Status and Plans of Area C: Deep Learning, Gain of knowledge by substantiated data-driven methods

Gregor Kasieczka
(gregor.kasieczka@uni-hamburg.de)

IDT ErUM-Data Pilot / Collaboration Meeting

2019-09-30

CLUSTER OF EXCELLENCE
QUANTUM UNIVERSE

Emmy Noether-Programm
Deutsche Forschungsgemeinschaft
DFG

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

PIER
Partnership of Universität Hamburg and DESY

Bundesministerium für Bildung und Forschung

Deep Learning in Particle Physics:

Why *are we doing it?*

- Better results due to exploiting correlations and low level inputs
- New ways of analysing data
    - Trying to solve more difficult problems
- Resource efficiency
    - Faster decision making and simulation
    - Likely no great increase in personell or computing budgets
    - End of Moore's law?

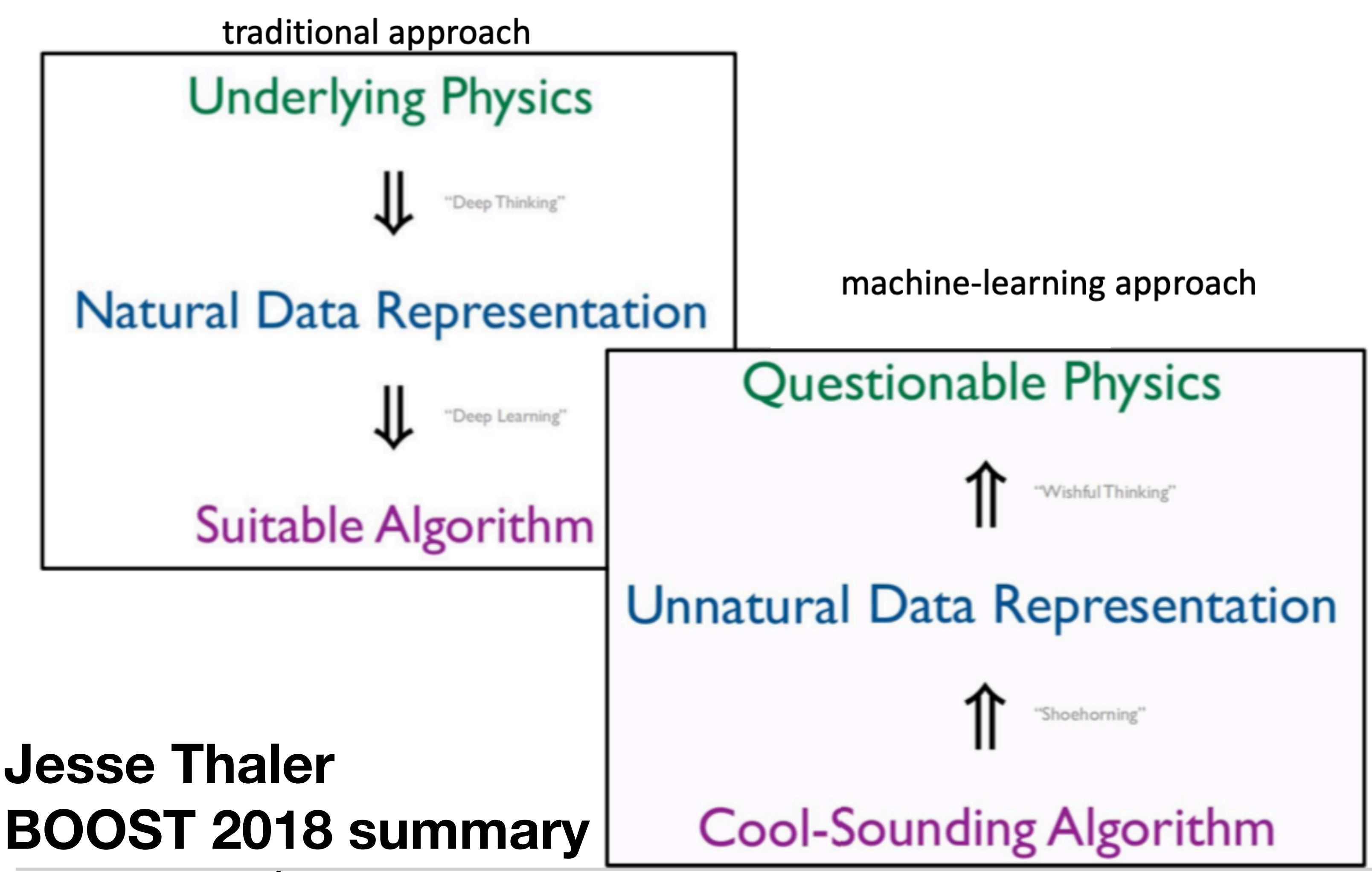
Deep Learning in Particle Physics:

*Why are **we** doing it?*

- "Infinite" amounts of high quality training data
- Interesting structured data at multiple scales
- Detailed understanding of systematic uncertainties

What is the "correct" way to represent our data for ML applications?

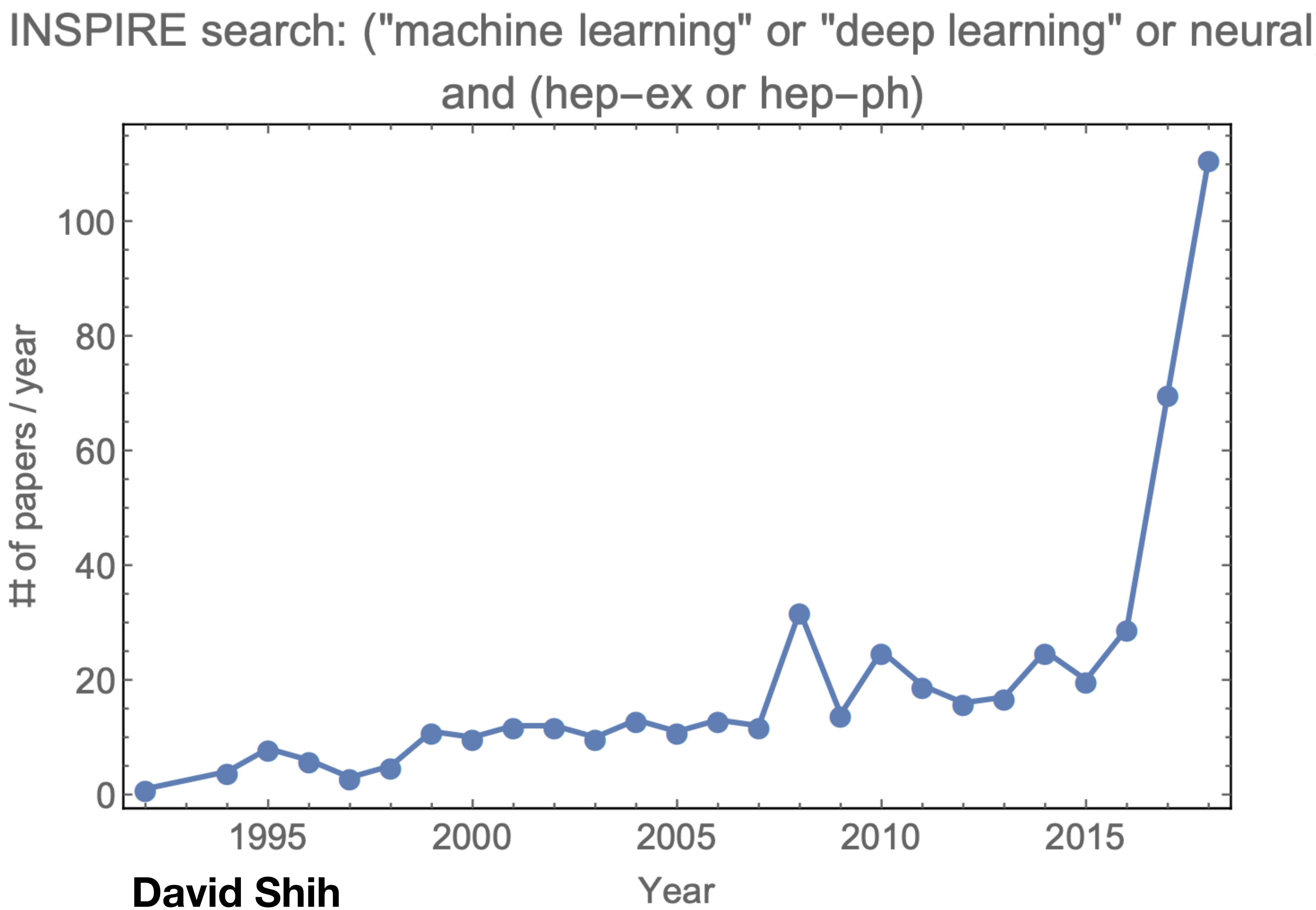
(will be problem specific)
(would not rely on CS to solve it for us)



Jesse Thaler
BOOST 2018 summary

Deep Learning in Particle Physics:

*We **are** doing it.*



David Shih
BOOST 2019

Overview

- Classification & Regression
- Generation
- Anomaly detection
- Robustness
- Fast Inference & Tools

C1) Sensornahe Verarbeitung von Daten • Signalfilter, Rauschunterdrückung • Verarbeitung von zeitabhängigen Signalen	**C2) Objektrekonstruktion** • Spur- und Clusterrekonstruktion, Jetbildung, Ereignisrekonstruktion • Fragestellungen für Anordnung, Reihenfolge, Zuordnungen von Daten • Optimierungen zur Extraktion kleiner Signale bei großem Untergrund
C3) Netzwerkbeschleunigte Simulationen • Generative adversarial networks, Anpassung von Simulationen an Datenverteilungen • Evaluationsverfahren für die Qualität der Netzwerksimulationen	**C4) Qualität von Netzwerkvorhersagen** • Reduzierung experimenteller systematischer Unsicherheiten • Spezielle Lernstrategien • Vorhersagenrelevante Information • Unsicherheiten von Vorhersagen

Many thanks to all groups for sending material!!

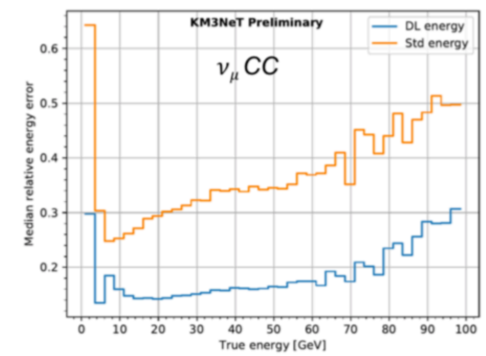
Classification & Regression

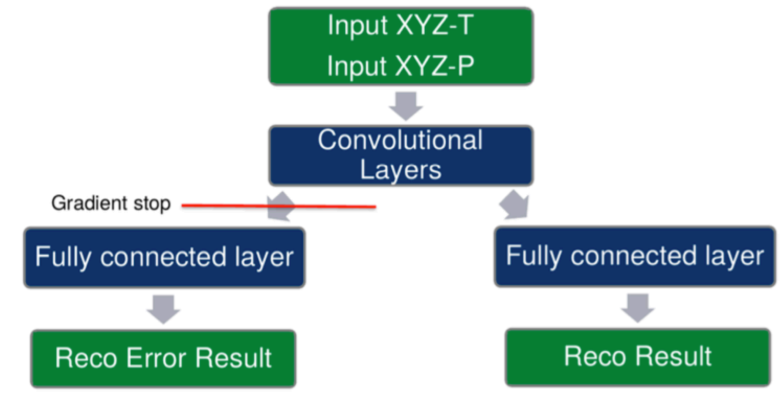
ECAP: deep learning in astroparticle physics

ERLANGEN CENTRE FOR ASTROPARTICLE PHYSICS

- KM3NeT (ORCA) – neutrino oscillation for determination of mass hierarchy
  - **Michael Moser, Steffen Hallmann, Stefan Reck, Thomas Eberl, Gisela Anton**
  - Deep CNNs
    - Classification (background, neutrino types)
    - Regression (energy, direction)

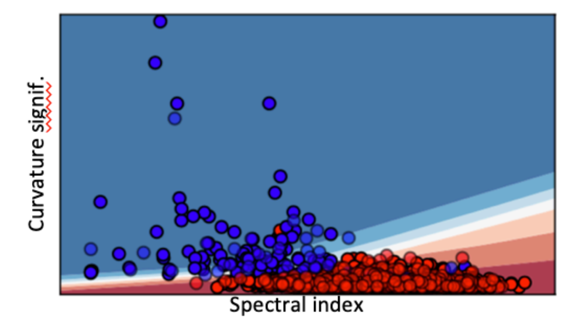
DNN energy reconstruction



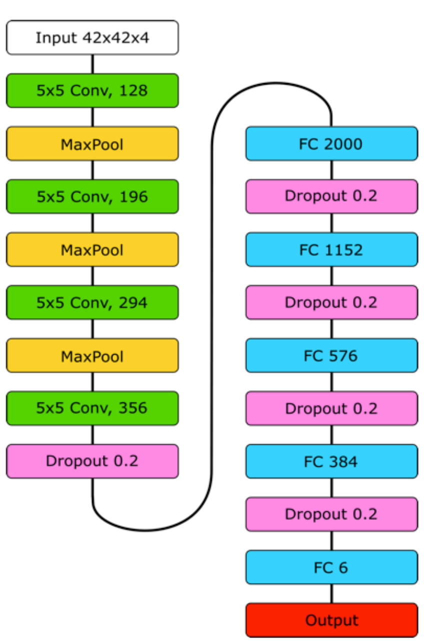


- Fermi LAT – gamma-ray astrophysics
  - **Aakash Bhat, Dmitry Malyshev**
  - Machine learning
    - Classification of unidentified point sources

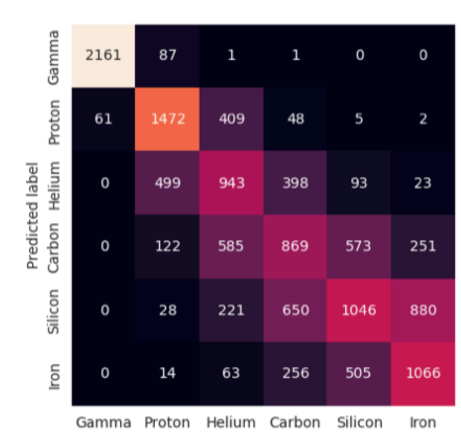
NN classification



- H.E.S.S. – high energy gamma-ray astrophysics
  - **Christina Hillig, Matthias Buchele, Stefan Funk**
  - Deep CNNs
    - Classification of showers (gamma rays, different types of nuclei)
    - CR energy reconstruction



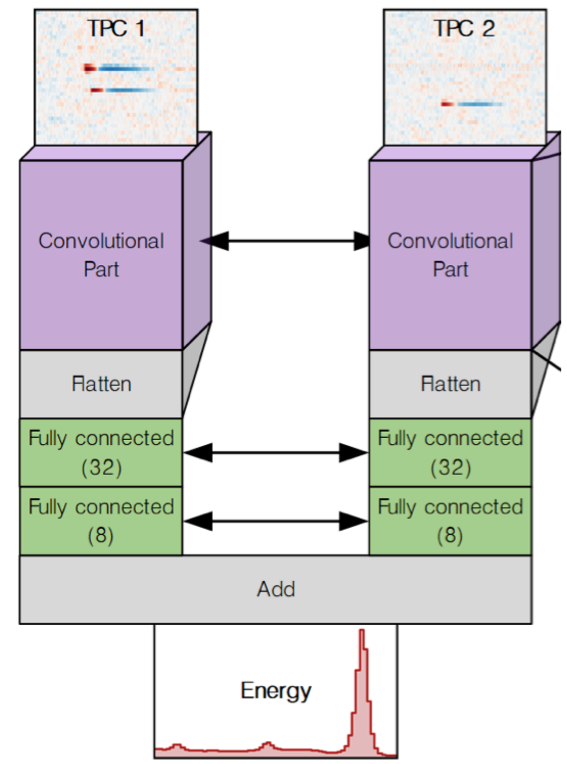
DNN CR classification



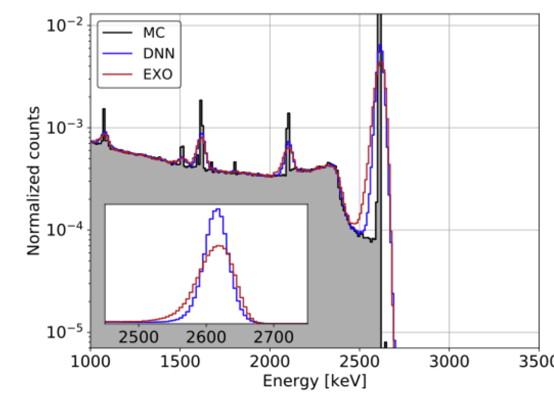
ECAP: deep learning in astroparticle physics

ERLANGEN CENTRE FOR ASTROPARTICLE PHYSICS

- EXO – double beta decay
- **Tobias Ziegler, Federico Bontempo, Thilo Michel, Gisela Anton**
  - Deep CNN
    - Background rejection
    - Energy reconstruction
  - GANs:
    - Monte Carlo improvement

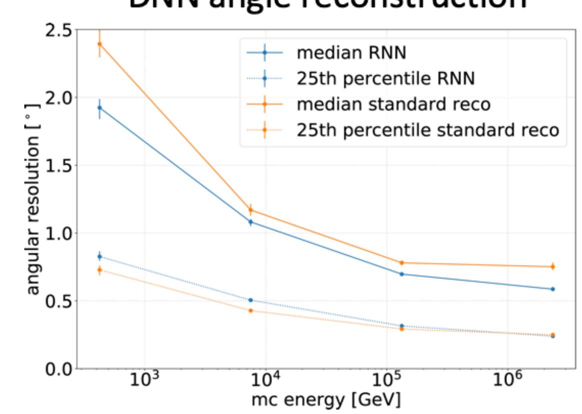


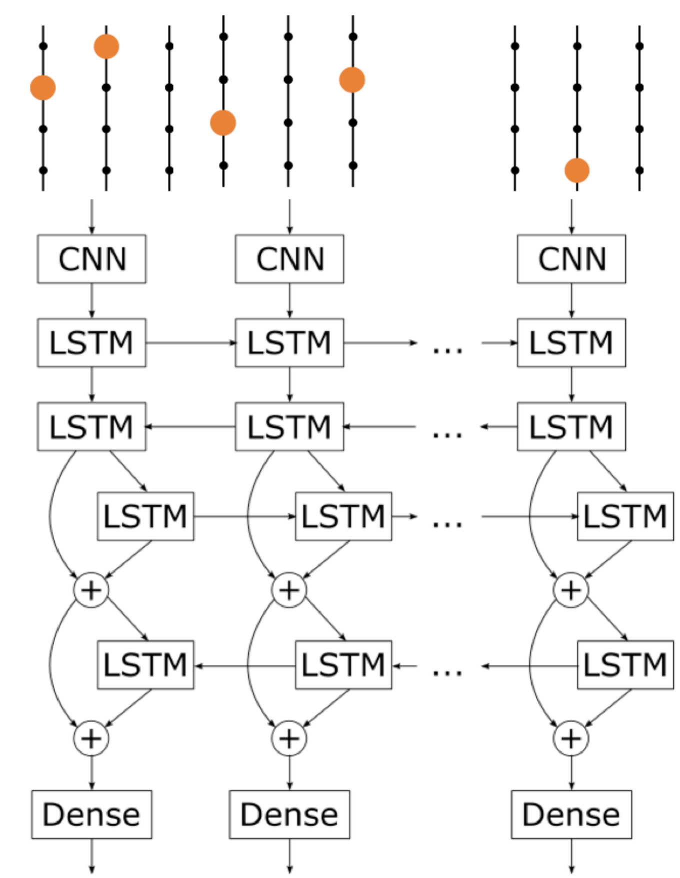
DNN energy reconstruction



- IceCube – high energy astronomical neutrinos
  - **Gerrit Wrede, Thorsten Glüsenkamp, Gisela Anton**
  - Deep CNN + recurrent neural networks (LSTM)
    - Direction reconstruction
    - Muon energy loss in ice

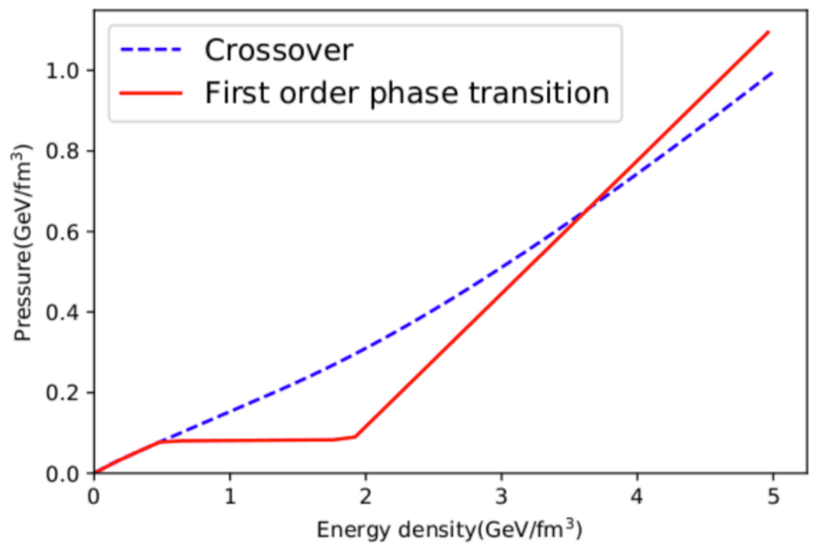
DNN angle reconstruction

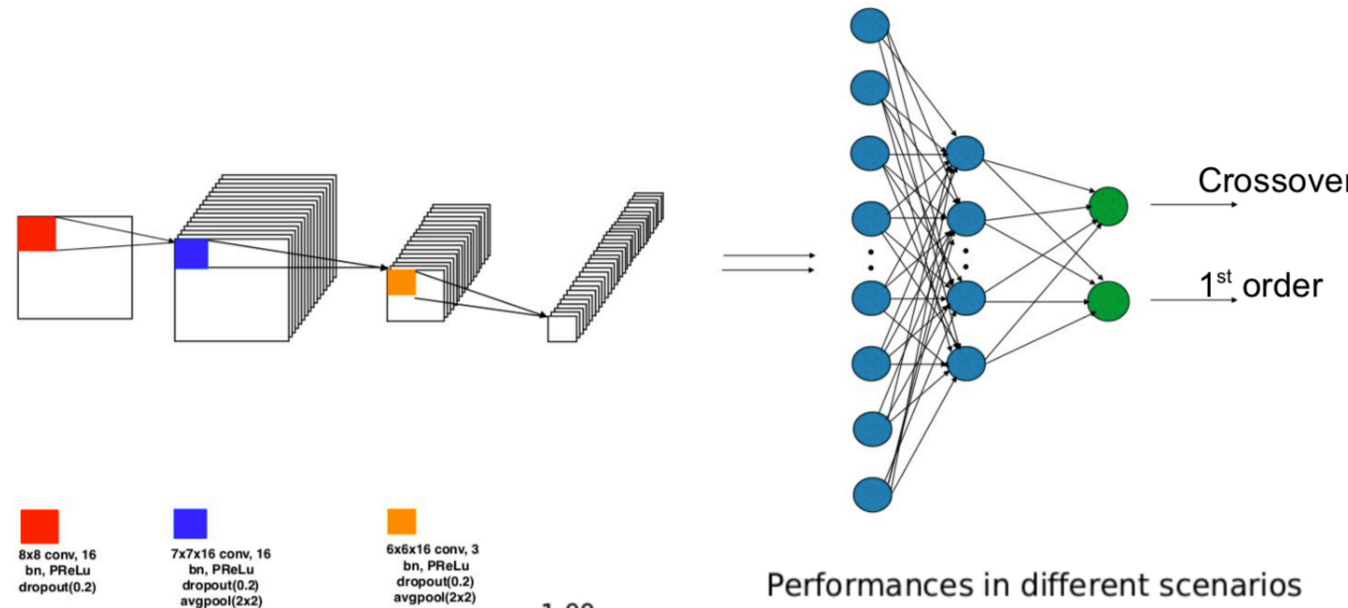




Projekts 2: Identifying nature of QCD transition in HIC with Deep Learning

(Paper to be online soon)



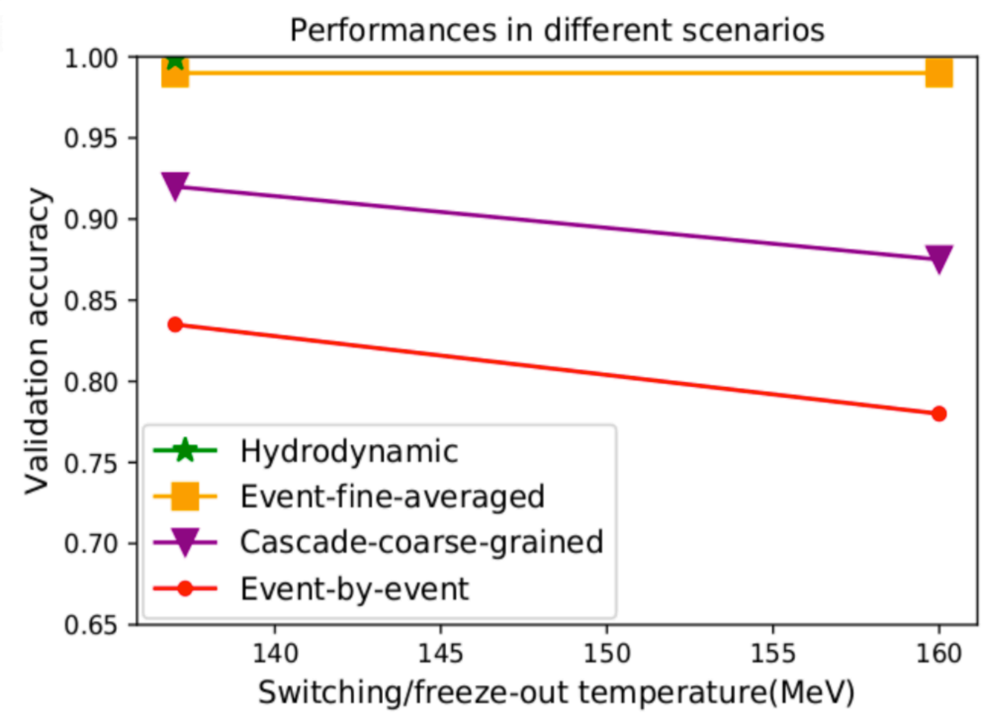


1) hadronic cascade "afterburner" is considered:
finite number of particles, resonance decays

2) Information about EoS in early dynamics is **not swiped away** inside the final-state pion spectra – *perspective from deep CNN*

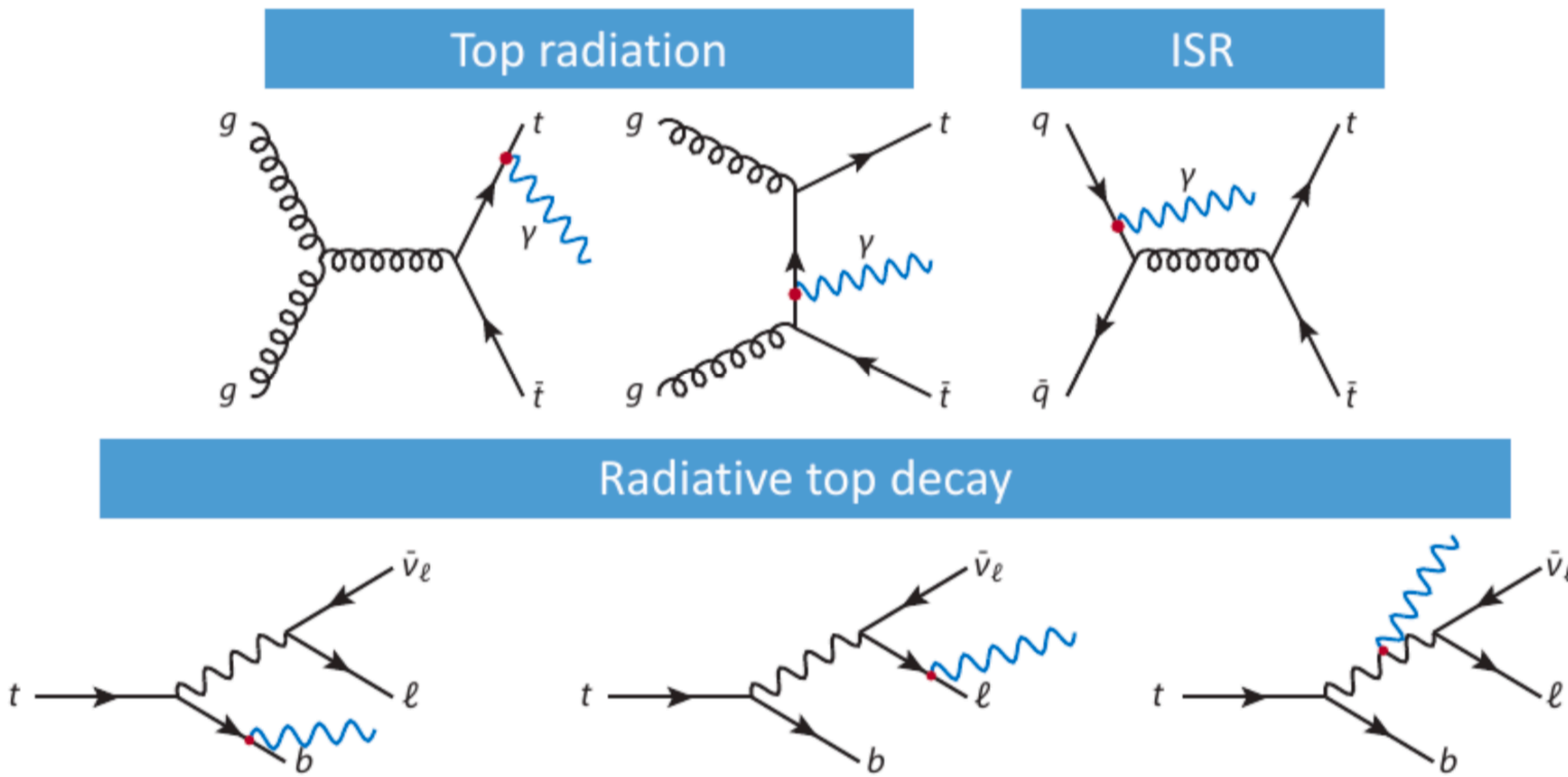
3) more **stochasticity** from resonance decays and elongated hadronic cascade **diminish** the correlation

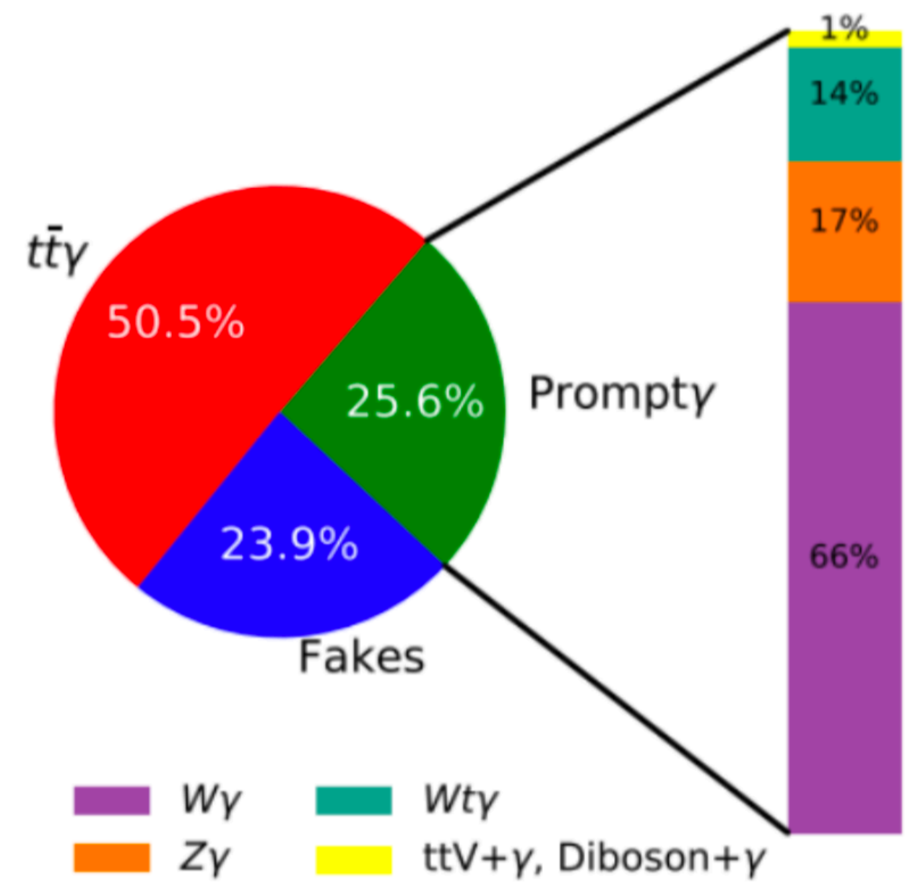
4) **enhancement of statistics** and **reduction of fluctuations** helps **facilitating** the revealing of EoS information via deep CNN



Example: $t\bar{t}$+γ

- tt+photon production sensitive to electroweak coupling of the photon

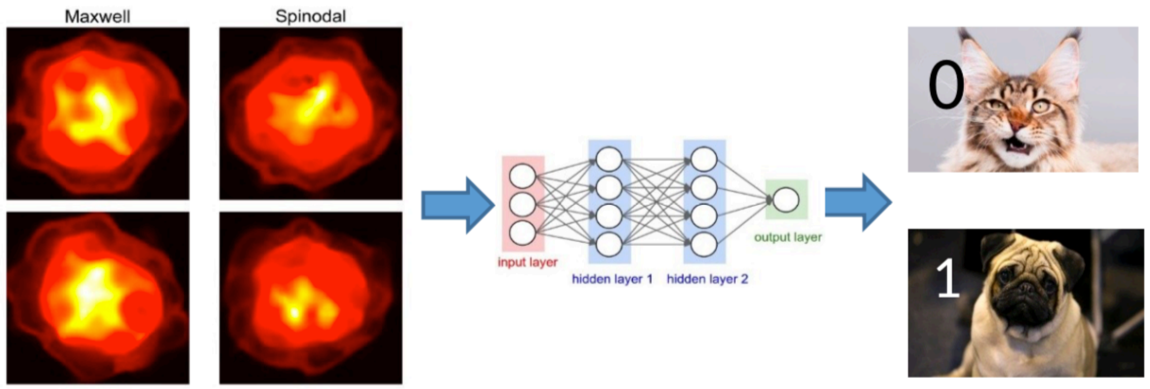


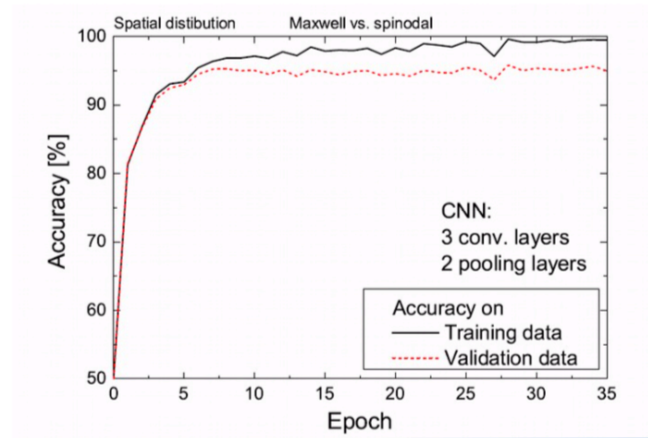


- Application of machine learning:
  - Identification of prompt photons
  - Multi-class classification of signal and background processes
  - Differentiation of photon origin

Projekts 1: Identifying Spinodal clumps in HIC using deep learning

(Paper is with journal, should be published soon)



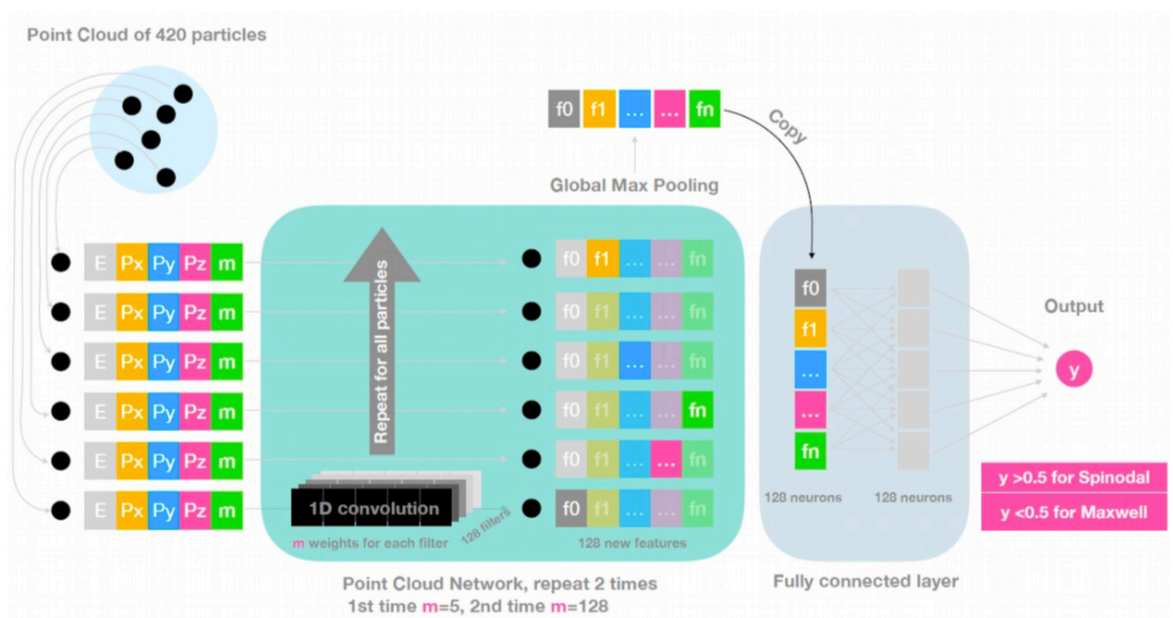


A CNN works well to identify the clumps in coordinate space event-by-event > 94% accuracy

- But: experiments measures discrete particles/tracks.
- Translating this into a 2D image for a CNN may loose information.

Use a point cloud network

- Each particle has 5 features
- For events with less than the maximum particles, pad with zeros.
- Can deal with discrete particles and varying list length.



