Deep Learning - Challenge Problem

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

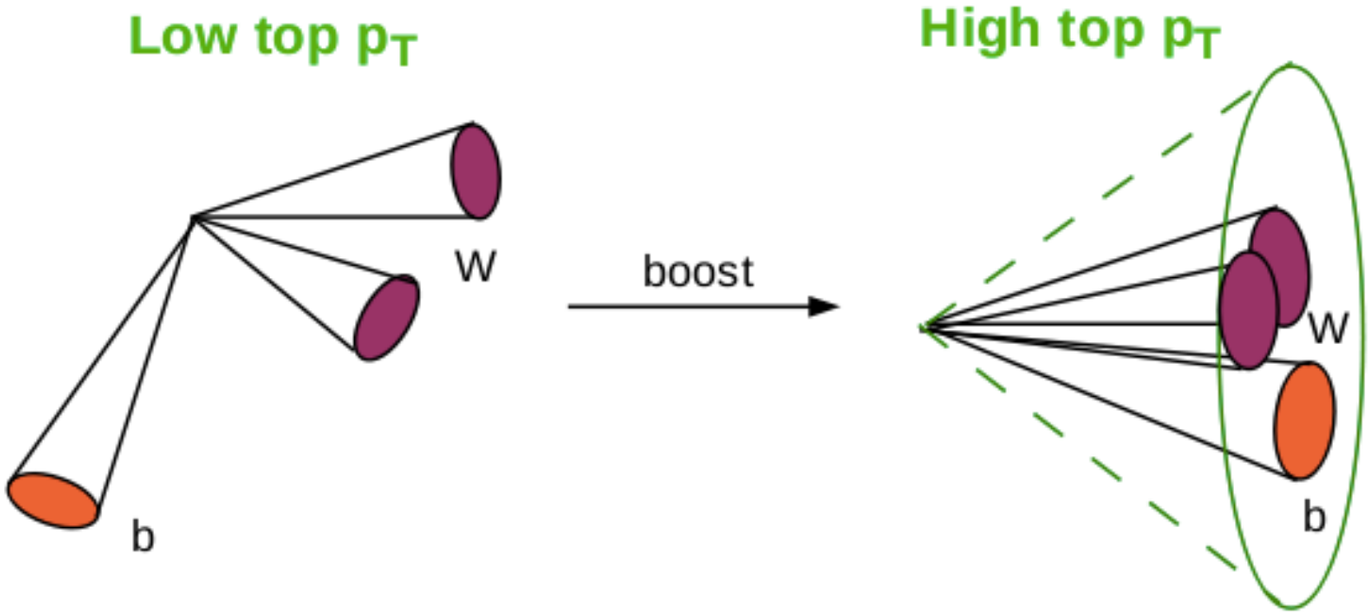
Zurich - ML4HEP Mini School
2019-02-04 - 2019-02-05

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Emmy Noether-Programm
Deutsche Forschungsgemeinschaft DFG

Bundesministerium für Bildung und Forschung

Heavy Resonance Tagging



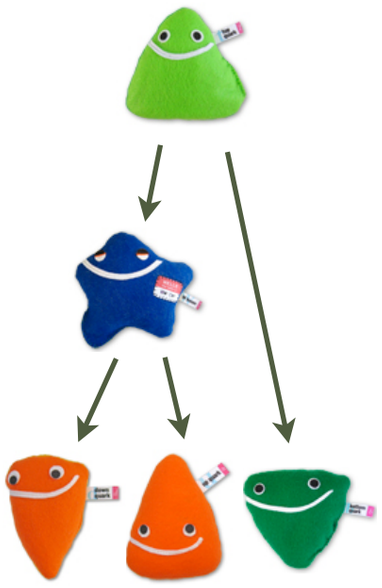
- Hadronically decaying top/Higgs/W/Z
- Contained in one (large-R) jet
- How to distinguish from light quark/gluon jets (and from each other)
- For new physics searches (and SM studies)

