# Application of Generative Adversarial Networks to electron-proton scattering

Yasir Alanazi (ODU), Pawel Ambrozewicz (JLab), Michelle Kuchera (Davidson Coll.), Yaohang Li (ODU), Tianbo Liu (JLab), R. Evan McClellan (JLab), Wally Melnitchouk (JLab), E. Pritchard (Davidson Coll.), Raghu Ramanujan (Davidson Coll.), M. Robertson (Davidson Coll.), Nobuo Sato (JLab), R. Strauss (Davidson Coll.), Luisa Velasco (U. Dallas)

# Table of Contents

1. Motivation
2. Architecture
3. Results
4. Conclusions

# Motivation

Any experiment requires a detailed Monte Carlo simulation for at least two reasons:

- **Before:** Assessment of capabilities of the experimental apparatus,
- **After:** Analysis of the experimental data and extraction of physics the experiment was set out to measure.

The latter proceeds through 3 main stages:

- a vertex level event generation,
- propagation through an experimental setup,
- an event reconstruction at a detector level.

# Motivation

Modern Monte Carlo event generators, based on pseudo random number generators, have some drawbacks:

- They rely on theoretical models and their assumptions,
- Parameters are tuned to suit a particular experimental task,
- Processing an event, depending on the complexity of the setup, is mostly time consuming.
- Any physics analysis operates on sets of events satisfying certain criteria, collected into histograms, so all corrections are done at the histogram level.

# Motivation

Using Deep Learning in event generation provides solutions to some of those issues. It allows to:

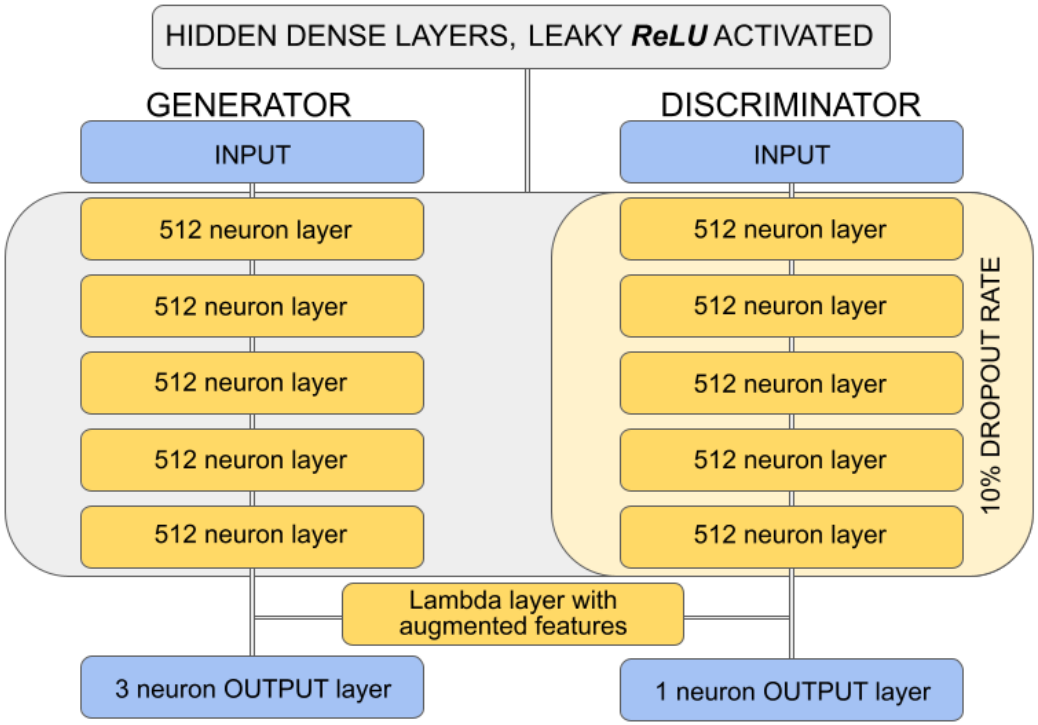
- create a theory agnostic event generator,
- generate efficiently events, although the training might be long,
- compactify experimental data - the size can be reduced from large number gigabytes to megabytes.

