Arijit Bhattacharjee
Phil Davies
Maryann Staincliffe
Dongwen Luo
DATA 601
10 February 2023

Develop image processing application to detect and classify goat hooves

Abstract

The hoof conformation of dairy goats serves as an indicator of the health of the animal, as poor conformation is associated with increased risk of hoof lesions and lameness, decreased reproductive performance, and reduced milk production [1]. There has been a recent study by Deeming et al. that developed a reliable method for assessing hoof conformation in dairy goats using subjective measures. However, subjective scores have poor inter and intra-observer reliability and are affected by experience [1]. Therefore, the aim was to develop an image processing application to automatically detect hoof deformity from photos and to produce summary outputs. 343 photographs used in Deeming et al. paper were used as the training data set and the metadata of the subjective score (toe length, heel shape, fetlock shape, claw splay, and claw shape) was used for labeling it (refer to Appendix A). Convolutional Neural Network was used to train the models leveraging transfer learning to identify and classify hooves. Trained models were downloaded and integrated with Shiny App which acted as a user-facing application. The model was able to detect 79.6% of hooves and classify 87.2% of these correctly. This suggests that image-processing models with summary outputs can be implemented to monitor animal health.

Introduction

This project was approved by AgResearch, Ruakura. AgResearch is the Crown Research Institute tasked with delivering leading agricultural science and innovation to benefit the wider New Zealand economy [2].

This project aims to develop image processing techniques to automatically detect hoof conformation from photos and to produce a Shiny App with summary outputs to give the farmer an overview of their animal health.

Hoof conformation refers to the physical dimensions and shape of the hoof. As hoof conformation serves as an indicator of an animal's health, an accurate assessment of hoof conformation helps in the identification of at-risk animals [1]. Aspects of the subjective hoof conformation described by Deeming et al. were used in dairy goats. The assessment included five subjective scores: (1) toe length, (2) heel shape, (3) fetlock shape, (4) claw splay, and (5) claw shape. Each aspect was scored on a 3-point ordinal scale (0, 1, and 2), except for the fetlock shape, which was scored on a binary scale (0 or 1); a 0 was 'normal' in all cases. Claw splay was scored, only when the claw shape was scored as a 0 (i.e., both claws were straight) [1]. Deeming et al. paper gives a visual representation of the different hoof aspects and their respective scores (refer to Appendix A).

An animal with good conformation will have a score of 0 across the hoof aspects [1]. The hooves photographs used in Demming et al. paper are to be used as our training dataset. The goat farms allowed the use of photographs for internal research work of AgResearch only and hence these photos can't be published or shared outside of AgResearch.

Developing an image processing model to detect hoof deformity automatically involves annotating the images to create masks, which can then be used by Mask R-CNN[4] for predicting the hoof and classifying the image. Mask R-CNN is a Convolutional Neural Network developed using Faster RCNN to solve image segmentation tasks [3]. It has a two-stage framework. The first stage scans the image and generates areas likely to contain an object (proposals). The second stage classifies the proposals and generates bounding boxes and masks [3]. It is simple to train and easy to generalize to other tasks. It is one of the high-performing models and an efficient method. Potential limitations can be long training time, good resolution images, and a high number of images required for training. As the training dataset is small, the approach of transfer learning was selected to load initial model weights which are pre-trained weights using the COCO dataset [5]. As the transfer learning completes, the trained weight for our model is saved as an H5 file (refer to Appendix B) that can be used to make predictions. Most of the code was written in python. Integrating the image processing model with the Shiny App (refer to Appendix C) involved using the Reticulate[6] package in R (refer to Appendix D). It allows the incorporation of python functions and scripts into the R code [6]. The Shiny App prompts a user to load an image and produces an input image with the bounding boxes, masks, class labels, prediction scores over detected hooves, and summary text like class and probability which can be used to monitor the health of the animals.

It is important to ensure that reliable results are obtained. Assessment of the hoof conformation needs to be validated to ensure results are accurately indicating how the summary output relates to the conformation. A way to validate summary output is to use a different set of hoof images as a validation and test set and compare the output against the actual subjective measures.

