Fast Simulation with Generative Adversarial Networks

F. Carminati (CERN), G. Khattak (CERN), S. Vallecorsa (CERN)
Outline

Introduction
   Our model: 3D Convolutional GAN
   Physics Performance

Genetic Algorithms for training and architecture optimisation

Summary
**3D convolutional GAN**

**Condition** training on input **particle energy** and incident angle, **Custom losses**
**Auxiliary regression tasks** assigned to the discriminator

**Generator:**
- Latent space
- Angle
- Energy
- 9x9x8x8 Reshape
- Upsampling (0, 0, 0)
- Convolution 1
- 8x8x6
- Batch normalization
- Zero padding
- Convolution 3-5 are similar
- 6x4x6
- Convolution 6
- 6x3x5
- Convolution 7
- 1x2x2
- Generated Image

**Discriminator:**
- Image
- Convolution 1
- 16x5x6
- Convolution 2
- 8x5x6
- Dropout
- Convolution 3
- 8x5x6
- Convolution 4
- 8x5x6
- Average Pooling
- Flattening

Lambda 1
Lambda 2
Lambda 3

- Angle
- Sum
- Bin counts
- Real/fake
- Energy
Two steps training

- Train on 100-200 GeV energy range
- Transfer learning to full spectrum
3DGAN Performance

- Convergence and discriminator performance
  - Stable test loss
  - Discriminator Real/Fake probability peaks at ~50%
  - Correct incident angle
- Comparison to Monte Carlo
  - Shower Shapes, Sampling Fraction
  - Correlations
  - Sparsity, etc.
- “In-house inception score
  - Compare TriForce\(^{(1)}\) classification and regression on GAN/GEANT4
- Image Quality Analysis

\(^{(1)}\) Matt Zhang, https://github.com/BucketOfFish/Triforce_CaloML
Generalisation

Training and architecture hyper-parameters optimisation

Different geometries, read-out patterns, energy scales
  Tuning the right architecture cannot be done by hand
  Full parameter scan is resource/time consuming.

Test different optimisation approaches:
  Sequential Model-Based Optimization
    Optimize initial architecture candidate, defining a finite set of states to explore
  Reinforcement Learning
    Network accuracy is the reward function. Architecture or hyper-parameter modification are actions
  Evolutionary Algorithms
    Can allow simultaneous weights training and architecture optimisation

mpi-learn/nnlo integrates an optimisation engine (mpi-opt)
Evolutionary Approach

Genetic Algorithm to train and optimize neural networks simultaneously

- **GA** can train Neural Network
  - **Global** instead of local minima
  - **Complex and indirect cost** functions are possible
  - **Highly Scalable**
- Currently used in hyper-parameter scans
  - Architectures are encoded as a chromosome
  - Weights are trained by gradient descent for evaluation of each individual
  - Time and resource intensive
Challenges

GA for GAN

- Network Size:
  - Trainable parameters in **millions** for 3DGAN model
- Large Computing resources?
- Architecture Optimization:
  - Flexible and stable
- Adversarial Training:
  - Simultaneous training of **two networks**
- Inexact solution:
  - A **hybrid approach** can incorporate SGD as a callback
Initial Implementation

GA with pytorch

- Reduced complexity:
  - Data reduced to two dimensions
  - Simplified discriminator regression task
  - Weight update
- Implement evolutionary algorithm
  - Testing batch training with GA
    - Smaller batches resulted in better training
  - Random updates vs. random weights
    - Random weights allow faster convergence
  - Number of offsprings
    - 6-8 optimum
  - Bias
    - Trained better without using bias
  - Comparison to RMSprop (lr=0.01)
    - Faster convergence
    - Lower accuracy
- Update weights and architecture at the same time can be much faster than hyper parameter scan based on GA
- Accuracy can be improved by further tuning or by adding a call back for gradient update after model stops improving
Plan

• Implement first combined architecture and training search on the simple classifier use case
• Estimate performance and needs in terms of computing resources
  • Several techniques can be explored (indirect weight encoding and asynchronous update)
• Increase complexity (network, dataset, …)
Thank you

Questions?
Detector output as 3D image

Array of absorber material and silicon sensors

CLIC Electromagnetic calorimeter design

Sparse images
Highly segmented (pixelized)
Large dynamic range

Segmentation is critical for particle identification and energy determination.
Indirect Weight Encoding

Evolving Filters

- Image is convolved with a filter kernel
  - Convolutional filter kernels can be used to extract interesting features from an image.
  - Digital filters can perform operations like smoothing and edge detection etc.
- The weights in a kernel are meaningful in their relation with other weights and its position
- Indirect coding can allow to learn the underlying relation of weights instead of fixed values
- Hyper Neat * uses indirect coding for neural network evolutionary training as Compositional Pattern Producing Networks CPPNs
- Same concept can be extended to CNN


Some commonly used edge detection filters

![HyperNEAT CPPN](http://axon.cs.byu.edu/~dan/778/papers/NeuroEvolution/stanley3**.pdf)
Asynchronous Update*

Genetic Training

- A pool of random chromosomes is created
- Fitness for each chromosome is evaluated
- Offsprings are obtained (mutation and/or cross over)
- Offsprings can be evaluated in parallel:
  - Fitness is compared with lowest fitness from the current pool
  - If fitness is equal or higher, then the offspring replaces the parent with lowest fitness
- Different offsprings can take different time for evaluation
- No need for inter communication
- No loss in performance with number of threads