Categorizing Equus Members with Deep Learning

Miguel Corona

May 2020

1 Introduction

The objective of the project was to develop a convolution neural network capable of distinguishing the members of the Equus genus which is composed of horses, donkeys, and zebras. The network was limited to the classification of the pure breeds and thus hybrids such as mules are omitted as these are part horse and part donkey. The development of the model is documented within the report along with attempts to improve its performance. The model's performance was compared to that of one using a pretrained convolution base to assess any potential improvements.

2 Data Preparation

2.1 Image Selection

A total of 1044 images of horses, zebras, and donkeys were collected for use of the convolution neural network. The images were selected such that each image only contains the animal associated with the image's label. Any images containing hybrids or containing animals belonging to another classification other than the image's label were excluded. Images with multiple instances of the same class were permitted into the dataset.

2.2 Preprocessing Data

The images were selected to emphasize their respective label. Images containing multiple labels were cropped to ensure that a single class is present within the image. Some samples are derived from a single image if cropping an image was capable of producing an additional sample of its respective class. All samples were adjusted to ensure they adhere to the 256 x 256 dimensions with the use of PIL.

2.3 Data Distribution and Visualization

The distribution of the data is defined in Table 1 along with the visualization of the data in Figure 1. A data imbalance exists in the datasets as there are

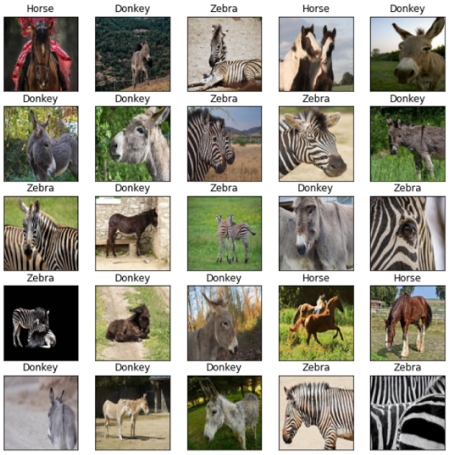


Figure 1: Visualization of Equus Samples and Labels

Table 1: Data

Donkey	391
Horse	312
Zebra	339

more samples of donkeys than those of either the zebra or horse class. The favoring of donkey samples is due to the variability in the species. There was an initial concern that not having enough samples of the donkey class may lead to the mislabeling of a young horse or young zebra.

2.4 Data Normalization

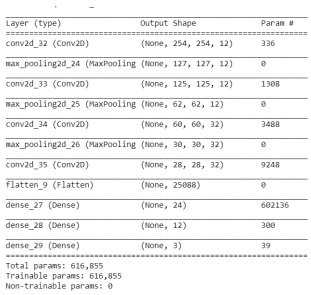
The samples were normalized along the RGB channels within the images. The normalization on these channels shall prevent any potential skewing that may occur due to the presence of grayscale images within the datasets. The model shall be capable of converging to the desired trainable parameters much more quickly with normalized channels.

3 Split and Evaluate on Test Set

A series of convolution neural networks were constructed to evaluate against a validation set and a test set. The candidate models were evaluated via hold-out validation which restricts the number of samples available for training as the validation set is treated similarly to a test set and not used in training. The 1044 samples were distributed across the sets as described in Table 2.

Table 2: Data Split

Dataset	Donkey	Horse	Zebra	Total
Training	240	185	185	610
Validation	75	65	64	204
Test	76	64	90	230



Layer (type)	Output Shape	Param #
conv2d_32 (Conv2D)	(None, 254, 254, 12)	336
max_pooling2d_24 (MaxPooling	(None, 127, 127, 12)	0
conv2d_33 (Conv2D)	(None, 125, 125, 12)	1308
max_pooling2d_25 (MaxPooling	(None, 62, 62, 12)	0
conv2d_34 (Conv2D)	(None, 60, 60, 32)	3488
max_pooling2d_26 (MaxPooling	(None, 30, 30, 32)	0
conv2d_35 (Conv2D)	(None, 28, 28, 32)	9248
flatten_9 (Flatten)	(None, 25088)	0
dense_27 (Dense)	(None, 24)	602136
dense_28 (Dense)	(None, 12)	300
dense_29 (Dense)	(None, 3)	39

Total params: 616,855
Trainable params: 616,855
Non-trainable params: 0

Figure 2: Benchmark Architecture

The architecture resulting from adjustments to the hyperparameters based on performance on the validation set is given by Figure 2.

