Mid-Term Evaluation Report

Winter Project: Vision Transformer (ViT)

Name: Yogesh Verma
Roll Number: 231201

---

1. Introduction

This project focuses on studying and understanding **Vision Transformers (ViT)**, a modern deep learning architecture for computer vision tasks. To develop a strong foundation for this work, the initial phase emphasizes learning essential concepts of machine learning and deep learning, progressing from basic models to advanced architectures such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs)**.** While CNNs have traditionally served as the backbone of vision-based models due to their ability to capture local and spatial features, recent advancements demonstrate that Transformer-based architectures—originally designed for sequence modeling—can effectively process image data as well. In this context, the project is structured around five machine learning assignments completed during the course, which progressively cover fundamental to advanced topics including neural networks, CNNs, batch normalization, RNNs, and sequence generation. Each assignment integrates theoretical understanding with practical implementation using Python and Google Colab, collectively building the conceptual and technical foundation required to study Vision Transformers.

---

2. Week 1: Foundations of Machine Learning and Neural Networks

Week one focused on establishing core concepts in machine learning and neural networks.

Python and essential scientific libraries were used for implementation, with NumPy for numerical computation, Pandas for data preprocessing, and Matplotlib for visualization.

Supervised learning was studied as a function approximation problem, where a model learns a mapping f(x)=yf(x) = yf(x)=y. Neural networks enable the modeling of complex, non-linear relationships through stacked non-linear transformations.

Activation functions such as Sigmoid, Tanh, ReLU, and Softmax were explored. ReLU is widely used in hidden layers due to its efficiency, while Softmax is used in output layers for multi-class classification.

Loss functions including MSE and MAE for regression and Cross-Entropy for classification were studied. Model training was performed using Gradient Descent and its variant, Stochastic Gradient Descent (SGD), with the learning rate controlling update steps.

Regularization techniques such as L1/L2 regularization and Dropout were explored to reduce overfitting. The Adam optimizer was studied for its adaptive learning rate and faster convergence.

K-Means clustering was implemented as an introduction to unsupervised learning, providing insight into pattern discovery in unlabeled data.

---

3. Week 2: Convolutional Neural Networks and Batch Normalization

Week two focused on CNNs, the dominant architecture for computer vision prior to transformers.

CNNs address the inefficiency of fully connected networks for image data through local connectivity, weight sharing, and translation invariance. Convolutional operations apply learnable filters to extract features, while pooling layers reduce spatial dimensions and computational cost.

Key concepts studied included padding, stride, receptive fields, and pooling operations. CNNs learn hierarchical representations, progressing from low-level features such as edges to high-level object representations.

Batch Normalization was studied as a technique to stabilize and accelerate training by normalizing layer activations. It reduces internal covariate shift, enables higher learning rates, and improves overall model robustness.

---

4. Week 3: Recurrent Neural Networks

Week three introduced models designed for sequential data.

RNNs process sequences by maintaining a hidden state that captures information from previous time steps. At each time step, the hidden state is updated based on the current input and the previous state, allowing the model to handle variable-length sequences.

Training RNNs using Backpropagation Through Time (BPTT) introduces challenges such as vanishing and exploding gradients. Techniques like gradient clipping and truncated BPTT were studied to improve training stability.

---

5. Week 4: Modern RNN Architectures

To overcome the limitations of vanilla RNNs, gated architectures were explored.

5.1 Long Short-Term Memory (LSTM)

LSTMs use gating mechanisms—forget, input, and output gates—to control the flow of information through the network. This design allows LSTMs to capture long-term dependencies effectively.

5.2 Gated Recurrent Unit (GRU)

GRUs simplify the LSTM architecture by combining gates and reducing the number of parameters, often achieving comparable performance with faster training.

5.3 Advanced Sequence Models

Additional concepts included Deep RNNs, Bidirectional RNNs, and the Encoder–Decoder architecture. The encoder–decoder framework is particularly important as it forms the conceptual foundation for the Transformer architecture.

Assignments Report

Assignment 1: Python, NumPy, and Pandas Fundamentals

This assignment introduced Vision Transformers (ViT), which apply transformer-based self-attention mechanisms to image classification. Images were divided into fixed-size patches and processed as sequences, similar to tokens in NLP. Key concepts such as patch embeddings, positional encoding, self-attention, and transformer encoder blocks were studied. The assignment highlighted ViT’s ability to capture long-range dependencies more effectively than traditional CNNs.

Assignment 2: Neural Networks and Optimization

Assignment 2 focused on the fundamentals of neural networks, including forward and backpropagation, activation functions, loss functions, and gradient descent. Practical implementation helped develop intuition about parameter updates, learning rates, and common training challenges such as slow convergence and hyperparameter sensitivity.

Assignment 3: Intermediate Deep Learning Concepts

This assignment emphasized structured implementation and debugging of neural networks. Key focus areas included tensor dimensions, parameter updates, and efficient use of NumPy. It strengthened the ability to read, debug, and modify machine learning code, preparing for more complex architectures.

Assignment 4: CNNs and Batch Normalization

Assignment 4 involved implementing a CNN for image classification and improving it using Batch Normalization. Theoretical understanding focused on how batch normalization stabilizes training while maintaining flexibility through learnable scale (γ) and shift (β) parameters. Experimental results showed faster convergence and improved generalization compared to the baseline CNN.

Assignment 5: Character-Level RNN

The final assignment implemented a character-level RNN to generate dinosaur names. Text data was preprocessed into character indices, and the model was trained using Backpropagation Through Time with gradient clipping to prevent exploding gradients. A sampling function was used to generate names during training. Over time, the model learned realistic linguistic patterns, producing dinosaur-like names with common suffixes, demonstrating effective sequence learning.