Investigating deep convolutional autoencoders to mitigate systematic differences between data and simulations

Stefan Geißelsöder
2018 February 20.
Big Data Science in Astroparticle Research, Aachen

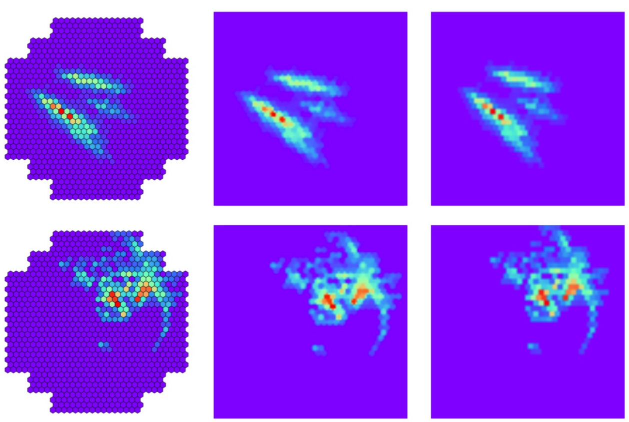
ERLANGEN CENTRE FOR ASTROPARTICLE PHYSICS
FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
NATURWISSENSCHAFTLICHE FAKULTÄT

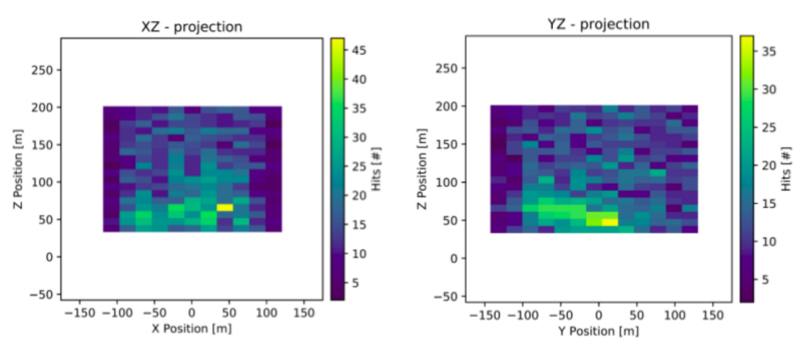
Motivation

Observation from multiple experiments:
subtle differences between data and simulations

- Simulations fine by eye & in distribution checks
- usually fine for analytically motivated reconstructions
- mostly fine for shallow learning (expert-designed features)

- Deep Learning can rely on subtle systematic differences
- Performance estimates can be unreliable





How to approach that?

