Google Summer of Code Project Proposal

Exploring Vision Transformers for Anomaly Detection

Kalyaan Rao
raokalyaan2001@gmail.com
+917304613970

Thadomal Shahani Engineering College, Mumbai
Bachelor of Engineering in Artificial Intelligence and Data Science

India / GMT + 5:30
Github

Contents -

1 Overview

1.1 Project Abstract

Convolutional Neural Networks have been dominant in Computer Vision applications for a decade. Today, they are being outperformed and replaced by Vision Transformers (ViT) with a higher learning capacity. The fastest ViTs are essentially a CNN/Transformer Hybrid, combining the best of both worlds.

This project aims to demonstrate the potential of Vision Transformers when it comes to building a robust and efficient model for anomaly detection or binary classification from the image data. We aim to illustrate that the ViT are on par with the classical CNNs on image detection when it comes to training from scratch with distillation, in the lower accuracy regime and for lower network complexities optimised for Edge devices.

1.2 Benefits to the Community

Even though ViTs outperform CNNs in multiple cases, their dominance has not yet been established. Every little step counts. The open source community will benefit both from the study conducted and the code produced during the project. The community can use the source codes as a starting point for further investigation.

2 Goals and Deliverables

2.1 Goals

The goal of this project is to establish the performance of Vision transformers in comparison to the SoTA CNN. The main goal will be to build a model for binary classification as provided in the description page with the dataset as provided. In addition, the project would also like to test out the various other Vision transformers as suitable for the requirement of our dataset.

2.2 Deliverables

Considering the goals and the time available, I divide the deliverables into two categories : required and optional. Required refers to the deliverables that must be finished during the stipulated time frame of GSoc and are the main aims of the project whereas optionals means the deliverables if the time allows for the same.

Required

1. Python code(s) that contains:
    1.1. Code to train the ViT Model (including the fine tuning) and the various different architectures (encoding methods and variational ansatz).

1.2. Code to train several SoTA CNN models as a benchmark for comparison with the VITs.

2. Train the VIT models including fine tuning using tutorials like Jupyter and Colab Notebook(s) that use the python code above and compare them to the SoTA CNN Models like ResNet, AlexNet, VGG and so forth.

3. A white paper containing a summary of the previous related works, an explanation of the different ViT model implementation and how it is trained and fine tuned. This white paper also doubles as the final GSoC report.

Optional

If the project's final results are considered publish - worthy by the mentors, the white paper will be written into a research paper and will be submitted to a conference, poster, talk/workshop or journal.

2 Related Works and Recommendations

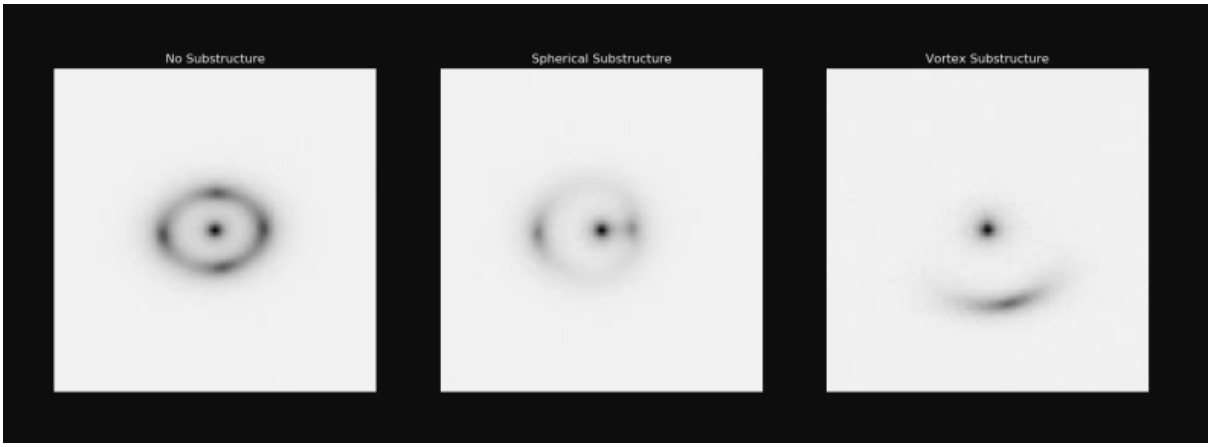
2.1 Related Works

The project can be divided into two parts mainly being the dataset and the architecture of the ViT that can be used. This section will discuss some related works and papers to support the claim.