Some Classical solutions:
(aka jet substructure)

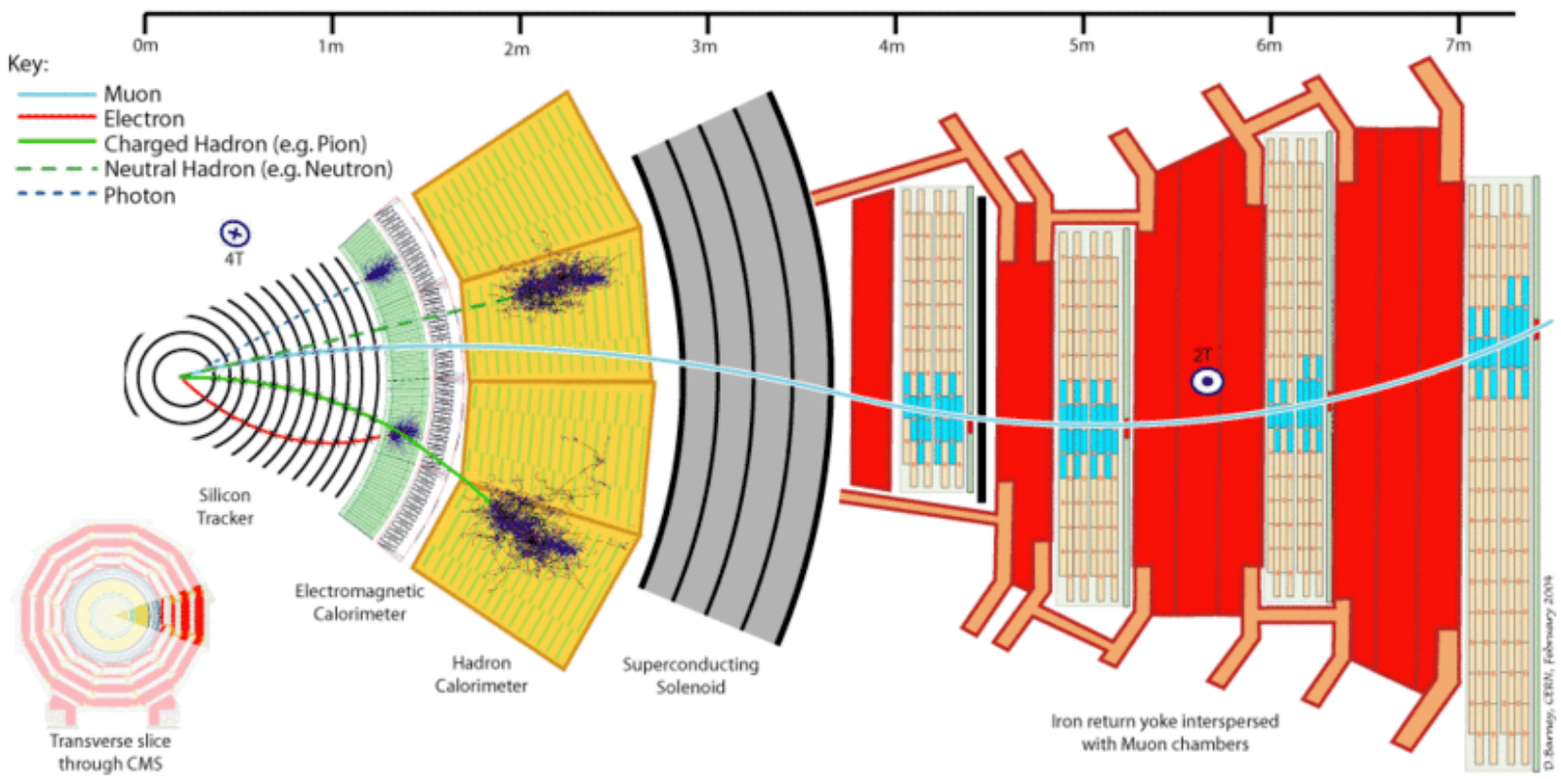
- Mass
  Calculate using a grooming algorithm (eg mMDT/softdrop or pruning)
- Centers of hard radiation
  n-subjettiness or energy correlation functions
- Flavour
  b tagging of large-R jets or subjets
- Soft substructure
  Color connection
- Inclusive reconstruction
  HEPTopTagger V2, HOTVR
- Other substructure variables
  Shower deconstruction, template tagger, …

