ML Singapore!
Deepfake Face Detection Application

Group 6
Do Gia Hien (A0262365A), He Yifan (A0257866J),
Liang Wenzhong (A0262325L), Luo Zhongyi (A0257979Y)

Abstract

With advances in deepfake technology, detecting deepfake images have become crucial to combat threats like fraud and disinformation. We propose a deepfake face detection application that can automatically identify whether a given face image is real or AI-generated. Our approach leverages efficient transfer learning from state-of-the-art convolutional neural network (CNN) architectures like ResNet, DenseNet, EfficientNet, InceptionNet and VGG on 3 deepfake detection datasets.

Introduction

With the rise of deepfake technology, synthetically generated fake media like images and videos have become a serious concern.

In December 2023, an online video surfaced allegedly depicting Prime Minister Lee Hsien Loong endorsing an investment product, highlighting the risks associated with artificial intelligence and its potential for disseminating misinformation (Ng and Hamzah 2023). Concurrently, a deepfake video featuring Deputy Prime Minister Lawrence Wong promoting an investment scam circulated across social media platforms such as Facebook and Instagram (Koh 2023).

Such deepfake scams could potentially threaten privacy, trust, and social stability in Singapore. Hence, we introduce our deepfake face detection application in order to help with the prevention of such deepfake scams.

In this project, we will define **deepfake images** as images of human faces which are generated by AI models such as Stable Diffusion (Rombach et al. 2021) and state-of-art Generative adversarial networks, specifically StyleGAN (Karras, Laine, and Aila 2019). However, we exclude images taken by humans but are manually processed afterwards (e.g., being photo-shopped), due to ethical concerns (to be discussed in a later section) and difficulties in fetching a sufficient amount of meaningful image data.

Application Details

Our application is an end-to-end deepfake face detection tool, which can be used by Singaporean netizens to correctly identify any deepfake images of human faces they encounter on social media platforms. Specifically, given an image posted on any social media platform, users can first use our application to locate any human face inside the image, and then it will classify the face as either real or fake.

Our application utilises YOLO5Face [1] (Qi et al. 2022), a real-time, high accuracy face detection system, to identify and locate all faces inside the image provided by the users (Figure 1).

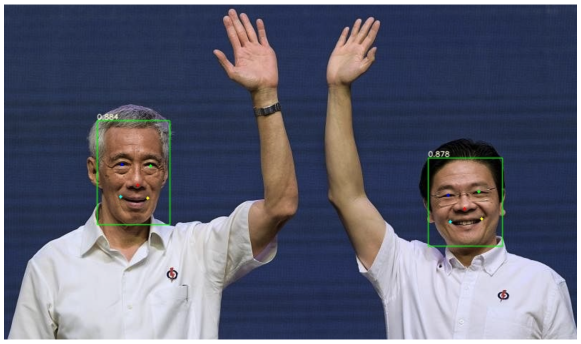


Figure 1: Detection result of YOLO5Face, credit: (Zachariah and Goh 2024)

The core functionality is to classify any faces present in uploaded images as real or deepfake/computer-generated. This will leverage our high-accuracy yet efficient CNN models trained using transfer learning.

Datasets

To train and evaluate our deepfake face detection models, we leveraged three publicly available datasets containing both real and synthetically generated (deepfake) face images:

1. Real vs Fake Faces Dataset (140K images) [2]

[1]https://github.com/deepcam-cn/yolov5-face

[2]https://www.kaggle.com/datasets/xhlulu/140k-real-and-fake-faces

(a) Fake image (b) Real image

Figure 2: Two images from this dataset

This dataset from Kaggle contains around 140,000 images, with an equal split of real and fake (deepfake) face images. The real images are from the Flickr-Faces-HQ (FFHQ) dataset [3], which is a high-quality image dataset of human faces, originally created as a benchmark for GAN, while the fake ones are generated by styleGAN [4]. This large and balanced dataset formed the foundation for our model training.

