End-to-End Deep Learning Reconstruction for CMS Experiment

An Application for Google Summer of Code 2021

Mentoring Organization:

ML4Sci - Machine Learning for Science

Sub Organization

End-to-End Deep Learning (E2E)

Mentors

Emanuele Usai (Brown University)
Davide DiCroce (University of Alabama)
Shravan Chaudhari (BITS Pilani Goa)
Sergei Gleyzer (University of Alabama)
Michael Andrews (Carnegie Mellon University)

Name: Purva Chaudhari
Email: purva.chaudhari02@gmail.com
purva.chaudhari18@vit.edu

Google Summer of Code

Table of Contents

Candidate

Personal Information

Name	Purva S. Chaudhari
University	Vishwakarma Institute of Technology, Pune, India
Stream	B.Tech Computer Science (3rd Year Undergraduate)
Github	Purva-Chaudhari_github
CV	Purva-Chaudhari_CV
LinkedIn	Purva-Chaudhari_LinkedIn
Website	https://purva-chaudhari.github.io/My-Portfolio/index.html
Time Zone	Indian Standard Time (GMT + 5:30) (IST)

Information and Motivation

I am Purva Chaudhari, a third-year student at Vishwakarma Institute of Technology, Pune pursuing B.Tech. in Computer Science Engineering. I have a strong foundation of object oriented programming in C++, python and java. I have had my hands on Deep Learning(Computer Vision and Natural Language Processing) and application development (android java, flutter dart) for two years now.

I am passionate about developing various models using innovative Deep Learning approaches and deploying them for various applications. I have limited knowledge and experience in High Energy Physics but the thing I loved most aboutEnd-to-End Deep Learning Reconstruction for CMS Experiment is that it took me very little time to learn and understand how to code and deploy Deep Learning models on the CMSSW (Compact Muon Solenoid Software) Framework. I have implemented various Deep Learning architectures like VGGs, Resnets, LSTMs, RNNs, Fully Connected Neural Networks, etc. I have also implemented various Machine Learning algorithms like Regressions, KNN(K-Nearest Neighbors), Heuristic Clustering, K-Means, SVM, Random Forest, Decision Trees, etc.

I have good experience in the following programming languages: Python,, C++,Java, MATLAB, etc. Also, I have experience in using Tensorflow, Keras, Pytorch, Sklearn, etc. for various Computer

Vision, Sentiment Analysis and Natural Language Processing Projects. You may refer to my CV for more information about my projects and experiences.

Summary

Importance

One of the important aspects of searches for new physics at the Large Hadron Collider (LHC) involves the identification and reconstruction of single particles, jets and event topologies of interest in collision events. The End-to-End Deep Learning (E2E) project in the CMS experiment focuses on the development of these reconstruction and identification tasks with innovative deep learning approaches.

This project focuses on the integration of E2E code with the CMSSW inference engine for use in reconstruction algorithms in offline and high-level trigger systems of the CMS experiment.

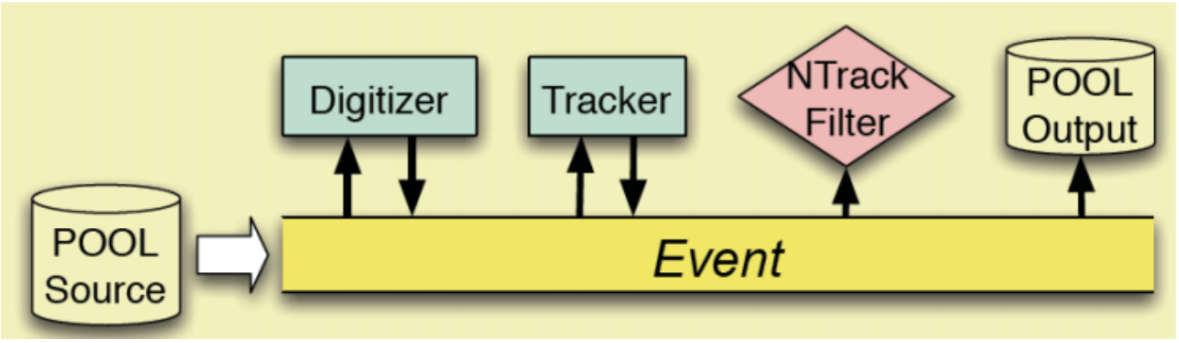
Overview

CMSSW Experiment

The CMS apparatus was identified, a few years before the start of the LHC operation at CERN, to feature properties well suited to particle-flow (PF) reconstruction. One of the important aspects of searches for new physics at the Large Hadron Collider (LHC) involves the identification and reconstruction of single particles, jets in collision events in the collision experiment. The Compact Muon Solenoid Collaboration (CMS) uses a particle flow (PF) reconstruction approach that converts raw detector data into progressively physically-motivated quantities until arriving at particle-level data. Such high-level features are the result of a rule-based casting of the raw detector data into progressively more physically-motivated quantities. While such approaches have proven to be useful, they are highly dependent on our ability to completely and effectively model all aspects of particle decay phenomenology. Powerful image-based Convolutional Neural Network (CNN) algorithms have emerged, capable of training directly on raw data, learning the pertinent features unassisted - so-called end-to-end deep learning classifiers.

CMSSW Framework

Figure 1: Flow of data in Event.



