PROJECT PROPOSAL

Google Summer of Code 2021

Graph Neural Networks for End-to-End Particle Identification with the CMS Experiment

---

Organization: ML4SCI

Sub Organizations:

E2E
University of Alabama
Brown University
Carnegie Mellon University

Name: Shravan Sunil Chaudhari
Email id: f20170736@goa.bits-pilani.ac.in

Table of Content

---

1. My Introduction:

● Personal Information:

Name	Shravan Sunil Chaudhari
University	Birla Institute of Technology and Science-Pilani, Goa Campus
Github	Shravan_S_Chaudhari_Github
CV	Shravan_S_Chaudhari_CV
TimeZone	Indian Standard Time (GMT+5:30) (IST)
Address	B-4 Priyanka Meadows, Ashoka Marg, Nashik Road, Nashik, Maharashtra, India-422011

● About Me:

I am a senior year undergraduate student pursuing (Hons.) Bachelors of Engineering in Electronics and Instrumentation at BITS Pilani University, India. I have a strong background in applied and large scale machine learning, big data science, computer vision and signal and image processing. I am well versed with Tensorflow, Tensorflow-Lite, Pytorch, Pytorch Lightning, Horovod, Sklearn, Scipy, Pandas, Numpy and Torch Geometric Frameworks and also with programming languages like C/C++, Python, Java, MATLAB, Kotlin, Octave, Solidity for Ethereum. You may refer to my CV for more information about my projects and my experiences. I started contributing to the End-to-End Deep Learning for High Energy Physics project as a Google Summer of Code (GSoC) 2020 student and have been involved in the research of developing and optimizing End-to-End scalable Deep Learning algorithms for high-energy physics since then. As part of the Google Summer of Code 2020 project, I developed End-to-End Deep Learning inference code on CPUs and integrated it with CMS Software Framework (CMSSW). You may see more details about the project here. As a member of the CMS E2E research group, I have also worked on parallelizing and optimising the training and inference of deep learning models in tensorflow/keras or pytorch using GPU clusters and computing nodes provided by Nvidia by participating in several Nvidia GPU hackathons.

Currently, I am pursuing a research internship (bachelor's thesis) at CERN and the University of Alabama under the supervision of Dr. Sergei Gleyzer where I am working on developing Graph Neural Networks (GNNs) and comparing its performance with the Convolutional Neural Network (CNN) based approaches for identification of high energy particles (Boosted Top Jets) and integrating the end to end reconstruction of boosted top jet with the CMSSW (CMS Software) framework.

My interest in the proposed project stems from the experience I have gained by working on various research projects during my undergraduate degree especially from the project I am working on as part of my bachelor's thesis. This project is closely aligned with my current work and extends it to the identification of Tau particles. This project will provide me the opportunity to implement and study Graph based deep learning approaches for Tau particle identification and benchmark the results after careful analysis and comparison with the CNN based approaches. It will also provide me the opportunity to optimize them for better performance and efficiency when deploying the trained models for both the approaches on the CMSSW inference engine using the E2E framework developed by me during last year's GSoC program with the CERN-HSF organisation.

- FAQs:

How much time will I be able to contribute to this project? What are other commitments this summer?
The official GSoC period is from 7st June to 23rd August (2020). I can easily give 45-55 hours a week. 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.

Preferred medium of communication:
I am fine with Mattermost, Slack, Email, Gitter, Hangouts, Whatsapp, Telegram, or any other similar medium of communication. My preferred communication language is English.

---

2. Background:

The Large Hadron Collider (LHC) is the world's largest and most powerful particle accelerator, colliding protons at the highest energy ever recorded in the lab. The higher the luminosity (proportional to the number of collisions in a given amount of time), the more data the experiments can gather to allow them to observe rare processes. The High-Luminosity LHC (HL-LHC), which will be operational from the end of 2027, will allow the physicists to study about known mechanisms in greater detail, such as Higgs boson, and observe rare new phenomena with even greater precision. HL-LHC will deliver ten times more collisions than the LHC over its entire lifetime.

