CAB420 A2 REPORT

Milly Chambers – n11318546

Ethan Bett – n10752251

Nathan Knell – n11115637

Ton Ngo – n10756205

Executive Summary

AI-generated imagery has seen a rapid increase in popularity in recent years which has blurred the line between authentic and generated content, raising significant ethical and security concerns. Wotj deepfake incidents increasing by 550% between 2019 and 2023 (SECURITY HERO, 2023) and AI driven fraud projected to cost $40 billion by 2027 (Deloitte Centre for Financial Services, 2024), reliable AI content detection methods are crucially needed. This project investigates whether the chosen four deep learning techniques: SVM, VGG-Style CNN, Siamese Triplets, and Vision Transformers can effectively distinguish AI-generated images from human created ones. Performance from these machine learning methods are evaluated upon their accuracy, computational efficiency, and scalability for real-world applications.

Methods Selected

1. SVM with HOG & LDA – used as a traditional baseline using gradient based features and linear classification.
2. VGG-Style CNN – a deep convolutional network with architectural enhancements for artifact detection.
3. Siamese Triplet Network – a metric-learning approach utilising triplet loss for discriminative embeddings.
4. Vision Transformer – a self-attention-based model for global context understanding.

Key Findings

- VGG-Style CNN achieved the highest performance with 99.6% accuracy and a 0.9% error rate, clearly demonstrating excellent ability in distinguishing AI-generated images from their human created counterpart.
- Siamese Triplet Network performed strongly with a 97.82% accuracy, leveraging metric learning for robust embeddings.
- SVM with HOG + LDA was an effective baseline achieving a 78.86% accuracy but was significantly outperformed by deep learning methods.

The results found clearly highlight that deep learning models, especially CNNs. Are highly effective at distinguishing Ai-generated images, whilst traditional methods lag behind in performance. The VGG-Style CNN's near-perfect accuracy suggests it as a promising candidate for real-world applications, though the Siamese Triplet Network's strong performance indicates that metric learning is a viable alternative.

Implications

This study confirms that machine learning can reliably detect AI-generated images, with deep learning models offering incredible accuracy. Future work could potentially explore generalisability across diverse datasets and emerging generative models to maintain detection efficacy in an evolving threat landscape.

Introduction and Motivation

The rapid advancement of artificial intelligence has transformed digital content creation, with sophisticated generative models such as DALL-E and MidJourney enabling people to produce highly realistic AI-generated images. Though these tools offer great creative potential, they also introduce significant ethical and security concerns. The proliferation of AI-generated images raises critical questions about authenticity in the digital age: what happens when generated content becomes indistinguishable from human created images? Between 2019 and 2023 deepfake videos saw an increase by 550% (SECURITY HERO, 2023), and Deloitte predicts that AI-driven fraud losses could reach $40 billion in the United States by 2027 (Deloitte Centre for Financial Services, 2024). Additionally, 59% of consumers report difficulty distinguishing between human made and AI-generated media, highlighting the urgent need for reliable detection methods.

Our project investigates whether machine learning techniques can reliably differentiate between AI-generated images from human-created ones. Our primary objectives with this project are:

1. Evaluate how different machine learning methods perform in classifying image origins
2. Assess the performance of each model to determine optimal detection strategies.
3. Examine computational efficiency and scalability for real-world deployment.

Previous research into detecting AI-generated content has explored:

- Frequency based detective such as a Fourier analysis of GAN artifacts.
- Deep Learning approaches such as CNNs and vision transformers trained on large datasets.
- Hybrid methods combining neural networks with traditional feature extraction.

Our work build upon these foundations, but puts greater focus on a controlled dataset of Shutterstock images and an AI-generated counterpart to minimise bias, as well as a greater focus on deep learning approaches rather than traditional or hybrid.

Related Work

Histogram of Oriented Gradients with Support Vector Machines (HOG + SVM) represents the foundational approach, predating the deep learning methods. This method extracts handcrafted features based on gradient distributions in image regions, which are then classified using Support Vector Machines. While easily interpretable, this approach often struggles with large datasets and the sophisticated artifacts produced by modern generative models, demonstrating the necessity for more advanced feature learning capabilities (Agarwal, Singh, & D, 2021).

MesoNet marked a crucial transition towards deep learning based detection by proposing compact convolutional networks specifically designed for mesoscopic analysis of facial manipulations (Afchar, Nozick, Yamagishi, & Echizen, 2018). MesoNet achieved great results on early datasets such Deepfake and for Face2Face. However, its shallow architecture limits its ability to capture complex features, as performance decreases significantly with image compression or newer generation techniques.

Modern CNN architectures including XceptionNet, EfficientNet, ResNet and VGG style networks have revolutionized deepfake detection through sophisticated feature learning capabilities. XceptionNet utilises depthwise separable convolutions for efficient parameter usage (Chollet, 2016), EfficientNet applies compound scaling for optimal resource utilisation (Tan & Le, 2019), ResNet leverages residual connections to enable deeper connections (Mistry, 2025), and VGG style architectures build hierarchical representations through deep convolutional layers (Grigoryan, 2023). Inception-ResNet-v2 represents an advanced evolution that combines residual connections from ResNet with convolutional layers from XceptionNet, demonstrating how many of these different architectures can be integrated to achieve greater performance for deepfake detection (Kaushik, Doshi, Mal, & Malviya, 2024). Despite their differences, these approaches all share common limitations, such as performance degradation under compression and generalisation challenges across datasets (Ramanaharan, Guruge, & Agbinya, 2025). EfficientNet's larger variants demand substantial computational resources, VGG-style networks require extensive parameters, ResNet needs careful architectural modifications for optimal performance, and Inception-ResNet-v2 introduces a large amount of architectural complexity through its hybrid design.

Vision Transformers (ViT) represent a shift from convolutional architectures towards attention based mechanisms, excelling at capturing global context and long-range dependencies within images to effectively identify telltale signs of artificial images (Dosovitskiy, et al., 2020). Hybrid CNN-Transformer networks attempt to leverage the complementary strengths of both architectural paradigms by combining feature extractors like XceptionNet and EfficientNet with transformer based fusion mechanisms (Khan & Dang-Nguyen, 2022). Both approaches share the limitation of requiring extensive training data and requiring a significant computational overhead when compared to pure CNN methods.

Siamese Networks employ metric learning principles to discriminate between authentic and artificial images by learning similarity metrics rather than direct classification. Different approaches like this feature their own advantages in terms of generalization and adaptability, and their own disadvantage like the careful selection of triplets / pairs that can be very computationally expensive during training (Samrouth, Housseini, & Deforges, 2025).

By systematically comparing these representative approaches on the same dataset using consistent evaluation protocols, we aim assess the capabilities and identify the performance gap between traditional and modern approaches, and gain insight into optimal detection strategies for different deployment scenarios and resource constraints.

Data

The dataset for this machine learning research task originated from a Kaggle Competition: Detect AI vs. Human Generated Images, which focuses on distinguishing AI-generated images from authentic human images. For this project, we utilised only the competition's predefined training set, as it provided a substantial volume of labelled data suitable for supervised learning while also reducing the total sample size to reduce computational cost. The original Kaggle test set was also not used due to it being unlabelled. The pre-defined training dataset consisted of authentic images sourced from Shutterstock's collection of high quality, human-created photographs across various categories, with approximately one third of the included images containing human subjects. These authentic images are systematically paired with AI-generated counterparts produced using the latest generative models.

This Kaggle training data was then further re-partitioned where it was split into 70% training, 15% validation, and 15% test using stratified sampling to maintain class balance in each subset. This specific split was chosen to ensure sufficient training data whilst reserving adequate samples for both validation and testing.

SVM Preprocessing

1. Dataset Subsampling

Initially, the data was split into 70% training, 15% validation, and 15% test. However, to facilitate more experimentation and to manage the computational load by processing many images, while keeping HOG feature extraction in mind, subsampling was applied. Subsampling was applied to the training and validation set where only 40% of the training and validation set were used. The test set was not touched, enabling an unbiased evaluation. Even with the subsampling, memory usage was capped at 90% (32gb total) and when testing with only SVM with HOG, the execution time was over 9 hours (Refer to Table 1).

2. Loading and conversion
    - Images are initially loaded in RGB
    - Each image is resized to a uniform 128x128 pixels
    - RGB images are then converted to grayscale, as HOG features are computed based on local intensity gradients.
3. Data Augmentation

Data augmentation was not used for the SVM with HOG because HOG relies on local intensity gradients to capture features. Common augmentations such as blurring, adjustments to brightness, contrast, hue and saturation could alter or obscure the subtle artifacts and characteristic gradient patterns that can help differentiate AI-generated images from human ones. For instance, AI-generated images can show a certain

sheen or filter to it, and such augmentations could mask these visual cues that HOG might otherwise detect. Therefore, the priority was to analyse features from unaltered image data to preserve any of these artifacts.

