Use of Neural Networks for Cosmic ray simulations

Pranav Sampathkumar
IAP, KIT
December 8, 2021

Overview

Goal

To try and replace certain modules in CORSIKA with a neural network counterpart inorder to speedup cosmic ray simulations for High energies. High energy simulations provide a valuable window into high energy interactions not possible with current accelerators.

Why?

Explcit Monte Carlo simulations are slow and memory intensive at high energies. Novel techniques are needed to bypass explicit simulations.

What's new?

Use of neural networks to avoid the explicit Monte Carlo simulations or to use the networks to improve the quality of data with fewer or less detailed simulations

Explicit Simulations

- Time complexity of CORSIKA simulations raise approximately linearly with primary particle energy.
- Thinning (a weighted sampling method) is done to reduce particle content while preserving shower properties to leading order.

Advantages of using neural networks

- It is automatically parallelized by using standard libraries.
- Automatic GPU integration.
- Dimentionality Reduction (Reduction of phase space into "essential features").

Explicit Simulations

- Time complexity of CORSIKA simulations raise approximately linearly with primary particle energy.
- Thinning (a weighted sampling method) is done to reduce particle content while preserving shower properties to leading order.

Advantages of using neural networks

- It is automatically parallelized by using standard libraries.
- Automatic GPU integration.
- Dimentionality Reduction (Reduction of phase space into "essential features").

