Identifying Dog Breeds from Images: A Deep Learning-Based Approach

Achyut Karnani ak19g21@soton.ac.uk
Samyuktha Babuganesh sb9n22@soton.ac.uk
Abhishek Upmanyu au1u22@soton.ac.uk
Indrajeet Sen is2n22@soton.ac.uk

Abstract

Recent years have witnessed a revolutionary transformation in the analysis and processing of images through the emergence of deep learning and computer vision techniques. Given the availability of various dog breeds and the potential real-world use cases at animal shelters and veterinary offices, the recognition and classification of dog breeds from images was considered.

1. Introduction

The research aims to develop and assess the performance of a state-of-the-art Vision Transformer (ViT) and pre-trained models such as ResNet34, VGG16, and ShuffleNet, for the purpose of categorizing dog breeds based on the "Dog Breed Identification" dataset [10] available on Kaggle. The results produced by the aforementioned models will be compared and evaluated. By conducting a comparative study of the data, we will gain valuable insights into the strengths and limitations of each model. Such analysis will help clarify the applicability of each model and provide a foundation for future tasks related to optimisation.

2. Dataset Analysis

The Dog Breed Identification dataset comprises of images that have been annotated with labels identifying the breed of each dog. The dataset consists of over 10000+ high-quality images of 120 different dog breeds, sourced from various locations. The data was visualised and found to be relatively balanced. The dataset has been divided into three sections: training, validation, and test set. The training set contains 70% of the images, the validation set contains 20%, and the test set contains 10%. All the images are stored in JPG format.

3. Experiment Setup

Training is accomplished using Stochastic Gradient Descent (SGD) optimizer, which has a learning rate of 0.001 and a momentum of 0.9. This approach ensures reliable convergence and minimizes oscillations in the loss function. The model is trained for 10 epochs to optimize computing efficiency while providing adequate exposure to the dataset.

3.1. Data Augmentation

Improving the generalisation of deep learning models, thus preventing them from over-fitting is one of the most important problems in deep learning. Data augmentation is one technique that can be used to reduce over-fitting. The main aim of "Data Augmentation is to bake translational invariances into the dataset such that the resulting models will perform well despite these challenges" [5].

Four distinct data augmentation techniques are used. Run 1 employs standard transformation, while Runs 2-5 employs a variety of techniques, like random rotation (30-degree rotation), Colour Jitter (the brightness, contrast, saturation, and hue of pictures in the training dataset are randomly perturbed), random horizontal flip, random resizing, and centre cropping (256-pixel cropping followed by 224-pixel center cropping). Run 6 used a combination of all the 4 augmentation techniques.

4. CNN

Yann LeCun et al. [6], first developed Convolution Neural Networks (CNNs), that specialise in processing grid-like input, such as photographs. Their architecture takes advantage of local connectivity, shared weights, and translation invariance to enable efficient learning of image features.

- **Convolution layers**: Three convolutional layers with 32, 64, and 128 output channels. Each convolutional layer has a kernel size of 3 and padding of 1.
- **Averaging pooling**: Two max-pooling layers with a kernel size of 2 and stride 2, are applied after each convolutional layer. This is followed by two fully connected layers with 500 and 120 output features.
- **Dropout**: To prevent overfitting, a dropout layer with a dropout rate of 0.3 is placed between the fully connected layers. The dropout layer is then applied, and the fully connected layers are linked sequentially.
- **Output**: ReLU activation is applied to the output of the first fully connected layer. The output of the second fully connected layer determines the model's output.

5. ResNet34

Kaiming He et al. [8], introduced ResNet, a convolutional neural network architecture. The architecture of ResNet includes a residual block consisting of convolution layers, normalisation and ReLU function. The main advancement of ResNet is the addition of "skip connections" or "shortcut connections", which allow the network to learn residual functions rather than the underlying mapping directly. These connections alleviate the vanishing gradient problem. The architecture used for this experiment is as follows:

- **Input**: The image is processed using a convolutional layer that employs 64 filters of size 7x7, with a stride of 2 and padding size of 3.
- **Stage 1**: Following the processing of the input image by the convolutional layer, the output is passed through a max pooling layer with a stride of 2 and a kernel size of 3x3.
- **Stage 2**: The network has 16 residual blocks, each with two convolutional layers and filters of size 3x3. The number of filters increase from 64-512 in each block. The residual connection bypasses the convolutional layers and adds the original input to the second convolutional layer's output. This enables the network to learn more complicated mappings and solve the vanishing gradient problem.
- **Average Pooling**: Following the residual blocks, the output is passed through a global average pooling layer, which averages each feature map in the output.
- **Feed Forward Network**: The global average pooling layer's output is then sent via a fully connected layer with 120 output units.