2.1.1 Dataset and Preprocessing

Dataset

For the given project, dataset is given which consists of a set of simulated gravitational lensing images with and without sub structures.



Image(s) showing simulated gravitational lenses
With and without sub structures.

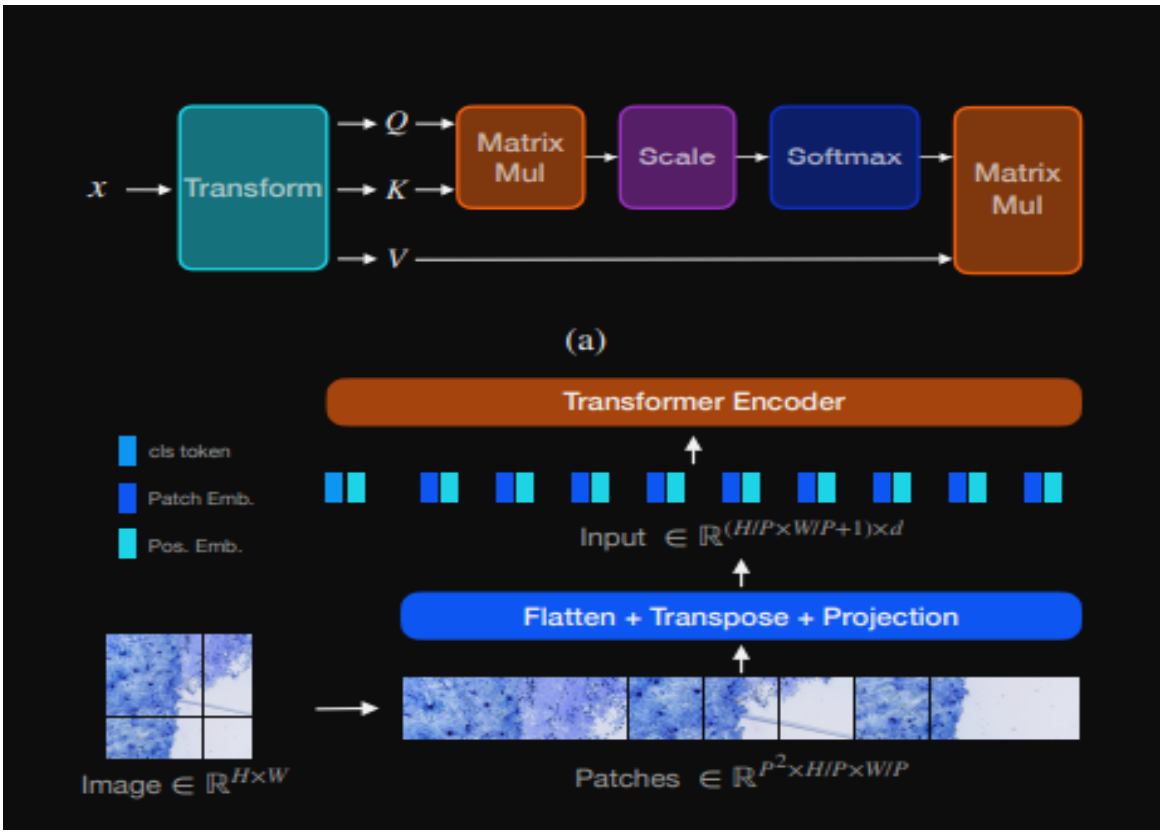
Stephen et al. have tried to use the SoTA image classification convolutional models such as ResNet50, VGG16 and ImageNet to classify the simulated images of gravitational lenses and so forth.

Augmentation

During the training, we can make use of augmentation and transformations. This can be achieved through translations and rotations by up to 90 degrees. These all constitute invariant transformations with respect to the underlying sub structure that allow the network to learn the images through seeing the images in different augmentations. [1]

Architecture

In contrast to the traditional pre-trained models that have been used for classifying the images, we aim to use a vision transformer that does the same work as mentioned in the paper [2].



Vision Transformer

We can make use of distillation through attention for training data efficient image transformers. Knowledge distillation refers to the process of transferring the knowledge from a large unwieldy model or set of models to a single smaller model that can be practically deployed under real-world constraints. Essentially, it is a form of model compression that was first successfully demonstrated by Bucilua and collaborators in 2006 [4]

Knowledge distillation is performed more commonly on neural network models associated with complex architectures including several layers and model parameters. Therefore, with the advent of deep learning in the last decade, and its success in diverse fields including speech recognition, image recognition, and natural language processing, knowledge distillation techniques have gained prominence for practical real-world applications.

