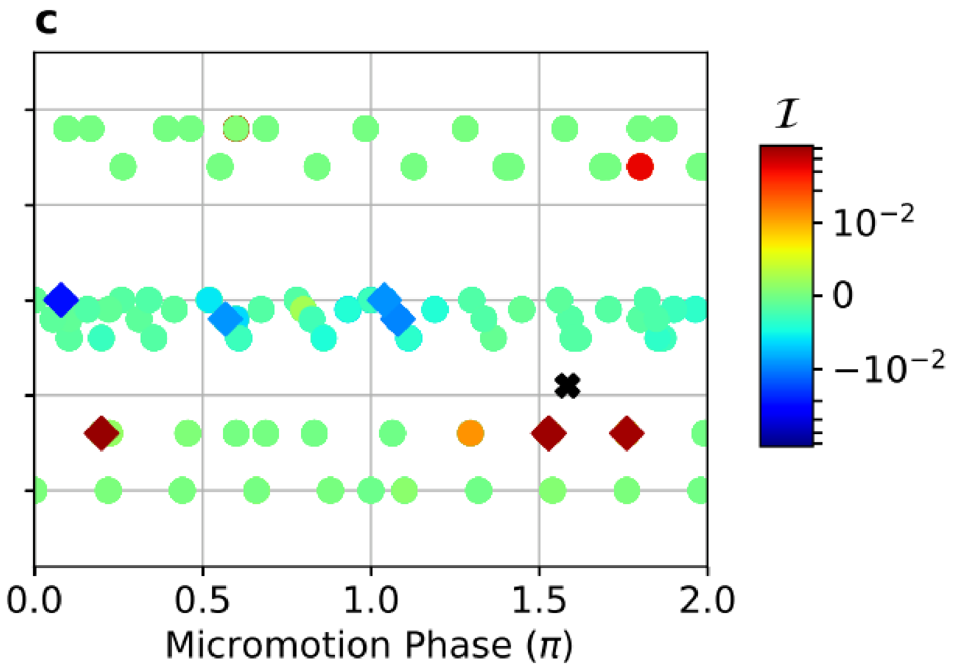
Autoencoder with a question neuron!

Did it work? Let's check with influence functions!





fixing the micromotion phase



The most influential points are localized around the same micromotion phase as test point