Source

It is the overall collection of software built around a framework, an Event Data Model (EDM), and Services needed by the simulation, calibration and alignment, and reconstruction modules that process event data so that physicists can perform analysis.The CMSSW event processing model consists of many plug-in modules which are managed by the Framework. Some of the modules are:

- EDProducer: creates new data to be placed in the Event.
- EDAnalyzer: studies properties of the Event.
- EDFilter: decides if processing should continue on a path for an Event.
- OutputModule: Stores the data from the Event.

Aim

An extension of work done as a part of GSoc 2020 the main aim is to integrate/interface E2E with the CMSSW inference engine and test and benchmark the inference on CPU and GPUs

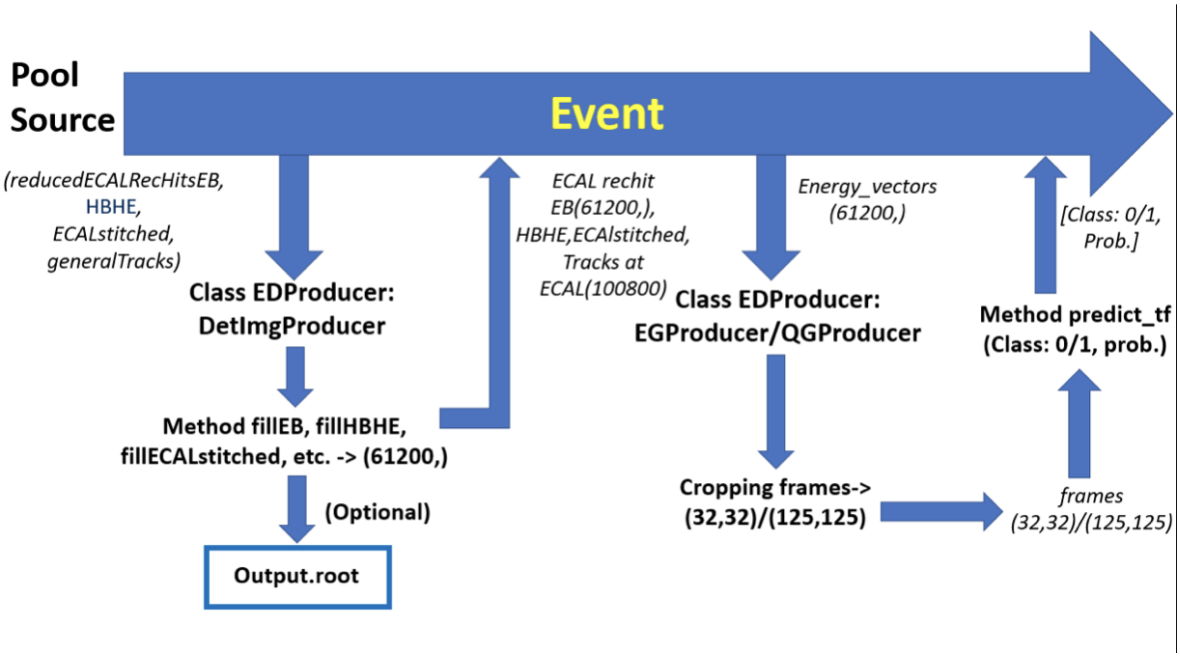
Project

Previous work

Click here to view the codebase
The work flow of the project as a part of GSoC 2020 included

- Reading detector images and storing the extracted image vectors to edm root file
- Extracting jet seed coordinates.
- Preparing the frames for inference.
- Running the inference and storing the predictions.
- Configuration files in cmssw.

Figure 2: Code Structure



Source

As a part of the project I would like to propose:

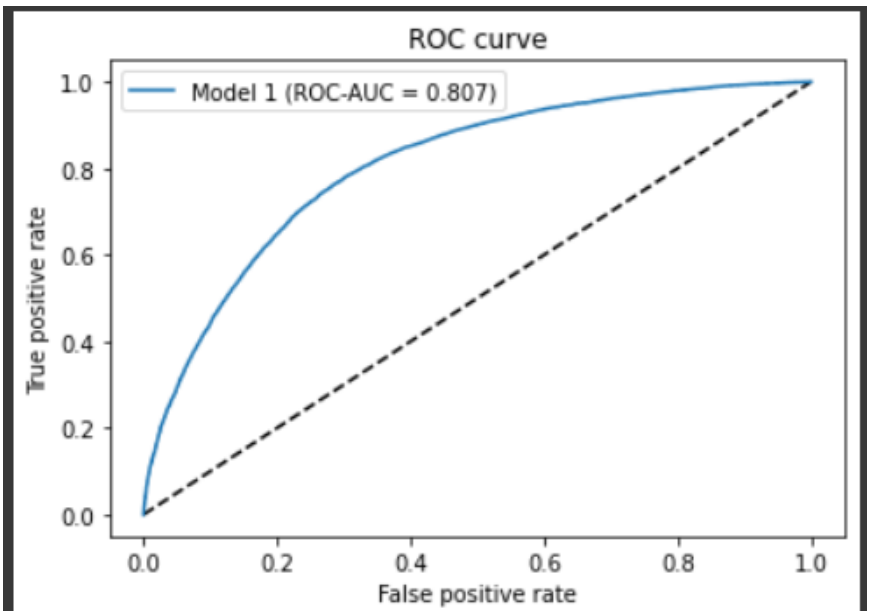
- ➢ Extend and integrate E2E with the CMSSW inference engine
- ➢ Referring to the previous year's work (and after discussion with a mentor),I understand that currently we have electron-photon and Quark-Gluon classifications which further is to be extended to
  - Identification of tau particles
  - Identification of beauty and charm quark jets
  - Muon Identification
- ➢ Since the interface currently supports only tensorflow we can extend it to other frameworks like pytorch.
- ➢ Test and benchmark the inference on CPU and GPUs
- ➢ Providing documentation (and blog) of the work done