VGG-Style CNN Preprocessing

1. Loading and Decoding
    - Images loaded in RGB
    - JPEG decoding via inbuilt TensorFlow function
2. Resizing
    - Images resized to 256x256 using bilinear interpolation
    - Images are centre cropped then resized
3. Normalisation
    - Pixel values converted to float32 values and scaled to 0,1 range
4. Data Augmentation (training split only)
    - Random horizontal and vertical flips
    - Random brightness augmentation (±30%)
    - Random contrast adjustments (0.7-1.3x)
    - Random hue variation (±10%)
    - Random saturation adjustments (0.7-1.3x)
    - Gaussian blur with random σ (0.1-1.5)
5. Batch Processing
    - Image batches of 32
    - Prefetching to optimise GPU utilisation
    - Automatic parallelisation via TensorFlow's data pipeline

Figure 1: Images Before and After Augmentation for VGG-Style CNN

The created preprocessing pipeline aims to balance computational efficiency with feature preservation. Images are resized to a consistent 256x256px for sufficient resolution without excessive overhead, while normalisation stabilises the training. The augmentations such as random flips, brightness/contrast shifts, hue/saturation adjustments and blur are applied only during the training phase to improve generalisation by simulating real-world variations without distorting key AI-generated artifacts. These transformations force the model into focussing on structural inconsistencies rather than superficial noise. Batch processing with prefetching optimises GPU utilisation, helping to ensure efficient training whilst maintaining detection accuracy.

Siamese Triplets Preprocessing

1. Images are loaded in RGB format into train, test and validation datasets. Size is determined by pre-agreed values of 70%/15%/15%
2. Images are resized to 256x256 in batches of 32.
    a. No padding was used to avoid the model training on padding added to images.
    b. TensorFlow Image resize method was left as the default – bilinear.
3. TensorFlow functions are used to apply the following augmentations to the training set only:
    a. random_flip_left_right: Images have a 50% chance to be flipped horizontally (left to right).
    b. random_brightness: Images have their brightness randomly increased or decreased by a maximum of 20% from the starting value X.
    c. random_contrast: Images have their contrast randomly increased or decreased between the bounds specified (lower = 0.8, higher = 1.2).

Figure 2: Images Before and After Augmentation for Siamese Triplets Model

Figure 1 shows images before and after the augmentation was applied, however, the original images have already been resized to 256 x 256. These augmentations were only applied to the training set and were chosen as they mirror the variations you would expect from real images. No padding was deliberately applied to prevent the model from

Vision Transformer Preprocessing

1. Images are loaded in RGB format into train, test and validation datasets. Size is determined by pre-agreed values of 70%/15%/15%
2. Data is then split further into subsets of size 17500/4500/4500 (train/test/validation).
3. Images are resized to 256x256 in batches of 32.
    a. No padding was used to avoid the model training on padding added to images.
    b. TensorFlow Image resize method was left as the default – bilinear.
4. TensorFlow functions are used to apply the following augmentations to the training set only:
    a. random_flip_left_right: Images have a 50% chance to be flipped horizontally (left to right).
    b. random_brightness: Images have their brightness randomly increased or decreased by a maximum of 20% from the starting value X.
    c. random_contrast: Images have their contrast randomly increased or decreased between the bounds specified (lower = 0.8, higher = 1.2).
    d. random_saturation: Images have their saturation randomly increased or decreased between the bounds specified (lower = 0.8, higher = 1.2).

Figure 3: Images Before and After Augmentation for the Vision Transformer Model

Methodology

SVM

A Support Vector Machine (SVM) classifier using Histogram of Oriented Gradients (HOG) features, followed by Linear Discriminant Analysis (LDA) was implemented for the task as a non-deep learning baseline. This contrasts with the other classifiers involving deep learning methods. The HOG+LDA+SVM model serves as a non-deep learning benchmark for evaluating performance and computational cost against deep learning models.

For the HOG+LDA+SVM method, RGB images were converted to grayscale to optimise HOG feature extraction, reduce dimensionality, and reduce computation. Due to computational limitations with the large dataset, 40% of the training (resulting in 22386 training samples) and 40% of the validation (resulting in 4797 validation samples) data were subsampled for this non-deep learning model's experimentation, while the full test set was retained for unbiased evaluation (19988 testing samples). The training samples resulted in over a 2:1 ratio between the number of samples and number of HOG features, ideally improving generalisation and mitigating overfitting. Subsampling allowed for more manageable experimentation where it balanced performance and execution time. An input image resolution of 128x128 pixels was used to manage HOG feature dimensionality, ensure computational feasibility, and optimised with considerations for the chosen HOG parameters.

HOG is a feature descriptor used for classification of object appearance and shape using gradient distributions in portions of an image. An 8x8 pixel cell size balanced detail and computational cost. Cells per block were set to 2x2, which enables local normalisation over the pixel area. Nine orientations were used in each cell's histogram to capture gradient directions. This set of parameters resulted in 8,100 features per image, balancing feature detail, dimensionality, and computational time. Another parameter applied was L2-hys, where it normalises the pixels in the block.

LDA was applied for supervised dimensionality reduction due to the dataset containing labels for classification. LDA addresses the high dimensionality of HOG features. This reduces computational cost and overfitting risk with the training data, improving generalisation, performance, and training times. The number of components from LDA is now set to 1 from the 8100 HOG features, since the dataset only has two classes.

SVM with a linear kernel was used for the classification of LDA + HOG features. The SVM seeks an optimal hyperplane for class separation. A linear kernel was chosen for its computational efficiency and mitigates overfitting with many features, enabling faster times and better performance. The regularisation parameter C was set to 1.0, balancing low training error and good generalisation. Features were also standardised using StandardScaler due to the SVM's sensitivity to feature scaling.

VGG Style CNN

Our VGG-based image classification system utilises a modified VGG-16 architecture, which is specifically optimised to for our task of distinguishing AI generated images from human created ones. Retaining the core structure of sequential 3x3 convolutional blocks, we implemented three key enhancements to address the unique challenges of synthetic artifact detection. Dropout scaling was first implemented, with 0.35 in early layers to 0.6 in classifier layers. This was chosen as through experimentation it was found that stronger regularisation was needed to prevent overfitting to low-level artifacts. This dropout configuration aids the model in maintaining sensitivity to both local anomalies and global compositional features.

In the training process, instead of utilising the standard Adam optimiser we used AdamW which decouples weight decay from gradient updates which was particularly beneficial given our model's depth. Additionally, we implemented a training warmup with the first 5 epochs having a learning rate of 1e-5 to 1e-4. This helped to minimise early training instability while gradient clipping safeguards us from destructive updates during later epochs. These modifications collectively helped to reduce validation accuracy spikes we encountered during our experimentation phase when building the model.

Additional architectural extensions further differentiate our approach from conventional VGG applications. Instead of traditional flattening, we used global average pooling, preserving spatial relationships that are critical for detecting distributional inconsistencies in AI-generated images. Each convolutional block includes batch normalisation and L2-SP regularisation, with focal loss to address class imbalance in our chosen dataset. Our developed model demonstrated a noteworthy sensitivity to high-frequency artifacts through learned spectral filters in early layers, whilst deeper layers showed excellent performance in identifying semantic incongruities characteristic of AI generated images.

Siamese Triplet Network

A Siamese Triplet Network was used to learn discriminative embeddings to distinguish between real and AI-generated images. This approach combines metric learning principles with deep convolutional neural networks to create a feature representation system. This methodology builds upon previous Siamese Networks (Bromley, Guyon, & LeCun, 1993) and the triplet loss formula used by FaceNet (Schroff, Kalenichenko, & Philbin, 2015).

This approach differs from traditional classification networks. Rather than training a model to predict class labels, the model learns to map input images into a high dimensional embedding space where similar images are clustered together.

Architecture Overview and Information Flow:

The Siamese Triplet Network Consists of three parallel networks that share the same parameters. Three input images are processed simultaneously: an anchor image, a positive image (same class as the anchor), and a negative image (different class from anchor). An example of this triplet is featured below in Figure 3.

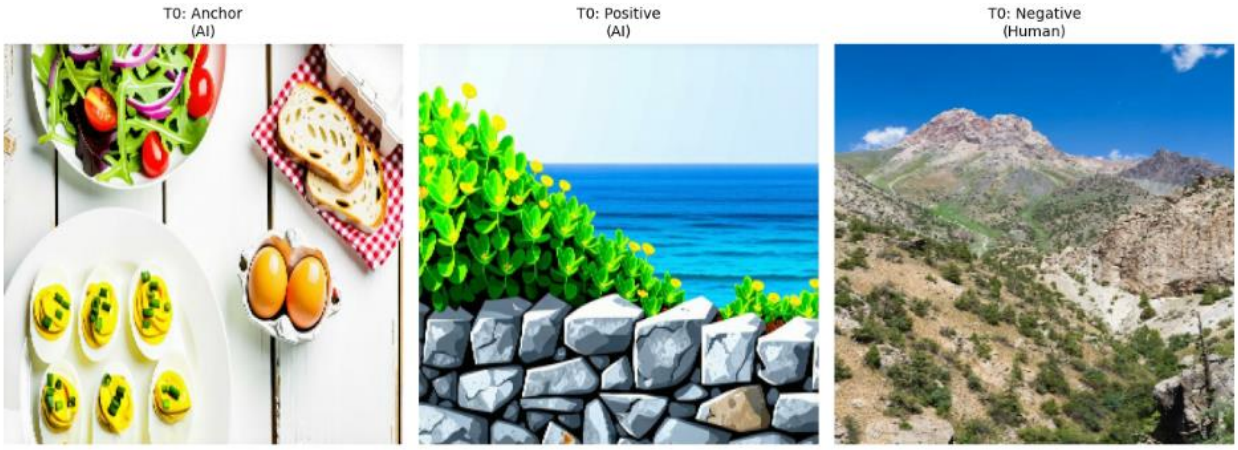

Figure 4: A Triplet of Input Images

Each image passes through the shared 'backbone' DCNN independently, using identical parameters to ensure a consistent embedding space. The backbone outputs a 256-dimensional embedding vector for each input image and Triplet Loss operates on the three images to ensure there is a suitable relative distance between the embeddings. Gradients then flow back through the backbone, updating the parameters.

