Capstone Project Report

James Cook University Cairns

Hunter Kruger-Ilingworth (14198489)

August 14, 2025

Contents

List of Figures

List of Tables

Introduction

"AI as a creative tool has improved leaps and bounds since its early incarnations. Generating still images that are interesting, sophisticated and photorealistic is now an easy process that can be done by anybody with an interest, some patience and determination" [2]. The prevalence of AI among the general public has led to extensive dialogue about their ethical implications in producing artwork, or for manufacturing misinformation. One such example of the use of AI is tools like *DALL-E* and *Stable Diffusion*, which generate images from text prompts. In the current digital age, we cannot rely on the transparency of the source of an image, so there exists a need to be able to identify whether an image has been generated by an AI, or a real artist. Figure 1 shows some examples of AI generated images, which are becoming increasingly difficult to distinguish from real images as the technology improves, underscoring the need for reliable detection methods.

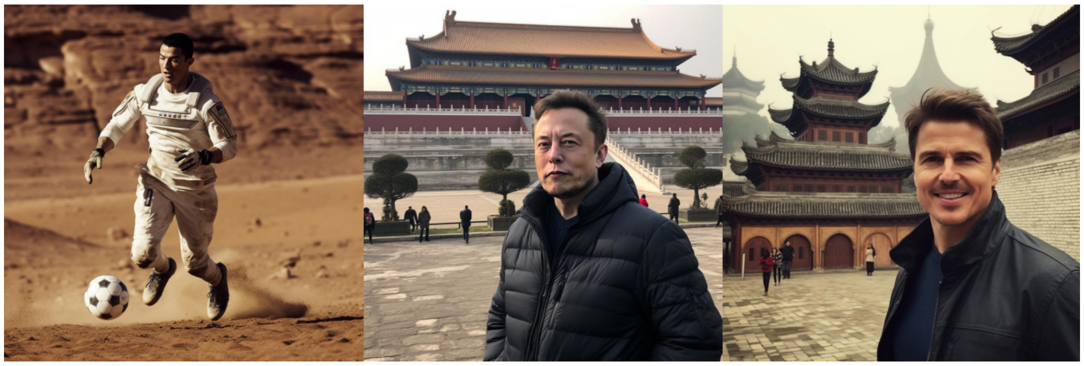

Fig. 1. AI generated images [1]

[1] evaluates both human and AI capabilities in detecting fake images, showing that humans are often deceived by advanced image generation models, while AI-based detection algorithms outperform humans but still misclassify 13% of images. The study introduction of the Fake2M dataset and new benchmarks (HPBench and MPBench) aims to drive further research and improve the reliability of AI-generated content detection.

[3] presents a computer vision approach to distinguishing AI-generated images from real ones, utilising a synthetic dataset created with Latent Diffusion, classification via Convolutional Neural Networks, and interpretability through Grad-CAM. Achieving nearly 93% accuracy, the study also introduces the CIFAKE dataset, a large collection of real and synthetic images, to support further research on the detection of AI-generated imagery.

It is therefore evident that past research has shown that it is possible to classify images as AI-generated or not, with a high degree of accuracy. This report will explore the process of training a neural network to classify images as either AI-generated or not, and then deploying this model to AWS SageMaker for inference, and evaluating it against comparable models.

Dataset

The dataset used for this project was a competition dataset from Hugging Face, held in 2023 [4]. The dataset consists of 62,060 images, and is 2.37GB in size, being pre-split into training and tesing sets, as summarised in tables 1 and 2, where it can be seen that the testing set has the class labels withheld due to the competition setting, restricting this analysis to the 18,618 training images, which we can sub-divide and validate with known labels.

Feature	Description
`id`	Index filename `34.jpg`
`image`	The Image object (rgb 512x512 resolution)
`label`	Binary class label [1=AI, 0=not AI]

Table 1: Dataset features and their descriptions.

