Machine Learning techniques for Monte Carlo generation

Anja Butter

ITP, Universität Heidelberg

arXiv:1907.03764, 1912.08824, 1912.00477, 2006.06685, and 2008.06545

with Armand Rousselot, Marco Bellagente, Sascha Diefenbacher, Gregor Kasieczka, Benjamin Nachman, Tilman Plehn, and Ramon Winterhalder

The HEP trinity

Theory

Fundamental Lagrangian

- Perturbative QFT

Standard Model vs. new physics

- Matrix elements, loop integrals

Experiment

Complex detector

- ATLAS, CMS, LHCb, ALICE, ...

Reconstruction of individual events

- Big data: jet images, tracks, ...

Precision simulations

First-principle Monte Carlo generators

- Simulation of parton/particle-level events
- Herwig, Pythia, Sherpa, Madgraph, ...

Detector simulation

- Geant4, PGS, Delphes, ...

$\Rightarrow$ **Unweighted event samples**

Neural networks for precision simulations

Problems in MC simulations

- Event generation:
  - High-dimensional phase space
  - Low unweighting efficiency
  - Higher order: exponential growth in computing time
- Highly complex full detector simulation $\rightarrow$ very slow
- HL-LHC: factor 25 in data
  - $\rightarrow$ reduction of statistical uncertainties by ~ 5
  - $\rightarrow$ need for percent level precision
- Limited resources: Precision vs. computing time

Advantages of neural networks

- Flexible parametrisation
- Interpolation properties
- Fast evaluation
- Multiple generative models: GAN, VAE, normalizing flow

Possibilities for ML in event generation

Event generation

- Generating 4-momenta
- $Z > ll$, $pp > jj$, $pp > t\bar{t}$+decay

[1901.00875] Otten et al. **VAE** & **GAN**

[1901.05282] Hashemi et al. **GAN**

[1903.02433] Di Sipio et al. **GAN**

[1903.02556] Lin et al. **GAN**

[1907.03764, 1912.08824] Butter et al. **GAN**

[1912.02748] Martinez et al. **GAN**

[2001.11103] Alanazi et al. **GAN**

Monte Carlo integration

- Estimating matrix element
- Neural importance sampling

[1707.00028] Bendavid, **Regression** & **GAN**

[1810.11509] Klimek and Perelstein

[1912.11055] Bishara and Montull **Regression**

[2001.05478] Bothmann et al. **NF**

[2001.05486, 2001.10028] Gao et al. **NF**

[2002.07516] Badger and Bullock **Regression**

Detector simulation

- Jet images
- Fast shower simulation in calorimeters

[1701.05927] de Oliveira et al. **GAN**

[1705.02355, 1712.10321] Paganini et al. **GAN**

[1802.03325, 1807.01954] Erdmann et al. **GAN**

[1805.00850] Musella et al. **GAN**

[ATL-SOFT-PUB-2018-001, ATL-SOFT-PROC-2019-007] ATLAS **VAE** & **GAN**

[1909.01359] Carazza and Dreyer **GAN**

[2005.05334] Buhmann et al. **VAE**

Unfolding

- Detector to parton/particle level distributions

[1806.00433] Datta et al. **GAN**

[1911.09107] Andreassen et al.

[1912.0047] Bellagente et al. **GAN**

[2006.06685] Bellagente et al. **NF**

NO claim to completeness! *Review: Generative Networks for LHC events [2008.08558]*

Boosting standard event generation...

1. Generate phase space points

2. Calculate event weight

$$w_{event} = \boxed{f(x_1, Q^2) f(x_2, Q^2)} \times \boxed{\mathcal{M}(x_1, x_2, p_1, \ldots p_n)} \times \boxed{J(p_i(r))^{-1}}$$

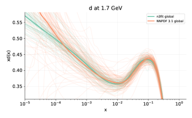
3. Unweighting via importance sampling
   $\rightarrow$ optimal for $w \approx 1$

$$w_{event} = \underbrace{f(x_1, Q^2) f(x_2, Q^2)}_{\text{PDF}} \times \overbrace{\mathcal{M}(x_1, x_2, p_1, \ldots p_n)}^{\text{Matrix element}} \times \underbrace{J(p_i(r))^{-1}}_{\text{Phase space mapping}}$$

Boosting standard event generation...