The Compact Muon Solenoid (CMS) is a general purpose experiment designed to observe a wide range of phenomena produced in high-energy collisions at the LHC. The CMS detector is built around a solenoid magnet (cylindrical coil of superconducting cable) that generates a magnetic field of 4 Tesla. The research at the CMS Experiment ranges from the study of the Standard Model (including the Higgs Boson) to searching for extra dimensions (including gravitons and tiny black holes) and particles that can make up dark matter. The CMS experiment has been designed with a 2-level trigger system, Level 1 (L1) Trigger and High Level Trigger (HLT). The current L1 trigger is fully hardware based with the maximum output rate of 100 kHz and microsecond latency. The High Level Trigger (HTL) is a software based sequence of reconstruction steps of increasing complexity keeping the event rate at about 800 Hz on average. The filtering process uses the full granularity data from the detector and the selection is based on sophisticated offline-quality reconstruction algorithms involving CPU expensive processes. The HLT algorithms use a dedicated version of the software framework and the reconstruction code used for offline reconstruction and analysis. The flexibility of the HLT allows the evaluation of Graphics Processing Unit (GPU), which not only provides faster and efficient event selection, but also includes the possibility of new complex triggers that were not previously feasible [7].

Machine Learning algorithms are advancing in high energy physics owing to their applications in particle and event identification, physics analysis, detector reconstruction, simulation and trigger. Most of the existing work on high energy particle reconstruction relies on the inputs provided by the Particle Flow (PF) algorithm used to convert detector level information to physics objects which sometimes fail to reconstruct or reconstruct with defects using this approach. The end-to-end deep learning technique unites deep learning algorithms and low level detector representation of collision events. The Graph Neural Networks for End-to-End Particle Identification with the CMS Experiment aims to extend the end-to-end deep learning project for high energy physics by using graph based approaches. Here we focus on benchmarking the results for identification of tau particles followed by a comparison with the CNN based approaches.

3.Project Introduction and Motivation:

An important part of new physics searches at the LHC involves classification of collision events or distinguishing potential signal events from those coming from background processes. Many machine learning techniques currently used rely on inputs provided by the Particle Flow (PF) algorithm used to convert detector level information to physics objects [21]. The Particle Flow algorithm has many advantages due to its ability to greatly reduce the size and complexity of particle physics data while providing a physically intuitive representation in physics analyses. However, there is some invariable loss of information from reducing the data set complexity. Despite the very high reconstruction efficiency of PF algorithms, some physics objects may fail to be reconstructed, are reconstructed imperfectly, or exist as fakes. Therefore, it is advantageous to consider end-to-end reconstruction that allows a direct application of machine learning algorithms to low-level data representation in the detector. While machine learning techniques based on traditional inputs like hand-engineered features or particle 4-momenta have been widely successful for understanding the Standard Model of Particle Physics, they are highly dependent on our ability to model all aspects of the particle decays and detector response. They potentially lose information in the process that may hinder more exhaustive searches for physics Beyond the Standard Model (BSM) [22].

Deep Learning algorithms are capable of not only training on high-level features, but also performing feature extraction [8]. Deep neural networks scale linearly with input data size [23] and are able to learn highly non-linear representations of the data. The End-to-End (E2E) Deep Learning for High Energy Physics project focuses on developing end-to-end deep learning algorithms with low level data representation [24, 25, 26] for particle and event classification (electrons vs. photons, quarks vs gluons, boosted top jets, taus, Higgs vs backgrounds, etc).

This project aims to implement graph neural network based identification of low momentum tau particles followed by the integration of the code with the CMS Software (CMSSW) Framework for end-to-end identification task. Graph neural networks have the ability to learn efficiently from position invariant unordered dataset. Convolutional Neural Networks (CNNs) on the other hand rely on the spatial features and learn better from local spatial associations than from the position invariant data. The tau particle jets are multichannel images with each channel providing a unique dimension to better represent the jet. In order to exploit the low-level information from all the sub detectors which provide various layers in the final jet image generated such as tracker layers (pT, dz, d0), calorimetric layers (ECAL, HCAL) and pixel layers (BPIX and FPIX layers), graph neural network have the potential to learn from the interplay of such layers by correlating and linking the nodes in unique ways with the aim of achieving optimal graph classification results. Each jet is generally treated as a graph with the channels as node features along with the coordinate positions of the nodes. During the tenure of this project, various graph attention networks, graph convolutions, graph pooling layers, node aggregations and clustering to determine the graph edges will be studied and implemented.