Triplet-Based vs Pair-Based Learning:

This implementation features triplet-based learning rather than a standard Siamese Twin Network. This traditional approach processes pairs of images using contrastive loss to minimise the distance between positive pairs while maximising the distance between negative pairs through thresholds.

This method suffers from several limitations, the most prominent being that minimising the distance between positive pairs and maximising the difference between negative pairs is done separately. This

leads to the model achieving specific distance values between classes rather than learning meaningful relationships between them.

Triplet loss addresses this by reducing positive distances and increasing negative distances simultaneously, leading to the model focusing on relative relationships rather than fixed distances. Learning from multiple comparisons also gives the model more context, leading to better generalisation. This results in an embedding space with better clustering and class separation.

Backbone Network Architecture:

The backbone network represents the core feature extraction engine, transforming the raw pixels from the image into a meaningful feature space. The model's architecture – which can be seen in Figure 4 – draws inspiration from VGG style networks, with modifications and improvements made for this specific task.

The network consists of five convolutional blocks, each block uses a 3x3 kernel and features batch normalisation, ReLU activation and max pooling. The blocks feature a progressive increase in filter count and decrease in spatial resolution output from 128x128x32 in Block 1 to 8x8x512 in Block 5.

These blocks follow established principles where early layers capture simple details like edges while deeper layers capture more complex combinations relevant for this specific task of distinguishing between real and AI generated images.

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 256, 256, 3)	0
conv2d (Conv2D)	(None, 256, 256, 32)	896
batch_normalization (BatchNormalization)	(None, 256, 256, 32)	128
max_pooling2d (MaxPooling2D)	(None, 128, 128, 32)	0
conv2d_1 (Conv2D)	(None, 128, 128, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 128, 128, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 64, 64, 64)	0
conv2d_2 (Conv2D)	(None, 64, 64, 128)	73,856
batch_normalization_2 (BatchNormalization)	(None, 64, 64, 128)	512
max_pooling2d_2 (MaxPooling2D)	(None, 32, 32, 128)	0
conv2d_3 (Conv2D)	(None, 32, 32, 256)	295,168
batch_normalization_3 (BatchNormalization)	(None, 32, 32, 256)	1,024
max_pooling2d_3 (MaxPooling2D)	(None, 16, 16, 256)	0
conv2d_4 (Conv2D)	(None, 16, 16, 512)	1,180,160
batch_normalization_4 (BatchNormalization)	(None, 16, 16, 512)	2,048
max_pooling2d_4 (MaxPooling2D)	(None, 8, 8, 512)	0
global_average_pooling2d (GlobalAveragePooling2D)	(None, 512)	0
dense (Dense)	(None, 1024)	525,312
dropout (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524,800
embeddings (Dense)	(None, 256)	131,328

Figure 5: Siamese Triplet Network Architecture

The transition from spatial feature maps to fixed-size embeddings is done through global average pooling, which computes the mean value across each 8x8 spatial location for each of the 512 features. The resulting 512-dimension vector is passed through two fully connected layers for the final transformation. The first dense layer features 1024 neurons with ReLU activation and 20% Dropout, while the second dense layer features 512 neurons with ReLU activation. The final embedding layer creates the 256-dimensional output embedding without any activations.

Classification Pipeline:

The trained backbone is used to create reference embeddings for the entire training dataset. These reference embeddings are stored alongside their class label to create an embedding map. To classify new images, the input image is pre-processed to match the backbone input. These images are processed using the backbone to create an L2 normalised embedding vector to match the reference embedding space. Using cosine distance as the similarity metric, the k nearest neighbours (k-NN) are identified and the resulting prediction is based on the majority class.

Triplet Loss Formula:

Several modifications have been made to the standard triplet loss formula used by FaceNet to increase training stability and convergence (Schroff, Kalenichenko, & Philbin, 2015). Cosine distance is used instead of Euclidean distance, providing scale invariance and increasing the stability of high dimensional data (Alvin, 2023). Softplus activation is used instead of **max(0,)** to create smoother gradients throughout training. L2 regularisation is also utilised to prevent feature collapse.

Training Strategy and Hyperparameter Configuration:

The hyperparameter configuration was tuned through experimentation to balance computational efficiency and model performance. The embedding dimension was set to 256, which provides sufficient representational capacity. A batch size of 32 triplets (96 individual images) was chosen as a good balance between computational efficiency and gradient stability. A smaller batch sizes of 16 resulted in very unstable training and validation losses while a higher batch size of 64 caused memory issues due as it is loading in and processing 192 images at a time.

The Adam optimizer was used with a conservative learning rate of 0.00003, allowing us to train for longer without overfittings, and a gradient clipping of 0.5 to prevent gradient explosion (Goldberg & Hirst, 2017).

Only 40 batches were processed per epoch rather than the entire dataset. Processing the entire dataset would require 1750 batches per epoch, leading to a large computational overhead and risk of overfitting. This approach forces the model to learn more robust features, rather than overfitting specific images. As there are approximately 56 000 images in the dataset, using each image as an anchor in the triplet would result in approximately 168 000 images being processed every epoch.

Vision Transformer

A Vision Transformer (ViT) model was implemented as an alternative advanced deep learning architecture, which relies on self-attention mechanisms rather than traditional convolutional layers (Dosovitskiy et al., 2021). ViTs split images into patches, which are then flattened and projected into lower-dimensional space, before a Transformer is applied to learn patterns and relationships. Their reliance on self-attention mechanisms enables them to capture global context and long-range dependencies within images, something a traditional CNN does not feature (Acharya, 2023).

While CNNs have traditionally been used for computer vision classification tasks, ViTs represent a relatively new and novel method of handling various computer vision tasks. Our ViT model was optimised for the binary image classification task being investigated here by implementing a sigmoid activation function and binary cross entropy loss. This was possible as we are only classifying images into two categories (AI or Human).

ViT models feature the ability to transfer learning; they can utilise the weights of a model pre-trained on larger datasets, such as ImageNet, which can improve performance particularly where only a small dataset is available for training. Our ViT model was trained from scratch on a subset of 26.5K images in a 70/15/15 split to more fairly compare its performance with the other models implemented.

Due to computational limitations, several concessions were made with the model. Systematic experimentation was performed with the hyperparameters, resulting in a scaled-down ViT to suit our limitations and requirements.

While our image size of 256x256 preserved more fine detail to improve accuracy, it increased VRAM consumption and training time significantly. A small patch size of 8x8 results in a larger number of patches and thus a higher number of patch tokens, improving the performance of our Transformer and self-attention mechanisms but also increasing computational requirements.

Typical ViTs will use 12+ layers and attention heads, which allow them to detect different types of complex patterns more effectively. Our ViT was scaled down to 8 attention heads, which should still result in each head seeing a 64-dimension subspace, matching the per-head size used in the base implementation of ViT, while reducing compute requirements. 8 transformer layers were used as this provided the best balance of computational requirements, training time, and risk of overfitting.

The Adam optimiser was used with a conservative learning rate of 0.0001. Adam is an optimiser commonly used with ViT models, and the conservative learning rate promotes stability and convergence while training.

Evaluation and Discussion

SVM

Method	Training Accuracy	Test Accuracy	Macro F1-Score	Total Time (seconds)	AUC
SVM with HOG	0.9998	0.7675	0.76	33707	0.84
SVM with HOG and LDA	0.9364	0.7886	0.79	471	0.87

Table 1: Performance Metrics and Execution time for SVM Models with HOG and HOG & LDA

The HOG + LDA + SVM model was evaluated as a non-deep learning baseline model and performed relatively well in classifying AI and human images. An initial implementation using only HOG features with SVM was also explored, where no LDA was applied. However, this approach showed significant overfitting, with a training accuracy of 0.9998 compared to a test accuracy of only 0.7675, and was computationally intensive, requiring 33707 seconds for execution (Refer to Table 1). For this reason, the HOG+LDA+SVM method was explored and is the subject of baseline analysis. We examined performance metrics including accuracy, Macro F1-score, AUC, computational efficiency, and plotted a confusion matrix to further analyse classification performance.

The HOG+LDA+SVM model achieved a training accuracy of 0.9364, a test accuracy of 0.7886, and a Macro F1-score of 0.79 (Refer to Table 1). This is an improvement in test performance and a reduction in overfitting compared to the HOG+SVM model without LDA. The model achieved a good discriminative ability with an AUC score of 0.87 (Refer to Table 1). The inclusion of LDA also drastically improved computational efficiency, reducing the total execution time to 471 seconds, around 72 times faster (Refer to Table 1). However, the drastic difference in execution time could have been due to computational resource limits, where memory was hitting 90% usage (32gb total).

