

COMP9417: Machine Learning

2024 T1 – External Group Project: Squad404

George Luo
Z5317638

Yifei Tian
Z5444016

Luke Wang
Z5440342

Xiaolu Liu
Z5415908

Lijun Guo
Z5446133

1.Introduction

Image classification has been an age-old machine learning problem which has had an undeniable impact in numerous fields such as medical imaging, autonomous vehicles and biometrics just to name a few. With constant advancements in technology, we have gained access to increasingly powerful hardware with immense data processing capabilities. This has given rise to highly effective models from complex CNNs such as VGG-19, seen in [Simonyan et al., 2014] to the innovative transformer-based models such as ViT explored by [Dosovitskiy et al., 2021] and beyond. However, these models require substantial data for effective training, presenting challenges in data-constrained environments.

This project aims to explore methods to tackle few shot learning problems in such cases where the dataset is very limited to create a more accessible approach to image classification. We have been challenged to create a model to tackle a binary image classification task that can generalize over a diverse collection of datasets given only 5-10 images per class. We have approached this by incorporating convolutional neural networks in their ability to extract rich and descriptive features, as well as prototypical networks in their ability to generalize well over limited datasets [Snell et al., 2017].

2.Exploratory Data Analysis

We were provided with 3 sample datasets to train our model on. These datasets included ‘Is Epic Intro,’ ‘Needs Respray,’ and ‘Is GenAI.’ All datasets were balanced.

‘Is Epic Intro’ featured a collection of 10 spectrogram images representing the first 30 seconds of a song to determine whether the song has an epic feeling. The class can be determined by analysing the patterns of intensity across different frequencies over time. This dataset limited our ability to augment as many transformations would lead to significant information loss.

‘Needs Respray’ featured a collection of 12 images of pavement to determine if there are active weeds growing indicating the need for a chemical respray. Here, a very small portion of each image differentiated between classes where the features were heavily based on texture and colour, meaning a potential need for a more complex feature extractor. Additionally, certain negative images including active vegetation outside of the region of interest potentially tricked the model into a positive prediction such as the negative example in fig 1. These issues were both addressed with our attention module, specifically the spatial attention component.

‘Is GenAI’ featured a collection of 20 scenery images to determine which are real and which have been generated by AI. This task hinged on subtle nuances in patterns, textures, and their interrelationships—details so fine they were

difficult for the human eye to identify. Our attention module, particularly its channel component, was instrumental in addressing these distinctions.

Overall, each dataset presented unique challenges and required distinct approaches, highlighting the necessity for a model design that effectively generalizes across a diverse range of datasets.

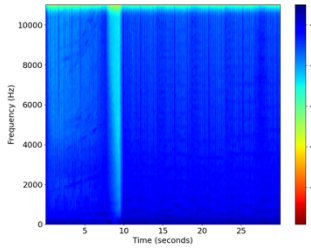
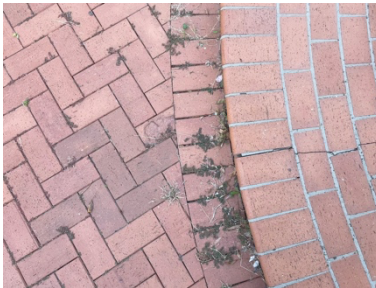

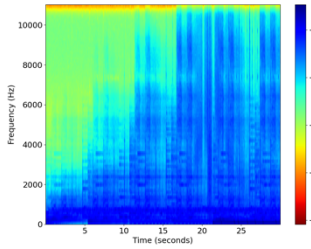
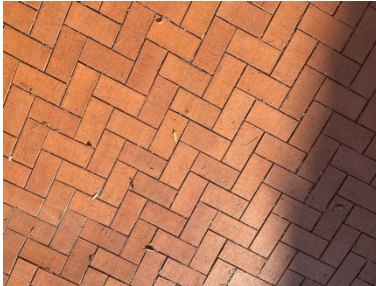

	Is Epic	Needs Respray	Is GenAI
Positive Example			
Negative Example			

Fig 1: Example positive and negative images from all 3 datasets

3.Methodology

3.1 Overview

The design decisions in our model were motivated by our objective to create a system that could extract rich and very specific features yet could still generalize well over an extremely limited dataset. Due to the nature of the datasets provided, where differentiating features were highly specific and textural, a highly descriptive feature extractor was needed; however, using a deep CNN on such limited datasets inevitably led to overfitting. Wrestling with ideas to these issues led us to our final model design. Augmenting the dataset to improve generalization. Using MobileNetv2 as a lightweight feature extractor alongside a CBAM attention module to allow for features that would not overfit, yet still captured necessary information through channel and spatial attention. Then using the extracted features to create support and query sets to classify and train through a prototypical network, which has been proven by [Snell et al., 2017] to be highly effective in generalizing few shot problems.