Evaluations on the validation set were performed with the use of EarlyStopping and ModelCheckpoint callbacks to select the training parameters that yield the minimal validation loss. The loss on the validation set serves as an estimate to the loss expected on the test set and as such the parameters that yield the minimal loss on the validation set were selected for evaluation on the test set. The objective was to acquire a model capable of generalizing the data and bias on the training set was sacrificed to ensure improved performance on the validation and test sets. The learning curves are depicted on Figure 3 along with the accuracy and loss metrics in Table 3.

The contents of Table 3 indicate that the model achieved statistical power as its accuracy is greater than 33%. The learning curves indicate that the accuracy and loss for the training data and the validation data followed the same trend

Table 3: Benchmark Evaluation

Dataset	Loss	Accuracy
Training	0.3065	0.8934
Validation	0.6688	0.7304
Test	0.7379	0.7304

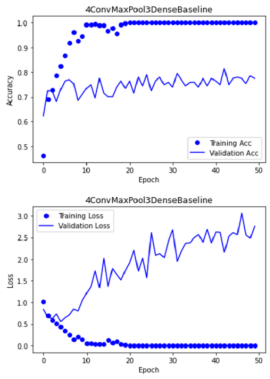


Figure 3: Evaluation on Validation Set

through the seventh epoch prior to diverging. The divergence is an indicator that the data begins to overfit the data as the performance on the training set improves as the evaluations on the validation set degrade. The model performed better on the on evaluating the validation set than on the test set and this may be attributed to the data imbalance that exists between the validation and test sets. A validation set that more resembles the test set in terms of class representation may alleviate the discrepancies between the validation and test metrics.

Two additional models were constructed with modified hyperparameters to observe they're evaluations against the datasets. The architecture of these additional candidate models are depicted in Figure 4. The candidate models were trained and evaluated against the validation set to yield the learning curves as depicted in Figure 5 along with the metrics in Table 4.

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 254, 254, 6)	168
max_pooling2d_3 (MaxPooling2	(None, 127, 127, 6)	0
conv2d_5 (Conv2D)	(None, 125, 125, 6)	330
max_pooling2d_4 (MaxPooling2	(None, 62, 62, 6)	0
conv2d_6 (Conv2D)	(None, 60, 60, 16)	880
max_pooling2d_5 (MaxPooling2	(None, 30, 30, 16)	0
conv2d_7 (Conv2D)	(None, 28, 28, 16)	2320
flatten_1 (Flatten)	(None, 12544)	0
dense_3 (Dense)	(None, 24)	301080
dense_4 (Dense)	(None, 12)	300
dense_5 (Dense)	(None, 3)	39

Total params: 305,117
Trainable params: 305,117
Non-trainable params: 0

Model: "sequential_2"

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 254, 254, 12)	336
max_pooling2d_6 (MaxPooling2	(None, 127, 127, 12)	0
conv2d_9 (Conv2D)	(None, 125, 125, 32)	3488
max_pooling2d_7 (MaxPooling2	(None, 62, 62, 32)	0
flatten_2 (Flatten)	(None, 123008)	0
dense_6 (Dense)	(None, 24)	2952216
dense_7 (Dense)	(None, 12)	300
dense_8 (Dense)	(None, 3)	39

Total params: 2,956,379
Trainable params: 2,956,379
Non-trainable params: 0

Figure 4: Candidate Models

The results of the candidate models' evaluations on the validation and test set as depicted in Figure 5 and Table 4 indicate that the number of parameters influenced by the hyperparameters affect the model's performance. The Benchmark model and the Half Filter model yield a comparable test accuracy even with each layer in the Half Filter containing half the filters as the equivalent