2. DeepFake and Real Images Dataset (Le et al. 2021)[5]

(a) Fake image (b) Real image

Figure 3: Two images from this dataset

We also utilised this Kaggle dataset which contains around 190K images - 95K real images from various publicly available sources, and 95K deepfake images generated using publicly available deepfake models and techniques.

3. Real vs AI Generated Faces Dataset [6]

(a) Fake image (b) Real image

Figure 4: Two images from this dataset

[3]https://github.com/NVlabs/ffhq-dataset

[4]https://www.kaggle.com/c/deepfake-detection-challenge/discussion/121173

[5]https://www.kaggle.com/datasets/manjilkarki/deepfake-and-real-images

[6]https://www.kaggle.com/datasets/philosopher0808/real-vs-ai-generated-faces-dataset

To increase diversity, we included this dataset for over 50K deepfake face images from FaceSwap [7], Synthetic Faces High Quality (SFHQ) dataset (Beniaguev 2022), Stable Diffusion and ThisPersonDoesNotExist [8].

We combined the images from these three datasets to create our final training set of around 450K images, with a 50-50 split between real and deepfake faces. Then we scaled the images by $1./255$ for normalisation, and performed some basic data augmentation techniques including random crops, flips, and rotations to the training data to further increase diversity.

The datasets cover a wide range of face images in terms of age, gender, ethnicity, background, image quality and deepfake generation techniques. This diversity helps improve the generalisation capability of our trained models and robustness to different deepfake types.

CNN Models

For this project, we will explore using transfer learning from several popular convolutional neural network (CNN) architectures: ResNet, DenseNet, EfficientNet, InceptionNet, and VGG. CNNs are especially useful for computer vision tasks such as image recognition and classification because they are designed to learn the spatial hierarchies of features by capturing essential features in early layers and complex patterns in deeper layers (Craig and Awati 2024). These models were initially pre-trained on large datasets like ImageNet to learn rich representations of natural images.

Transfer Learning

Transfer learning is defined as a machine learning (ML) method that reuses a trained model designed for a particular task to accomplish a different but related task (Kanade 2022).

We can leverage the rich hierarchical representations of the pre-trained models which were learned from massive natural image datasets. This allows for achieving high accuracy on our deepfake detection task using relatively smaller customised training sets.

Layer (Type)	Output Shape
inception_v3	(None, 5, 5, 2048)
GlobalAveragePooling2D	(None, 2048)
Dense	(None, 512)
Dropout	(None, 512)
Dense	(None, 128)
Dropout	(None, 128)
Dense	(None, 1)

Table 1: Transfer Learning Structure (Using InceptionNetV3 as an example)

In this project, we use the structure of layers as shown in Table 1, with the Pooling layer for reducing spatial dimensions, Dense layers with ReLU activation function for training and Dropout layers for regularisation.

[7]https://faceswap.dev/

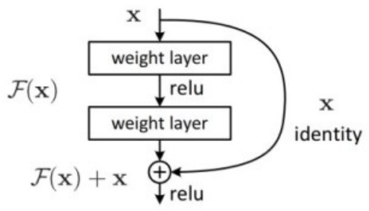
[8]https://thispersondoesnotexist.com

ResNet

ResNet50 (He et al. 2015) is a deep neural network architecture that was introduced by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in 2015. The key innovation of ResNet is the introduction of ”skip connections” or ”residual connections,” which allow the network to bypass one or more layers. This helps to address the problem of vanishing gradients, which can occur in very deep neural networks and make them difficult to train effectively.

ResNet popularizes the approach of using **Skip Connections**. In mathematical terms, it would mean $y = x + F(x)$ where y is the final output of the layer.

