Generative–Invertible Networks for the LHC

Tilman Plehn

Universität Heidelberg

Transregio 6/2020
Machine Learning for LHC

LHC: fundamental understanding of big data

- data-driven, but all about QFT...
  1. precision predictions from first principles
  2. interpretation frameworks [SMEFT, SUSY]
  3. best use of the data
- 1991 visionaries: NN-based quark-gluon tagger

**USING NEURAL NETWORKS TO IDENTIFY JETS**

Leif LÖNNBLAD*, Carsten PETERSON** and Thorsteinn RÖGNVALDSSON***

*Department of Theoretical Physics, University of Lund, Sölvegatan 14A, S-22362 Lund, Sweden

Received 29 June 1990

A neural network method for identifying the ancestor of a hadron jet is presented. The idea is to find an efficient mapping between certain observed hadronic kinematical variables and the quark-gluon identity. This is done with a neuronic expansion in terms of a network of sigmoidal functions using a gradient descent procedure, where the errors are back-propagated through the network. With this method we are able to separate gluon from quark jets originating from Monte Carlo generated e^+e^- events with \( \sim 85\% \) approach. The result is independent of the MC model used. This approach for isolating the gluon jet is then used to study the so-called string effect.

In addition, heavy quarks (b and c) in e^+e^- reactions can be identified on the 50% level by just observing the hadrons. In particular we are able to separate b-quarks with an efficiency and purity, which is comparable with what is expected from vertex detectors. We also speculate on how the neural network method can be used to disentangle different hadronization schemes by compressing the dimensionality of the state space of hadrons.

⇒ not a question *if* experimentalists will use ML
The Machine Learning Landscape of Top Tagger


1 Institut für Experimentalphysik, Universität Hamburg, Germany
2 Institut für Theoretische Physik, Universität Heidelberg, Germany
3 Center for Cosmology and Particle Physics and Center for Data Science, NYU, USA
4 NHCT, Dept. of Physics and Astronomy, Rutgers, The State University of NJ, USA
5 Jozef Stefan Institute, Ljubljana, Slovenia
6 Theoretical Particle Physics and Cosmology, King’s College London, United Kingdom
7 Department of Physics, University of British Columbia, Canada
8 Department of Physics, University of California, Santa Barbara, USA
9 Faculty of Mathematics and Physics, University of Ljubljana, Ljubljana, Slovenia
10 Center for Theoretical Physics, MIT, Cambridge, USA
11 CP3, Université catholique de Louvain, Louvain-la-Neuve, Belgium
12 Physics Division, Lawrence Berkeley National Laboratory, Berkeley, USA
13 Simons Inst. for the Theory of Computing, University of California, Berkeley, USA
14 National Institute for Subatomic Physics (Nikhef), Amsterdam, Netherlands
15 LPTHE, CNRS & Sorbonne Université, Paris, France
16 III. Physikalisches Institut A, RWTH Aachen University, Germany

gregor.kasieczka@uni-hamburg.de
plehn@thi-heidelberg.de

July 24, 2019

Abstract

Based on the established task of identifying boosted, hadronically decaying top quarks, we compare a wide range of modern machine learning approaches. Unlike most established methods they rely on low-level input, for instance calorimeter output. While their network architectures are vastly different, their performance is comparatively similar. In general, we find that these new approaches are extremely powerful and great fun.

Content

1 Introduction
2 Data set
3 Tagger
3.1 Image-based taggers
3.1.1 CNN
3.1.2 ResNeXt
3.2 4-Vector-based taggers
3.2.1 TopoDNN
3.2.2 Multi-body N-subjetiness
3.2.3 TreeN
3.2.4 P-CNN
3.2.5 ParticleNet
3.3 Theory-inspired taggers
3.3.1 Lorentz Boost Network
3.3.2 Lorentz Layer
3.3.3 Latent Diligent Allocation
3.3.4 Energy Flow Polynomials
3.3.5 Energy Flow Networks
3.3.6 Particle Flow Networks
4 Comparison
5 Conclusion

References
(Theory) Networks beyond classification

Phase space networks

- **MC integration**  [Bendavit (2017)]
- **NNVegas**  [Klimek (2018), Carrazza (2020)]

Event generation

- **parton densities**  [NNPDF (since 2002)]
- **amplitudes**  [Bishara (2019), Badger (2020)]
- **neural importance sampling**  [Bothmann (2020)]
- **i-flow in SHERPA**  [Gao (2020)]