Table 4: Candidate Model Evaluations

Dataset	Model	Loss	Accuracy
Training	Half Filter	0.2869	0.9148
Validation	Half Filter	0.7116	0.7255
Test	Half Filter	0.7297	0.7174
Training	Reduced Layers	0.2666	0.9311
Validation	Reduced Layers	0.7205	0.6961
Test	Reduced Layers	0.6856	0.7087

layer in the Benchmark model. The validation loss and accuracy were found to be better in the Benchmark model than the Half Filter model suggesting that there is a need for complexity in terms of trainable parameters to yield accurate results. The Benchmark model also performed better on the validation set than the Reduced Layers model as expected due to a deeper network's ability to learn more information. Although the Benchmark contains more layers, it is not more complex in terms of trainable parameters as the the Reduced Layers model as the Reduced Layers model contains more parameters. The increase in trainable parameters in the Reduced Layers is due to the lack of additional MaxPool layers that exist in the Benchmark. The added complexity due to the additional trainable parameters enables the Reduced Layers model to overfit the data much sooner than the Benchmark model further suggesting that the Benchmark model is the candidate model suitable for further study.

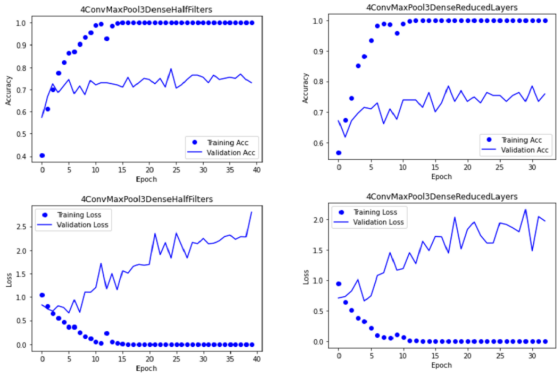


Figure 5: Candidate Evaluations

4 Overcoming Overfitting

The prior executions indicate that the model overfits the training data and fails to generalize the data. The candidate models for the Equus classifier were

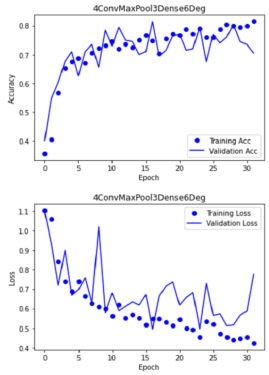


Figure 6: Data Augmentation: Rotations and Translations

capable of yielding strong accuracy on the training data, but did not meet the same level on the validation and test sets evaluations indicating that the training data is within the model's capacity. The capacity of a model is tied to the number of trainable parameters which can be interpreted as the model's ability to memorize characteristics of the training data. Data augmentation and regularization were individually applied to observe the manner in which each these these overcome the overfitting.

4.1 Effects of Data Augmentation

The evaluations indicate that the training data is within the model's capacity and that increasing the number of samples can potentially alleviate the overfitting of the limited training set. Adding samples to the training can provide new information into the model and thus combat overfitting, but was not an option due to the limited size of all the datasets. No new samples were collected nor were the validation and test set samples redistributed into the training set. New samples were provided into the model with the use of data augmentation in which a new sample is derived from an existing sample. The augmented samples were derived from the samples in the training set as the validation set and the test set must not be altered due to the need of having a representation for the generalization error.

The samples in the training set were augmented so that the images contain translations and rotations. The training samples were modified with the use of the ImageDataGenerator. The images were transformed with the configurations: rotations were restricted to a maximum of 10 degrees, translations were contained within 20% along each axis, and nearest was set as the fill mode for the images.

Table 5: Effects of Data Augmentation

Dataset	Loss	Accuracy
Training	0.5037	0.8033
Validation	0.4592	0.8088
Test	0.5737	0.7565

The results of the data augmentation execution along with the learning curves are contained in Figure 6 and Table 5. The loss and accuracy metrics on the validation and test sets improved due to the data augmentation. The introduction of data augmentation permits additional epochs for training as the model is no longer prone to the overfitting as depicted by the learning curves. Each epoch introduced a sample that was new to the model allowing for the model to adjust its trainable parameters based on the loss on the new sample. The learning curves for the validation accuracy and the validation loss are more aligned with those of the training set. The divergence in the accuracy and loss metrics were alleviated through the introduction of augmented images as the model must learn through new images per epoch.