---

4. Previous Work

In [17] the proposed approach uses particle clouds to represent the multichannel jet images for quark-gluon tagging and top tagging. It follows the implementation of PointNet on point clouds which is a standard representation used for 3D objects. PointNet is a graph based approach which treats 3D objects as graphs (represented as point clouds) and implements ParticleNet inspired by PointNet to perform the tagging. In this project I propose to follow a similar representation but with different features. The convolution operation in ParticleNet is performed with the help of Edge Convolution block with dynamic graph updates i.e. each edge convolution block defines the edges of the graph using kNN clustering. This approach is called Dynamic Graph Convolution Neural Network (DGCNN). Figure 1 shows the difference between point clouds and particle clouds.

The study in [20] proposes a unique way to define graphs called neural relational inference for Variational AutoEncoders. The neural relational links proposed in the study might potentially perform better than conventional kNN based clustering for our case. Combining these with attention

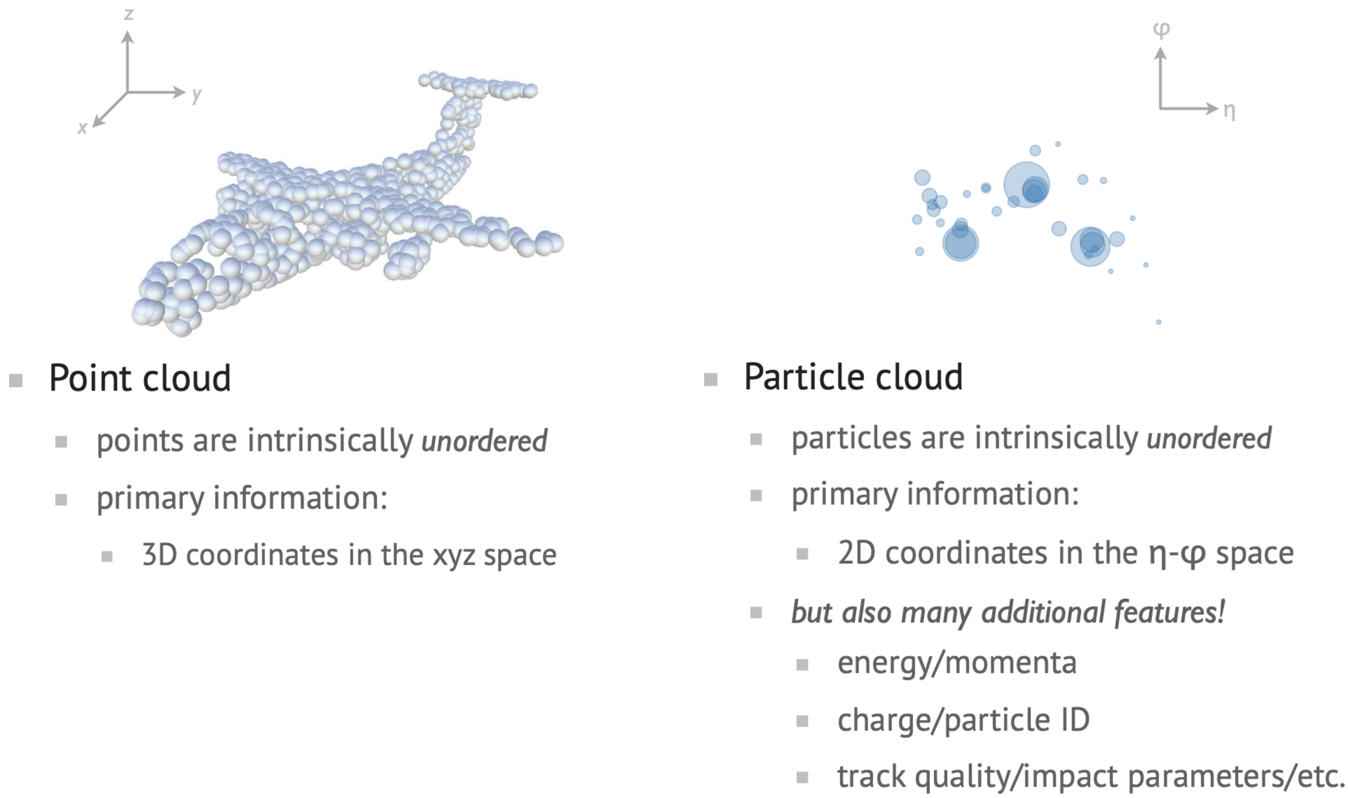


Figure 1. *(source)*

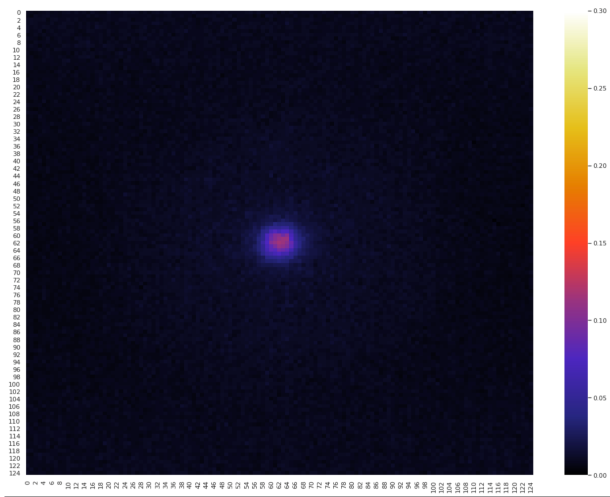
---

5. CMS Detector and Images:

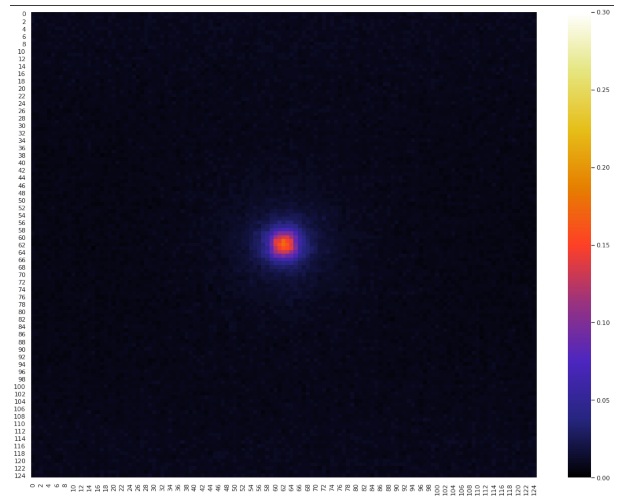