1) Change nothing: see if pipeline behaves as expected on data

- Effort & doesn't solve the issue

2) Improve simulations

- Often not realistic

3) If distributions in data are known: classify a mixed set for data-MC

- Easy & fast
- Fine if classification fails - doesn't help otherwise

4) Train on calibration or reconstructed data

- Can introduce bias

5) Train **unsupervised autoencoder** on the data

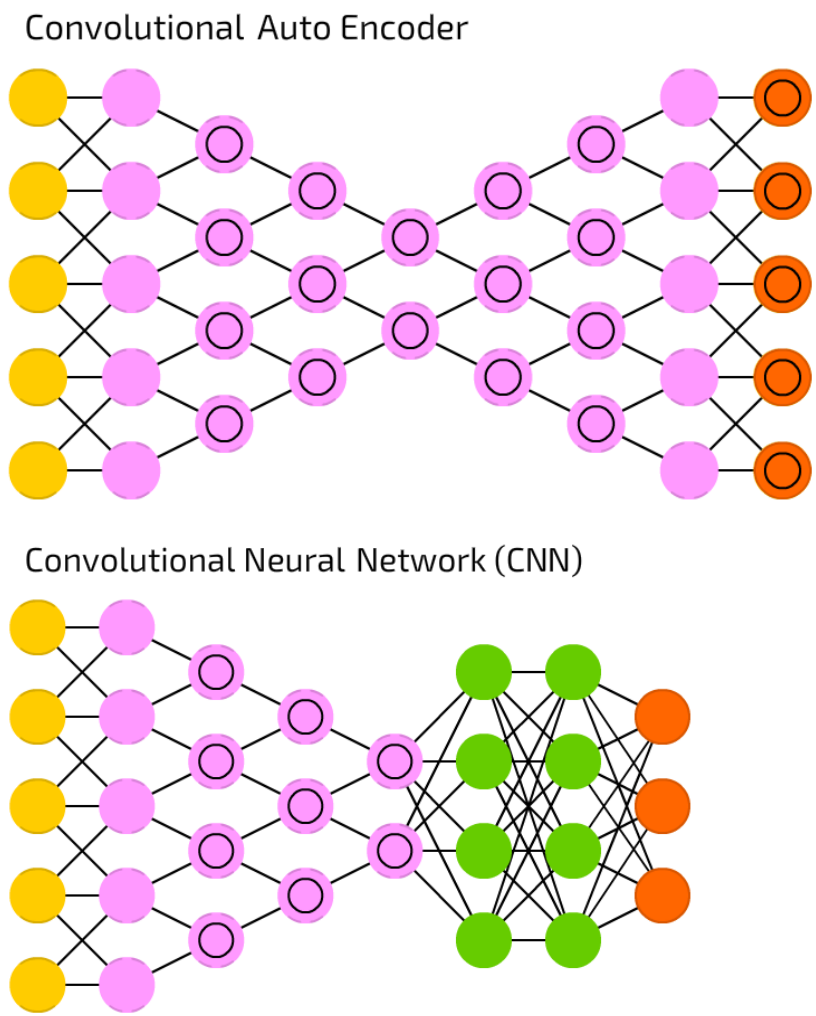
- Helps
- Comes at a cost

6) Maybe WGANs?

Unsupervised autoencoder

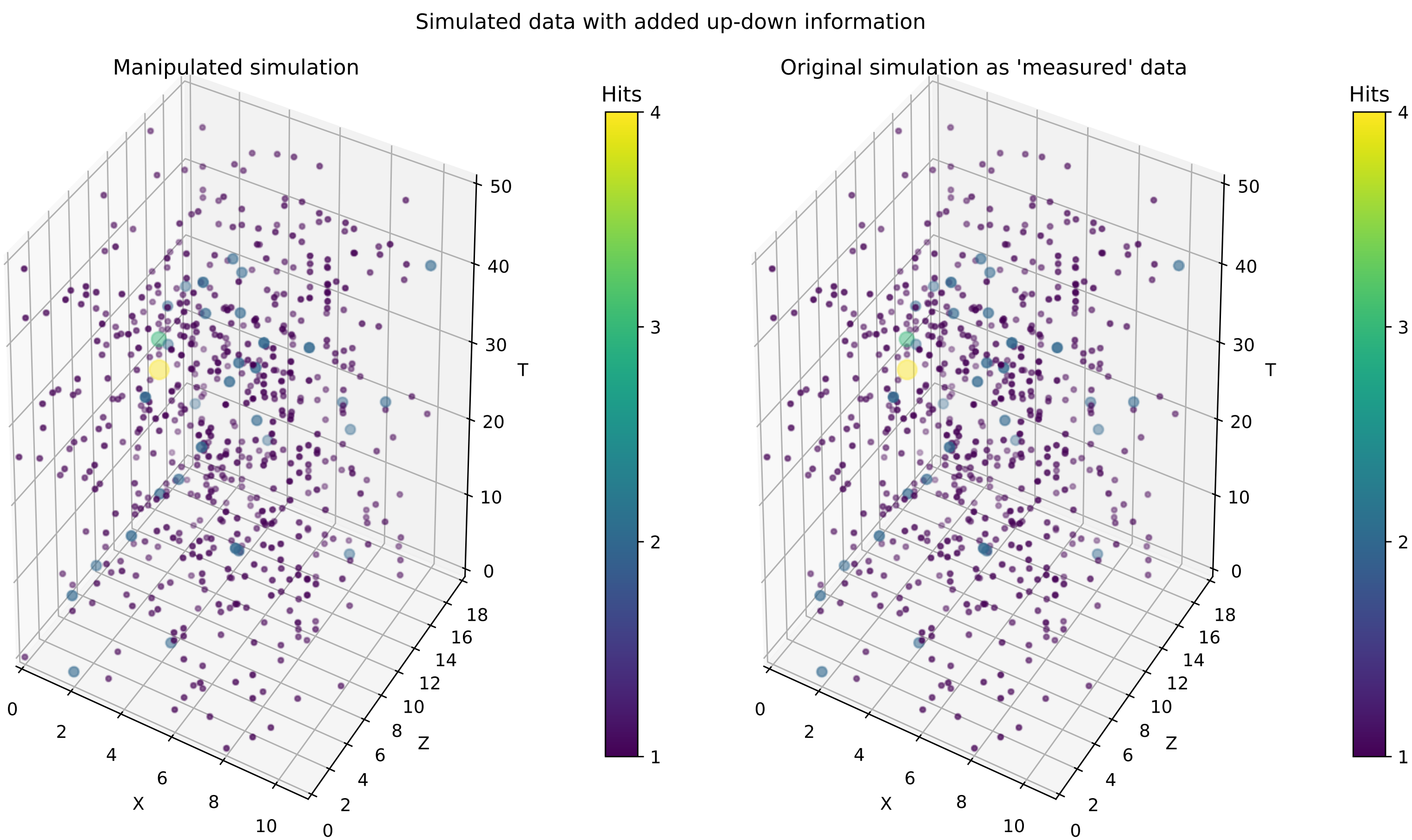
- Pretrain autoencoder on data
- freeze encoder part
- add (dense) layers
- train supervised