Data

Training dataset

The hooves photographs used in Demming et al. paper on hoof conformation was available in the animal welfare drive at AgResearch. The photographs were of the left front and left hind hoof in the lateral and dorsal aspect of 1035 goats across 16 farms. A digital camera (Canon Powershot, SX530) was used to take photographs. Photographs were taken in the yards outside of the milking parlor where goats were standing on a horizontal level concrete surface, which ensured they were bearing weight evenly on all four limbs. Photographs were taken at approximately 50 cm from the goat [1]. A number of CSV files with Image ID and subjective hoof score for that image were also available in the animal welfare drive (refer to the supplementary material).

The number of potential images that can be used for training was 2070. However, due to issues with hooves being dirty, missing metadata (subjective scores), missing photos, and poor photo quality [1], only 343 photos were found to be suitable for the training data. Out of the total 343 photos, 178 photos were taken from the dorsal aspect and the remaining 165 were taken from the lateral aspect (Table 1). Photographs taken from the lateral aspect can be used to assess three out of the five subjective scores: (1) toe length, (2) heel shape, (3) fetlock shape. The remaining two: (4) claw splay, and (5) claw shape can be assessed using photographs taken from the dorsal aspect [1].

An issue of class imbalance was observed in the training dataset (Table 1). Less number of images were available for highly deformed (higher scores) hooves.

Class	Shape	Splay	Growth	Heel	Fetlock
0	83	33	59	86	157
1	70	36	67	45	8
2	25	14	39	34	NA
Total	178	83	165	165	165

Table 1 Training data distribution across hoof aspect and subjective score

Test and validation dataset

The first step involved submitting an application on Agresearch's Animal Ethics system for taking goat hoof photographs as it involved handling animals. Approval was obtained from the ethics committee for taking photographs on a single farm with 15 goats. An iPhone was used to take photographs of all four hooves of each goat. Two photographs per hoof were taken: (1) lateral aspect and (2) dorsal aspect. To ensure more image count, photographs were taken before and after hoof trimming. A total of 240 images were taken but 116 images were rejected because of poor angle, paint, or hairy hoof. Photographs were taken in the outside yards and goats were standing on a horizontal level ground, which ensured they were bearing weight evenly on all four limbs. Photographs were taken at approximately 50 cm from the goat. A whiteboard with information like date, hoof, name, and trim was kept beside the hooves to be referred to later for identification if required. A CSV file was created containing an Image ID to capture respective subjective hoof scores (refer to the supplementary material).

Of the remaining 124 photos (79 dorsal aspect and 45 lateral aspect), 62 photos (40 dorsal aspect and 22 lateral aspect) were randomly chosen to be used as the validation set (Table 2) and the remaining (39 dorsal aspect and 23 lateral aspect) was kept aside to be used as the test set (Table 3).

None of the goats had highly deformed hooves, which resulted in test and validation data being not representative of all classes

Class	Shape	Splay	Growth	Heel	Fetlock
0	26	21	21	11	22
1	14	5	1	11	0
2	0	0	0	0	NA
Total	40	26	22	22	22

Table 2 Validation data distribution across hoof aspect and subjective score

Class	Shape	Splay	Growth	Heel	Fetlock
0	34	29	23	13	23

1	5	5	0	10	0
2	0	0	0	0	NA
Total	39	34	23	23	23

Table 3 Test data distribution across hoof aspect and subjective score

Methodology

Data retrieval

Photos that would be used in the training were copied from the animal welfare drive and saved to NeSI infrastructure. Images with dorsal aspect were loaded to the hoof_shape and the hoof_splay folders. Images with lateral aspect were loaded to the hoof_side folder (refer to Appendix E). A single CSV file for each aspect (lateral, dorsal) was created containing a unique Image ID, renamed Image ID, and subjective scores referencing the many CSV files in the animal welfare drive (refer to supplementary folder). Photos with missing subjective scores in the CSV file were not considered for training. Poor quality photos and photos with dirty hooves were previously deleted from the animal welfare drive and were not present when we had access to it. The final count of photos that were of good quality and had respective subjective scores was 343.
Photos to be used as validation and test set and their respective CSV file containing subjective scores were also uploaded to NeSI (refer to Appendix E).