The most influential points are smeared out across different micromotion phases

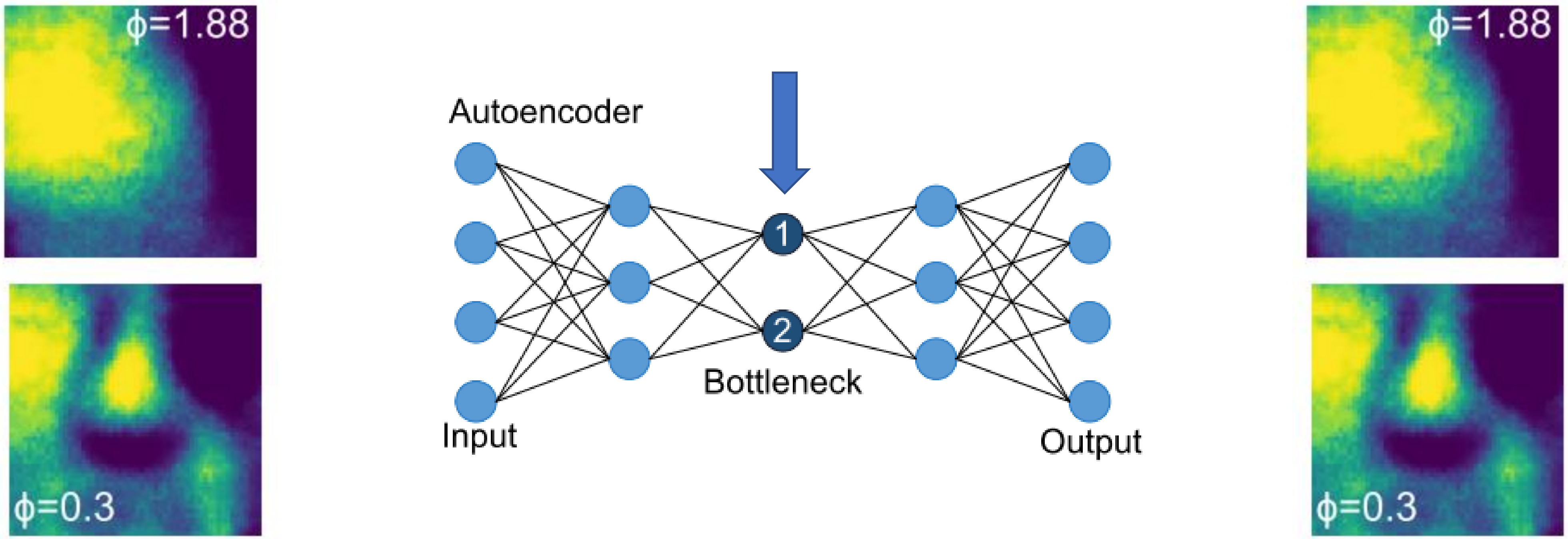
Outline

methods

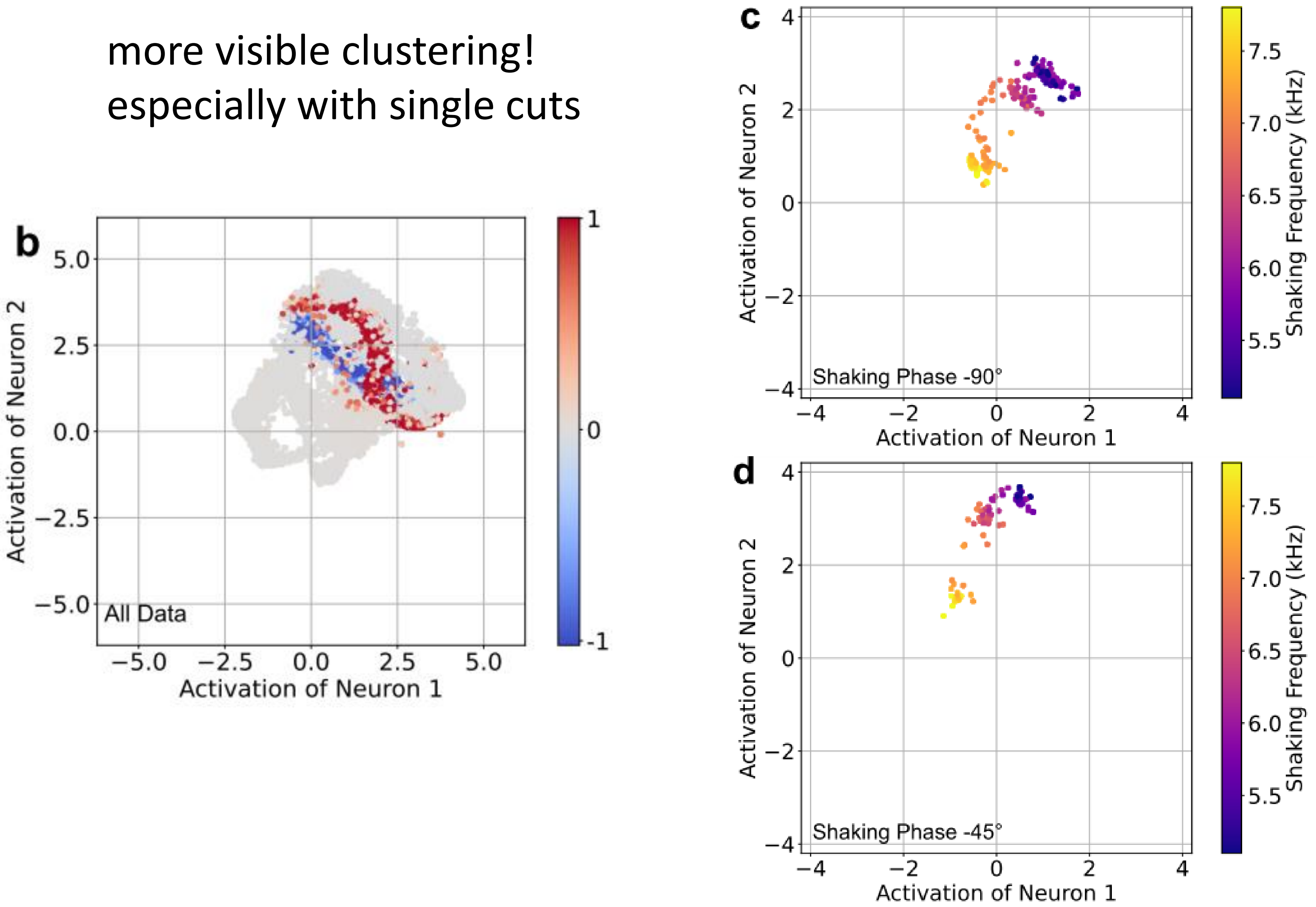
- Ultracold system and experimental data
- ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions

results

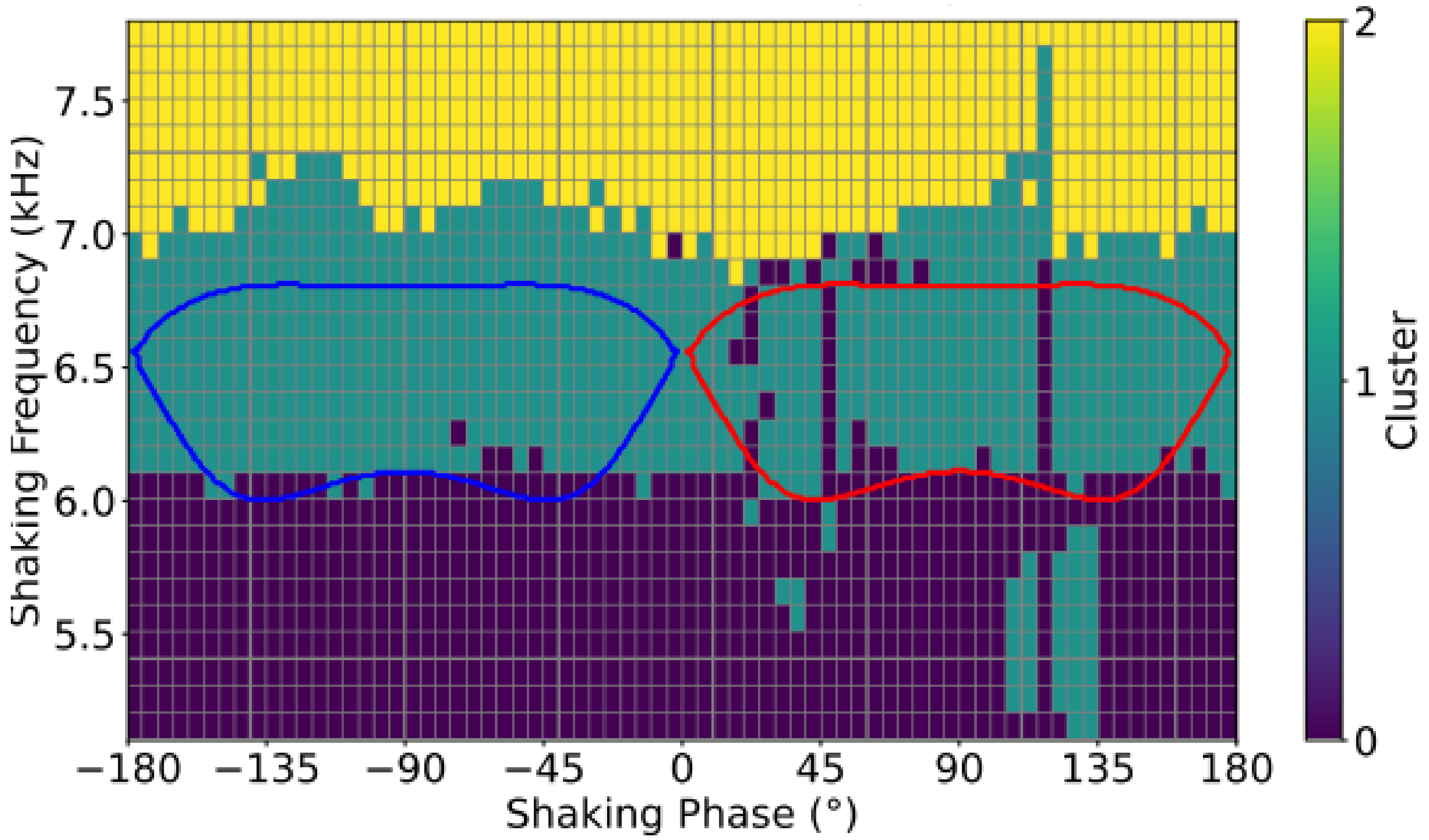
- Unsupervised methods for raw data
- Micromotion phase removal (supervised ML)
- **Unsupervised methods for postprocessed data**
- Similarity analysis with influence functions (supervised ML)