Generative networks

- **Jet Images**  [de Oliveira (2017), Carazza (2019)]
- **Unfolding**  [Datta (2018), Bellagente (2019), Bellagente (2020)]
- **Templates for QCD factorization**  [Lin (2019)]
- **Models**  [Erbin (2018), Otten (2018)]
- **Event subtraction**  [Butter (2019)]
Inspiration from art

**GANGogh**  [Bonafilia, Jones, Danyluk (2017)]

- can networks create new pieces of art?
- train on 80,000 pictures  [organized by style and genre]
- map noise vector to images
- generate flowers
Inspiration from art

GANGogh  [Bonafilia, Jones, Danyluk (2017)]

- can networks create new pieces of art?
- train on 80,000 pictures  [organized by style and genre]
- map noise vector to images
- generate portraits
Inspiration from art

**GANGogh**  [Bonafilia, Jones, Danyluk (2017)]
- can networks create new pieces of art?
- train on 80,000 pictures  [organized by style and genre]
- map noise vector to images

**Edmond de Belamy**  [Caselles-Dupre, Fautrel, Vernier]
- trained on 15,000 portraits
- sold for $432,500
⇒ all about marketing and sales
GAN basics

MC crucial for LHC physics

- goal: data-to-data with fundamental physics input only
- MC challenges
  - higher-order precision in bulk
  - coverage of tails
  - unfolding to access fundamental QCD
- neural network benefits
  - best available interpolation
  - structured latent space
  - lightning speed, once trained
  - inversion solved
  - training on MC and/or data, anything goes
- GANs the cool kid
  - generator trying to produce best events
  - discriminator trying to catch generator, competing towards equilibrium
- INNs the theory hope
  - flow networks specifying ways to linking spaces
  - invertible network the new tool
GAN algorithm

Example: LHC events

- training: true events \( \{x_T\} \) following \( p_T(x) \)
  - output: generated events \( \{r\} \rightarrow \{x_G\} \) following \( p_G(x) \)
- discriminator constructing \( D(x) \)  
  \[ L_D = \langle - \log D(x) \rangle_{x \sim p_T} + \langle - \log(1 - D(x)) \rangle_{x \sim p_G} \rightarrow -2 \log 0.5 \]
- generator producing truth-like events  
  \[ L_G = \langle - \log D(x) \rangle_{x \sim p_G} \]
- loss function evaluated over batch
- noise reduction/stabilization: gradient penalty  
  \[ \text{[alternatively WGAN]} \]

\Rightarrow \text{statistically independent copy of training events}
1– How to GAN LHC events

Idea: replace ME for hard process [Butter, TP, Winterhalder]

– medium-complex final state $t \bar{t} \rightarrow 6$ jets
  $t/\bar{t}$ and $W^\pm$ on-shell with BW $6 \times 4 = 18$ dof
  on-shell external states $\rightarrow 12$ dof [constants hard to learn]

– flat observables flat [phase space coverage okay]
1— How to GAN LHC events

Idea: replace ME for hard process  [Butter, TP, Winterhalder]

- medium-complex final state $t\bar{t} \rightarrow 6 \text{jets}$
  
  $t/\bar{t}$ and $W^\pm$ on-shell with $\text{BW} 6 \times 4 = 18 \text{ dof}$

  on-shell external states $\rightarrow 12 \text{ dof}$  [constants hard to learn]

- flat observables flat  [phase space coverage okay]

- direct observables with tails  [statistical error indicated]

- constructed observables similar
1– How to GAN LHC events

**Idea:** replace ME for hard process \cite{Butter, TP, Winterhalder}

- medium-complex final state $t\bar{t} \rightarrow 6$ jets
  
  $t/\bar{t}$ and $W^{\pm}$ on-shell with BW $6 \times 4 = 18$ dof
  
  on-shell external states $\rightarrow 12$ dof \[constants hard to learn\]

- flat observables flat \[phase space coverage okay\]

- direct observables with tails \[statistical error indicated\]

- constructed observables similar

- improved resolution \[1M training events\]
1– How to GAN LHC events

Idea: replace ME for hard process  [Butter, TP, Winterhalder]

- medium-complex final state $t\bar{t} \rightarrow 6$ jets
  $t/\bar{t}$ and $W^\pm$ on-shell with BW $6 \times 4 = 18$ dof
  on-shell external states $\rightarrow 12$ dof  [constants hard to learn]
- flat observables flat  [phase space coverage okay]
- direct observables with tails  [statistical error indicated]
- constructed observables similar
- improved resolution  [10M generated events]
1– How to GAN LHC events

Idea: replace ME for hard process  [Butter, TP, Winterhalder]