3.2 Pre-processing/Augmentation

Pre-processing involved resizing images to 224x224 and normalizing pixel values based on mean pixel values of ImageNet, which MobileNetv2 was trained on. We set aside 20% of images as test images.

Augmentation played a critical role in preparing our dataset for effective training by enhancing the diversity of our data to minimize the overfitting inherent in problems with small datasets.

We employed multiple image augmentation techniques to introduce variability in the given image samples, which could assist the model in learning a broader range of perspectives and conditions. The augmentation pipeline includes:

- Random Horizontal and Vertical Flips: These transformations mirror the images along the horizontal and vertical axes, simulating the scenario where objects appear in different orientations.
- Random Rotation: Sample images are rotated by a random angle up to 20 degrees, clockwise or counterclockwise. This variation helps in developing rotational invariance.
- Colour Jitter: Adjustments in brightness, contrast, saturation, and hue are made randomly within a specified range. It helps the model to adapt to various environmental conditions.
- Salt and Pepper Noise: We introduce a specific amount of salt and pepper noise to the images. It simulates real-world corruption seen in digital images, teaching the model to focus on relevant features instead of anomaly sporadic pixels.

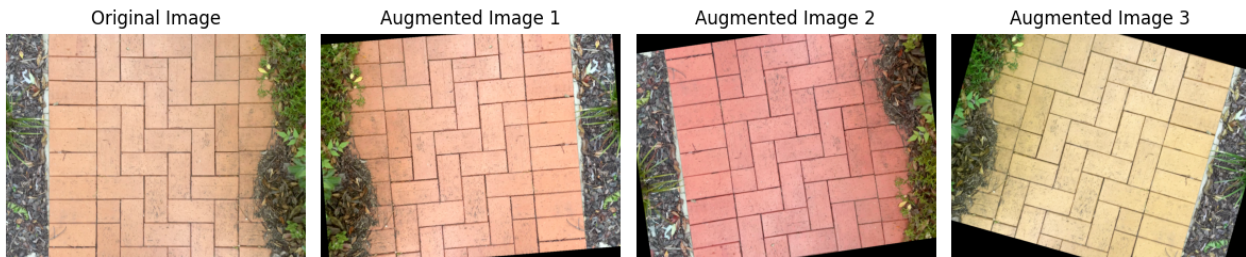


Fig. 2: Examples comparing original and augmented images.

Hyperparameter tuning involved adjusting rotation, colour jitter, and noise levels to enrich the dataset while minimizing information loss. We found that applying 15 augmentations per image struck an optimal balance between dataset diversity and redundancy. These augmentations have contributed positively towards model robustness against overfitting. By diversifying the given original sample images, the model learns to generalize from the augmented dataset, hence improving its performance on unseen data.

3.3 MobileNetv2 Feature Extraction

Convolutional neural networks were ideal for our image classification task due to their ability to extract rich, hierarchical features, making them a natural fit for the provided datasets. Utilizing transfer learning [Hussain et al., 2018] with a pre-trained model enhances efficiency and ensures better generalization, given our dataset sizes. By adopting the pre-trained weights and modifying the top layers, we integrated our own modules to tailor the model to our specific needs.

Initially, we were deciding amongst MobileNetv2 [Sandler et al., 2019] and ResNet18 [He et al., 2015] for our choice of the more appropriate base for our feature extractor.

ResNet18 utilizes residual networks to create a much deeper architecture that extracted more specific features. This seemed crucial due to the nature of the sample datasets where differentiating features were highly textural. For example, in ‘Needs Respray,’ classification focused on small textural differences where the weeds in the negative class appeared more dry. In ‘Is GenAI,’ very subtle differences seemed only capturable by deeper architectures to identify small nuances in how the AI would generate an image.