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

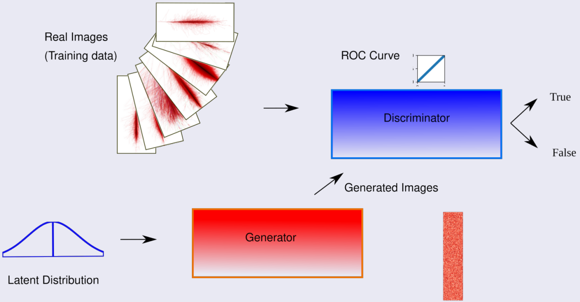


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

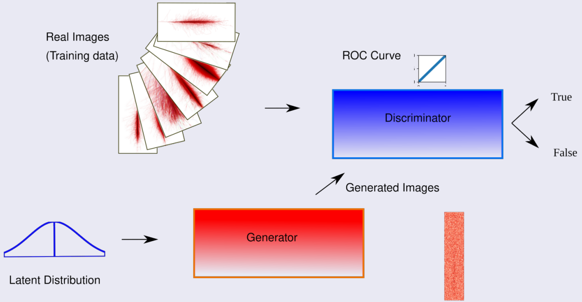


Figure: GAN

GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

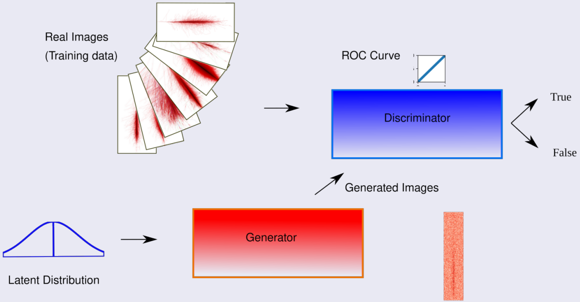


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

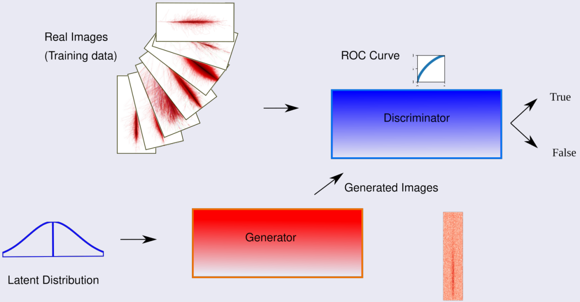


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

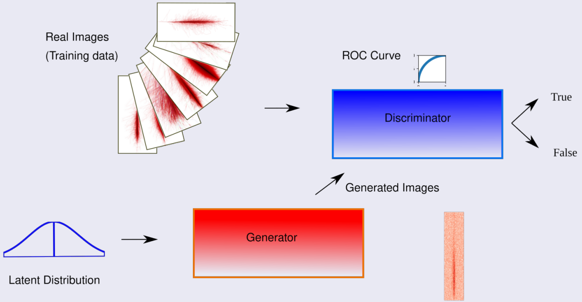


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

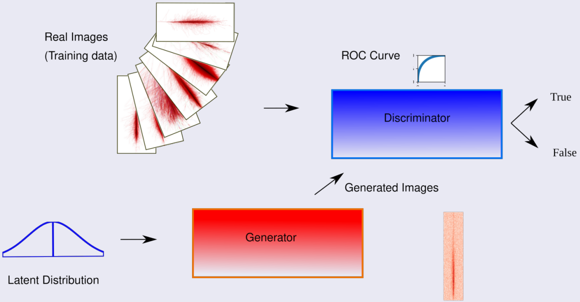


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

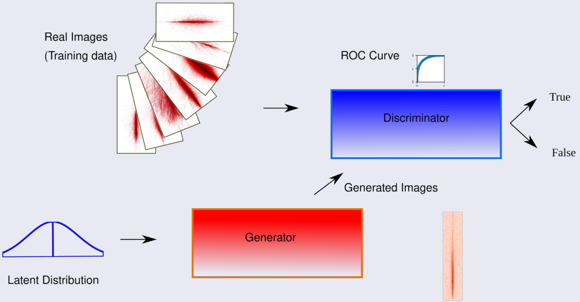


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

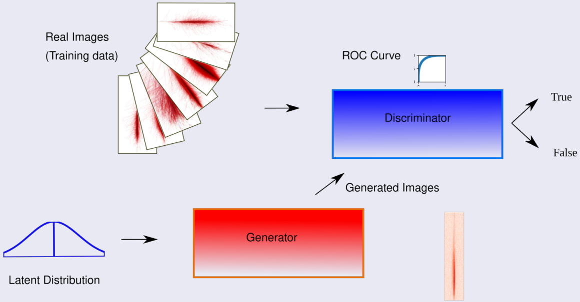


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

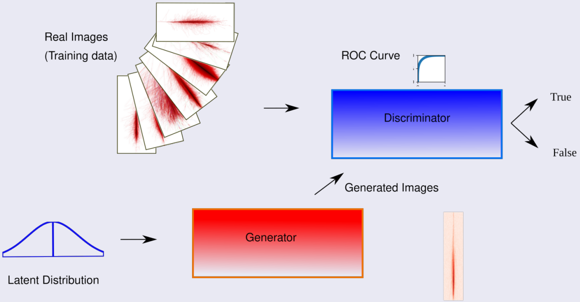


Figure: GAN

GAN

Generative Adversarial Neural network (GANs) pit two networks against each other inorder to generate data emulating the training data

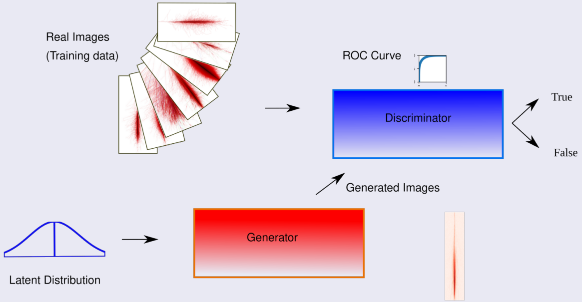


Figure: GAN

Attempts at GAN

Previous Attempts

- CaloGAN:Simulates 3D particle showers in multilayer calorimeters (PhysRevD.97.014021).
- Precise simulation of electromagnetic calorimeter showers using a Wasserstein Generative Adversarial Network (ArXiv: 1807.01954).
- Numerous other examples from other fields of physics such as cosmology (CosmoGAN (1706.02390), CAMELS Project).

Challenge

The challenge here is to be able to generate the intermediate tracks or some representation of tracks - “Particle level”, rather than generate images of detector responses - “Detector Level”.