ϕ=1.88
ϕ=0.3
Autoencoder
1
2
Bottleneck
Input
Output
ϕ=1.88
ϕ=0.3

more visible clustering!
especially with single cuts



We finally arrived to *a* phase diagram

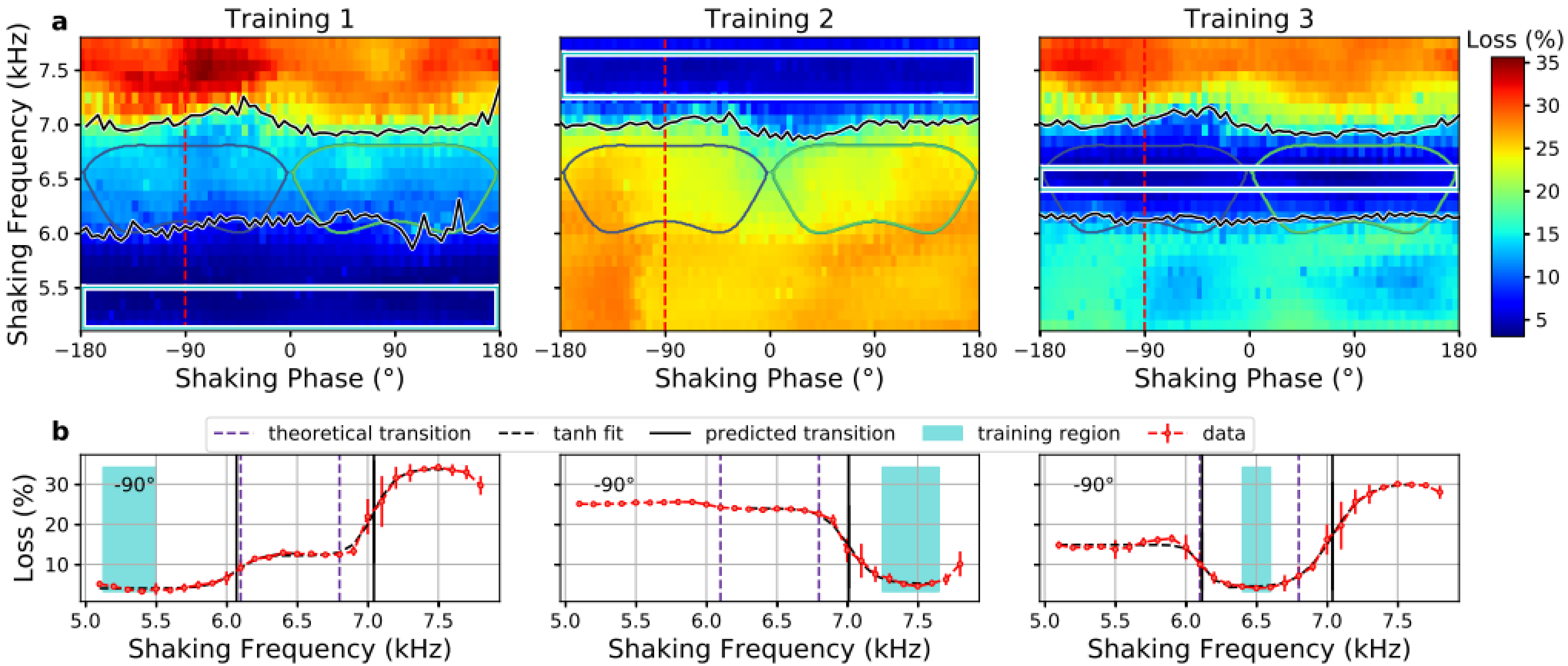


