Mid-Evaluation Report

Vision Transformers Project

1. Introduction

This report presents an overview of the academic progress and technical preparation completed during the first half of the Vision Transformers project. The objective of this phase was to build a solid foundation in deep learning by covering essential programming skills, mathematical principles, and neural network architectures. The learning journey so far has included Python programming, mathematical concepts for machine learning, classical neural networks, convolutional models, and sequence-based architectures, all of which are critical for understanding and implementing Vision Transformer models.

2. Project Overview and Objectives

Vision Transformers mark a significant transition from convolution-centric vision models to attention-based architectures originally developed for sequential data. To work effectively with such models, it is necessary to first acquire a strong grasp of the underlying prerequisites. These include:

- Proficiency in Python programming
- Mathematical understanding of optimization techniques and loss functions
- Knowledge of neural network training procedures
- Familiarity with convolutional neural networks for visual representation learning
- Conceptual understanding of sequence models such as RNNs, LSTMs, and GRUs

Accordingly, the initial weeks of the project were structured to systematically build these foundational skills before moving on to transformer-based vision architectures.

3. Week 0: Python Programming and Data Handling Fundamentals

The focus of Week 0 was to establish a strong programming base using Python, the primary language for machine learning research and implementation.

Python Fundamentals

Core programming concepts such as loops, conditional statements, functions, lists, and dictionaries were revised and applied. Emphasis was placed on writing modular and reusable code, which is essential for scalable machine learning workflows.

Object-Oriented Programming

Python classes were introduced to understand how models can be structured using object-oriented principles. A small illustrative model was implemented to demonstrate how parameters, forward computation, and outputs can be organized within a class-based framework.

Numerical Computing with NumPy

NumPy was used extensively for vectorized computation and matrix operations. Since neural network training relies heavily on linear algebra operations, gaining fluency with NumPy arrays and broadcasting rules was a key outcome of this week.

Data Analysis and Visualization

Pandas was introduced for handling datasets, while Matplotlib was used for plotting and visualizing data trends, outputs, and losses.

Assignment 1

The first assignment was completed using Google Colab and focused on reinforcing Python programming and numerical computation skills. Tasks included designing a `DataSample` class, implementing logical operations through custom functions, and performing NumPy-based array manipulations such as slicing and vectorized calculations. Basic data handling with Pandas and visualization with Matplotlib were also incorporated.

Outcome:
The assignment was successfully completed, strengthening confidence in Python programming, numerical computation, and data processing—skills that form the backbone of subsequent deep learning tasks.

4. Week 1: Mathematical Foundations and Core Machine Learning Concepts

Week 1 concentrated on the mathematical principles that govern machine learning models and optimization algorithms.

Mathematical Concepts

Fundamental topics such as vectors, matrices, gradients, and basic probability were studied to understand how learning algorithms operate mathematically.

Activation Functions and Softmax

The importance of activation functions in introducing non-linearity was examined. The softmax function was explored in the context of multi-class classification.

Loss Functions

Different loss functions for regression and classification tasks were analyzed to understand how model performance is quantified.

Regression Models

Linear regression was introduced as a baseline supervised learning technique, followed by logistic regression for binary classification.

Optimization Techniques

Gradient descent, stochastic gradient descent, and related optimization strategies were studied to understand how models minimize error during training.

Unsupervised Learning

K-means clustering was covered as an example of unsupervised learning, along with evaluation metrics for assessing clustering quality.

Assignment 2

The second assignment involved implementing regression models and optimization algorithms in Google Colab. The focus was on translating mathematical theory into practical implementations, analyzing loss behavior, and evaluating model performance.

Outcome:
The assignment was completed correctly and helped solidify understanding of optimization dynamics, loss minimization, and model evaluation—providing a strong mathematical foundation for neural networks.

5. Week 2: Convolutional Neural Networks

Week 2 introduced convolutional neural networks, which traditionally form the backbone of computer vision systems and serve as an important comparison point for Vision Transformers.

Convolution Operations

Convolution and cross-correlation were studied to understand how local patterns are extracted from image data.

Convolutional Layers

The internal working of convolution layers was explored, including how filters produce feature maps from input images.

CNN Terminology

Concepts such as stride, padding, kernel size, receptive field, and channels were discussed to understand architectural design decisions.

Assignment 3

This assignment focused on implementing convolution operations and analyzing how different hyperparameters affect output feature maps. Visualizations were used to better understand spatial feature extraction.

Outcome:
The assignment was completed successfully and provided clear insight into CNN mechanics, helping build intuition about feature extraction in vision tasks.

6. Week 3: Recurrent Neural Networks and Sequence Modeling

Week 3 shifted focus to sequence modeling, which is conceptually crucial for understanding attention mechanisms and transformer architectures.

Motivation for Sequence Models

The need to model sequential data such as text and time-series was discussed, emphasizing temporal dependencies.

Recurrent Neural Networks

The structure and functioning of RNNs were studied, including hidden states and information flow across time steps.

Language Modeling

Language models were introduced as a key application, and perplexity was studied as a performance metric.

Data Representation

One-hot encoding was explored as a method for representing categorical sequence data.

Training Challenges

Backpropagation through time and issues such as vanishing and exploding gradients were analyzed. Gradient clipping was introduced as a stabilization technique.

Assignment 4

The fourth assignment involved implementing RNN-based models in Google Colab, analyzing training behavior, and addressing instability issues using gradient clipping.

Outcome:
The assignment demonstrated stable sequence modeling and strengthened understanding of temporal learning mechanisms.

7. Week 4: Advanced Recurrent Models and Encoder–Decoder Architectures

Week 4 focused on improving sequence modeling through advanced architectures that overcome the limitations of basic RNNs.

Limitations of Vanilla RNNs

Problems related to long-term dependency modeling were examined.

LSTM and GRU

LSTM networks were studied in detail, including gate mechanisms that control information flow. GRUs were explored as a simpler yet effective alternative.

Deep and Bidirectional RNNs

Stacked and bidirectional RNN architectures were analyzed to understand how representational power can be enhanced.

Encoder–Decoder Framework

The encoder–decoder architecture was studied as a foundational idea that leads directly to attention mechanisms and transformer models.

Assignment 5

This assignment involved implementing LSTM, GRU, and encoder–decoder models and analyzing their effectiveness in modeling long-range dependencies.

Outcome:
The assignment was completed successfully and strengthened readiness for understanding transformer-based architectures.

8. Role and Individual Contribution

Throughout this phase of the project, I independently studied the required theoretical concepts and completed all programming assignments. I worked extensively with Python, NumPy, Pandas, and Matplotlib, implemented core machine learning algorithms, and explored neural network training techniques. I also analyzed both convolution-based and sequence-based models to understand their strengths and limitations, which is essential for transitioning to Vision Transformer models.

9. Progress Summary and Learning Outcomes

By the mid-point of the project, I had developed a strong conceptual and practical understanding of programming, optimization methods, neural networks, CNNs, and sequence models. These components collectively establish a solid foundation for studying attention mechanisms and Vision Transformer architectures.

10. Conclusion

The mid-evaluation phase has effectively prepared me for advanced work on Vision Transformers. The structured, week-wise approach ensured gradual progression from fundamentals to more complex architectures. With these prerequisites in place, I am well-equipped to move forward into transformer-based vision modeling in the subsequent stages of the project.