Data strategies

The class imbalance in the training dataset needs to be handled, otherwise, it could lead to an inefficient model. As there are more examples of undeformed hooves, the model will get exposed to such images more frequently during training and pick more of these signals relative to a deformed hoof. This may result in incorrect classification for deformed hooves. The best option would be to collect more images of deformed hooves that can be added to the training dataset. But considering the unavailability of that option currently, class weights can be used to handle the issue of class imbalance while training the model. Another option would be to supplement training data with augmented images. Moreover, the number of images is less than what is generally required for training CNN models, and getting more images is not a viable option at this point. So, transfer learning can be used. Supplementing training data with augmented images would also be helpful.

Task strategies

It was emphasized that having an initial working application would be more helpful. So the initial focus was to set up the system and pipeline, train a working model and get a Shiny App running with bare minimum details. Improving the model, pipeline and the Shiny App would be a subsequent and continuous process.

Tools and system setup

VGG Image Annotator was used to annotate the hoof and the subjective hoof aspect score in the images [7]. Version 2 was used.

Mask R-CNN project from GitHub that works on TensorFlow 2.0 was used [4]. The Mask R-CNN model generates bounding boxes and segmentation masks for each instance of an object in the image. It's based on Feature Pyramid Network (FPN) and a ResNet101 backbone[3]. The README.md file within the GitHub repository [4] chalks out the steps to set up the library and get it working.

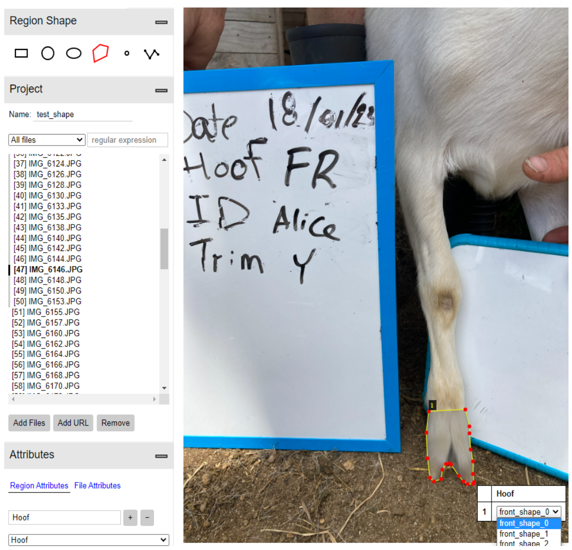
The local system and the current infrastructure was found inadequate to train the model. We leveraged computer resources and infrastructure provided by NeSI (New Zealand eScience Infrastructure) (refer to Appendix I) to train the Mask R-CNN model. A pipeline for training and validation was set up using Jupyter notebooks. The computer resources available to us included 4 CPUs, 128 GB RAM, and a GPU (seven instances with 5GB each). A couple of virtual environments were created to meet the package installation requirements and corresponding two kernels to run the notebooks - one with GPU and another without GPU (for debugging and other small tasks). There was also an option to adjust CPU count and RAM configurations based on our usage requirements.

RStudio was installed locally to build the Shiny App and the required packages were installed. The Anaconda tool was also installed locally. A virtual environment with the same libraries and the software that were used to train and test the Mask R-CNN model in the NeSI server was created. This particular Conda environment was set as the Python interpreter for R (Tools →

Global Options → Python → Python interpreter → Conda Environments). The reticulate package would be able to incorporate python functions and scripts in R [6].

Annotation

For claw shape, photos taken from the dorsal aspect were loaded into the tool (selecting Project → Add files) and annotated on a 3-point ordinal scale (0, 1, and 2) in accordance with the metadata in the CSV file (selecting Attributes→ Type → dropdown) and the assigned class name (refer to appendix G). A single image was selected and the boundary was defined as polygons (Figure 2a, 2b).