CMS is a multi-purpose detector composed of several cylindrical sub detector layers, with both barrel and end cap sections, encasing a primary interaction point. It features a large B=3.8 T solenoid magnet to bend the trajectories of charged particles that aid in pT measurement \cite{tdr-magnet}. At the innermost layers, close to the beamline, there is a silicon tracker used to reconstruct the trajectory of charged particles and find their interaction vertices. The tracker can be divided in two parts: the silicon pixel detector and silicon strip detector. The first silicon pixel detector is the innermost part and consists of three layers in the barrel region (BPIX) and three disks in the endcap region (FPIX). Each layer is composed of pixel sensors that provide a very precise position of the passage of a charged particle. The pixel detector provides crucial information for vertexing and track seeding. The outer part of the tracking system is composed of several layers of silicon strip. These provide a precise position in the phi coordinate, but not in the \eta coordinate. This is followed by the electromagnetic calorimeter (ECAL), made of lead-tungstate crystals, to measure the energy of electromagnetically interacting particles, then the hadronic calorimeter (HCAL), made of brass towers, to measure the energy of hadrons. These are surrounded by the solenoid magnet which is finally encased by the muon chambers to detect the passage of muons. Figure 2 shows the shower plots for ECAL, HCAL and one of the pixel layers of the Boosted Top Jets. The Tau particle jets have a similar dataset with multichannel images (8 to 9 channels).

