

# Mid-Evaluation Report

## Vision Transformers Project

### Table of Contents

1. Introduction
2. Project Overview and Objectives
3. Week 0: Foundations of Python and Data Handling
4. Week 1: Mathematical and Machine Learning Fundamentals
5. Week 2: Convolutional Neural Networks (CNNs)
6. Week 3: Recurrent Neural Networks (RNNs) and Sequence Modeling
7. Week 4: Modern Recurrent Architectures and Encoder–Decoder Models
8. Progress Summary and Learning Outcomes
9. Current Status and Next Steps
10. Conclusion

## 1. Introduction

This mid-evaluation report documents the learning progress, conceptual foundations, and technical preparation undertaken for the Vision Transformers project. The project aims to build a strong understanding of deep learning fundamentals, neural network architectures, and sequence modeling techniques that are essential prerequisites for comprehending and implementing Vision Transformer (ViT) models. The work completed so far spans Python programming, mathematical foundations for machine learning, classical neural networks, convolutional models, and recurrent architectures.

## 2. Project Overview and Objectives

Vision Transformers represent a shift from traditional convolution-based vision models to attention-based architectures originally designed for sequence modeling. To effectively work on this project, it is necessary to develop:

- Strong programming proficiency in Python
- Mathematical understanding of optimization and loss functions
- Knowledge of neural networks and training techniques
- Familiarity with CNNs for visual feature extraction
- Understanding of sequence models (RNNs, LSTMs, GRUs) that motivate transformer-based approaches

The first half of the project therefore focuses on building these prerequisites in a structured, week-wise manner.

### **3. Week 0: Foundations of Python and Data Handling**

Week 0 focused on establishing a strong programming foundation using Python, which is the primary language used for machine learning and deep learning research.

#### **Python Basics: Loops, Functions, Lists, and Dictionaries**

Core Python constructs such as loops and conditional statements were used to control program flow. Functions were implemented to modularize code and improve reusability. Lists and dictionaries were studied as essential data structures for handling datasets, parameters, and model outputs.

#### **Classes and a Tiny Model Example**

Object-oriented programming concepts were introduced through Python classes. A simple model example was implemented using classes to understand how model parameters, forward passes, and outputs can be encapsulated in a clean structure. This laid the foundation for understanding neural network modules later.

#### **NumPy**

NumPy was used for numerical computation, vectorized operations, and matrix manipulation. Understanding NumPy arrays is critical since neural network operations such as matrix multiplication and broadcasting rely heavily on it.

#### **Matplotlib and Pandas**

Matplotlib was used for data visualization, enabling graphical interpretation of trends, losses, and outputs. Pandas was introduced for data loading, preprocessing, and analysis, which is essential when working with large datasets in vision tasks.

#### **Assignment 1: Python Programming and Numerical Computing (Colab Submission)**

As part of Week 0, I completed Assignment 1 using Google Colab to apply core Python and numerical computing concepts in a practical setting. The assignment focused on strengthening programming fundamentals essential for machine learning workflows.

Key tasks included implementing a **DataSample class** to understand data encapsulation, designing **custom Python functions** for logical operations, and performing **NumPy-based array manipulations** such as slicing, diagonal extraction, and vectorized computations relevant to neural network operations. Basic **Pandas data handling and Matplotlib visualization** were also used to analyze and present results.

**Result:** The assignment was completed successfully, with all components producing correct outputs. It reinforced proficiency in Python programming, numerical computation, and data handling, forming a strong foundation for subsequent machine learning and Vision Transformer-related work.

## 4. Week 1: Mathematical and Machine Learning Fundamentals

Week 1 focused on the mathematical backbone of machine learning models and optimization techniques.

### Basics of Mathematics for Machine Learning

Key concepts such as vectors, matrices, gradients, and basic probability were covered. These concepts form the basis for understanding how neural networks learn from data.

### Activation Functions and Softmax

The role of activation functions in introducing non-linearity into neural networks was studied. Softmax was explored in detail as a normalization function used in multi-class classification.

### Regression and Classification Losses

Different loss functions were studied for regression and classification problems, highlighting how learning objectives are defined for different tasks.

### Linear and Logistic Regression

Linear regression was introduced as a foundational supervised learning algorithm, followed by logistic regression for binary classification problems.

### Measuring Error, Gradient Descent, and Optimizers

Error measurement techniques and loss minimization were studied using gradient descent, stochastic gradient descent, and common optimization strategies.

### K-Means Clustering and Evaluation Metrics

K-means clustering was explored as an unsupervised learning method, along with evaluation metrics to assess model performance.

### Assignment 2: Mathematical Foundations and Core ML Algorithms (Colab Submission)

As part of Week 1, I completed Assignment 2 to apply mathematical concepts and core machine learning algorithms through hands-on implementation in Google Colab. The assignment focused on translating theoretical understanding into practical model behavior.

Key components included implementing **linear and logistic regression**, analyzing **loss functions for regression and classification**, and applying **gradient descent-based optimization** to minimize errors. The assignment also involved experimenting with

**evaluation metrics** to assess model performance and understanding the behavior of learning algorithms.

**Result:** The assignment was completed successfully, with correct implementation of regression models and optimization procedures. It strengthened conceptual clarity in loss minimization, optimization dynamics, and model evaluation, providing a solid mathematical foundation for deeper neural network and Vision Transformer-related learning.

## 5. Week 2: Convolutional Neural Networks (CNNs)

Week 2 focused on convolutional neural networks, which form the traditional backbone of computer vision systems and provide important context for Vision Transformers.