Claim: Insensitive to differences



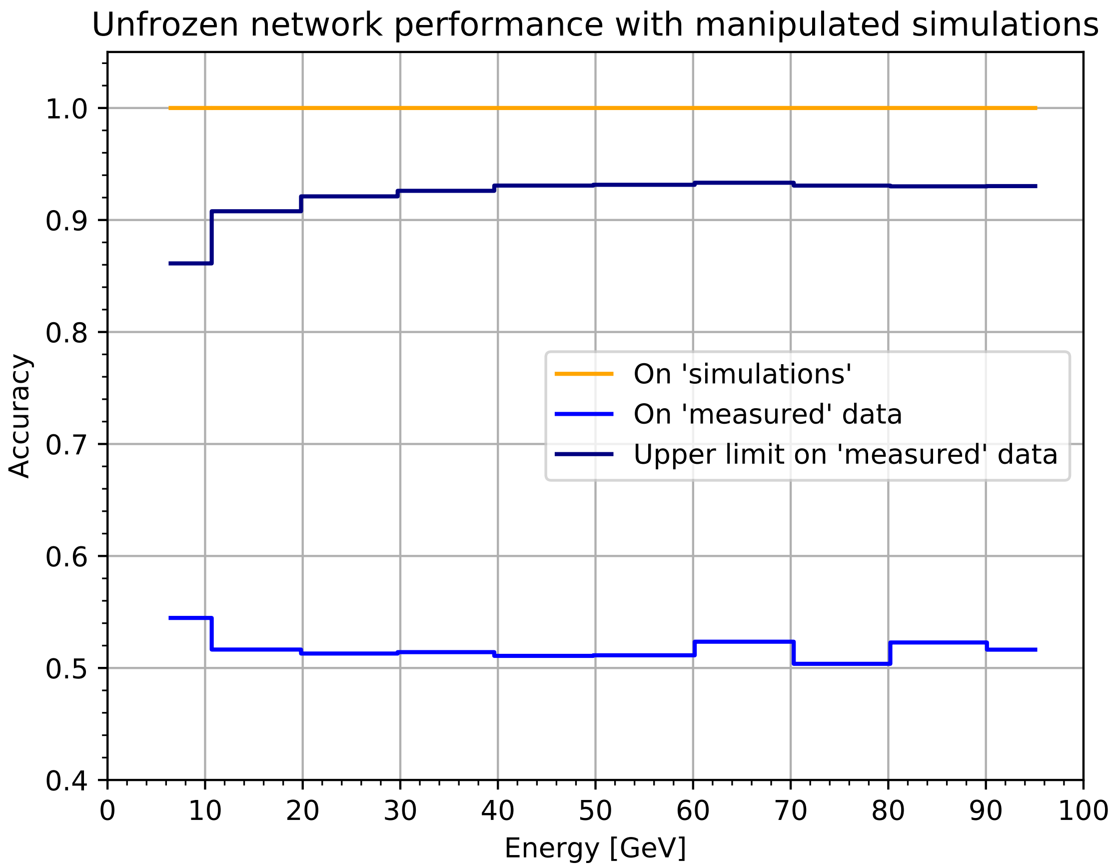
KM3NeT/ORCA test

- 4D XYZT data, in 3D projections
- Binary classification, set one bin to the correct class in “simulations”
- Worst case for supervised: Maximal correlation with target value



Does this work?

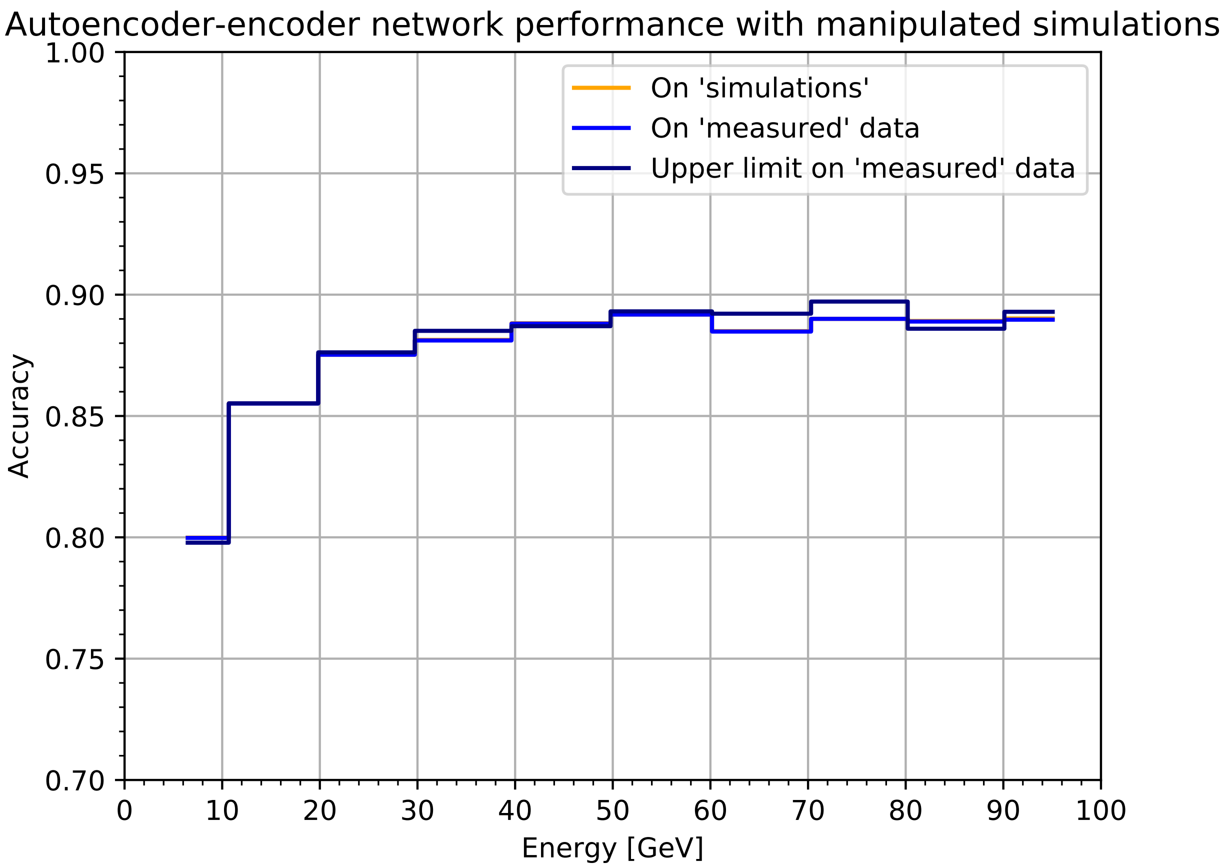
Supervised training on “simulations” (with true class information):