6. ShuffleNet

ShuffleNet is a very efficient and lightweight convolutional neural network (CNN) architecture presented by Zhang et al. [7], in their 2017 article. The primary principle of ShuffleNet is to lower the computational cost of convolutions while retaining high accuracy by using channel shuffling operations. For cross-channel information flow and to avoid information bottlenecks, the channel shuffle operation combines feature maps from several channels. The architecture used is as follows:

- **Input**: RGB images of size 224x224.
- **Stage 1**: This stage consists of a single convolution layer with an output of 24 channels, batch normalisation layer and ReLU activation function.
- **Stage 2,3,4**: These stages consists of 4,8,4 building blocks respectively which perform a combination of 1x1 and 3x3 grouped convolutions. The output of these blocks start from 58 channels and increase to 464. A channel shuffle operation to the grouped convolution is received by the 3x3 convolution layer. The output from shuffling is then concatenated with input feature maps to preserve spatial features. Then, they are passed through batch normalisation and ReLu Activation function,
- **Stage 5**: This stage consists of a single convolution layer with output of 1024 channels, batch normalisation layer and ReLU activation function.
- **Feedforward Neural Network**: The feature vector produced by global average pooling of stage-5 ouput is fed to a feed forward neural netwok with output of 120 channels.

7. VGG16

Simonyan, Karen & Zisserman, Andrew [4], proposed a Convolutional Neural Network model known as VGG16. The authors "demonstrated that the representation depth is beneficial for the classification accuracy" [4]. The model consists of a total of 16 layers, with 13 Convolution Layers and 3 Fully Connected Layers. The model architecture is as follows:

- **Input**: The model takes images of size 224x224. Moreover, this model was trained on the ImageNet dataset and has a total of around 138 million trainable parameters.
- **Convolution Layer with ReLU**: There were a total of 13 convolution layers stacked on top of each other. The kernel was of size 3x3 and stride of 1. Each convolution layer was followed by the ReLU activation function.
- **Pooling Layer**: After every 2 convolution layers, a Max pooling layer is used, with a kernel size of 2 and stride of 2.
- **Fully Connected Layers**: 3 Fully connected layers with the ReLU activation function are used for classification. The first 2 layers have 4096 units and the last layer has 120 units. The original model was used to classify 1000 classes and so the units of this layer was changed to 120. Moreover, the drop-out layer was also used to prevent over-fitting.

8. Vision Transformer (ViT)

Dosovitskiy A et al. [1], proposed the use of transformer models, for image classification. Moreover, "The pure attention-based mechanism by which ViT models process their inputs is a significant departure from the familiar decade-long ubiquitous use of convolution networks for computer vision" [9].

Vision Transformers consist of the following architecture: Patch Embedding, Classification Token, Positional Embedding, Transformer model and Classification Head.

The images need to be transformed appropriately and thus, will be pre-processed using 3 steps, namely: Patch Embedding, adding the Classification Token and adding the Positional Embedding.

8.1. Pre-processing

Patch Embedding Each image will be split into patches of size 16x16. Thus, for an image of size 224x224, there will be a total of 196 patches per images. Each patch will be flattened using a dense layer and the output of each patch will now be a vector with a dimension of 768 (16x16x3).
Classification Token Transformer models require an additional token, known as the class token, in order to perform classification tasks. This is a learnable parameter added to the top of the patches.
Positional Embedding The order of the patches is not preserved. This is essential information and so positional embeddings were used to ensure that the information related to the location is preserved.

8.2. Transformer model

Vaswani, Ashish & Shazeer, et al. [2], proposed the transformer architecture in their famous paper "Attention Is All You Need", where the transformer consists of both encoder and decoder layers. However, for Vision Transformers, only the encoder layer will be used. The Encoder layer consists of 3 sublayers, namely: Multi-Head Attention layer, MLP layer and the Norm Layer.