Class Label	Train Count	Test Count
AI (1)	10,330 (55.5%)	NA
Not AI (0)	8,288 (45.5%)	NA
Total	18,618	43,442

Table 2: Counts of each class label in the training and testing sets, where it can be seen that the testing set has the class labels withheld due to the competition setting

Listing 1 shows the code used to load the dataset, and split it into training, validation, and test sets have a look at this later sub splitting of the data was reasoned to be the best strategy, as having a smaller dataset on which to train, `small_train`, it would be more conducive to iteration in the hyperparameter tuning process, and the optimal hyperparameters could then be transferred onto the main model, which would be trained on the full training dataset. The code demonstrates how the data is imported. Since Hugging Face datasets are not natively compatible with TensorFlow, the dataset is converted to a TensorFlow dataset using the `to_tf_dataset()` method.

Listing 1: Data Wrangling Script NEED TO CHANGE

```python

with open("token.txt", "r") as file:
    hugging_face_token = file.read().strip() # must have credentials to access the dataset

login(token=hugging_face_token)
raw_dataset = load_dataset('competitions/aiornot') # hugging face method to import the data

dataset = raw_dataset['train'] # remove the unused 'test' set with no labels

# split dataset into:
# - small_train (10% of original)
# - train (70% of original)
# - test (20% of original)

# First split: small_train (10%) and remainder (90%)
split_1 = dataset.train_test_split(train_size=0.1, seed=RANDOM_SEED)
small_train = split_1["train"]
remainder = split_1["test"]
split_2 = remainder.train_test_split(train_size=0.7 / 0.9, seed=RANDOM_SEED)
train = split_2["train"]
test = split_2["test"]

is_any_data_unused = not(small_train.num_rows + train.num_rows + test.num_rows == dataset.num_rows)
assert not is_any_data_unused, "Some data is unused in the splits."

def format_dataset(hugging_face_dataset, batch_size=32, shuffle=True):
    """convert a hugging face dataset to a TensorFlow dataset"""
    dataset = hugging_face_dataset.with_format(type='tf', columns=['image', 'label'], output_all_columns=True)
    dataset = dataset.to_tf_dataset(columns='image', label_cols='label', batch_size=batch_size, shuffle=shuffle)

    # normalse the image channels to be in the range [0, 1] rather than [0, 255]
    dataset = dataset.map(lambda image, label: (tf.cast(image, tf.float32) / 255.0, label), num_parallel_calls=tf.data.
      AUTOTUNE)
    # return prefetched dataset
    return dataset.prefetch(buffer_size=tf.data.AUTOTUNE)

small_train_tf = format_dataset(small_train, batch_size=32, shuffle=True)
train_tf = format_dataset(train, batch_size=32, shuffle=True)
test_tf = format_dataset(test, batch_size=32, shuffle=False)
```

This data was converted to npz files, due to their efficient storage and loading capabilities. After conversion, the data was uploaded to an S3 bucket for easy access during model training and evaluation. Uploading to S3 buckets was critical, as it meant that there was no longer a reliance on the RAM allocation of the sagemaker notebook environment - it could be loaded in from the S3 bucket in batches, which has the benefit of being a more scalable solution.

Modelling

- detail the structure of the model, eg. the training was called in a notebook (main.ipynb) but the training and the model was defined in discrete python files of their own in accordance with modularity best practices.

Model Structure and Hyperparameter Tuning

The model is a convolutional neural network (CNN) designed for binary image classification. It accepts full-size images of shape $(512, 512, 3)$, which are downscaled to $(256, 256, 3)$ for computational efficiency. The architecture consists of two convolutional layers, each followed by a pooling layer (either max or average pooling, chosen as a hyperparameter) and optional dropout for regularization. After flattening, a dense layer with tunable units and optional dropout is added, followed by a single sigmoid output neuron for binary classification.

Key hyperparameters include the number of filters and kernel sizes in each convolutional layer, pooling type, dropout usage and rates, dense layer units, optimizer type, and learning rate. The model uses either Adam or RMSprop optimizers, with the learning rate also being tunable. It was deemed important to test at least 2 different optimisers, as it is known that different optimisers can entail tradeoffs in convergence speed and final loss.

Adam was chosen as one of the considered optimisers, as it is known to converge rapidly, and rectifies vanishing learning rate and high variance, therefore being the most popular optimiser [5]

RMSProp was considered as the second optimizer due to continue

Early stopping is employed to prevent overfitting by monitoring validation loss and restoring the best weights if no improvement is seen for 5 epochs.