4.2 Effects of Regularization

4.2.1 Regularization

Regularization can prevent complex models from overfitting the training data by placing penalties on the model for its complexity or through the addition of noise. Applying a regularization technique on the parameters allows for models with the same hyperparameters to yield different training parameters that may result in different outputs. L2 regularization apply a cost dependent on the training parameters and aim to set the parameters to values that yield less volatile outputs. The training parameters can be perturbed by adjustments to the architecture by adding artificial noise to a layer's output via dropout.

4.2.2 L2 Regularization

Ridge regression, or L2 regularization, adds a cost proportional to the square of each trainable parameter onto the loss function. The loss contribution due to the square of each weight's magnitude is controlled via the regularization parameter.

The results of L2 regularization are contained in Figure 7 and Table 6 and indicate that an appropriate regularization parameter can prevent or minimize the divergence between the training and validation loss. The learning curves for regularization parameter 0.01 converge the accuracy and loss of the validation set to those of the training set, but the loss and accuracy on the datasets are

Table 6: Effects of L2 Regularization

Dataset	Reg Param	Loss	Accuracy
Training	0.001	0.5159	0.8197
Validation	0.001	0.6519	0.7549
Test	0.001	0.6123	0.7652
Training	0.01	0.7043	0.7689
Validation	0.01	0.7766	0.7010
Test	0.01	0.8716	0.6652
Training	0.0005	0.4197	0.8525
Validation	0.0005	0.5930	0.7353
Test	0.0005	0.6147	0.7739

not better than those of the benchmark. Improvements in performance were observed with a regularization parameter of 0.001 and 0.0005 albeit with learning curves indicating that divergence occurred between the training and validation sets.

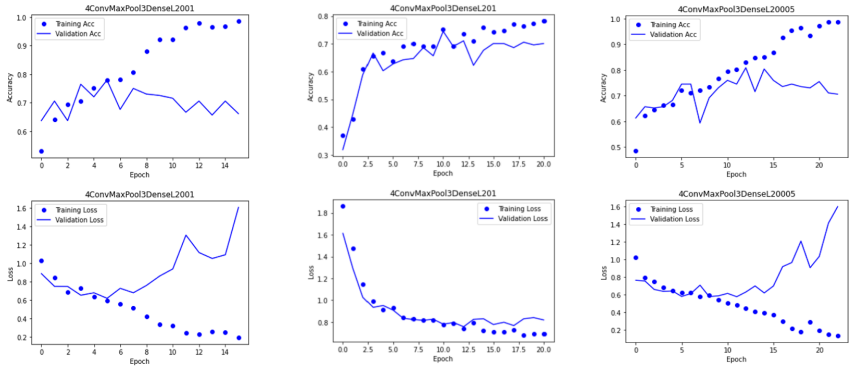


Figure 7: L2 Regularization

L2 regularization yielded performance improvements in loss and accuracy over the benchmark. The classification accuracy and the loss improved as the regularization parameter decreased indicating that the model may need larger weights to minimize the loss. Regularization parameters with values between 0.001 and 0.0005 may be worth further study as 0.001 parameter keeps the curves aligned prior to diverging resuming divergence with value 0.0005.

4.2.3 Dropout

Dropout is a regularization technique in which noise is injected into the output of a layer by randomly setting tensor values to zero. The portion of tensor values

Table 7: Effects of Dropout

Dataset	Drop Rate	Loss	Accuracy
Training	0.2	0.4410	0.8049
Validation	0.2	0.5589	0.7745
Test	0.2	0.4784	0.7609
Training	0.35	0.4137	0.8443
Validation	0.35	0.5482	0.7843
Test	0.35	0.5321	0.7565
Training	0.5	0.5892	0.7443
Validation	0.5	0.6081	0.7353
Test	0.5	0.5975	0.6870

set to zero are dependent on the dropout rate which traditionally ranges between the 0.2 and 0.5. We observe the behavior of the network with a Dropout layer introduced into the network.

The results of dropout indicate that the model's performance benefits from the inclusion of Dropout layers. The learning curves and evaluation metrics are contained within Figure 8 and Table 7. The loss and accuracy improvements were present across the training, validation, and test set for each dropout rate. The model benefited from the noise introduced in each epoch from the dropout, but the noise failed to mitigate the divergence in the learning curves. Deviations in the learning curves are not surprising due to the randomized nature of dropouts on each epoch. The dropout rate of 0.35, the midpoint of the range, appears to yield the greatest improvement though it is difficult confirm without additional executions.