Pre-Gsoc tasks done

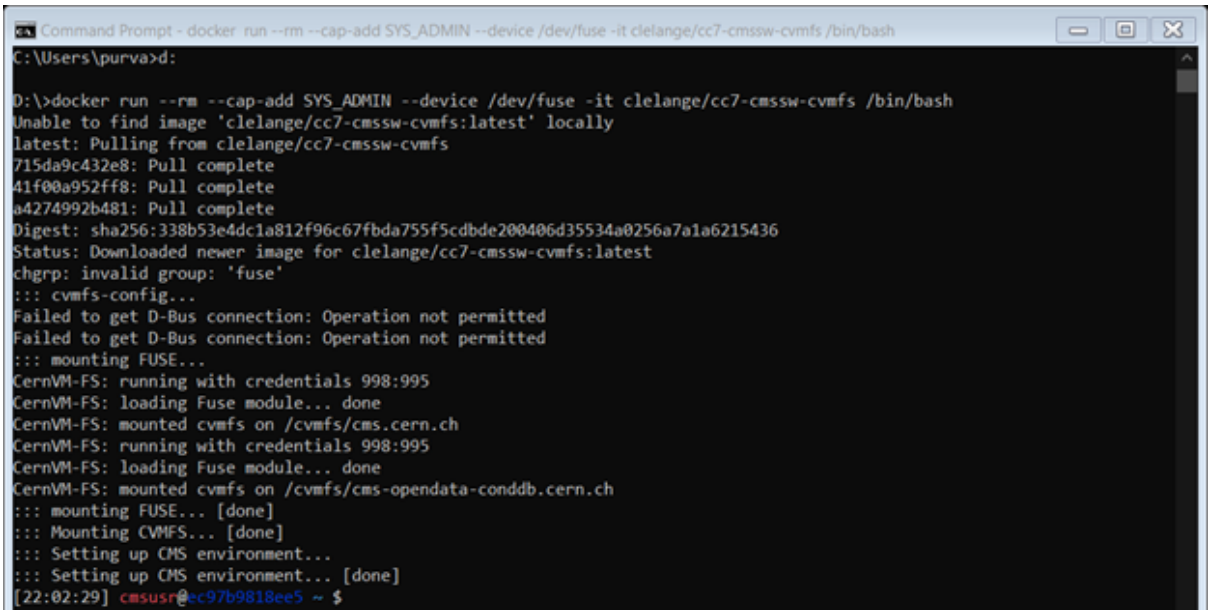
Task 1:Electron/photon classification
Description: 32x32 matrices (two channels - hit energy and time) for two classes of particles electrons and photons impinging on a calorimeter

Using Keras and Pytorch

Obtained ROC Curve (keras model):



Task 2: Setup CMS Software Framework (CMSSW) on the local machine (with docker)



Task 3: Deep Learning based Quark-Gluon Classification. (Partially done till now, working on it, might finish it later. Please refer my github link shared for the codebase).

Description: 125x125 matrices (three channelled images) for two classes of particles quarks and gluons impinging on a calorimeter

Click here to view my github

Project planning and solution approach

The project can be extended to include identification of few more high energy particles like tau, muon and beauty quark jets. Here I have proposed some significant deep learning techniques from papers to identify them..

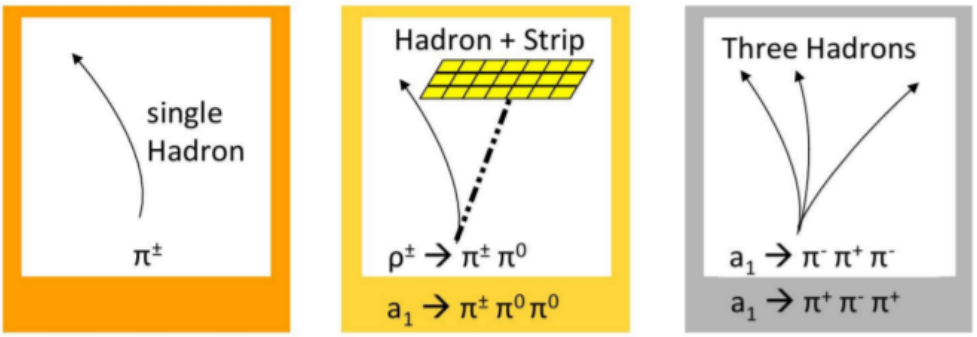
1. Identification of Tau particles

The tau (τ), also called the tau lepton, tau particle, or tauon, is an elementary particle similar to the electron, with negative electric charge and a spin of. Like the electron, the muon, and the three neutrinos, the tau is a lepton. There is now a lot of interest to search for different phenomena that include τ leptons in its signature. One of the examples of such searches is the decay of Higgs boson (H) into a pair of τ leptons which is the most sensitive channel for probing Higgs boson coupling to fermions. Searches with τ leptons in the final
state have high sensitivity to the production of additional Higgs bosons expected in the minimal supersymmetric standard model (MSSM). In addition, searches for different particles beyond the SM benefit significantly from any improvements made in τ lepton reconstruction and identification