Figure 2.

a) *ECAL overlay:*

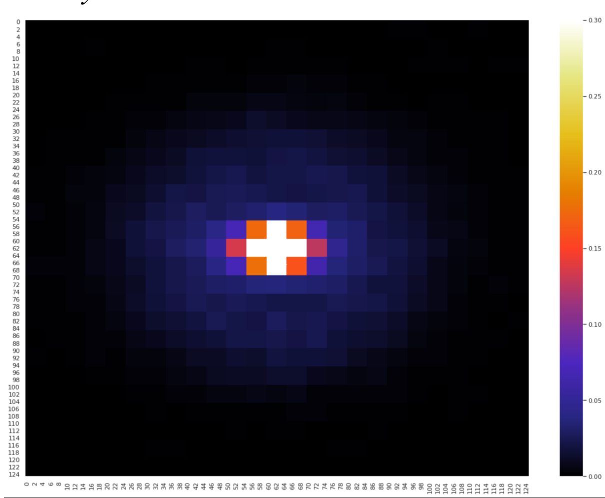


Boosted Top Jets (y=1)



Boosted Top Jets (y=0)

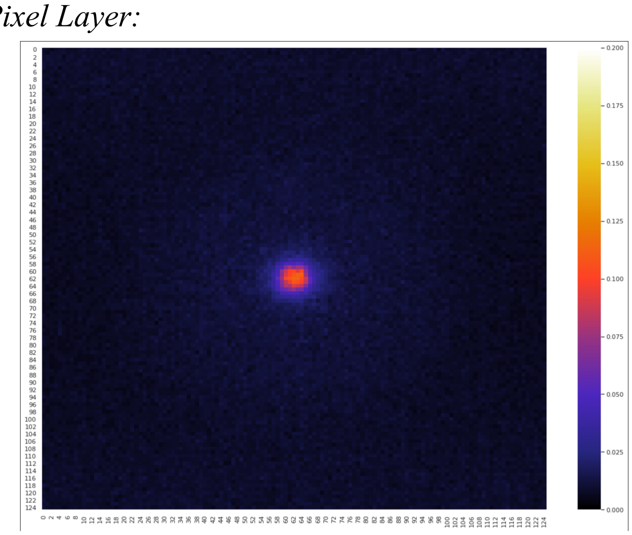
b) *HCAL overlay:*

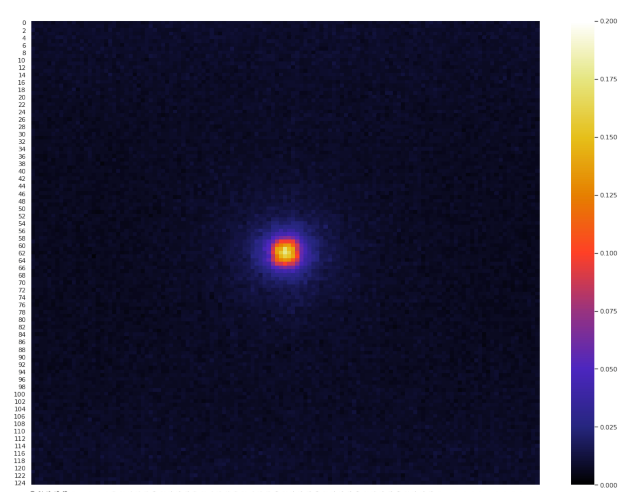


Boosted Top Jets (y=1)

Boosted Top Jets (y=0)

c) *1st Pixel Layer:*





Boosted Top Jets (y=1) *Boosted Top Jets (y=0)*

---

6. Deep Learning Architectures to be used:

There are various standard Deep Learning Architectures for Jet Tagging and reconstruction tasks mainly consisting of two types: CNN based architectures and GNN based architectures. Here, the GNN based approach is proposed below taking inspiration from [17, 18, 19, 20]. Combining attention methodology with various graph convolution operators, pooling techniques and clustering algorithms, I plan to propose the following approach:

6.1 Proposed Methodology and Model/s:

Ⅰ] **Preprocessing and EDA(Exploratory Data Analysis)**: Preprocessing essential for ensuring numerical stability in the optimizer, especially with large images and deep networks. EDA and Preprocessing are necessary for understanding the statistical distribution of the dataset and its features. For our project EDA would mostly consist of visualising the data as shower plots and particle clouds which would enable us to gain insights from the multichannel image dataset of Tau Particles as well as its corresponding graph representation.

