THE NOOBS

Neha Gaonkar (Team Leader)

Aum Thaker

INTRODUCTION

Exploring Mars' diverse terrains is crucial for understanding its geology and potential for life. This project aims to automate Mars terrain classification using the ViT Transformer, a state-of-the-art deep learning model. By leveraging this model, we seek to enhance the accuracy and efficiency of terrain classification compared to traditional methods.

Objective

1.) Develop a robust model that correctly predicts the terrain based on the input image provided.
2.) Preprocess the input data to feed the model.

PROCEDURE

1.) **Data Exploration**

1.) on viewing the dataset we found that the dataset was very imbalanced, with the following classes:

Other : 3651

Crater : 1062

Bright dune : 597

Slope streak : 335

Swiss cheese : 223

Dark dune : 216

Spider : 66

Impact ejecta : 51

2.) **Model Selection**

1.) Our choice for model was a transformer from which we shortlisted 2 models: ViT(vision transformer) and swin(sliding window transformer).

2.) The reason for choosing a transformer model was:

I. Transformers capture long range dependencies and global context.

II. Transformers are robust to spatial feature arrangement.

III. Transformers have shown remarkable results on several benchmarks such as ImageNet, COCO,LUNA etc..

Model	ImageNet(top1)	COCO(mAP)	LUNA16
ViT			
ViT-L	85.2	47	96.5

ResNet			
ResNet-50	76	38	93
ResNet-101	77.3	40.4	94.5
ResNet-152	78.3	41.2	94.8
EfficientNet			
EfficientNet-b0	77.1	33.8	93.5
EfficientNet-B7	84.7	42.	96.

IV. From the above comparison a transformer model.

2.) Our next goal was to select a transformer model. We shortlisted to 2 choice of models ie. ViT and swin.

3.) Below are our observations after comparing the 2 models:

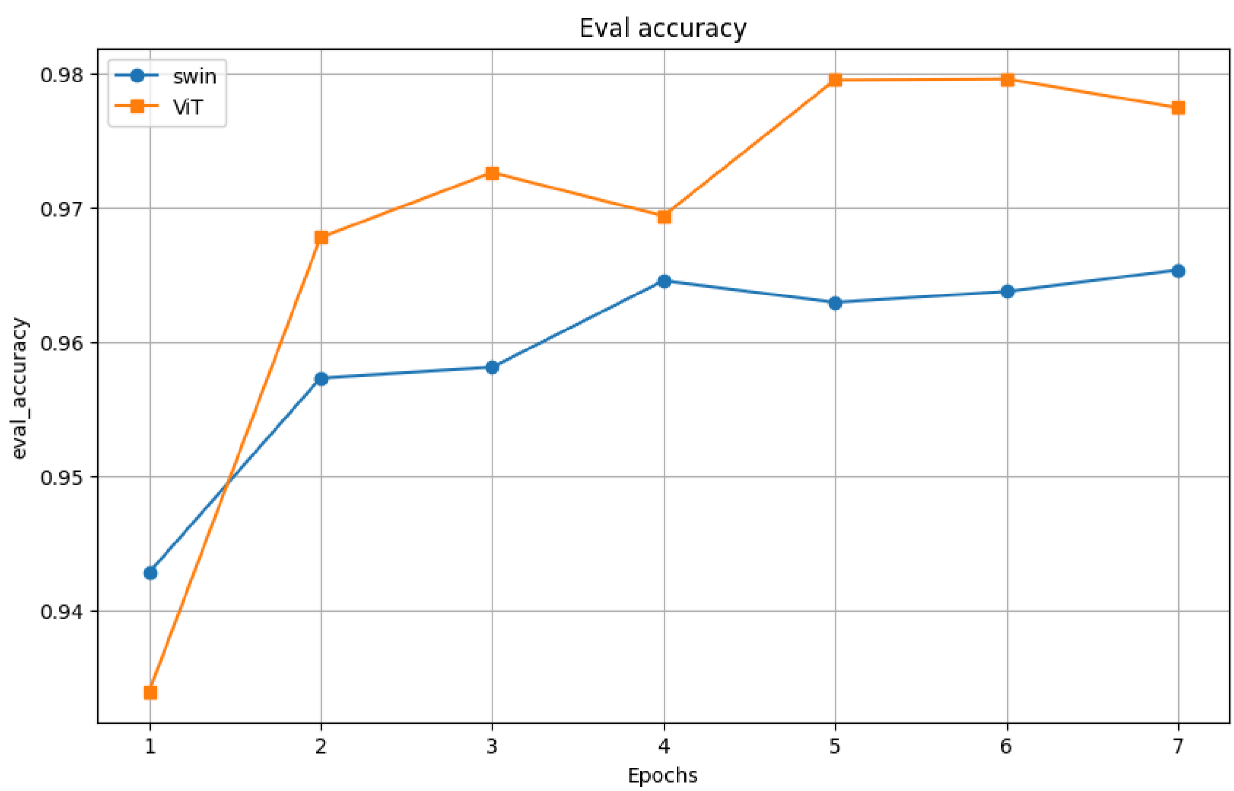


Fig 1. Evaluation accuracy (y-axis) vs. epochs (x-axis) for ViT Transformer and Swin Transformer models.