Perfect performance on “simulations”, random guessing for “data”

Does this work?

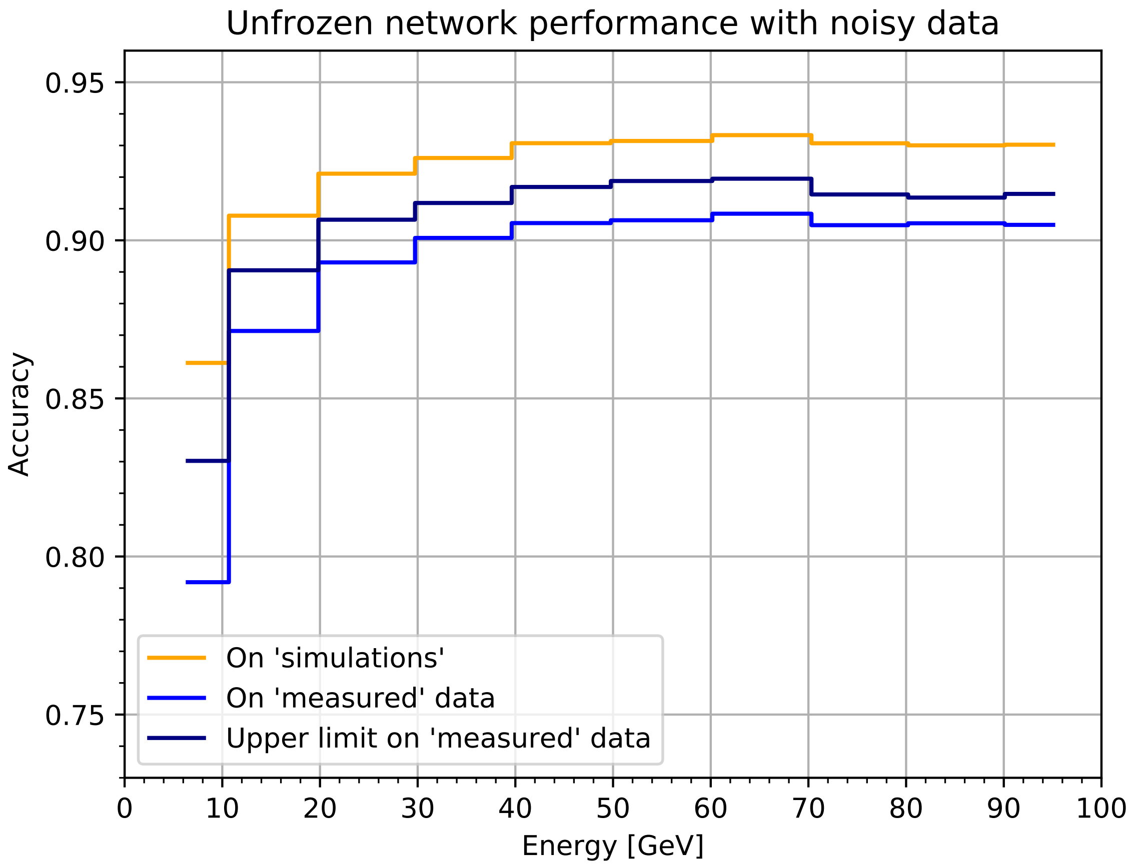
Training encoder unsupervised on “data”, dense supervised on “simulations”:



Hardly any change in performance!