Ultimately, we chose MobileNetv2 which was a faster and more lightweight CNN that despite less complex features, prevented overfitting due to its simplicity. After testing both bases, it was proven that MobileNetv2 gave better performance. Comparison of results from both architectures are further discussion in the evaluation section. Not only did ResNet18 overfit, but its deep features seemingly confused the model.

MobileNetv2 uses depthwise separable convolutions in place of traditional convolutional layers first introduced in MobileNetv1 [Howard et al., 2017]. This architecture splits a convolution in a depthwise and pointwise operation which greatly reduces computational cost. Given M input channels, the depthwise convolution applies a just single filter per input channel, resulting in M feature maps. These are subsequently combined by a set of $1 \times 1 \times M$ pointwise convolutions. As the operation does not require applying filters across all channels, efficiency is increased whilst maintaining rich feature extraction.

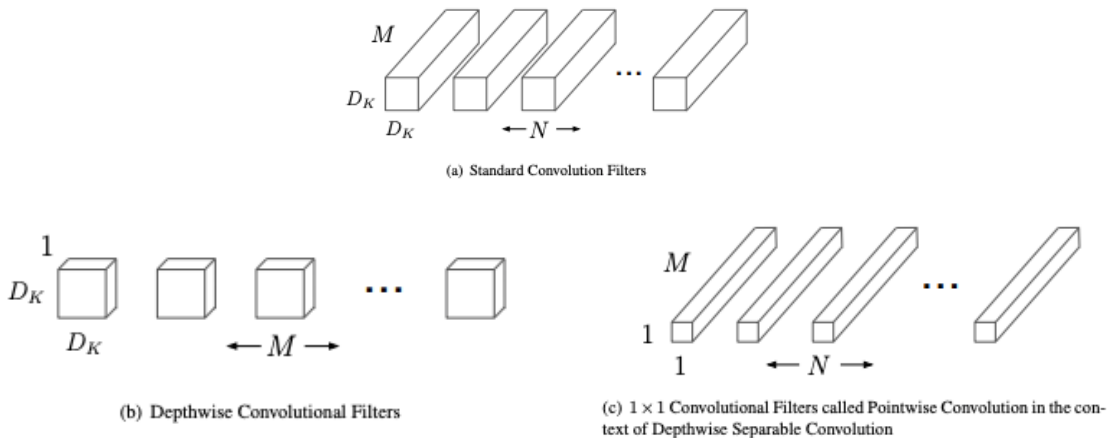


Fig 3: MobileNet's Depthwise and pointwise convolutions (Howard et al 2017)

Building on MobileNetv1, MobileNetv2 additionally introduces inverted residual blocks that further reduces computational costs. These blocks start with a low dimensional input and expand to a higher dimension. It then applies the depthwise convolution before reducing back down to a lower dimensionality [Sandler et al., 2019]. This essentially inverts the traditional residual block architecture seen in ResNet [He et al., 2015].

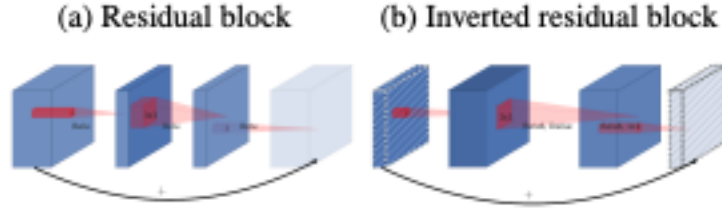


Fig 4: MobileNetv2’s inverted residual blocks (Sandler et al 2019)

3.4 CBAM Attention Layer

Once MobileNetv2 has extracted features from our images, we pass them through a CBAM attention layer based on a paper by [Woo et al., 2018] which allows us to capture more contextual information. This is especially necessary in our model as the selection of a lightweight CNN in MobileNetv2 does not capture sufficiently descriptive information given the sample datasets.

The CBAM model passes the features through a channel attention then spatial attention module sequentially before pooling, applying dropout for better generalization, passing through 2 more dense layers and a sigmoid activation. The result is passed to the prototypical layer. Hyperparameter tuning for these layers such as dropout values and dense layer dimensions were adjusted through thorough testing.