Towards an Understanding of the Correlations in Jet Substructure
D Adams et al (BOOST 2013 Participants), Eur.Phys.J. C75
Top Tagging, T Plehn, M Spannowksy, J.Phys. G39 (2012) 083001
Boosted Top Tagging Method Overview, GK, Proc. Top2017

Top Quark



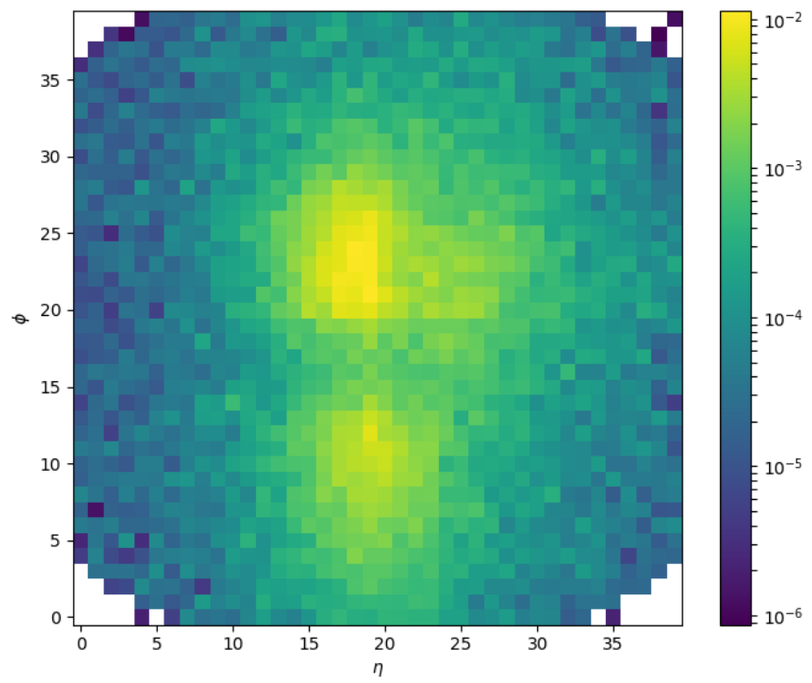
+



(jet images by C Daza)

- *Measure particle energies in calorimeter*
- *Reconstruct jet from individual measurements*
- *Image preprocessing*
  - *center, rotate, mirror, pixelate, trim, normalise*

=



1000 image average

Dataset Basics

- Goal: discriminate between hadronically decaying top quarks and QCD jets
    - Maximise area under ROC curve (AUC)
- Theory setting:
    - 14 TeV, hadronic tops for signal, qcd diets background, delphes ATLAS detector card with Pythia
    - No MPI/pile-up included
    - We cluster particle-flow entries (produced by Delphes E-flow) into anti-kT 0.8 jets in the pT range [550,650]
    - All top jets are matched to a parton-level top within $\Delta R = 0.8$, and to all top decay partons within 0.8
    - We also require |eta|_jet < 2

Version A: Constituents

- The leading 200 jet constituent four-momenta are stored, with zero-padding for jets with fewer than 200

- Constituents are sorted by pT, with the highest pT one first

- The truth top four-momentum is stored as truth_px etc.

- A flag (1 for top, 0 for QCD) is kept for each jet. It is called is_signal_new

- The variable "ttv" (= test/train/validation) is kept for each jet. It indicates to which dataset the jet belongs. It is redundant as the different sets are already distributed as different files.