1.3 Geometrical deep learning

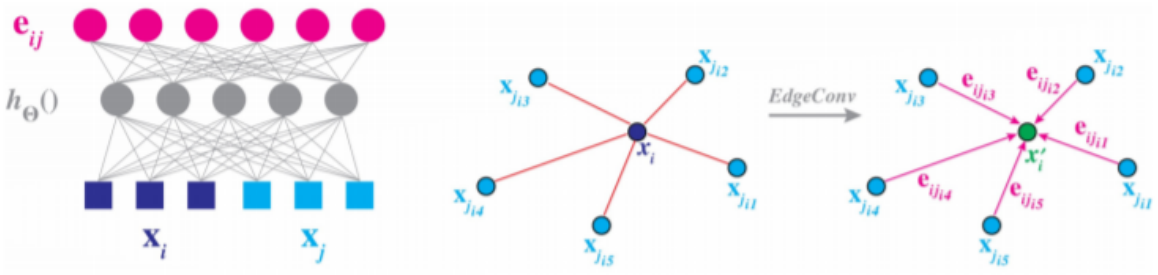
Data can be represented as a graph which can be irregular and have varying size, which means that most operations widely used for data on Euclidean domain, such as convolutions, cannot be used for this data and one need specific operations to efficiently extract features from such data. Recently there is a growing interest in extending deep learning approaches for such data.

An example of non-Euclidean data is a point cloud - scattered collections of points which comprise the output of the most 3D sensors. Qu and Gouskos, inspired by the notion of point cloud, propose to represent jets as an unordered set of its constituent particles, effectively a ”particle cloud”. Such particle cloud representation of jets is efficient in incorporating raw information of jets and also explicitly respects the permutation symmetry. This enables them to achieve state-of-the-art performance on two representative jet tagging benchmarks.



Reference

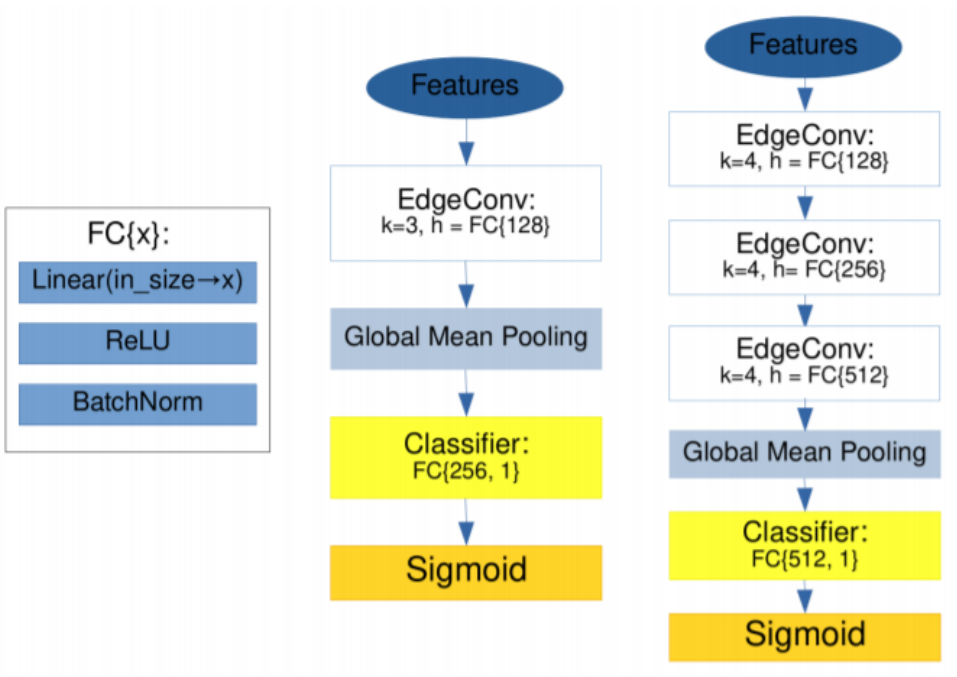
Edge Convolution:
For EdgeConv operation, the point cloud is first represented as a directed graph - points are the vertices of the graph, and each vertex is connected with its k nearest neighbours. For each edge of the graph, the edge features is computed as a eij = hΘ(xi , xj), where xi and xj are the points connected by the edge, and hΘ is a function parametrized by parameter vector Θ. Function hΘ is usually implemented as a multi-layer perceptron (MLP). EdgeConv operation is defined by applying a symmetric aggregation operation (e.g. P or max): x' i = hΘ(xi , xj)



Left: An example of computing an edge feature. Right: Visualisation of EdgeConv operation.