– medium-complex final state $t\bar{t} \rightarrow 6$ jets
  $t/\bar{t}$ and $W^\pm$ on-shell with BW $6 \times 4 = 18$ dof
  on-shell external states $\rightarrow 12$ dof  [constants hard to learn]
– flat observables flat  [phase space coverage okay]
– direct observables with tails  [statistical error indicated]
– constructed observables similar
– improved resolution  [50M generated events]
– concept promising
Intermediate resonances

GAN version of adaptive sampling

- generally 1D features
  - phase space boundaries
  - kinematic cuts
  - invariant masses \([\text{top}, \ W]\)

- batch-wise comparison of distributions, MMD loss with kernel \(k\)

\[
\text{MMD}^2 = \langle k(x, x') \rangle_{x, x' \sim P_T} + \langle k(y, y') \rangle_{y, y' \sim P_G} - 2 \langle k(x, y) \rangle_{x \sim P_T, y \sim P_G}
\]

\[
L_G \rightarrow L_G + \lambda_G \text{MMD}^2,
\]
Intermediate resonances

GAN version of adaptive sampling

- generally 1D features
  - phase space boundaries
  - kinematic cuts
  - invariant masses \([\text{top}, W]\)
- batch-wise comparison of distributions, MMD loss with kernel \(k\)
  \[
  \text{MMD}^2 = \langle k(x, x') \rangle_{x, x' \sim P_T} + \langle k(y, y') \rangle_{y, y' \sim P_G} - 2 \langle k(x, y) \rangle_{x \sim P_T, y \sim P_G}
  \]
  \[
  L_G \rightarrow L_G + \lambda_G \text{MMD}^2,
  \]

\[\Rightarrow\] minor impact of kernel function and width
2– How to GAN event subtraction

Idea: subtract event samples without bins [Butter, TP, Winterhalder]

- statistical uncertainty
  \[ \Delta_{B-S} = \sqrt{\Delta_B^2 + \Delta_S^2 \max(\Delta B, \Delta S)} \]

- applications in LHC physics
  - soft-collinear subtraction, multi-jet merging
  - on-shell subtraction
  - background/signal subtraction

- GAN setup
  1. differential, steep class label
  2. sample normalization
Subtracted events

How to beat statistics by subtracting

1— 1D toy example

\[ P_B(x) = \frac{1}{x} + 0.1 \quad P_S(x) = \frac{1}{x} \quad \Rightarrow \quad P_{B-S} = 0.1 \]

– statistical fluctuations reduced (sic!)
Subtracted events

How to beat statistics by subtracting

1— 1D toy example

\[ P_B(x) = \frac{1}{x} + 0.1 \quad P_S(x) = \frac{1}{x} \quad \Rightarrow \quad P_{B-S} = 0.1 \]

- statistical fluctuations reduced (sic!)

2— event-based background subtraction [weird notation, sorry]

\[ pp \rightarrow e^+ e^- \quad (B) \quad pp \rightarrow \gamma \rightarrow e^+ e^- \quad (S) \quad \Rightarrow \quad pp \rightarrow Z \rightarrow e^+ e^- \quad (B-S) \]
Subtracted events

How to beat statistics by subtracting

1— 1D toy example

\[ P_B(x) = \frac{1}{x} + 0.1 \quad P_S(x) = \frac{1}{x} \quad \Rightarrow \quad P_{B-S} = 0.1 \]

– statistical fluctuations reduced (sic!)

2— event-based background subtraction

\[ pp \rightarrow e^+ e^- \quad (B) \quad pp \rightarrow \gamma \rightarrow e^+ e^- \quad (S) \quad \Rightarrow \quad pp \rightarrow Z \rightarrow e^+ e^- \quad (B-S) \]

3— collinear subtraction

\[ pp \rightarrow Zg \quad (B: \text{matrix element}, \ S: \text{collinear approximation}) \]

⇒ applications in theory and analysis
3– How to GAN away detector effects

Bottom line from SFitter etc

- total rates lacking information
  STXS model-dependent
  unfolded distributions extremely convenient \([\bar{t}t\text{ results}]\)
- benefits
  access to hard matrix element/first-principles QCD
  matrix element method
- challenges
  non-invertible detector simulation
  model dependence

Grand goal: invert Markov processes \([\text{Bellagente, Butter, Kasiczka, TP, Winterhalder}]\)

- detector simulation typical Markov process
- inversion possible, in principle \([\text{entangled convolutions}]\)
- GAN task
  partons \(\xrightarrow{\text{DELPHES}}\) detector \(\xrightarrow{\text{GAN}}\) partons
  \(\Rightarrow\) full phase space unfolded
Standard GAN