In terms of architecture, if any layer ends up damaging the performance of the model in a plain network, it gets skipped due to the presence of the skip-connections.



x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 5: A Skip Connection block

ResNet50 (The version we use for our project) architecture can be broken down into 6 parts:

1. Input Pre-processing
2. $Cfg[0]$ blocks
3. $Cfg[1]$ blocks
4. $Cfg[2]$ blocks
5. $Cfg[3]$ blocks
6. Fully-connected layer

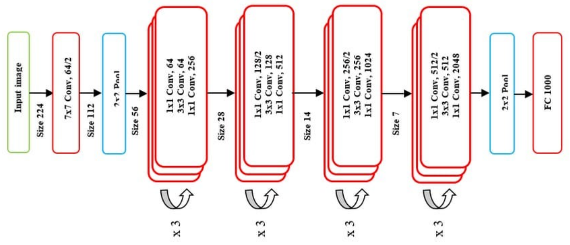


Figure 6: Resnet50 Architecture

DenseNet

DenseNet121 (Huang et al. 2018) is a CNN architecture introduced in 2016 by Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. DenseNet has two key features that make it stand out from other CNN architectures. First, it has a dense block structure, where each layer is connected to every other layer in a feed-forward fashion. Second, it uses bottleneck layers that help reduce the number of parameters without reducing the number of features learned by the network.

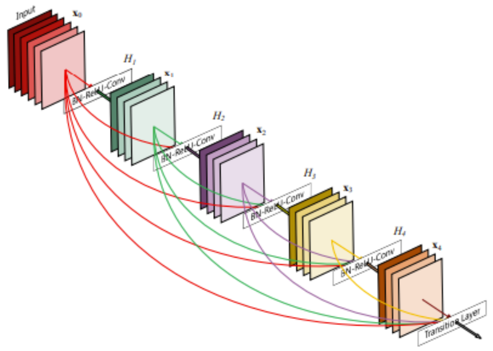


Figure 1: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.

Figure 7: DenseNet Architecture

DenseNets take a different approach compared to traditional CNNs by focusing on reusing features throughout the network. This means they need fewer parameters since they do not learn repetitive information. Some versions of ResNets have also shown that not all layers are necessary, as many contribute little on their own. DenseNets avoid this issue with their structure. They have thinner layers, usually around 12 filters, and they only add a small amount of new features each time. Another issue with deep networks is that they can be tough to train because of the information and gradient flow. DenseNets help solve this issue by making sure each layer can directly access the gradients from the loss function and see the original image, making the training process smoother and more efficient.

In addition, due to the feature reuse, the DenseNets can also achieve higher levels of performance on limited data, making it particularly suitable for tasks where the amount of training data is a constraint. The architecture allows for easy scaling and adjustments, making it adaptable for a wide range of applications beyond image classification, such as segmentation and detection tasks (Ruiz 2018).

EfficientNet

EfficientNetB0 (Tan and Le 2020) is a CNN architecture that uses a compound scaling method, which uniformly scales the network’s width (number of channels in each convolutional layer), depth (number of layers), and resolution of input images with a set of fixed scaling coefficients.

The scaling coefficients control how much to expand the network in terms of depth, width, and input image resolution. By adjusting these coefficients, EfficientNet can be scaled up or down to achieve different trade-offs between model size and accuracy (Kumar 2024).

EfficientNet uses Mobile Inverted Bottleneck (MBConv) layers, which are a combination of depth-wise separable convolutions and inverted residual blocks. Additionally, the model architecture uses the Squeeze-and-Excitation (SE) optimization to further enhance the model’s performance (Figure 8).

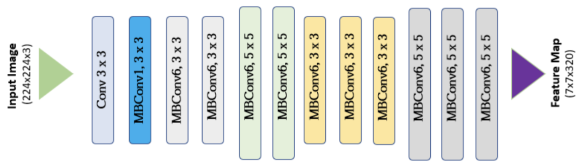


Figure 8: EfficientNet Architecture

This bottleneck design of MBConv layers allows the model to learn efficiently while maintaining a high degree of representational power.

The SE block uses global average pooling to reduce the spatial dimensions of the feature map to a single channel, followed by two fully connected layers, which helps the model learn to focus on essential features and suppress less relevant ones.

These layers allow the model to learn channel-wise feature dependencies and create attention weights that are multiplied with the original feature map, emphasizing important information (Potrimba 2023).