Ⅱ] **Network Architectures**:

i) **Basic models**: As mentioned in the timeline we will first start by implementing basic architectures involving edge convolution layers and stacking of few edge convolutional blocks followed by fully connected layers. The 2D **Convolution block** for images represents a layer that can be used to detect spatial features in an image, either working directly on the image data or on the output of previous **convolution blocks** which can be composed of multiple Convolutional Layers followed by Pooling layers or Batch Normalisation layers and their various combinations and stacks. This analogy is greatly used for graph based neural networks too where edge convolution operation is able to detect spatial features for closely connected nodes. The connections are determined by clustering algorithms like kNN, spectral clustering, etc. Hence, using various variants of these building blocks and optimizing them for better accuracy (or any other chosen metric/s) we can develop a few basic models which perform satisfactorily on the dataset.

ii) **Moving towards Complex models**: Here, the complex attention based graph neural networks will be experimented. The models will follow the implementations described in neural relational inference for interacting systems, graph attention operators along with various convolution techniques like SAGE convolution, Dense SAGE convolution, Graph Convolution, Dense Graph Convolution, etc.

After selecting the potential architectures based on their performance on metrics like Accuracy, Mean Squared Error(MSE), Receiver Operating Characteristic Curve i.e. ROC and ROC AUC score i.e. ROC Area Under Curve, etc.

iii) **Ensembling Approach**: A successful approach in Deep Learning to reduce the variance of neural network models is to train multiple models instead of a single model and to combine the predictions from these models. This is called Ensembling Learning which not only reduces the variance of predictions but also can result in predictions that are better than any single model. Following is a summary of Ensembling Techniques:

- Varying Training Data: The data used to train each member of the ensemble can be varied.
  - K-fold Cross-Validation Ensemble
  - Bootstrap Aggregation (Bagging) Ensemble
  - Random Training Subset Ensemble

- Varying Models: Training the same under-constrained model on the same data with different initial conditions will result in different models given the difficulty of the problem, and the stochastic nature of the learning algorithm.
  - Multiple Training Run Ensemble
  - Hyperparameter Tuning Ensemble
  - Snapshot Ensemble
  - Horizontal Epochs Ensemble

- Vertical Representational Ensemble

- Varying Combinations: The predictions of the models can be combined in various ways and depending on the performance metrics, the best way can be figured out.
  - Model Averaging Ensemble
  - Weighted Average Ensemble
  - Stacked Generalization (stacking) Ensemble
  - Boosting Ensemble
  - Model Weight Averaging Ensemble

6.2 Loss Functions to be used:

- Mean Absolute Error Loss
- Mean Squared Error Loss
- Binary Cross-Entropy Loss
- Hinge Loss
- Squared Hinge Loss

6.3 Metrics to be used:

- Accuracy
- F1 score
- Precision and Recall
- Receiver Operating Characteristic (ROC) Curve
- ROC AUC (Area Under Curve)

6.4 Platforms to be used:

C/C++, Python, Jupyter Notebook, Tensorflow, Keras, Pytorch, Sklearn, CMSSW Framework.

6.5 Hardware to be used:

GPU/CPU

---

6. Timeline:

Duration	**Tasks**
April 13 - April 30	Researching more about the Project which involves understanding the Physics aspect of the Project better, studying the latest researches done in the field and studying previous work done by the organizations for the same. Reviewing some graph neural network literature.
May 1 - May 17	Familiarizing myself with Torch Geometric and various graph neural network algorithms. Trying to reproduce benchmarked results to understand the problem and various algorithms used to solve them.

May 17 - June 7	**Community bonding period** ● Getting involved with mentors and defining clear goals. ● Reviewing and analysing existing potential attention based graph neural networks.
June 7 - July 11	**First Coding Period**
Week 1 (June 7 - June 13)	
June 7 - June 13	● Exploring the dataset. Performing Exploratory Data Analysis to gain some insights about the data set. Visualising Tau Particle showers for all the channels to determine the potential channels for study. ● Exploring different Graph based Deep Learning architectures suitable for learning from the provided dataset.
Week 2 (June 14 - June 20)	
June 14 - June 15	● Preprocessing the dataset and performing the necessary conversions to represent the images as graphs thereby preparing it for deep graph models for the classification tasks.
June 15 - June 20	● Starting with the implementation of variants of the ParticleNet [17] and Dynamic Graph Convolutional Neural Networks (DGCNNs) and observe their performance. ● Improvising the architecture and optimize the models accordingly to get the best version and evaluate it using metrics like accuracy, mse (mean squared error) or ROC AUC score.
Week 3 (June 21 - June 27)	
June 21 - June 27	● Inspecting all the stages of typical Graph Neural Networks observed above by trying various pooling strategies, clustering techniques like knn based clustering, spectral clustering, neural relational inference, etc. ● Implementing various attention based mechanisms on the previously developed model as an attempt to improvise the DGCNN based algorithms.
Week 4 (June 28 - July 4)	
June 28 - July 2	● Implementing other Graph Attention Neural Networks with SAGE Convolution, Graph Convolution, etc.
July 3 - July 4	● Verifying the results and creating the necessary documentation.