- The PCN works as good (if not better) than the CNN.
- Overall loss of information in momentum space (see talk tomorrow)

Going beyond FCN/CNNs:

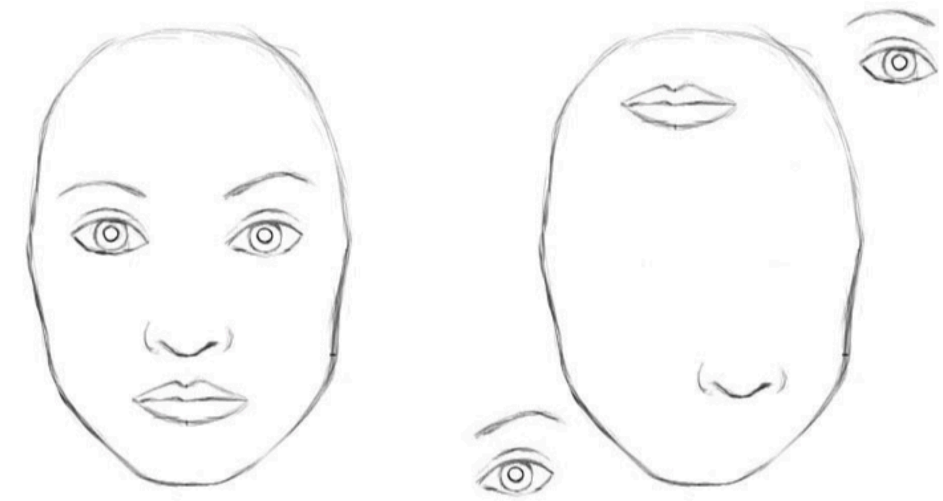
- **Point Cloud:**
  - **1D CNN + Max pooling**
  - **Similar to CMS flavour tag**
  - **(DeepSet approaches)**

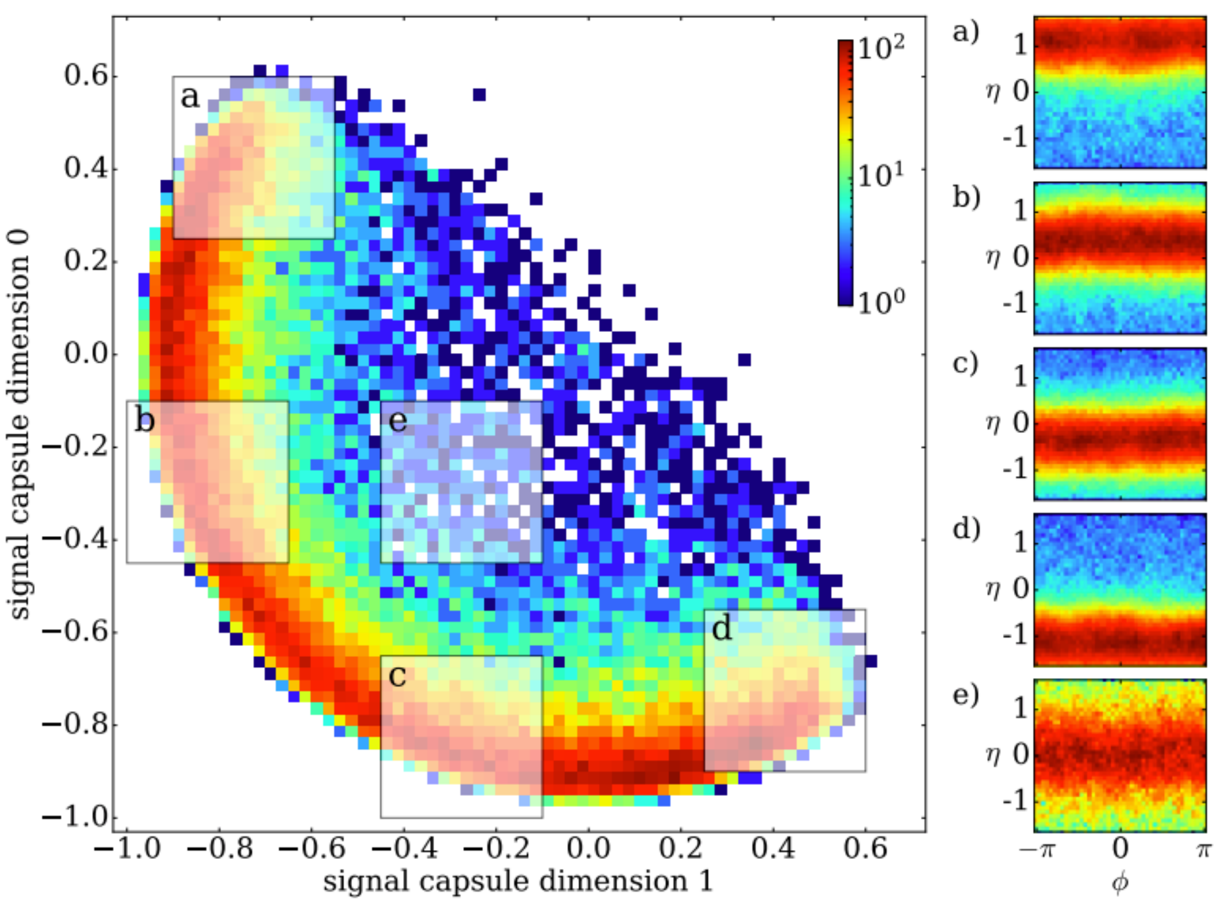
FIAS Frankfurt Institute for Advanced Studies

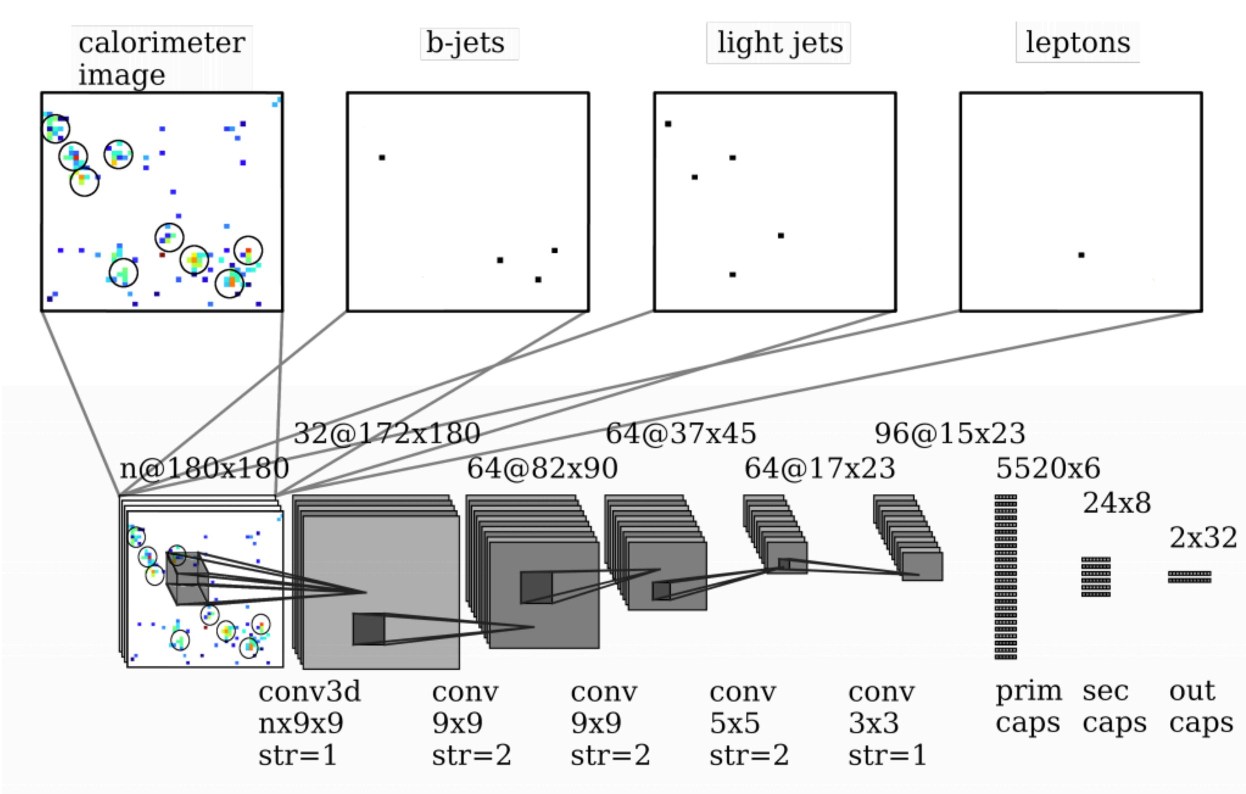
Capsule Networks

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

- Particle flow inputs for per-event classification
- Capsule Networks
  - Alternative to convolution (usual image technique)
  - learn instantiation vector
  - Interpreteable
- Calibration?

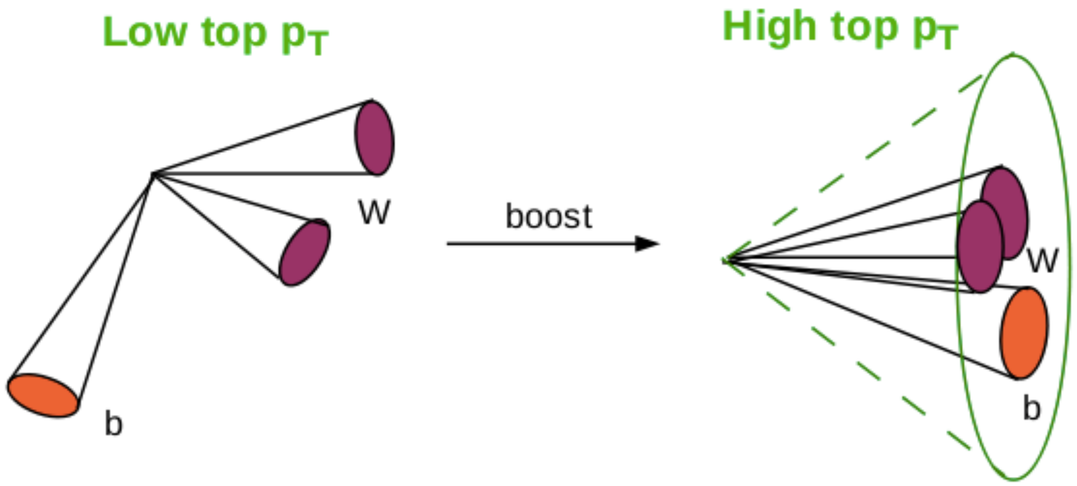




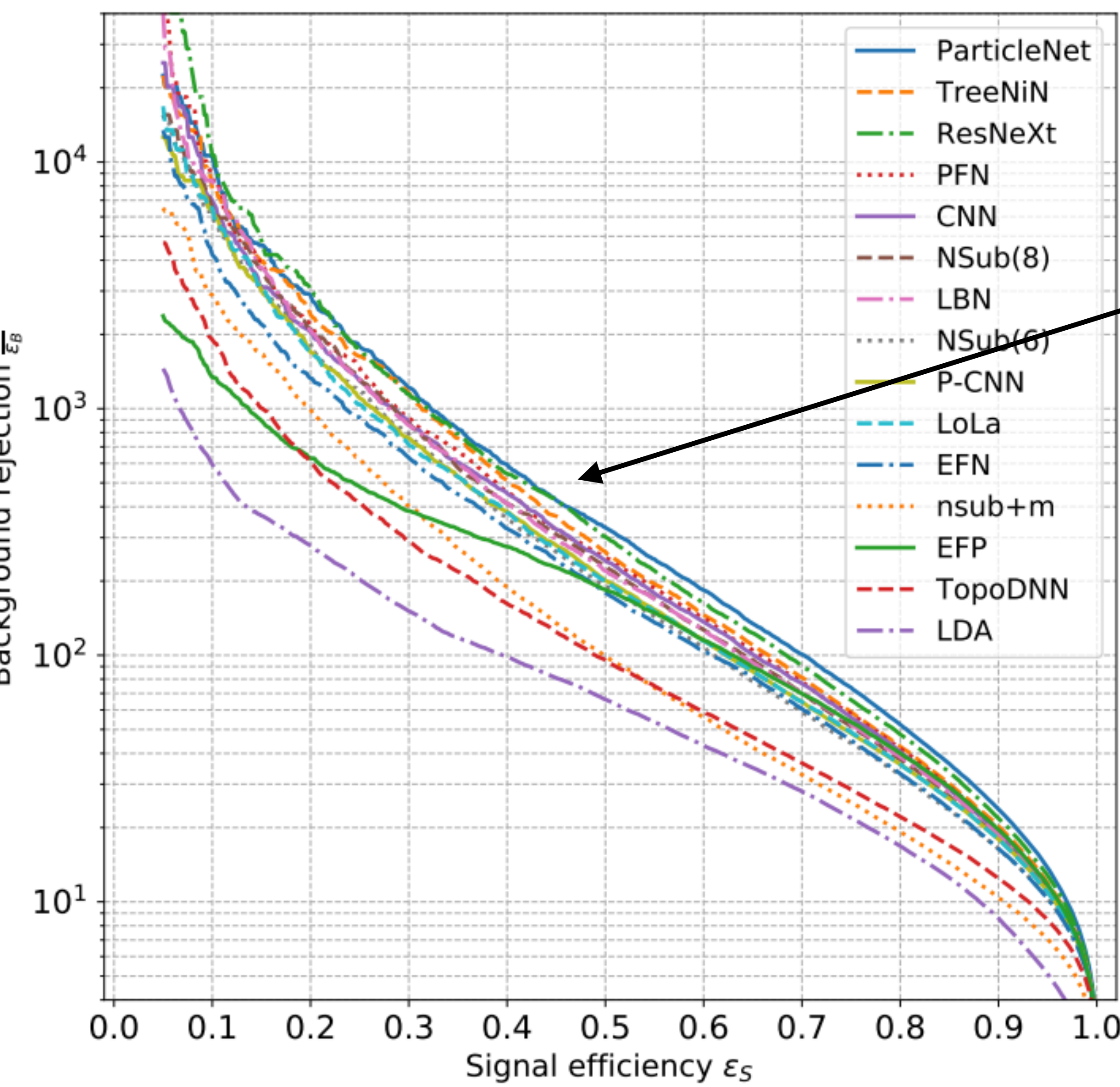


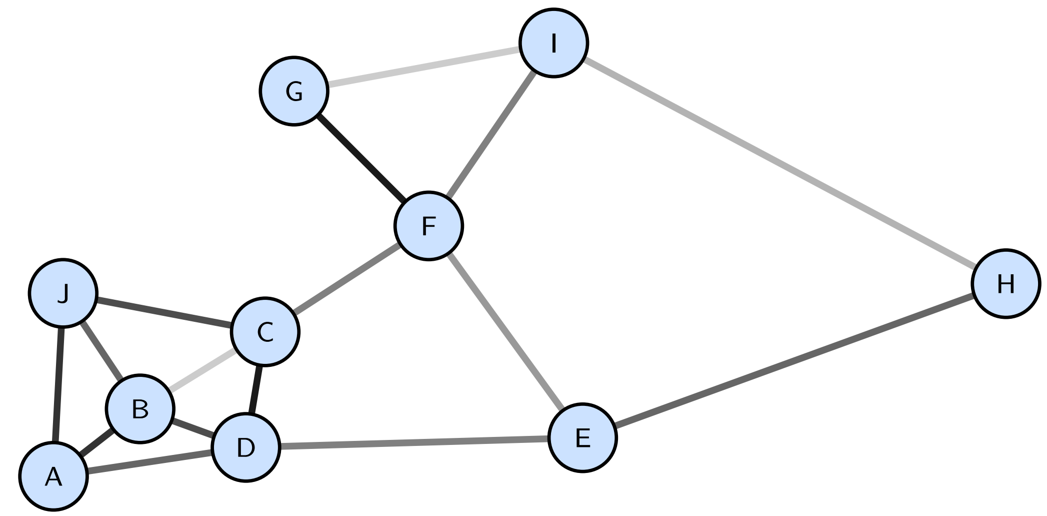
(1906.11265, with Tilman Plehn)

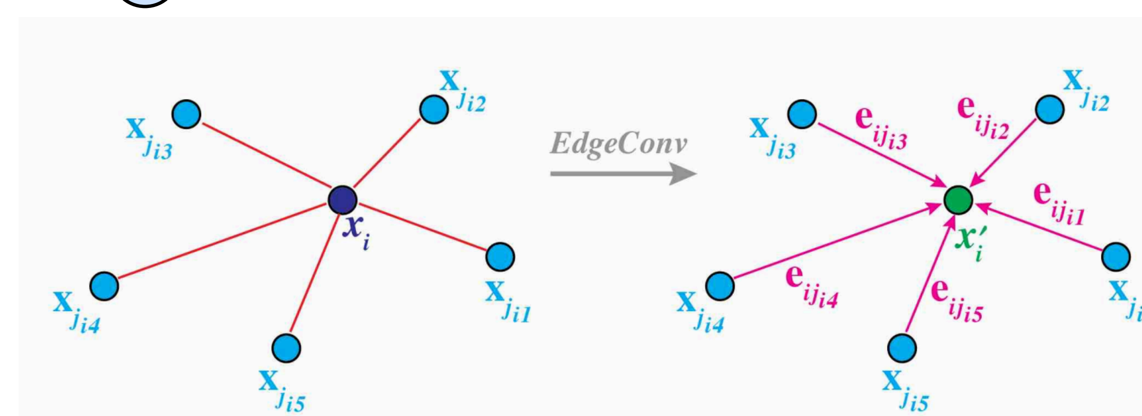
Heavy Resonance Tagging



Community performance comparison (toy dataset public): 1902.09914



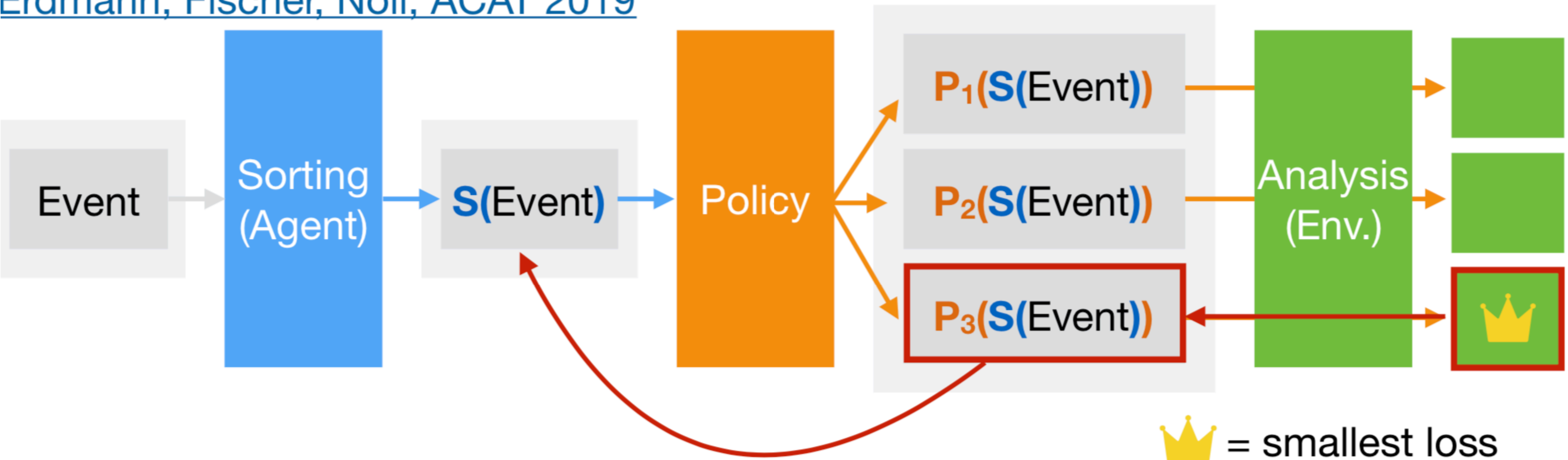


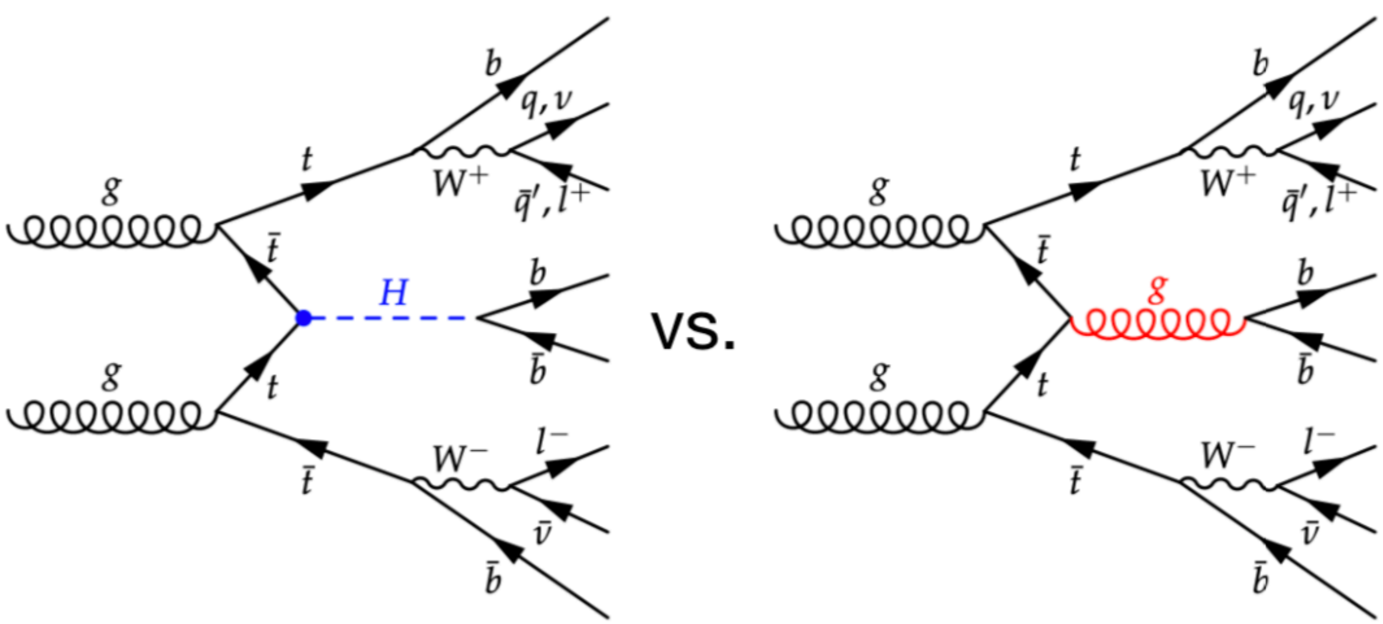


Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

CMS: Reinforced Particle Sorting

Erdmann, Fischer, Noll, ACAT 2019

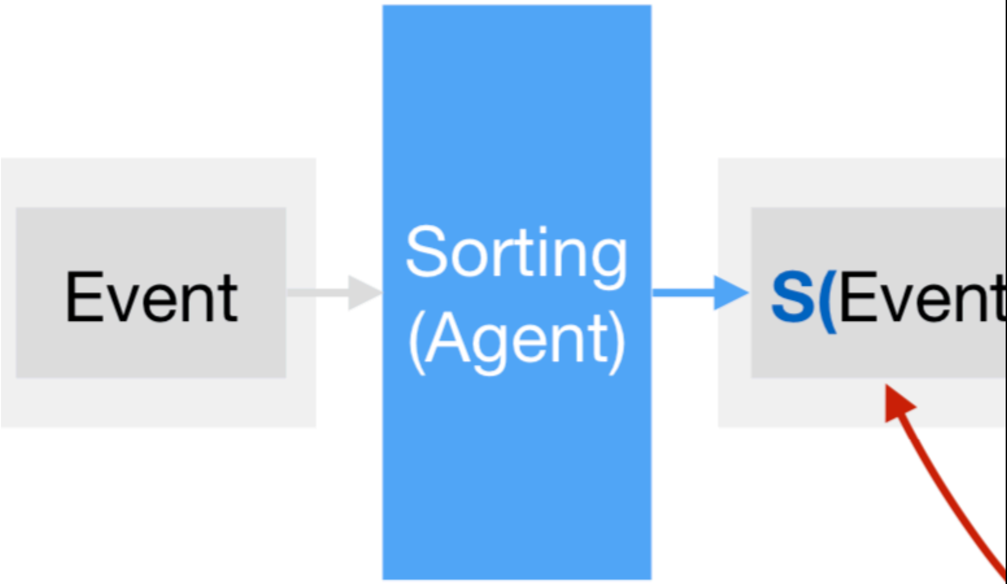


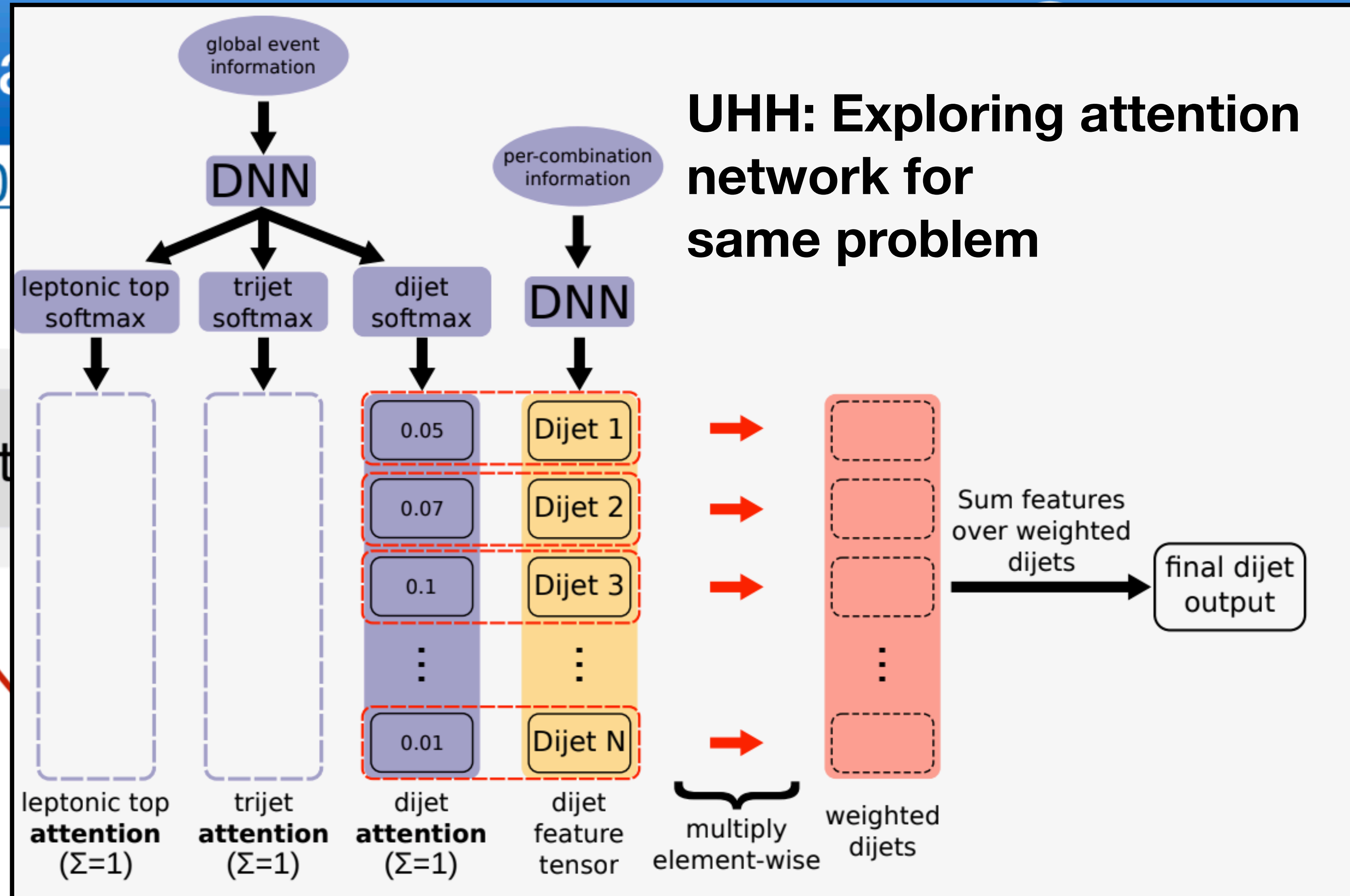


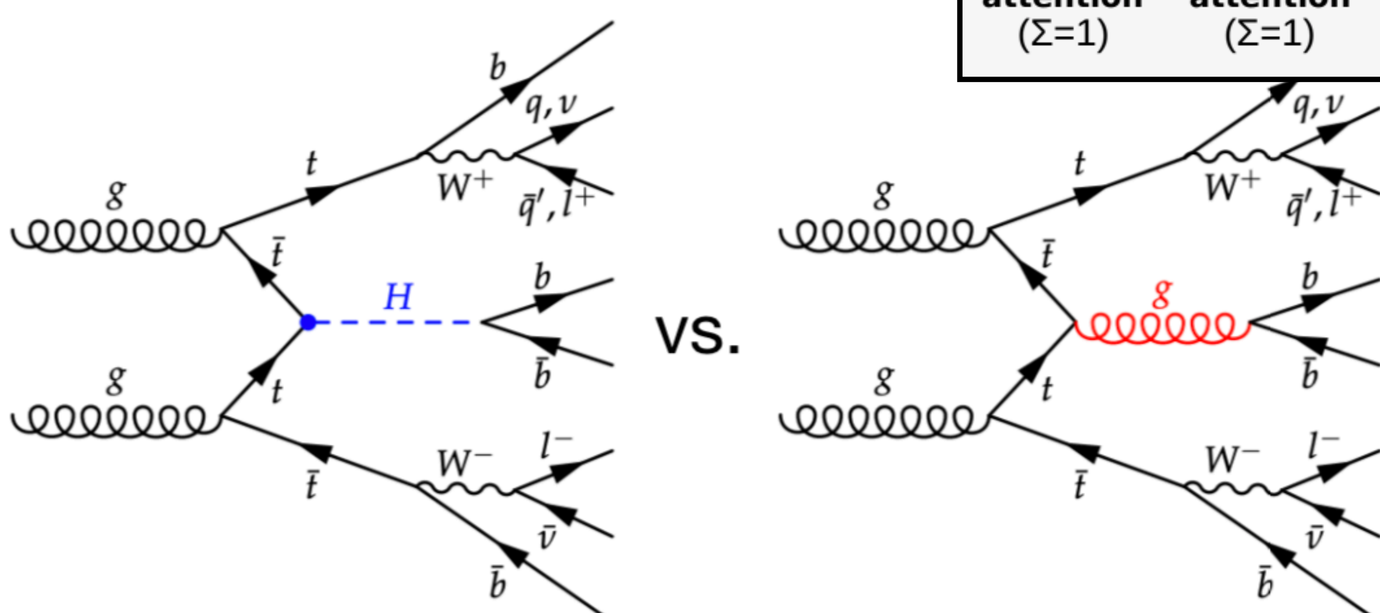
Classification performance improves with proper particle sorting!

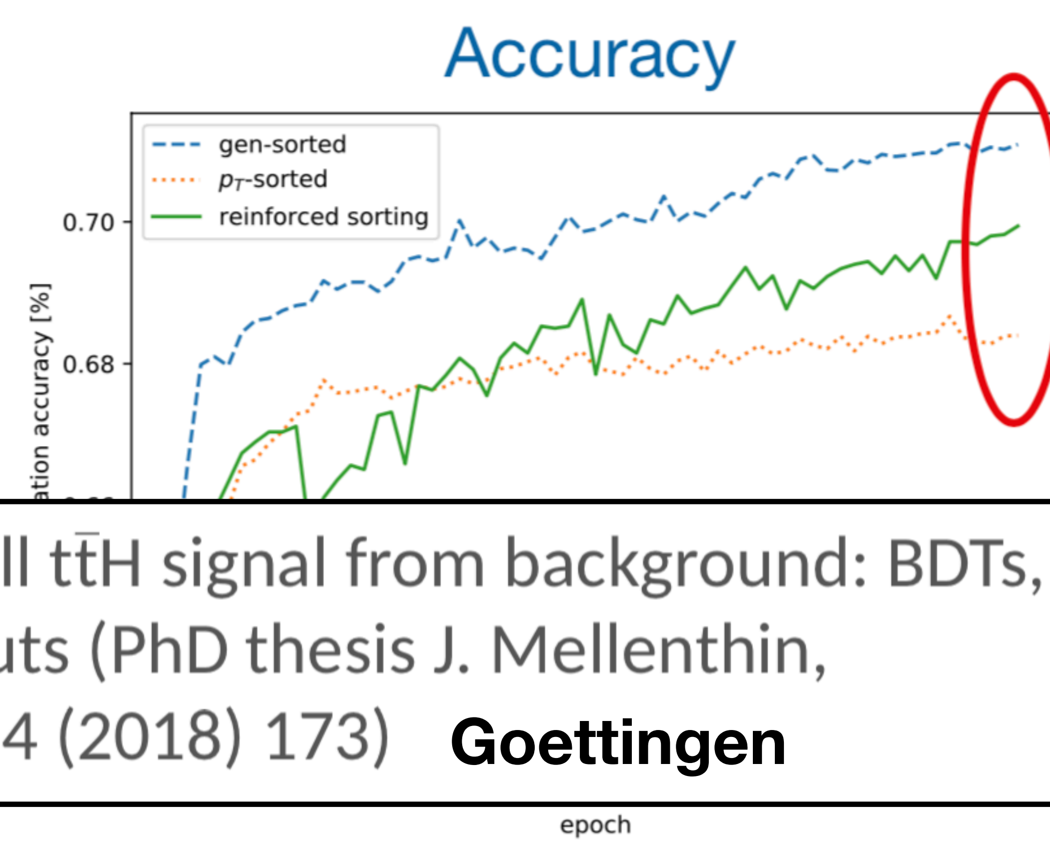
CMS: Reinforced Pa

Erdmann, Fischer, Noll, ACAT 20







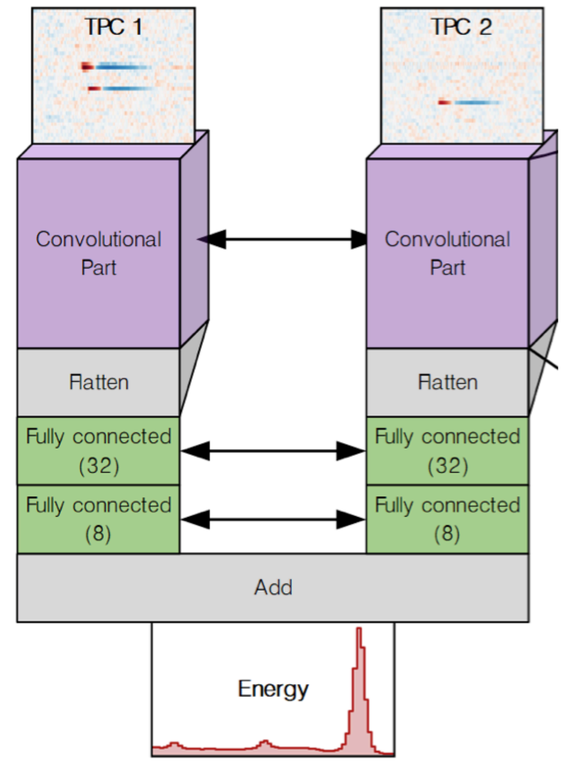


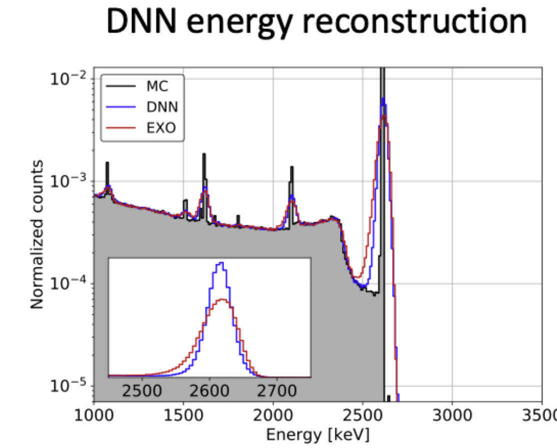
– Studied several approaches to distinguish small tt̄H signal from background: BDTs, shallow and deep NNs, low and high-level inputs (PhD thesis J. Mellenthin, publication of ttH observation, Phys. Lett. B 784 (2018) 173) **Goettingen**

Generation

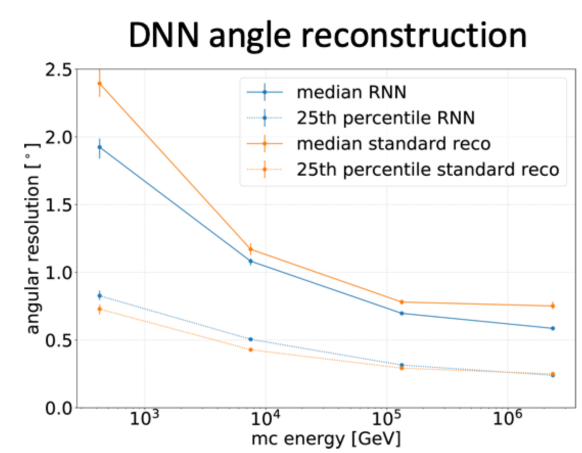
ECAP: deep learning in astroparticle physics

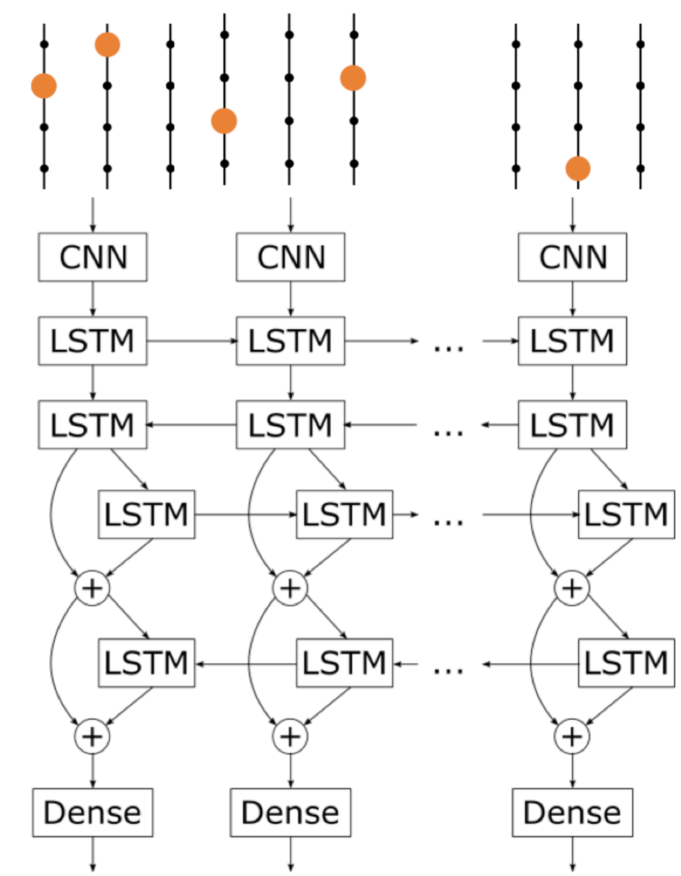
- EXO – double beta decay
- **Tobias Ziegler, Federico Bontempo, Thilo Michel, Gisela Anton**
  - Deep CNN
    - Background rejection
    - Energy reconstruction
  - GANs:
    - Monte Carlo improvement





- IceCube – high energy astronomical neutrinos
  - **Gerrit Wrede, Thorsten Glüsenkamp, Gisela Anton**
  - Deep CNN + recurrent neural networks (LSTM)
    - Direction reconstruction
    - Muon energy loss in ice





PXD background simulation using GANs at Belle II

M Srebre, T Kuhr, M Ritter, P Schmolz — Ludwig-Maximilians-Universität München

Pixel Vertex Detector (PXD)

- innermost subdetector, 2 layers of modules
- needed for trajectory and decay vertex reconstruction

Currently: simulating PXD background

- **Status quo**:
  - ▷ MC simulated PXD background based on physics processes
  - ▷ limited amount of individual samples
- **Problems**:
  - ▷ fine-grained pixel modules →PXD data is high in volume
  - ▷ input for every simulation, costs bandwidth and is slow
  - ▷ bias through repeatedly using same background samples

Generating background

Objective: train neural network to generate PXD background rather than simulating with Monte Carlo

- tested several GAN models (vanilla, DCGAN, WGAN), faced convergence difficulties
- exploring **Wasserstein Generative Adversarial Network with gradient penalty** (WGAN-GP) →successful, validation on-going
- efficient way to generate noise on-demand for every signal event
- realistic, statistically unbiased background

Dataset

- 40 × 100 000 PXD module outputs
- 250 × 768 gray-scale hitmaps (*"images"*)
- when PXD measurements available: train GAN with random-trigger measured samples of background processes