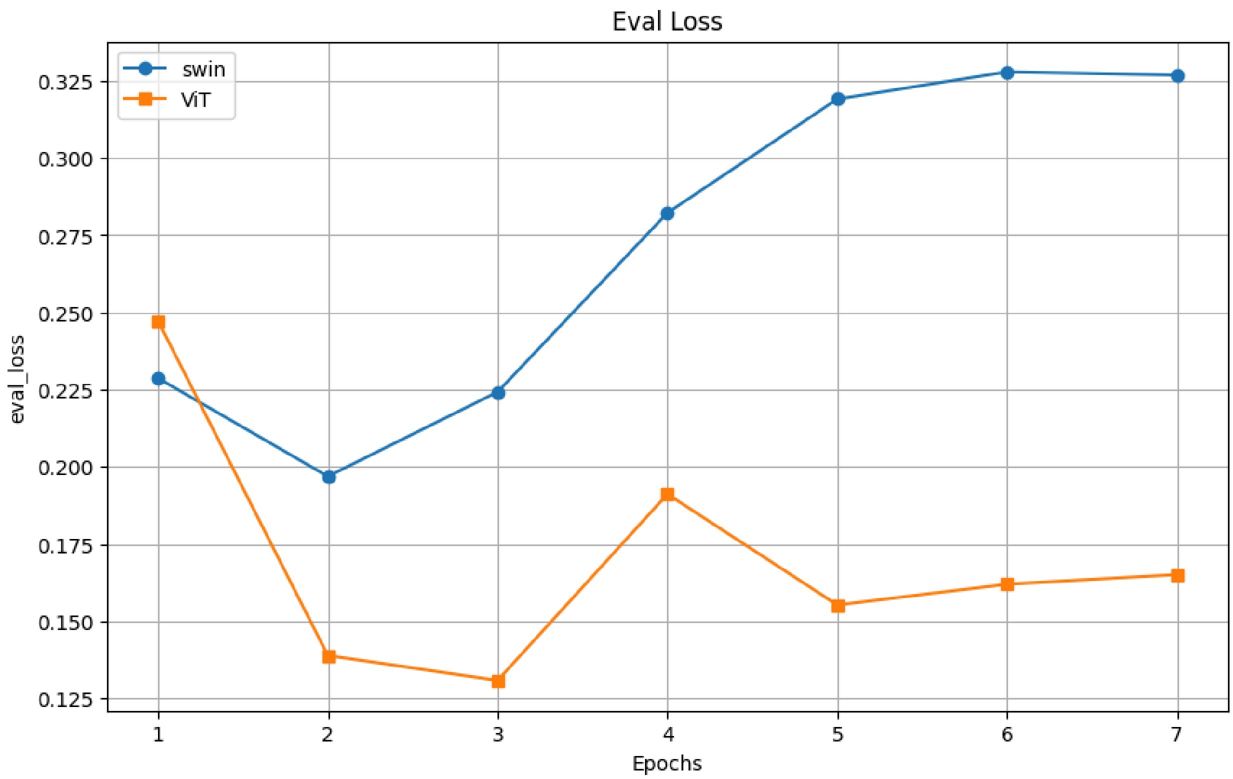


Fig 2. Evaluation loss (y-axis) vs. epochs (x-axis) for ViT Transformer and Swin Transformer models.

Note: the above is a summarization of multiple experiments conducted using different hyperparameters and the best results were taken for comparison.

4.) Also swin has at times failed to capture long context dependencies.

5.) Based on the results we decided to go with the ViT model.

3.) Data preprocessing

1.) We used the AutoImageProcessor class of the transformers library to perform preprocessing tasks.

2.) We first converted the images to RGB channels and then to pixel_values.

3.) The final output of the preprocessing stage is pytorch tensor pixel_values and its corresponding labels.

4.) Fine-tuning

1.) We fine-tuned ViT after several iterations to select optimal parameters.

2.) We selected the hyperparameters based on our experimental results and reading documentation.