Phase 1 Evaluation (July 12 - July 16)	● Testing the implemented models on the required platform and improving documentation and submission.
July 17 - August 15	**Second Coding Period**
Week 5 (July 4 - July 10)	
July 4 - July 10	● Understanding some additional complex architectures and implementing them. ● The architectures currently planned to use consist of Dense Graph Convolution, Dense SAGE Convolution, Dense GC Convolution for attempting to improvise the accuracy/ROC AUC score.
Week 6 (July 11 - July 17)	
July 11 - July 17	● Optimizing the best performing models by tuning their hyperparameters and improvising their architectures to improve their performance on the training and test datasets and taking care not to overfit the training dataset. The architecture/model showing better performance on the dataset will be chosen for further implementation. ● Testing the models on the standard benchmark datasets and comparing their performance with other standard implementations. ● Benchmarking the end-to-end inference on GPUs using the CMSSW framework
Week 7 and 8 (July 18 - July 31)	
July 18 - July 22	● Scaling the potential models to large datasets and training the models on the complete training dataset while validating on a separate dataset and optimising the inference for efficient performance on CMSSW framework using GPUs. ● Finally, testing the models on unseen dataset and benchmarking the results.
July 23	● Adding documentation and updating the repository.
July 24 - July 26	● Comparing the performance of the models researched above with the CNN based approaches. ● Working on deploying the trained models on CMSSW to facilitate end-to-end deep learning.
Final Evaluation (August 16 - August 23)	● Testing the implemented models on the required platforms and improving documentation and submission.

7. Deliverables:

- Exploratory Data Analysis (EDA) on the provided dataset. Study and inferences from the shower plots of all the channels averaged over a few thousands of jet samples.
- Code for Data preparation and preprocessing.
- Working Implementation of the promising models based on their performance on the training and validation dataset.
- Code for the training and testing the model including the optimisation for training via heterogeneous computing and data parallelisation for distributed training.
- Detailed analysis of the results of various notable implementations that were tried during the study.
- Integrating the end-to-end tau identification code with the CMSSW framework.
- Documentation summarising the study.

---

8. References:

[1] M. Andrews, M. Paulini, S. Gleyzer, and B. Poczos, "End-to-End Physics Event Classification with the CMS Open Data: Applying Image-based Deep Learning on Detector Data to Directly Classify Collision Events at the LHC", arXiv e-prints (Jul, 2018) arXiv:1807.11916, arXiv:1807.11916.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", in 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016 [10], pp.770–778.doi:10.1109/CVPR.2016.90.

[3] A. Rogozhnikova et al., "New approaches for boosting to uniformity", JINST 10 (2015).

[4] Sirunyan, A.M. et al. "Particle-Flow Reconstruction and Global Event Description with the CMS Detector." Journal of Instrumentation 12.10 (2017): P10003–P10003. Crossref. Web.

[5] Simonyan K., Zisserman A., Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv:1409.1556.

[6] Szegedy C., Going Deeper with Convolutions, arXiv:1409.4842.

[7] Kaiming H. et al., Deep Residual Learning for Image Recognition, arXiv:1512.03385.

[8] A. Klimentov, M. Grigorieva, A. Kiryanov, and A. Zarochentsev. 12(06):C06044–C06044, jun 2017.

[9] Steven Farrell et al. The HEP.TrkX Project: deep neural networks for HL-LHC online and offline tracking. EPJ Web Conf. , 150:00003, 2017.

[10] Thomas James, Level 1 Track Finding with an all FPGA System at CMS for the HL-LHC, arXiv:1910.12668v1 [physics.ins-det] 28 Oct 2019.