Matrix element

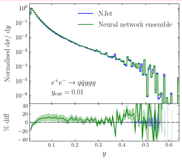
$$w_{event} = f(x_1, Q^2) f(x_2, Q^2) \times \mathcal{M}(x_1, x_2, p_1, \dots p_n) \times J(p_i(r))^{-1}$$



- NNPDF since 2002(!)
- genetic algorithm
- n3fit: determ. gradient descent
- S. Carrazza, J. Cruz-Martinez [1907.05075]

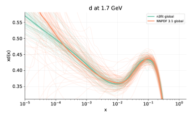
Phase space mapping

Boosting standard event generation...



- regression network
- learn amplitude or K factor
- S. Badger, J. Bullock [2002.07516]
- J. Bendavid [1707.00028]

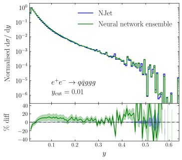
$$w_{event} = \boxed{f(x_1, Q^2) f(x_2, Q^2)} \times \boxed{\mathcal{M}(x_1, x_2, p_1, \ldots p_n)} \times \boxed{J(p_i(r))^{-1}}$$



- NNPDF since 2002(!)
- genetic algorithm
- n3fit: determ. gradient descent
- S. Carrazza, J. Cruz-Martinez [1907.05075]

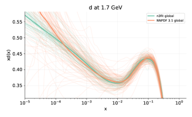
Phase space mapping

Boosting standard event generation...

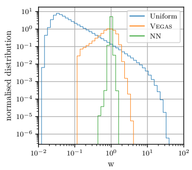


- regression network
- learn amplitude or K factor
- S. Badger, J. Bullock [2002.07516]
- J. Bendavid [1707.00028]

$$w_{event} = f(x_1, Q^2) f(x_2, Q^2) \times \mathcal{M}(x_1, x_2, p_1, \dots p_n) \times J(p_i(r))^{-1}$$



- NNPDF since 2002(!)
- genetic algorithm
- n3fit: determ. gradient descent
- S. Carrazza, J. Cruz-Martinez [1907.05075]



- learn efficient phase space mapping ($\to w \approx 1$)
- normalizing flow
- Gao et al. [2001.10028]
- Bothmann et al. [2001.05478]

… or train generative network directly on events

Possible gains:

- Generate more statistics
- Use network to sample phase space (similar to NF)
- Ship model instead of samples (profit from fast evaluation of NN)
- Replace fast detector simulations
- Use conditional generative network to interpolate between measured samples
- Use invertible architectures for unfolding

$\rightarrow$ Evaluate performance

A generative model

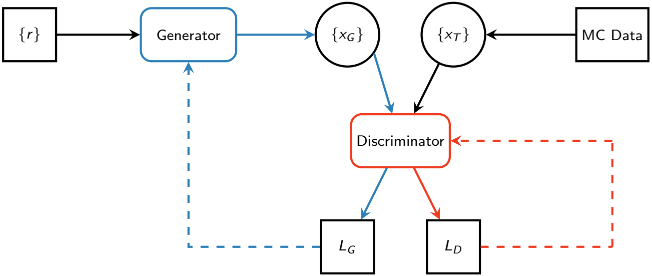
- Generative Adversarial Networks (GAN)
- Training data: true events $\{x_T\}$
  Output data: generated events $\{x_G\}$
- **Discriminator** distinguishes $\{x_T\}, \{x_G\}$ $[D(x_T) \to 1,\ D(x_G) \to 0]$

$$L_D = \big\langle -\log D(x) \big\rangle_{x \sim P_T} + \big\langle -\log(1 - D(x)) \big\rangle_{x \sim P_G} \xrightarrow{D(x) \to 0.5} -2 \log 0.5$$