Most suitable approach is to use Generative Adversarial Networks (GANs), where two separate networks, a generator and a discriminator, compete against each other. The former improves the quality of fake events while the latter improves its ability to distinguish the fake from the real events.

# Architecture

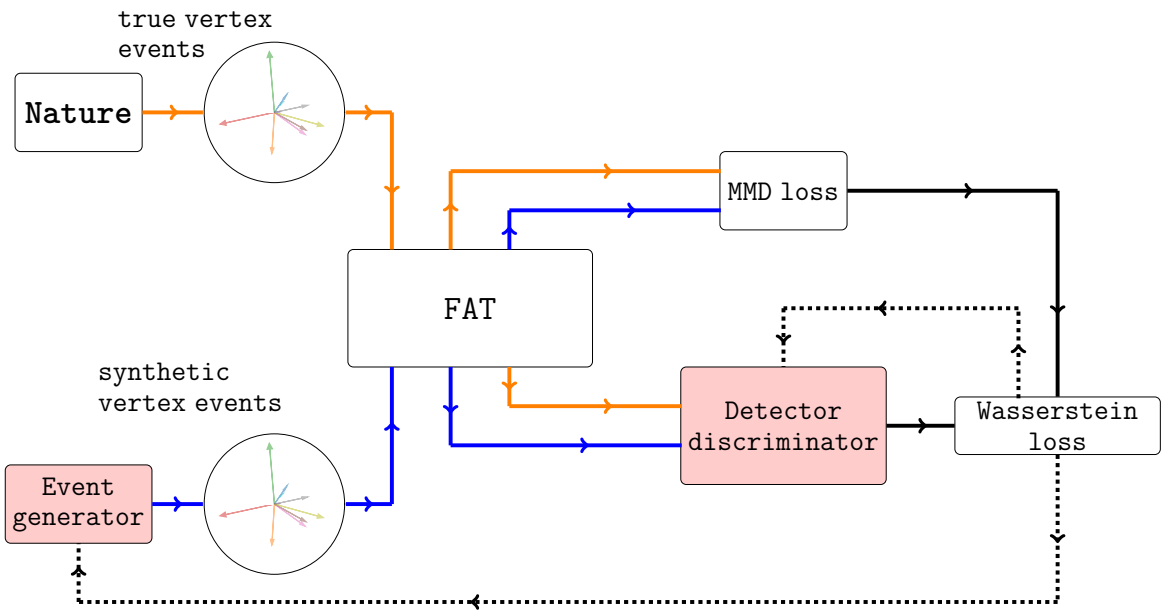
The FAT-GAN architecture consists of:

- The generator,
  *uses Wasserstein loss with gradient penalty,*
- **F**eature **A**ugmentation and **T**ransformation,
- The discriminator,
  *uses a combination of Wasserstein and Maximum Mean Discrepancy loss.*



# Architecture

## Full FAT-GAN Architecture

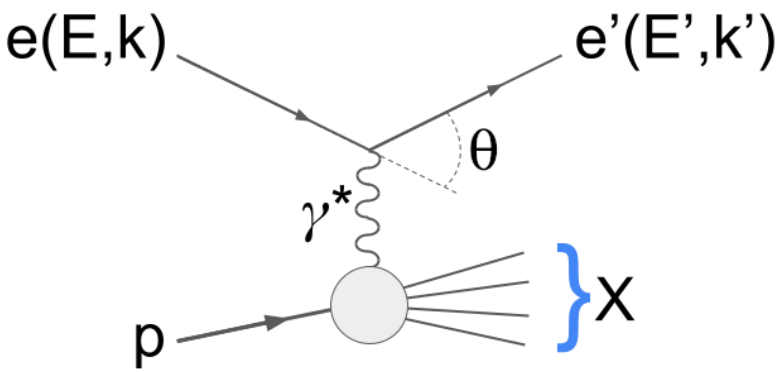



- load training samples x (real data)
- generate noise z (e.g. Gaussian)
- compute augmented features for x and z
- transform z into fake data G(z) with GAN
- evaluate real and fake data with discriminator D: D(x) and D(G(z))
- evaluate loss by subtracting the mean of the fake data discriminator from the mean of the real data discriminator
- update discriminator weights by taking the gradient of that computed loss w.r.t the previous discriminator weights