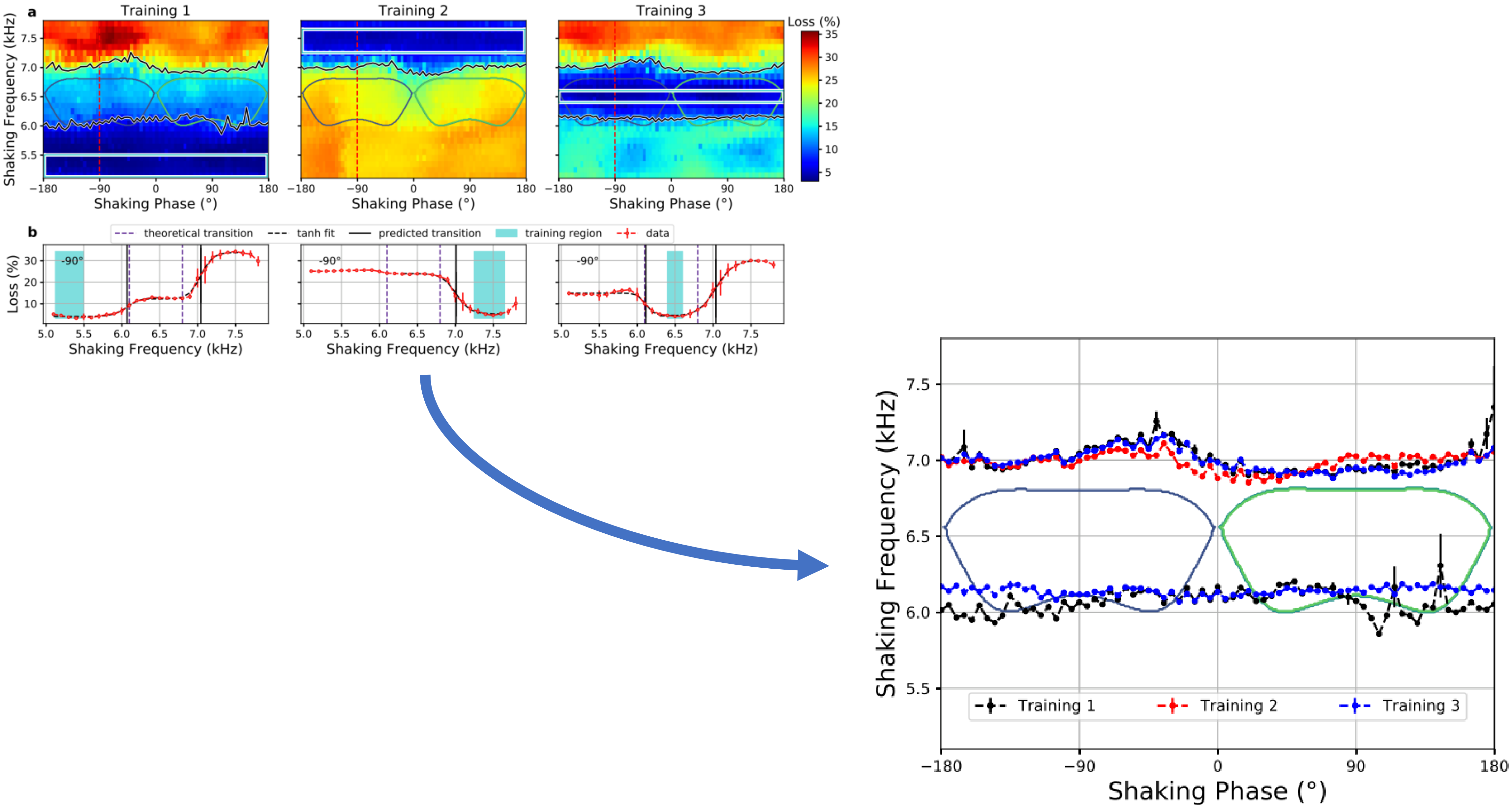
Problems:

- positive shaking phase is ugly
- no discrimination between two topological phases

Anomaly detection does better, but problem no. 2 stays.