- **Generator** fools discriminator $[D(x_G) \to 1]$

$$L_G = \big\langle -\log D(x) \big\rangle_{x \sim P_G}$$

$\Rightarrow$ **New statistically independent samples**



Why GANs? Features, problems and solutions

+ Generate better samples than VAE
+ Large community working on GANs

- Unstable training

Solutions

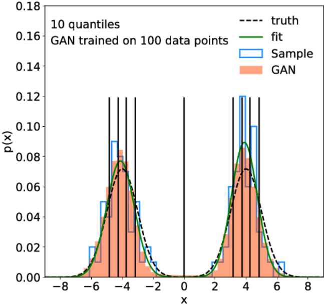
- Regularization of the discriminator, eg. gradient penalty [Ghosh, Butter et al., ...]
- Modified training objective:
  - Wasserstein GAN (incl. gradient penalty) [Lin et al., Erdmann et al., ...]
  - Least square GAN (LSGAN) [Martinez et al., ...]
  - MMD-GAN [Otten et al., ...]
  - MSGAN [Datta et al., ...]
  - Cycle GAN [Carazza et al., ...]
- Use of symmetries [Hashemi et al., ...]
- Whitening of data [Di Sipio et al., ...]
- Feature augmentation [Alanazi et al., ...]

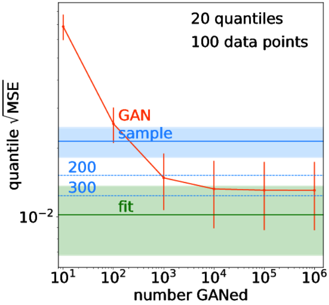
What is the statistical value of GANned events?

- Example: 1D camel function
- Compare Sample vs. GAN vs. 5 param.-fit (mean, width, relative height)
- Evaluation on quantiles

$$\mathrm{MSE} = \frac{1}{N_{\mathrm{quant}}} \sum_{j=1}^{N_{\mathrm{quant}}} \left(x_j - \frac{1}{N_{\mathrm{quant}}} \right)^2$$

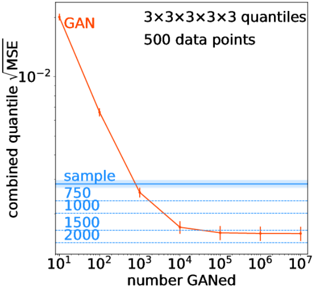
- Convergence to amplification factor 2.5 for 20 quantiles

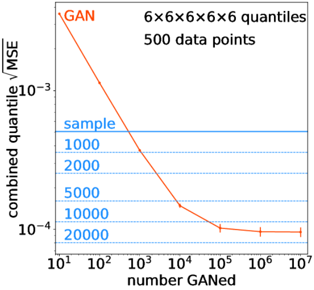




What is the statistical value of GANned events?

- Example: Sphere in 5 dimensions
- Sum over 5-dimensional quantiles
- Amplification factor increases for sparse quantiles:
  - 3 for 3^5 quantiles
  - 15 for 6^5 quantiles

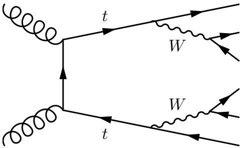


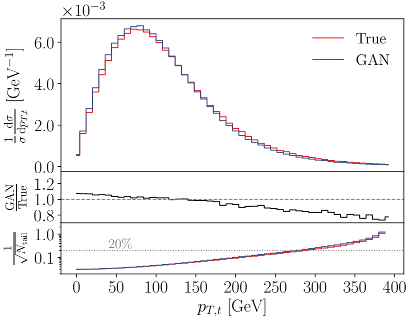


How to GAN LHC events

Idea: generate hard process

- Realistic LHC final state $t\bar{t} \to 6$ jets [1907.03764]
- 18 dim output [fix external mass, no mom. cons.]
- Flat observables precise
- Systematic undershoot in tails [10-20% deviation]