- Either work with a fully connected architecture or try something else

Version B: Images

- 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

- Images after preprocessing of constituents
- Use a convolutional approach

Deep-learning Top Taggers or The End of QCD?
GK, Tilman Plehn, Michael Russell, Torben Schell
JHEP 05 (2017) 006
Pulling Out All the Tops with Computer Vision and Deep Learning
S Macaluso, D Shih, 1803.00107

Ingredients

- Dense Layer (Fully connected layer)
  `keras.layers.Dense(number_of_nodes, activation='relu')`

- Convolutional Layer (For image data, kernel_size=(2,2) corresponds to a 2x2 pixel large filter)
  `keras.layers.Conv2D(number_of_filters, kernel_size, padding='same', activation='relu')`

- Flatten Layer (turn image data into a list of numbers. Needed to go from Conv to Dense layers)
  `keras.layers.Flatten()`

- First layer:
  - Can be anything, but needs arguments `input_shape=(80,)` (constituents) or `input_shape=(40,40,1)` (images) as extra argument. Only the first layer should have the input shape argument.

- Last layer of your network:
  - `keras.layers.Dense(2, activation='softmax')`
  - (recommend to start with `activation='relu'` for all other layers)

Documentation at: https://keras.io/

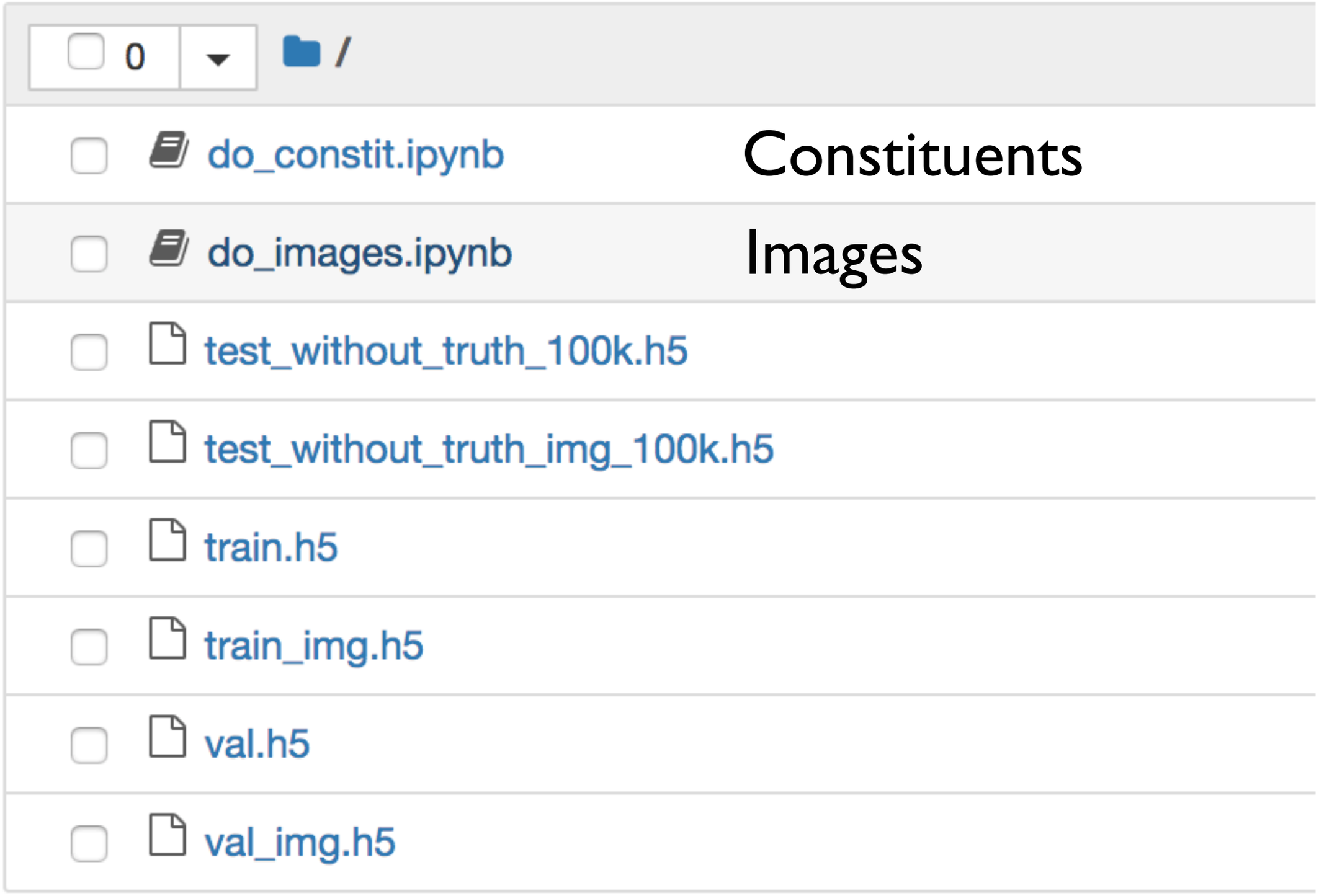
Some ideas

- Both versions:
    - more data, learning rate, optimiser algorithm, epochs, dropout, early stopping, **Smart use of val?**
- Version A (constituents)
    - more constituents, calculate physics quantities (mass?), other preprocessing (normalisation?), more layers, more nodes, …
- Version B (images)
    - More filters, filter size, more Conv layers, pooling, more dense layers/nodes, image processing, locally connected layers, ….

Getting Started

- Login to your AWS machine
  - ssh -i A1.pem -S ./tmp -L 8888:127.0.0.1:8888 ubuntu@YOUR_AWS_MACHINE
- `jupyter notebook`
- Connect to the notebook server
  - Look for Copy/paste this URL into your browser when you connect for the first time, to login with a token: http://localhost:8888/?token=90fdff46d8b41f570b5e547f338eb2e9f7481ef75d9b3ebd&token=90fdff46d8b41f570b5e547f338eb2e9f7481ef75d9b3ebd
- `mv .clng challenge`

Overview


0
/
do_constit.ipynb
Constituents
do_images.ipynb
Images
test_without_truth_100k.h5
test_without_truth_img_100k.h5
train.h5
train_img.h5
val.h5
val_img.h5

Data

Training
train.h5 and train_img.h5

1.2M labelled examples
Nominal sample

Validation
val.h5 and val_img.h5

20k labelled examples
Systematic applied: like “test”

