Project Proposal for Google Summer of Code 2023

Vision Transformers for End-to-End Particle Reconstruction for the CMS Experiment

Google Summer of Code @ ML4SCI Machine Learning for Science

Mentors:
Emanuele Usai
Ruchi Chudasama
Shravan Chaudhari
Sergei Gleyzer
Eric Reinhardt
Samuel Campbell

Devdeep Shetranjiwala
Dhirubhai Ambani Institute of Information and Communication Technology - DAIICT, India.

Contents

Personal Details

Name: Devdeep Shetranjiwala

Email: deveep0702@gmail.com, devdeepshetranjiwala@gmail.com

LinkedIn: https://www.linkedin.com/in/devdeep-shetranjiwala-4290b21ba/

Mobile: +91 9265816294

Country of Residence: India

Timezone: Indian Standard Time (UTC +5:30)

Github Handle: Devdeep-J-S

Language: English

University: Dhirubhai Ambani Institute of Information and Communication Technology - DAIICT, India.

Major: Information and Communication of Technology with a minor in CS.

Introduction

The field of computer vision has been dominated by Convolutional Neural Networks (CNNs) for years. However, Vision Transformers (ViT) have recently revolutionised this field through their self-attention mechanism, which has shown excellent results on various tasks. This project aims to develop a transformer-based algorithm for classifying high-energy particles in multi-channel simulated images from the calorimeter.

Synopsis

The field of computer vision has been revolutionised by the introduction of Vision Transformers (ViT) architecture, which has shown excellent results on various tasks.

This project aims to use ViT-based architectures to classify high-energy particles in the CMS experiment. The data consists of multi-channel simulated images from the calorimeter, and the aim is to classify the collision events accurately.

Previous attempts at this task have been made using CNN-based architectures, but we aim to explore the potential of ViT-based architectures, which have been shown to be much more powerful than vanilla CNNs.

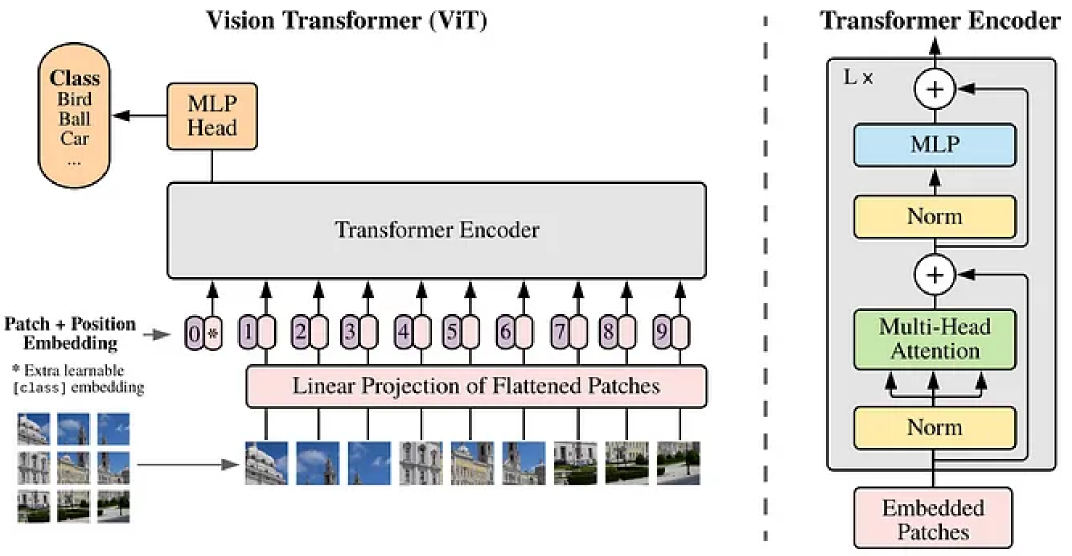


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Credit: Paper 1

The main goal of this project is to develop a transformer-based algorithm for classifying high-energy particles that can outperform existing CNN-based models. To achieve this, we will experiment with various ViT-based architectures and advanced techniques contemporary deep-learning researchers use.