Other example

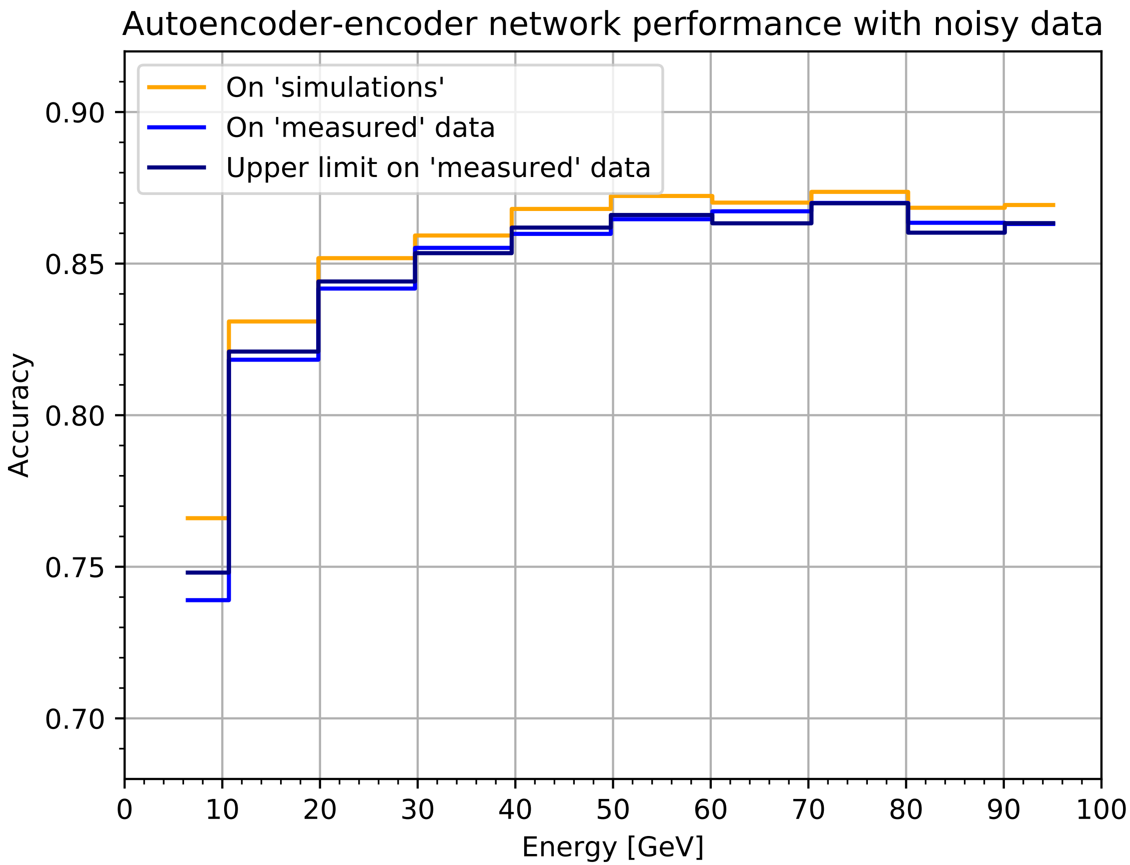
Noise for “simulations” 10 kHz, “data” 20 kHz
Best case for supervised learning: no correlation with target class



Still significant change in performance (+50% background)