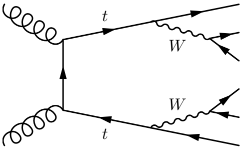


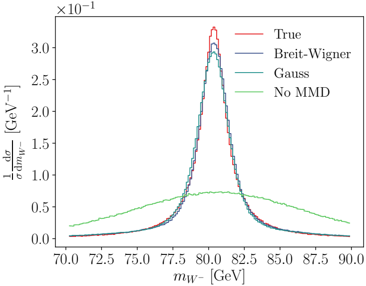
How to GAN LHC events

Idea: generate hard process

- Realistic LHC final state $t\bar{t} \to 6$ jets [1907.03764]
- 18 dim output
- Flat observables precise
- Systematic undershoot in tails [10-20% deviation]
- Sharp phase-space structures, not using Γ_W

$$\mathrm{MMD}^2(P_T, P_G) = \big\langle k(x, x')\big\rangle_{x,x' \sim P_T} + \big\langle k(y, y')\big\rangle_{y,y' \sim P_G} - 2\big\langle k(x, y)\big\rangle_{x \sim P_T, y \sim P_G}$$

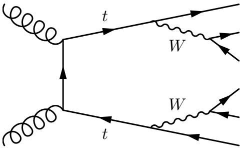


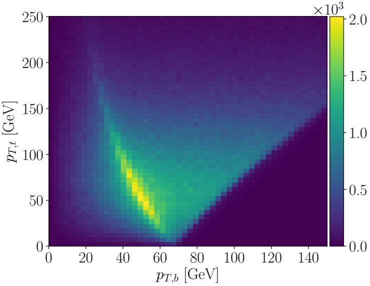


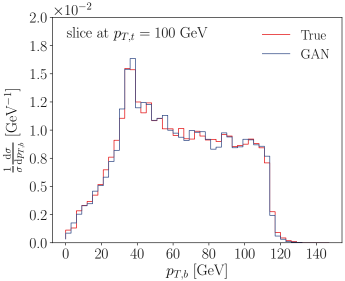
How to GAN LHC events

Idea: generate hard process

- Realistic LHC final state $t\bar{t} \to 6$ jets [1907.03764]
- 18 dim output
- Flat observables precise
- Systematic undershoot in tails [10-20% deviation]
- Sharp phase-space structures, not using Γ_W [MMD-loss]
- 2D correlations







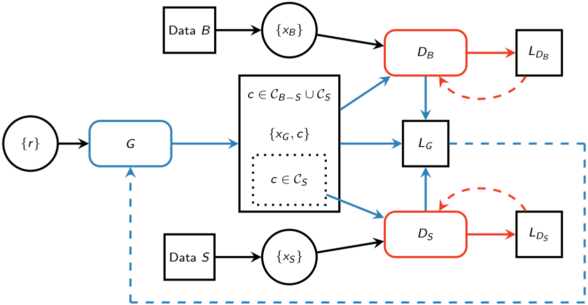
How to GAN event subtraction

Idea: sample based subtraction of distributions [1912.08824]

1. Consistent multidimensional difference between two distributions
2. Beat bin-induced statistical uncertainty [interpolation of distributions]

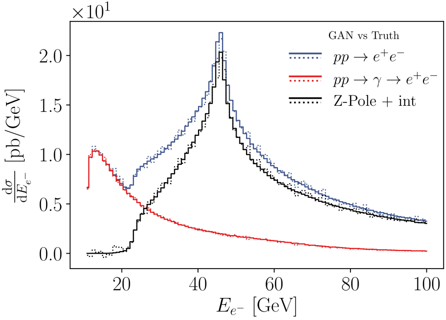
$$\Delta_{B-S} = \sqrt{n_B^2 N_B + n_S^2 N_S} > \max(\Delta_B, \Delta_S)$$

- Many applications:
  - Soft-collinear subtraction, multi-jet merging, on-shell subtraction
  - Background subtraction [4-body decays $\longrightarrow$ preserves correlations]