We will also generate a large amount of data and train the models on GPUs to speed up the training process. The expected outcome of this project is a highly accurate transformer-based image classification model for high-energy particle images, which can be used in particle physics experiments to improve our understanding of the subatomic world.

We will follow a well-planned implementation plan to achieve our goals involving several milestones. We will start by researching the current state-of-the-art ViT-based architectures and related techniques.

Next, we will generate a large amount of data using the CMS experiment's simulated images. We will then develop and test various ViT-based models on GPUs and fine-tune them for improved accuracy. We will also perform benchmarking to compare the performance of our models with existing CNN-based models. Finally, we will document our findings and make our code publicly available to facilitate further research in this area.

MileStones

I. Project Goals

1. The main goal of this project is to develop a transformer-based image classification architecture that can outperform CNNs in the classification of high-energy particle images. The project aims to experiment with various ViT-based architectures to achieve this goal.
2. Study the current literature on ViT and its application in computer vision and particle physics.
3. Collect and preprocess the data consisting of multi-channel simulated images from the calorimeter.
4. Implement ViT-based architectures and experiment with different ViT-based models to find the best possible classification accuracy.
5. Explore advanced techniques contemporary deep-learning researchers use to improve the model's classification accuracy.
6. Perform extensive benchmarking of the ViT-based model on CPU and GPU to compare its performance with the CNN-based model.

II. Expected Results

1. The expected result of this project is to develop a ViT-based image classification architecture that can outperform CNNs in the classification of high-energy particle images.
2. The project aims to achieve competitive results using transformers, and the transformer's performance will be benchmarked on CPU and GPU.

III. Outline of Approach and Implementation plan

Research and Understanding:

- Conduct a literature review on the current state-of-the-art methods for high-energy particle classification using deep learning.
- Study the architecture and functioning of ViT-based models and their advantages over traditional CNN-based models.
- Understand the principles of self-attention and its role in improving the performance of ViT-based models.
- Go in-depth into PyTorch and Tensorflow and their implementation of ViT-based models.

Data Generation:

- Generate simulated data for high-energy particle showers in the CMS experiment using Geant4 software.
- Create multi-channel images from the calorimeter data by projecting the energy deposits onto the detector layers.
- Simulate various high-energy particle collisions, including electrons, photons, and hadrons, to create a diverse dataset.

- Augment the dataset by adding various noises and distortions to the images.

Data Preprocessing:

- Normalize the pixel values of the images to a standard scale and remove any noise or artefacts present in the images.
- Divide the dataset into training, validation, and testing sets with a proper ratio.
- Split the data into batches for efficient training and evaluation.
- Implement data augmentation techniques such as rotation, flipping, and scaling to generate more training data.

ViT-based Model Implementation:

- Implement ViT-based architectures such as ViT-B/32, ViT-L/16, and ViT-H/14 using PyTorch and TensorFlow.
- Train the ViT-based model on the preprocessed data using a suitable optimiser and loss function.
- Experiment with various hyperparameters such as the number of attention heads, hidden layers, and learning rate to optimise the classification accuracy.
- Monitor the training process using metrics such as accuracy and loss to ensure the model converges.

Hyperparameter Tuning:

- Perform hyperparameter tuning using grid search or random search algorithms to find the best set of hyperparameters for the ViT-based model.
- Evaluate the model's performance on the validation set with different combinations of hyperparameters.

- Select the best model based on its classification accuracy and inference time.

Comparison with CNN-based Models:

- Implement traditional CNN-based models such as ResNet and VGG for high-energy particle classification.
- Train the CNN-based models on the same preprocessed data and evaluate their performance on the validation set.
- Compare the performance of ViT-based models with CNN-based models using extensive benchmarking on CPU and GPU.
- Measure the models' inference time, memory usage, and accuracy to determine which model is better suited for the task.