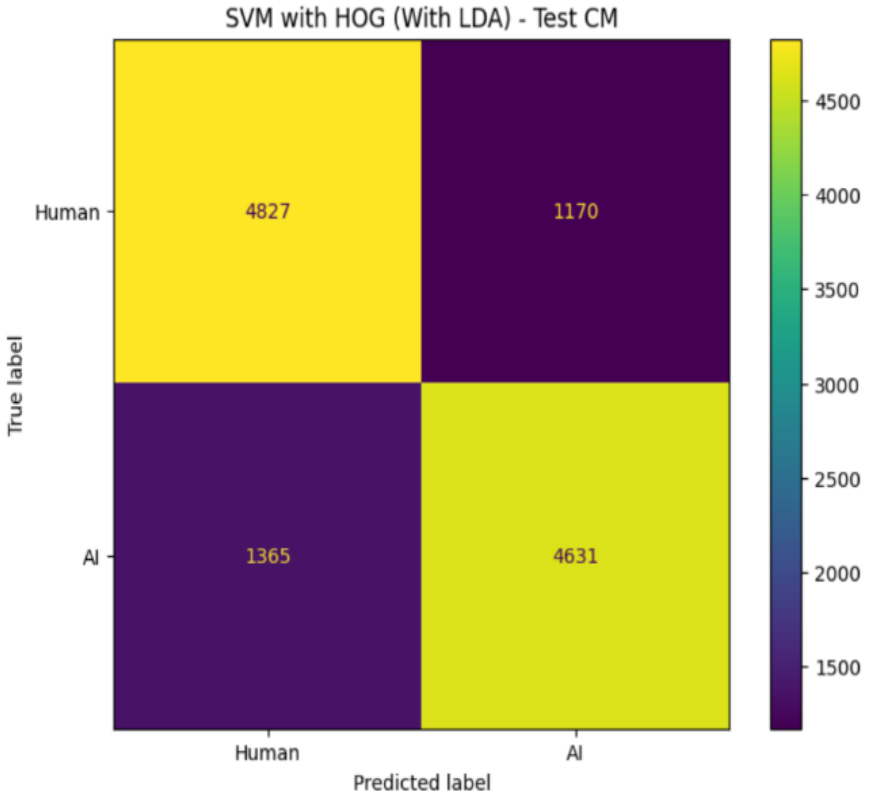

Figure 1: Confusion Matrix for the SVM with HOG and LDA on the test set

Test Classification Report:

	precision	recall	f1-score	support
Human	0.78	0.80	0.79	5997
AI	0.80	0.77	0.79	5996
accuracy			0.79	11993
macro avg	0.79	0.79	0.79	11993
weighted avg	0.79	0.79	0.79	11993

Figure 2: Classification Report for the SVM with HOG and LDA on the test set

The confusion matrix, in Figure 1, for the model on the test set, showed that 4,827 human instances and 4,631 AI instances were correctly classified. However, the model misclassified 1,170 human instances as AI and 1,365 AI instances as human. This indicated that the model struggled slightly more with classifying AI images, with a recall of 77% for the AI class compared to 80% for the human class (Refer to Figure 2). The precision for human images was 77%, while for AI images it was 79% (Refer to Figure 2). The overall error rate for the HOG+LDA+SVM model on the test set resulted in 21.14% (1-0.7886), meaning that around one out of five of the test samples were misclassified.

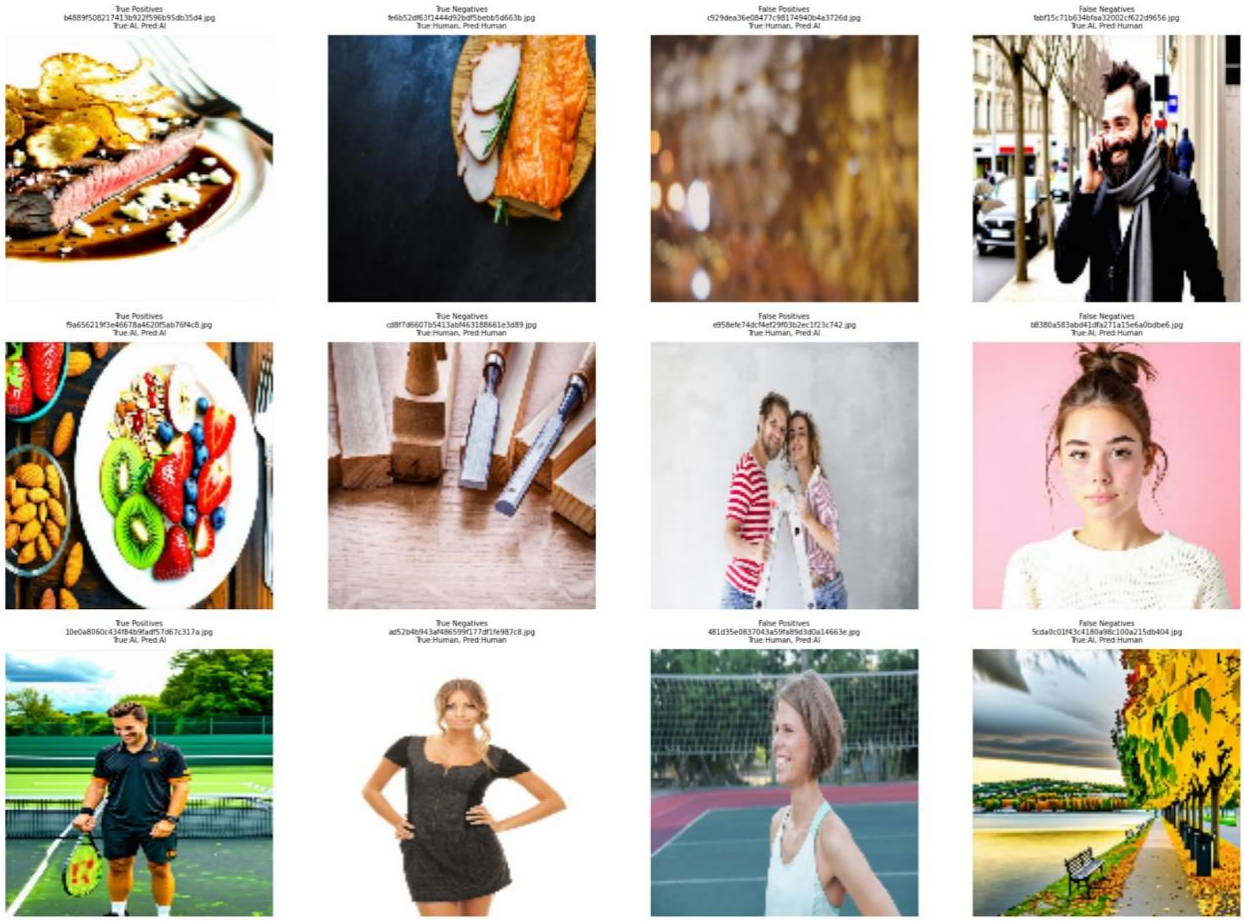

Figure 3: Examples of True Positive, True Negative, False Positive, and False Negative classifications by the SVM with HOG and LDA model on the test set

Figure 3 displays examples of the classification results, potentially offering valuable insights. Notably, the model seems to struggle with photorealistic AI-generated images that include people, resulting in False Negatives. In contrast, it demonstrates proficiency in identifying food. Furthermore, instances where human images are incorrectly classified as AI appear to be portraits, potentially impacted by any subtle textural differences. However, keep in mind that Figure 3 only presents 12 examples out of the total 19988 test samples.

VGG Style CNN

Our developed VGG architecture achieved incredible results with the given task, clearly demonstrating an outstanding ability to detect AI generated images. With an achieved 99.6% accuracy and an incredibly low error rate of 0.9%, the model approaches flawless classification performance with the chosen dataset. This performance is particularly noteworthy given the subtle and evolving nature of artifacts in modern AI-generated images, which confirms the effectiveness of the developed architectural modifications for this classification task.

Looking further into the precision-recall balance reveals the model's robust reliability. The 99.35% precision indicates when the model identifies an image as AI-generated, meaning that the model correctly classifies 993.5 out of 1000 images. This high precision is crucial for real-world applications where accusations of content being AI-generated could have serious consequences, such as in academia. To complement the high precision, the achieved recall of 98.77% is indicative that the model successfully detects nearly all given AI-generated images in the dataset, missing only 1.23%. The resulting F1 score of 0.9996 shows an optimal equilibrium between the two metrics, which showcases the model's ability to maintain both high sensitivity and specificity simultaneously.

The model's strong performance stems from several key design choices. The progressive dropout strategy helped to prevent overfitting to low-level artifacts whilst maintaining sensitivity to higher level features, which is evident through the minimal gap between training and validation performance. The AdamW optimiser with a learning rate warm up provided a stable convergence, with significantly fewer training instability moments compared to standard implementations. Architectural modifications such as global average pooling and focal loss handling of class imbalance significantly helped to boost model robustness across diverse image types. The results suggest our VGG implementation offers a particularly effective solution for applications that require both high accuracy and reliable performance across varying image quality levels.