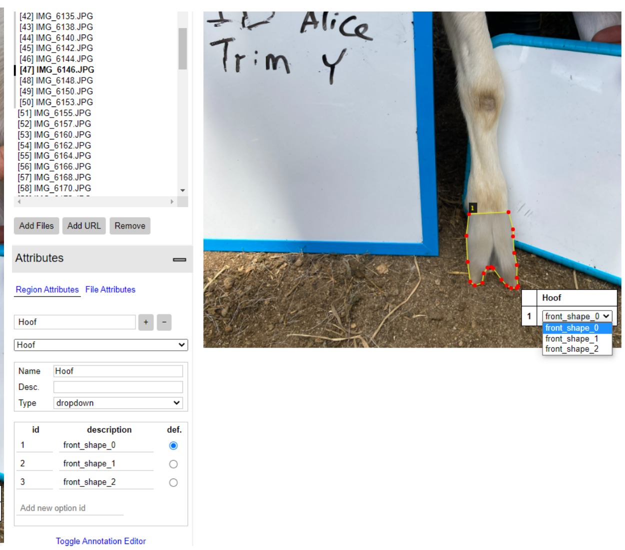


Figure 2a File upload and select region Figure 2b Attribute selection and labeling

Left-clicking the mouse on the image selects the vertex of the polygon and 'Enter' was pressed on the keyboard once the relevant region was enclosed within the polygon. The clicking process was repeated for all images. Once all the uploaded images were labeled, a single file with all the annotations was exported (select Annotation → Export annotations COCO format) and saved in a JSON format (refer to Appendix H).

Similar steps were repeated for claw splay and a different JSON file was exported and saved. Photos taken from the lateral aspect were loaded and sequentially annotated for toe length (0, 1, and 2), heel shape (0, 1, and 2), and fetlock shape (0 and 1) (refer to Appendix H). The three separate JSON files were exported, saved and uploaded to NeSI (refer to Appendix E). We got a total of five annotated JSON files for five hoof aspects.

Model

The jupyter notebook and other files are arranged in the NeSI system according to the directory structure shown in the appendices (refer to Appendix E).

The jupyter notebook (please refer to the supplementary folder for code) imports all the required libraries. A Class was defined to load the dataset and annotations (JSON file), add masks and also read the number of classes according to our classification requirement. Next, the data set was defined and loaded. Data was visualized before starting the training (Figure 3a, 3b). The image shape was (1024, 1024, 3) for all images.

Figure 3a Visualize instance mask for a given image

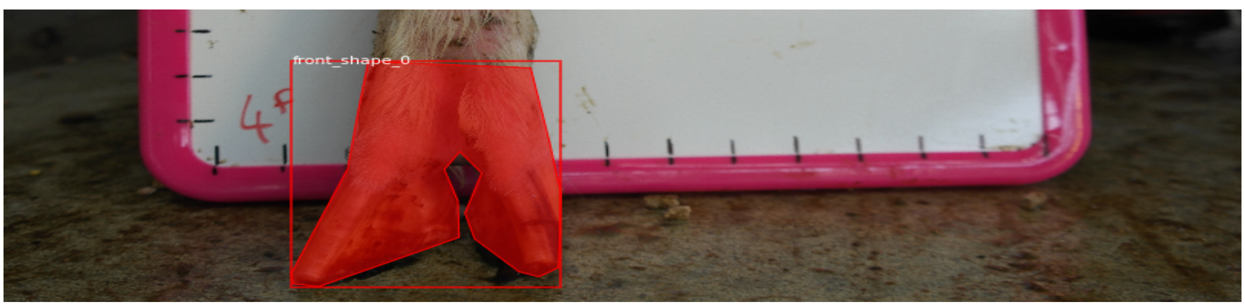


Figure 3b Visualize image with mask and bounding box

Configuration for the model was defined that includes names of the configuration, number of steps per epoch, learning rate, number of classes, and batch size. Then the directory to save log files and trained models were defined. Transfer learning was used and the path for loading the pre-trained weights was set. Next, we defined the model, loaded pre-trained weights, and trained the model.

Different configurations (batch size, learning rates, and epochs) were tried out (Table 4) for claw shape.

No of epochs	Batch size	Learning rate
47	2	0.001

50	4	0.001
50	2	0.0001

Table 4 Configurations for training claw shape model without image augmentation

The models were automatically saved in the designated folders (logs folder in our case) (refer to Appendix E). The validation loss curve for each model was plotted manually (Figure 4a,4b,4c) as the NeSI system doesn't support GUI (refer to Appendix J).