Reconstructing the parton level

- \( pp \to ZW \to (\ell\ell) (jj) \)
- broad \( jj \) mass peak
- narrow \( \ell\ell \) mass peak
- modified 2 \( \to \) 2 kinematics
- fun phase space boundaries
- GAN same as event generation [with MMD]
Standard GAN

Reconstructing the parton level

- \( pp \rightarrow ZW \rightarrow (\ell\ell) (jj) \)
- broad \( jj \) mass peak
  narrow \( \ell\ell \) mass peak
- modified 2 \( \rightarrow \) 2 kinematics
  fun phase space boundaries
- GAN same as event generation \([\text{with MMD}]\)
- full inversion fine

\[
\begin{align*}
\text{GAN} & \quad \text{Truth} \\
1.0 & \quad 1.0 \quad 1.0 & \quad 1.0 \quad 1.0 \\
0.8 & \quad 0.8 \quad 0.8 & \quad 0.8 \quad 0.8 \\
0.6 & \quad 0.6 \quad 0.6 & \quad 0.6 \quad 0.6 \\
0.4 & \quad 0.4 \quad 0.4 & \quad 0.4 \quad 0.4 \\
0.2 & \quad 0.2 \quad 0.2 & \quad 0.2 \quad 0.2 \\
0.0 & \quad 0.0 \quad 0.0 & \quad 0.0 \quad 0.0
\end{align*}
\]

\[
\begin{align*}
\frac{1}{2}\frac{d\sigma}{d p_{T,j2}} & \quad [\text{GeV}^{-1}] \\
0.0 & \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \\
0.1 & \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \\
0.2 & \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \\
0.3 & \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \\
0.4 & \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \\
0.5 & \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \\
0.6 & \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \\
0.7 & \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \\
0.8 & \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \\
0.9 & \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \\
1.0 & \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0
\end{align*}
\]

\[
\begin{align*}
\frac{1}{2}\frac{d\sigma}{d m_{jj}} & \quad [\text{GeV}^{-1}] \\
0.0 & \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0 \\
0.1 & \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \\
0.2 & \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \quad 0.2 \\
0.3 & \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \quad 0.3 \\
0.4 & \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \quad 0.4 \\
0.5 & \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \quad 0.5 \\
0.6 & \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \quad 0.6 \\
0.7 & \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \quad 0.7 \\
0.8 & \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \quad 0.8 \\
0.9 & \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \quad 0.9 \\
1.0 & \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0
\end{align*}
\]
Standard GAN

Reconstructing the parton level

- \( pp \rightarrow ZW \rightarrow (\ell\ell) (jj) \)
- broad \( jj \) mass peak
  - narrow \( \ell\ell \) mass peak
  - modified \( 2 \rightarrow 2 \) kinematics
  - fun phase space boundaries
- GAN same as event generation \[\text{[with MMD]}\]
- full inversion fine
- **problem**: kinematics cuts in test data \[\{88\%, 38\% \text{ events}\]\n
\[
\begin{align*}
  p_T,j_1 & = 30 \ldots 100 \text{ GeV} \\
  p_T,j_1 & = 30 \ldots 60 \text{ GeV} \quad \text{and} \quad p_T,j_2 = 30 \ldots 50 \text{ GeV}
\end{align*}
\]
Fully conditional GAN

Adding more random sampling to network

- map random numbers to parton level
  hadron level as condition  [matched event pairs]
Fully conditional GAN

Adding more random sampling to network

– map random numbers to parton level
  hadron level as condition  [matched event pairs]

– full inversion fine  [again]
Fully conditional GAN

Adding more random sampling to network

- map random numbers to parton level
  hadron level as condition  [matched event pairs]
- full inversion fine  [again]
- tougher cuts challenging $m_{jj}$  [14\%, 39\% events, no interpolation, MMD not conditional]

\begin{align*}
  p_{T,j_1} &= 30 \ldots 50 \text{ GeV} & p_{T,j_2} &= 30 \ldots 40 \text{ GeV} & p_{T,\ell^-} &= 20 \ldots 50 \text{ GeV} \\
  p_{T,j_1} &> 60 \text{ GeV}
\end{align*}

\begin{align*}
  (12) & \quad & (13)
\end{align*}
Fully conditional GAN

Adding more random sampling to network

- map random numbers to parton level
  hadron level as condition [matched event pairs]
- full inversion fine [again]
- tougher cuts challenging $m_{jj}$ [14%, 39% events, no interpolation, MMD not conditional]

\[ p_{T,j_1} = 30 \ldots 50 \ \text{GeV} \quad p_{T,j_2} = 30 \ldots 40 \ \text{GeV} \quad p_{T,\ell^-} = 20 \ldots 50 \ \text{GeV} \quad (12) \]
\[ p_{T,j_1} > 60 \ \text{GeV} \quad (13) \]
- pretty pictures in 2D