Initial Attempts

- Attempt to sample from universality distributions as a check for Monte Carlo simulation using GANs.

Attempts at GAN

Previous Attempts

- CaloGAN:Simulates 3D particle showers in multilayer calorimeters (PhysRevD.97.014021).
- Precise simulation of electromagnetic calorimeter showers using a Wasserstein Generative Adversarial Network (ArXiv: 1807.01954).
- Numerous other examples from other fields of physics such as cosmology (CosmoGAN (1706.02390), CAMELS Project).

Challenge

The challenge here is to be able to generate the intermediate tracks or some representation of tracks - "Particle level", rather than generate images of detector responses - "Detector Level".

Initial Attempts

- Attempt to sample from universality distributions as a check for Monte Carlo simulation using GANs.

Attempts at GAN

Previous Attempts

- CaloGAN:Simulates 3D particle showers in multilayer calorimeters (PhysRevD.97.014021).
- Precise simulation of electromagnetic calorimeter showers using a Wasserstein Generative Adversarial Network (ArXiv: 1807.01954).
- Numerous other examples from other fields of physics such as cosmology (CosmoGAN (1706.02390), CAMELS Project).

Challenge

The challenge here is to be able to generate the intermediate tracks or some representation of tracks - "Particle level", rather than generate images of detector responses - "Detector Level".

Initial Attempts

- Attempt to sample from universality distributions as a check for Monte Carlo simulation using GANs.

Results and lessons from GAN attempts

Results

- Made a general framework for quick GAN modelling in Python.
- Hyperparameter optimization with Optuna.

Results

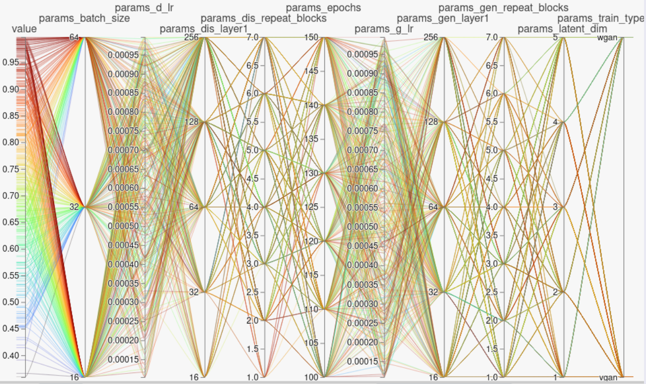


Figure: Multidimensional plot of the hyperparameters

Results and lessons from GAN attempts

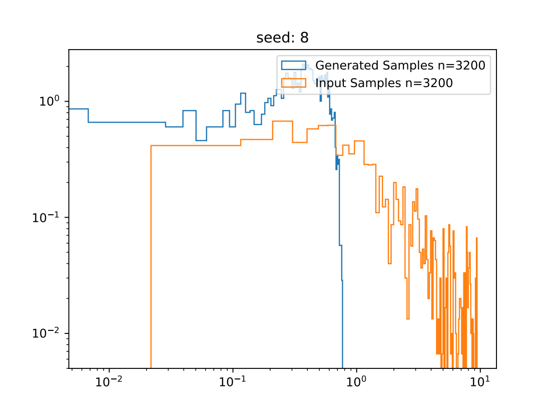


Figure: A particular instance of generated distribution trained with the universal distributions

Lesson

Very hard to learn the tail of the distributions because of bad samples.

Time for a different model?

Limitations of GAN

- Very hard to troubleshoot since the entire process is done in a single step.
- We realized, that non-gaussian distributions are hard to generate with a single output node. This is likely a fundamental limitation.
- We dont have a fixed representation when trying to generate data at a particle level.

When is GAN useful?

We still believe there is potential in the GAN approach, when trying to generate data at the detector level. This has been shown in numerous other works, but for our usecase, GANs are potentially a wrong design choice as they prove to be both intractable and for fixed representations.

What next?