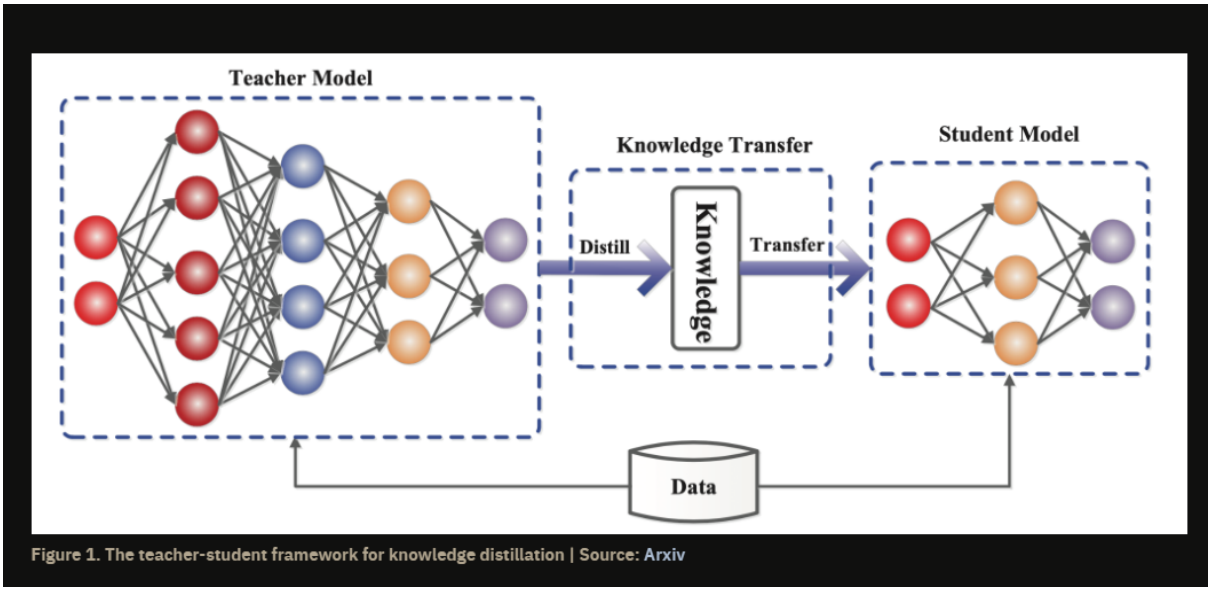


Figure 1. The teacher-student framework for knowledge distillation | Source: Arxiv

With the advent of Vision transformers, it can be inferred that they work best on large datasets. With our project mainly focusing on a relatively niche dataset, training is insufficient. Enormous training images must be used to achieve excellent performance. Therefore, a data-efficient transformer solution is urgently needed. Training on small datasets with ViT increases dependence on regularisation or data augmentation techniques. If trained on massive datasets, ViTs can achieve competitive performance despite lacking intrinsic properties of convolutions. However, due to the stacking of self-attention modules on multiple heads, ViTs remain prohibitively expensive. ViT compression has only recently come into existence, and the prior works

mainly focus on a few aspects, in contrast to the extensive literature and remarkable success of CNN compression.

Classification Part

We can divide the vision transformer into two parts as mentioned above: a student and a teacher. The teacher can be trained seperately as a classifier head with any one of the state of the art CNN transfer learning methods, followed by a student that is used with the help of a distillable vision transformer.

DViT uses a patch-based approach to process image data, where the input image is divided into small patches, which are then processed by a series of self-attention blocks. The self-attention mechanism allows the network to attend to different parts of the image when processing each patch, enabling the model to capture spatial relationships between features in the image.

The key innovation of DViT is its ability to be distilled into a smaller model, without significant loss in performance. This is achieved by training a larger teacher model and then using it to guide the training of a smaller student model. The student model is trained to mimic the activations and outputs of the teacher model, effectively transferring the knowledge learned by the larger model to the smaller one.

Optimizer

Adam optimizer is a widely used optimization algorithm for training Vision Transformer (ViT) models. It is an adaptive learning rate method that computes individual learning rates for each parameter based on estimates of the first and second moments of the gradients. The basic idea behind Adam is to use a combination of momentum and adaptive learning rates to converge quickly to a good solution while also avoiding overfitting.