Performance Metric	**Value**
Accuracy	0.996
Precision	0.9935
Recall	0.9877
F1-Score	0.9996
Percent Misclassification Error	0.9%

Table 2: VGG-Style CNN Performance Metrics

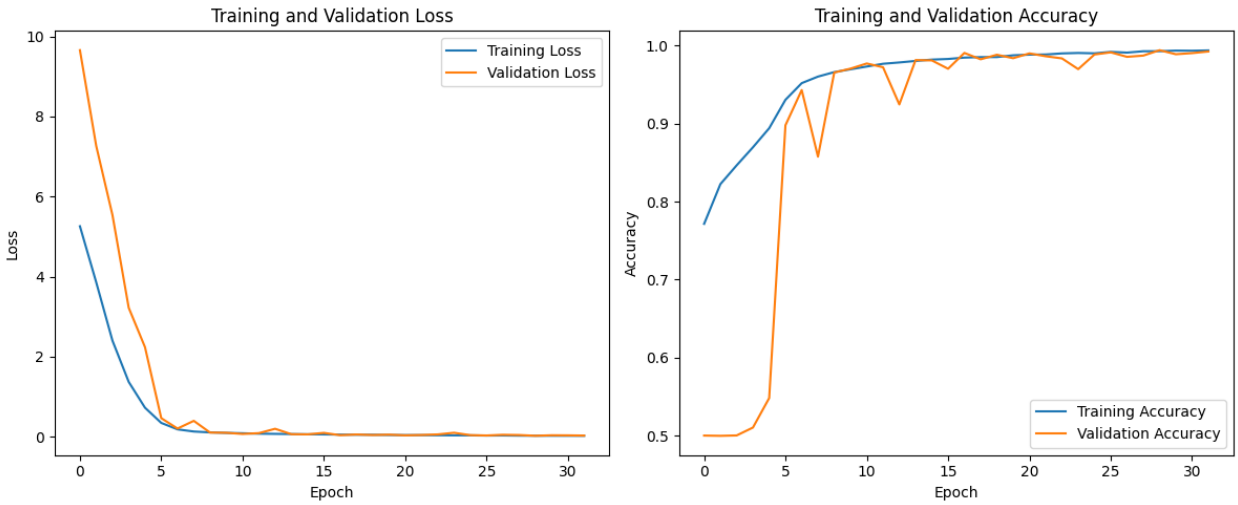

Figure 4: VGG-Style CNN Training and Validation Loss + Accuracy over 32 Epochs

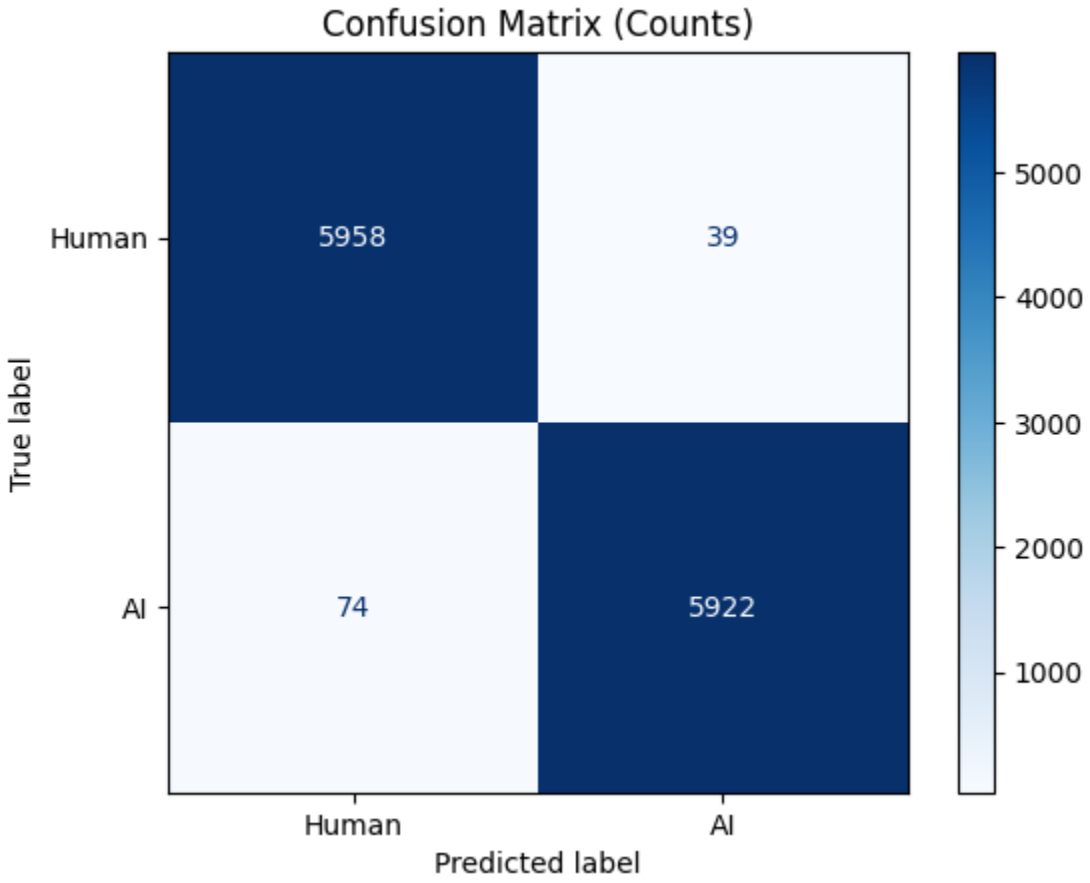

Figure 5: VGG-Style VGG Confusion Matrix

Siamese Triplet Network

Training Performance:

The training and validation loss curves in Figure 5 show a clear pattern of convergence, with both losses decreasing from initial values of approximately 0.740 and 0.85 respectively to final values of 0.2661 0.2438. As indicated by the green dot, the best validation loss of 0.2287 was achieved at epoch 173, and when model training finished at epoch 198 due to early stopping with a patience of 40, the model weights were restored to those from epoch 173. The training and validation curves track closely throughout the 198 epochs, which indicates that the model was able to learn features without significant overfitting. The conservative learning rate of 0.00003 helped maintain the stable training behavior.

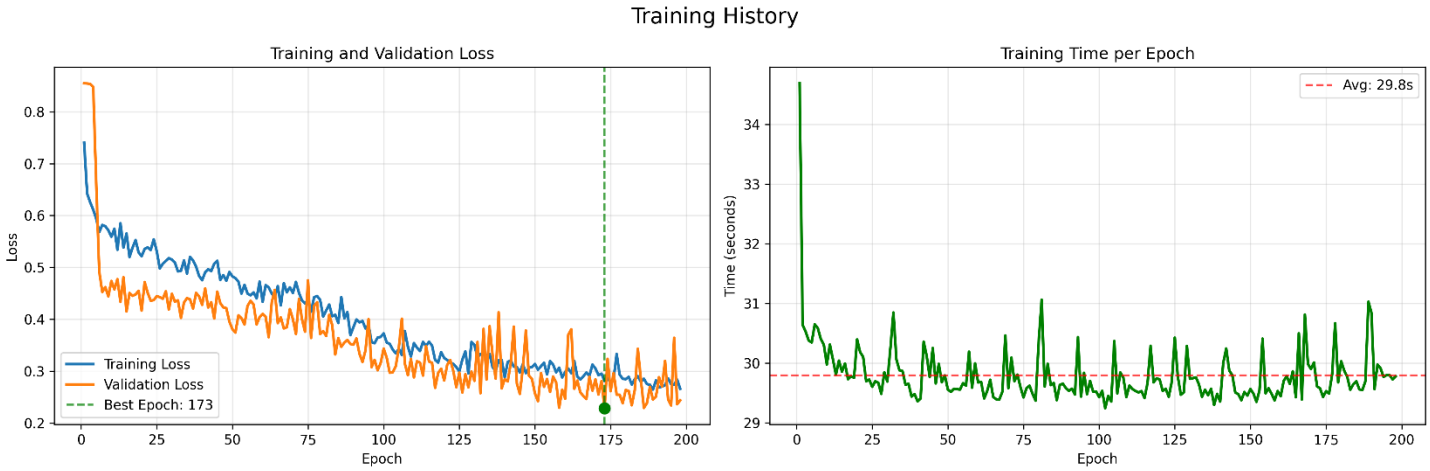

Figure 6: Training and Validation Loss per Epoch and Training Time per Epoch for the Siamese Triplet Network

The training time analysis shown on the right panel of Figure 5 highlights the average training time of 29.8 seconds per epoch, fluctuating between 29 and 31 seconds for all bar the first epoch. The low training time per epoch stems from the decision to only process 40 batches per epoch rather than the full dataset. The total training time for 198 epochs was 98.3 minutes when using an RTX 3080Ti with 12GB of VRAM.

Classification Performance:

A grid search was used to find the best value of K for k-NN classification. Figure 6 shows the k-NN Accuracy across k values from 1 to 9.

K Value	Accuracy
1	0.9765
3	0.9780
5	0.9782
7	0.9779
9	0.9781

Figure 7: k-NN Accuracy for different values of K

The results show a consistently high Accuracy for all values ok k, ranging from 97.65% for k = 1 to 97.82% for k = 5. As such k = 5 was chosen as the optimal number of neighbours for making predictions.