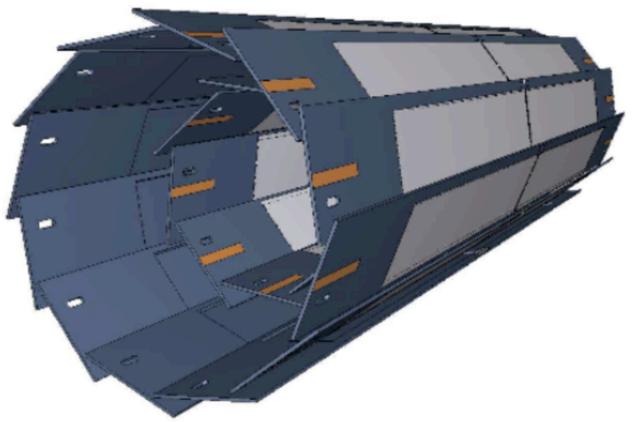
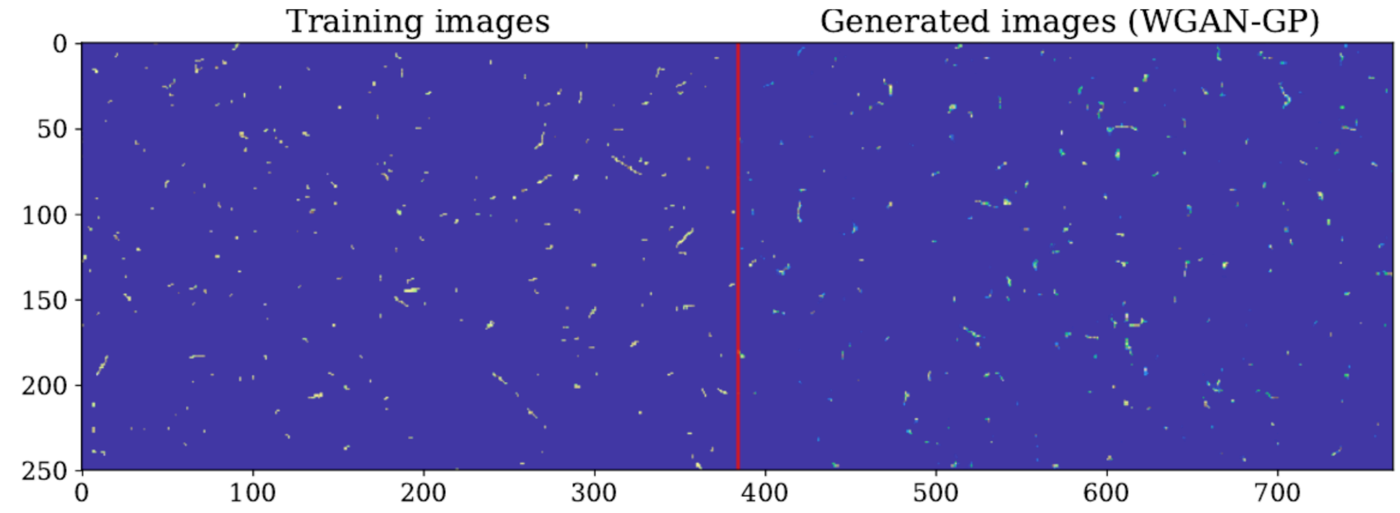
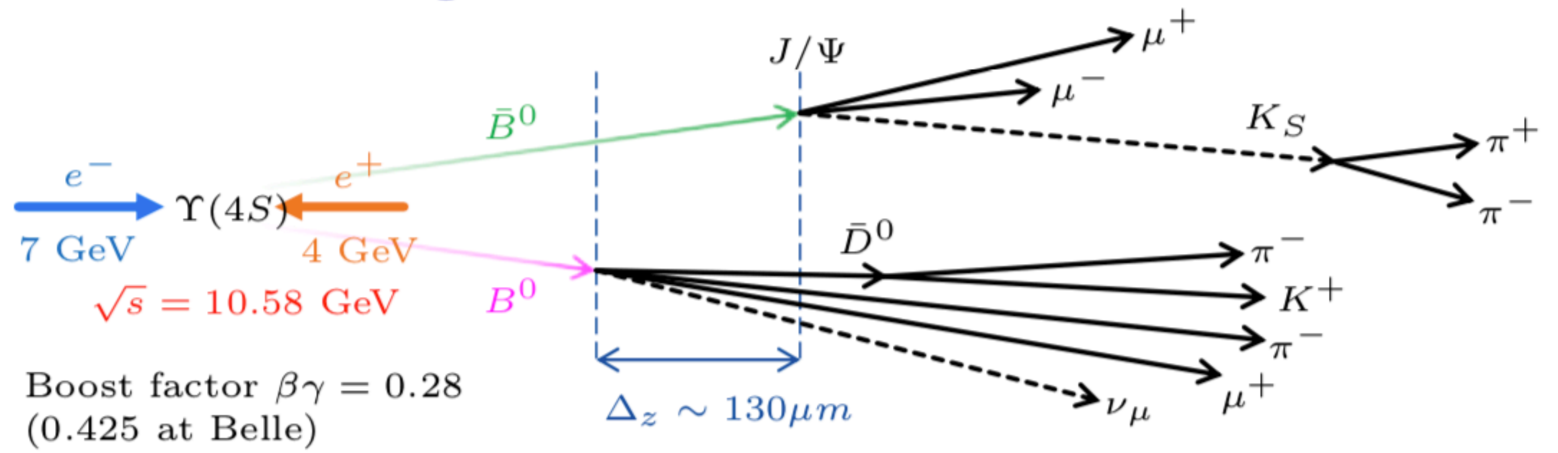


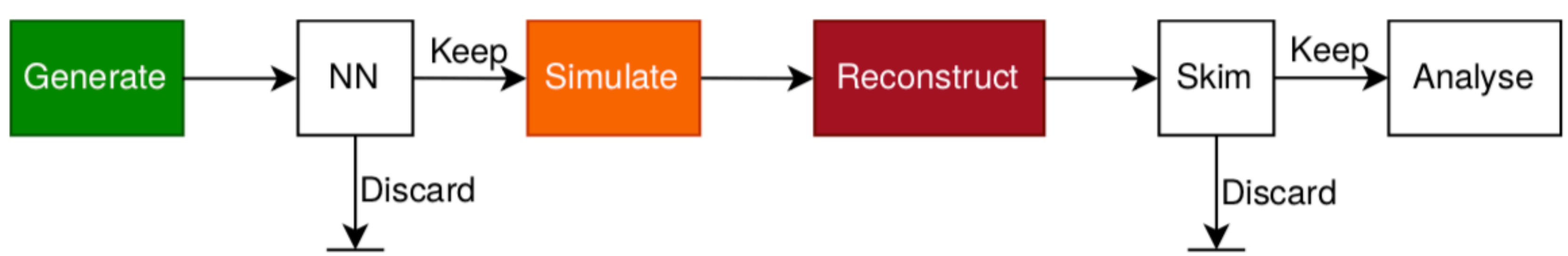
Figure: The Belle II PXD comprised of 40 modules, cylindrically arranged in two layers.



smartBKG: Selective Background Monte Carlo Generation



- Simulation of particle collisions computationally expensive
- Most simulations discarded during data-cleaning (*skim*)
- Idea: Can we figure out whether a collision is uninteresting at an early stage?
  - **Skip expensive steps**
  - Graph is natural representation of decay → **Graph Neural Networks**



LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Karlsruher Institut für Technologie

smartBKG: Dresden deep learning hackathon success

Entered as team for one week intensive work with machine learning mentor

- Dr. James Kahn (KIT, Prof. Forian Bernlochner)
- Kilian Lieret (LMU, Prof. Thomas Kuhr)
- Andreas Lindner (LMU, Prof. Thomas Kuhr)

Winners of **most creative team** for creation of modified graph convolutions

Original GCN:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}\right)$$

Modified GCN:

$$H^{(l+1)} = \sigma\left(H^{l}\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}} W_1^{(l)}\right)^T W_2^{(l)}\right)$$

Intuition: **custom weights for each node, considering neighbors**

Adapted from: Thomas N. Kipf, Max Welling, *Semi-Supervised Classification with Graph Convolutional Networks* (ICLR 2017)

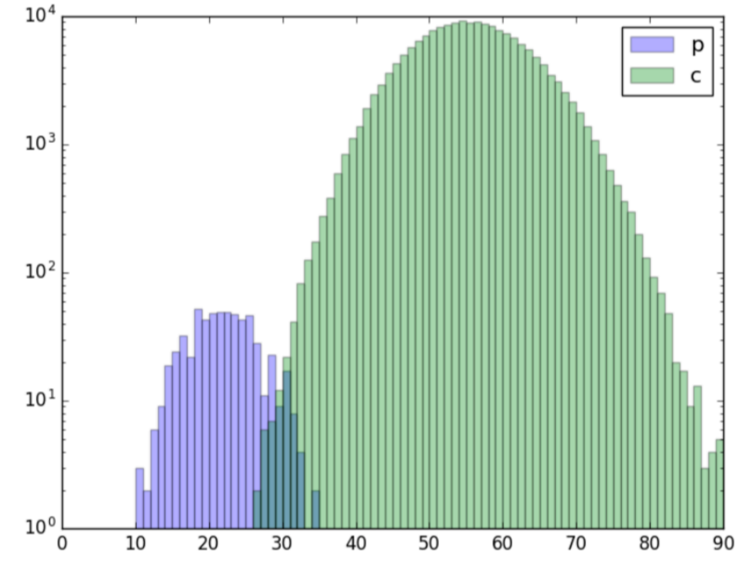
Anomaly Detection

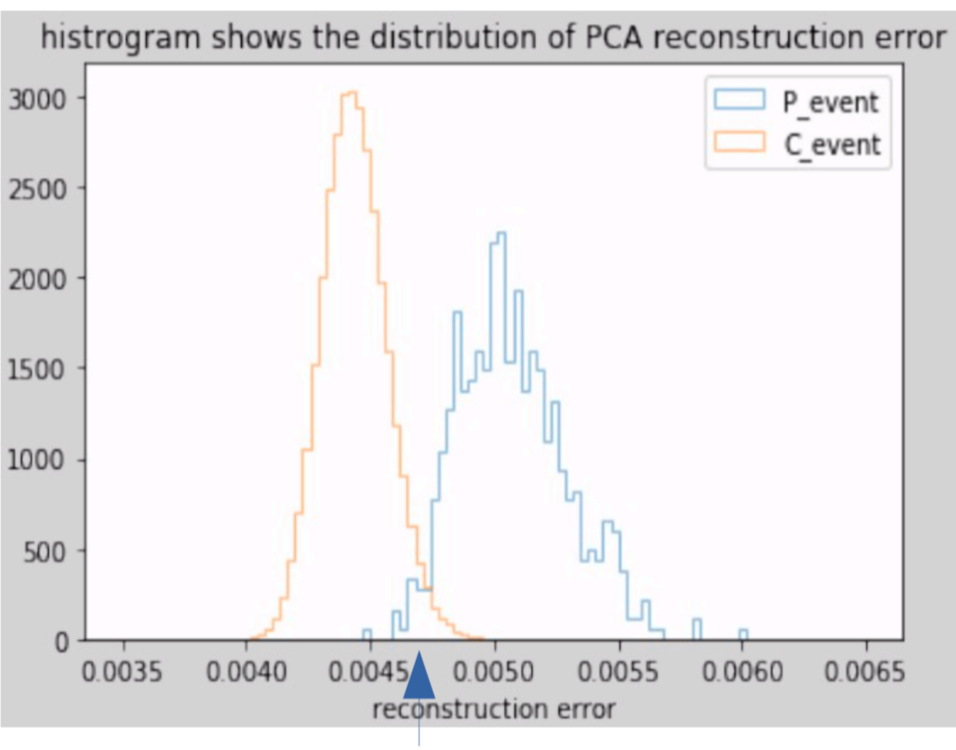
Projekts 3: Anomaly detection for HIC events selection

1) Unsupervised learning : **Principle Components Analysis** (PCA) and **Autoencoder** (AE) here can help selecting anomaly events

2) **Principle Components** (PC) from PCA or **Latent Space** from AE encode more compact information about the data structure

3) **Reconstruction Error** (RE) can help <u>identifying the anomalous</u>



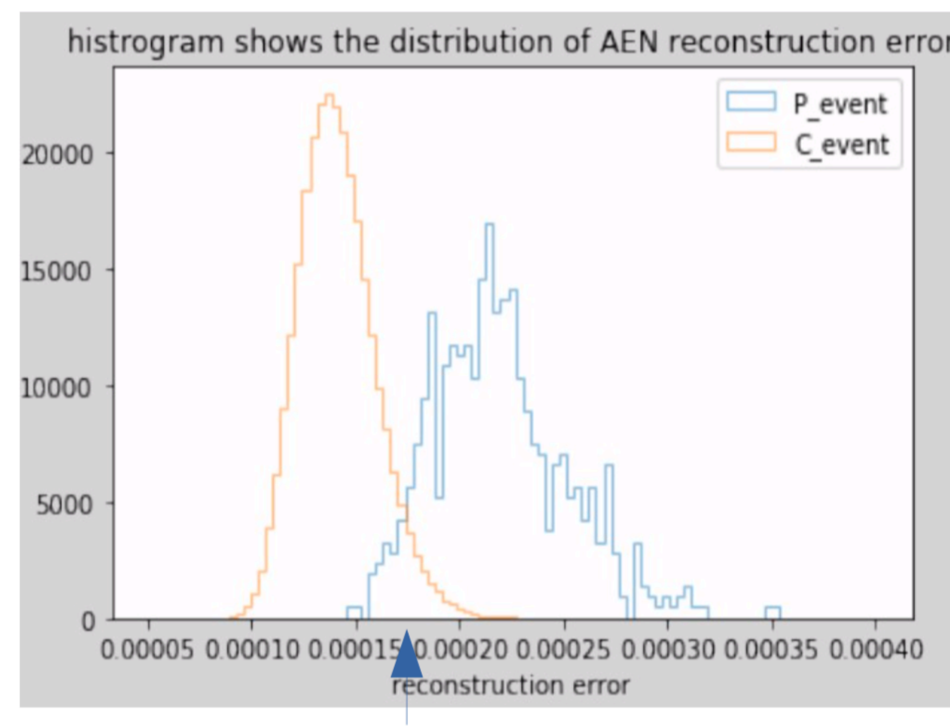


- PCA (left): 80 PCs from 20x20

  1.5% false alarm
  96% anomalies out

- Autoencoder(right): 2d latent

  4.65% false alarm
  93.83% anomalies out

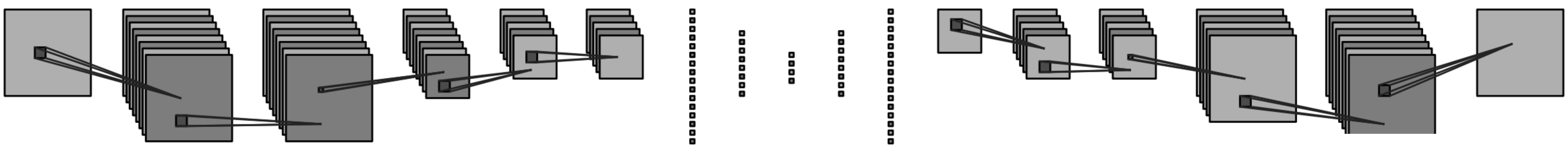


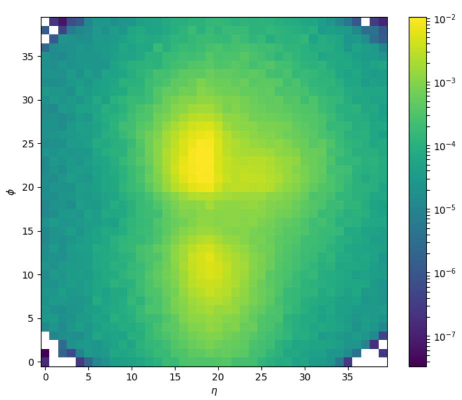
FIAS Frankfurt Institute for Advanced Studies

Model independent discovery

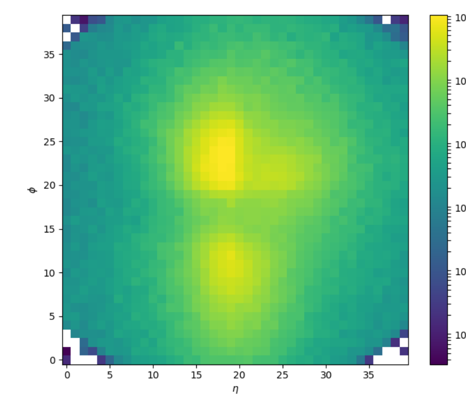
Search for new physics without specifying the model

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG





$$L_{\text{auto}} = \sum_{1600 \text{ pixels}} \left(k_T^{\text{norm,in}} - k_T^{\text{auto}} \right)^2$$

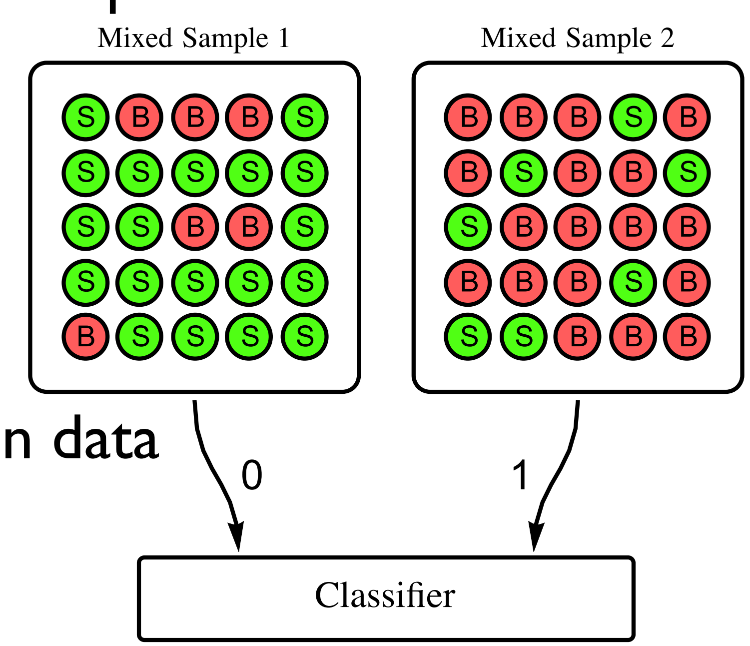


- Train on pure background sample
  - (Mass sidebands in data)
- New physics identified as anomaly
  - Tail of the loss function
- Complement dedicated searches

QCD or What?
T Heimel, GK, T Plehn, JM Thompson, 1808.08979
Searching for New Physics with Deep Autoencoders
M Farina, Y Nakai, David Shih, 1808.08992

Model independent bias

- **model independence means different things to different people**
- Learning New Physics from a Machine (1806.02350)
    - Shape differences - peak or tail - using high level variables
    - Trained on MC vs data
- Per-Jet Autoencoders (1808.08979 and 1808.08992)
    - Enhance mass peaks using anomalous substructure from low level inputs
    - Implicit model bias of Network
    - Trained on data
- CWoala Hunting (1902.02634):
    - Stronger discrimination by training a classifier between regions in data
    - So far only high level variables
- Per-Event Autoencoders (Trigger VAE) (1811.10276)
    - Trained on SM event cocktail, high level
    - Analyse produced stream

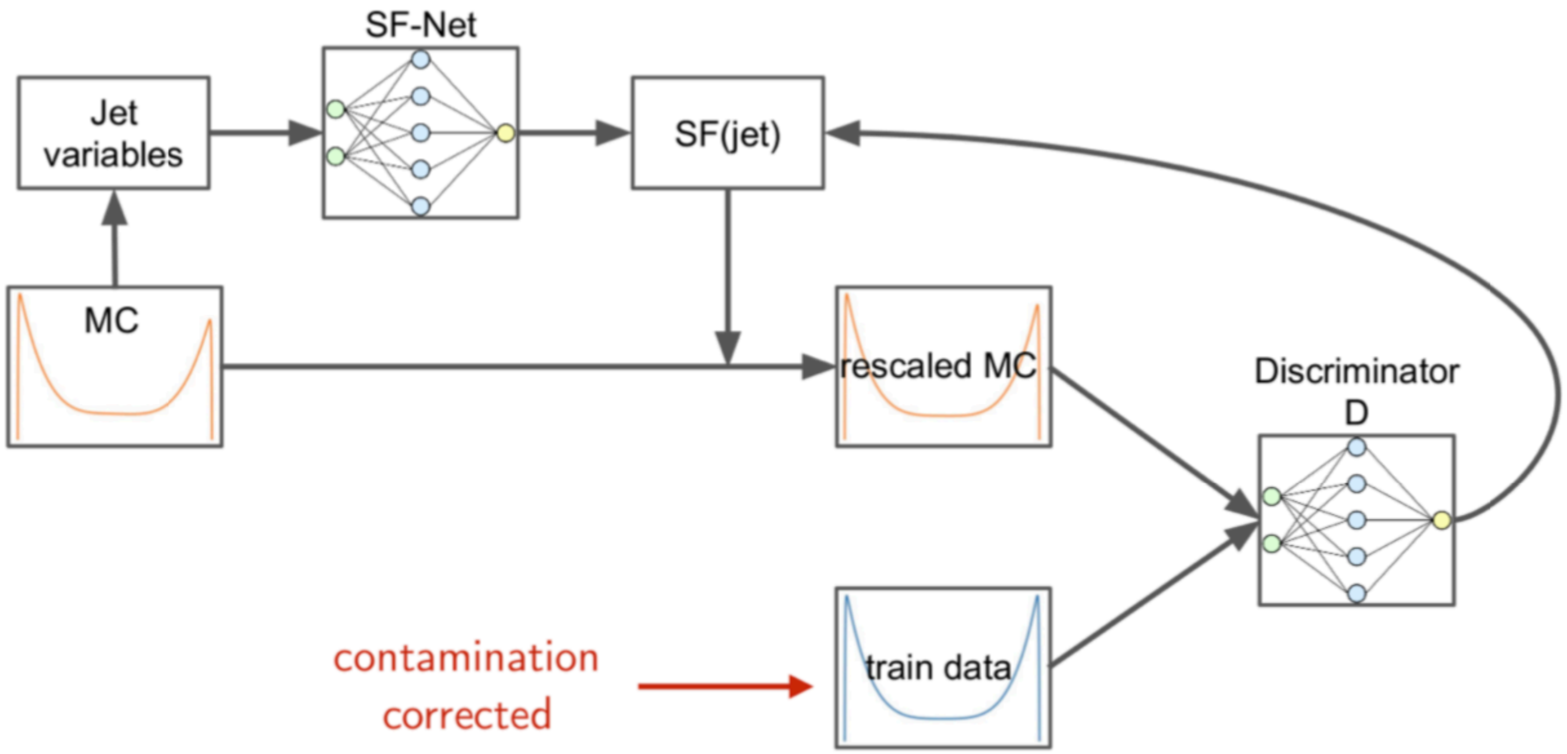


Robustness

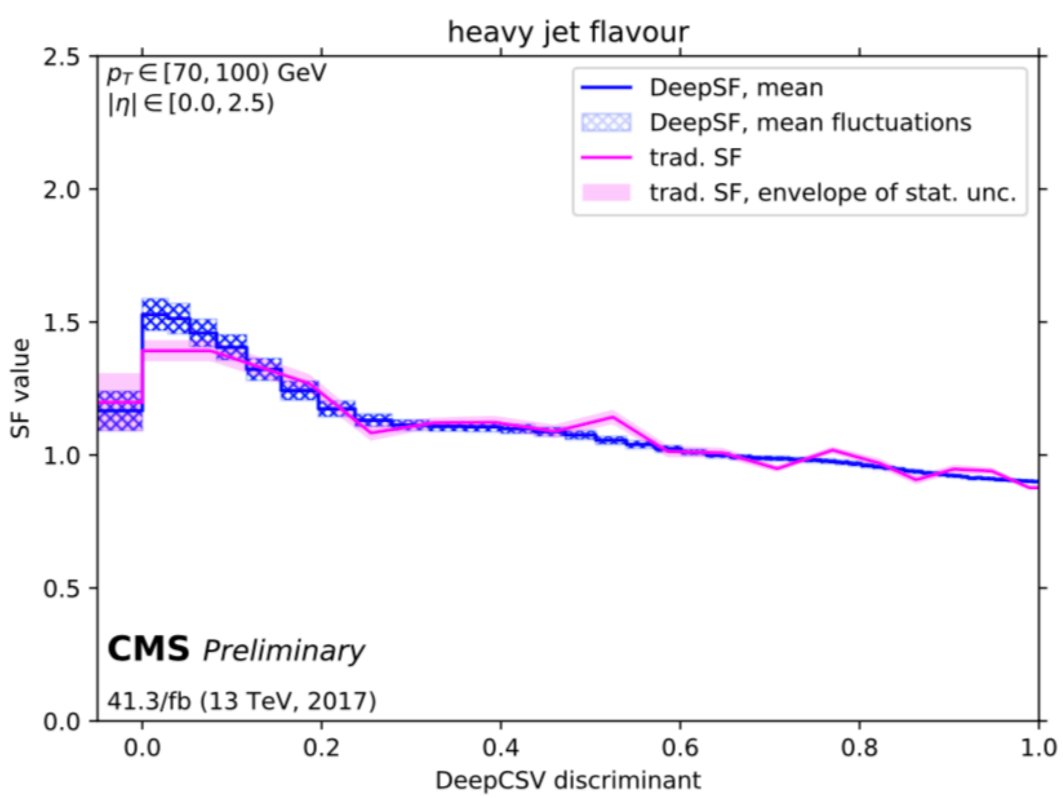
CMS: Deep Scale Factors

Erdmann, Fischer, ACAT 2019

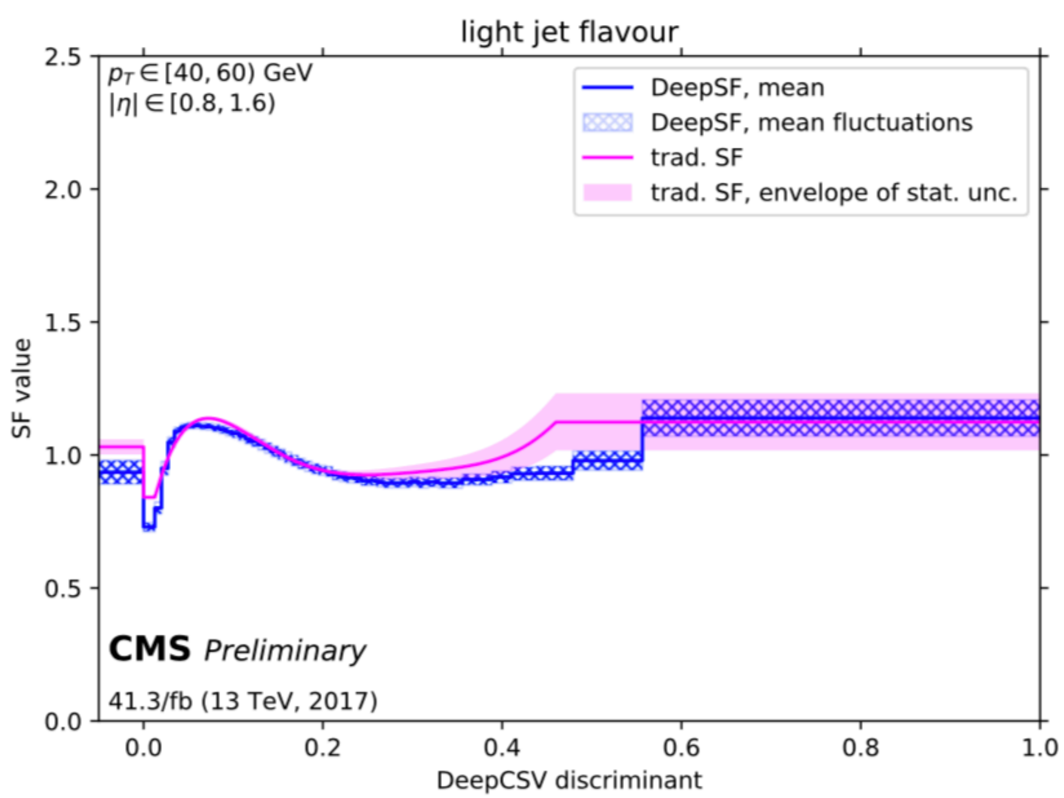
1. SF is calculated with jet variables from MC
2. MC is rescaled with SF
3. Discriminator (adversarial) discr. between data & rescaled MC
4. SF are recalculated with adversarial feedback



b jets

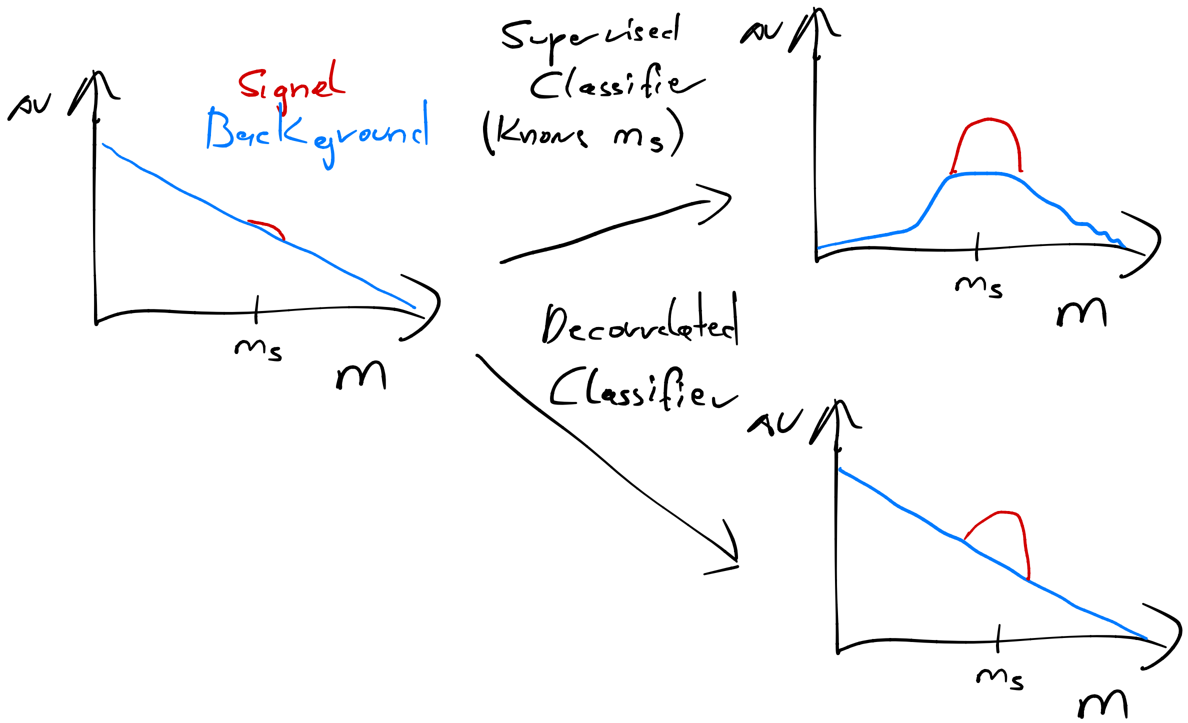


light jets



Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Decorrelation



$$x_{jk} = |X_j - X_k|$$ **Distances of all examples in batch for classifier output**

$$y_{jk} = |Y_j - Y_k|$$ **... for variable to decorrelate**

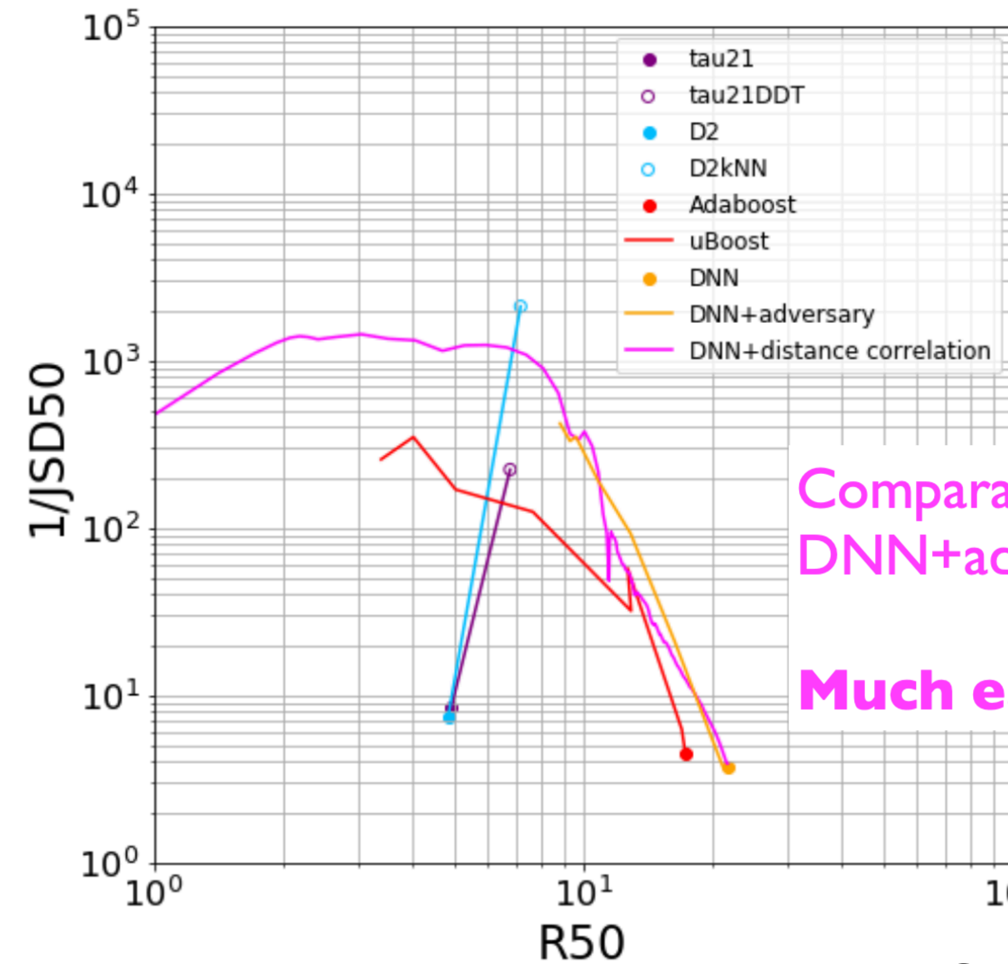
$$\hat{x}_{jk} = x_{jk} - \overline{x}_{j\cdot} - \overline{x}_{\cdot k} + \overline{x}_{\cdot\cdot}$$

$$\hat{y}_{jk} = y_{jk} - \overline{y}_{j\cdot} - \overline{y}_{\cdot k} + \overline{y}_{\cdot\cdot}$$

Center distributions

$$\mathrm{dCov}^2 = \frac{1}{n}\sum_j \sum_k \hat{x}_{jk}\hat{y}_{jk}$$ **And calculate average product per batch**

- Build robust classifiers by decorrelating output against another variable
- Strongest approach: adversarial training is difficult to tune and unstable
- Replace with regularizer term: distance correlation (**DisCo**)



Comparable performance to DNN+adversary.

Much easier to train.

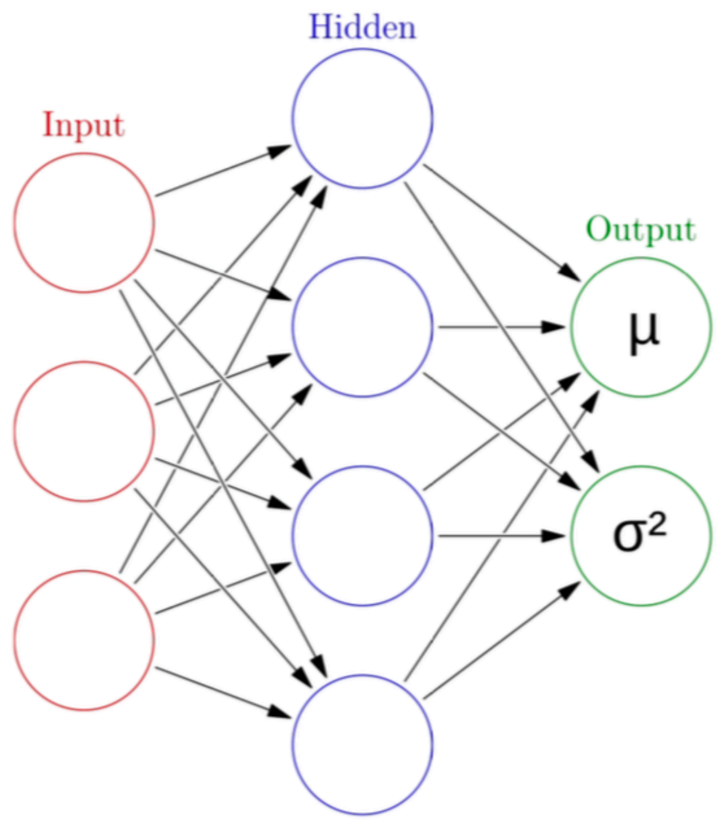
Work with David Shih, public soon

Recast of ATL-PHYS-PUB-2018-014
ATLAS - W boson tagging

Auger: Uncertainty Estimation

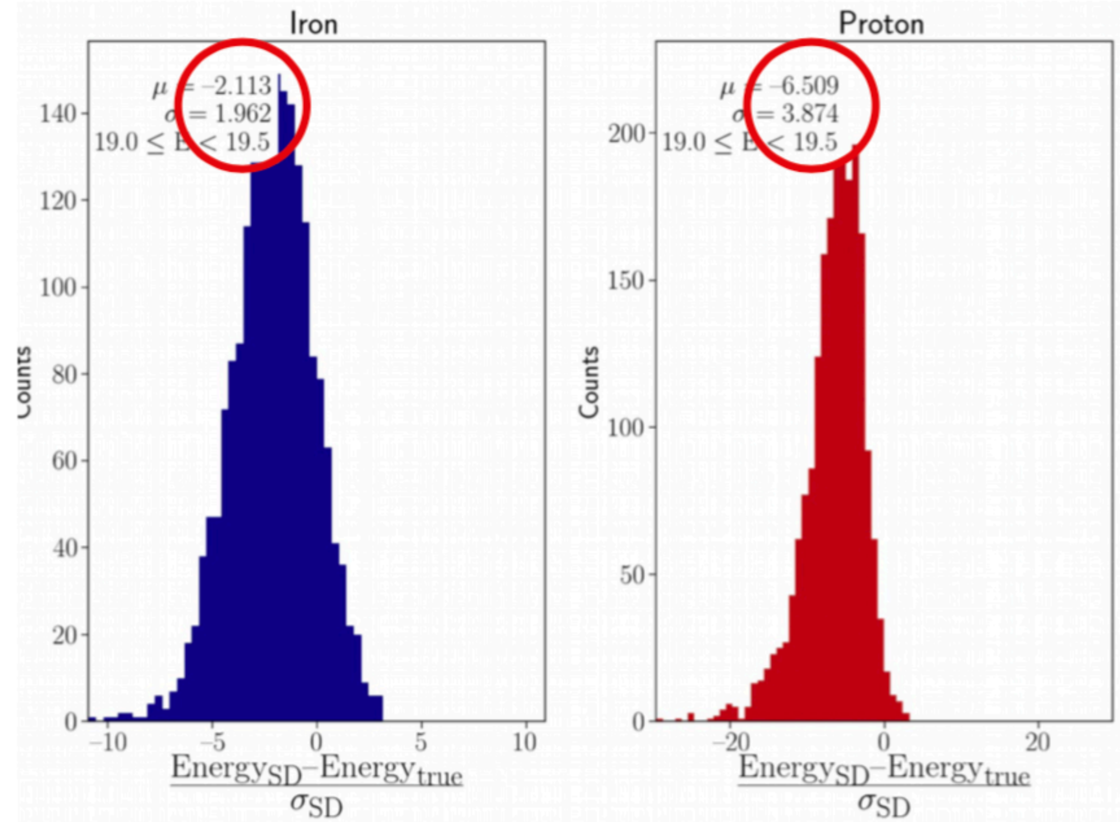
Straub bachelor thesis, Glombitza

- Allow for estimation of DNN uncertainties
- Assume normal distributed errors
  predict mean and variance
- Extend loss by taking variance in to account
- Average over DNN ensemble using Gaussian Mixtures models