No discrimination between topological phases.





a
Training 1
Training 2
Training 3
Loss (%)
Shaking Frequency (kHz)
Shaking Phase (°)
b
theoretical transition
tanh fit
predicted transition
training region
data
-90°
Loss (%)
Shaking Frequency (kHz)
Training 1
Training 2
Training 3
Shaking Frequency (kHz)
Shaking Phase (°)

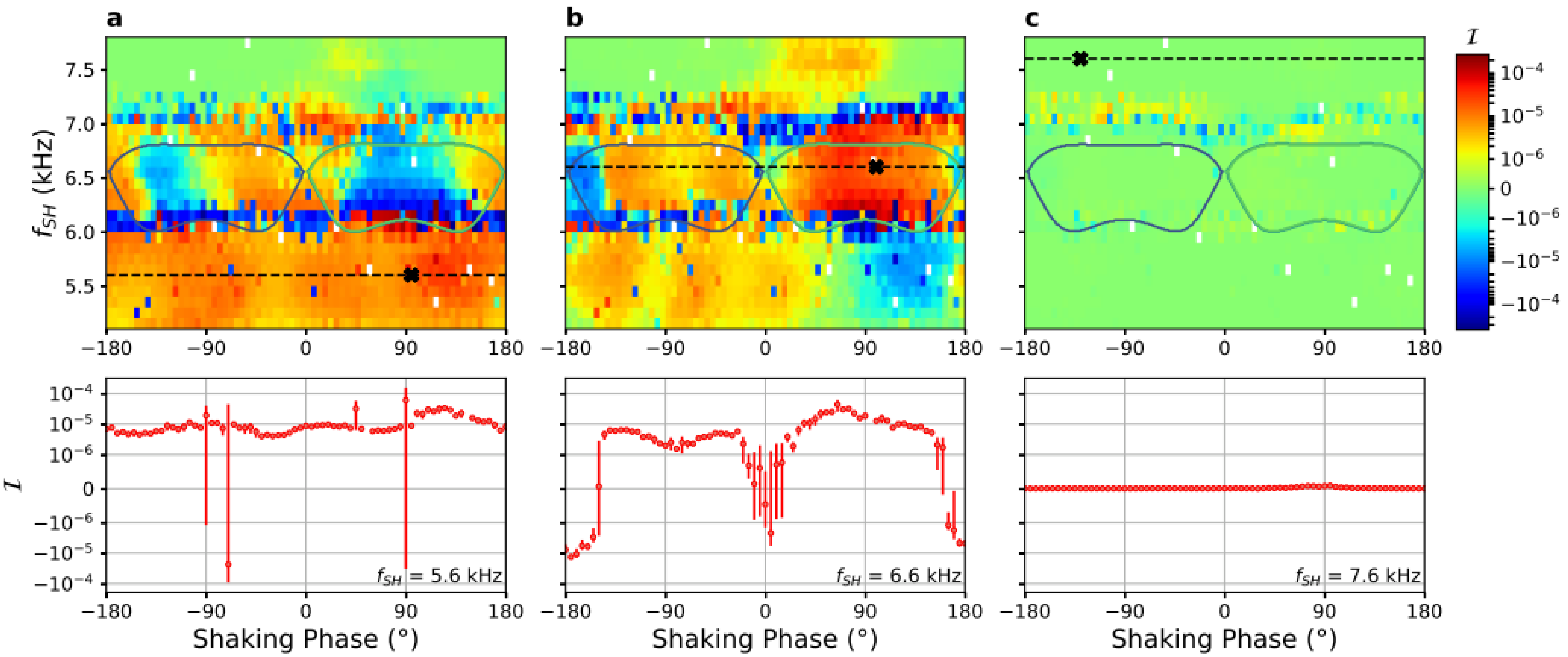
Outline

methods

results

- Ultracold system and experimental data
- ML methods: (variational) autoencoders, k-means clustering, anomaly detection, influence functions
- Unsupervised methods for raw data
- Micromotion phase removal (supervised ML)
- Unsupervised methods for postprocessed data
- Similarity analysis with influence functions (supervised ML)