When using the Adam optimizer for training a ViT model, it is important to use an appropriate learning rate schedule and weight decay regularisation. It is often recommended to use a warmup strategy during the initial training phase, where the learning rate is gradually increased to its maximum value. Additionally, weight decay regularisation is commonly used to prevent overfitting and improve generalisation performance.

2.2 Recommendations

Considering all things mentioned in the section 2.1, I suggest trying the following methods

1. **First Approach** - Try fitting the baseline ViT for the dataset chosen to obtain the binary classification of images present in the dataset. Given the amount of data a traditional ViT is trained on, it may take up to 5 - 6 hours to implement the same.
2. **Second Approach** - Use a Distillable ViT for the dataset chosen by implementing a teacher and a student model for the same. The teacher being a fine tuned classifier with a state of the art CNN model and the student being a Distilled Transformer
3. **Third Approach** - If time persists, explore the possibilities of dangling with Convolutional transformers.

If there are no changes made after further discussion with mentors during the project, the course will follow the following schedule -

- Dataset - 128 x 128 images simulated from the gravitational lensing dataset.
- Two approaches will be tested, the first being a traditional Vision Transformer and the other one being a more robust and efficient Distillable Vision Transformer for image efficient classification.
- The optimizer used will be Adam, if for any reason it does not work there can be other optimizers that can be used provided in the framework of choice.
- The performance metric of the models that will be compared includes the training and testing loss curve,the training and testing accuracy curve, and the final models' ROC curve and AUC score. This project's main numerical result is the accuracy and AUC score average from the best model of the Vision Transformers.
- The exact details of hyperparameters, e.g., number of epochs, batch size, number of convolution and classification layer, learning rate, will vary as they will be fine-tuned during model development.

I am recommending these because I have tried them myself. When I worked on my thesis, I tried many different approaches to classify MNIST dataset, and these approaches were the ones that worked.

3 Timeline and Research Plan

The timeline written here is based on the general timeline by GSoC.

3.1 Application and Review Plan (March 20 - April 4)

Try and get used to the architecture of Vision Transformer. Try all the various frameworks for the given and implement a somewhat near code to show the mentors and ask for feedback.

3.2 Community Bonding(May 4 - May 28)

The works that need to be done during this period are categorised into three. The timeline may overlap between categories as the works between categories are closely related.

Week 1-2: Planning (May 4 – 20 May) Get feedback from mentors about the proposal. Discuss with mentors the proposal's high-level decisions (goals, priority and deliverables, meeting schedule) that need to be changed and modify them accordingly. After that, discuss with mentors the more low-level decisions (dataset and preprocessing, the whole pipeline including optimizers, analyses, coding training procedures) to the minute details and modify the timeline and research plan accordingly.

Week 2-3: Early Works (24 May – 7 June) Prepare the working environment and repository. Obtain and clean the dataset. Do all necessary preprocessing (normalisation) in Jupyter Notebook.

Do the first approach - if the evaluation metrics are deemed satisfactorily by the mentors, analyse the performance. This will be done in the Jupyter Notebook. Make a short report on the performance, and discuss it with mentors at the first meeting in the ”Coding” period. This short report will be reused as the final report's material. The first approach finishes here. The ”Coding” period will be fully spare for the second approach.

Week 3: Bonding - I would also like to know more about mentors' and organisation's works other than GSoC. My dream is to become a researcher in artificial intelligence, and this is a precious opportunity to hear from people who are more experienced in the field.

3.3 Coding (7 June - 16 August)

Week 1(7 June - 14 June)

Work on preprocessing the dataset given for the project and try all the possible transformations and augmentations that may yield the best result or fit in accordance with the evaluation metrics

Week 2(14 June - 21 June)

Start implementing the main architecture and baseline for the vision transformer. Experimenting with the different knowledge distillation techniques using the teacher and student model. To choose an efficient teacher model, make use of different SoTA CNN models to find out which one is the best in terms of the goal of the project.

Week 3 (21 June – 26 July) & Evaluations (13 July – 17 July)

This is the central part of the project, developing the ViT model. I can spend about a month coding and developing good-performing models for my project. Considering that the training pipeline code is already done in week 2, I believe a month should also be enough for this project.