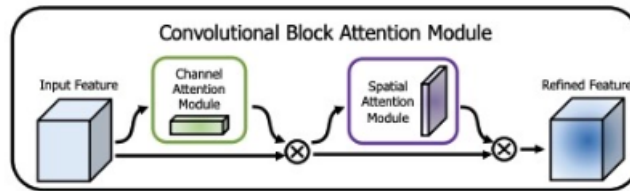


Fig 5: CBAM architecture (Woo et al, 2018)

The channel attention applies a max pool and an average pool to the feature maps extracted from MobileNetv2 to obtain 2 1 dimensional vectors. A sigmoid activation is applied to the sum of the 2 vectors and the resulting vector acts as weights for features. This allows certain features to contribute more, which is especially effective for ‘Needs Respray’ and ‘Is GenAI’ as very specific features within the image are needed to differentiate between classes. For hyperparameters, we adjusted the ratio for dimensionality reduction to 16.

The spatial attention module again applies max pooling and average pooling; however, this is done across the feature maps. Summing the output of the pooling operations creates a map of weights that can be applied to allow certain points of the images to contribute more to the prediction. This is effective especially in ‘Needs Respray,’ where certain images include vegetation outside of the path that is likely to confuse the model and be mistaken for weeds growing within the path as shown in the images in fig 2. For hyperparameters, we used a kernel size of 7.



Fig 6: Channel and attention modules within CBAM (Woo et al, 2018)

Testing our model with the attention module showed clear improvement as it greatly increased the prediction accuracy of both ‘Needs Respray,’ and ‘Is GenAI,’ further discussed in the evaluation section. The inclusion of this module will also inevitably create a safe way to extract better contextual information for new unseen datasets.

3.5 Prototypical Layer

Finally, extracted features are classified using a prototypical layer. Prototypical networks generate a prototype representation for each class based on the mean of its examples. Predictions are made from distance-based calculations from the query point to each class prototype [Snell et al., 2017].

For our model, the data is trained in episodes of randomly generated support and query sets. Once a feature vector is extracted from each example, a class prototype is created as the mean vector of features from the corresponding class. This prototype acts as a powerful representation of its given class, allowing for an accurate prediction despite a small number of examples per class that inherently generalizes well.

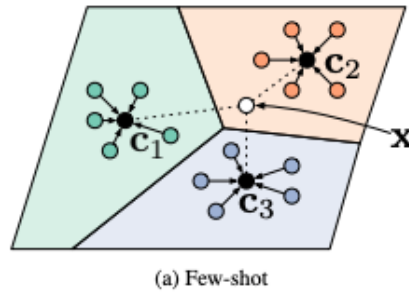


Fig 7: 3 class prototypical network (Snell et al, 2017)

Each query image is classified by calculating Euclidean distance to each prototype and predicted as the closer prototype. For each episode, a cross entropy loss function is applied and back propagated through the model to adjust the parameters. After extensive testing, we chose 5 support images per class, 8 query images per episode and

100 training episodes for a balance between effective training and manageable computational cost. For better convergence, we used an Adam optimizer and included a learning rate scheduler, initialized to 0.001 with a step size of 30 where the learning rate was adjusted based on a gamma value of 0.1 at each step.

At time of prediction once the model has been trained, a support set is created from the original unaugmented dataset. This generates our final class prototypes, factoring in all parameters learned from previous training episodes to predict unseen data.

This architecture allows the model to adapt quickly with minimal data, making prototypical networks a suitable choice for our binary few shot classification model given their inherent ability to generalize well and prevent overfitting.

3.6 Evaluation metrics

We use F1 score as the primary evaluation metric and accuracy as a secondary metric, because in scenarios with imbalanced classes, accuracy can be misleading. Extremely imbalanced datasets can result in high accuracy even if the model consistently predicts the majority class. However, the F1 score considers both precision and recall for each class, providing a more comprehensive evaluation of the model's classification performance.

F1 score also combines precision and recall, offering a holistic assessment of the model's performance. Unlike accuracy, which may mask performance disparities among classes, F1 score emphasizes the model's performance across all classes, offering insights into its overall classification capability.

Meanwhile, accuracy as a secondary metric can offer supplementary insights into model performance.

4.Results

4.1 Training Trends

With our prototypical networks-based model, we tested different configurations of feature extractors. The graph in fig 8 compares the performance of MobileNetv2, ResNet18 and MobileNetv2 with CBAM attention as feature extractors for the prototypical network on the Is GenAI dataset. We evaluated these in training using loss and F1 score, and in prediction using F1 score and accuracy.