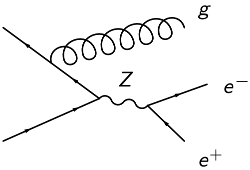
Example I: Z pole

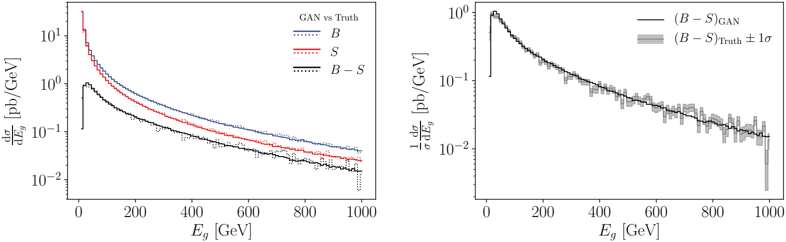
- Training data:
  - $pp \to e^+e^-$
  - $pp \to \gamma \to e^+e^-$
  - 1 M events per dataset, MadGraph5
- Generated events: Z-Pole + interference



Example II: Dipole subtraction

- Theory uncertainties $\rightarrow$ limiting factor for HL-LHC
- Higher order: Subtract diverging Catany Seymour Dipole from real emission term
- 1 M events per dataset, SHERPA





How to GAN away detector effects

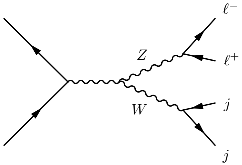
Idea: invert Markov process [1912.00477]

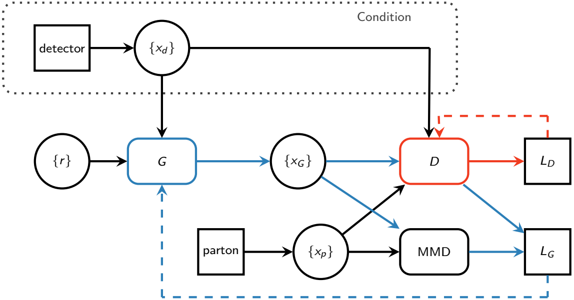
Detector simulation

- Typical Markov process
- Prior dependent inversion possible [Datta et al.]
- Aim: unfolding multidimensional phase space

Reconstruct parton level $pp \to ZW \to (ll)(jj)$

- GAN: no connection between input and discr.
  $\to$ use **fully conditional** GAN (FCGAN)



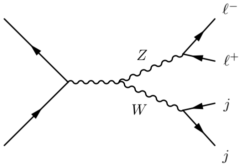


How to GAN away detector effects

Idea: invert Markov process [1912.00477]

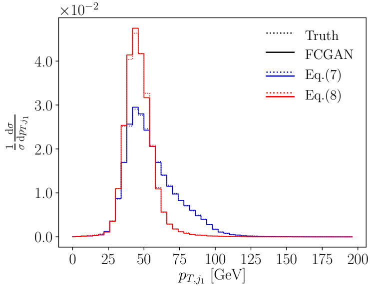
Reconstruct parton level $pp \to ZW \to (ll)(jj)$

- Use **fully conditional** GAN (FCGAN)
- Inversion works ✓



$$\text{Eq.(7)}: \quad p_{T,j_1} = 30 \ldots 100 \text{ GeV} \quad (\sim 88\%)$$

$$\text{Eq.(8)}: \quad p_{T,j_1} = 30 \ldots 60 \text{ GeV} \quad \text{and} \quad p_{T,j_2} = 30 \ldots 50 \text{ GeV} \quad (\sim 38\%)$$

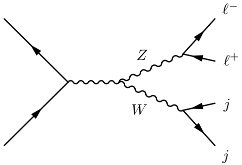


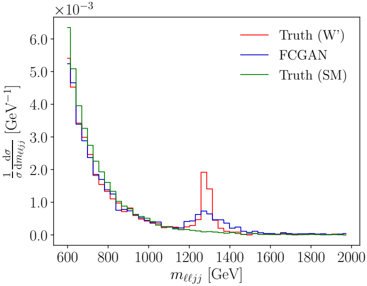
How to GAN away detector effects

Idea: invert Markov process [1912.00477]

Reconstruct parton level $pp \to ZW \to (ll)(jj)$

- Use **fully conditional** GAN (FCGAN)
- Inversion works ✓
- BSM injection ✓
  - train: SM events
  - test: 10% events with W' in s-channel