### Convolution and Cross-Correlation

The convolution and cross-correlation operations were studied to understand how local spatial features are extracted from images. This helped build intuition for feature extraction in vision tasks.

### Convolution Layers

The structure and working of convolution layers were explored, including how filters operate on input images to generate feature maps.

### Terminologies in Convolution

Key terminologies such as stride, padding, kernel size, receptive field, and channels were covered to understand architectural design choices in CNNs.

### Assignment 3: Convolution Operations and CNN Fundamentals (Colab Submission)

As part of Week 2, I completed Assignment 3 to gain hands-on experience with convolutional operations and core CNN concepts using Google Colab. The assignment emphasized understanding how spatial features are captured from image-like data.

The tasks involved implementing **convolution and cross-correlation operations**, analyzing the effect of **kernel size, stride, and padding**, and observing how these parameters influence output feature maps. This practical work helped bridge theoretical concepts with actual computational behavior in CNNs.

**Result:** The assignment was completed successfully, with correct implementation and visualization of convolutional operations. It strengthened understanding of spatial feature extraction and CNN mechanics, providing a strong foundation for comparing convolution-based models with transformer-based vision architectures.

## 6. Week 3: Recurrent Neural Networks (RNNs) and Sequence Modeling

Week 3 introduced sequence modeling techniques, which are conceptually important for understanding attention mechanisms and transformer architectures.

### Goal of Working with Sequences

The motivation for modeling sequential data such as text and time series was discussed, focusing on capturing temporal dependencies.

### Sequence Modeling and Latent Autoregressive Models

Sequence modeling approaches and latent autoregressive models were studied to understand how past information influences future predictions.

### Language Models and Perplexity

Language models were introduced as a key application of sequence modeling. Perplexity was studied as an evaluation metric to measure model effectiveness.

### Recurrent Neural Networks

The structure and working of RNNs were explored, including hidden states and temporal information flow.

### One-Hot Encoding

One-hot encoding was studied as a method for representing categorical and sequential data.

### Backpropagation and Gradient Clipping

Backpropagation through time and gradient clipping were studied to address training instability in RNNs.

### Assignment 4: Sequence Modeling with RNNs (Colab Submission)

As part of Week 3, I completed Assignment 4 to implement and analyze sequence modeling concepts using recurrent neural networks in Google Colab. The assignment focused on understanding temporal dependencies and training dynamics in sequential data.

Key tasks included implementing **RNN-based models**, applying **one-hot encoding** for sequential inputs, analyzing **language modeling behavior**, and studying training challenges such as **vanishing and exploding gradients**, addressed using **gradient clipping**.

**Result:** The assignment was completed successfully, demonstrating correct sequence modeling behavior and stable training with gradient clipping. It strengthened understanding

of temporal learning mechanisms, directly contributing to readiness for attention-based and transformer-style architectures.

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

Week 4 focused on advanced recurrent architectures that address the limitations of plain RNNs and directly motivate transformer-based models.

### Limitations of Plain RNNs

Issues such as vanishing and exploding gradients were analyzed to understand why basic RNNs struggle with long-term dependencies.

### Long Short-Term Memory (LSTM)

LSTM networks were studied in detail, including the forget gate, input gate, and output gate, which enable controlled information flow across time steps.

### Gated Recurrent Unit (GRU)

GRUs were explored as a simplified alternative to LSTMs, offering comparable performance with fewer parameters.

### Deep and Bidirectional RNNs

Deep RNNs and bidirectional RNNs were studied to understand how stacking layers and processing sequences in both directions improves representational capacity.

### Encoder–Decoder Architecture

The encoder–decoder framework was explored as a foundational concept leading to attention mechanisms and transformer architectures.

### Assignment 5: Advanced RNNs and Encoder–Decoder Modeling (Colab Submission)

As part of Week 4, I completed Assignment 5 to gain hands-on experience with advanced recurrent architectures and sequence-to-sequence modeling in Google Colab. The assignment focused on understanding how gated mechanisms and encoder–decoder structures improve long-range dependency modeling.

Key tasks included implementing and analyzing **LSTM and GRU-based models**, experimenting with **deep and bidirectional RNNs**, and studying the working of **encoder–decoder architectures** for sequence transformation tasks.

**Result:** The assignment was completed successfully, demonstrating stable training and improved handling of long-term dependencies. It strengthened conceptual and practical understanding of architectures that directly precede and motivate transformer-based models, including Vision Transformers.

## 8. Role and Contribution

Throughout this phase of the Vision Transformers project, I independently studied and implemented the foundational concepts required for deep learning and vision-based models. I worked extensively on Python programming, numerical computation using NumPy, and data handling with Pandas and Matplotlib. I explored core machine learning algorithms, implemented regression and optimization techniques, and gained hands-on understanding of neural network training.

I studied convolutional neural networks to understand traditional vision pipelines and analyzed their limitations in capturing global context. Additionally, I worked on sequence modeling concepts using RNNs, LSTMs, and GRUs, which helped me build intuition for attention-based architectures. This systematic preparation has enabled me to confidently transition toward understanding and implementing Vision Transformer models in the later stages of the project.

## 9. Progress Summary and Learning Outcomes

By the mid-point of the project, a strong conceptual and practical foundation has been developed in programming, optimization, neural networks, CNNs, and sequence models. These topics collectively prepare the groundwork for understanding attention mechanisms and transformer-based vision models.

## 10. Conclusion

This mid-evaluation phase has successfully established the essential prerequisites for the Vision Transformers project. The systematic, week-wise learning approach has ensured conceptual clarity and technical readiness for advanced topics in transformer-based vision modeling.