EfficientNet, with its compound scaling methodology, had an impact on our understanding of the balance between efficiency and accuracy in deep learning. By intelligently scaling width, depth, and resolution, it offers versatile models adaptable to various hardware constraints.

The architecture's lightweight and robust design consistently delivers high performance across several computer vision tasks.

InceptionNet

InceptionNetV3 (Szegedy et al. 2014) is a CNN architecture that Google developed to improve upon the performance of previous CNNs on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) benchmark (Russakovsky et al. 2015). It uses Inception Modules (Figure 9) that apply a combination of 1×1, 3×3, and 5×5 convolutions on the input data and utilises auxiliary classifiers to improve performance by learning a combination of local and global features from the input data.

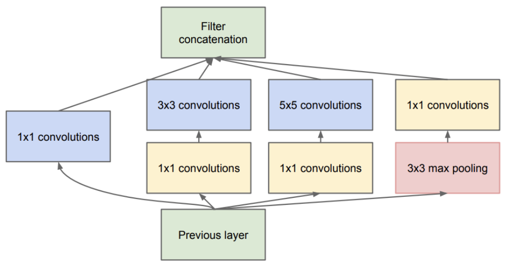


Figure 9: Inception Module

Traditional CNNs typically employ convolutional and pooling layers to extract features from input data. However, these networks have limitations in capturing both local and global features effectively. The inception blocks within the InceptionNet aim to address this issue by facilitating the learning of a blend of local and global features from the input.

Inception blocks tackle this through a modular approach, enabling the network to learn diverse feature maps across different scales. These maps are then concatenated to create a more holistic representation of the input, facilitating the capture of a broad spectrum of features, spanning from low-level to high-level.

Through the integration of inception blocks, the InceptionNet can glean a more comprehensive array of features from input data, thereby enhancing its performance in tasks such as image classification (Sivakumar 2023).

VGG

VGG16 (Simonyan and Zisserman 2015) is a CNN architecture developed for the ILSVRC in 2014 and achieved remarkable success. The VGG16 model architecture (Figure 10) is characterized by its uniformity and depth, comprising 16 layers that have trainable parameters. The model was designed to improve upon the architectures that were prevalent at the time by increasing depth and utilizing smaller 3×3 convolution filters to capture more complex features at various levels of abstraction.

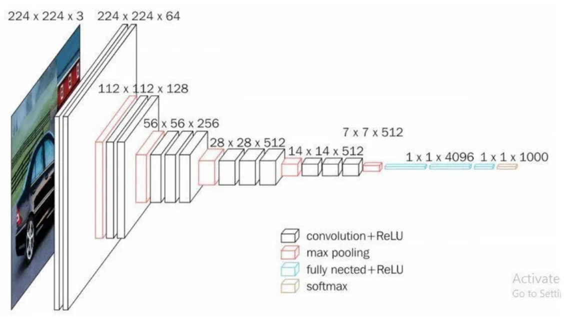


Figure 10: VGG Architecture

VGG16 consists of 13 convolutional layers, 5 max-pooling layers, and 3 fully connected layers, leading up to a final softmax classification layer. Its smaller convolution filters allows simpler structure and easier implementation compared to InceptionNet, while ensuring deeper layers for higher accuracy compared to AlexNet (Ruiz 2021).

VGG16 stands as a landmark in the development of CNN architectures for image recognition. Its design philosophy emphasizes depth and uniformity, proving that deeper networks with small and consistent filter sizes can achieve impressive results on challenging visual recognition tasks.

Experimental Set-up and Outcomes

Data Preprocessing

We utilized the combined dataset of around 9GB of real and deepfake face images (Around 451K images in total). The path of each image file within the three datasets were imported into a Pandas Dataframe along with its labels. After which it is converted to a Keras' DirectoryIterator, so that it can be fed into our model for training, validation and testing. All images were resized to a resolution of 224×224 pixels, and we performed standard data preprocessing, including scaling and data augmentation techniques such as random *rotation, shift, shear, zoom,* and *horizontal flip*. This helped to increase the diversity of the training data and improving the models' generalization capabilities.