There are different ways in which arguments can be fed into MLP, and in our work we use asymmetric edge function, which explicitly combines global shape structure with local neighbourhood information: hΘ(xi , xj) = h¯ Θ(xi , xj − xi)

Architecture:



Left: Single fully-connected layer; Center: ECN with 1 conv layer; Right: ECN with 3 conv layers

2. Identification of beauty quark jets

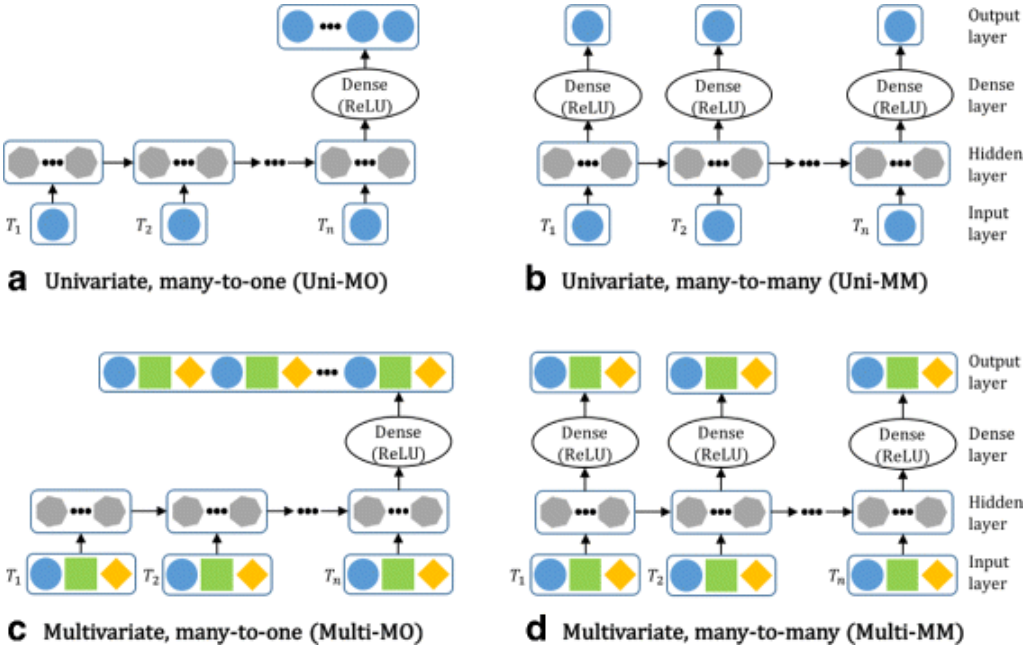
Identification of jets that originate from the hadronization of beauty (b) and charm (c) quarks is important for studying Standard Model (SM) processes and for searching for new physics. For example, the ability to efficiently identify b jets with minimal misidentification of c and light-parton jets is crucial for the measurement of top-quark production.

The study of high-energy beauty quarks is of great interest at the LHC because these particles are frequently produced in the decays of Higgs bosons and top quarks and are predicted to be important components of the decays of super-symmetric and other hypothetical particles. To identify jets that originate from high-energy beauty quarks, it is necessary to be able to determine whether particles were produced directly in the proton–proton collision or in the subsequent decay of a long-lived bound state at a location displaced from the proton–proton collision. Because jets typically contain between 10 and 50 particles, the number of potential discriminating features vary on a per-jet basis. Traditional jet-identification algorithms rely on either identifying secondary production points explicitly from the crossing of particle trajectories, a highly challenging task, or compressing the information with engineered features and neglecting the correlations between particles when using single-particle features. Although such algorithms have been combined with machine learning for some time, machine learning could also be used to improve identification by using the low-level particle features within a jet.

2.1 RNNs for beauty-quark identification:

A recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time series data. These deep learning algorithms are commonly used for ordinal or temporal problems, such as language translation, natural language processing (nlp), speech recognition, and image captioning; they are incorporated into popular applications such as Siri, voice search, and Google Translate. Applying an RNN to jet classification requires the particles in the jet to be ordered to form a sequence,such as by ranking them how incompatible they are with originating from the proton–proton collision. A set of features for each particle will be provided to the RNN, which would be trained to discriminate between beauty-quark jets and all other types of jet.

RNN Architecture:

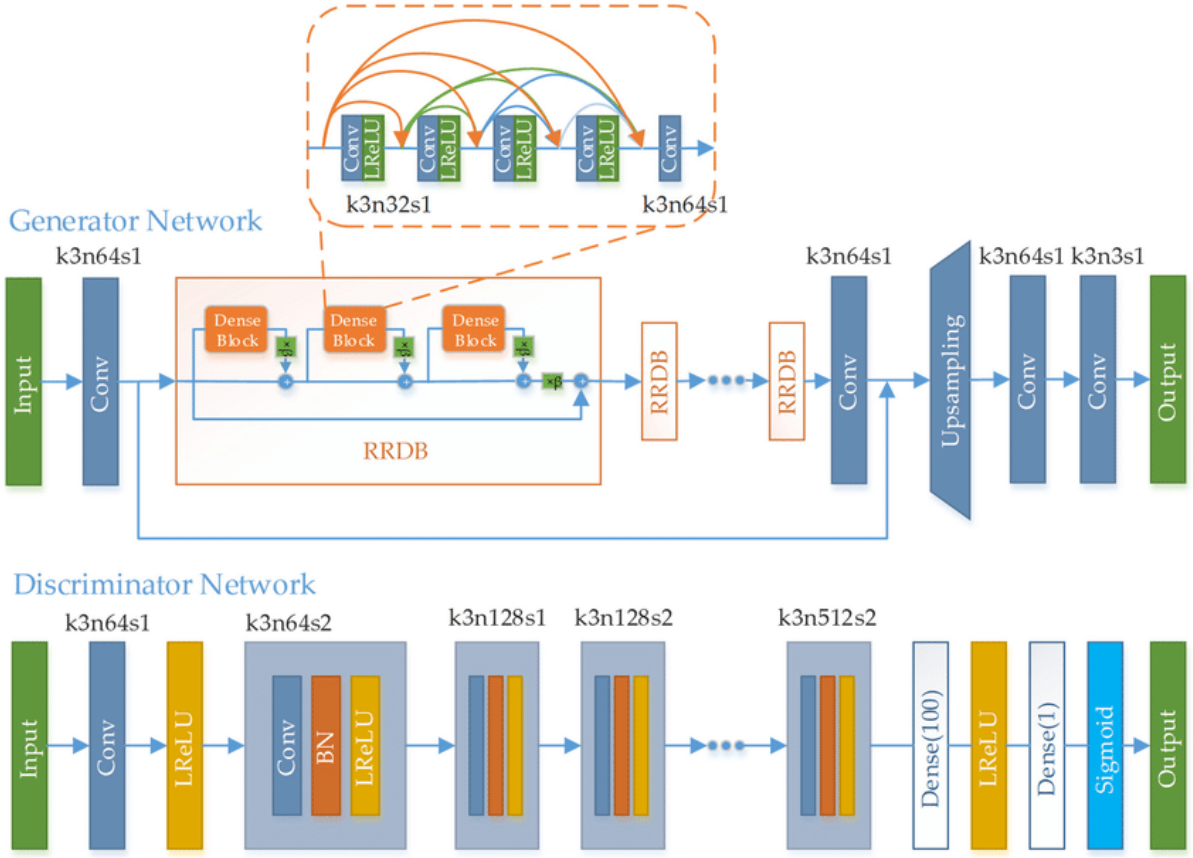


2.2 Use of GANs

Generative models, which learn the probability distribution of features directly, are capable of producing simulated data that closely approximate experimental data using tools such as generative adversarial networks and variational auto-encoders. A generative adversarial network uses a neural network, ‘the generator’, to generate potential data samples using random noise as input, while a second network, the ‘adversary’, penalizes the generator during training if the data that it generates can be distinguished from the training data. Although they are difficult to train, these networks can potentially generate large data samples much faster than existing simulation tools, offering the

possibility of providing the orders-of-magnitude-larger simulation samples that will be required by future experiments. Early work in this direction is encouraging, demonstrating that accurate simulations of a simplified calorimeter can be produced while achieving a marked decrease in the computational resources required. The adversarial approach can also be applied to training classifiers with the ability to enforce invariance to latent parameters. This represents a new way of making classifiers robust against systematic uncertainties and is a viable approach to avoid biasing a physical feature such as mass.

GAN Architecture



3. Muon Identification

The muon is an elementary particle similar to the electron, with an electric charge of −1 e and a spin of 1/2, but with a much greater mass. It is classified as a lepton. Because muons have a greater mass and energy than the decay energy of radioactivity, they are not produced by radioactive decay. However they are produced in great amounts in high-energy interactions in normal matter, in certain particle accelerator experiments with hadrons, and in cosmic ray interactions with matter. These interactions usually produce pi mesons initially, which almost always decay to muons. As with the

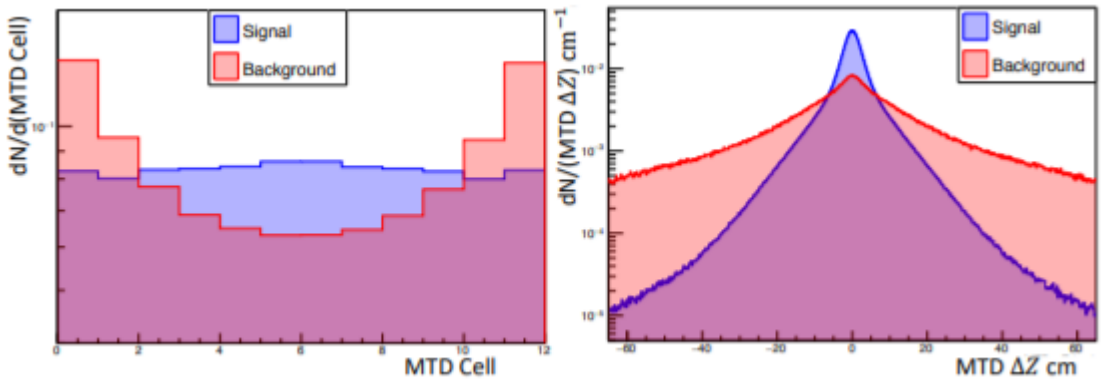
other charged leptons, the muon has an associated muon neutrino, denoted by νμ, which differs from the electron neutrino and participates in different nuclear reactions.

. The Muon Telescope Detector (MTD) has made muon identification over a large momentum range possible for the first time at STAR. However, even with the MTD, identification of pure muons can be challenging due to backgrounds from hadron punch through. The identification of dimuon pairs is further obscured by secondary muons originating from the weak decays of $\pi \rightarrow \mu+\nu$ and $K \rightarrow \mu+\nu$.

