National College of Ireland

Configuration Manual: An Optimised Deepfake Detection System Using Vision Transformers And FFT

MSc Research Project
MSc Data Analytics

Nobin Sabu
Student ID: x23313081

School of Computing
National College of Ireland

Supervisor: Eric Gyamfi

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Nobin Sabu
Student ID:	X23313081
Programme:	Msc Data Analytics **Year:** 2024-2025
Module:	Research Practicum
Lecturer:	Eric Gyamfi
Submission Due Date:	11-08-2025
Project Title:	An Optimised Deepfake Detection System Using Vision Transformers And FFT
Word Count:	490 **Page Count:** 5

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Nobin Sabu
Date:	11-08-2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

An Optimised Deepfake Detection System Using Vision Transformers And FFT

Nobin Sabu
Student ID: x23313081

1) System Requirements

Component	Requirement
OS	Windows 10+, macOS 11+, or Ubuntu 20.04+
Python	3.9–3.11
GPU	CUDA GPU (≥12GB VRAM recommended). CPU OK for inference, GPU strongly recommended for training
RAM	16GB min (32GB recommended)
Disk	~120GB for full FF++ videos; ~10–30GB for frames + checkpoints

---

2) Python Dependencies

Core:

pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu121

pip install timm==1.0.9 opencv-python numpy scipy scikit-learn matplotlib tqdm pandas pillow

Optional / Tooling:

pip install jupyter ipywidgets rich tensorboard onnx onnxruntime

3) Model I/O Configuration

Task: Binary classification (Real vs Fake)

Branch	Input	Size	Normalization	Backbone	Output

Branch	Input	Size	Normalization	Backbone	Output
Spatial (RGB)	RGB frame	224×224×3	ImageNet mean/std	Swin-Tiny (swin_tiny_patch4_window7_224)	768-d
Frequency (FFT)	FFT log-magnitude (gray→3ch)	224×224×3	mean=0.5 std=0.5	EfficientNet-B0	1280-d

Fusion & Head: concat → Dropout(0.3) → Linear(2048→1) → BCEWithLogits
Frame sampling: every 10th frame; max 10 frames/video
Output: frame-level logit → sigmoid(prob FAKE)

4) Data Pipeline

- **Dataset:** FaceForensics++ (c23 compression)
- **Sampling:** Every 10th frame, cap 10 per video
- **Transforms:**
    - RGB: resize 224, RandomHorizontalFlip(0.5), ToTensor, Normalize(ImageNet)
    - FFT: grayscale → fft2 → fftshift → log(1+|F|) → min–max [0,1] → replicate to 3ch → Normalize(mean=0.5, std=0.5)

5) Training Configuration

Parameter	Value
Epochs	20
Batch size	32 (adjust for VRAM)
Optimizer	AdamW (lr=1e-4, weight_decay=1e-2)
Loss	BCEWithLogitsLoss
Scheduler	StepLR(step_size=5, gamma=0.1)

Parameter	Value
AMP	Enabled (torch.cuda.amp)
Checkpoint	Save best by validation accuracy
Early stopping	Optional (patience 5)

---

6) Evaluation Metrics

- Accuracy (primary)
- Precision, Recall, F1
- ROC-AUC
- Confusion matrix
- (Optional) per-manipulation breakdown

---

7) Baseline Results

- **Best Val Acc:** ~85.86% (≈epoch 15)
- **Final Train Acc:** ~99.47% (epoch 20)

---

8) Execution Instructions

As the full pipeline is coded in a single Jupyter Notebook (deepfake-detection.ipynb), the actual execution consists of simply going through the cells of the Notebook in order.

A) Open the Notebook:

1. Run file deepfake-detection.ipynb on Kaggle, Jupyter Lab, vscode.
2. Turning on GPU acceleration prior to run.

B) Execution Flow:

1.Setting up and Imports-Compose dependencies; set constants (path, hyperparameters).

2. Dataset Preparation - Read FF++ C23 dataset, sample frames, create FFT images and store it under real/ and fake/.

3.Transforms & Dataloaders - Define RGB/FFT stream preprocessing, do train/validation split.

4.Model Definition = Build Hybrid Swin Transformer + EfficientNet model with fusion head.

5.Training Loop - Train specified epochs with mixed precision and saving the best checkpoint.

6.Test - Load the best checkpoint, calculate the metrics, draw the confusion matrix**.**

7.Inference -Predict on one image.

C) Run:

• Run all cells.

• Change corresponding hyperparameters in the configuration cell on top of the notebook.

• Make sure that paths to datasets are right in relation to your environment

---

9) Example Config (Notebook Variables)

```
seed: 42
data:
  rgb_dir: data/frames_rgb
  fft_dir: data/frames_fft
  img_size: 224
  every_n: 10
  max_per_video: 10
train:
  epochs: 20
  batch_size: 32
  amp: true
  optimizer:
    name: adamw
    lr: 1.0e-4
    weight_decay: 1.0e-2
  scheduler:
    name: steplr
```

```
    step_size: 5
    gamma: 0.1
model:
  spatial_backbone: swin_tiny_patch4_window7_224
  freq_backbone: efficientnet_b0
  dropout: 0.3
  fusion_dim: 2048
paths:
  out_dir: outputs
  ckpt_dir: outputs/checkpoints
logging:
  tensorboard: true
```