Individual CNN models

We utilized the pre-trained architectures from the deep learning framework such as the TensorFlow(v2.10.1) and the Keras API. The individual CNN models were trained using the same datasets and applied the same fine-tuning classification top as described in the Transfer learning section. In addition, we used the Adam optimizer and binary cross-entropy loss function for all the models. Furthermore, to mitigate overfitting, we reduce the learning rate by 80% if the model's validation loss has not decreased for more than 2 epochs. The training was conducted on both a local machine using RTX 4060, and online platforms (e.g. Google Colab and Kaggle Notebook). Our models and code can be found in our GitHub repository.[9]

Ensemble Learning

To further improve the performance of the deepfake detection system, we implemented an ensemble learning approach. We created a new model that took the input image and passed it through each of our fine-tuned CNN models in parallel. The output features from the individual models were then concatenated and fed into a custom classification head, consisting of dense layers and dropout for regularization. The ensemble model was similarly compiled with the Adam optimizer and binary cross-entropy loss function.

Outcomes and Evaluation

We tested our models on a testing subset of the dataset, which consists of around 5.5K images, and evaluated the performance of the individual CNN models and the ensemble model on the held-out test set, using the F1 score and accuracy as the key metrics. The results are summarized in the table below:

CNN Models	F1 score	Accuracy (%)	Inference time (ms)
ResNet50	0.924	91.65	7.4
DenseNet121	0.954	95.13	11.7
EfficientNetB0	0.841	82.72	<u>5.6</u>
InceptionNetV3	<u>0.955</u>	<u>95.17</u>	15.7
VGG16	0.746	72.51	8.7
Ensemble Model	0.961	95.81	27.1

Table 2: Testing Results on RTX 4060 Laptop GPU
(Note: Inference time is per image of size 224×224×3)

[9]https://github.com/cs3264-group-6/Deepfake-Face-Detection

Conclusion and Discussion

Model Comparison

Performance across different models shows different levels of appropriateness in deepfake face detection. For instance, the VGG16 model with its simple structure might not be the best choice for such a complex task due to the large number of images in the dataset and overfitting tendency, thus we decided to not include the VGG16 models in our ensemble learning model as its accuracy and inference time are both the worst.

Among the rest of the models, DenseNet121 and InceptionNetV3 have the highest F1 scores, with InceptionNetV3 marginally leading. However, DenseNet121 has a slightly faster inference time of 11.7 ms compared to InceptionNetV3's 15.7 ms. Both models outperform ResNet50 in accuracy, although ResNet50 has a faster inference time at 7.4 ms. EfficientNetB0, while having the fastest inference time, significantly lags in F1 score and accuracy. In terms of balanced performance, DenseNet121 offers the best trade-off between high accuracy and moderate inference time, making it potentially the best model for applications that require both high predictive performance and speed.

The ensemble model only resulted in a minor improvement in accuracy, which was within the margin of error. Therefore, we decided to select the DenseNet121 model as the final choice for our deepfake face detection application.

While the results are promising, we acknowledge the limitations of our current experimental setup. Our dataset, while diverse, still lacks a substantial amount of real-world face images, which could impact the model's generalization to more diverse, in-the-wild scenarios.

In the future, we plan to expand the dataset by incorporating more real-world face images and exploring alternative model architectures to further improve the detection accuracy and robustness of our deepfake face detection application. In particular, we would like to experiment with the Vision Transformer Model (ViT) (Dosovitskiy et al. 2021), that utilized the popular Transformer model architecture (Vaswani et al. 2023) and saw significant success in the current date Large Language Models. We would also like investigate the performance of the ViT compared to the state-of-art CNN models within the field of deepfake detection.

Interesting Technical Insights

Through this ML project, we have gained an interesting and counter-intuitive insight, which is the limited performance benefits observed from ensemble learning. Prior to this project, we hypothesised that combining predictions from multiple CNN models would allow us to substantially improve the overall robustness and accuracy. However, in our case, we found that simple ensemble averaging did not significantly boost the deepfake detection performance beyond what we could achieve with single, well-tuned models.

