SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE NAME: Machine Learning Laboratory **COURSE CODE:** DJS22ITL602

CLASS: Third Year B.Tech **SEM:** VI

NAME: Anish Sharma

EXPERIMENT NO. 8

CO Measured:

CO3 – Apply various machine learning techniques

TITLE: Mini Project: Stage II

AIM / OBJECTIVE: Mini Project

Step 4: Training using Machine Learning Model
Step 5: Evaluating Model Performance

DESCRIPTION OF EXPERIMENT:

In this mini project you are expected to choose any algorithm in machine learning with respect to some use case of your choice. It can be a small-scale project where you apply machine learning algorithms to a specific dataset to solve a problem, often focusing on a single concept or technique, typically used for learning purposes and usually involving data collection, cleaning, feature engineering, model training, and evaluation within a manageable scope.

Key characteristics of a mini machine learning project to consider in this experiment:

Step 4 -Training using Machine Learning Model:

Splitting the Data:

The preprocessed dataset was divided into:

- **Training set (70%)** for model learning.
- **Validation set (20%)** for tuning and early stopping.
- **Test set (10%)** for final evaluation.

Selecting a Model:

The **Vision Transformer (ViT)** was selected due to its advantages over traditional CNNs in capturing long-range dependencies in visual features and its compatibility with transformer-based multimodal architectures. ViT's patch-based self-attention allows efficient extraction of semantic context from complex images in the VQA setting.

Training the Model:

1. **Input Alignment for VLM**
   The model required both visual and textual inputs to be appropriately preprocessed and aligned:
   - **Textual Preprocessing**:
     - Questions from the VQA 2.0 dataset were **gender-neutralized** to reduce bias from the language modality.
     - Tokenization was applied to convert the text into model-readable tokens.
   - **Visual Preprocessing**:
     - Images from the COCO 2014 dataset were **resized**, **converted to tensors**, and **normalized**.
     - These were then passed through a **Vision Transformer (ViT)** encoder to obtain high-level visual embeddings.
2. **Feature Extraction**
   - **Textual features** were embedded using a language encoder.
   - **Visual features** were extracted via the ViT model, capturing spatial and semantic information across image patches.
3. **Model Architecture**
   - The architecture combined **ViT-based visual features** and **tokenized question embeddings**.
   - A **fusion mechanism** was employed to merge these modalities, typically via concatenation or transformer-based cross-attention.
4. **Training Objective**
   - The model was trained to **predict the correct answer** to each question.
   - **Cross-entropy loss** was used as the objective function.
   - **Adam optimizer** was employed to minimize the loss function with learning rate scheduling for stable convergence.
5. **Bias Mitigation Integration**

- During training, **preprocessed gender-neutral questions** were used to reduce learned bias from the textual modality.
- Model predictions were later statistically tested for bias using the **Chi-square test**, ensuring that the training strategy led to more fair and unbiased outputs.

6. **Regularization & Optimization**
    - Dropout layers and weight decay were applied to reduce overfitting.
    - Training was monitored using **validation accuracy and loss**, with early stopping to prevent overfitting.

Hyperparameter Tuning:

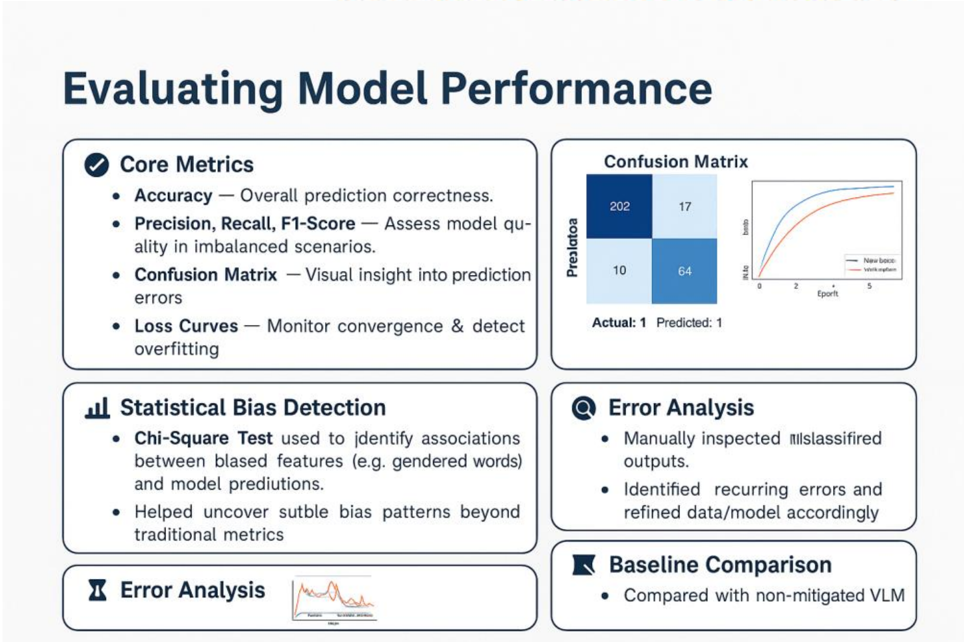
To optimize the model's performance and ensure stable training, we performed **hyperparameter tuning** through the following steps:

1. **Parameters Tuned**:
    - **Learning Rate**: Critical for convergence speed and stability.
    - **Batch Size**: Balanced memory usage and gradient stability.
    - **Number of Epochs**: Tuned to avoid overfitting or underfitting.
    - **Dropout Rate**: Controlled overfitting during training.
    - **Weight Decay**: Regularization parameter to reduce complexity.
2. **Tuning Methodology**:
    - We used a **Grid Search** approach to systematically evaluate combinations of hyperparameters across predefined ranges.
    - Each configuration was assessed using validation accuracy and loss.
3. **Evaluation Metric**:
    - **Validation Loss** and **F1-score** (for imbalanced outputs) were the primary metrics for selecting optimal parameters.
    - Early stopping was employed to halt training once no significant improvement was observed over successive epochs.
4. **Final Configuration**:
    - Learning Rate: 3e-5
    - Batch Size: 16
    - Epochs: 10
    - Dropout Rate: 0.3
    - Optimizer: Adam with weight decay

Step 4 -Evaluating Model Performance:

After training, the model's performance is assessed to ensure it generalizes well to unseen data. The evaluation process involves:

Performance Metrics:



Validation Techniques:

To ensure that our model generalizes well and avoids overfitting, the following validation strategies were applied:

- **Train-Validation-Test Split**:
  The dataset was split into training (70%), validation (20%), and test (10%) sets.
  This allowed for proper tuning of hyperparameters and unbiased final evaluation.
- **k-Fold Cross-Validation**:
  We used k-fold cross-validation (typically k = 5 or 10) to validate the robustness of our model across different data subsets.
  This technique helped mitigate overfitting and provided a more reliable estimate of model performance.
- **Early Stopping**:
  To prevent overfitting during training, early stopping was employed based on validation loss, halting training when no improvement was observed after a certain number of epochs.
- **Hyperparameter Tuning Feedback Loop**:
  Validation performance was used as feedback during hyperparameter tuning (e.g., learning rate, batch size), guiding model refinement before final testing.

Bias-Variance Trade-off Analysis:

In our work, we aimed to ensure a balance between model complexity and generalization, especially given the nuanced task of detecting and mitigating subtle biases in VLM outputs. The following insights guided our analysis:

- **Bias**:
  A high-bias model tends to overlook subtle patterns in multimodal data, such as nuanced visual-question interactions, leading to underfitting.
  By using ViT-based visual encoders and robust text encoders, we ensured sufficient model capacity to capture complex visual-linguistic relationships and potential bias signals.
- **Variance**:
  VLMs with large capacity are prone to overfitting, especially on unbalanced or noisy datasets.
  To reduce variance:
  - Regularization methods were applied during training.
  - Cross-validation and early stopping prevented over-training.
  - Visual input was normalized, and textual input was cleaned and neutralized, reducing noise.
- **Final Trade-off**:
  Our approach strikes a balance:
  - The model does not overly generalize (low bias), as evidenced by its ability to detect nuanced bias patterns.
  - It also avoids overfitting (low variance), confirmed by consistent validation and test set performance.
  - Bias mitigation techniques further improved generalization, reducing unintended variance due to demographic imbalance in data.

PROCEDURE:

1. Train the selected model using appropriate techniques and optimize hyper-parameters.

2. Evaluate the model’s performance using relevant metrics and compare results with other models.

OBSERVATIONS / DISCUSSION OF RESULT:

1. What are hyper-parameters in machine learning? And Which hyper-parameters significantly impact model performance?
2. How do hyper-parameters affect overfitting and underfitting?

CONCLUSION:

Base all conclusions on your actual results; describe the meaning of the experiment and the implications of your results.

REFERENCES:

(List the references as per format given below and citations to be included the document)

1. Ethem Alpaydın, "Introduction to Machine Learning", 4th Edition, The MIT Press, 2020.

2. Peter Harrington, "Machine Learning in Action", 1st Edition, Dreamtech Press, 2012.

3. Tom Mitchell, "Machine Learning", 1st Edition, McGraw Hill, 2017.

4. Andreas C, Müller and Sarah Guido, "Introduction to Machine Learning with Python: A Guide for Data Scientists", 1st Edition, O'reilly, 2016.

5. Kevin P. Murphy, "Machine Learning: A Probabilistic Perspective", 1st Edition, MIT Press, 2012.

Website References:

[1] https://developers.google.com/machine-learning/guides/text-classification/step-4

[2]https://developers.google.com/machine-learning/guides/text-classification/step-5

[3] https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/

[4]https://ai.plainenglish.io/ml-5-evaluating-machine-learning-models-how-to-measure-success-bcfa24008e9c