There is a lot of research needed for the representation to use when using neural networks at a particle level. We currently move away from that and decide to emulate the shower based on the source functions given to CONEX. This fixes our representation of the shower, and we likely need another model, which is tractable and handles source functions simular to our understanding of physics.

Time for a different model?

Limitations of GAN

- Very hard to troubleshoot since the entire process is done in a single step.
- We realized, that non-gaussian distributions are hard to generate with a single output node. This is likely a fundamental limitation.
- We dont have a fixed representation when trying to generate data at a particle level.

When is GAN useful?

We still believe there is potential in the GAN approach, when trying to generate data at the detector level. This has been shown in numerous other works, but for our usecase, GANs are potentially a wrong design choice as they prove to be both intractable and for fixed representations.

What next?

There is a lot of research needed for the representation to use when using neural networks at a particle level. We currently move away from that and decide to emulate the shower based on the source functions given to CONEX. This fixes our representation of the shower, and we likely need another model, which is tractable and handles source functions simular to our understanding of physics.

Time for a different model?

Limitations of GAN

- Very hard to troubleshoot since the entire process is done in a single step.
- We realized, that non-gaussian distributions are hard to generate with a single output node. This is likely a fundamental limitation.
- We dont have a fixed representation when trying to generate data at a particle level.

When is GAN useful?

We still believe there is potential in the GAN approach, when trying to generate data at the detector level. This has been shown in numerous other works, but for our usecase, GANs are potentially a wrong design choice as they prove to be both intractable and for fixed representations.

What next?

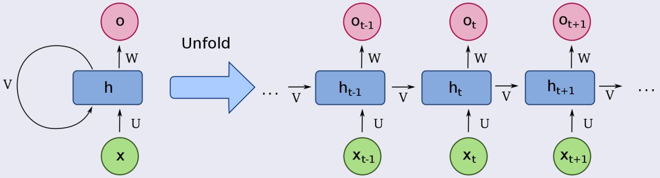
There is a lot of research needed for the representation to use when using neural networks at a particle level. We currently move away from that and decide to emulate the shower based on the source functions given to CONEX. This fixes our representation of the shower, and we likely need another model, which is tractable and handles source functions simular to our understanding of physics.

Recurrent Neural Networks

Reason

Recurrent neural networks are more suited towards generating time series data. It is similar to our usecase of propagating through the atmosphere.

Design



We step though height and generate a table for every height.

CONEX Inspired Framework

Goal

Use Recursive neural networks to step through the height and generate the source function for every step

CONEX

Internally conex takes the source function at the end of the hadronic cascade and then solves the 1D cascade equation.

Design Philosophy

We would like to do something similar to what CONEX does. We build a recursive neural network which essentially tries to find the function f in, $y(x + \Delta x) = f(y(x))$. which would be the solution to the cascade equation. We will call f the stepping function from now.

Plans

Short term goal

- We generate data by making CONEX write the intermediate steps to a file in a purely EM cascade.
- We then train the neural network with these intermediate steps and hope the neural network can emulate the stepping function.
- We chose to do this with CONEX, because EM Cascade is relatively simple and we can get the data faster than using C8.

Challenge

Finding the right model which will enable us to do this.

Plans

Mid Term Goal

- If we are able to get the neural network to step similar to CONEX, then we are on the right track.
- Second goal, is to code a piping setup which takes slices from an actual shower from C8 and generates the source functions.
- Once we have the data and the source functions, we can train the RNN with actual C8 data this time.

Challenge

We are moving from using aggregate data from CONEX to using individual data from C8. We need a way to make the neural network learn from "rare data". This is vital for us to be able to learn about fluctuations.

Plans

Long Term Goal

- Method 1 (3D showers)
  - Modify the network to generate the entire 3D distribution.
- Method 2 (More complicated Showers)
  - Add additional interactions into C8 and check if the RNN is able to pickup the stepping function.
  - We would need to modify the network so that we can introduce new stuff in between.

Challenge!

- Think and modify the source function (which is specialized for 1D) to be able to be applicable for 3D and more complex showers.
- Performance ?

THANK YOU!