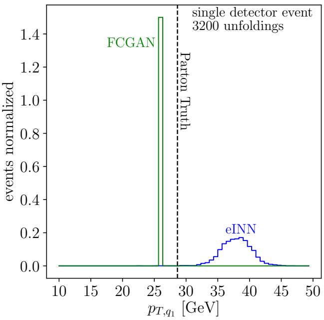


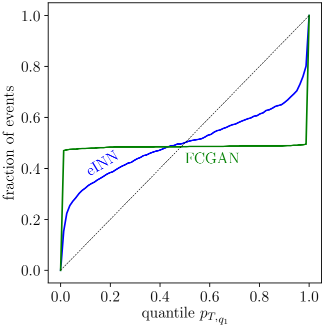


Curing shortcomings with invertible structure

- cGAN calibration curves: mean correct, distribution too narrow
- INN: Normalizing flow with fast evaluation in both directions

$$\begin{pmatrix} x_p \\ r_p \end{pmatrix} \xleftrightarrow[\leftarrow \text{ unfolding:}\bar{g}]{\text{PYTHIA,DELPHES:}g\rightarrow} \begin{pmatrix} x_d \\ r_d \end{pmatrix}$$



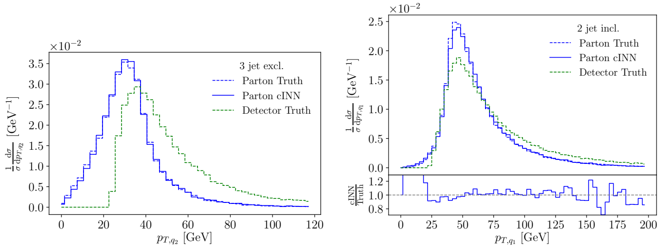


Conditional invertible neural networks

Condition INN on detector data [2006.06685]

$$x_p \xleftrightarrow[\leftarrow \text{ unfolding: } \bar{g}(r, {\color{red}f(x_d)})]{g(x_p, {\color{red}f(x_d)}) \rightarrow} r$$

- Use detector information x_d of arbitrary dimension
- Cross check 2/3/4 jet exclusive channels



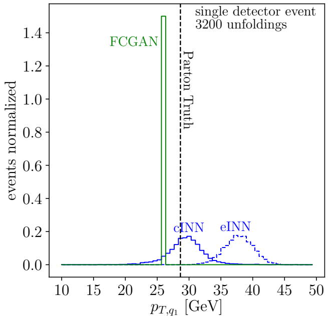
Conditional invertible neural networks

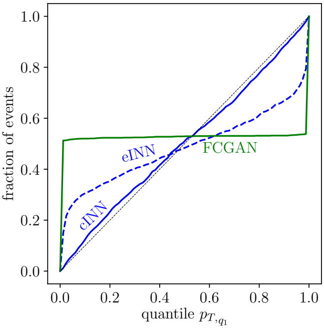
Condition INN on detector data [2006.06685]

$$x_p \xleftrightarrow[\leftarrow \text{ unfolding: } \bar{g}(r, {\color{red}f(x_d)})]{g(x_p, {\color{red}f(x_d)})\rightarrow} r$$

Minimizing $L = \left\langle \frac{||g(x_p, f(x_d))||_2^2}{2} - \log \left| \frac{\partial g(x_p, f(x_d))}{\partial x_p} \right| \right\rangle_{x_p \sim P_p, x_d \sim P_d} - \log p(\theta)$

$\rightarrow$ calibrated parton level distributions





Summary

- We can boost standard event generation using ML

- GANs can learn underlying distributions from event samples
- Possibilities to stabilize GAN training: gradient penalty, WGAN-GP, LSGAN,...
- MMD improves performance for special features

- Successful sample based subtraction implemented
- Applications: background subtraction, soft-collinear subtraction, . . .

- Unfold high-dimensional detector level distributions with cGANs and INN
- Stable under insertion of new data, proper calibration achieved by cINN

Important next steps

1. Quantify uncertainties (eg. Bayesian networks)
   - including correlations
2. High precision
3. Automization
   - move away from hand engineered networks