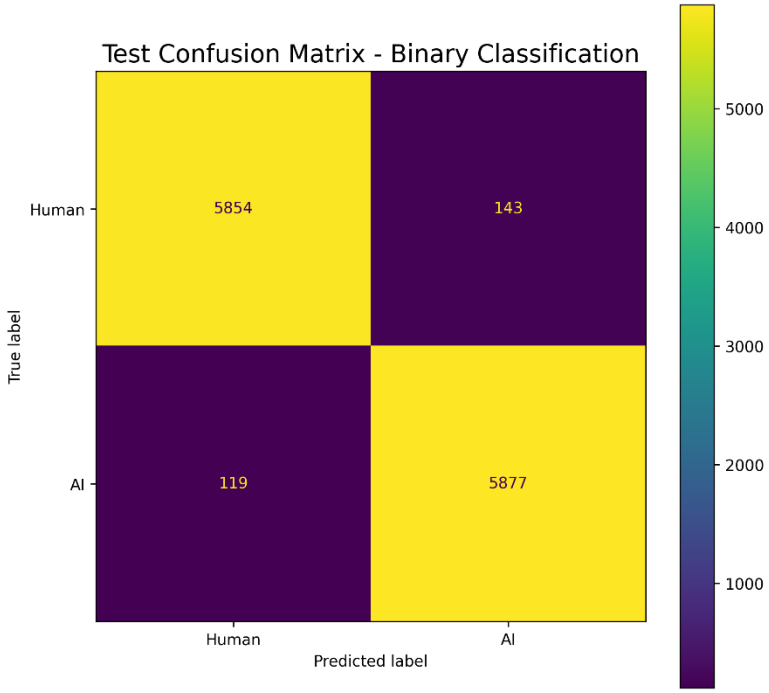

Figure 8: Confusion Matrix for Siamese Triplet Network

The confusion matrix presented in Figure 7 highlights the model's great performance across the test set. The matrix shows 5877 true positives and 5854 true negatives. This results in a misclassification rate of 2.38% for human and 1.98% for AI images, which demonstrates well balanced performance across both classes.

The Classification Report shown in Figure 14 further validate the models performance. The network achieved a precision of 98.01% for 97.82% for AI images. The recall values of 97.62% and 98.02% highlight that the model can detect most of the AI images in the test set. Recall is particularly important in this case as we want to catch as many AI images as possible.

The model achieves consistent results across all metrics for both macro and weighted averages, with a value of 97.82% for precision, recall and F1 Score. This uniformity indicates that the model preforms equally well on both classes. This makes sense given the perfectly balanced testing set (5997 Human vs 5996 AI images). The overall accuracy of 97.82% across the 11993 test images shows the models incredible performance distinguishing between the real and AI generated images in this dataset.

Class	Precision	Recall	F1-Score	Support
Human	0.9801	0.9762	0.9781	5997
AI	0.9762	0.9802	0.9782	5996
Accuracy			0.9782	11993
Macro Average	0.9782	0.9782	0.9782	11993
Weighted Average	0.9782	0.9782	0.9782	11993

Figure 8: Classification Metrics for Siamese Triplet Network

The t-SNE embedding visualisation presented in Figure 15 provides compelling evidence of the model's ability to learn discriminative representations. The plot clearly shows two distinct clusters with the real images represented in purple on the left, and AI-generated images on the right. The clear separation between these clusters demonstrates that the 256-dimensional embeddings capture enough detail to differentiate between the two classes.

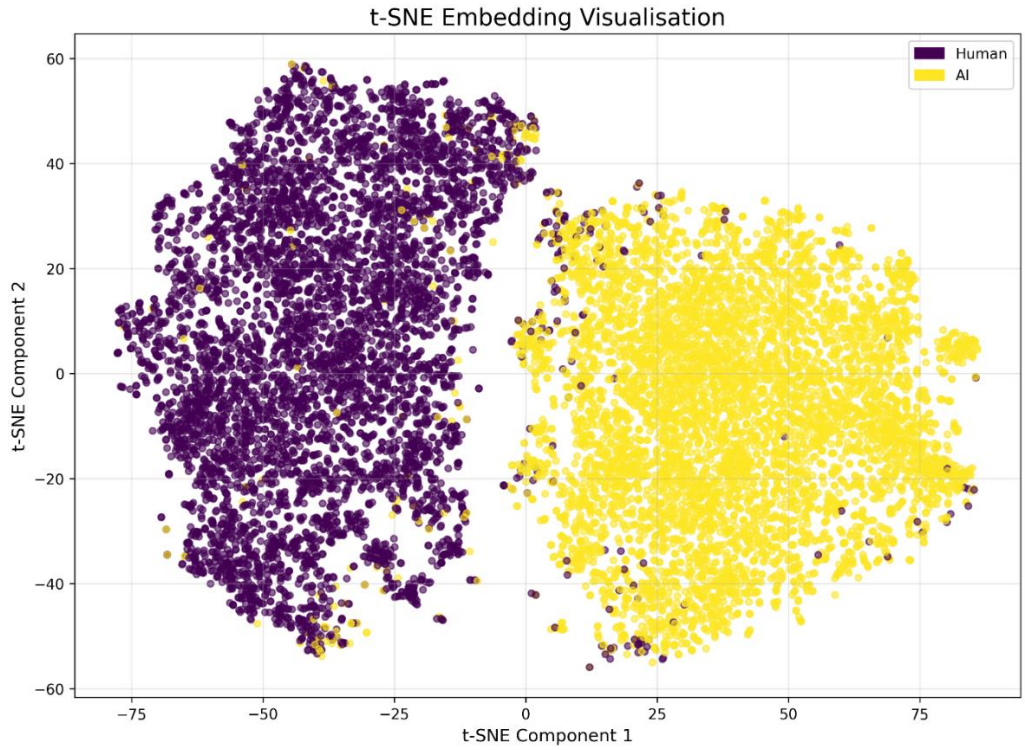

Figure 9: t-SNE Visualisation of Test Set Embeddings

The tight grouping of each cluster and the clear separation boundary between the two indicates that the triplet loss function was able to effectively pull same class images together while pushing different class images apart in the embedding space.

Grad-CAM Analysis:

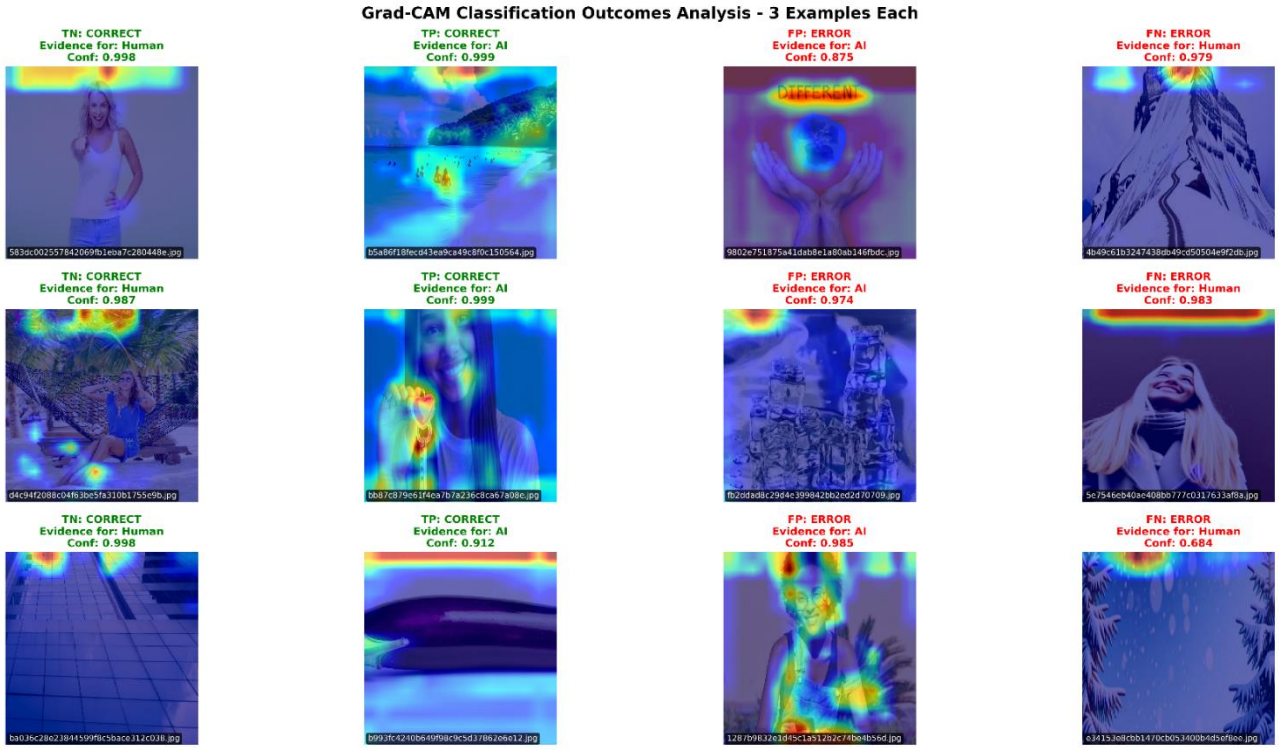

Figure 9: Grad-CAM Heatmaps for Different Classification Outcomes of the Siamese Triplet Network

Grad-CAM (Gradient-weighted Class Activation Mapping) is a technique used to visualise what regions of the image are most important when the model makes a decision (Selvaraju, et al., 2016). Figure 10 shows heatmaps that highlight the areas of an image that contribute to the decision made by the model.