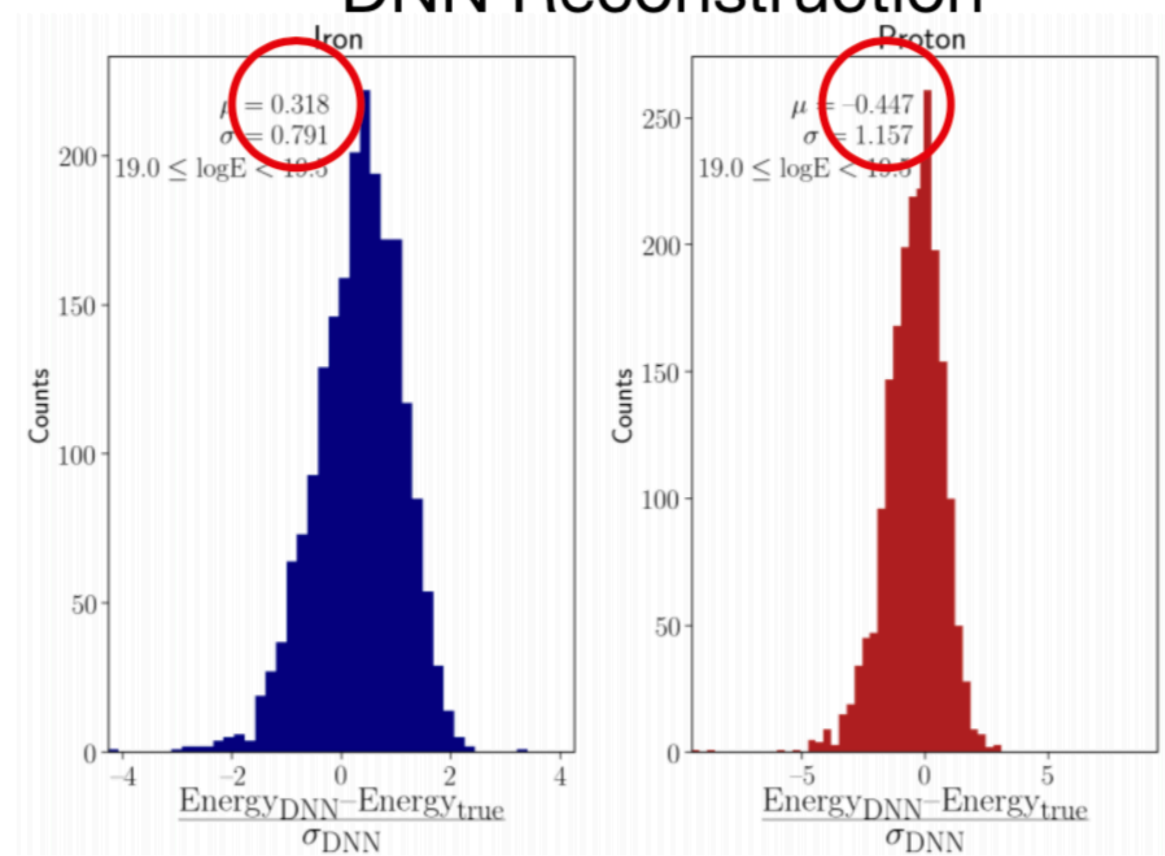


$$-\log p_\theta(y|x) = \frac{\log\sigma_\theta^2(\mathrm{x})}{2} + \frac{\left(y-\mu_\theta(x)\right)^2}{2\sigma_\theta^2(x)}$$

Standard Reconstruction

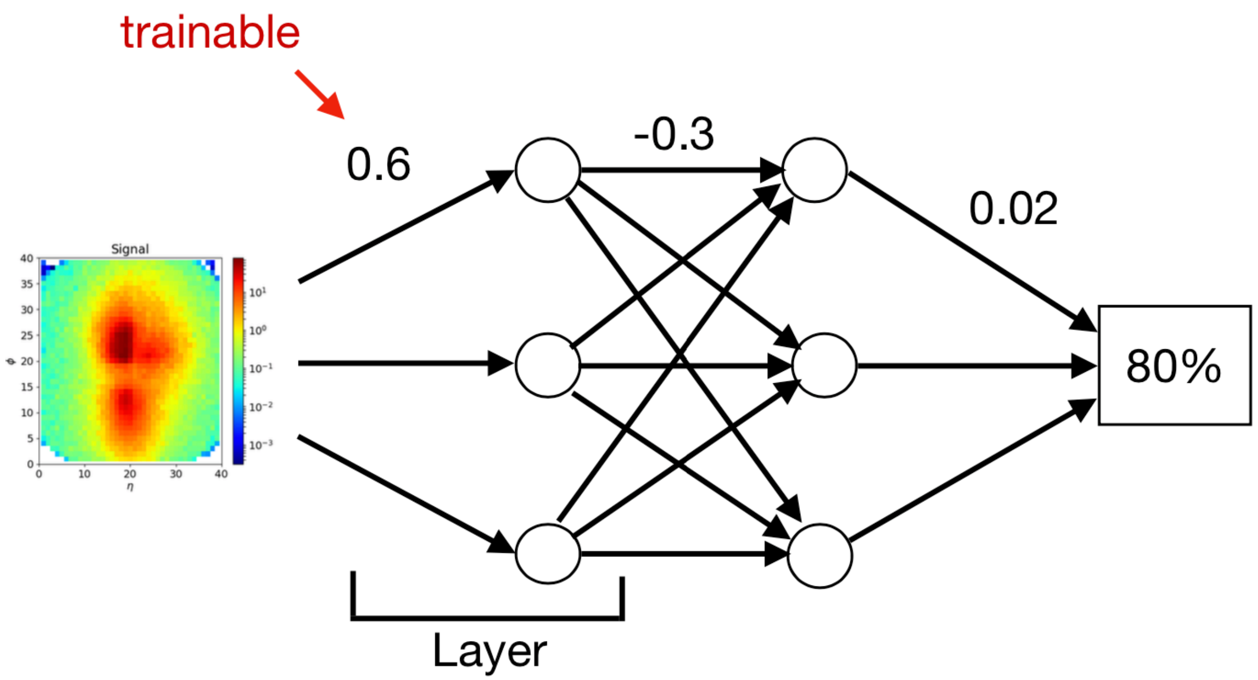


DNN Reconstruction



Quantifying Uncertainty

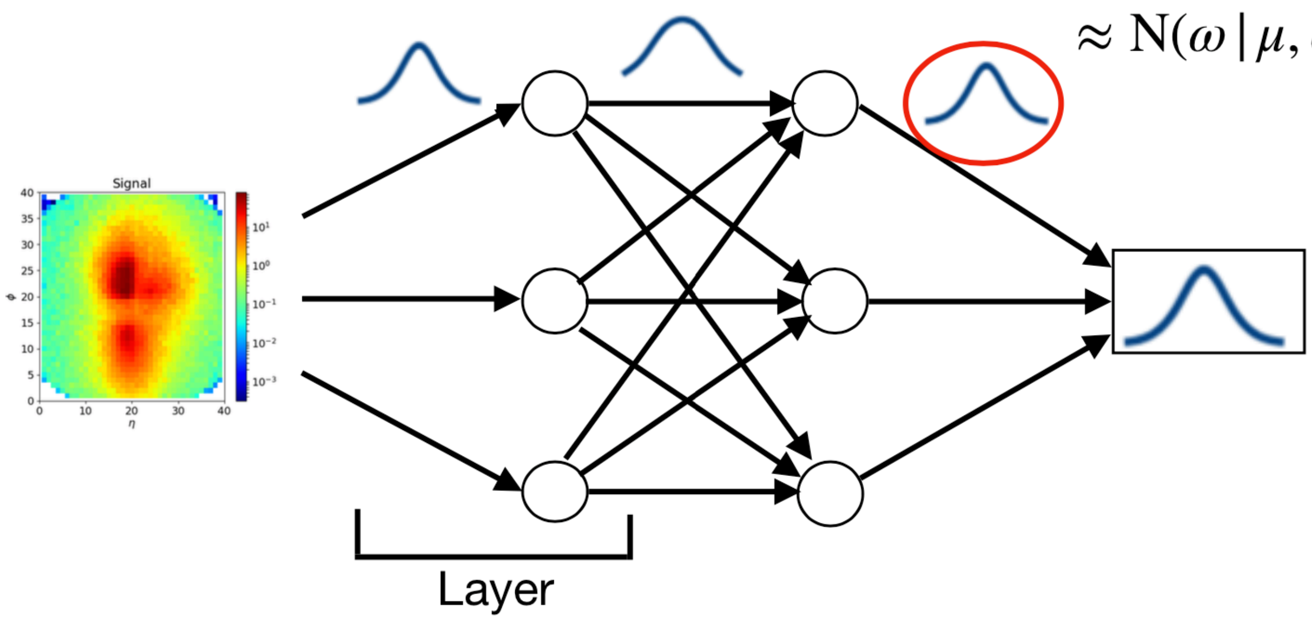
Standard (Deterministic) Neural Net



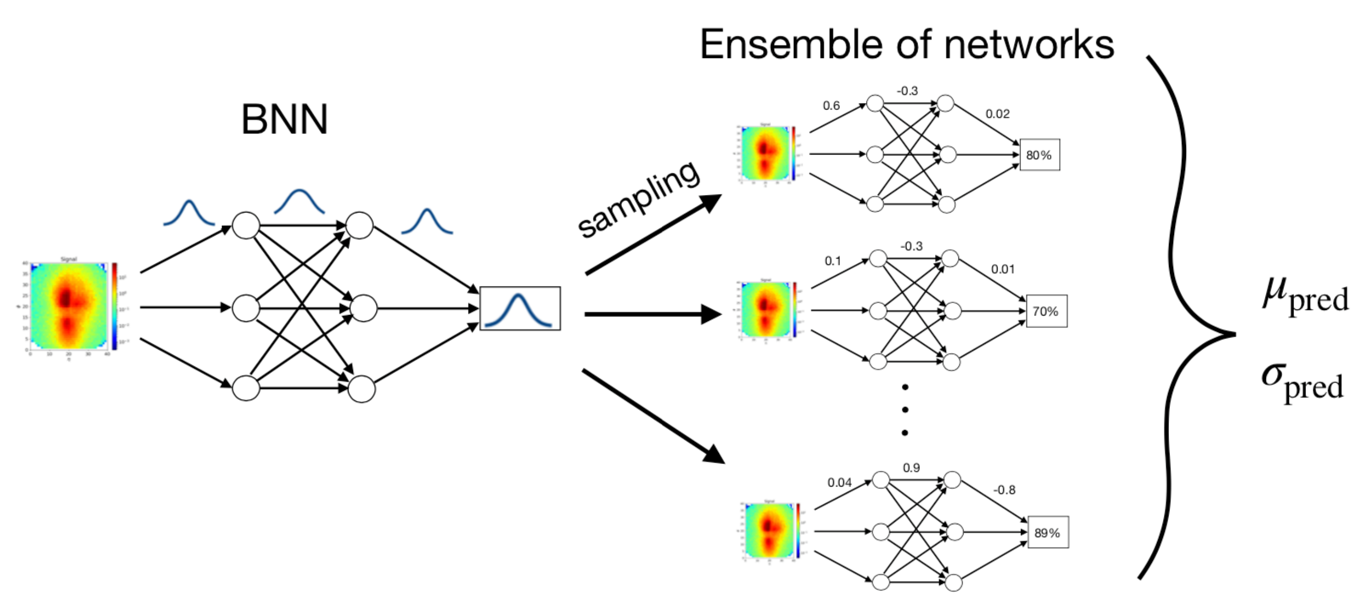
Bayesian Neural Net

Bayes theorem prior

$$p(\omega \,|\, C) \propto p(C \,|\, \omega)\; p(\omega)$$
$$\approx \mathrm{N}(\omega \,|\, \mu, \sigma)$$

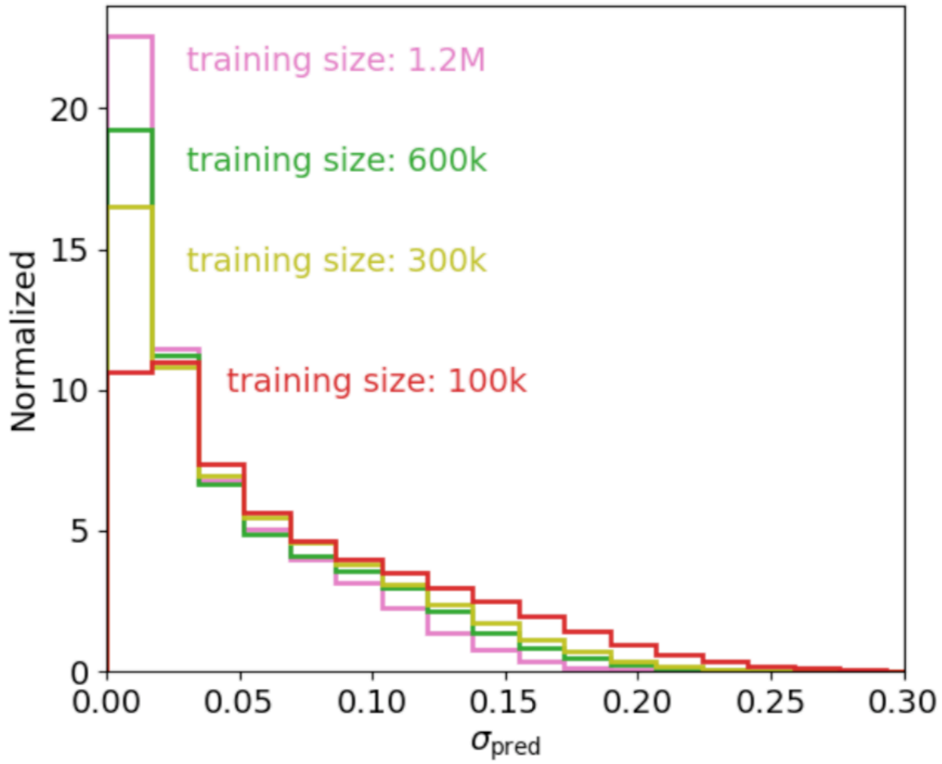


- Provide per-prediction uncertainty on neural network output: Bayesian networks
- Weights replaced by probability distributions
- Prediction vis MC sampling

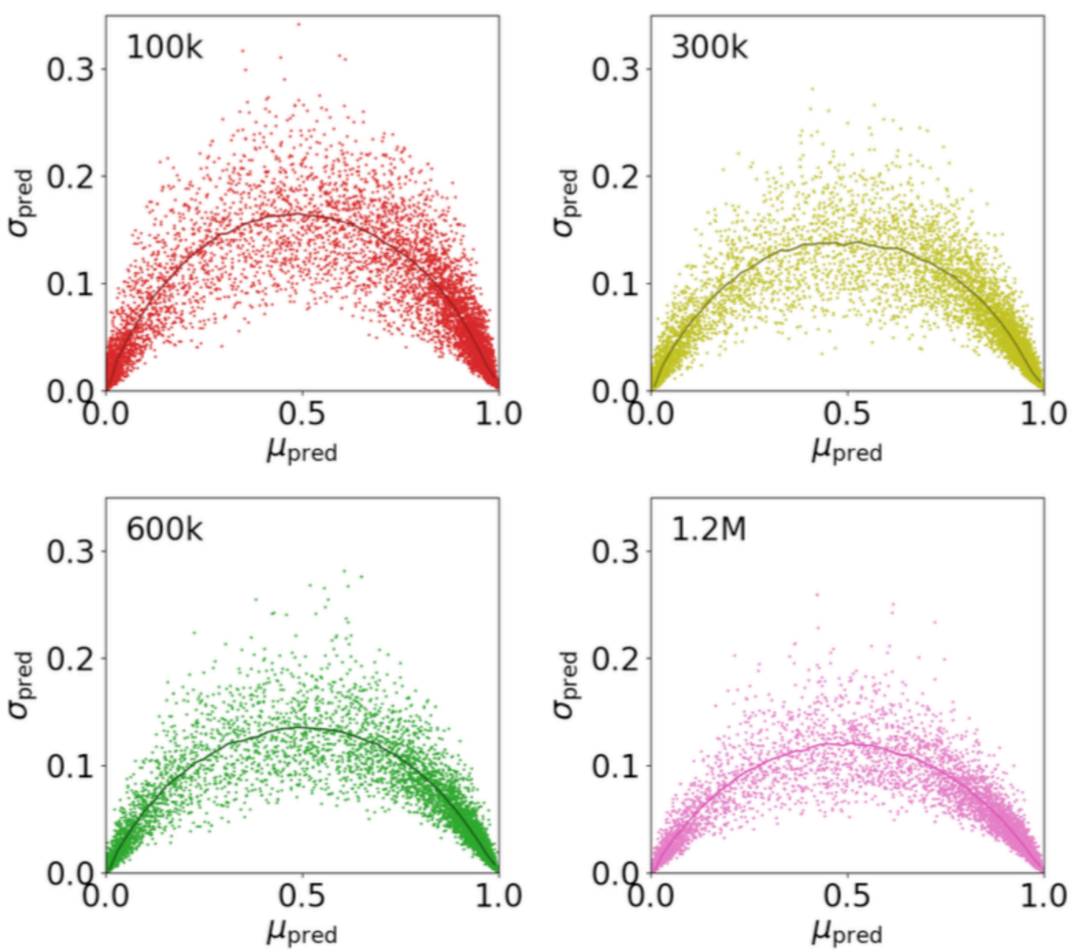


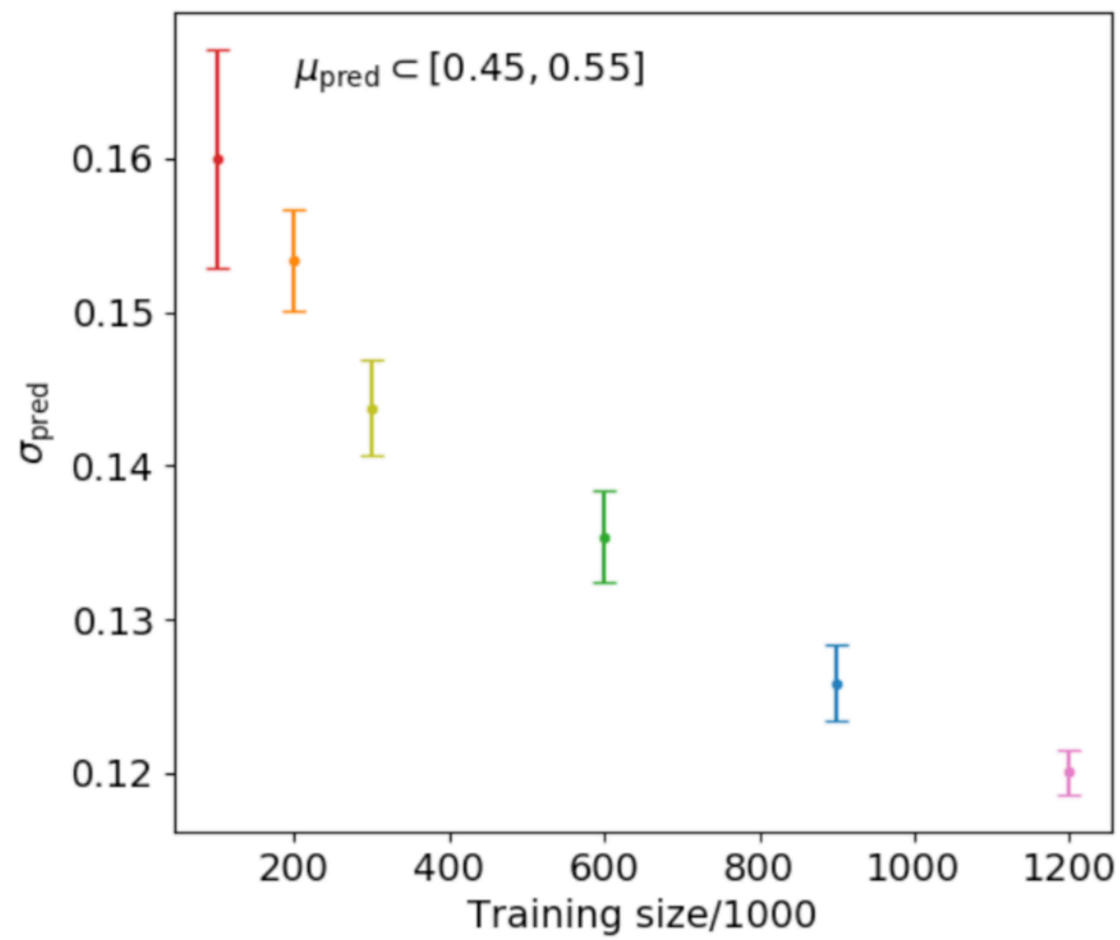
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Statistical Uncertainty



BNN captures effect of finite training data





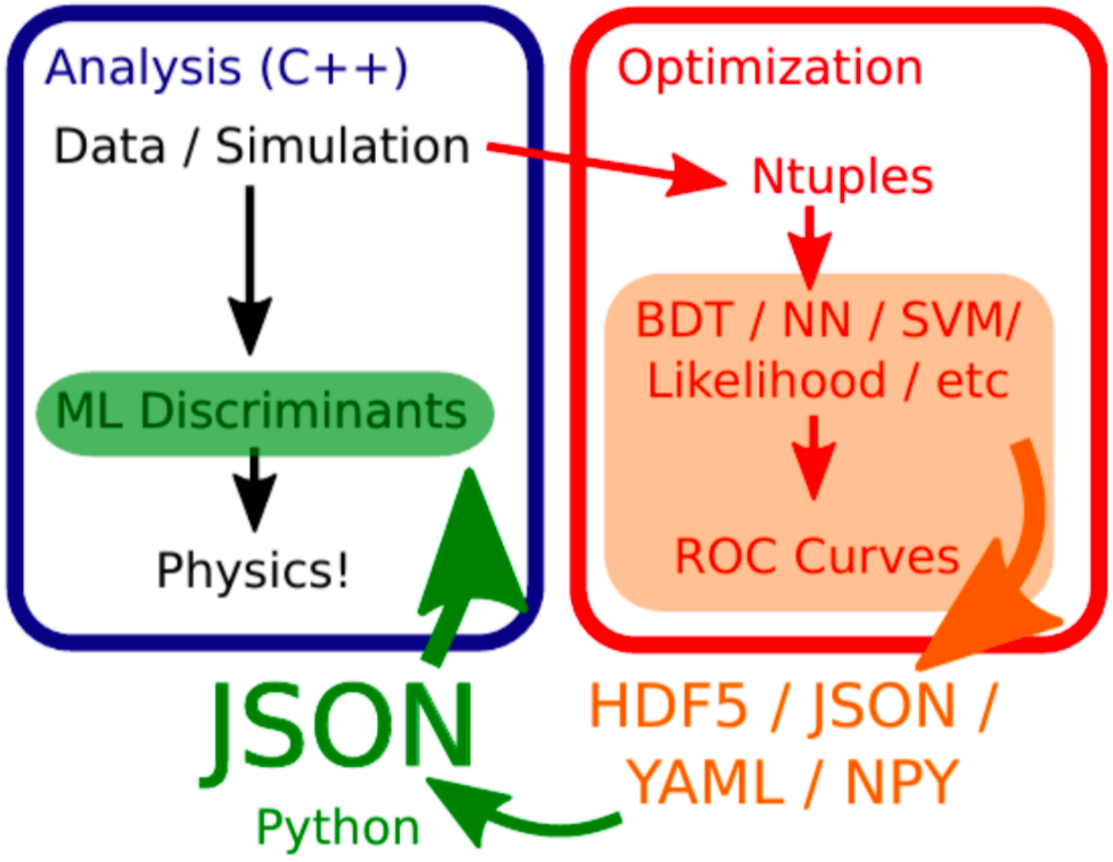
Classification sigmoid correlates mean and standard deviation

Next: Application to top mass measurement

Fast Inference and Tools

lwtnn

- Tool developed with contribution from our group (J. Smith)
- Idea: training and testing can be separated:
  - Often: training in complex environment
  - Testing/application: trained network has to be applied to new data, i.e. in new analysis, for many systematics, on trigger level, etc.
  - Application should often run with limited CPU usage



- lwtnn (LightWeight Trained Neural Network):
  - Converts saved NNs to JSON format for several popular formats (Keras, Scikit Learn, etc.)
  - Reconstructs NN from JSON file
  - Run NN in fast/light-weight C++ code

    → reduces CPU time significantly if NNs have to be applied many times
  - Available at: https://github.com/lwtnn/lwtnn

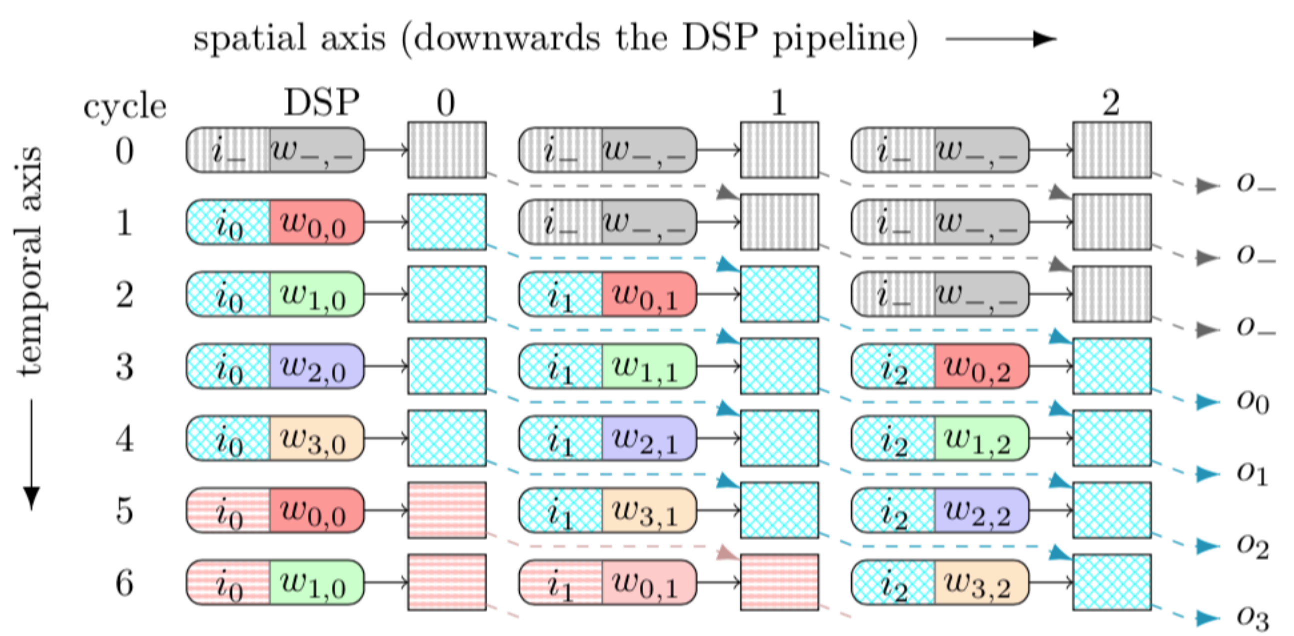
Reminder: Aim of the project in Mainz

- **Processing of detector data at extremely high rates**
  - Not possible to store data due to its size
  - Usage of GPUs not possible due to their too high latency
  - Data has to be processed and filtered locally, maybe directly at the corresponding sensors
- **Solution: deep neural networks** as replacement for iterative algorithms, that can be **efficiently evaluated on FPGAs**
- **Test environment: ATLAS L1 Trigger** (40 MHz rate)

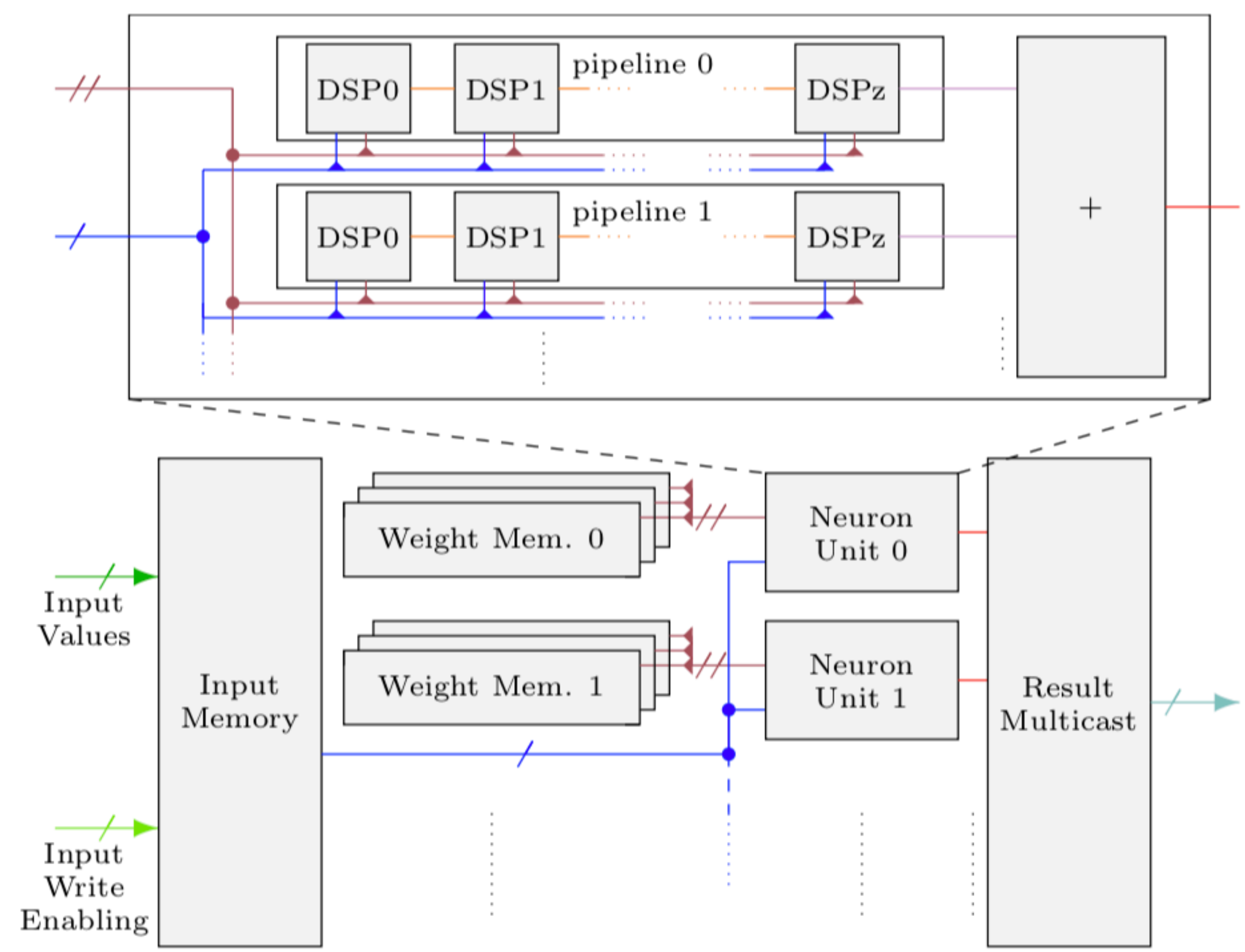
Current status

- **First implementation of Dense, 2D-Convolution and MaxPooling layer done**
  - Paper on implementation details as well as performance evaluation accepted for publication by JINST

arXiv:1903.10201

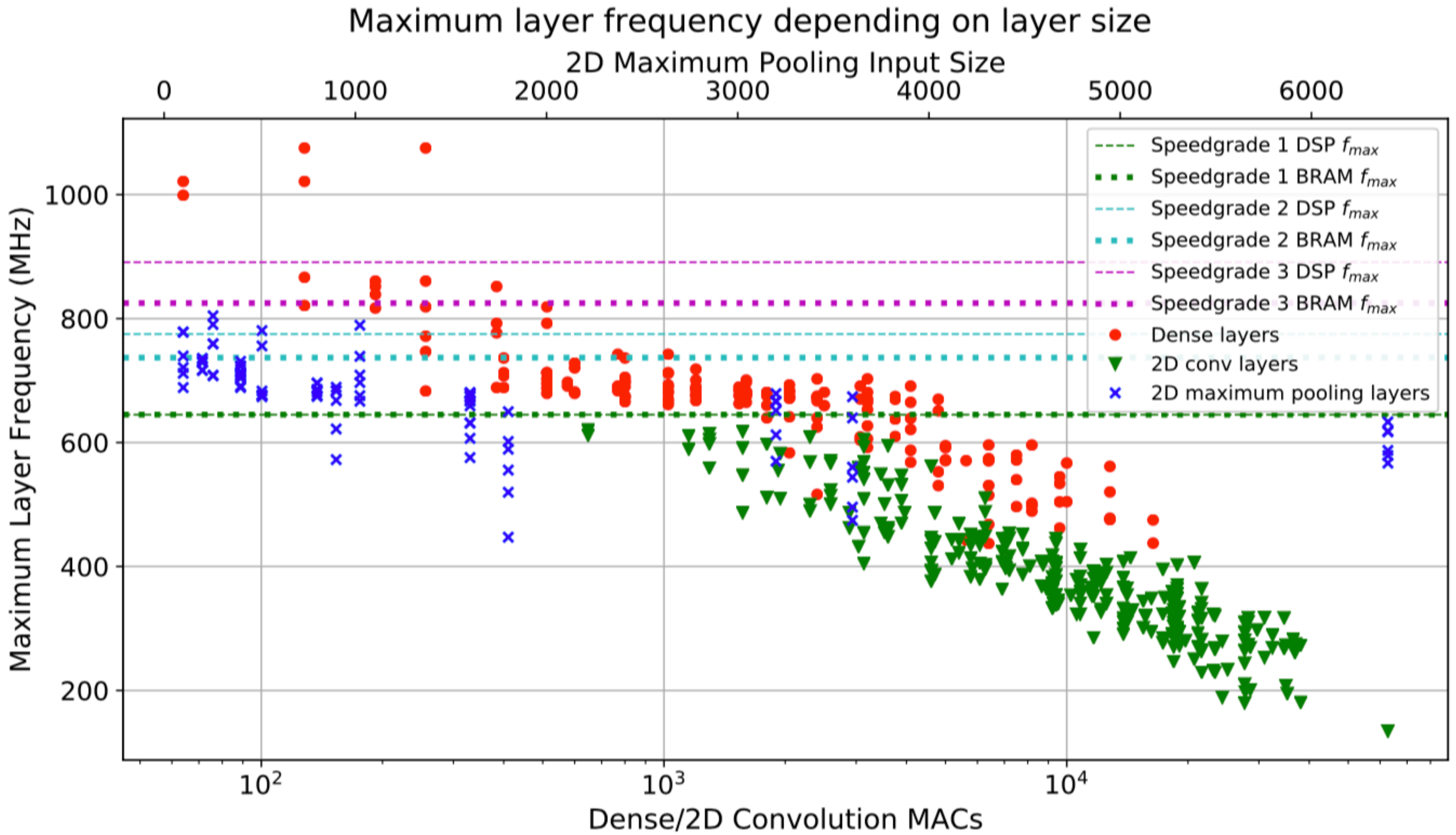


(a) Data flow schematic for the fully-connected layer. The color (pattern) coding is as follows: gray (vertical lines) refers to no data set available/idling, while cyan (crosshatch) and pink (horizontal lines) refer to data of a first and second data set, respectively. The inputs of neighboring pipeline stages are updated in subsequent cycles and only once per data cycle, while the weight sequence for each individual DSP is repeated during each data cycle.



Current status

- **First implementation of Dense, 2D-Convolution and MaxPooling layer done**
  - Paper on implementation details as well as performance evaluation accepted for publication by JINST

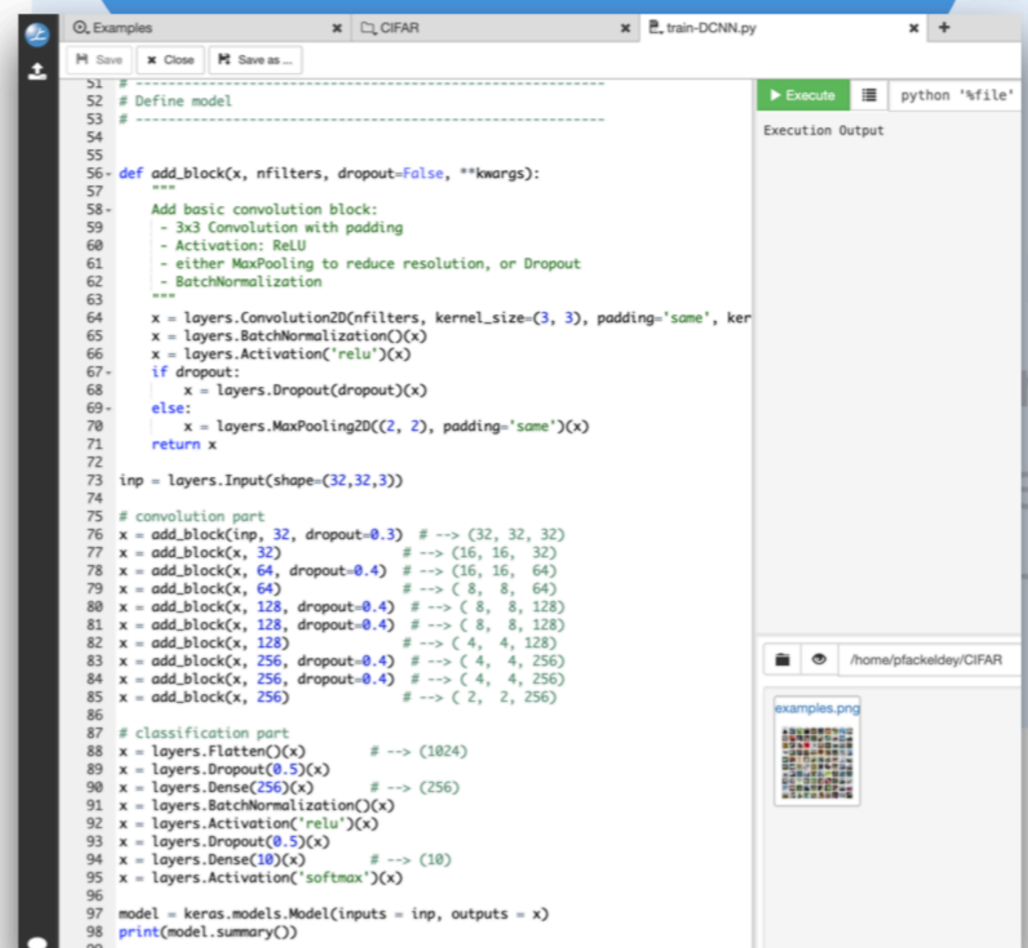


JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Research Projects: VISPA

VISPA

Use via webbrowser



- 20 NVIDIA GTX 1080
- Storage, NFS, Network
- ~200 CPU cores

Soon upgrade: +9 GPU RTX5000/6000

Closing

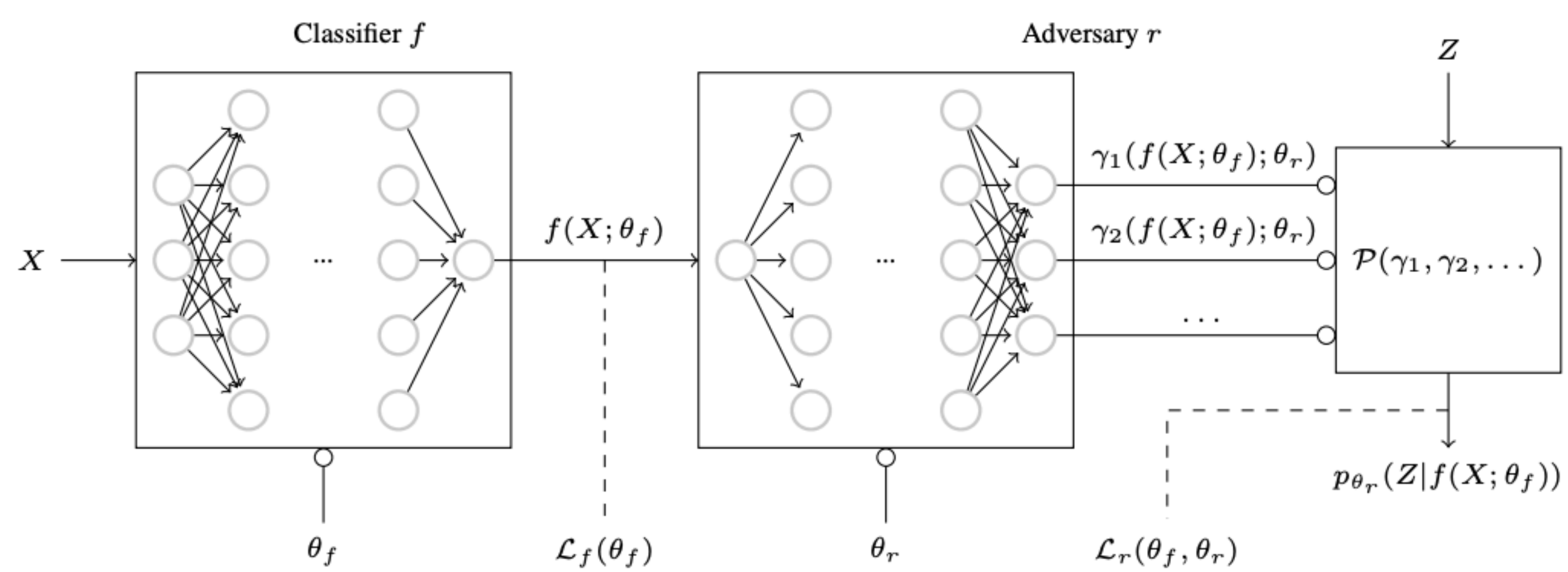
Conclusions

- Impressive and wide-ranging number of machine learning studies in ErUM data pilot project
- Main topic still classification
  - Exploration of different architectures
- Useful progress in new directions: simulation, anomalies, robustness, …
- Fast inference crucial for future applications (online and offline)
- Many groups also active in educating students in these new methods
  - Regular curriculum and special events/schools
- How to benefit from potential synergies?