Training dataset:

The data used in this approach was collected by the STAR detector from p+p collisions at $\sqrt{s} = 200$ GeV during the 2015 RHIC run. The events were selected using the dimuon trigger requiring that at least two MTD signals be measured within a timing window.

Muon candidate tracks were required to have a $pT > 1$ GeV/c, have a distance of closest (DCA) approach to the collision vertex of DCA < 3 cm, be reconstructed from more than 15 clusters in the TPC, have a ratio of reconstructed clusters to possible clusters greater than 0.52 to reject split tracks, and to have at least 10 clusters used for the dE/dx measurement to ensure a reasonable dE/dx resolution.



Ref: https://arxiv.org/pdf/1908.05645v1.pdf

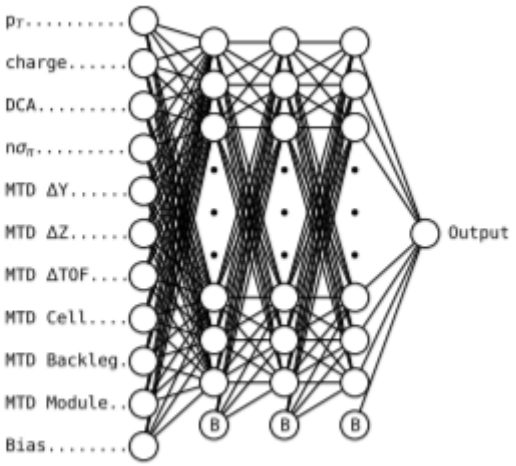
3.1 Shallow Neural Network

The use of dense Multilayer Perceptrons (MLP), a type of feed-forward ANN, are trained as continuous classifiers for the purpose of muon identification.

A shallow artificial neural network is defined by the presence of a single hidden layer of neurons between the input and output layers. In this approach there is performance of a large set of SNNs as a function of the number of neurons in their hidden layer. The models are trained using the Toolkit for Multivariate Data Analysis with ROOT (TMVA).

The following figure is an example of a dense multilayer perceptron neural network architecture. The shallow neural networks have only a single hidden layer of neurons between the input and

output layers. The deep neural networks have two or more. Bias neurons in the hidden layers are marked with a ”B”.



Ref: https://arxiv.org/pdf/1908.05645v1.pdf

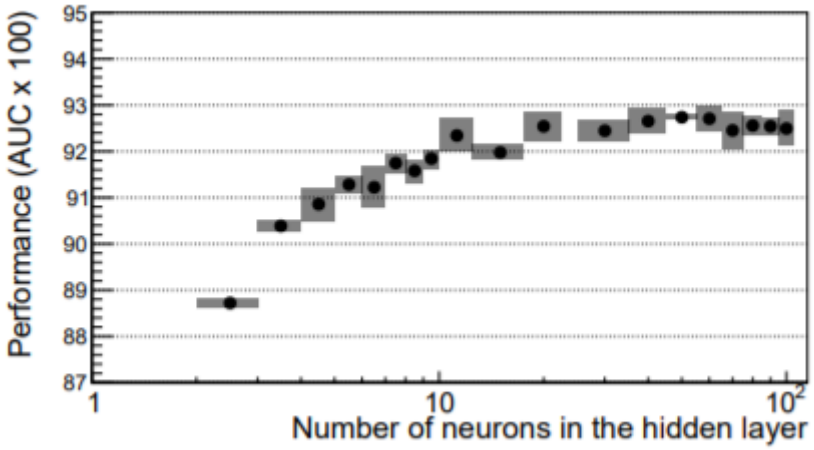
The SNN models include a bias neuron in each of the input and hidden layers to account for trivial offsets in the mean value of the data. The bias neuron is always ”on” - i.e. it provides an input of 1 so that weights between it and other neurons are constant factors. The use of a bias node in the input and each hidden layer has become standard practice in neural network architecture design

3.2 Deep Neural Networks and Hyperparameter Optimization

Deep neural networks, in contrast to SNNs which contain only a single hidden layer, contain two or more hidden layers. The additional hidden layers can allow a network to learn complex relationships between input features with far fewer neurons and connections than a shallow network would need. Depending on the application it is also common for DNNs to combine various types of layers, such as convolutional layers, to promote the learning of specific types of relationships

For the case of muon identification, the goal is to determine if DNNs can provide a better classification performance than SNNs with a reasonable number of neurons. The process of determining the optimal
DNN architecture is often referred to as hyperparameter optimization. A grid search strategy is used in this case to search the optimal DNN architecture on a grid of the number of hidden layers and the number of neurons in each layer.