Multi-Head Attention Layer (MHSA)

- **Query, Key and Value Matrices**: The patch embeddings along with the class token and the positional embedding will be passed to this layer, where the Query, Key and Value matrices are computed using Linear layers, without an activation function. These matrices are representations of the patches of images.
- **Attention Matrix**: The attention matrix is constructed using the Query and Key matrices, which represents the relevance between patches of images.
- **Softmax Layer**: The softmax function is applied to get a probability distribution over the data.
- **Attend**: The dot product of the output of the softmax layer and the Value matrix is given as the final output of the MHSA layer.

MLP Layer This is a sequential multi-layer perceptron, which consists of 2 linear layers, the GELU activation layer and 2 drop-out layers to prevent over-fitting.

Norm Layer Geoffrey et al. [3], proposed the Layer normalisation, similar to batch normalisation but with some variations. Batch normalisation calculates the mean and standard deviation over a batch size, whereas Layer normalisation calculates this over each neuron in the layer and so, is independent of the batch.

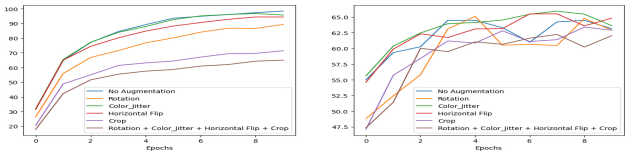
8.3. Classification Head

The output of the transformer, will be used to perform image classification and this can be done using the Class token appended to the embedding. Only the class token is sent to a softmax layer to output the probability of the classes.

9. Performance

ResNet, ShuffleNet, VGG16 and Vision Transformer models were experimented with using 5 different geometric transformations, namely: Rotation, Colour Jitter, Horizontal Flip, Crop and a combination of all 4. The training and validation accuracy of the same are shown in figures 1, 2, 3 and 4 respectively. The corresponding test accuracy of the models are shown in tables 1, 2, 3, 4.

A Baseline CNN model was also run for 10 epochs and was not able to achieve results comparable with the pre-trained models. Thus, the values were not used for comparison and evaluation.



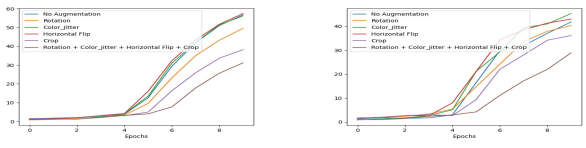
(a) Training Accuracy (b) Validation Accuracy

Figure 1: ResNet - Training and Validation Accuracy

ResNet34

Transformations	Test Accuracy
None	61.48%
Rotation	61.89%
Colour Jitter	62.22%
Horizontal Flip	62.78%
Crop	**63.12%**
Rotation, Colour Jitter, Horizontal Flip and Crop	62.58%

Table 1



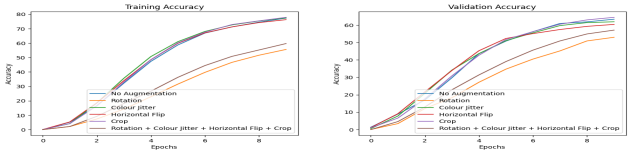
(a) Training Accuracy (b) Validation Accuracy

Figure 2: ShuffleNet - Training and Validation Accuracy

ShuffleNet

Transformations	Test Accuracy
None	52.05%
Rotation	44.03%
Colour Jitter	**52.05%**
Horizontal Flip	52.04%
Crop	52.04%
Rotation, Colour Jitter, Horizontal Flip and Crop	38.06%

Table 2



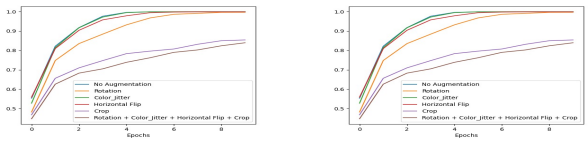
(a) Training Accuracy (b) Validation Accuracy

Figure 3: VGG16 - Training and Validation Accuracy

VGG16

Transformations	Test Accuracy
None	62.13%
Rotation	51.95%
Colour Jitter	66.63%
Horizontal Flip	**67.41%**
Crop	59%
Rotation, Colour Jitter, Horizontal Flip and Crop	56.84%