Hyperparameter Tuning with Hyperband

The Keras Tuner's Hyperband algorithm is used for hyperparameter optimization. Hyperband efficiently allocates resources by quickly eliminating poorly performing configurations and focusing on promising ones, thus saving computational time compared to exhaustive search. Its main strength is the ability to explore a large hyperparameter space efficiently. However, it may miss optimal configurations if the early stopping criteria are too aggressive or if the search space is not well-defined.

Model Evaluation and Deployment

Using an endpoint, the model was then evaluated on its precision, recall, f1 score, and accuracy. Figure 2 shows the endpoint configuration in Sagemaker. This allowed for evaluation of the models performance, whilst not being limited by the RAM limitations of the notebook environent in Sagemaker studio. Figure 3 presents the confusion matrix for the model's predictions. It can be seen that continue here

Table 3: Tunable Hyperparameters

Hyperparameter	**Range/Choices**
filters1	32 to 128 (step 16)
kernel1	3, 5
layer_1_pool_type	max, avg
dropout1	0.1 to 0.5 (if used)
filters2	32 to 128 (step 16)
kernel2	3, 5
layer_2_pool_type	max, avg
dense_units	64 to 256 (step 32)
dropout2	0.1 to 0.5 (if used)
optimizer	adam, rmsprop
learning rate	1×10^{-5} to 1×10^{-2} (log scale)

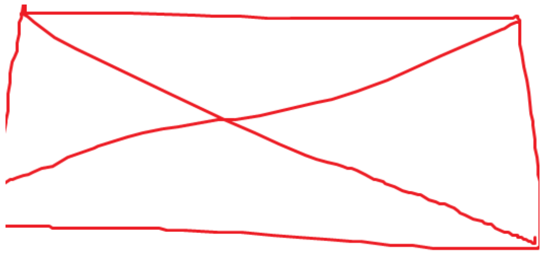

Fig. 2. Evidence of an endpoint in Sagemaker

Transfer Learning

The list of the possible CNN structures available for use for transfer learning can be seen in [6]. Of these, it was reasoned that this model would be either well suited to the characteristics of the dataset or just an effective model for its computational cost, therefore being a candidate for its ability to demonstrate transfer learning with the intention of finding a higher parameter/computational cost model down the track

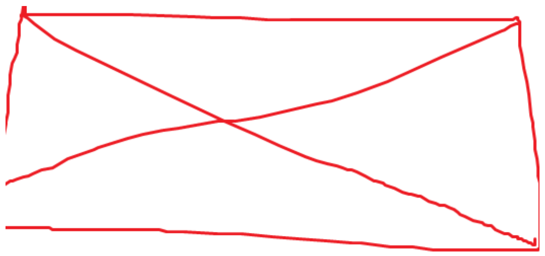

Fig. 3. Confusion Matrix

References

[1] Z. Lu, D. Huang, L. Bai, J. Qu, C. Wu, X. Liu, and W. Ouyang, "Seeing is not always believing: Benchmarking human and model perception of ai-generated images," 2023. [Online]. Available: https://arxiv.org/abs/2304.13023

[2] C. Scott *et al.* (2024) Exploring the ethics of artificial intelligence in art. Accessed: 22 July 2025. [Online]. Available: https://www.artshub.com.au/news/opinions-analysis/exploring-the-ethics-of-artificial-intelligence-in-art-2694121/

[3] J. J. Bird and A. Lotfi, "Cifake: Image classification and explainable identification of ai-generated synthetic images," 2023. [Online]. Available: https://arxiv.org/abs/2303.14126

[4] Hugging Face Datasets, "competitions/aiornot," https://huggingface.co/datasets/competitions/aiornot, 2023, accessed: 22 July 2025.

[5] M. A. K. Raiaan, S. Sakib, N. M. Fahad, A. A. Mamun, M. A. Rahman, S. Shatabda, and M. S. H. Mukta, "A systematic review of hyperparameter optimization techniques in convolutional neural networks," *Decision Analytics Journal*, vol. 11, p. 100470, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772662224000742

[6] K. Team. (2024) Keras applications. Accessed: 22 July 2025. [Online]. Available: https://keras.io/api/applications/