IV. Optional Milestones

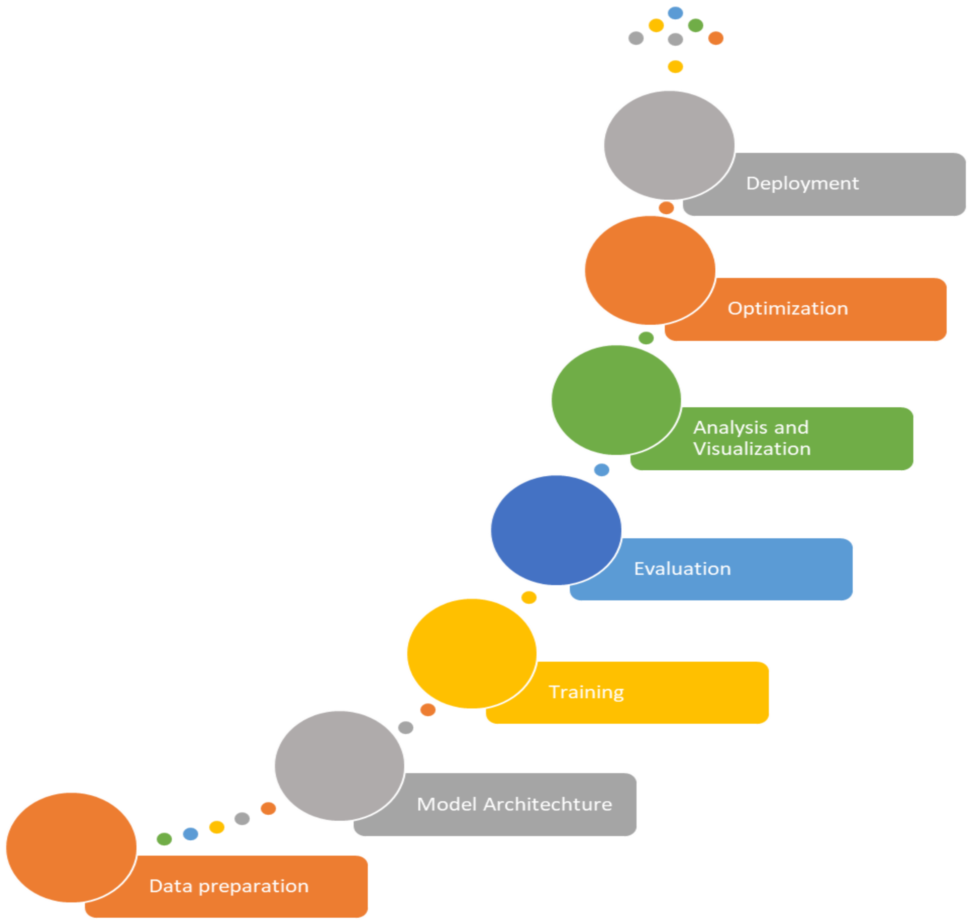
Implementation of Additional Advanced Techniques:

1. Additional advanced techniques such as data augmentation, transfer learning, and adversarial training will be implemented to improve the ViT-based model's performance further. Data augmentation techniques such as rotation, flipping, and scaling will be used to generate more training data.
2. Transfer learning techniques will leverage pre-trained models for improved performance. Adversarial training techniques will improve the model's robustness against adversarial attacks.

Development of a Web-Based User Interface:

1. A web-based user interface will be developed to allow users to upload images and obtain particle classification results using the ViT-based model. The user interface will be developed using web technologies such as React and JavaScript.

2. The interface will be integrated with the ViT-based model to allow users to obtain real-time particle classification results.