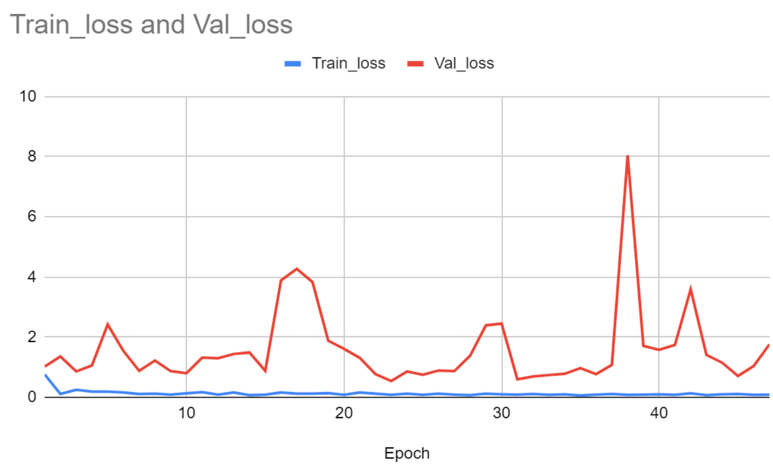


Figure 4a Training and validation loss curve for learning rate 0.001 and batch size 2

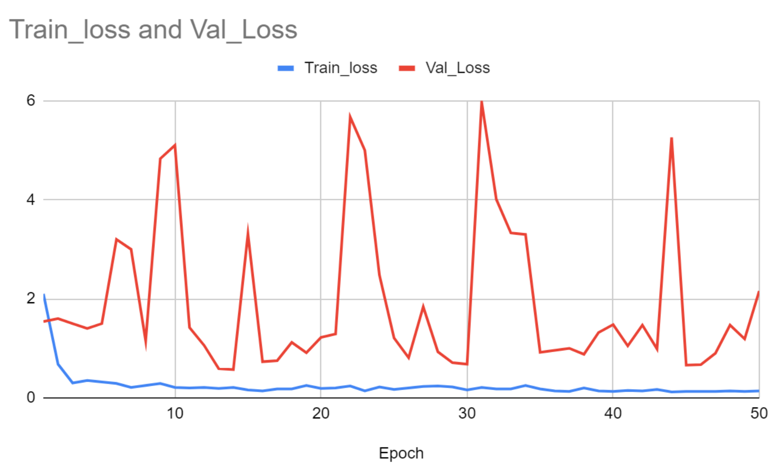


Figure 4b Training and validation loss curve for learning rate 0.001 and batch size 4

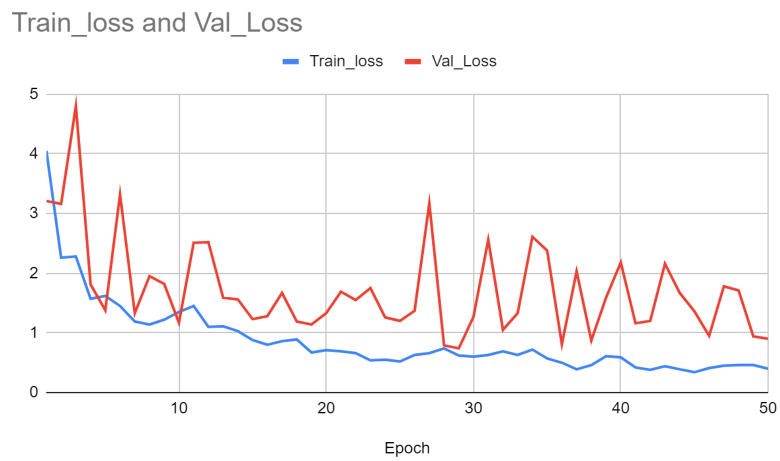


Figure 4c Training and validation loss curve for learning rate 0.0001 and batch size 2

A couple of additional configurations were tried with augmented images (Table 5). Models were trained, saved, and the validation loss curve was plotted (Figure 5a, 5b).