1. Supervised learning with labels provided by anomaly detection scheme
2. Similarity analysis of the ML model with **influence functions**…



3. **We finally discriminate between topological phases!**

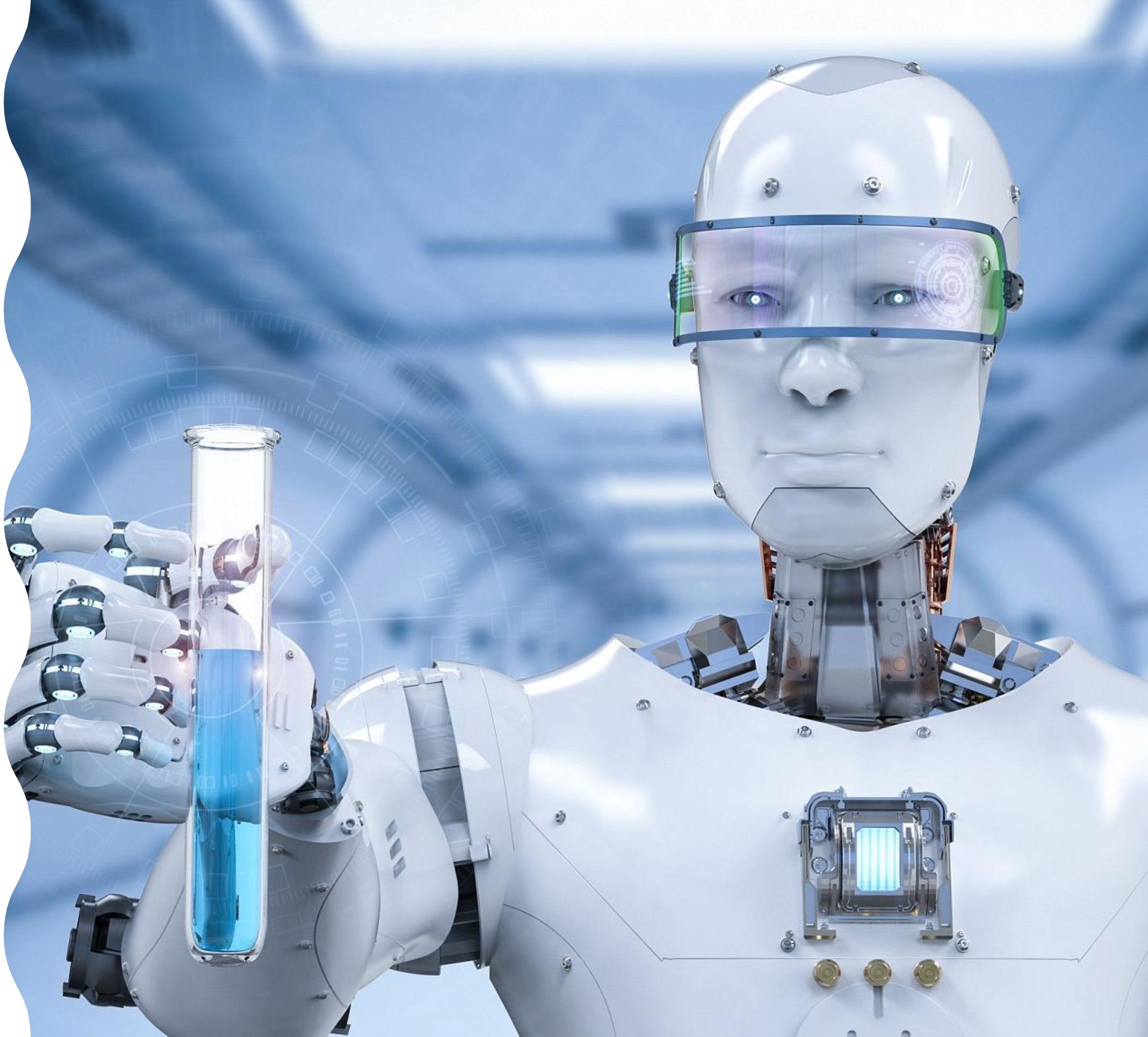
Summary and conclusions

- We applied autoencoders and influence functions to experimental data on topological Haldane model
- We *struggled* and fixed the micromotion phase