V. Reference Paper

Paper 1:

[2010.11929] An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Paper 2:

[1807.11916] End-to-End Physics Event Classification with CMS Open Data: Applying Image-Based Deep Learning to Detector Data for the Direct Classification of Collision Events at the LHC

Paper 3:

[1902.08276] End-to-End Jet Classification of Quarks and Gluons with the CMS Open Data

VI. Other Deliverables

Weekly report and blog

Timeline

Period	Deliverables	Specific Goals
(Communicating Bonding period) *May 4 - May 28*	➔ Community bonding report ➔ Literature review, data sources, problem analysis ➔ Familiarize with the CMS experiment and the task at hand.	➔ Communicate and bond with students and mentors ➔ Explore the CMS experiment and high-energy particle classification. ➔ Set up the development environment and become familiar with the tools and frameworks required for the project.
(Week 1) *May 28 - June 3*	➔ Literature Review and Data Generation	➔ Generate simulated data for high-energy particle showers in the CMS experiment ➔ Create multi-channel images from the calorimeter data ➔ Study the literature on ViT-based models and self-attention ➔ Learn PyTorch and TensorFlow in depth ➔ Develop a preprocessing data pipeline to clean and normalise the data
(Week 2) *June 4 - June 10*		

(Week 3) *June 11 - June 17*	➔ ViT-based Model Implementation	➔ Implement ViT-based architectures such as ViT-B/32, ViT-L/16, and ViT-H/14 using PyTorch and TensorFlow ➔ Train the ViT-based model on the preprocessed data using a suitable optimiser and loss function ➔ Experiment with various hyperparameters such as the number of attention heads, hidden layers, and learning rate to optimise the classification accuracy ➔ Monitor the training process using metrics such as accuracy and loss to ensure the model converges
(Week 4) *June 18 - June 24*		
(Week 5) *June 25 - July 1*	➔ Hyperparameter Tuning	➔ Perform hyperparameter tuning using grid search or random search algorithms to find the best set of hyperparameters for the ViT-based model ➔ Evaluate the model's performance on the validation set with different combinations of hyperparameters
(Week 6) *July 2 - July 8*		
Mid-term Evaluation *(July 10 - July 14)*		

(Week 7) *July 9 - July 15*	➔ Evaluation of models	➔ Select the best model based on its classification accuracy and inference time ➔ Develop a testing pipeline to evaluate the final model's performance on the test set
(Week 8) *July 16 - July 22*		
(Week 9) *July 23 - July 29*	➔ Comparison with CNN-based Models	➔ Implement traditional CNN-based models such as ResNet and VGG for high-energy particle classification ➔ Train the CNN-based models on the same preprocessed data and evaluate their performance on the validation set ➔ Compare the performance of ViT-based models with CNN-based models using extensive benchmarking on CPU and GPU ➔ Measure the model's inference time, memory usage, and accuracy to determine which model is better suited for the task
(Week 10) *July 30 - August 5*		
(Week 11) *August 6 - August 12*	➔ Optional Milestones	➔ Implement transfer learning techniques to leverage pre-trained models for improved performance ➔ Implement adversarial training techniques to improve the model's robustness against adversarial attacks.

(Week 12) *August 13 - August 19*		➔ Develop a web-based user interface using React and JavaScript to allow users to obtain real-time particle classification results.
(Final Week: Submit final work product to final mentor evaluation) *August 20 - August 27*	➔ Final evaluation ➔ Project submission	➔ Evaluate the final product and ensure that all project goals have been met. ➔ Submit project report and code to assigned mentor ➔ Project presentation
Final Evaluation *August 28 - September 4*		
Results are announced *September 5*		

Prerequest challenges

Github repo link: https://github.com/Devdeep-J-S/Vision-Transformers-CMS

Learning from challenge

Task 1: The main challenge of this task was to design a deep learning model that could accurately classify electrons and photons based on their energy and time information. I overcame this challenge by experimenting with various neural network architectures, regularisation techniques, and hyperparameter tuning to achieve the best possible performance. This task helped me to gain a deeper understanding of deep learning techniques and their applications in particle identification.

Task 2: The main challenge of this task was to design a convolutional neural network (CNN) architecture that could accurately classify quarks and gluons based on their energy information. I overcame this challenge by experimenting with various CNN architectures, regularisation techniques, and hyperparameter tuning to achieve the best possible performance. This task helped me to gain a deeper understanding of CNNs and their applications in particle identification.

Task 3: The main challenge of this task was to train a transformer model to achieve performance similar to that of CNNs in Task 1. I overcame this challenge by better understanding transformer models, attention mechanisms, and transfer learning techniques. This task helped me appreciate the advantages and limitations of using transformer models for image classification tasks and compare their performance to traditional CNNs.