There could be a few potential reasons for this. Our binary classification task may be too simplistic for ensemble benefits to be fully realised. Tasks with higher-dimensional output spaces could probably leverage ensemble diversity better. Our ensemble implementation technique could also be sub-optimal, as we used solely CNN models as the weak learner. Other types of ML models with different natures such as ViT, MLP or SVM should be used together with CNN in ensemble learning to form a state of mutual compensation, where the weakness of one type of model can be compensated for by the strengths of another type of model. Moreover, advanced methods like reinforcement learning-based dynamic ensembles could potentially work better. We take this as motivation to continue exploring and understanding about assessing when and how to deploy ensemble methods effectively.

Ethical Concerns

1. Privacy and Consent

   During data collection and usage, we must ensure that the real-person facial image data used for training is collected and used with informed consent. This sensitive personal privacy data must be carefully used and stored, closely following regulations such as the Personal Data Protection Act (PDPA).

2. Bias and Fairness

   We must ensure that the dataset is diverse in different demographics such as age, gender, ethnicity, etc. Otherwise, the model might be biased and exhibit inaccuracy when predicting images with certain characteristics that are under-represented in the original dataset.

3. Misuse of Technology

   This model can also be used to create a more convincing deepfake images, as understanding the detection mechanisms can lead to methods that circumvent them. Thus, there is a risk that deepfake detection technology could be used to generate more authentic deepfake images, which might be even harder for models to distinguish.

Roles and Contributions

Do Gia Hien	Model training (ResNet) ResNet
He Yifan	Model training (InceptionNet) Introduction, Application Details, Datasets, Transfer Learning, EfficientNet & InceptionNet
Liang Wenzhong	Model training (DenseNet, EfficientNet, Ensemble Learning) Experimental Set-up and Outcomes
Luo Zhongyi	Model training (VGG) DenseNet & VGG, Conclusion and Discussion

References

Beniaguev, D. 2022. Synthetic Faces High Quality (SFHQ) dataset.

Craig, L.; and Awati, R. 2024. convolutional neural network (CNN).

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385.

Huang, G.; Liu, Z.; van der Maaten, L.; and Weinberger, K. Q. 2018. Densely Connected Convolutional Networks. arXiv:1608.06993.

Kanade, V. 2022. Transfer Learning Definition, Methods, and Applications — Spiceworks.

Karras, T.; Laine, S.; and Aila, T. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. arXiv:1812.04948.

Koh, S. 2023. Deepfake video of DPM Lawrence Wong promoting investment scam circulating on social media.

Kumar, D. 2024. EfficientNet — Scaling Depth,Width,Resolution - DhanushKumar - Medium.

Le, T.-N.; Nguyen, H. H.; Yamagishi, J.; and Echizen, I. 2021. OpenForensics: Large-Scale Challenging Dataset For Multi-Face Forgery Detection And Segmentation In-The-Wild. In *International Conference on Computer Vision*.

Ng, H. S.; and Hamzah, F. 2023. PM Lee urges vigilance against deepfakes after 'completely bogus' video of him emerges.

Potrimba, P. 2023. What is EfficientNet? The Ultimate Guide.

Qi, D.; Tan, W.; Yao, Q.; and Liu, J. 2022. YOLO5Face: Why Reinventing a Face Detector. arXiv:2105.12931.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2021. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752.

Ruiz, P. 2018. Understanding and visualizing DenseNets.

Ruiz, P. 2021. VGG Very Deep Convolutional Networks (VGGNet) - What You Need to Know.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3): 211–252.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556.

Sivakumar, D. 2023. Introduction to InceptionNet - Scaler Topics.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2014. Going Deeper with Convolutions. arXiv:1409.4842.

Tan, M.; and Le, Q. V. 2020. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv:1905.11946.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2023. Attention Is All You Need. arXiv:1706.03762.

Zachariah, N. A.; and Goh, Y. H. 2024. PM Lee's handover to DPM Wong: Is a snap GE on the cards?