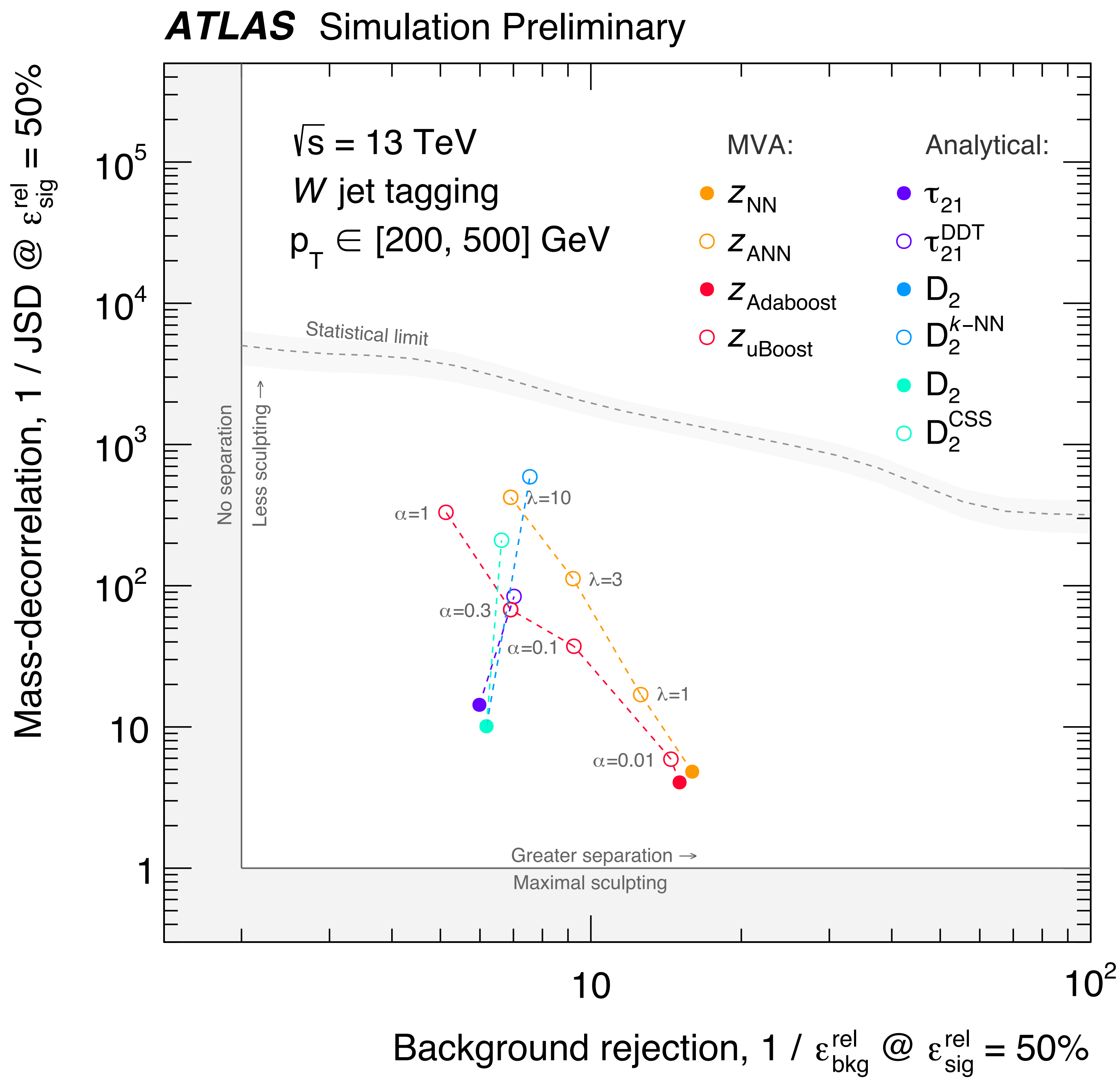
Bonus Slides

Approaches

- **Obscurity**:
    - Do not give mass [will be using this as stand-in for any variable we want to decor relate agains] as input
    - Simple, does not work
- **Data planing** (1709.10106, 1908.08959):
    - Reweight input distributions to be flat
    - Simple, limited power
- Designing Decorrelated Taggers - **DDT** (1603.00027):
    - Linearly transform output to be stable for one working point by subtracting for each bin
- Add **KL/JS divergence** to loss
    - Promising idea, but only works for one working point. Binning needed.
- Use complex **adversarial ML** (1611.01046, 1703.03507)
    - Powerful, hard to tune



Comparison

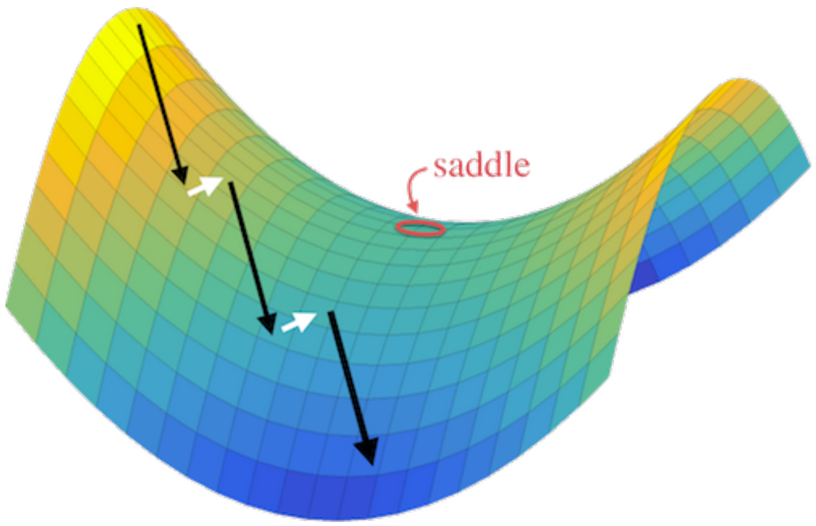


ATL-PHYS-PUB-2018-014

Problem

- Adversarial training is inherently unstable (hard to set up and sensitive to hyper parameter changes)
  - Looking for a saddle point

$$\min_{\theta_{\text{clf}}} \max_{\theta_{\text{adv}}} L_{\text{clf}}(y(\theta_{\text{clf}})) - \lambda L_{\text{adv}}(y(\theta_{\text{clf}}), m; \theta_{\text{adv}})$$



- Find a regulariser term that fulfils the same goal but allows simple training to convergence

$$\min_{\theta_{\text{clf}}} L_{\text{clf}}(y(\theta_{\text{clf}})) + \lambda C_{\text{reg}}(y(\theta_{\text{clf}}), m)$$

- Use distance correlation!

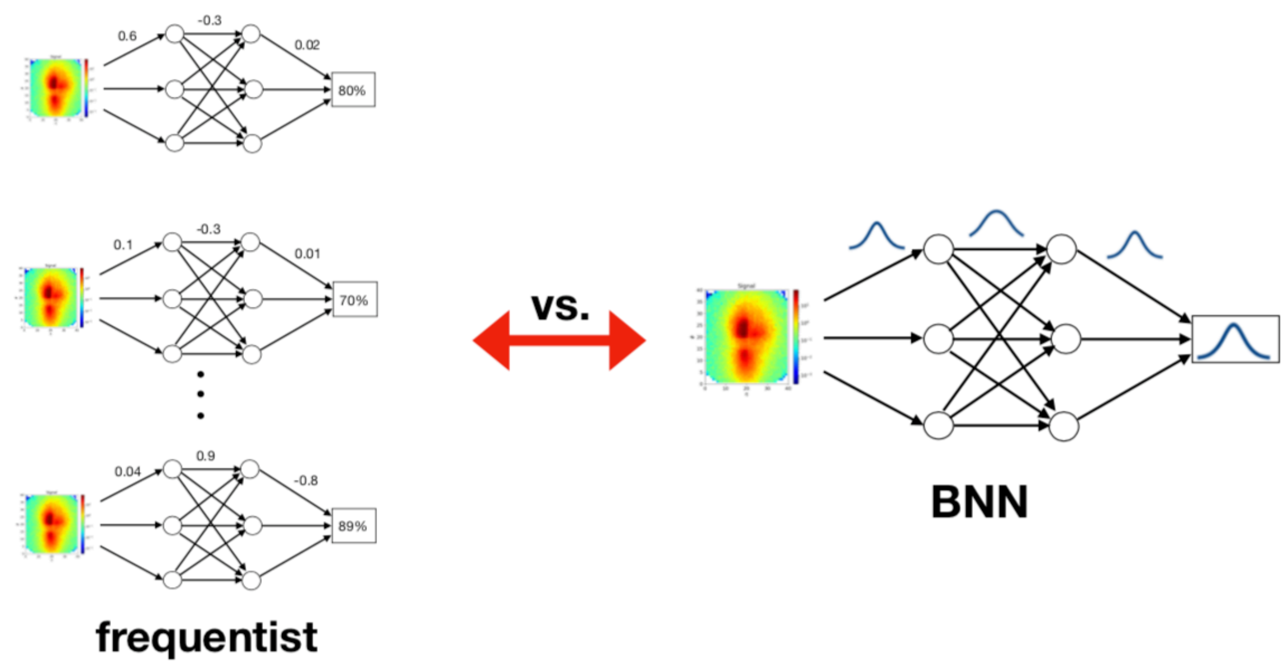
ParticleNet = Graphs

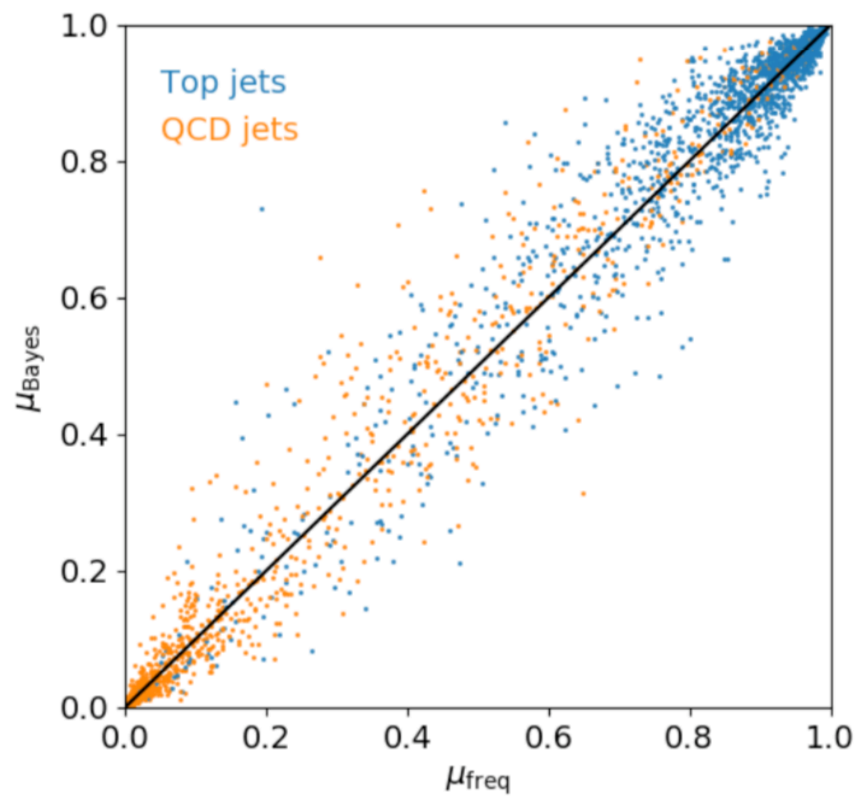
- Images are a convenient representation, but do not capture real structure of our measurements
- Alternative: Graphs
  - Vertex: Particle
  - Edge: Distance (for example in eta-phi space)
- Active development of graphs on CS side, but already HEP applications:
  - Particle Net (best performing top tagger in community study, based on EdgeConv)
  - Calorimeter Clustering (1902.07987)
  - Tracking (1810.06111)

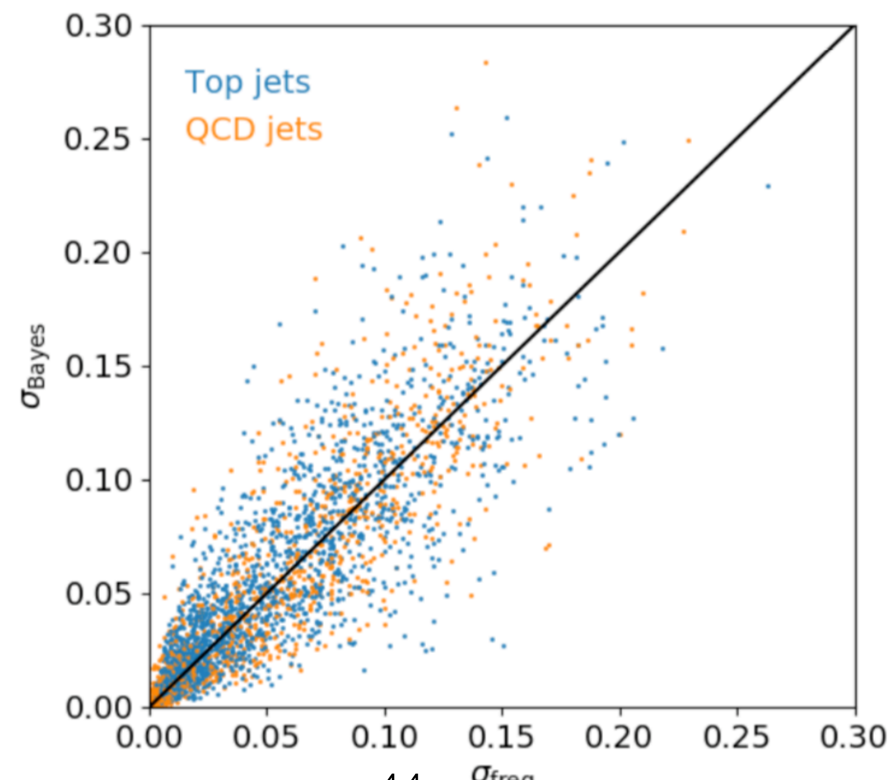
1902.08570

Validation

- Train N deterministic networks or statistically independent events
- Calculate mean and standard deviation of ensemble

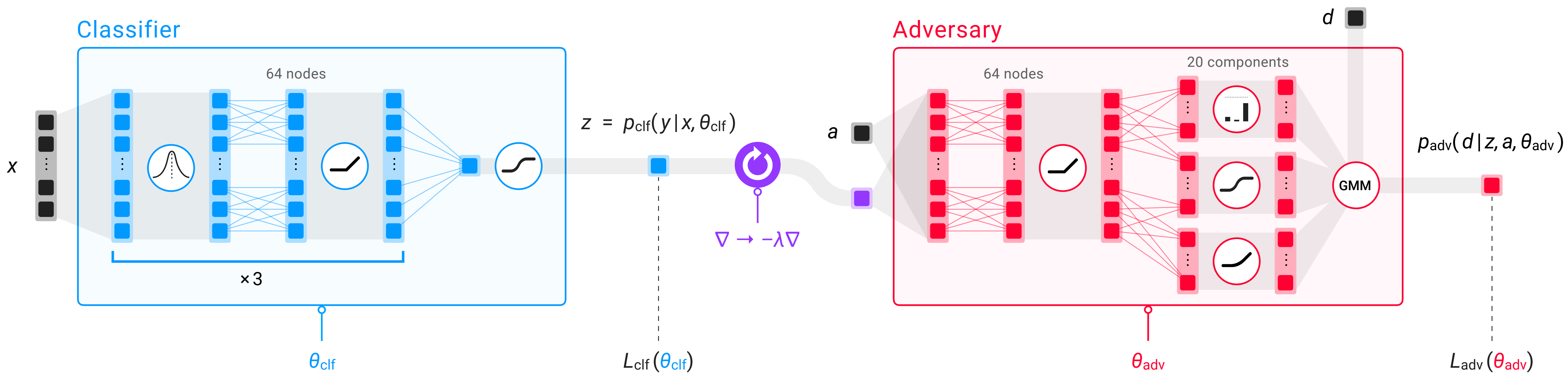






Result agrees with frequentist expectation

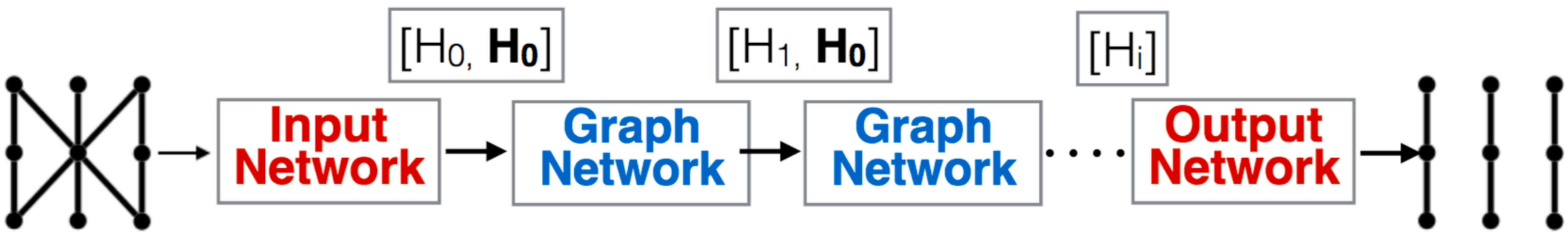
ATLAS implementation



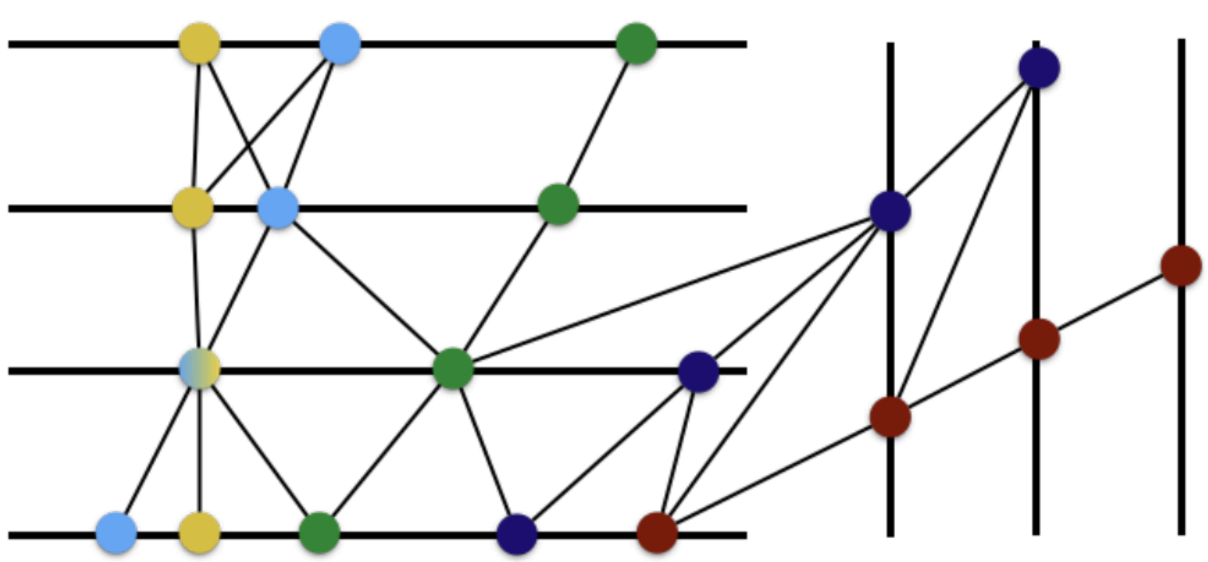
- Similar to learning to pivot, uses gradient reversal
- Input: high level substructure variables

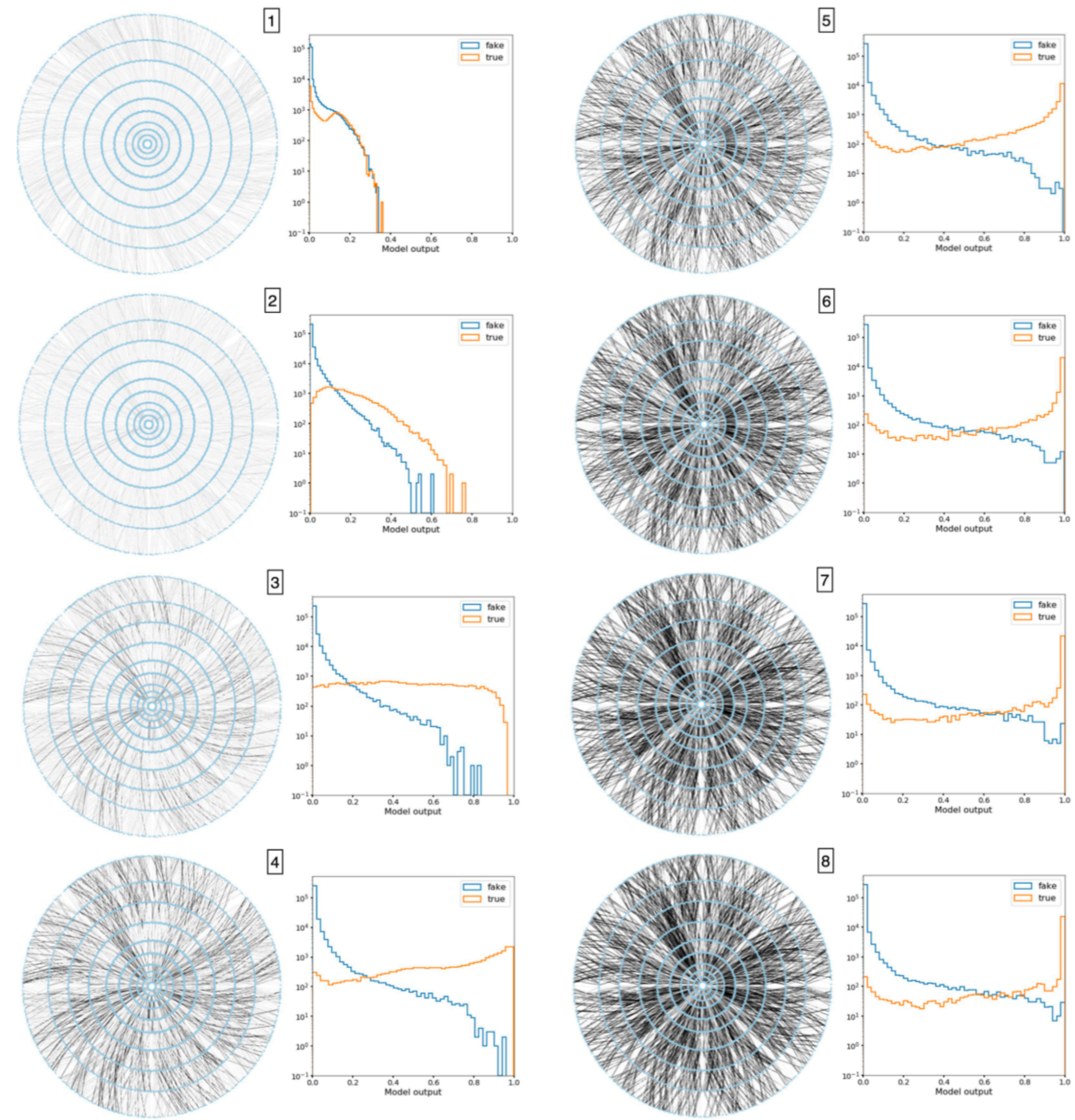
ATL-PHYS-PUB-2018-014

Tracking



- Big challenge: fast high multiplicity tracking
- Graph representation:
  - Hits = Nodes
  - Edges = Hits belong to same track

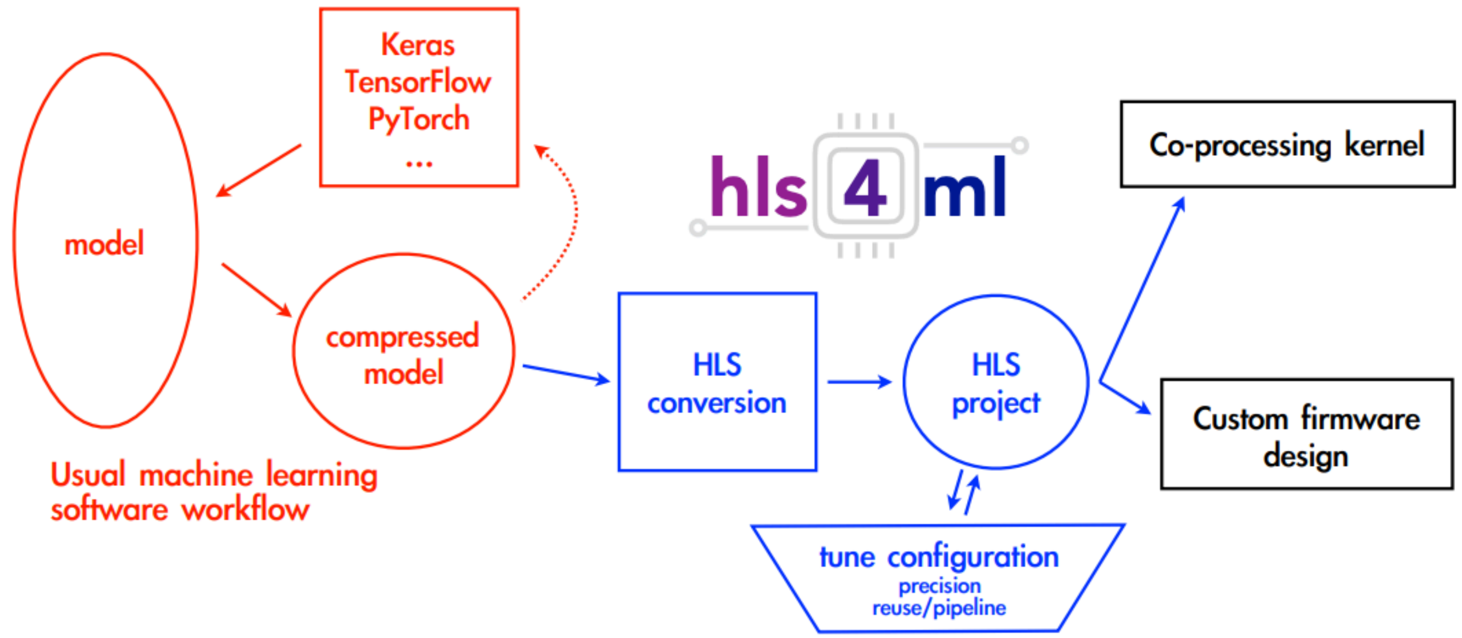


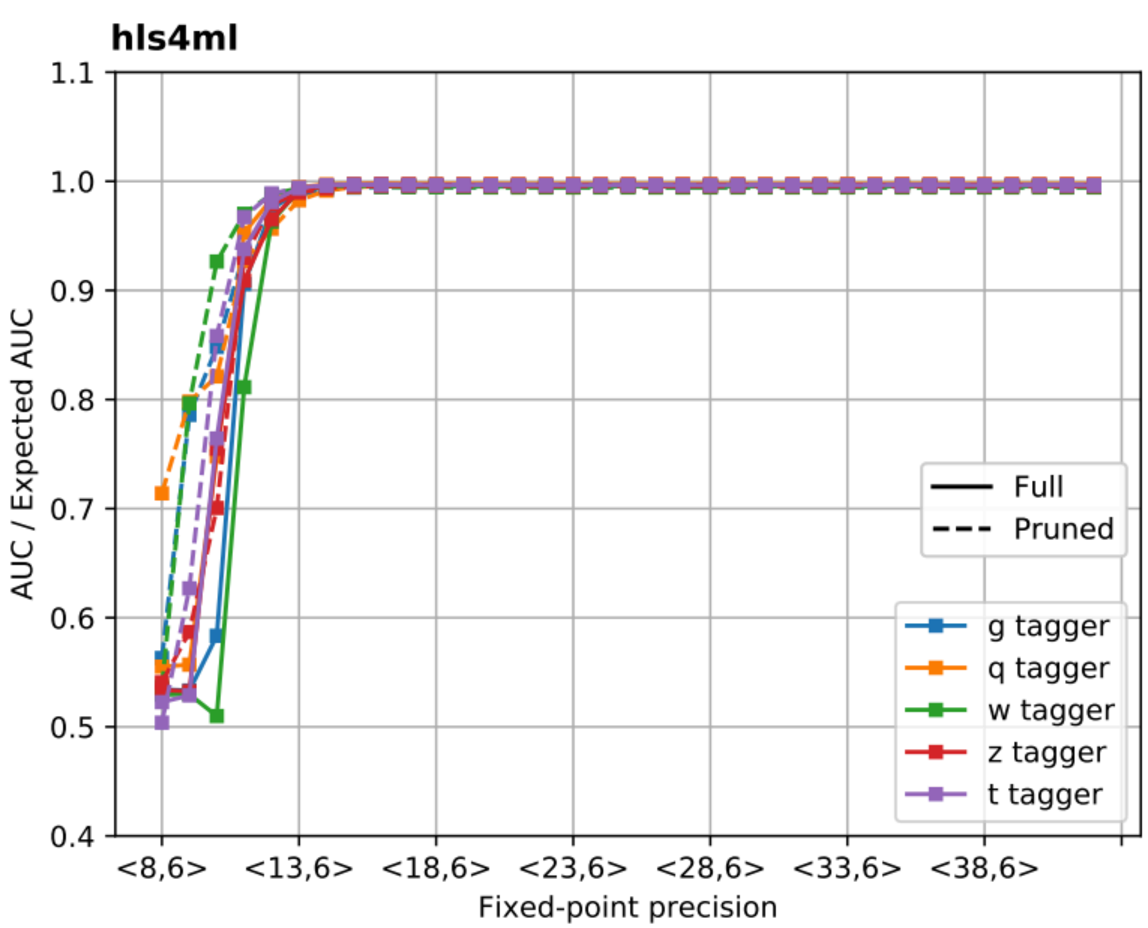


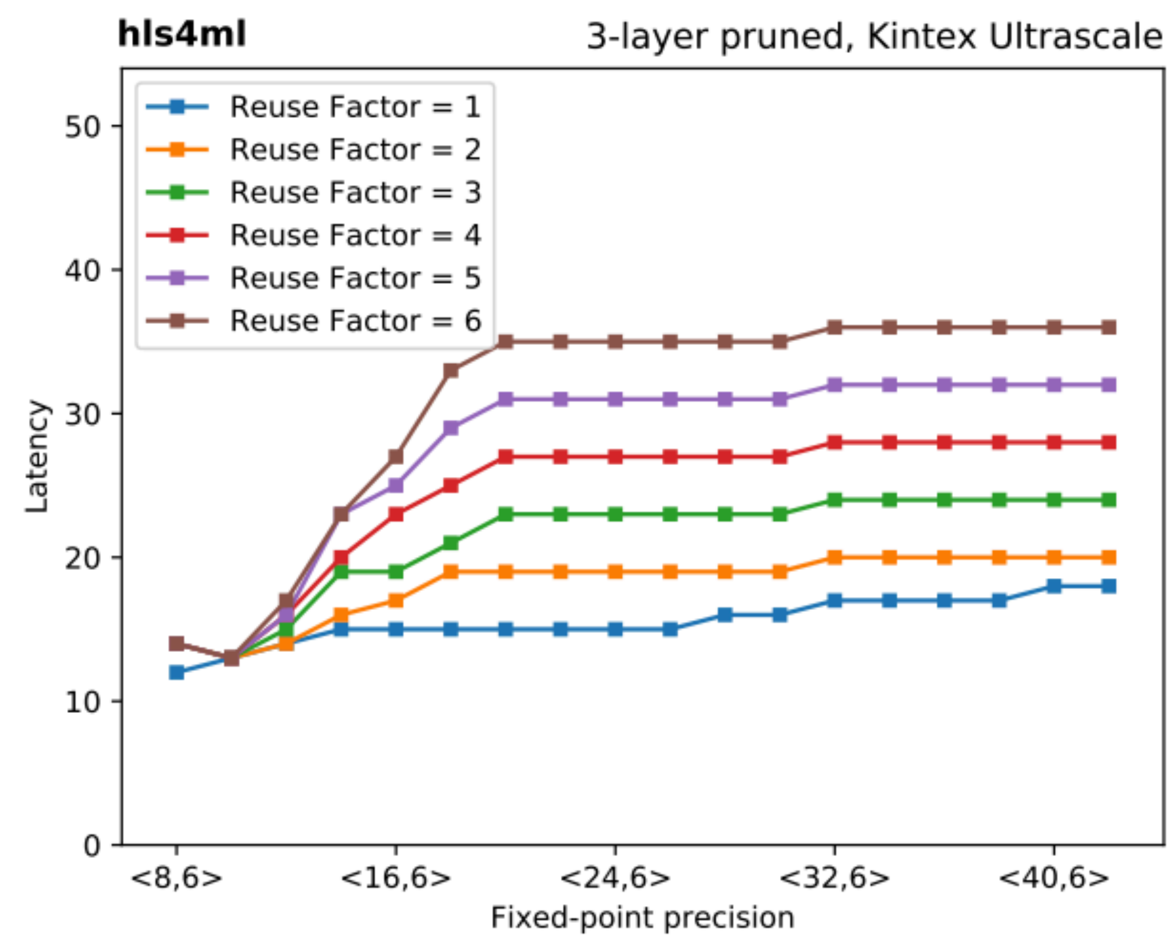
Jean-Roche Vlimant, Hammers&Nails 2019
1810.06111

Fast Decisions

- Use neural networks in L1 Trigger
- Trained offline using normal tools, then translated and optimised for running on FPGAs

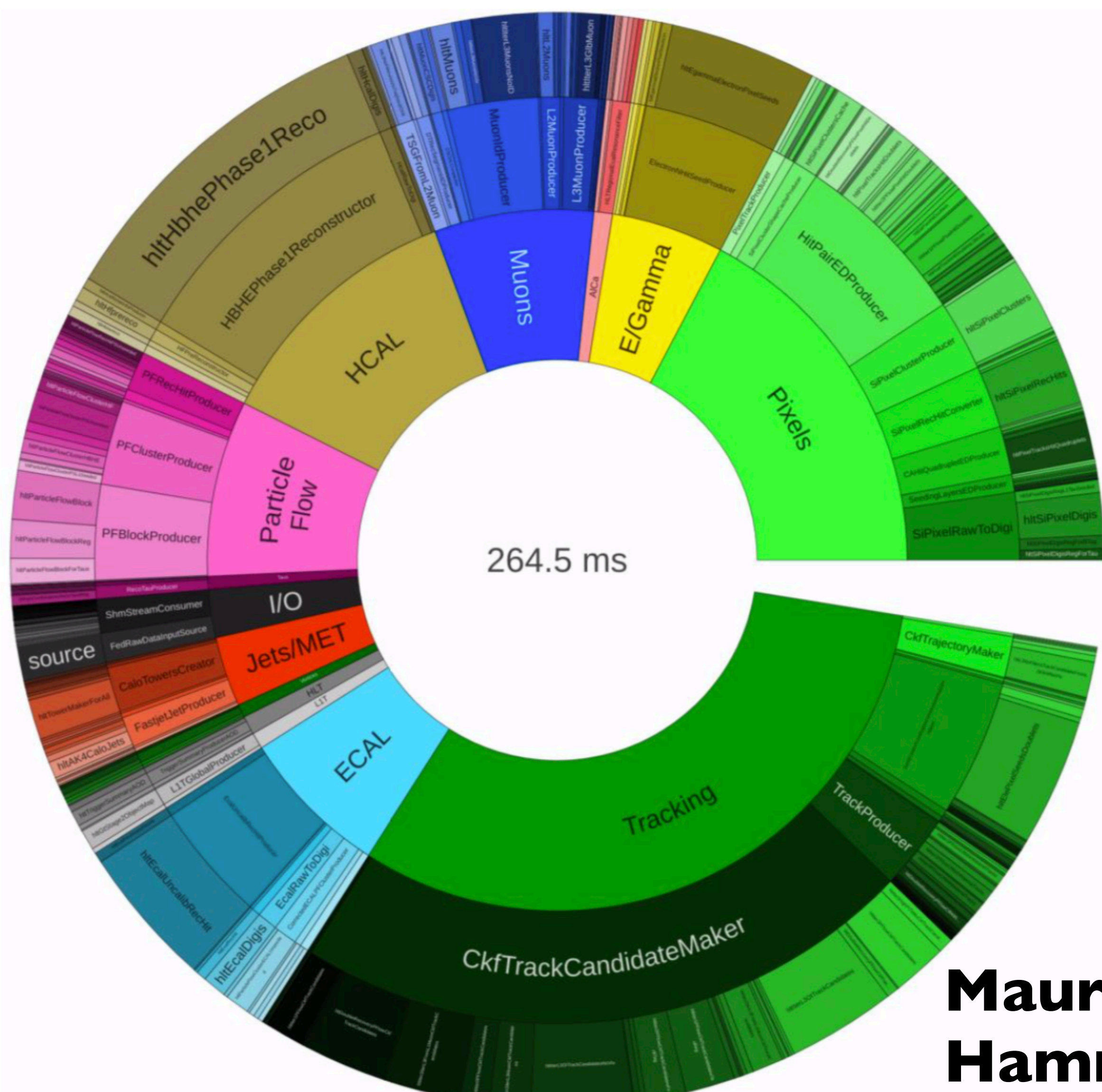


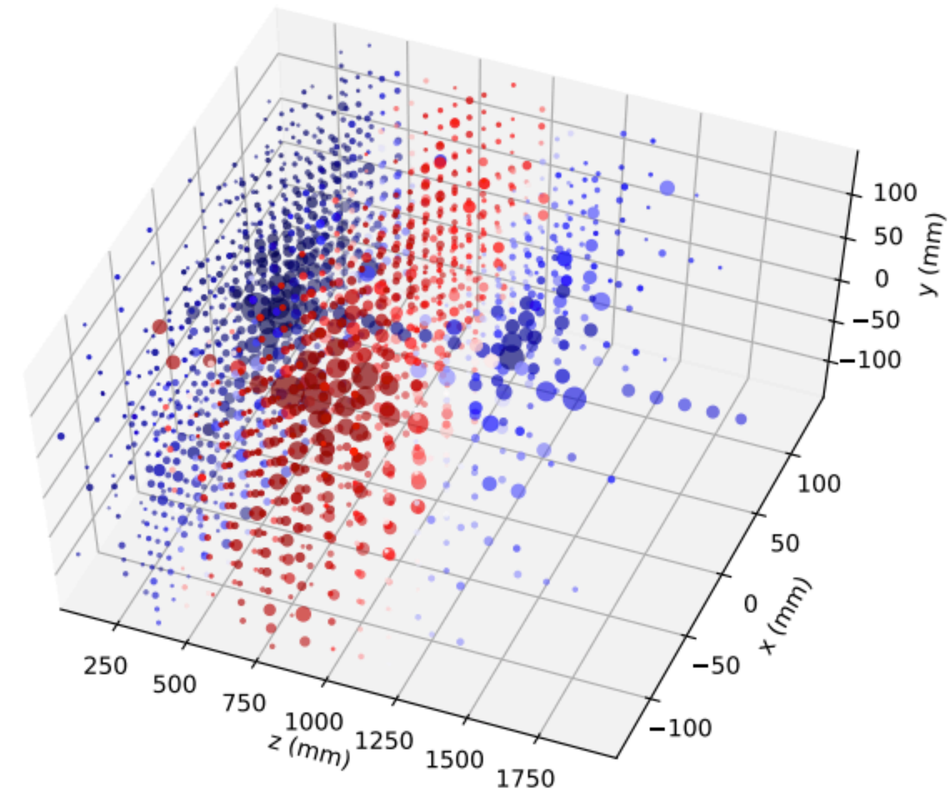




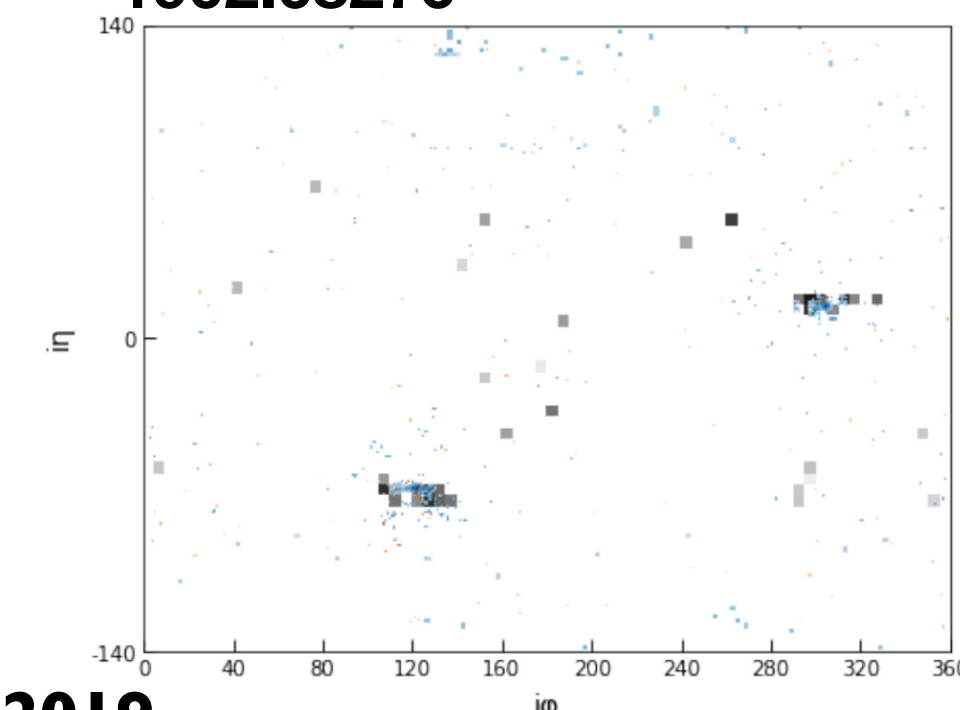
Low Level Reconstruction

- Replace traditional algorithms for reconstruction, object ID and calibration with deep learning
- Increase physics performance and/or resource usage
- Superficially less attractive, potentially much more useful
- End-to-end learning?