[11] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, End-to-End Event Classification of High-Energy Physics Data, J. Phys. Conf. Ser. 1085 (4) (2018) 042022. doi:10.1088/1742-6596/1085/4/042022.

[12] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, End-to-End Physics Event Classification with CMS Open Data: Applying Image-Based Deep Learning to Detector Data for the Direct Classification of Col- lision Events at the LHC, Comput. Softw. Big Sci. 4 (1) (2020) 6. arXiv:1807.11916, doi:10.1007/s41781-020-00038-8.

[13] M. Andrews, J. Alison, S. An, B. Burkle, S. Gleyzer, N. Narain, M. Paulini, B. Poczos, E. Usai, End-to End Jet Classification of Quarks and Gluons with the CMS Open Data, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, arXiv:1902.08276v2 [hep-ex] 23 Oct 2020.

[14] Valentina Gori, The CMS High Level Trigger, International Journal of Modern Physics: Conference Series, arXiv:1403.1500v1 [physics.ins-det] 6 Mar 2014.

[15] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, Exploring End-to-end Deep Learning Deep Learning Applications for Event Classification at CMS, EPJ Web of Conferences 214, 06031 (2019), CHEP 2018, https://doi.org/10.1051/epjconf/201921406031.

[16] The CMS Collaboration, Particle-flow reconstruction and global event description with the CMS detector, European Organisation for Nuclear Research (CERN), arXiv:1706.04965v2 [physics.ins-det] 9 Oct 2017.

[17] Huilin Qu, Loukas Gouskos, Jet Tagging via Particle Clouds, arXiv:1902.08570v3 [hep-ph] 30 Mar 2020.

[18] V. Mikuni, F. Canneli, ABCNet: an attention based network for particle tagging, Eur. Phys. J. Plus (2020) 135:463, https://doi.org/10.1140/epjp/s13360-020-00497-3.

[19] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio`, Yoshua Bengio, Graph Attention Networks, ICLR 2018, arXiv:1710.10903v3 [stat.ML] 4 Feb 2018.

[20] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, Richard Zemel, Neural Relational Inference for Interacting Systems, arXiv:1802.04687v2 [stat.ML] 6 Jun 2018.

[21] The CMS Collaboration, Particle-flow reconstruction and global event description with the CMS detector, European Organisation for Nuclear Research (CERN), arXiv:1706.04965v2 [physics.ins-det] 9 Oct 2017

[22] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, Exploring End-to-end Deep Learning Deep Learning Applications for Event Classification at CMS, EPJ Web of Conferences 214, 06031 (2019), CHEP 2018, https://doi.org/10.1051/epjconf/201921406031

[23] Steven Farrell et al. The HEP.TrkX Project: deep neural networks for HL-LHC online and offline tracking. EPJ Web Conf. , 150:00003, 2017

[24] Thomas James, Level 1 Track Finding with an all FPGA System at CMS for the HL-LHC, arXiv:1910.12668v1 [physics.ins-det] 28 Oct 2019

[25] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, End-to-End Event Classification of High-Energy Physics Data, J. Phys. Conf. Ser. 1085 (4) (2018) 042022. doi:10.1088/1742-6596/1085/4/042022.

[26] M. Andrews, M. Paulini, S. Gleyzer, B. Poczos, End-to-End Physics Event Classification with CMS Open Data: Applying Image-Based Deep Learning to Detector Data for the Direct Classification of Col- lision Events at the LHC, Comput. Softw. Big Sci. 4 (1) (2020) 6. arXiv:1807.11916, doi:10.1007/s41781-020-00038-8.

➢ **Reference Links:**

[i] https://arxiv.org/abs/1902.08276
[ii] https://arxiv.org/abs/1910.07029
[iii] https://arxiv.org/abs/1807.11916
[iv] https://iopscience.iop.org/article/10.1088/1742-6596/1085/4/042022
[v] https://arxiv.org/abs/1706.04965
[vi] https://machinelearningmastery.com/ensemble-methods-for-deep-learning-neural-networks/

[vi] https://medium.com/data-science-bootcamp/understand-the-softmax-function-in-minutes-f3a59641e86d
[vii] https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/
[viii]https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/
[ix]https://indico.cern.ch/event/766872/contributions/3357992/attachments/1831591/2999672/ParticleNet_IML_20190417_H_Qu.pdf