3.) At optimal parameters we achieved the following results:

Eval_loss:0.02

Eval_accuracy:0.9693795326349718

Eval_precision:0.9698723433684743

Model Details

Our model has been finetuned on the base model -

google/vit-base-patch16-224

The hyperparameters used for finetuning were:

Train batch size: 8

Gradient checkpointing: True

Train epochs: 5

Learning rate: 6e-5

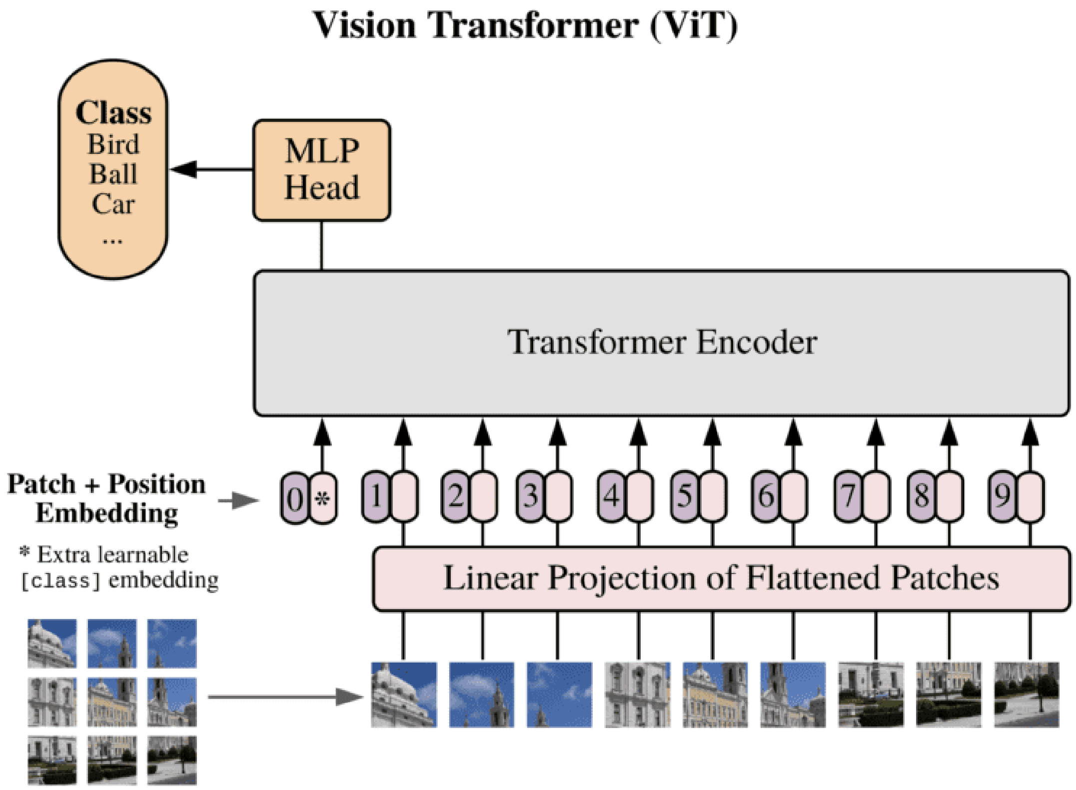
Lr scheduler:linear

Metric for best model: accuracy

Eval strategy:epoch

Save strategy: epoch

Model Architecture



1. **Hierarchical Structure**: Uses a multi-scale approach to capture global features by processing images through multiple stages.

2. **Linear Complexity**: Achieves efficiency with linear complexity relative to image size, enabling effective handling of high-resolution images.

3. **Multi-Stage Design**: Progressively reduces spatial resolution and increases feature dimensionality through several stages.

4. **Patch Merging**: Aggregates features from adjacent patches to build richer, more comprehensive representations.

Challenges:

1. **Data Imbalance**: To handle that we tried data-augmentation of minority class, adding class-weights so as to penalize the majority class but all these had their own drawbacks, so we decided to use precision along with accuracy to deal with data-imbalance.
2. **Computational resources:** All the cloud services have a limited amount of time for free gpu(T4 in case of google-colab/kaggle). To manage our sessions we tried to run as much code locally as possible and used the cloud services only for training purposes.

Result:

1.) For the aforementioned hyperparameters the following metrics were obtained:

Eval_loss : 0.2000304013490677

Eval_accuracy : 0.9693795326349718

Eval_precision : 0.9698723433684743

Training_loss : 0.09432842338097192

Total_flos : 1.9219046894272512e+18

Total_steps : 6200

Environmental Impact

The net estimated CO2 emission using the Machine Learning Impact calculator scale is around 11.32kg of CO2.

Developed by: Neha Gaonkar , Aum Thaker

Model Type: Transformer

Hardware Type: RTX 30

Compute Region:South Asia

Carbon Emitted: 11.32kg(aggregate)