# Results

## Example: Inclusive Deep Inelastic Scattering (DIS)

Kinematics Description




Features used in the GAN training,

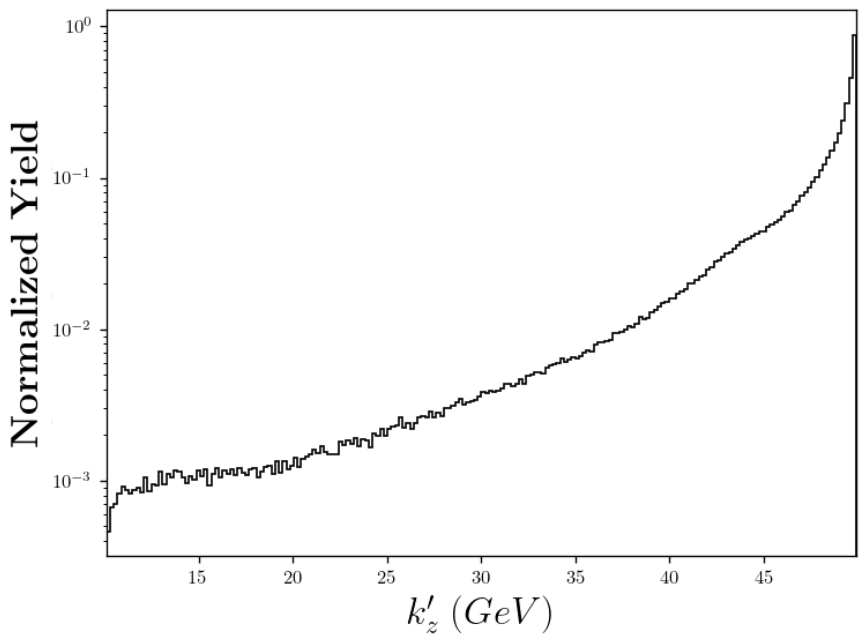
- the transverse momentum components of the scattered electron, $k'_x$, $k'_y$
- the longitudinal momentum of the scattered electron, $k'_z$, transformed to $\log(E - k'_z)$

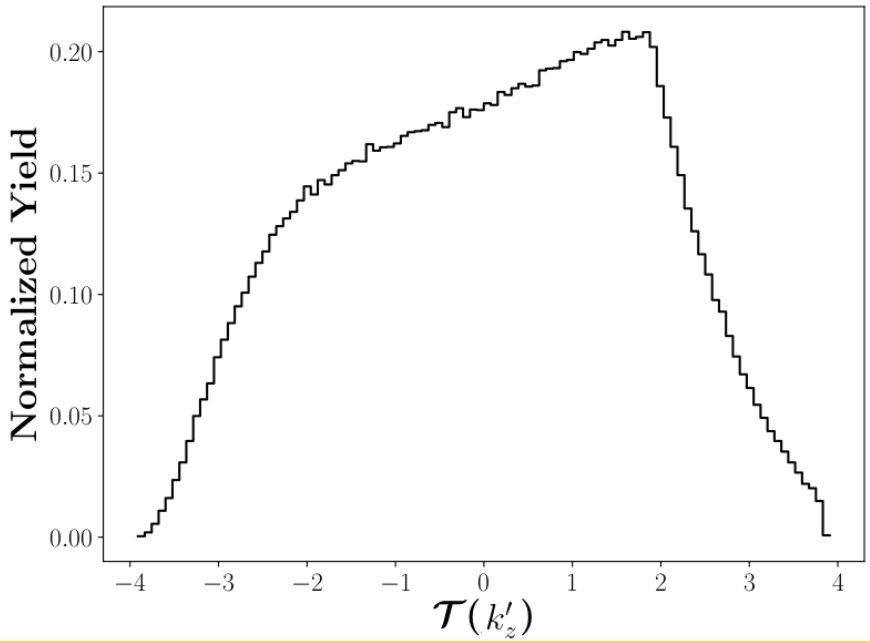
Augmented features, $E'$, $k'_T$, $k'_i \cdot k'_j$, (where $i, j = 1, 2, 3; i \neq j$) are calculated in a Lambda layer