The model's decision making often relies heavily on background elements rather than the primary subject. This pattern can be observed across all three TN examples, with the second example focusing on the surrounding environment including the trees behind her and the sand below. The same background prioritisation appears in the third TP and second FN example where the activation is concentrated across the top portions of the image. These examples suggest that the model has likely learnt that rendering artefacts, lighting inconsistencies and subtle environmental details provide reliable indicators for distinguishing between authentic and AI generated images.

There are also several examples that show the model has also learnt several specific features. The first TP example focuses on the people in the water while the second TP focuses on the love heart in her hand, indicating that the model can identify uncanny characteristics when they appear overly artificial. The third FP example's heavy focus on the subject reveals it has also learnt certain natural human features, though this led to a misclassification. This is likely a result of her authentic features being misinterpreted as artificial potentially due to the lighting or image postprocessing.

Another interesting example is where the text in the first FP tricks the AI into believing it is fake. This indicates that the model has learnt to associate text and other graphic design elements with AI generation.

While the model demonstrates exceptional performance on this dataset, the specific patterns revealed through the Grad-CAM heatmaps highlight potential limitations when encountering images generated by different AI models or generation techniques. The model's focus on particular rendering artefacts, lighting inconsistencies, and environmental details likely reflect the specific weaknesses and characteristics of the AI generation method used to create the fake images in the dataset.

The real images in the dataset were provided by Shutterstock and all share Shutterstock's recognisable style of images, providing a clear contrast between the generation patterns of the AI used.

This will likely prove problematic when used against different generation models as they will most likely exhibit different artifact patterns and rendering styles.

Vision Transformer

The Vision Transformer model was evaluated as a relatively new and novel approach to computer vision tasks like image classification, with high expectations of its performance. While the ViT model did not outperform the Siamese Triplet network nor the VGG-Style CNN model, the results were nonetheless impressive:

Epochs/Training Time	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Errors
25 epochs/189.75 mins	91.76%	93.49%	89.80%	91.61%	0.9732	8.24%

Table 1: ViT Performance Metrics

The ViT model was 91.76% accurate when classifying images as AI or human-generated, and a precision of 93.49%. This means that the model correctly labels about 918 images out of 1000 and correctly predicts an image as AI-generated in 93.49% of cases. These results coupled with the high F1-score (91.61%) indicate the model performs well with the classification task and dataset being used. It is worth considering that with a recall score of 89.8%, the model does incorrectly label AI-generated images as human in 10.20% of all cases. This is important to note as failing to detect a "deepfake" may have serious consequences from both an ethical and legal perspective.

A relatively low error rate of 8.24% was recorded – 371 of 4500 images were misclassified. A slight preference was noted where AI images were misclassified as human (see fig 11).

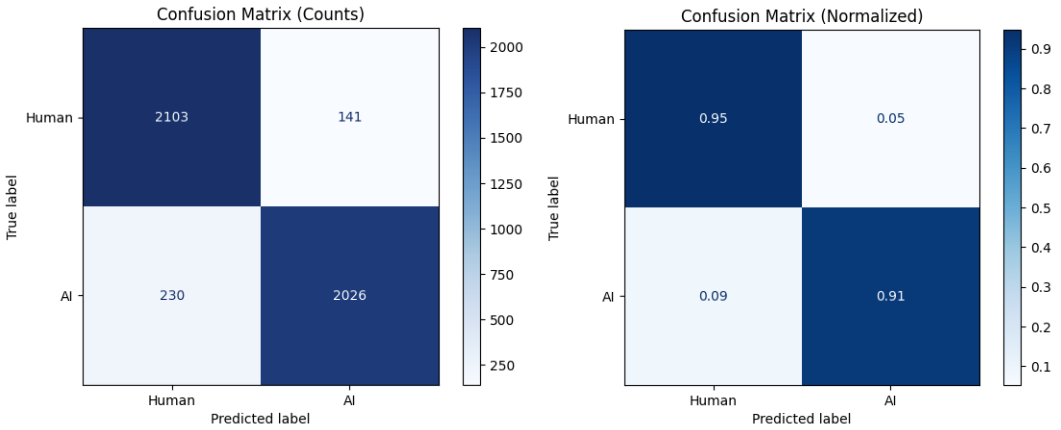

Figure 10: ViT Confusion Matrix

During training, a clear pattern of convergence was noted, with some jitter present (most notably towards the final epoch – see figure 13). Table 2 provides additional evidence of convergence; all accuracy scores are centred around 0.9163 with a variance of 0.0248 between the highest and lowest results.

It should be noted that earlier versions of the model consistently cut training short with a patience level of 5, often terminating before 10-15 epochs. In this iteration of the ViT model, the loss and accuracy curves track each other reasonably well throughout the full 25 epochs, though further training should be possible and will likely improve the model's performance and convergence.

Train Acc	Train Loss	Val Acc	Val Loss	Test Acc	Test Loss
0.9266	0.1849	0.9018	0.2422	-	-
0.9193	0.2011	-	-	0.9176	0.2056

Table 2: ViT Epoch Results

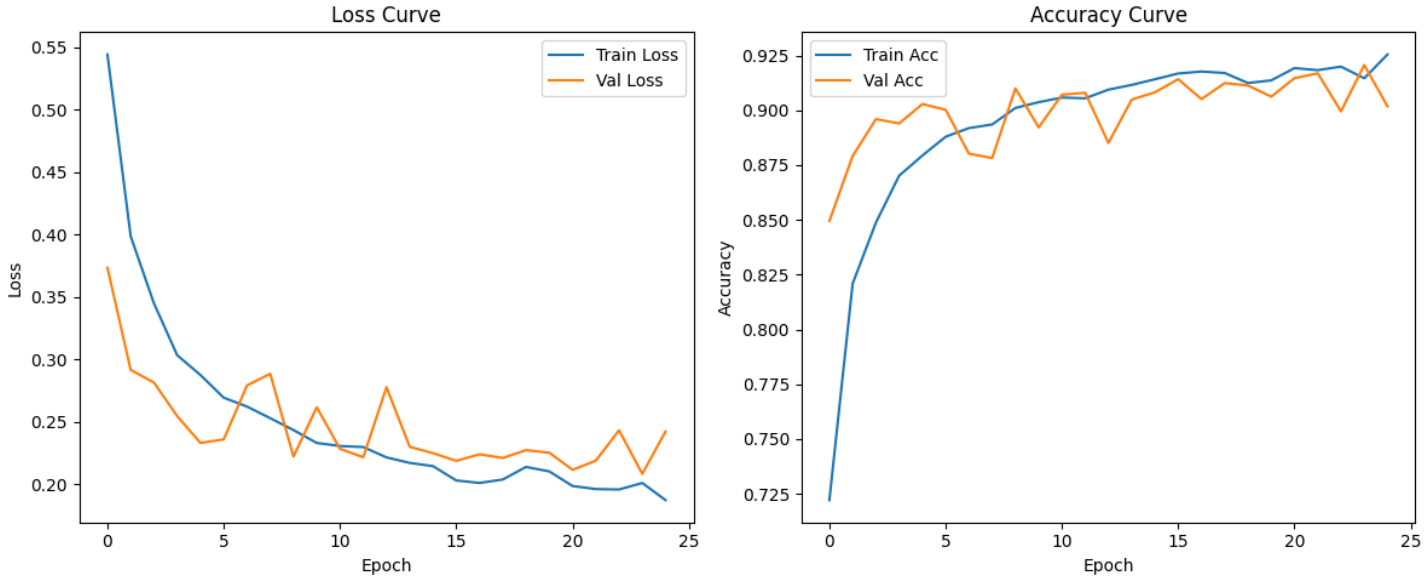

Figure 11: ViT Training History

Due to a hardware failure while building the model, an A100 GPU was used via Google Colab to train the models. Despite the high processing power and VRAM allocation, the model proved to be quite computationally taxing, which is why the model was limited to 25 epochs of training. A total training time of 189.75mins was noted for this iteration, with each epoch taking ~6.58mins, other than the first which took 26.8mins to run. Testing and evaluation took less than one minute to complete.

While this ViT model was trained from scratch like the other deep-learning models implemented, ViTs do feature transfer learning, which allows them to leverage pre-trained models on large datasets like ImageNet, which is then tuned for a specific use on a smaller dataset, thus improving performance and accelerating training. As the dataset used for this project came from a Kaggle challenge set, we are fortunate to have an example of such a model pre-trained using "google/vit-base-patch16-224". The *2025WAI ViT Image Classification* model (Li, 2025) was implemented with smaller images to match the pre-trained model, a variable learning rate that updated each epoch. This resulted in a Kaggle (F1 -score) of 0.99849. Future iterations of our ViT would be well-suited to implement features such as those demonstrated in Li's model.