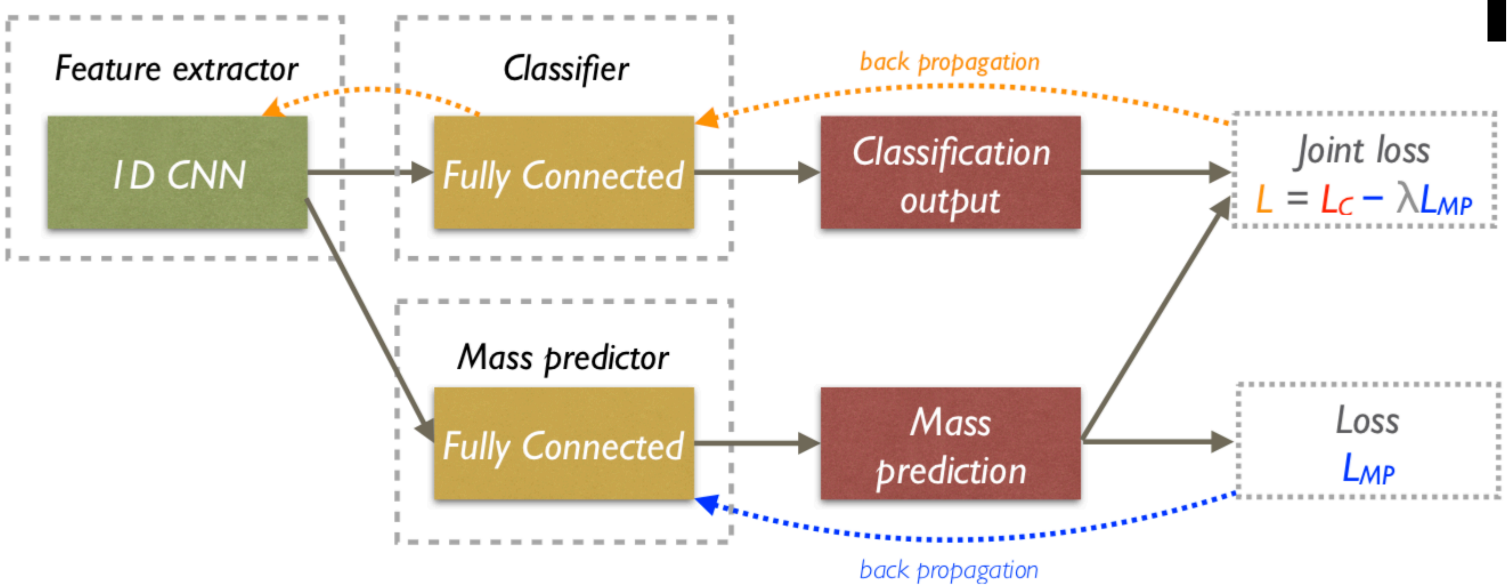


1902.08276

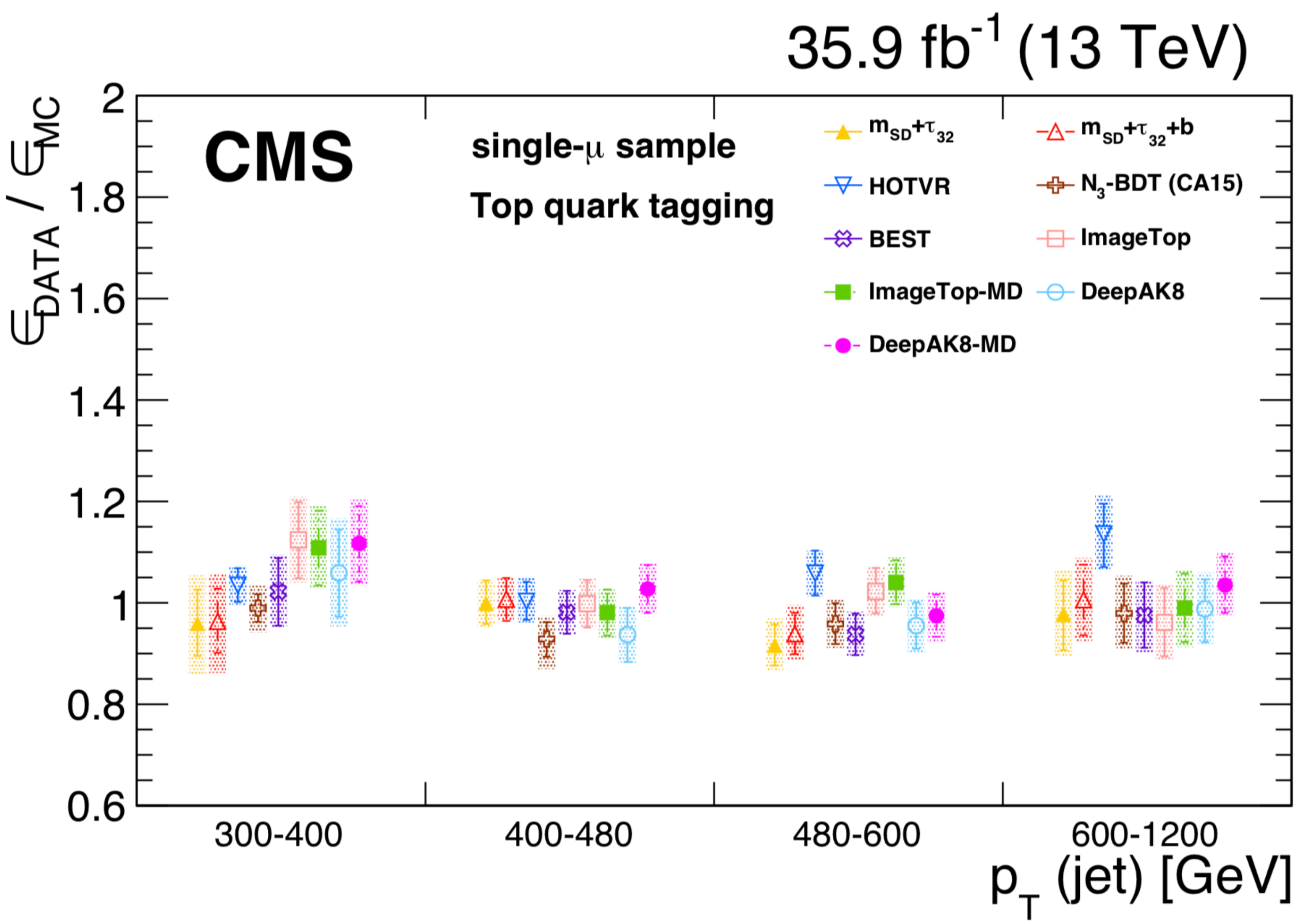


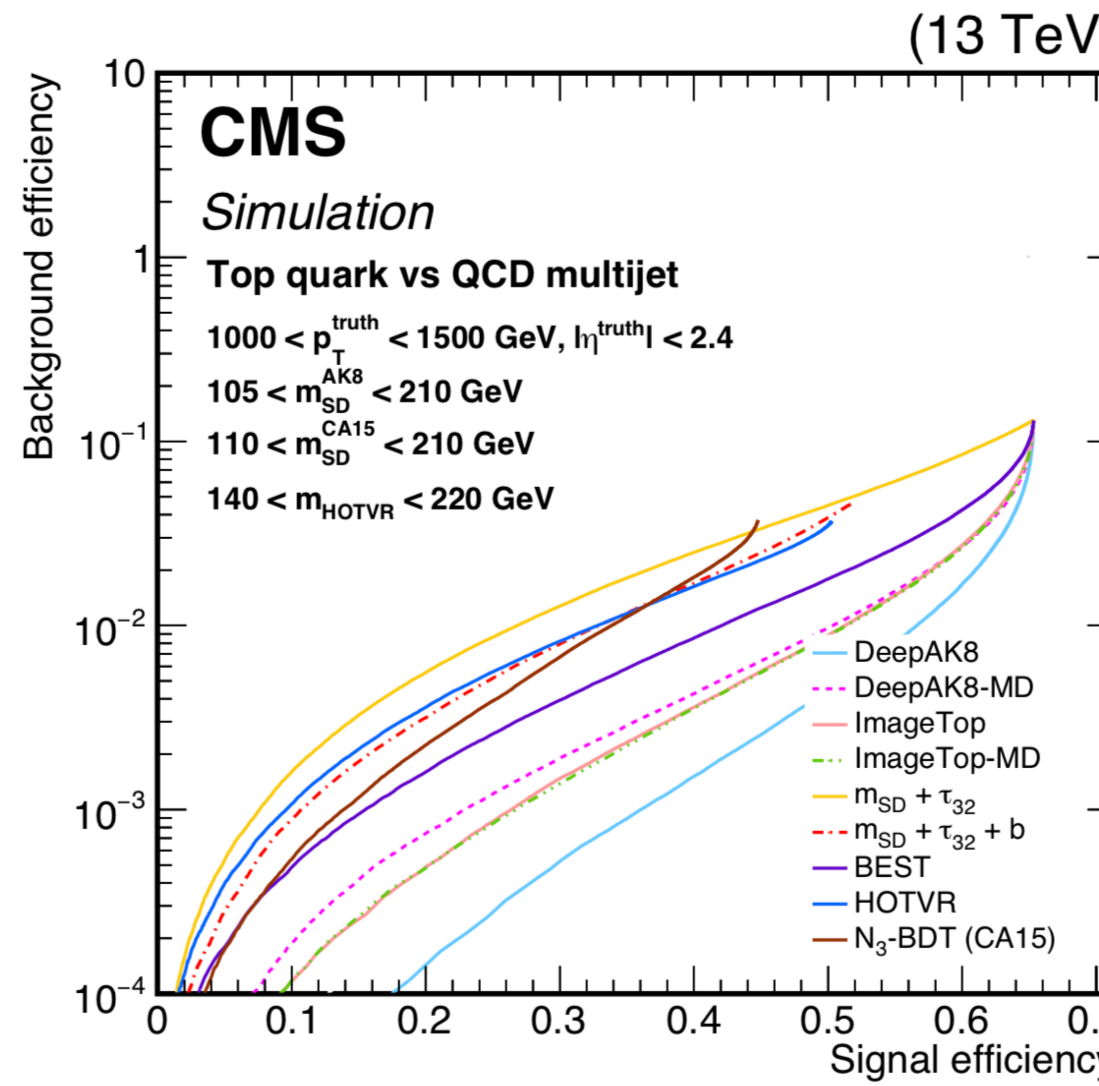
Maurizio Pierini
Hammers&Nails 2019

Heavy Resonance Tagging in CMS



Performance on realistic simulation and data in CMS: JME-18-002





Distance Correlation

$$x_{jk} = |X_j - X_k|$$

Distances of all examples in batch for classifier output

$$y_{jk} = |Y_j - Y_k|$$

... for variable to decorrelate

$$\hat{x}_{jk} = x_{jk} - \overline{x}_{j.} - \overline{x}_{.k} + \overline{x}_{..}$$

$$\hat{y}_{jk} = y_{jk} - \overline{y}_{j.} - \overline{y}_{.k} + \overline{y}_{..}$$

Center distributions

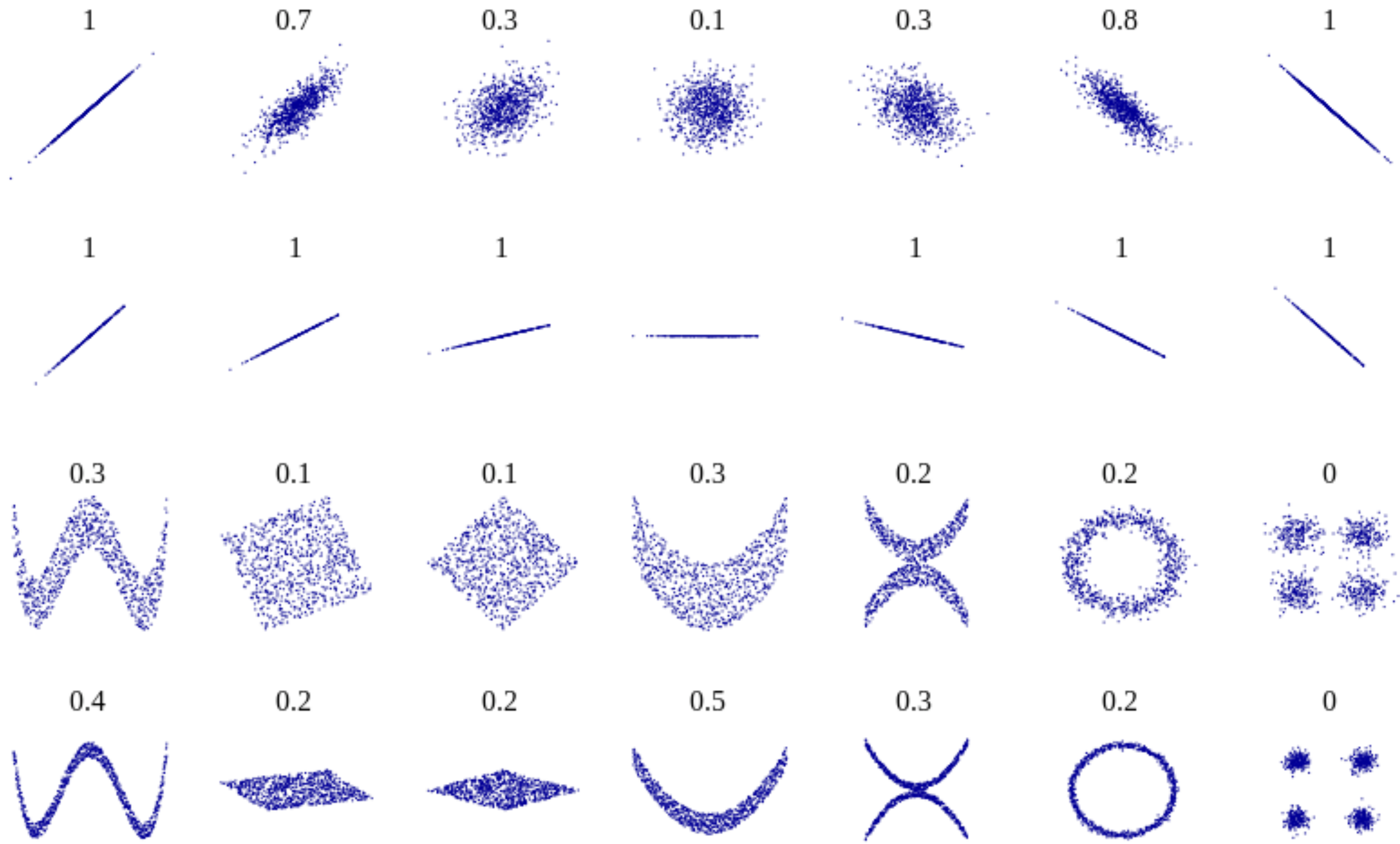
$$\mathrm{dCov}^2 = \frac{1}{n} \sum_j \sum_k \hat{x}_{jk} \hat{y}_{jk}$$

And calculate average product per batch

Some nice properties:

- Zero iff X,Y are independent; positive otherwise!
- Computationally tractable!
- Doesn't require binning!

0803.4101, 1010.0297



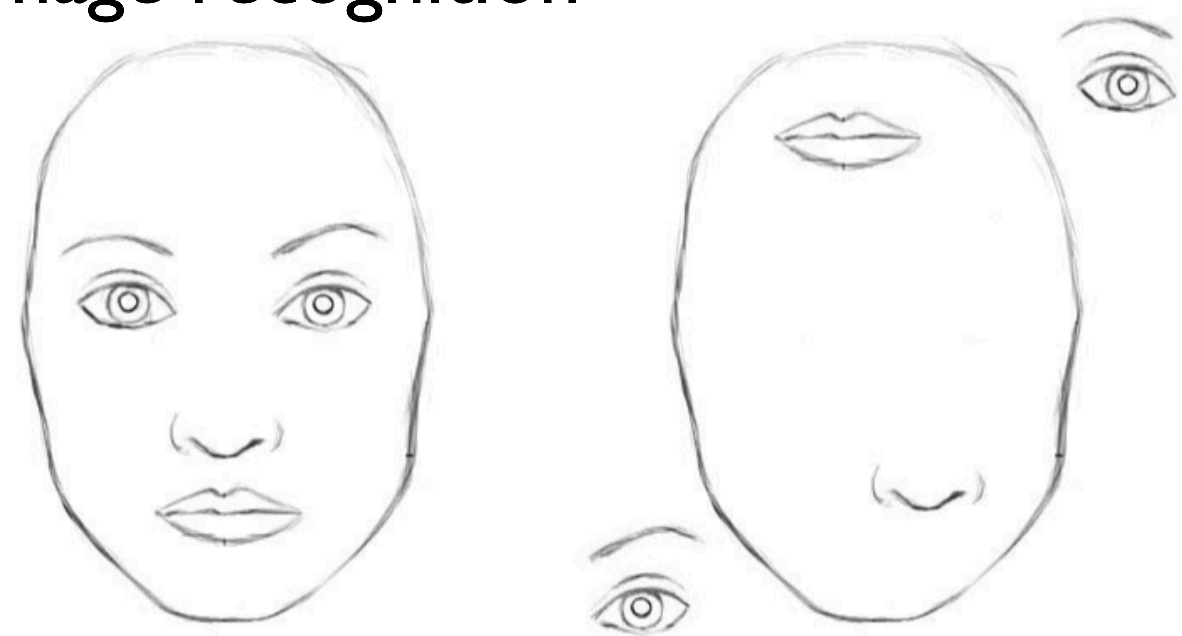
(Wikipedia)

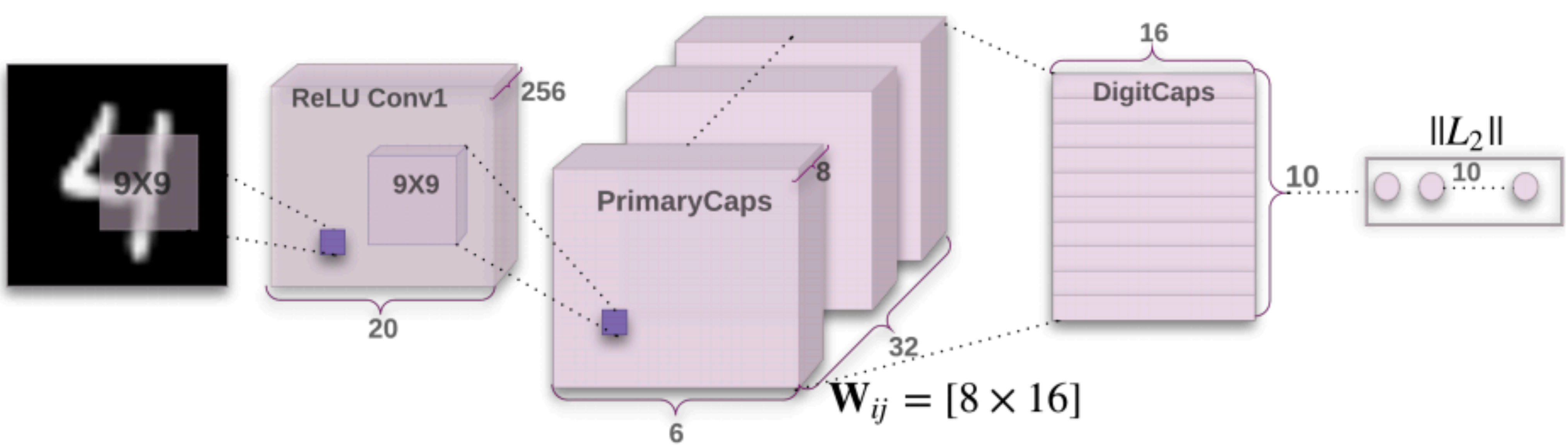
Capsules

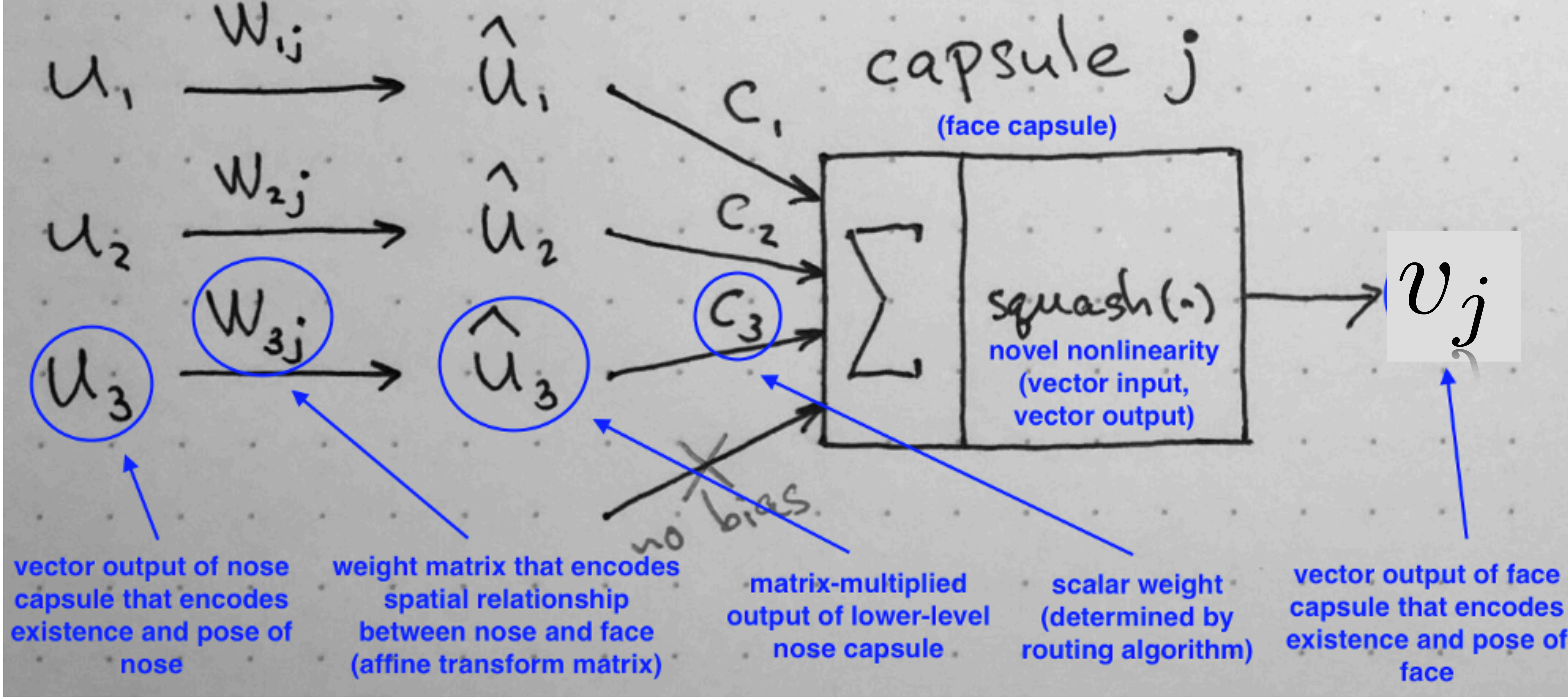
Dynamic Routing Between Capsules
S Sabour, N Frosst, GE Hinton
1710.09829
(medium.com)

Motivation

- CNNs learn features, problem of spatial correlation
- Capsules are a new building block for image recognition
- Learn *instantiation vector*
- Connection by agreement (co-firing)







Softmax & Routing:

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

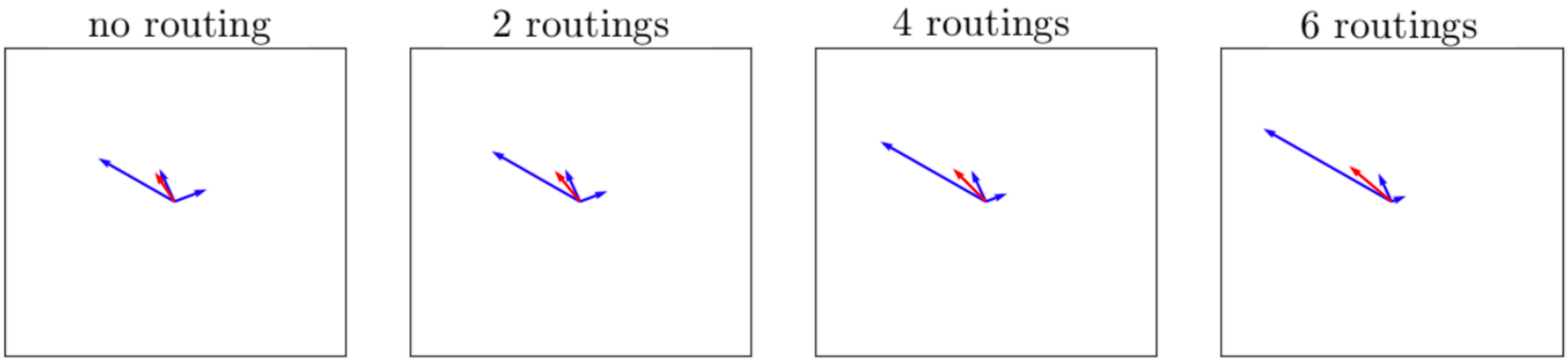
$$b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i} . \mathbf{v}_j$$

Squash:

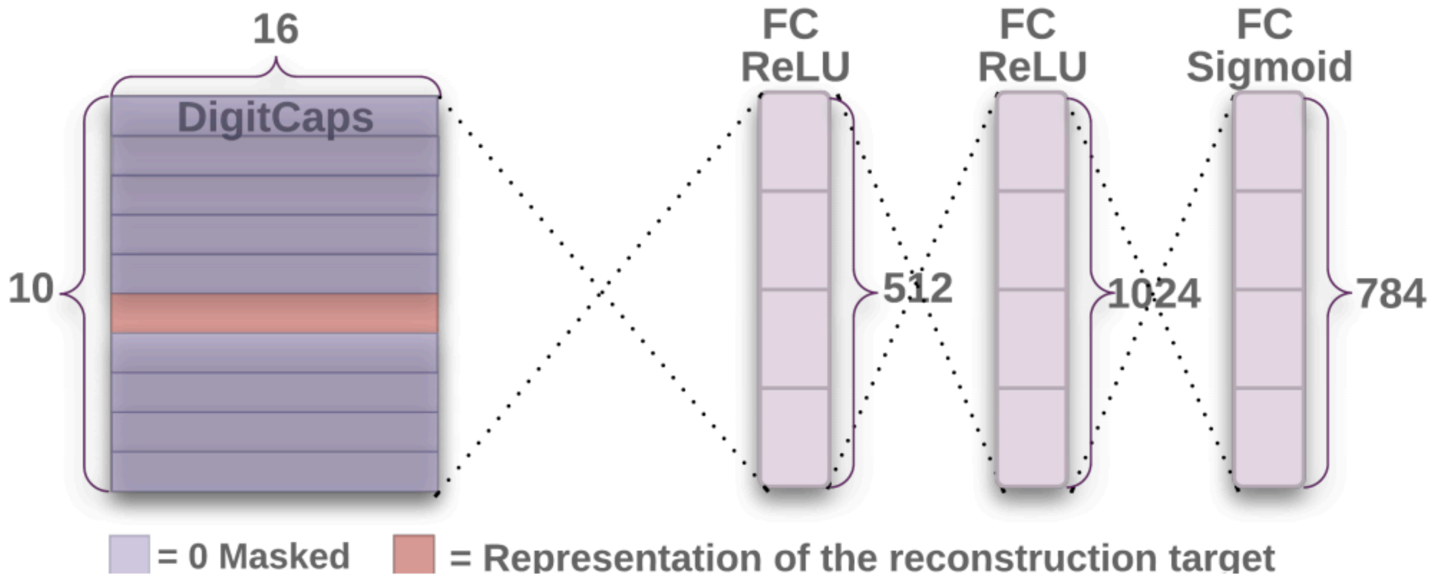
$$\mathbf{v}_j = \frac{||\mathbf{s}_j||^2}{1 + ||\mathbf{s}_j||^2} \frac{\mathbf{s}_j}{||\mathbf{s}_j||}$$

- Vector instead of scalar representation
  - Instantiation and relative positioning
- Routing by agreement

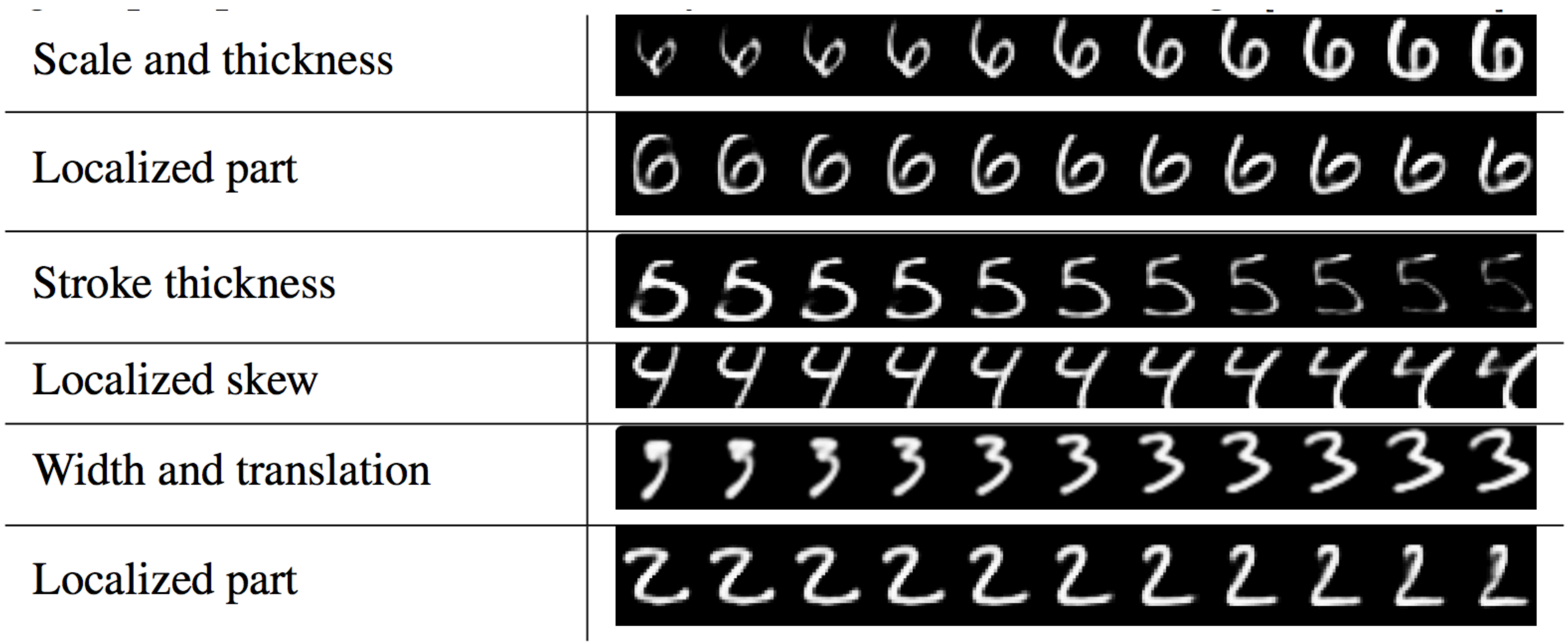
Dynamic Routing Between Capsules
S Sabour, N Frosst, GE Hinton
1710.09829
pechyonkin.me



Routing by agreement



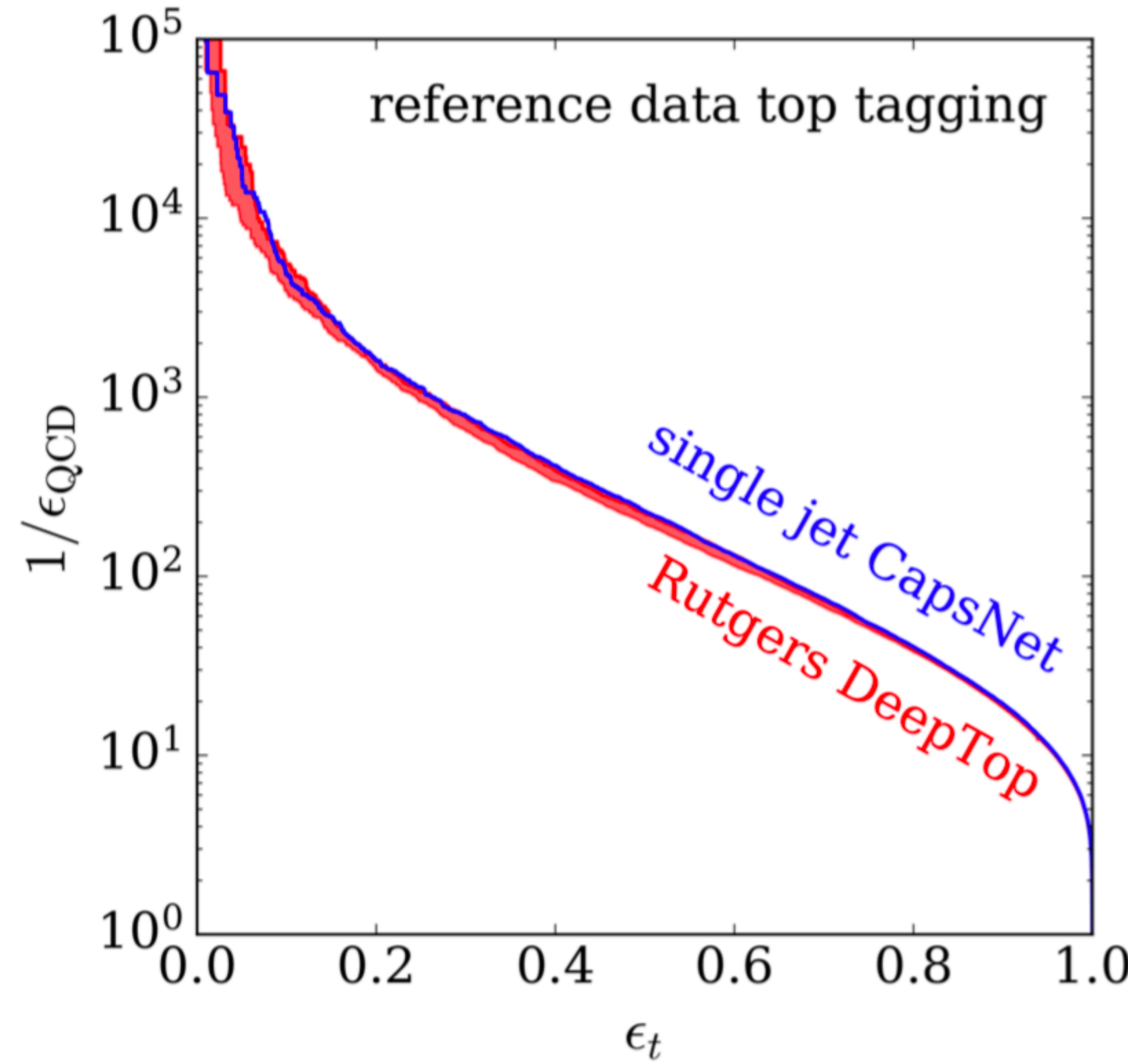
Learned Instantiation

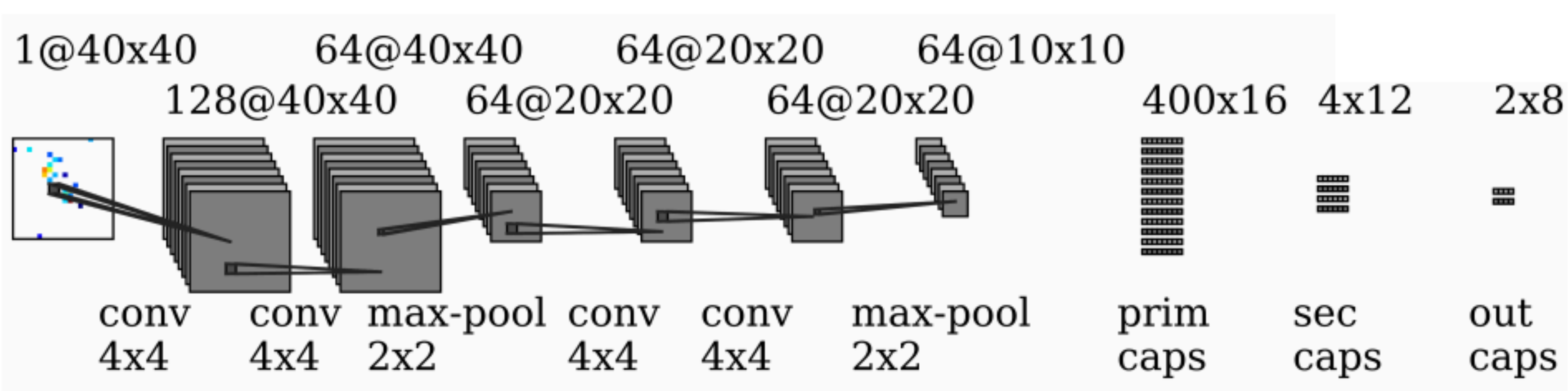


Dynamic Routing Between Capsules
S Sabour, N Frosst, GE Hinton
1710.09829

Transfer to Physics

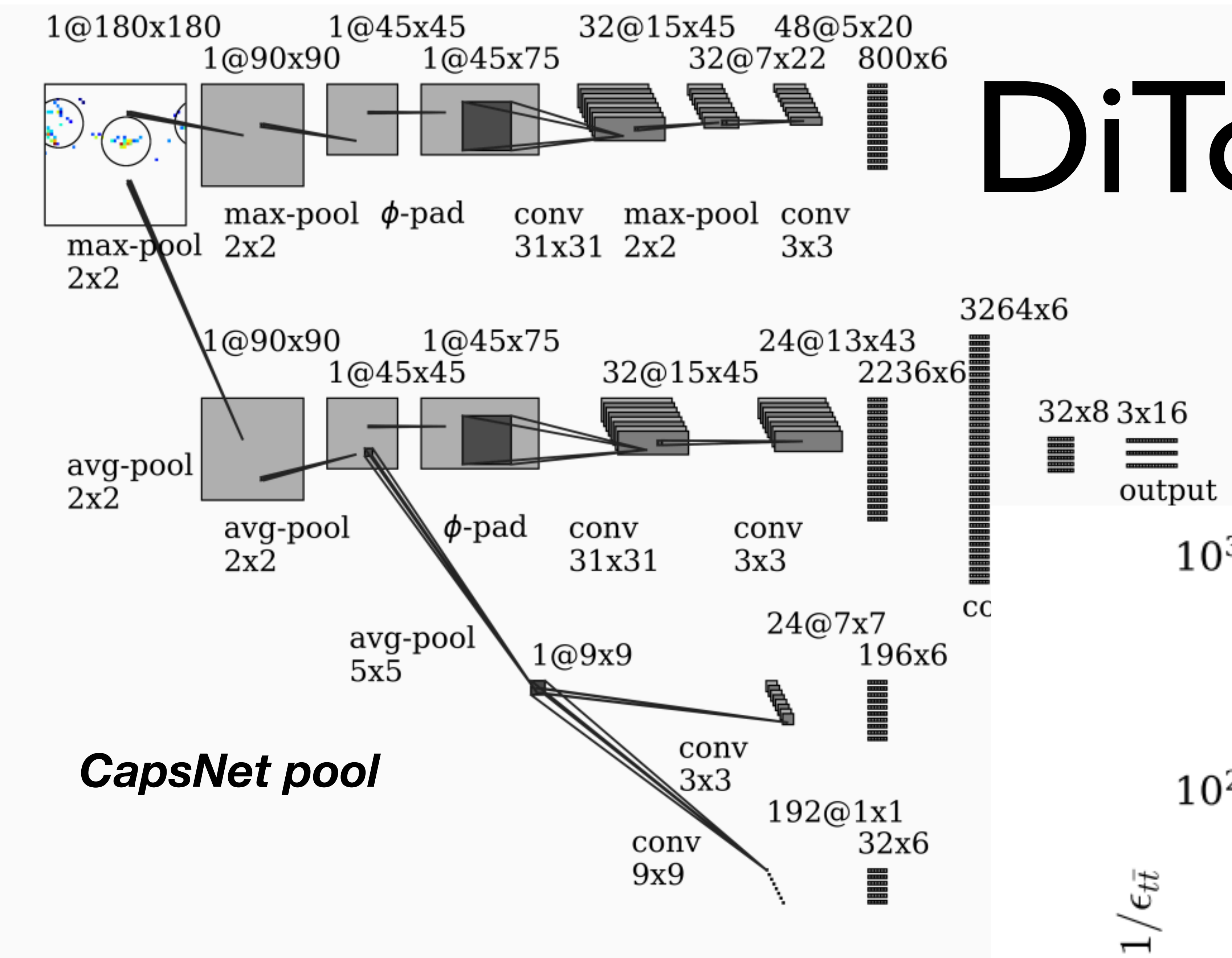
- Promised advantages of capsule networks:
  - Better interpretability of learned capsules
  - Better performance than CNN in dense environments (multiple particles overlapping)
  - Extract substructure and global information simultaenously