# Results

Abrupt limits in physics variables distributions, driven by the conservation laws, resulted in poor reproduction of the DIS channel.

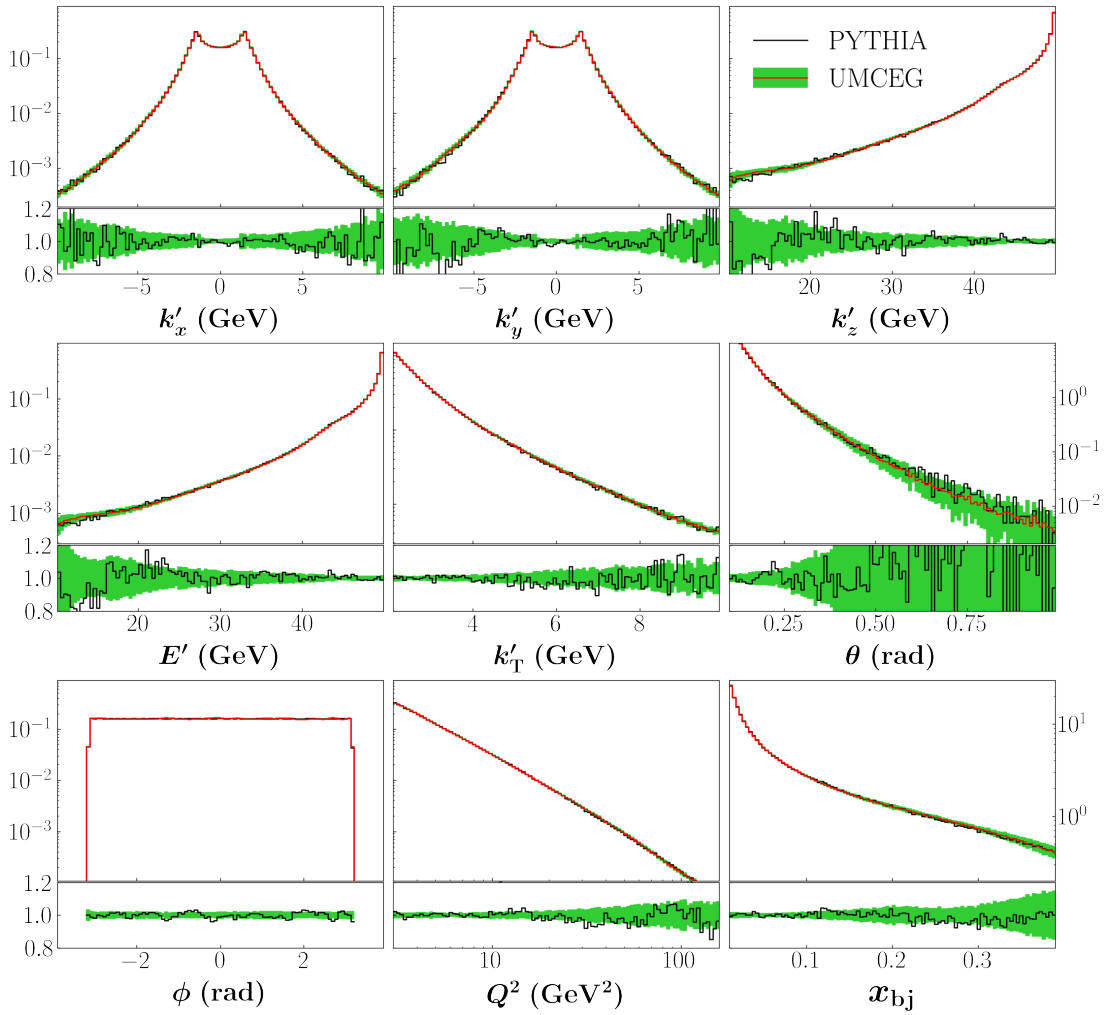
A longitudinal momentum transformation was utilized to flatten the particle distributions: $\tau(k'_z) = \log(E_{beam} - k'_z)$