Model Comparison

Method	Accuracy	Precision	Recall	F1-Score	Number of Epochs	Training Time (Minutes)	Error Rate
VGG-Style CNN	99.6%	99.35%	98.77%	99.96%	32	48	0.9%
Siamese Triplet Network	97.82%	97.82%	97.82%	97.82%	198	98.3	2.18%
Vision Transformer	91.76%	93.49%	89.80%	91.61%	25	189.75	8.2%
SVM with HOG+LDA	78.86%	78.89%	78.86%	78.86%	-	7.85	21.14%

The VGG style CNN achieves near-perfect performance on the training dataset as it is the most suited for this specific task. CNNs excel at detecting local patterns and artifacts through hierarchical feature learning, which is exactly what is needed to identify AI generation artifacts like unnatural textures, inconsistent lighting and subtle rendering flaws.

The Siamese Triplet Network also performs strongly, however it faces a fundamental limitation compared to the other models; it is designed for metric learning rather than direct classification. While t-SNE visualization in Figure 9 above shows excellent separation between human and AI images, it requires an additional k-NN step that introduces potential errors. While the metric learning approach is powerful for generalisation, it adds complexity that isn't necessary for binary classification.

Despite its sophistication and leveraging of self-attention via the Transformer architecture, the Vision Transformer underperforms when compared to the two deep-learning models. ViTs excel at capturing global context and long-range dependencies within images, but AI detection relies more on local artifact detection. Another limitation is that its scaled down architecture – which was necessary due to computational constraints – lacks the capacity to fully leverage ViT's strengths. Even with the scaled-down architecture the model still took nearly twice as long as the next most complex model to train. It is also important to note the less-than stellar recall and error rates; given the potential legal and ethical ramifications of incorrectly identifying an AI image as human-generated (and vice versa), these rates indicate that the models may not be sufficient without further adaptation (or outright replacement).

SVM with HOG+LDA represents the performance ceiling of handcrafted features. HOG features capture gradient information well, but they are fundamentally limited to detecting patterns the designers anticipated. Modern AI generators create artifacts that are too subtle and varied for traditional feature descriptors to reliably identify.

Conclusion and Future Works

The experiments conducted during this project successfully addressed our key objectives. We were able to successfully:

- Evaluate how different machine learning methods perform in classifying image origins by training four unique models on the same dataset.
- Assess each model to determine optimal detection strategies by evaluating the performance metrics for model and comparing them.
- Examine computational efficiency and scalability for real-world deployment by evaluating resource requirements for each model, as well as time taken for training and evaluation.

The VGG-Style CNN exhibited the best metrics on all fronts; It achieved an accuracy, precision, and F1-score of over 99% and took a mere 48 minutes to train. The Siamese Triplet Network scored similar results, but with an error rate and training time roughly double that of the VGG-Style CNN. Finally, and surprisingly, the Vision Transformer underperformed significantly compared to the other deep-learning models, while taking nearly four times as long as the VGG-Style CNN to train, and exhibiting an error rate of 8.2%. The SVM model provided an ideal baseline for comparison as a non-deep learning method. While it was quick to train, it was inaccurate and had an error rate of ~21%.

We identified (based on the above results) that the VGG-Style CNN is the most ideal choice for real-world deployment, due to its outstanding performance and reasonable training time. Most importantly, metrics such as Recall and Error rate (when combined with the fantastic Accuracy and Precision values) indicate that this model is most ideal for classifying AI vs human-generated imagery. The high degree of confidence with which we can see the model classifying AI-generated imagery correctly is paramount when considering the ethical and legal ramifications of ai-generated imagery in modern society.

Future iterations of these models could employ the following changes to improve performance:

- ViT: The ViT model could be pre-trained via transfer learning on a larger dataset such as ImageNet. A hybrid model utilising transfer learning from a CNN model would also present an interesting challenge for future research and evaluation.
- Grad-CAM visualisations could be implemented for all models to provide a more thorough analysis of how the model is working internally.
- Siamese Triplet Network: The triplets may be optimised by selecting better positive and negative samples and anchors. This will improve performance and lower the error rate. This would increase complexity, but it would be worth investigating whether the tradeoff in computational requirements is worth the improvements.

Appendix

Milly Chambers - 25%

- Primary developer of VGG-Style CNN architecture (implementation, tuning, and evaluation)
- Author of all sections of the report related to the VGG-Style CNN
- Executive Summary and Intro sections

Ethan Bett - 25%

- Primary developer of Siamese Triplets architecture (implementation, tuning, and evaluation)
- Author of all sections of the report related to the Siamese Triplets
- Primary author of Model Comparison
- Primary author of Related Works

Nathan Knell - 25%

- Primary developer of Vision Transformer architecture (implementation, tuning, and evaluation)
- Author of all sections of the report related to Vision Transformers.
- Secondary author of Model Comparison
- Primary author of Conclusion

Ton Ngo – 25%

- Primary developer of SVM method (implementation, tuning, and evaluation)
- Author of all sections of the report related to SVM
- Primary author of Data excluding the preprocessing of each model except SVM
- Compiled presentation

Bibliography

Afchar, D., Nozick, V., Yamagishi, J., & Echizen, I. (2018). *MesoNet: a Compact Facial Video Forgery Detection Network*. Retrieved from ArXiv: http://arxiv.org/abs/1809.00888

Agarwal, H., Singh, A., & D, R. (2021, September 21). *Deepfake Detection Using SVM*. Retrieved from IEEE Xplore: https://ieeexplore.ieee.org/document/10721497

Alvin, T. P. (2023, May 12). *Siamese Neural Networks with Triplet Loss and Cosine Distance*. Retrieved from Towards Data Science: https://towardsdatascience.com/siamese-neural-networks-with-tensorflow-functional-api-6aef1002c4e/

Bromley, J., Guyon, I., & LeCun, Y. (1993). *Signature Verification using a "Siamese" Time Delay Neural Network*. Morgan-Kaufmann. Retrieved from https://proceedings.neurips.cc/paper_files/paper/1993/file/288cc0ff022877bd3df94bc9360b9c5d-Paper.pdf

Chollet, F. (2016). *Xception: Deep Learning with Depthwise Separable Convolutions*. Retrieved from ArXiv: http://arxiv.org/abs/1610.02357

Deloitte Centre for Financial Services. (2024, May 29). *Generative AI is expected to magnify the risk of deepfakes and other fraud in banking*. Retrieved from Deloitte Centre for Financial Services: https://www2.deloitte.com/us/en/insights/industry/financial-services/financial-services-industry-predictions/2024/deepfake-banking-fraud-risk-on-the-rise.html

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., . . . Houlsby, N. (2020). *An Image is Worth 16x16 Words: Transformers for Image Recognition*. Retrieved from arXiv: https://arxiv.org/abs/2010.11929

Goldberg, Y., & Hirst, G. (2017). *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers. Retrieved from https://machinelearningmastery.com/exploding-gradients-in-neural-networks/

Grigoryan, A. A. (2023, August 14). *Understanding VGG Neural Networks: Architecture and Implementation*. Retrieved from Medium: https://thegrigorian.medium.com/understanding-vgg-neural-networks-architecture-and-implementation-400d99a9e9ba

Kaushik, A., Doshi, D. N., Mal, S., & Malviya, L. (2024, June 21). *Advanced Deepfake Detection Using Inception-ResNet-v2*. Retrieved from Springer Nature Link: https://link.springer.com/chapter/10.1007/978-981-97-2053-8_11

Khan, S. A., & Dang-Nguyen, D.-T. (2022, September). *Hybrid Transformer Network for Deepfake Detection*. Retrieved from ACM Digital Library: https://dl.acm.org/doi/fullHtml/10.1145/3549555.3549588

Mistry, R. (2025, Feburary 22). *Understanding ResNet: A Deep Dive into Residual Neural Networks*. Retrieved from Medium: https://medium.com/@rohanmistry231/understanding-resnet-a-deep-dive-into-residual-neural-networks-6d8c8c227fd0

Ramanaharan, R., Guruge, D. B., & Agbinya, J. I. (2025, March 28). *DeepFake video detection: Insights into model generalisation — A Systematic review*. Retrieved from Science Direct: https://www.sciencedirect.com/science/article/pii/S2543925125000075

Samrouth, K., Housseini, P. E., & Deforges, O. (2025, February 19). *Siamese Network-Based Detection of Deepfake Impersonation Attacks with a Person of Interest Approach*. Retrieved from ACM Digital Library: https://dl.acm.org/doi/10.1145/3708352

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). *FaceNet: A Unified Embedding for Face Recognition and Clustering*. Retrieved from arXiv: http://arxiv.org/abs/1503.03832

SECURITY HERO. (2023). *2023 STATE OF DEEPFAKES*. Retrieved from SECURITY HERO: https://www.securityhero.io/state-of-deepfakes/#key-findings

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2016). *Grad-CAM: Why did you say that? Visual Explanations from Deep Networks via Gradient-based Localization*. Retrieved from arXiv: https://arxiv.org/abs/1610.02391

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. Retrieved from ArXiv: http://arxiv.org/abs/1905.11946

Li, Z. (2025) 2025WAI VIT image classification, 2025WAI ViT Image Classification. Available at: https://www.kaggle.com/code/ucas0v0zhuoqunli/2025wai-vit-image-classification/notebook (Accessed: 02 June 2025).

Acharya, A. (2023) *Introduction to vision transformers (VIT)*, *Encord*. Available at: https://encord.com/blog/vision-transformers/ (Accessed: 25 May 2025).