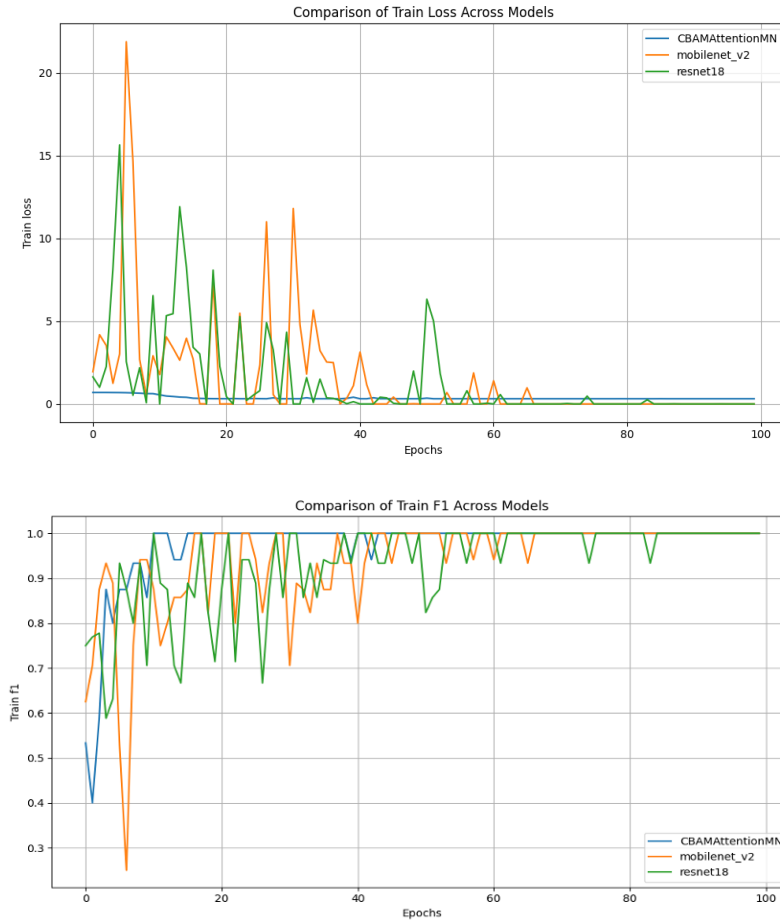


Fig 8: Loss and F1 trends comparison on different feature extractor configurations

From the loss and F1 trends across all configurations, it can be observed that the CBAM module demonstrates remarkable superiority in terms of convergence speed and stability. Its loss stabilizes at around 0 at the 20th epoch, whereas pure CNN configurations experience prolonged periods of instability and larger fluctuations in loss, requiring around 60 epochs to reach a similar level of stability. Similarly, the attention module achieves an F1 score of 1 around the 15th epoch where the pure CNN feature extractors take almost 65 epochs for the same level of performance.

4.2 Prediction result

We also compared these three models in terms of their F1 scores and accuracy for prediction. The results are presented in fig 9 below:

Model	Dataset	F1 score	Accuracy
MobileNetv2 → CBAM attention → Prototypical Network	Is Epic	1	100%
	Needs Respray	0.91	90%
	Is GenAI	0.91	90%
MobileNetv2 → Prototypical Network	Is Epic	1	100%
	Needs Respray	0.83	80%
	Is GenAI	0.75	80%
ResNet18 → Prototypical Network	Is Epic	0.91	90%
	Needs Respray	0.67	50%
	Is GenAI	0.40	40%

Fig 9: Prediction evaluation of different configurations

From the table, it is evident that for all datasets that the CBAM model with the MobileNetv2 base achieved the highest F1 score and accuracy. And it can be observed that MobileNet_v2 also had good performance on three datasets.

5. Discussion

5.1 Model Comparison and Insights

We evaluated combinations of different base CNNs with and without added attention alongside a prototypical layer to classify. Our final model used a MobileNetv2 base with an added CBAM attention module as a feature extractor for a prototypical network which excelled over all datasets. Despite ResNet extracting deeper features, its tendency to overfit makes it less effective than a lightweight feature extractor like MobileNetv2. Although simple CNNs like MobileNetv2 may result in high bias for tasks that require highly detailed feature extraction, this can be addressed

through mechanisms like attention which provide sufficient information through context. The prototypical layer allowed for efficient generalization despite the limited number of examples. Given their nature, it would be interesting to see this model used on a multi-class classification problem, possibly even with a changing number of classes.