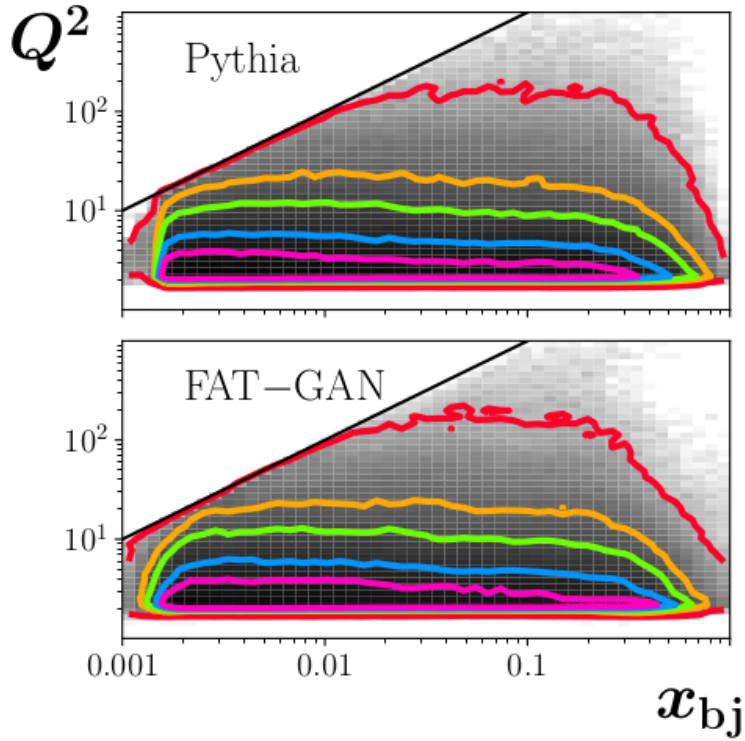



# Results

1D DISTRIBUTIONS OF FEATURES AND RECONSTRUCTED VARIABLES




2D CORRELATIONS OF RECONSTRUCTED VARIABLES



# Conclusions

In conclusion, our GAN-based Monte Carlo Event Generator

- was trained on PYTHIA data at the vertex level,
- reproduces the DIS phase space appreciably well,
- picks up correctly correlations between variables that were not used in the training,
- awaits incorporation of detector effects via a folding GAN, which will allow using experimental data in training.

**Acknowledgments**

This work is in part supported by Jefferson Lab LDRD project No. LDRD2122.

# Architecture

- **Loss functions:**
  - The discriminator: improved Wasserstein loss with a gradient penalty,

$$L_D = \left(\mathbb{E}[D(\widetilde{F})] - \mathbb{E}[D(F)]\right) + \lambda\, \mathbb{E}_{\widehat{F}\sim P_{\widehat{F}}}\left[(\|\nabla_{\widehat{F}} D(\widehat{F})\|_2 - 1)^2\right],$$

  - The generator: Wasserstein loss aided by a two sample test based on kernel Maximum Mean Discrepancy (MMD),

$$L_G = -\mathbb{E}[D(\widetilde{F})] + \eta\, MMD^2(F, \widetilde{F}),$$

$\mathbb{E}[\cdot]$ represents an expectation value,
$F$, $\widetilde{F}$ represent real and fake samples respectively,
$\lambda$, $\eta$ - hyperparameters optimized to improve training stability.