Test
test_without_truth_100k.h5 and
test_without_truth_img_100k.h5

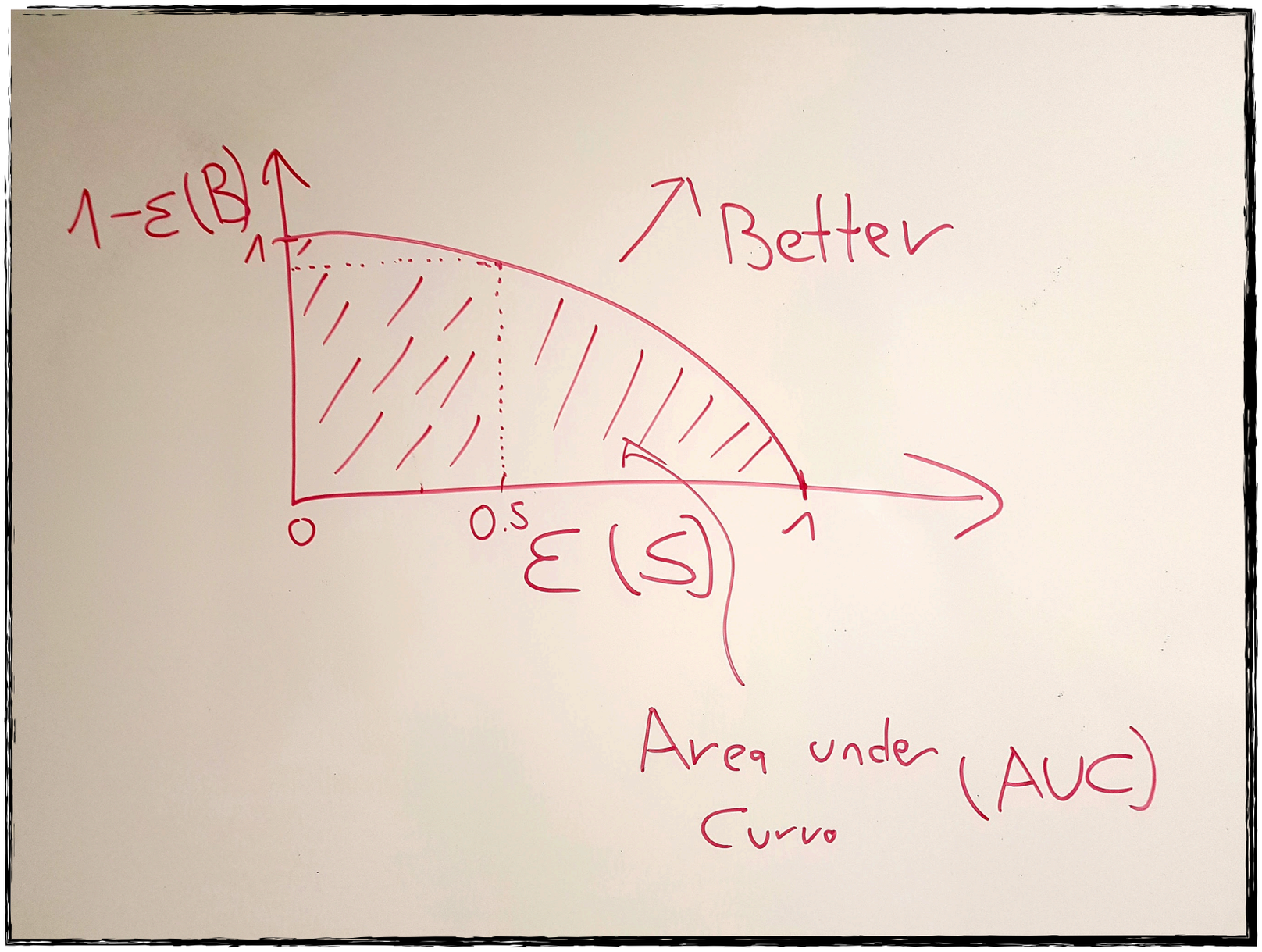
100k *unlabelled* examples
Systematic applied: like “real” data

- Based on
  - https://goo.gl/XGYju3
- But with a systematic twist (-; (Thanks to Sven!)

Your task

- Calculate probability to be signal for each jet in the test sample
- Make sure to *keep the ordering!*
- Use images or constituents as input - both are fine.
- Produce a zipped .npy file and upload to challenge page
- Winner ist best area-under-curve on subset of the test sample*
  (*we keep the right to veto solutions)
- Submissions close **today at 16:10**
  (Hamburg time)

Performance Measures



ROC:
Receiver operation characteristic

AUC:
Area under curve

Other caveats

- There will be a small symbolic price for the best submission
- Team work is encouraged, individual submissions are ok as well
- Fair-play: Don’t mess with other peoples machines, etc.

Way to submission

- Train network

- Apply to test sample

- Download result.zip file
- Rename to something meaningful:
  - kasieczka_v2.zip
  - **(please no funny characters, especially no space)**

- Upload to:
- https://desycloud.desy.de/index.php/s/AngokY8QNHBxy3k

- Leaderboard at:
  - https://github.com/gkasieczka/Challenge/blob/master/README.md

Score in Leaderbord is based on random 30% of data!
Use full sample for final ranking..