No of epochs	Batch size	Learning rate	Augmentation
44	2	0.0001	Fliplr(1), Flipud(1), rotate=(-45, 45), rotate=(-90, 90), scale=(0.5, 1.5)
29	2	0.001	Fliplr(1), Multiply((0.8, 1.2), LinearContrast((0.75, 1.5)

Table 5 Configurations for training claw shape model with image augmentation

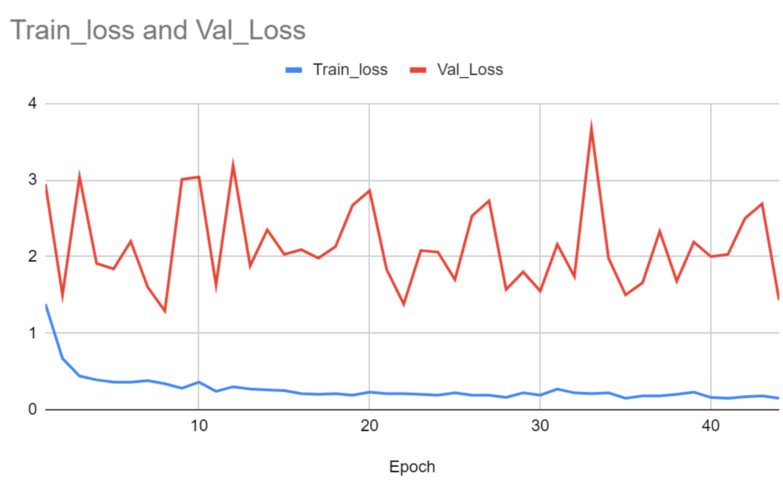


Figure 5a Training and validation loss curve for learning rate 0.0001 and batch size 2

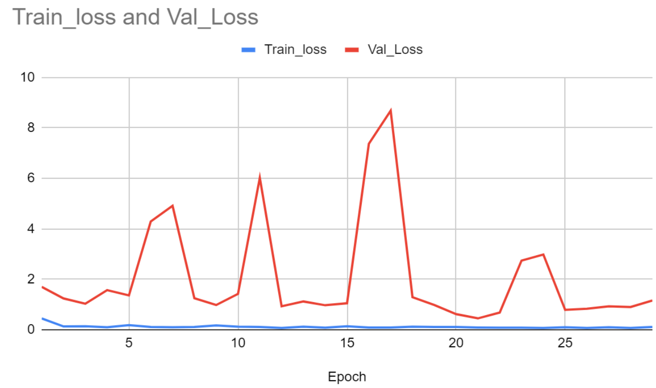


Figure 5b Training and validation loss curve for learning rate 0.001 and batch size 2

A function to calculate the Mean Average Precision[9] was defined (please refer to the supplementary folder for code). For each configuration, the model with the lowest validation loss was selected and the Mean Average Precision was calculated on the test data. The model with the highest Mean Average Precision on the test data, across different configurations, was selected as the best model.

Similar steps were followed for the other four different hoof aspects (toe length, heel shape, fetlock shape, and claw splay). Configuration for each model was different like configuration names, steps per epoch, and the number of classes (please refer to the supplementary folder for code). However, multiple configurations and respective validation loss plots couldn't be tried out, as GPUs were in high demand and unavailable most of the time in the last few days (refer to Appendix J).

Configuration used for claw splay was Number of epochs = 40, Batch size = 2, and Learning rate = 0.001. The validation loss curve was plotted (Figure 6)

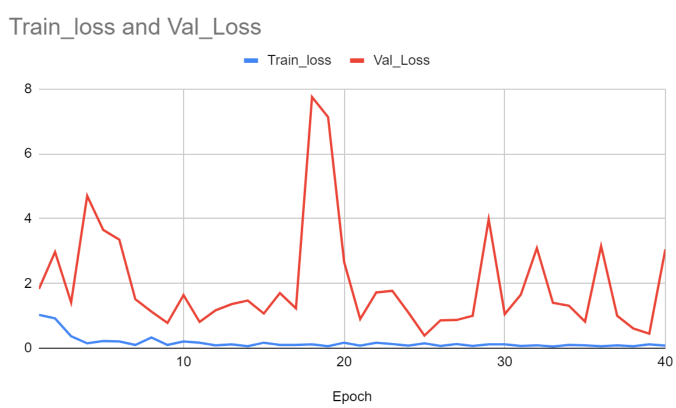


Figure 6 Training and validation loss curve for learning rate 0.001 and batch size 2

Configuration used for toe length or growth was Number of epochs = 40, Batch size = 2, and Learning rate = 0.001. The validation loss curve was plotted (Figure 7)