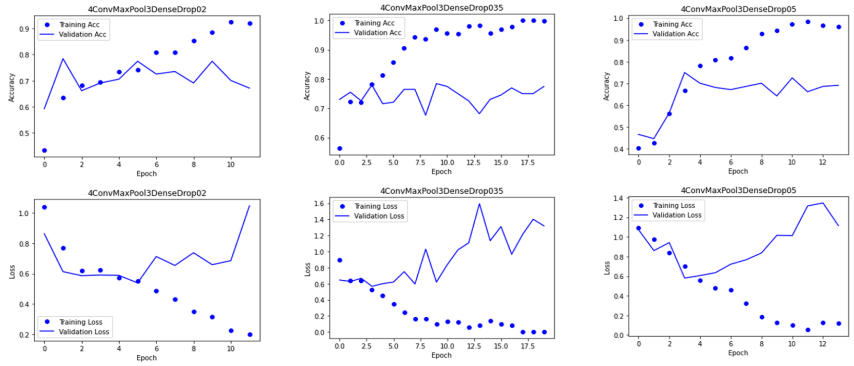


Figure 8: Dropout

Table 8: Evaluation Pretrained Network

Dataset	Loss	Accuracy
Training	0.0151	1.0000
Validation	0.2398	0.8971
Test	0.2199	0.9087

5 Using a Pretrained Architecture

Pretrained networks are networks whose parameters have been saved after training on their respective dataset. The use of pretrained networks is an effective strategy to develop deep learning on a limited sized dateset as the Equus dataset even if the outputs of the pretrained network and Equus classifier are disjoint. The information learned from the pretrained network can be utilized by the developed classifier in its training via feature extraction.

5.1 Feature Extraction

Feature extraction is the practice of utilizing the patterns learned from the convolution base of a pretrained network to train a classifier. The convolution base knows generic patterns from its training that can be applied onto another classifier such as the Equus classifier. The VGG16 convolution base was selected for feature extraction to feed its recognized features into the Equus classifier. The layers within the convolution base were frozen to ensure that only the trainable parameters of dense network were modified across the learning iterations.

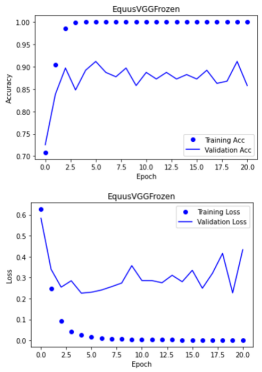


Figure 9: Feature Extraction

The results indicate that the classifier benefited from the use of the VGG16 convolution base. The learning curves and metrics are depcited in Figure 9 and Table 8. The patterns known by the convolution base were utilized in the training of the dense classifier and its performance and accuracy across the datasets

improved compared to those of the benchmark. The loss values acquired with feature extraction are minimized to yield stronger accuracy metrics that the benchmark could not achieve. The learning curves indicate that the validation loss fluctuating during the epochs, but was bound as the loss never reached the extent of the benchmark. The loss learning curves indicate that the model overfits the data as is expected due to the small sample size and the omission of data augmentation. The accuracy on the training set converges to its maximum value and the accuracy of the validation set improves considerably with respect to the benchmark. The convolution base of the VGG16 model is portable and can be used additional classifiers.

6 Conclusion

The objective of the exercise was met as a convolution neural network with statistical power was constructed for Equus classification. An initial model was developed and capable of overfitting the data prior to the evaluations against the validation and test sets. The hyperparameters were adjusted to yield the minimal loss on the validation loss as the validation loss serves as an estimate to the generalization loss. The resulting model remained prone to overfitting after adjustments to the hyperparameters. The effects of data augmentation and regularization on mitigating overfitting were observed in order to improve the model's ability to generalize the data. The efforts involved in developing the Equus classifier produced a model with statistical power for the model's architecture. Feature extraction was performed with the Equus classifier to observe the benefits of utilizing a pretrained network's convolution base. The use of a pretrained network outperformed the model indicating that the data is suitable for a model to learn and additional adjustments can be made to the model's architecture to further improve its performance.