Advanced DL Project Proposal: AI-Generated Image Detection

Ji Sung Han David Huynh Shahzad Jalil Luke Shen

1 Introduction

The chosen Kaggle competition is Detect AI vs Human Generated Images. The goal is to develop a machine learning model capable of accurately distinguishing AI-generated images from human-created ones. The competition evaluates submissions primarily using the **F1-Score**, The objective of this project is to develop an **ensemble model** for AI-generated image detection. By leveraging various architectures such as CNNs and Transformers, the model considers multiple factors to enhance its predictive capability.

2 Data Description

The dataset consists of labeled images: $y = 1$ (AI-generated images) $y = 0$ (Human-created images) These images vary in resolution and quality, necessitating preprocessing for consistency.

Key Challenges:

- **Class imbalance**: AI-generated images may be underrepresented.
- **Style variability**: Human-created images exhibit diverse textures and structures.
- **Risk of overfitting**: The model might memorize AI-generated patterns rather than generalizing them.

Preprocessing and Enhancements: To standardize the dataset and improve robustness:

- Resize images based on the model architecture: **EfficientNet-B4**: 380×380, **ResNet-50**: 224×224, **Swin Transformer**: 256×256
- Normalize pixel values based on dataset mean and standard deviation.
- Apply **data augmentation** techniques, including:
  - *MixUp, CutMix*: Improve feature diversity.
  - *Color jittering*: Enhance generalization.
  - *Affine transformations (rotation, scaling, translation)*: Introduce positional variance.

3 Methodology and Evaluation Plan

3.1 Machine Learning Techniques

We propose a **Late Fusion ensemble model** combining:

- **EfficientNet-B4**: Captures fine texture details.
- **ResNet-50**: Extracts high-level structural features.
- **Swin Transformer**: Detects global consistency and long-range dependencies.

Each model outputs a probability score, which is aggregated using weighted averaging:

$$f_{\text{final}}(x) = w_1 f_{\text{EfficientNet}}(x) + w_2 f_{\text{ResNet}}(x) + w_3 f_{\text{Swin Transformer}}(x) \tag{1}$$

where w_1, w_2, w_3 are optimized by Bayesian Optimization based on model performance metrics.

3.2 Optimization Strategies

To ensure generalization and stability, we apply:

1. Individual Model Optimization

- **EfficientNet-B4**:
  - Utilizes *transfer learning*, *MixUp*, and *cosine annealing* for training stability.
  - Optimized with **Binary Cross-Entropy (BCE) Loss**.
  - **Hyperparameter tuning:** Learning rate, weight decay, dropout rate, augmentation strength.
- **ResNet-50**:
  - Incorporates *batch normalization*, *gradient clipping*, and *Stochastic Weight Averaging (SWA)* to improve training stability.
  - Uses **Focal Loss** to handle class imbalance by giving more weight to difficult samples.
  - **Hyperparameter tuning:** Learning rate, momentum, batch size, number of frozen layers.
- **Swin Transformer**:
  - Optimized with *AdamW*, *warmup scheduling*, and *attention heatmaps* for interpretability.
  - Trained using **BCE Loss** with label smoothing to reduce overconfidence in predictions.
  - **Hyperparameter tuning:** Learning rate, number of layers, attention dropout, window size.

2. Ensemble Model Optimization

- **Weighted Averaging**: Dynamically optimizes (w_1, w_2, w_3) based on confidence scores.
- **Bayesian Optimization**: Fine-tunes ensemble weights for optimal performance.
- **Stacking Model**: In addition to weighted averaging, we implement a stacking approach where a logistic regression model is trained on the individual model outputs to enhance final predictions.
- **Cross-Validation**: Applies K-fold validation to ensure stability.

3.3 Evaluation Metrics

We will evaluate models using:

- **AUC-ROC**: Measures classification capability; higher values indicate better performance.
- **F1 Score**: Balances Precision and Recall to assess model effectiveness.

The best-performing model will be selected based on these metrics (higher is better).

4 Project Timeline

- **Week 1**: Data exploration and preprocessing.
- **Week 2**: Model architecture implementation.
- **Week 3**: Individual model training and optimization.
- **Week 4**: Ensemble model development and fine-tuning.
- **Week 5**: Model evaluation, hyperparameter adjustments, comparison, and report writing.