The following figure depicts the signal vs. background rejection power as a function of the number of neurons (NNHL) in the hidden layer of a shallow neural network. The performance of the SNNs are quantified using the AUC - the area under the background rejection vs. signal efficiency curve



Ref: https://arxiv.org/pdf/1908.05645v1.pdf

Training DNNs can require significantly more time and larger labeled samples compared to SNNs to reach convergence

Deliverables

Minimal Set of Deliverables

- Extend and integrate E2E with the CMSSW inference engine
- Referring to the previous year's work (and after discussion with a mentor),I understand that currently we have electron-photon and Quark-Gluon classifications which further is to be extended to
  - Identification of tau / beauty and charm quark jets (whichever is prioritized)
- Extend the interface to other frameworks like pytorch.
- Test and benchmark the inference on CPU and GPUs
- Providing documentation (and blog) of the work done

Extended Set of Deliverables

- Implementation of Muon Identification

Possible Constraints

Probably the most challenging part of the integration of a new framework (like pytorch) into the interface, which I think may be solved once getting more familiarized with the working and deployment of models. Additionally, I think at this point my plan might be a bit too ambitious and I

could lack time to implement everything that I would like to do, but I would implement the best optimized strategy with the help of the mentors

Implementation Plan

Tentative Timeline

I have included a week wise schedule of the main module I will be working. Along with it I will be doing experiments/prototypes with one part of the code while finishing the transition for other parts.

Time and Duration	Tasks
Pre-GSoC	
15 April - 17 May	➢ Improving the model and accuracy of the tasks. ➢ Studying the latest researches done in the field and studying previous work done by the organizations for the same ➢ Familiarizing with the CMSSW Framework going through their repositories and testing a few basic codes and libraries using the CMSSW inference engines.
Community Bonding Period	
17 May - 7 June	➢ Understanding organization workflow, communication and contribution process. ➢ Reviewing existing algorithms or Deep Learning architecture implementations ➢ Discussing the shortcomings (if any) of the previous implementations ➢ Prioritizing implementation
Coding Period Begins	
Week 1 7 June - 13 June	➢ Exploring the dataset. Performing Exploratory Data Analysis to determine the essence of the algorithm/ neural network architecture that can be used on the dataset.

	➢ Exploring different Machine Learning algorithms, Deep Learning architectures and combinations of both that can be implemented and will be beneficial for tau/ beauty quark identification for CMS experiment and selecting efficient implementation methods.
Week 2- 3 14 June - 24 June	➢ Preprocessing the dataset and preparing it to feed to the algorithm/architecture and splitting it into training, validation and test datasets uniformly. ➢ Finishing the basic implementation of the selected architectures or Machine Learning algorithms and developing the models to train them on the dataset after choosing appropriate metrics, loss functions, optimizers, etc.
Week 3-5 25 June - 7 July	➢ Optimizing each model by tuning the hyperparameters of each model and improving with their architecture (changing number of neurons, kernel size, number of hidden layers, Convolutional blocks, etc.) to improve the results in terms of the metrics chosen and comparing them on training, validation and test datasets. Taking care of the fact to not make the models too complex for initial testing on the CMSSW inference engine
Week 5-6 8 July- 18 July	➢ Testing the trained models (which performed decently on the dataset) on the CMSSW inference engine and fixing bugs. ➢ Adding documentation and updating the repository.
Mid Evaluations	
Week 7 19 July - 18 July	➢ Testing the implemented models on the required platform and improving documentation and submission. ➢ Buffer period, to catch if left behind
Week 8-9 19 July - 7 August	➢ Working on including a new framework (Pytorch)

	➢ Testing modals developed on Pytorch in the CMSSW environment and fixing the bugs
Week 10-11 8 July - 23 August	➢ Integrating the final implementation in CMSSW ROOT. ➢ Time for covering up tasks due to unexpected delays ➢ Documentation of all the GSoC work ➢ Writing blog
Final Evaluation	

FAQ

1. How much time will I be able to contribute to this project? What are other commitments this summer?

The official GSoC period is from 7th June to 23th August (2021). I can easily give 20-30 hours a week till 31st July and 20-25 hours a week after that (have provided it according to revised GSOC rules). I intend to complete most of the work before 31st July.
Other than this project, I do not have any other commitments/vacations planned for the summers. Also, I do not have any internships this summer other than this project. Currently my University End Semester exams are scheduled till 20th May. (Fixed schedule is yet to be out) This may cause my little unavailability during community bonding period. Else I am available.

2. Preferred medium of communication:

I am fine with Email, Zoom, Gmeet, Gitter, Slack, Skype or any other similar medium of communication. My preferred communication language is English.

3. What makes you a good candidate for the project?

I am to-be computer scientist, deeply passionate about research and software development in the fields of Physics and Computational Biology. Besides considering the importance of the project of End-to-End Deep Learning Reconstruction for CMS Experiment, I found the project a great match to my skill set and would like to use my calibre to contribute to it. Being working on Machine Learning and Deep Learning techniques of Computer Vision for over more than a year now, I think I can come up with solutions and figure out different ways of implementation on my own. Besides as pre-gsoc tasks, I got very good insight, understanding and working vision of the project, so I can very well figure out ways for implementation

4. Are you interested to further support the project after the end of the fellowship?

Yes Definitely! I am deeply passionate about Research in the field of Computational Science and am looking forward to working for the project and organization post GSOC and being a long term member of the community.

References:

https://arxiv.org/pdf/1611.05531.pdf
https://arxiv.org/pdf/2103.11769.pdf
https://arxiv.org/pdf/2006.08639.pdf
https://www.osti.gov/servlets/purl/1504380
https://microboone.fnal.gov/wp-content/uploads/MICROBOONE-NOTE-1080-PUB.pdf
https://www.sciencedirect.com/science/article/pii/S0375947418303178
https://shra1-25.github.io/E2eDLrecReport/#
https://www.desy.de/f/students/2019/reports/Kirill.Bukin.pdf