Table 3



(a) Training Accuracy (b) Validation Accuracy

Figure 4: Vision Transformer - Training and Validation Accuracy

10. Discussion

The Vision Transformer (73.78%) and VGG16 model (67.41%) had the best test accuracy when horizontal flip was used as the geometric transformation. ShuffleNet shows comparable test accuracy values for Colour Jitter (52.05%), Horizontal Flip (52.04%) and Crop (52.04%). Lastly, ResNet has the highest test accuracy of 63.12% with

Vision Transformer

Transformations	Test Accuracy
None	68.29%
Rotation	26.41%
Colour Jitter	41.58%
Horizontal Flip	**73.78%**
Crop	58.81%
Rotation, Colour Jitter, Horizontal Flip and Crop	36.49%

Table 4

the Crop transformation. Thus, Horizontal Flip was found to result in considerably higher accuracy, when compared to the other geometric transformations and was thus, chosen as a benchmark for comparison. Moreover, the test accuracy values of all the models showed that rotation is the transformation that gives the lowest performance.

Dosovitskiy A et al. [1], trained Vision Transformers on the ImageNet-21k dataset, which consists of 14 million images and showed that the model was able to achieve a higher performance on multiple state-of-the-art benchmarks. They had also experimented on medium sized datasets and found the resulting accuracy to be lower than ResNet. In our approach, we made use of pre-trained Vision Transformers but had also experimented with ViT without pre-trained weights and found that the former was able to produce results better than ResNet34, ShuffleNet and VGG16 when Horizontal flip was applied. However, the ViT model without the pre-trained weights failed to converge, as reported in [1]. Hence, the pre-trained Vision transformer was trained and the results were recorded.

Thus, ResNet, ShuffleNet, VGG16 and the Vision Transformer models with horizontal flip were compared. The Vision Transformers model with pre-trained weights and Horizontal flip transformation resulted in the highest test accuracy for multi-class image classification.

11. Conclusion

In conclusion, this study has demonstrated the effectiveness of state-of-the-art deep learning techniques such as Vision Transformers (ViT), and deep pre-trained CNNs. The comparison and evaluation of various models such as ResNet34, VGG16, ShuffleNet and ViT have provided insights into the strengths and limitations of each approach.

There is a scope for future research to explore other advanced deep learning techniques and architectures that can further improve the accuracy and efficiency of these models. For instance, more advanced techniques such as Generative Adversarial Networks (GANs) and Diffusion Models can be used to generate more synthetic training data to further improve the performance of these models.

References

[1] Dosovitskiy A et al. (2020) "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale," arXiv, (2020 10 22).

[2] Vaswani, Ashish & Shazeer, Noam & Parmar, Niki & Uszkoreit, Jakob & Jones, Llion & Gomez, Aidan & Kaiser, Lukasz & Polosukhin, Illia. (2017). Attention Is All You Need.

[3] Ba, Jimmy & Kiros, Jamie & Hinton, Geoffrey. (2016). Layer Normalization.

[4] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556.

[5] Shorten, C. and Khoshgoftaar, T. M. (2019) "A Survey on Image Data Augmentation for Deep Learning," Journal of Big Data, 6(1), pp. 1–48. doi: 10.1186/s40537-019-0197-0.

[6] Gradient-based learning applied to document recognition. (1998, November 1). IEEE Journals & Magazine — IEEE Xplore. https://ieeexplore.ieee.org/document/726791

[7] Zhang, X. et al. (2017) ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices, arXiv (Cornell University). Cornell University. Available at: https://doi.org/10.1109/cvpr.2018.00716

[8] He, K. (2015, December 10). Deep Residual Learning for Image Recognition. arXiv.org. https://arxiv.org/abs/1512.03385

[9] Bhojanapalli, Srinadh & Chakrabarti, Ayan & Glasner, Daniel & Li, Daliang & Unterthiner, Thomas & Veit, Andreas. (2021). Understanding Robustness of Transformers for Image Classification.

[10] Will Cukierski (2017). Dog Breed Identification. Kaggle. https://kaggle.com/competitions/dog-breed-identification