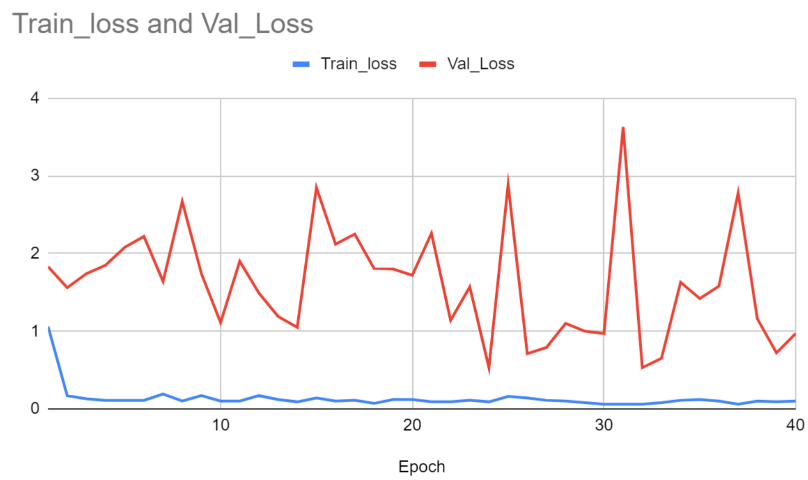


Figure 7 Training and validation loss curve for learning rate 0.001 and batch size 2

Configuration used for heel shape was Number of epochs = 40, Batch size = 2, and Learning rate = 0.001. The validation loss curve was plotted (Figure 8)

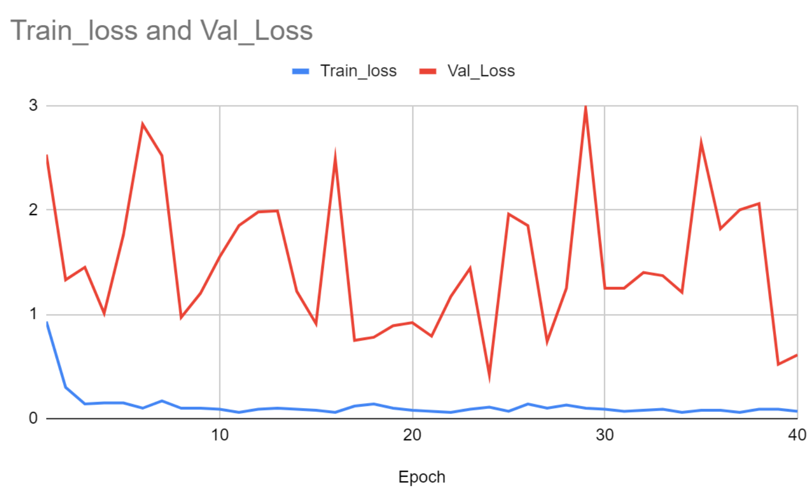


Figure 8 Training and validation loss curve for learning rate 0.001 and batch size 2

Configuration used for fetlock shape was Number of epochs = 40, Batch size = 2, and Learning rate = 0.001. The validation loss curve was plotted (Figure 9)

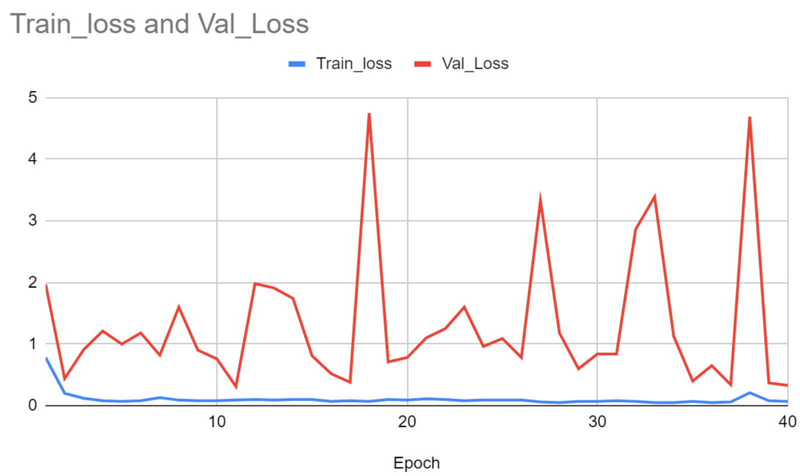


Figure 9 Training and validation loss curve for learning rate 0.001 and batch size 2