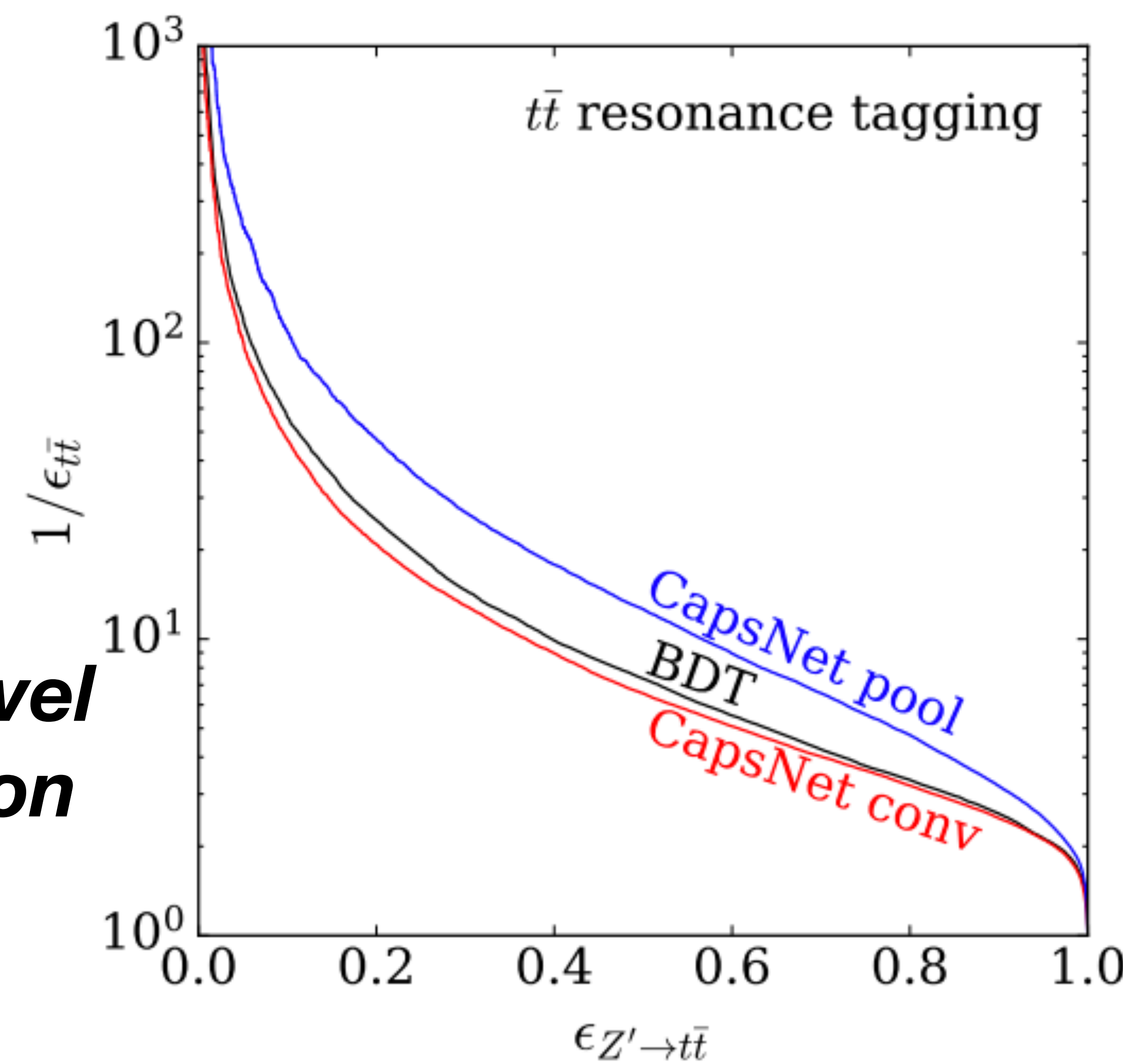




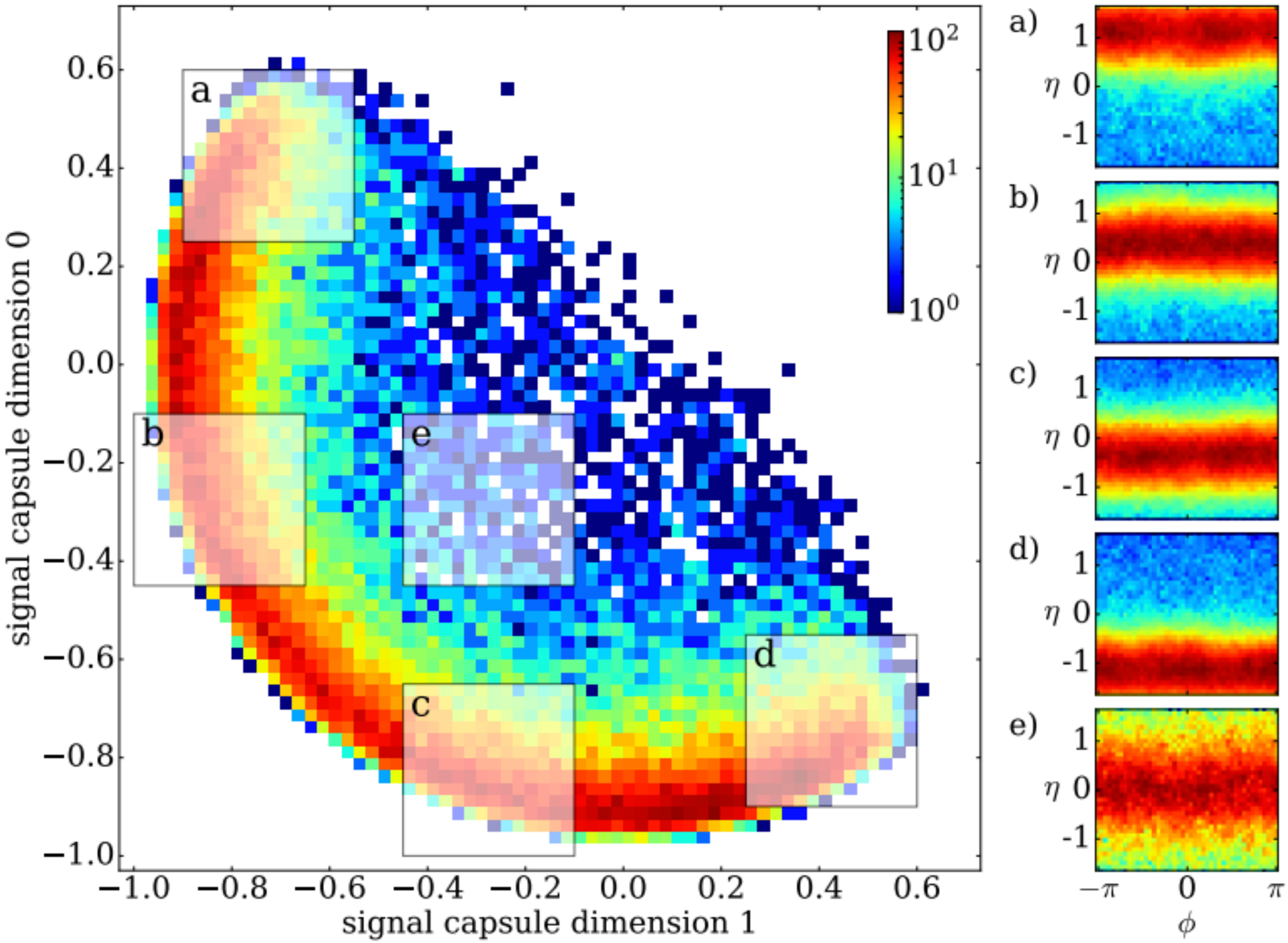
Can reproduce CNN on standard top tagging dataset

DiTop vs QCD



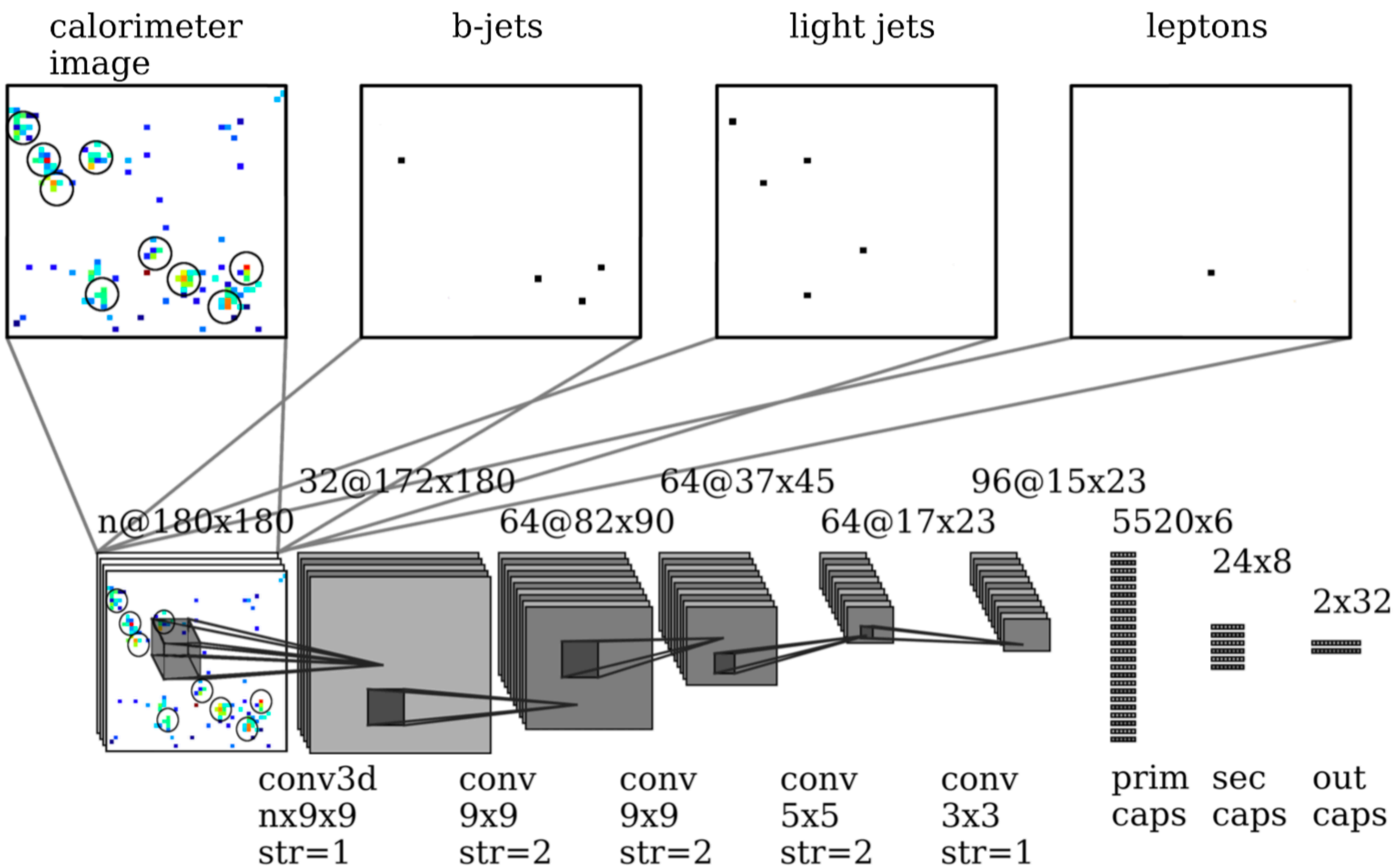


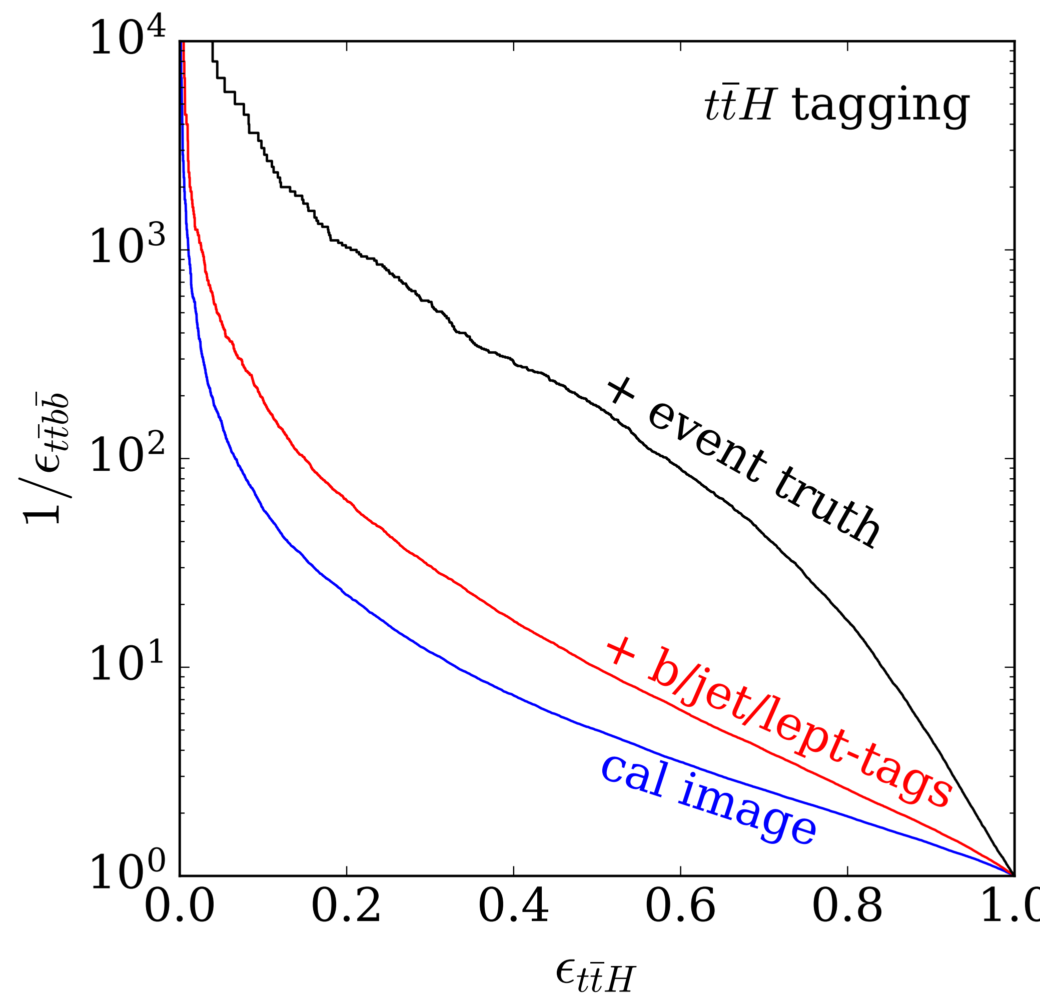
Improved performance over high-level event variables as well as convolution based CapsNet



Can interpret output capsule distributions

ttH(bb) vs ttbb





Capsules learn reconstruction of complex final state on event level using multiple input sources

Bayesian Networks

Bayes theorem

Weights given model **Model given weights** **Prior weights**

$$p(\omega|C) = \frac{p(C|\omega)\, p(\omega)}{p(C)}$$

Model evidendence

Now we can sample and predict response for new data points:

$c^* \ldots \mathrm{New\ datapoint}$

$$p(c^*|C) = \int d\omega\, p(c^*|\omega)\, p(\omega|C)$$

Usually do not know a closed form of this. Intractable.

Approximate with Gaussian:

$$q(\omega)\ldots\text{Gaussian distributions parametrised by }\mu\text{ and }\sigma$$

$$\int d\omega\; p(c^*|\omega)\; p(\omega|C) \approx \int d\omega\; p(c^*|\omega)\; q(\omega)$$

Successful if distributions agree:
(measure by Kullback-Leibler divergence)

$$\mathrm{KL}[q(\omega), p(\omega|C)] = \int d\omega\; q(\omega)\;\log\frac{q(\omega)}{p(\omega|C)}$$

Still more simplification needed to actually calculate:

$$
\begin{aligned}
\mathrm{KL}[q(\omega), p(\omega|C)] &= \int d\omega\ q(\omega)\ \log \frac{q(\omega)p(C)}{p(C|\omega)p(\omega)} \\
&= \mathrm{KL}[q(\omega), p(\omega)] + \log p(C) \int d\omega\ q(\omega) - \int d\omega\ q(\omega)\ \log p(C|\omega)
\end{aligned}
$$

Still more simplification needed to actually calculate:

$$\mathrm{KL}[q(\omega), p(\omega|C)] = \int d\omega\ q(\omega)\ \log \frac{q(\omega)p(C)}{p(C|\omega)p(\omega)}$$

$$= \mathrm{KL}[q(\omega), p(\omega)] + \cancel{\log p(C) \int d\omega\ q(\omega)} - \int d\omega\ q(\omega)\ \log p(C|\omega)$$

Second term is not relevant:

$$L = \mathrm{KL}[q(\omega), p(\omega)] - \int d\omega\ q(\omega)\ \log p(C|\omega)$$

usual expected likelihood

for Gaussians:

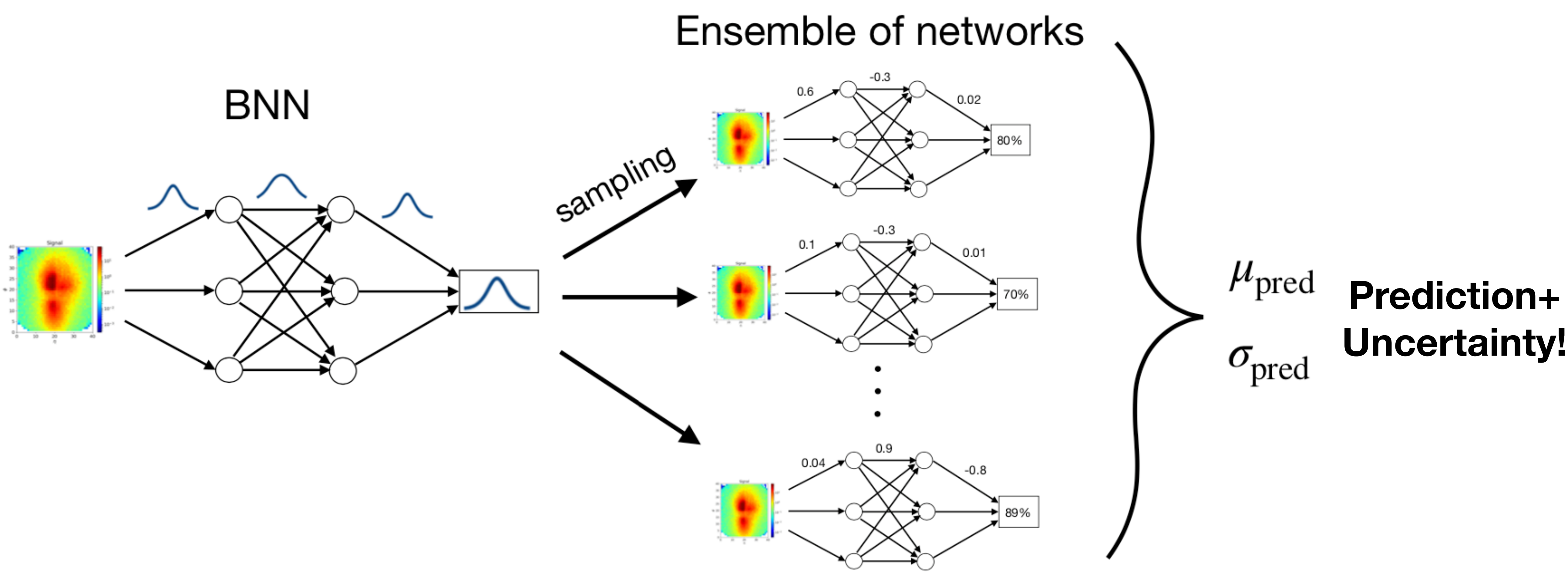
$$\mathrm{KL}[q(\omega), p(\omega)] = \log \frac{\sigma_p}{\sigma_q} + \frac{\sigma_q^2 + (\mu_q - \mu_p)^2}{2\sigma_p^2} - \frac{1}{2}$$

Training

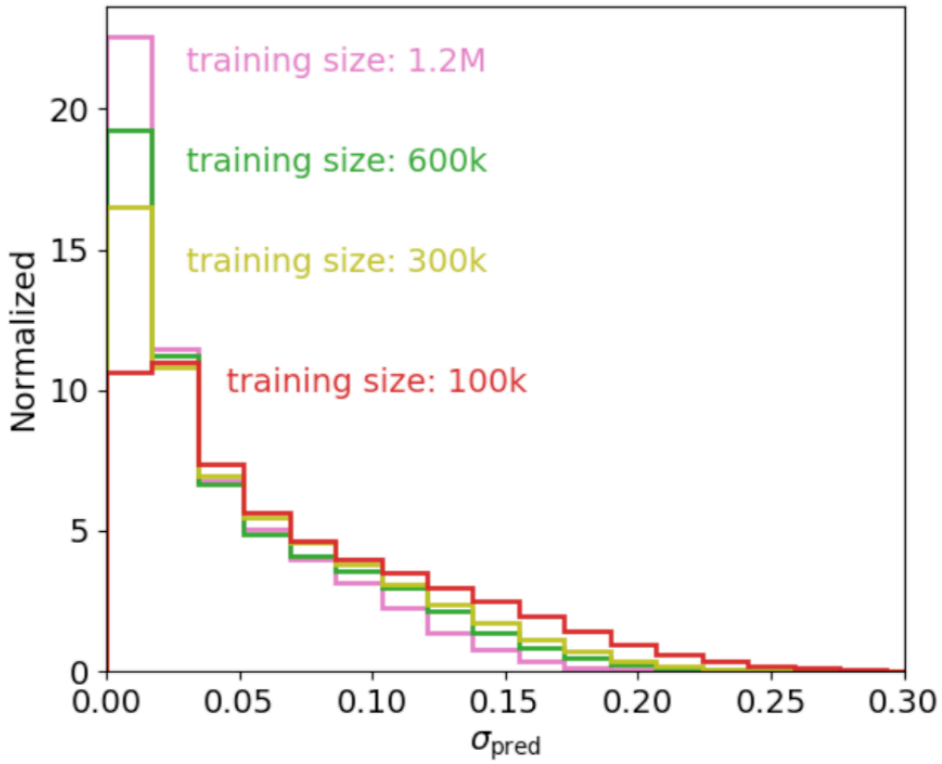
Approximate via MC sampling:

$$p(c^*|C) \approx \int d\omega \; p(c^*|\omega)\; q_{\mu,\sigma}(\omega) \approx \frac{1}{N}\sum_j^N \; p(c^*|\omega_j(\mu,\sigma)) \equiv \mu_{\text{pred}}$$

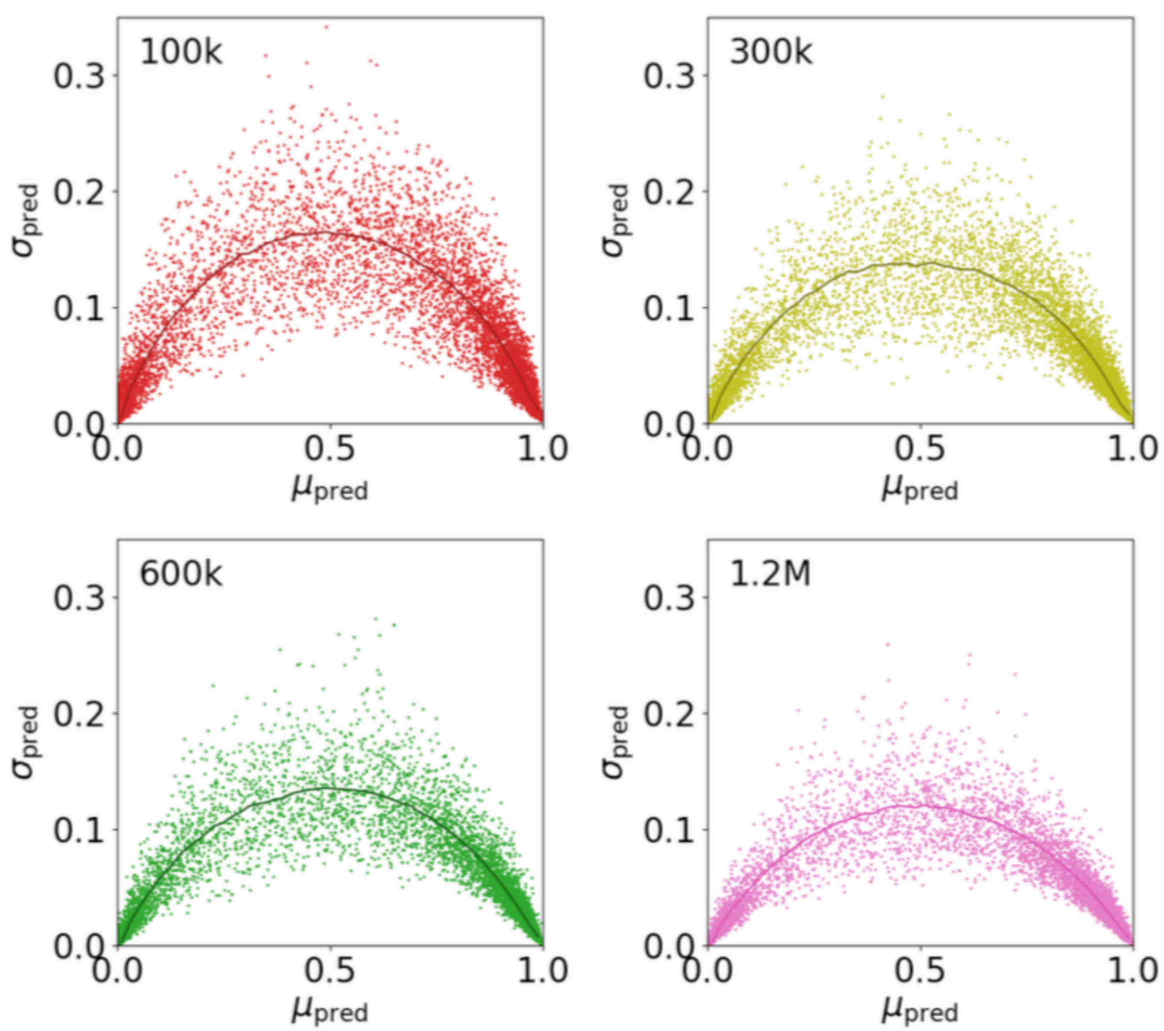
$$\sigma^2_{\text{pred}} = \frac{1}{N}\sum_j^N \left[p(c^*|\omega_j(\mu,\sigma)) - \mu_{\text{pred}}\right]^2$$

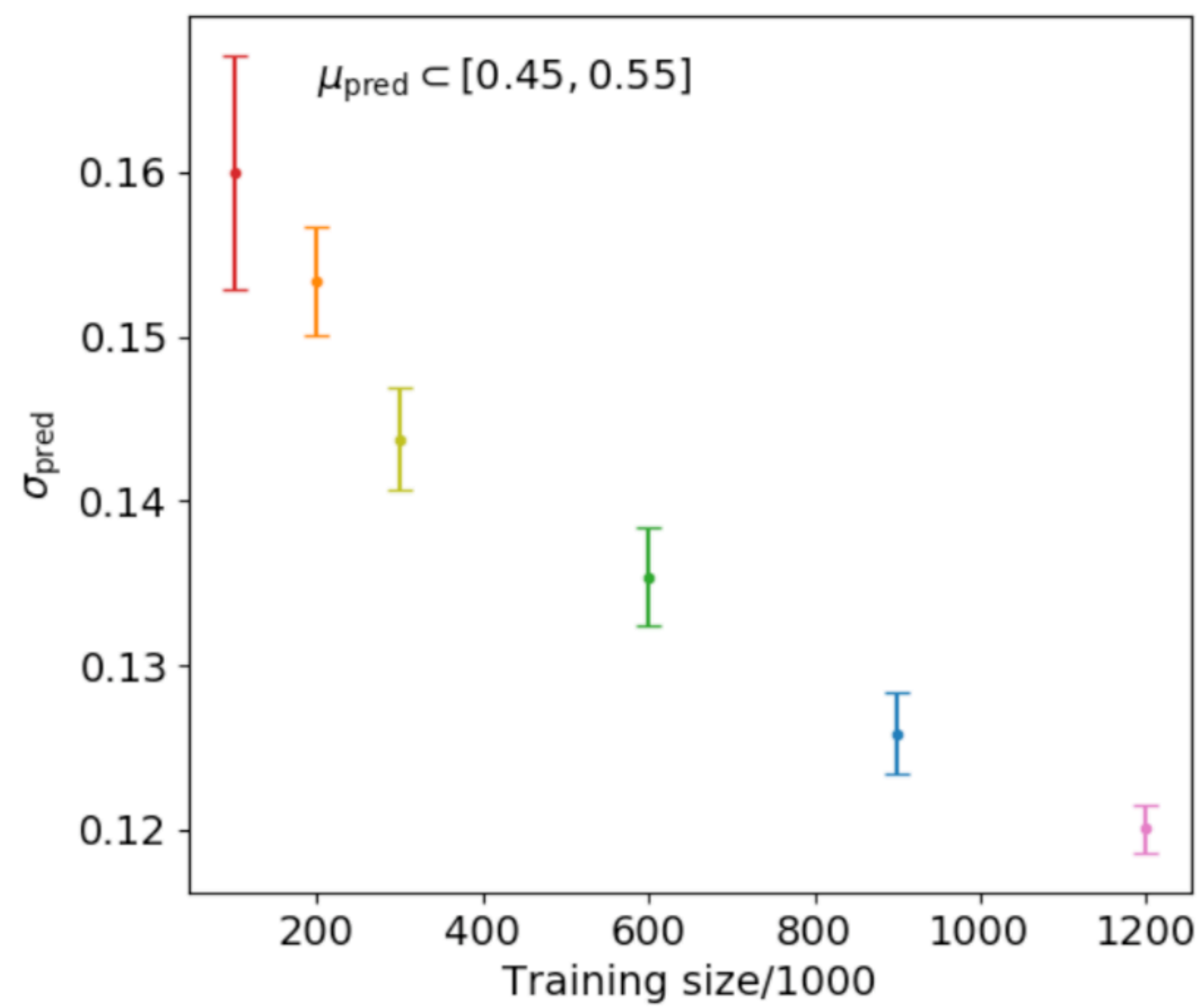


Statistical Uncertainty



BNN captures effect of finite training data

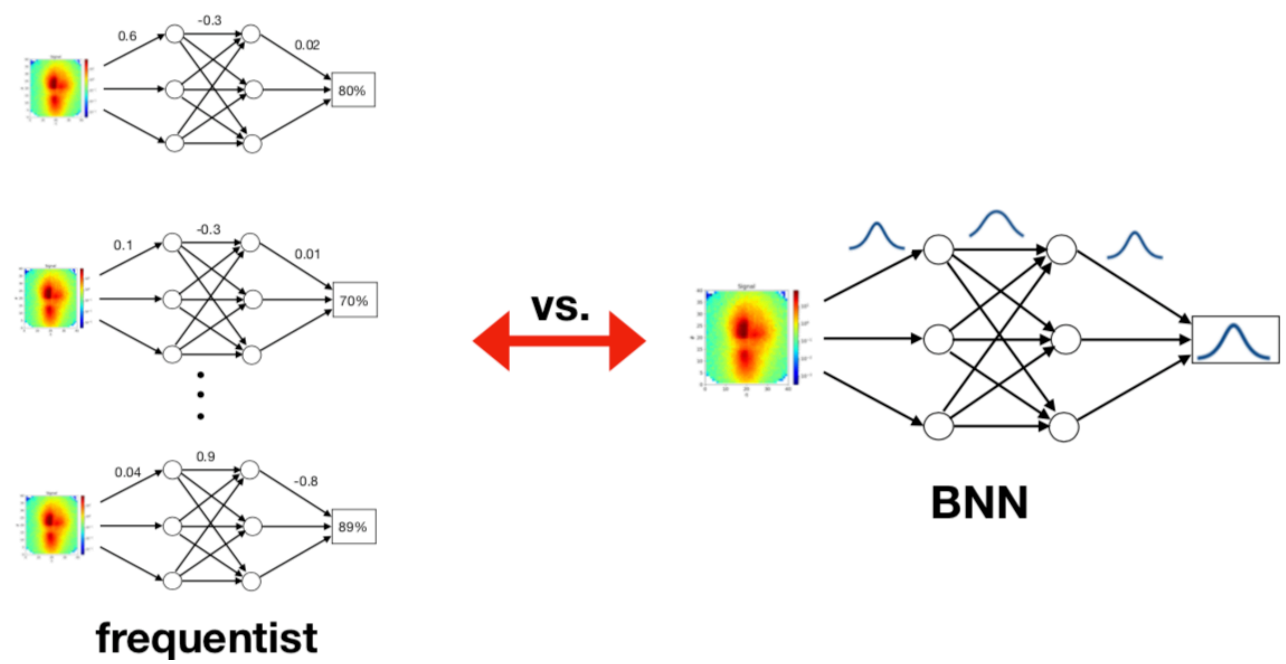


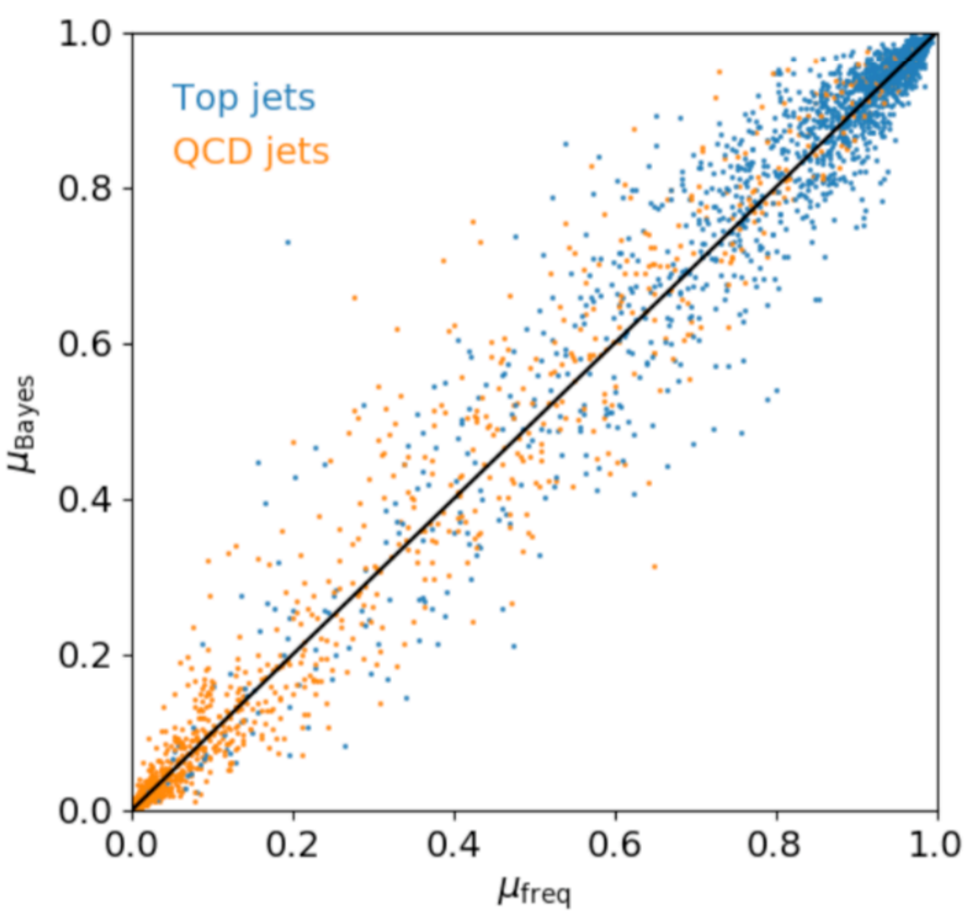


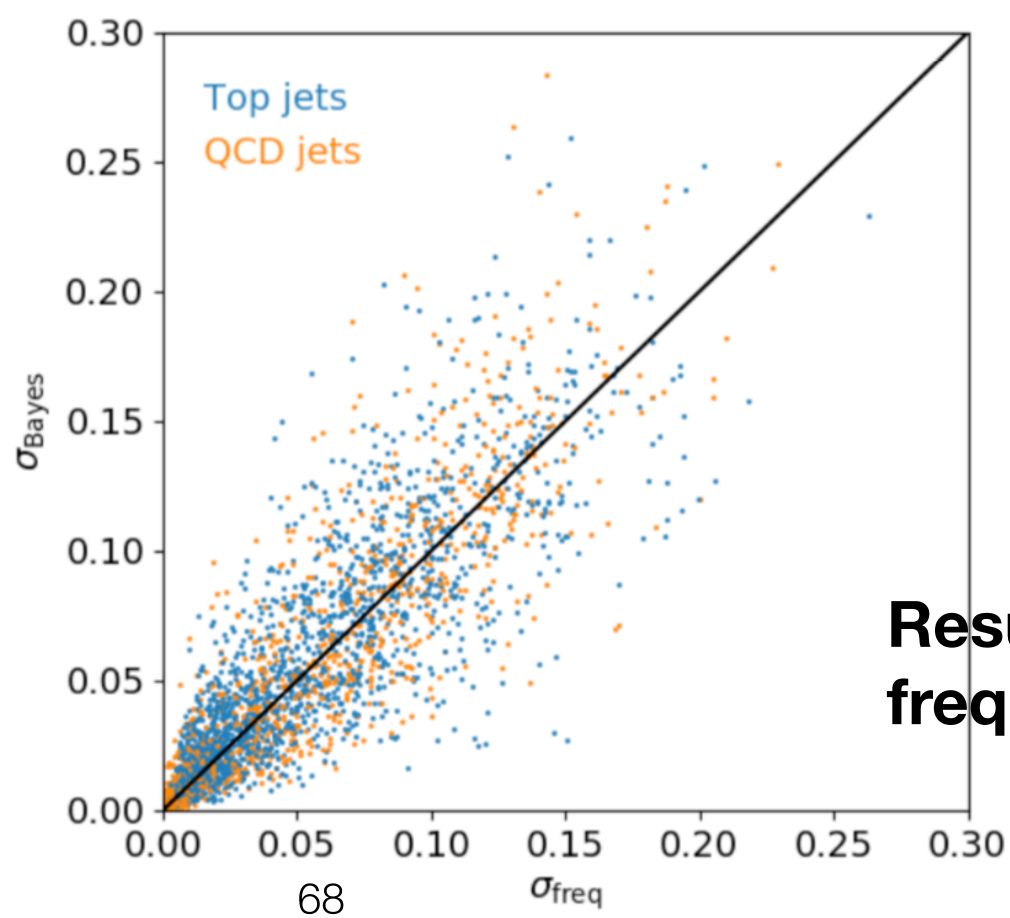
Classification sigmoid correlates mean and standard deviation