5.2 Limitations and Future Work

Prototypical networks proved to be an effective method for this task however, this architecture heavily depends on the quality of the prototypes. If the support set is not a highly generalised representation of its classes (ie there are many outliers), the prototypical network will struggle to classify effectively. A future iteration could potentially include some mechanism to filter poor examples from the support set during training.

In addition, we used a simple feature extractor to reduce variance and although the attention mechanism somewhat amended this issue, this model may still struggle in tasks which require very specific features. Reasonable performance was seen in such cases in 'Needs Respray,' and 'Is GenAI,' however, their results were still lacking compared to a dataset like 'Is Epic.' Future methods should explore ways to extract deep features without losing generalizability.

Prototypical networks are typically used for multi-class problems. It was proved here that they are still applicable to binary tasks however, in future, exploration into an architecture more tailored to binary tasks such as Siamese networks [Koch et al., 2015] may be worthwhile.

6. Conclusion

In this exploration of few-shot image classification, we have created a model that can reliably classify restricted datasets, demonstrating its high feasibility for real-world applications. By integrating data augmentation, using MobileNetv2 as a feature extractor, CBAM to enhance feature attention and prototypical networks as a generalisable classification architecture, we have effectively addressed the challenges posed by the "Is Epic Intro," "Needs Respray," and "Is GenAI" datasets. This integration significantly enhanced the model's focus on subtle textures and contextual details, which are crucial for accurate classification, in a way that can generalize over a diverse range of datasets.

Although the model is currently optimized primarily for binary classification problems, it shows great potential for extension to multi-category problems. Our results suggest that while the model has been able to perform well on a limited number of datasets, larger datasets have the potential to further improve its performance. Subsequent research should continue to explore advanced attention mechanisms and network architectures and extend the model to a wider range of classification problems.

Citations

VGG19: Simonyan, K & Zisserman, A 2014, 'Very Deep Convolutional Networks for Large-Scale Image Recognition', *arXiv*, accessed April 2024, <<https://arxiv.org/abs/1409.1556>>.

Vision Transformer (ViT): Dosovitskiy, A, Beyer, L, Kolesnikov, A, Weissenborn, D, Zhai, X, Unterthiner, T, Dehghani, M, Minderer, M, Heigold, G, Gelly, S, Uszkoreit, J & Houlsby, N 2021, 'An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale', *arXiv*, accessed April 2024, <<https://arxiv.org/abs/2010.11929>>.

Transfer Learning: Hussain, M & Jordan, J & Diego, R 2018, 'A Study on CNN Transfer Learning for Image Classification', *ResearchGate*, accessed March 2024, <https://www.researchgate.net/publication/325803364_A_Study_on_CNN_Transfer_Learning_for_Image_Classification>.

CBAM Attention: Woo, S, Park, J, Lee, J-Y & Kweon, IS 2018, 'CBAM: Convolutional Block Attention Module', *arXiv*, accessed April 2024, <<https://arxiv.org/abs/1807.06521>>.

Prototypical Networks: Snell, J, Swersky, K & Zemel, R 2017, 'Prototypical Networks for Few-shot Learning', *arXiv*, accessed March 2024, <<https://arxiv.org/abs/1703.05175>>.

MobileNetv1: Howard, AG, Zhu, M, Chen, B, Kalenichenko, D, Wang, W, Weyand, T, Andreetto, M & Adam, H 2017, 'MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications', *arXiv*, accessed March 2024, <<https://arxiv.org/pdf/1704.04861.pdf>>.

MobileNetv2: Sandler, M, Howard, A, Zhu, M, Zhmoginov, A & Chen, L-C 2019, 'MobileNetV2: Inverted Residuals and Linear Bottlenecks', *arXiv*, accessed March 2024, <<https://arxiv.org/pdf/1801.04381.pdf>>.

ResNet18: He, K, Zhang, X, Ren, S & Sun, J 2015, 'Deep Residual Learning for Image Recognition', *arXiv*, accessed March 2024, <<https://arxiv.org/abs/1512.03385>>.

Siamese Networks: Koch, G, Zemel, R & Salakhutdinov, R 2015, 'Siamese Neural Networks for One-shot Image Recognition', CMU, accessed April 2024, <<https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf>>