⇒ 1. FCGAN unfolding at work
BSM injection

Different training (MC) and actual data... [not in v1, thank you to Ben Nachman]

...or model dependence of unfolding
...or localization in latent space

- train: SM events
test: 10% events with $W'$ in s-channel ⇒ any guesses?
BSM injection

Different training (MC) and actual data... [not in v1, thank you to Ben Nachman]

...or model dependence of unfolding

...or localization in latent space

- train: SM events
  test: 10% events with $W'$ in s-channel ⇒ any guesses?
4– Unfolding as inverting

**Invertible networks?** [Bellagente, Butter, Kasieczka, TP, Rousselot, Winterhalder, Ardizzone, Köthe]

- network as bijective transformation — normalizing flow
  Jacobian tractable — normalizing flow
  evaluation in both directions — INN [Ardizzone, Rother, Köthe]
- building block: coupling layer

\[ x_d \sim g(x_p) \quad \text{with} \quad \frac{\partial g(x_p)}{\partial x_p} = \begin{pmatrix} \text{diag } e^{s_2(x_p, 2)} & \text{finite} \\ 0 & \text{diag } e^{s_1(x_d, 1)} \end{pmatrix} \]

- eINN: padded by random numbers

\[
\begin{pmatrix} x_p \\ r_p \end{pmatrix} \xrightarrow{\text{PYTHIA, DELPHES: } g} \begin{pmatrix} x_d \\ r_d \end{pmatrix}
\]

\[
\begin{pmatrix} \tilde{x}_p, \tilde{r}_p \end{pmatrix} \xrightarrow{\text{unfolding: } \tilde{g}} \begin{pmatrix} \tilde{x}_d, \tilde{r}_d \end{pmatrix}
\]
4– Unfolding as inverting

Invertible networks? [Bellagente, Butter, Kasieczka, TP, Rousselot, Winterhalder, Ardizzone, K"othe]

- network as bijective transformation — normalizing flow
  Jacobian tractable — normalizing flow
  evaluation in both directions — INN [Ardizzone, Rother, K"othe]
- building block: coupling layer

\[ x_d \sim g(x_p) \quad \text{with} \quad \frac{\partial g(x_p)}{\partial x_p} = \begin{pmatrix} \text{diag } e^{s_2(x_p, 2)} \\ 0 \\ \text{diag } e^{s_1(x_d, 1)} \end{pmatrix} \]

- eINN: padded by random numbers

\[
\begin{pmatrix} x_p \\ r_p \end{pmatrix} \xrightarrow{\text{PYTHIA, DELPHES: } g} \xrightarrow{\text{unfolding: } \tilde{g}} \begin{pmatrix} x_d \\ r_d \end{pmatrix}
\]

\[ \Rightarrow \text{ proper sampling} \]
Conditional INN

Even more random sampling: conditional network

- same procedure as for GAN
- parton-level events from random numbers
Conditional INN

Even more random sampling: conditional network

- same procedure as for GAN
- parton-level events from random numbers
- calibration for statistical unfolding
Conditional INN

Even more random sampling: conditional network

- same procedure as for GAN
- parton-level events from random numbers
- calibration for statistical unfolding

Unfolding extra jets

- detector-level process $pp \rightarrow ZW + \text{jets}$ [variable number of objects]
- parton-level hard process chosen $2 \rightarrow 2$ [whatever you want]
- ME vs PS jets decided by network [including momentum conservation]

⇒ proper statistical inversion!
Outlook

Machine learning a great tool box

LHC physics is big data
jet classification was a starting point

generative networks exciting for theory
advantage 1: NN interpolation
advantage 2: training on MC and/or data
advantage 3: latent space structures
advantage 4: properly invertible

Any ideas for serious applications?
Dynamic MMD

Technical side-remark: dynamic MMD

- minimal input
  functional form of correlation \( m_{ij} \)
  kernel shape (irrelevant) and resolution
- Adaptive resolution?

Technical side-remark: dynamic MMD implementation

- multiple fixed-width kernels
- multiple kernels for conditional input
- cooling kernel [from SD of generator \( m_{ij} \)]

⇒ Technical implementation still open...
Superresolution GANs (preview)

Getting inspired  [Blecher, Butter, Keilbach, TP + Irvine]

- take high-resolution calorimeter images
down-sample to 1/8th 1D resolution
GAN inversion
- works because the GAN learn structure  [showers are QCD]
- start from low-resolution calorimeter images
GAN high-resolution images
- energy of constituents no.1,10,30

⇒ GANs are kind of magic