Validation

- Train N deterministic networks or statistically independent events
- Calculate mean and standard deviation of ensemble

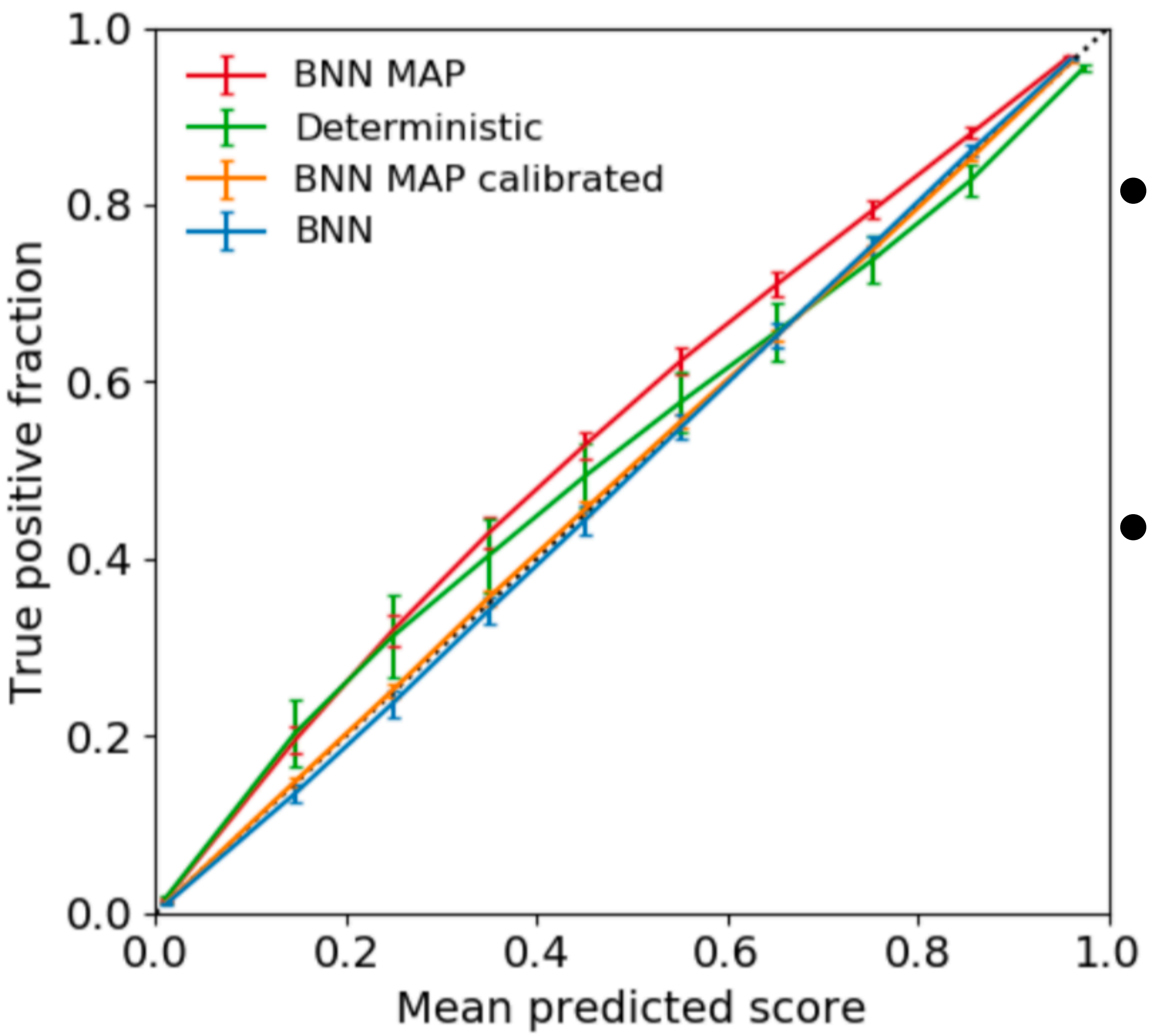




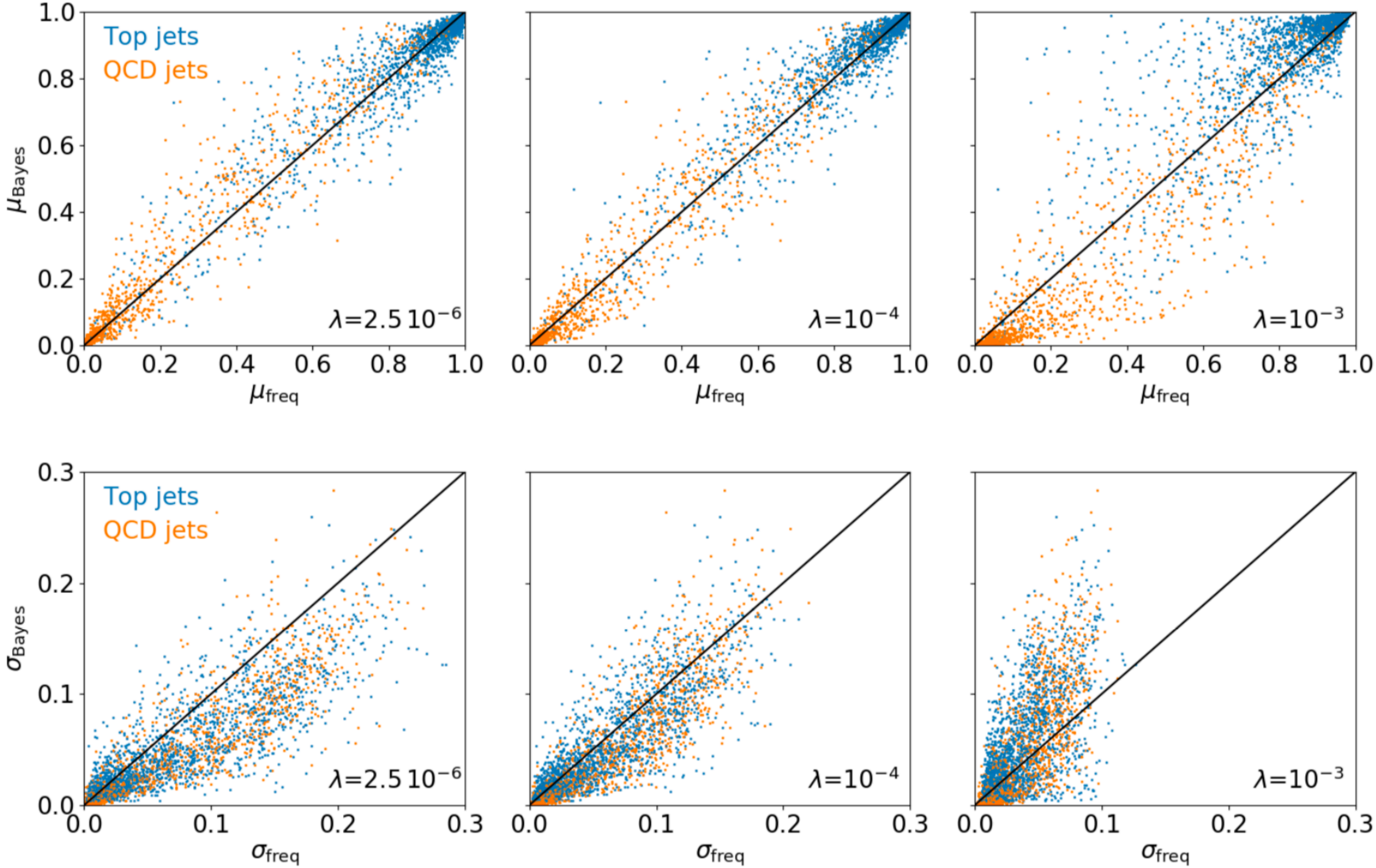


Result agrees with frequentist expectation

Calibration



- BNN is calibrated
    - ie can interpret network output as probability
- Nice, but not really important in HEP usage



$$L = -\log p(C|\omega) - \frac{\mu^2}{2\sigma^2_{\text{prior}}} + \cdots$$

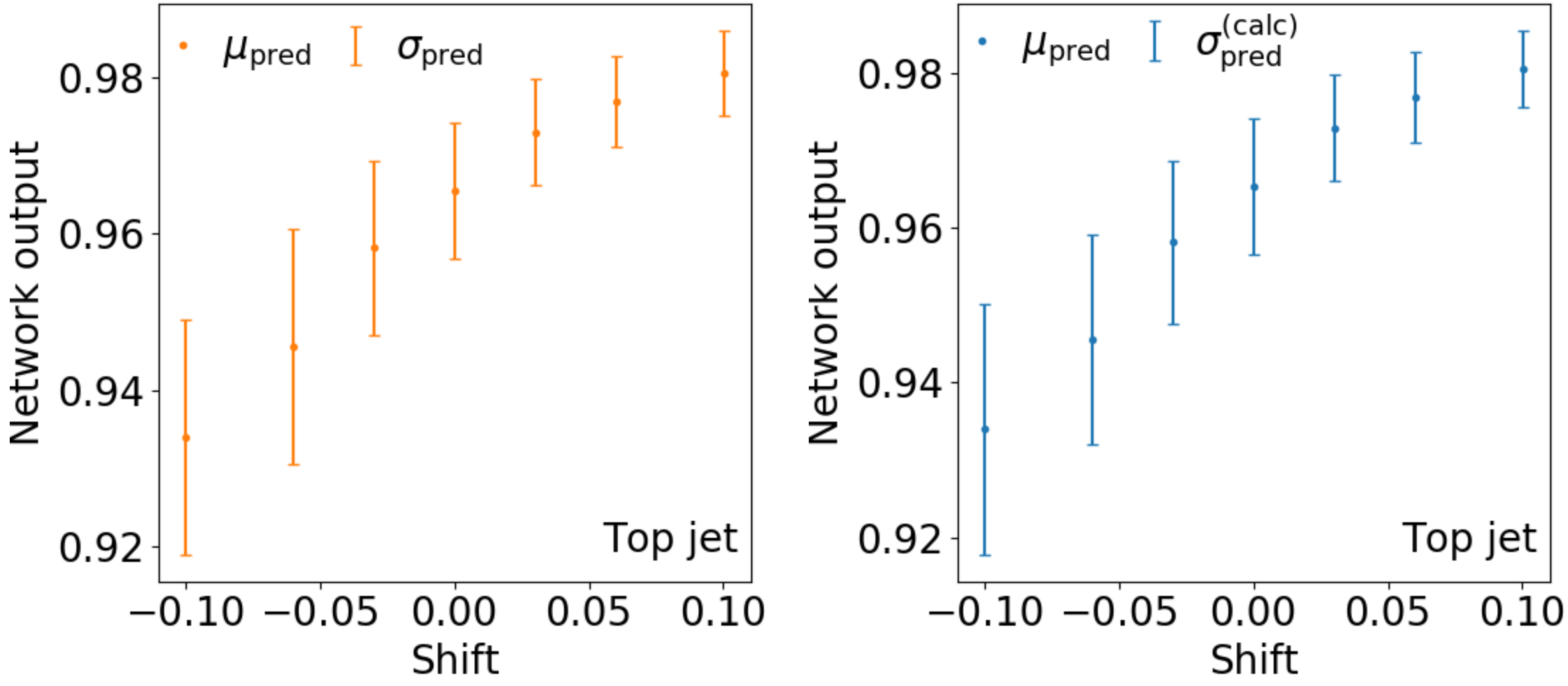
BNN

$$L = -\log p(C|\omega) - \lambda|\omega|^2$$

Determinsitic network with regulariser

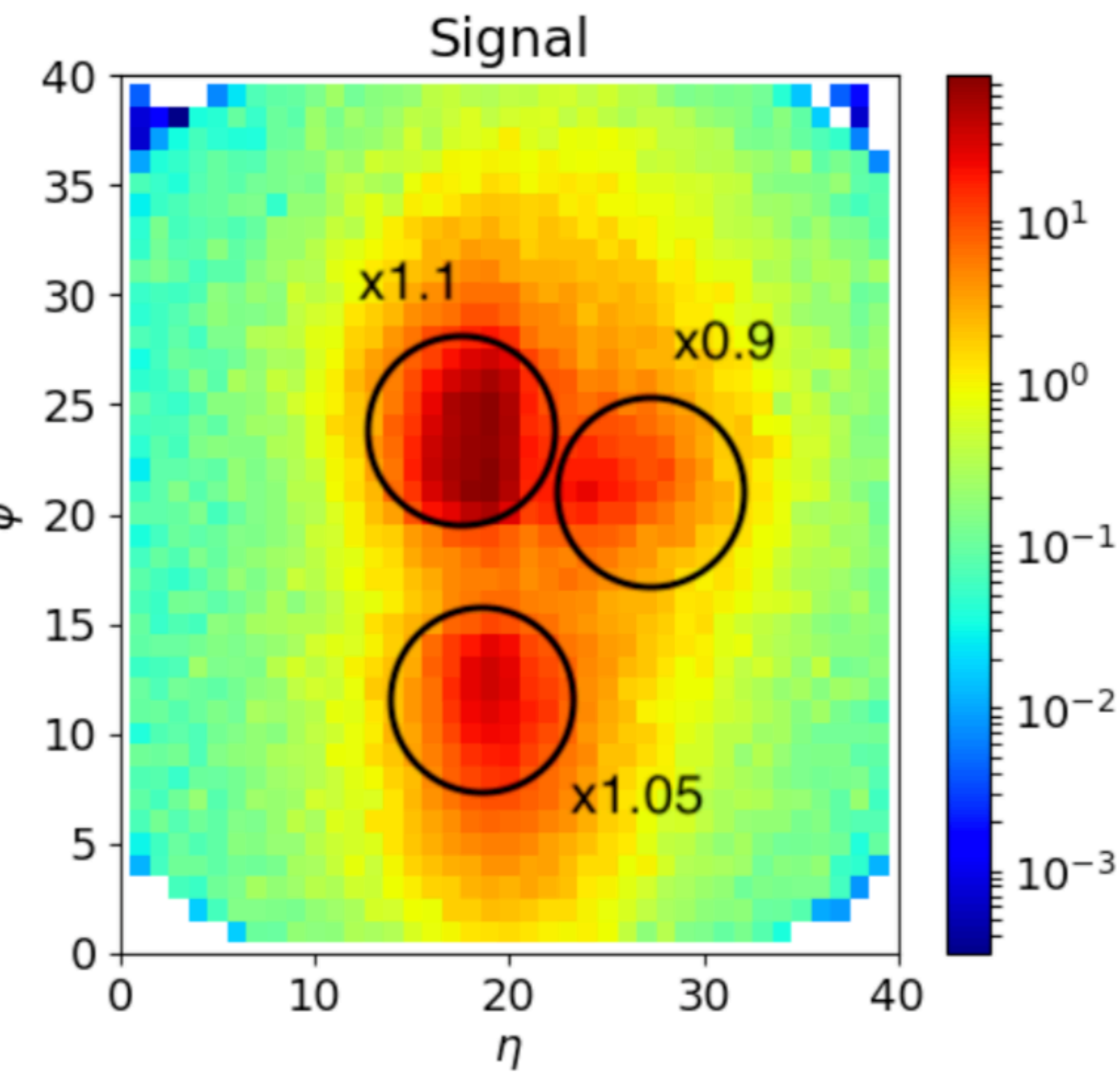
Toy Uncertainty I

JES Inspired: Rescale energy of leading constituent

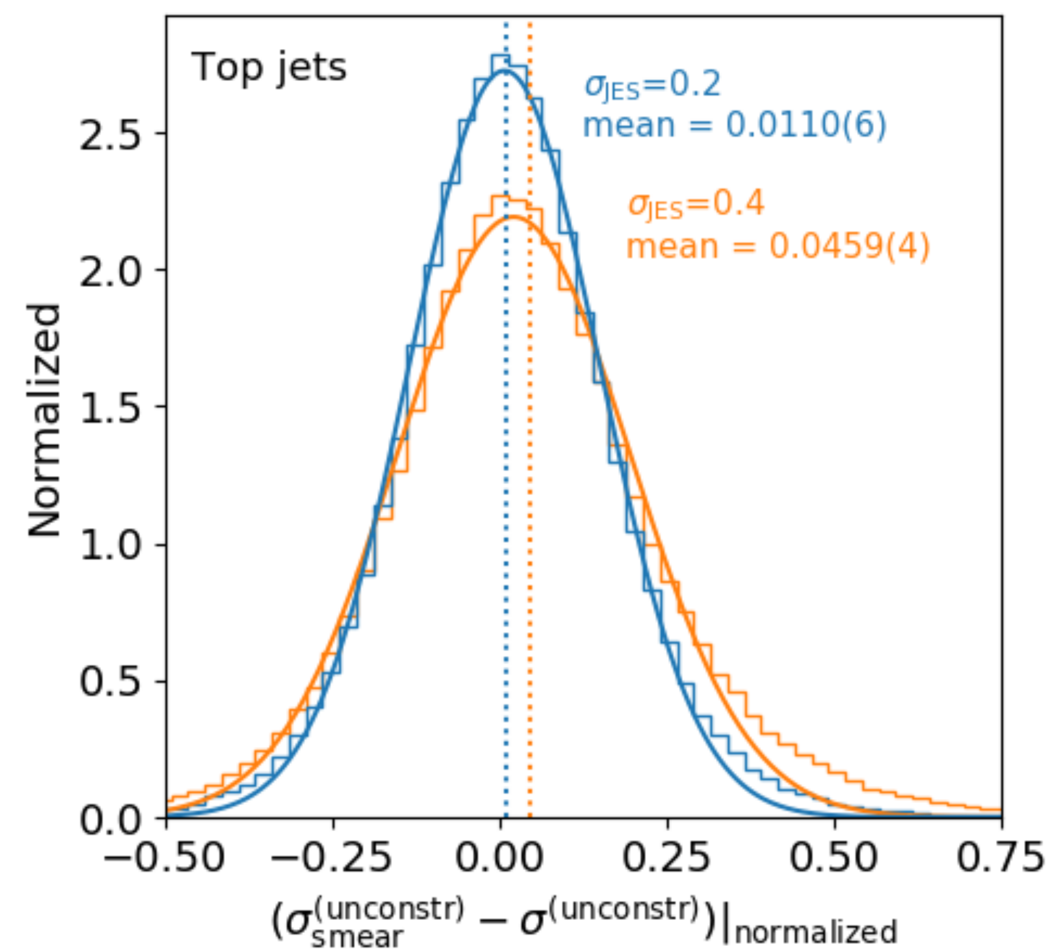


Dominated by shift of predictive mean

Toy Uncertainty II



JES Inspired: Define sub-regions and for each event rescale measurements in each sub-region with an independent scale drawn from a Gaussian



Posterior

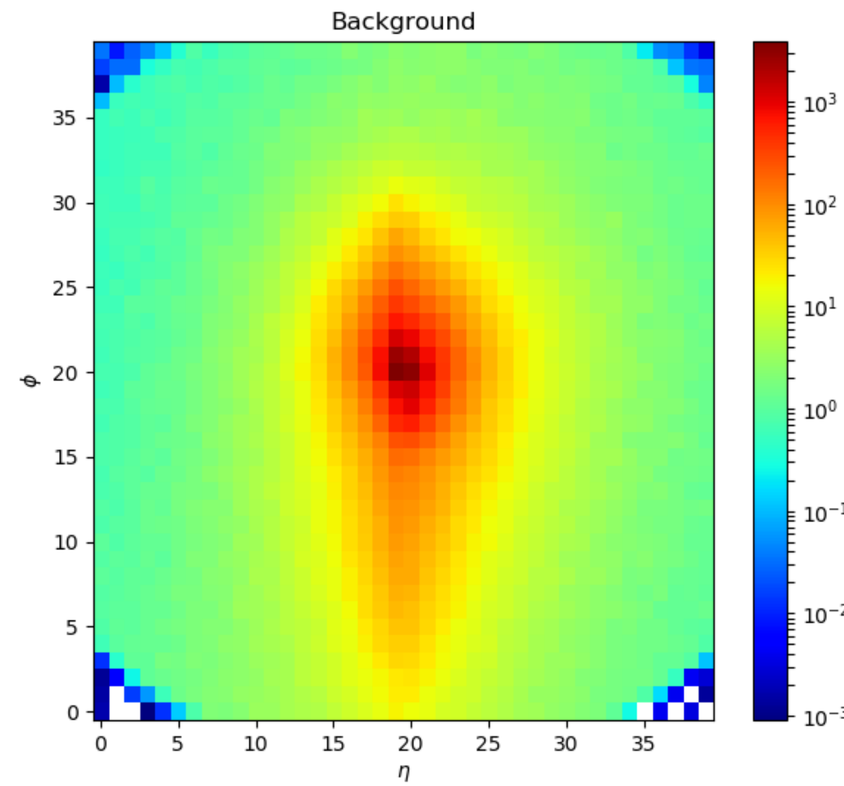
- Bayesian NN give us a measure of uncertainty for a network output
    - Double the weights + MC sampling: somewhat slower, but not significant
- Matches what we expect from sampling over multiple trainings in a frequentist sense
- Other approaches:
    - Simple approximation of Bayes via Dropout
    - Improvements to BBB in Dustin’s talk last week
- Systematic uncertainties (in the HEP sense) - ie testing domain adaptation properties:
    - Effect seen, need to study in more detail for practical procedure

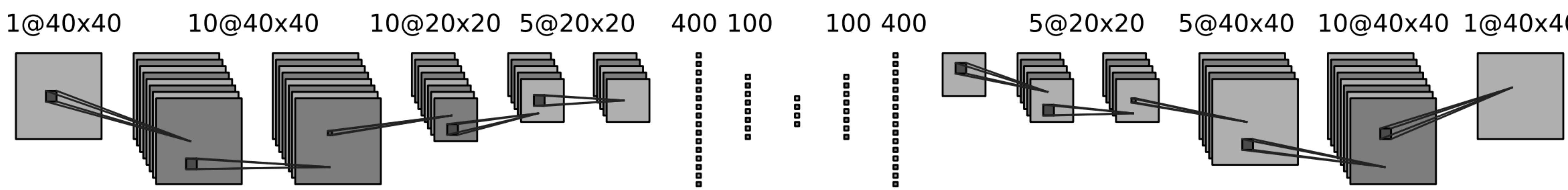
Anomalies

Architecture I

- Reconstruct energy with calorimeter (improve resolution using tracker)
- Cluster energy deposits into jet
- Preprocess:

  - center → rotate → flip (twice) → pixelate → crop → normalise
    - center: centroid is at (0/0)
    - rotate: principal axis is vertical
    - flip: in (x<0, y>0)-plane maximum intensity
    - crop: to nxn images
    - normalise: intensity of each pixel divided by total intensity



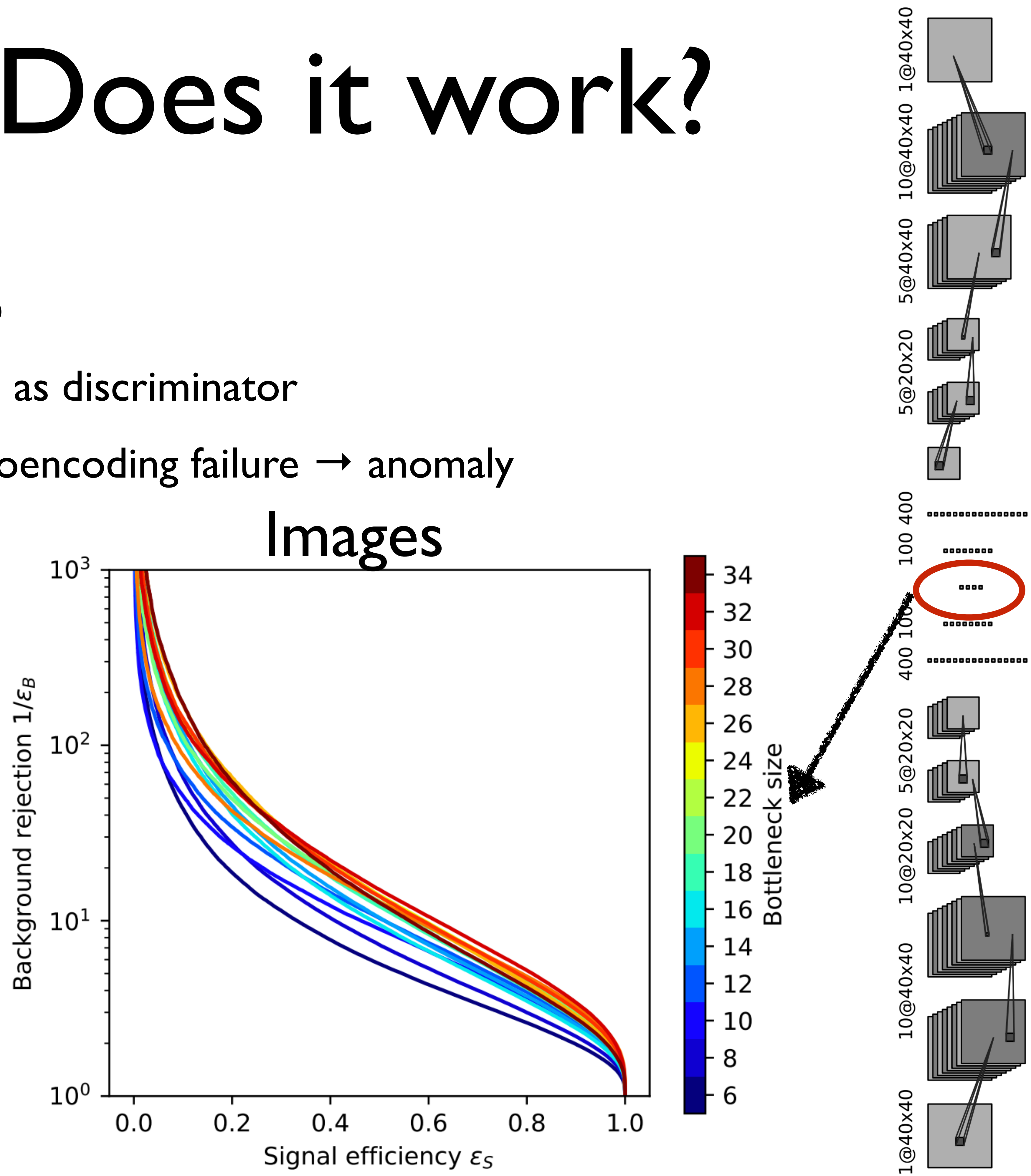


Convolutional network

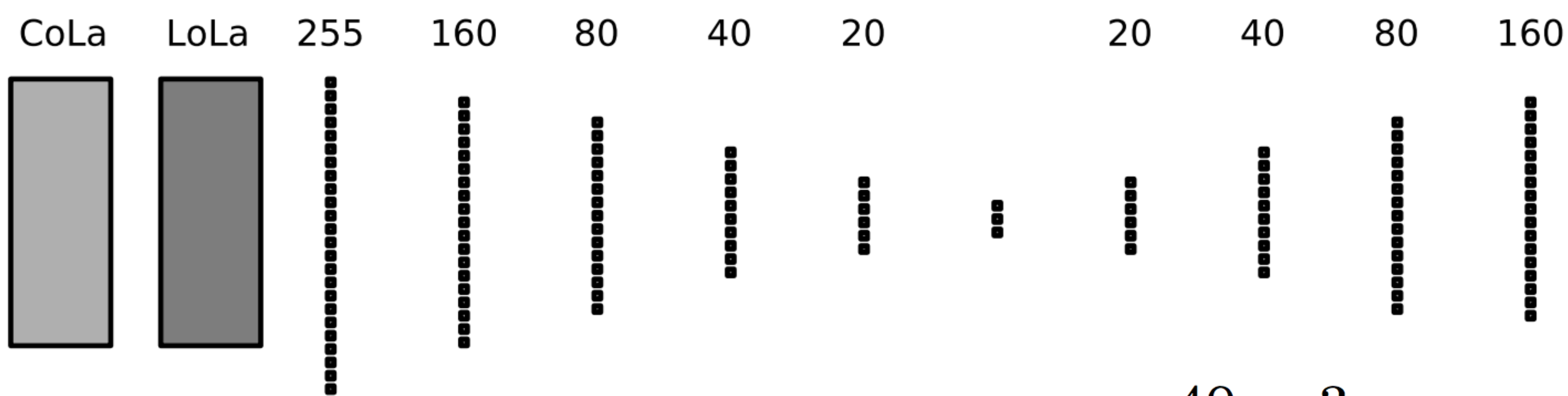
$$L_{\text{Auto}} = \sum_{\text{Pixels } ij} \left(X_{ij} - \widetilde{X}_{ij} \right)$$

Does it work?

- Train on QCD only
- Test on top vs QCD
- Cut on loss function as discriminator
  - Large loss → autoencoding failure → anomaly



Architecture II



*Constituent network**

$$L_{\text{auto}} = \sum_{j=1}^{40} \sum_{i=0}^{3} \left(\tilde{k}_{i,j}^{\text{in}} - \tilde{k}_{i,j}^{\text{auto}} \right)^2$$

$$k_{\mu,i} = \begin{pmatrix} E_0 & E_1 & \cdots & E_N \\ p_{x,0} & p_{x,1} & \cdots & p_{x,N} \\ p_{y,0} & p_{y,1} & \cdots & p_{y,N} \\ p_{z,0} & p_{z,1} & \cdots & p_{z,N} \end{pmatrix}$$

Combination Layer (**CoLa**): create linear combinations:

$$k_{\mu,i} \xrightarrow{\text{CoLa}} \widetilde{k}_{\mu,j} = k_{\mu,i}\, C_{ij}$$

Lorentz Layer (**LoLa**): Use resulting matrix to extract physics features.
Main assumption is the Minkowski metric

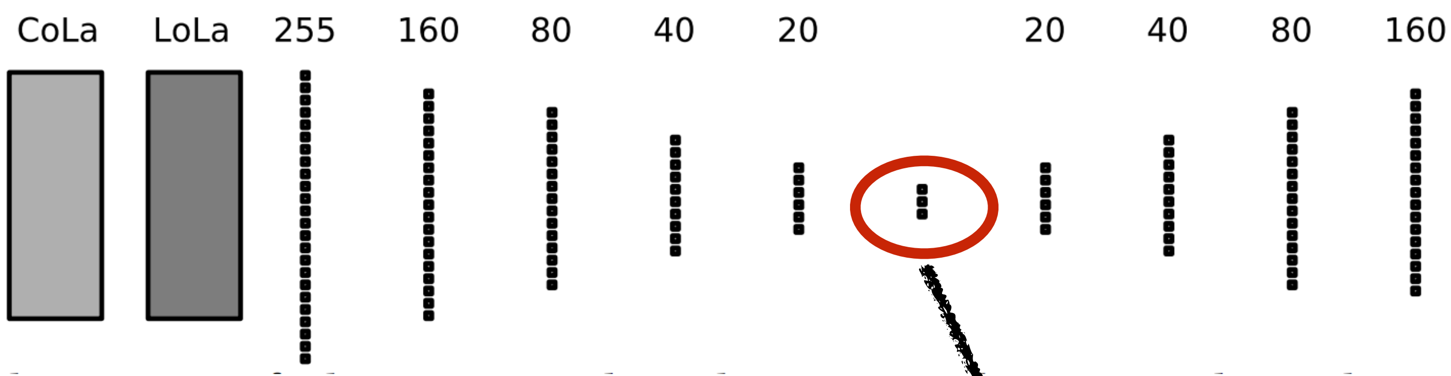
$$\widetilde{k}_{\mu,i} \to \sum_j (\widetilde{k}_i - \widetilde{k}_j)_\mu (\widetilde{k}_i - \widetilde{k}_j)_\nu \eta^{\mu\nu} B_{ij}$$

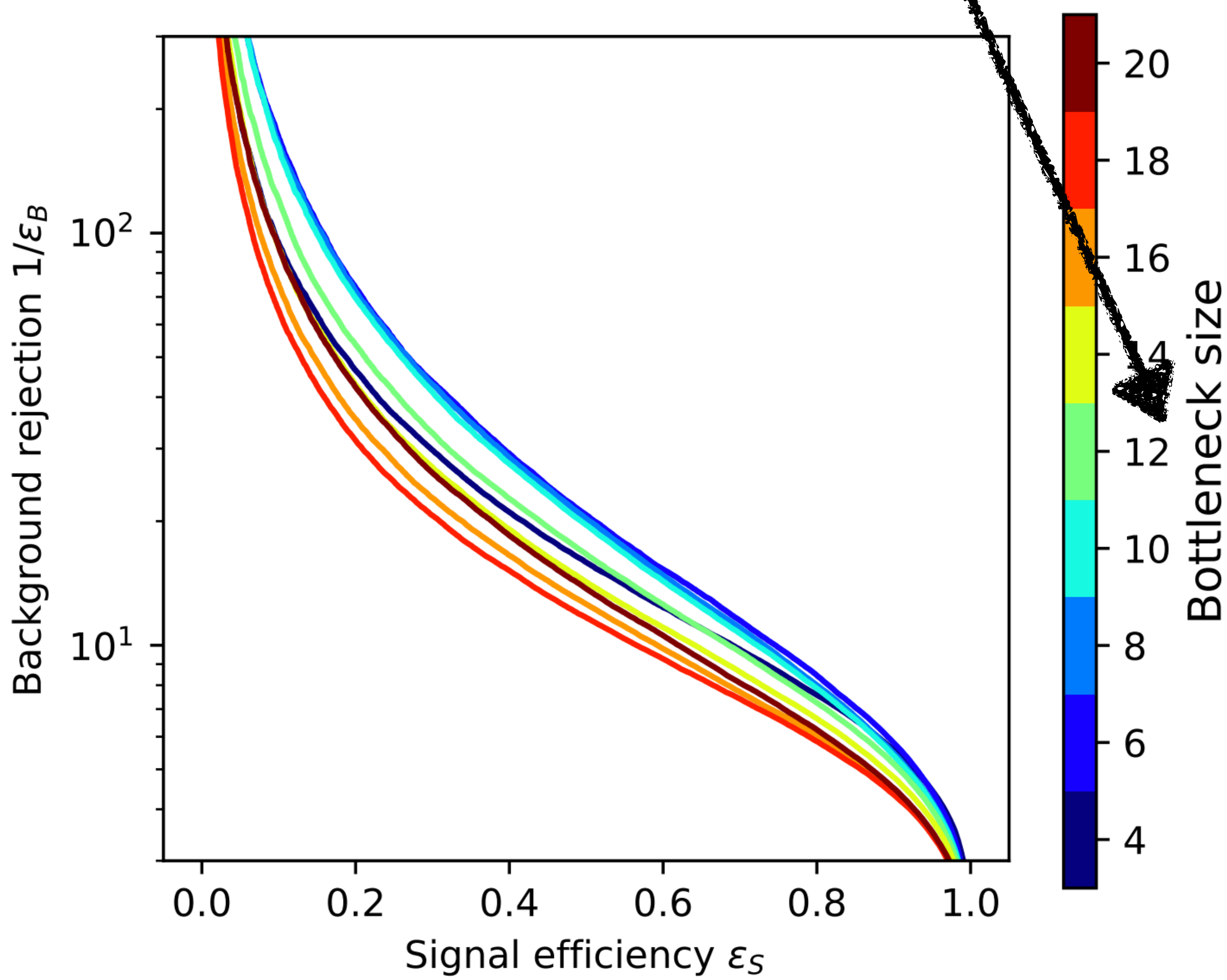
Can implement autoencoders using any architecture!

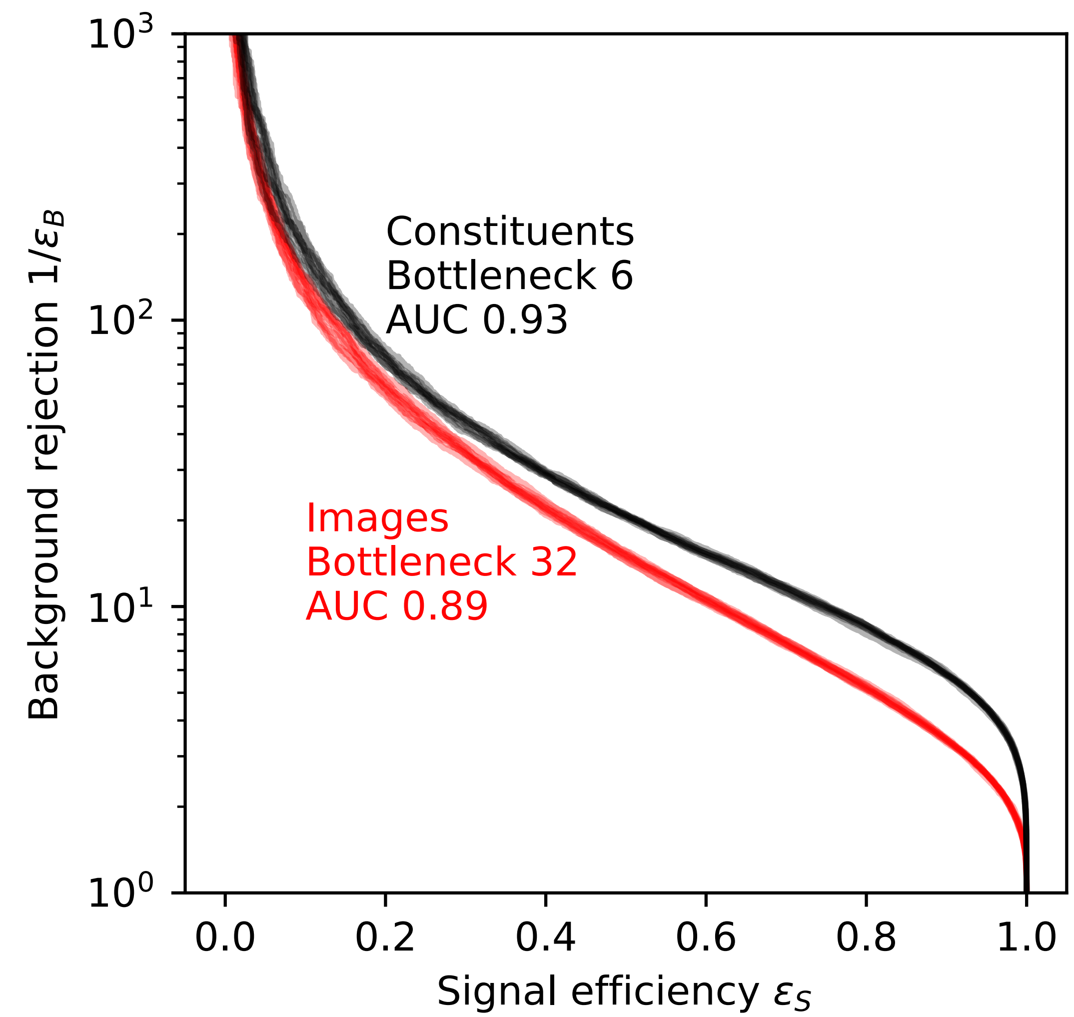
* from: *Deep-learning Top Taggers & No End to QCD*
A Butter, GK, T Plehn, M Russell
1707.08966

Does it work?

LoLa







Different ROC curve as well as bottleneck size preferred by different architectures