The strategy is to focus on one type of convolution neural network as a teacher model per week. Every week, a combination of one convolutional neural network followed by distillation will be trained and fine-tuned. The fine-tuning here includes hyperparameters (number of epochs, batch size, learning rate), normalisation, optimizer, number of convolution layers, pooling layers, etc.

At the end of the week, if the model shows good performance with a potential to be better, we will spend another week fine-tuning the model.
If it doesn't show good performance, we will start next week with another convolution circuit ansatz. The decision is made at the weekly meeting with mentors.

We will stick to this strategy and training procedures discussed during Community Bonding, but it is hard to plan for certain every step in the model development since there are many moving parts in it, and every decision depends on the numerical results of the model. Note that after every training, the configurations, the trained model, and that model's statistical performance will be saved. Write a short evaluation for mentors during the "Evaluations" period.

Week 8(26 July - 2 August)

After we obtain the best configuration of the ViT model, we will focus on that model only and compare it to the performances of the classical models. Some of the classical models may need to be re-trained as the number of trainable parameters of that model may be too small or too large compared to the best CNN model.

We want to compare models with about the same number of trainable parameters to ensure fairness.
Visualise the comparison, e.g., the training and testing loss curve, the training and testing accuracy curve, and the final models' ROC curve and AUC score. Tabulate the average accuracy and AUC score of the best model from both CNN and ViT methods.

Write performance comparison analysis from the visualisations, e.g., which model performs better, how the number of parameters and layers affect CNN performance, what could have been done to potentially make the ViT model better. This analysis will be reused as the final report's material.

Week 9 (2 August – 9 August) Clean up the Jupyter Notebook for ViT model training. Write comments and markdowns explaining everything done in the notebook. This notebook will serve as documentation and tutorial for people interested in trying ViT, similar to what I did for the evaluation test but in more detail.

Make any changes to the performance analysis made in week 8 as per feedback.

Week 10 (9 August – 16 August)

Write the final report/white paper. This report will contain:

- Short Introduction (project summary, motivation, related works, dataset). Contents in the project proposal can be reused.
- Explain the best ViT model's implementation that is found during model development and how it was trained in minute details. Most of the contents are actually already written in the tutorial-like documentation made in week 9, so this is just a matter of moving and reformatting them.
- Performance comparison analysis made in week 8 (no reformatting as it can be directly reused).

- If the first approach gives good results, then the short report that was made will be integrated here as well.
- Conclusions and future work recommendations.

Discuss with mentors whether the white paper will be written as a research paper and published or not. Because the date of acceptance of publishers may not be in the GSoC timeline, I am willing to spend time outside GSoC to write the research paper and make the submission.

3.4 Students Submit Code and Final Evaluations (16 August - 23 August)

Make any changes to the tutorial-like docs made in week 9 and the final report made in week 10 as per feedback. Write final evaluation for mentors.This is a spare week in case some works in the timeline need more time, emergency events, etc.

4 About Me

I'm currently a pre final year engineering student from the Artificial Intelligence and Data Science department at Thadomal Shahani Engineering College. The department's vision has allowed me to expand my knowledgeable horizon across multi disciplinary fields such as Data Science, Machine Learning, Computer Vision and NLP. I have good proficiency in python and C++.

I am the kind of person who always tries my best and puts 100% of my focus into things that I am passionate about. During college studies, machine learning and quantum computing are the two fields that excite me the most. In the next section, I explained briefly how I got into those fields and my experience in them.

4.1 Machine Learning and So on

I have always been fascinated with the working of the universe ranging from the smallest of quanta to the massive scales of the universe. From watching VSauce back in 7th grade about black holes and PBS Space time for the cosmos, I have come a long way. Interstellar is one movie that has inspired me to take up this project. After getting into college, my passion for artificial intelligence rekindled with cosmology. I have recently also started to learn the basics of Quantum Computing as well, to further my knowledge.

In conjunction with the coursework of my department, I have taken a Coursera course in applying machine learning techniques for Computer Vision problems. I learned to use deep learning & gradient boosting for particle identification and Bayesian optimization for tracking system optimization. I have also taken deep learning courses in GANs, computer vision, and sequence models both in college and online.

5 Other Commitments

- I have my term end semester exams from the 10th of May to 22nd of May and also accounting for the trivial stuff mentioned in the timeline during this period, I can devote about 2 - 3 hours per day during the course of my exam.
- After my exam, I can devote about 6 - 8 Hours per day, since I have mostly nothing to do during the last semesters of my coursework as the majority of the part has been completed prior.