Overall, these tasks helped me develop a well-rounded understanding of deep learning techniques and their applications in particle identification. They also gave me valuable skills and knowledge for working on the Vision Transformers for End-to-End Particle Identification with the CMS Experiment project.

Biographical Information

I. Educational background

I am a 3rd-year student pursuing a Bachelor's in Information and Communication Technology with a minor in Computational Science from Dhirubhai Ambani Institute of Information and Communication Technology-DAIICT.

I am hardworking when it comes to academics and hold an 8 CGPA.

- EXTRACURRICULAR

IEEE IAS STUDENT BRANCH COMMITTEE DAIICT: Webmaster

February 2022 – Present | Gandhinagar, India

- Worked on websites and apps for numerous IEEE events, like i.Fest, Summer school, etc.
- Led team of web developers to develop events websites.

II. Technical Interest

I am passionate about software development and research, niching down to web development and machine learning and its applications.

I have completed several courses in machine learning, including ones that cover classification, regression, and clustering algorithms.

I have hands-on experience implementing these algorithms using Python and libraries like Scikit-learn, PyTorch and TensorFlow. I have also worked on several projects that involve processing and analysing large datasets, which would be useful for this project.

Regarding programming skills, I am proficient in Python, C, C++, and JavaScript and have experience with data manipulation and visualisation libraries like Pandas and Matplotlib.

I am comfortable working with Jupyter notebooks and have experience with version control tools like Git. I am also good at web development using the Django-python framework. I am looking forward to growing and developing these skills more and more by contributing to this organisation.

III. Software Development Experience/ Projects

1. EVENT MANAGER WEB APP

Tech used: Python-Django, Twilio API, Google SMTP, SQLite

- Event Manager is a tool to record and retrieve the data of events effectively.
- An application with CRUD operations and user authentication.
- The web app has features like host sign-in, email and SMS notifications, Database Management and easy-to-use UI.
- Impact : Won WOC4.0 Web development Compition 1st Prize.

2. RESTAURANT AUTOMATION

Tech used: C, C++, Shell Script

- The problem is an advanced version of the Dining-Philosopher problem.
- Developed a code to handle the production and distribution of a product and applied it to manage a Restaurant.
- A program that uses the concepts like Concurrent Programming and Process management.

3. Woc 5.0 - WEATHER PREDICTION APP

Tech used: Python, Sklearn, Tensorflow, Flask

- Implemented an app to predict the weather by analysing data and categorising them into specific labels. Such a problem can be solved using various Machine learning techniques.

4. LABOR LIST AND WAGES MANAGEMENT TOOL

Tech used: Python-Django, PostgreSQL, HTML, CSS, Bootstrap

- A tool to effectively record and retrieve the data of Labor and Supervisors.
- An application with CRUD operations.
- The web app has features and database management.

5. OPEN SOURCE - VIDEO STREAMING - RTSP CLIENT-SERVER

Tech used: Python, Tkinter

- Implementation of a video streaming server and client that communicate using the Real-Time Streaming Protocol (RTSP) and Realtime Transfer Protocol (RTP).

IV. Communication

Working Hours - I am flexible with my timings and can work dedicatedly for hours together. Also, I will ensure that GSoC will be my only significant commitment this summer and give it my full attention.

Communication preferences - I am comfortable with any platform and can adjust whatever suits the mentors.

In case of work delay due to personal or other reasons, I assure you that I will work longer and more efficiently to complete the backlog in the available time. I plan to write weekly progress blogs on my medium publication to establish a systematic solution to ensure I get all week's work. I would share the same with my mentors while actively communicating with them.

V. Resume

GSOC_Resume_Devdeep.pdf

VI. Engagements during Summer

I have no commitments in the summer. I'll be staying back home for most of it. I have mentioned my typical working hours above; on average, I can spend 35-40 hours per week on the project.

Note: If the project goes to 350 hr length due to the complexity of the task or additional future milestones implementation, I am also comfortable with it.