- We recovered the full topological phase diagram without any (?) *a priori* knowledge of the underlying physics in a fully (?) unsupervised way

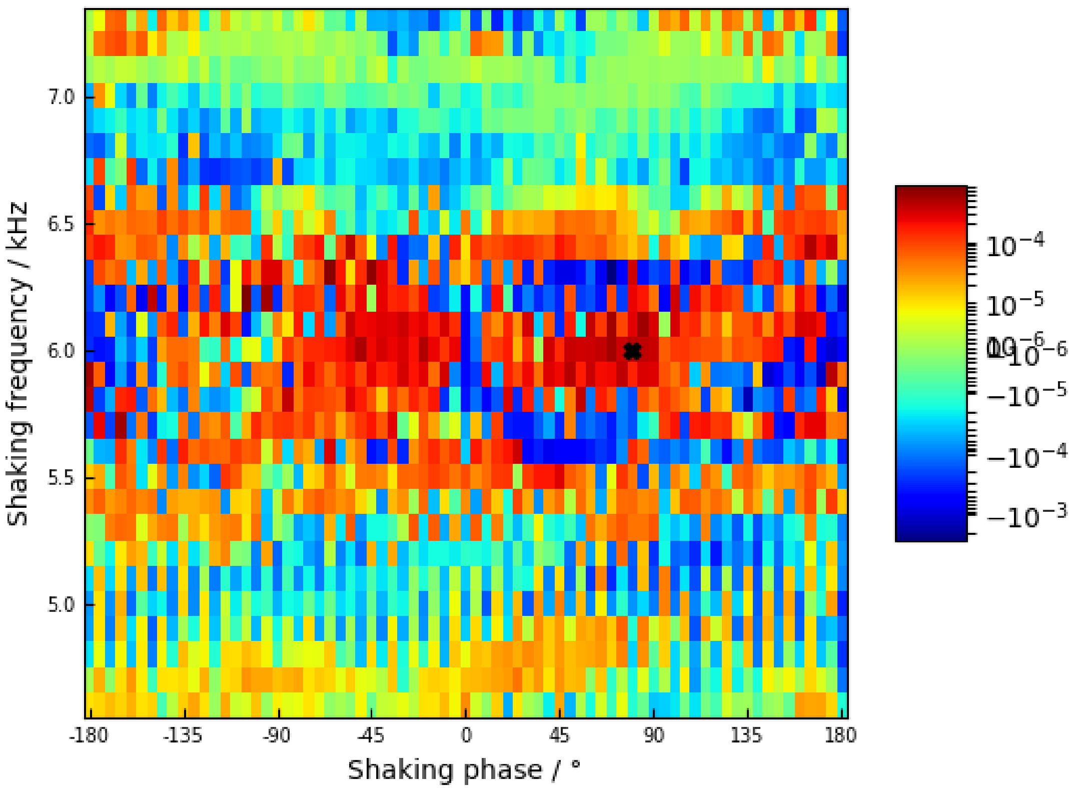
Thank you for your attention!

Mach. Learn.: Sci. Technol.
2 035037 (2021)

https://github.com/nkaeming/unsupervised-machine-learning-of-topological-phase-transitions-from-experimental-data



Final plot for influence functions without fixed micromotion phase



Anomaly detection and small boxes don’t work

It doesn’t generalize well beside the training region