Other example

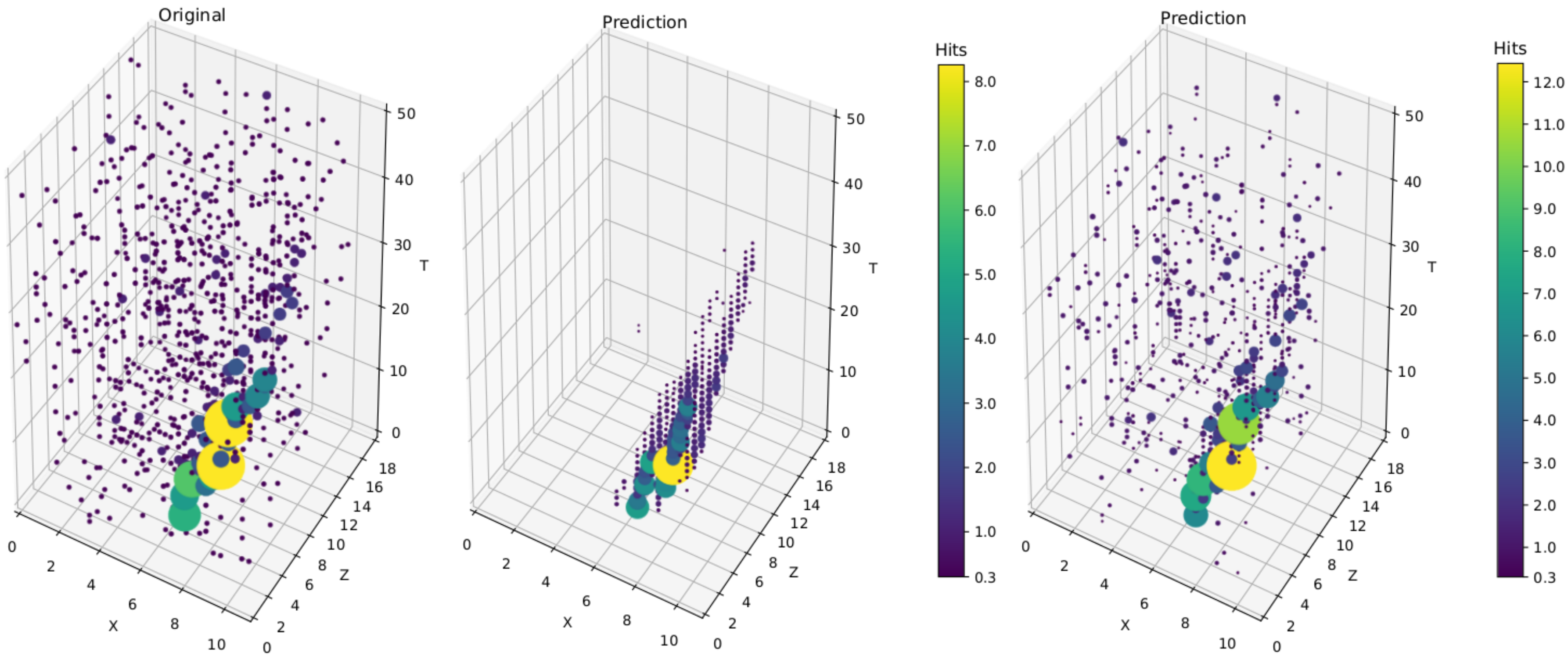
With unsupervised encoder training



Small changes, as good as trained directly on noisy “data”
The catch: Worse than supervised learning

Investigations

- Effect of different hyperparameters
- Data should contain all relevant signatures
- Autoencoder with smallest loss not always best for all tasks



All investigations by Stefan Reck @ ECAP, Friedrich-Alexander-University Erlangen-Nürnberg

Summary

- Subtle systematic deviations between data and sim. are common
- Deep Learning in danger to rely on them
  $\Rightarrow$ if so, estimates unreliable for real world application
- Unsupervised training of first layers on data
  - gives reliable estimates
  - doesn't reach best supervised performance so far
    (not always expected to be reachable)
  - optimization soon

Thank you for your attention!