Five separate best models were selected for five different hoof aspects [9]. For prediction, the configuration of the model was defined which was consistent with the training configuration. The batch size was set to 1 as inference was run on only one image at a time. Different hoof aspects (models) had slightly different configurations like configuration names and the number of classes. Trained weights were loaded, and Mean Average Precision was calculated for validation and test set to check if the loaded model is correct. The models were tested on the test dataset and the percentage of detection and correct classification were calculated manually for the best models (refer to Appendix K).

Shiny App

The best models for each hoof aspect were downloaded from the NeSI server. The next step involved integrating these models with a Shiny App. The reticulate package in R allows the incorporation of python functions and scripts in R[6]. An ui.R file and server.R file (please refer to the supplementary folder for code) were created to accept the user input image file, and call the functions defined in the python files based on the hoof aspect selected by the user. The image is processed and the result is displayed (Figure 10a, 10b). The downloaded model weights, python files, ui.R, and server.R were kept inside the same folder (refer to Appendix F).

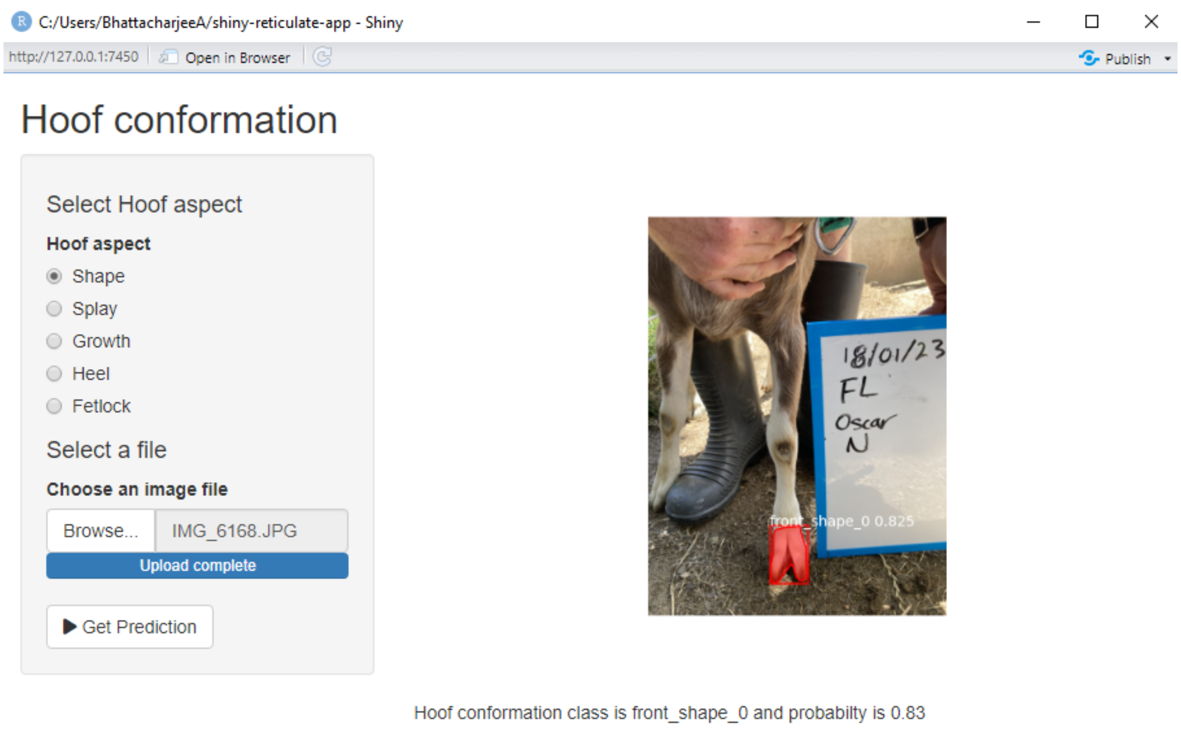


Figure 10a Hoof classified as a front_shape_0 with a probability of 0.83 (refer to Appendix G)

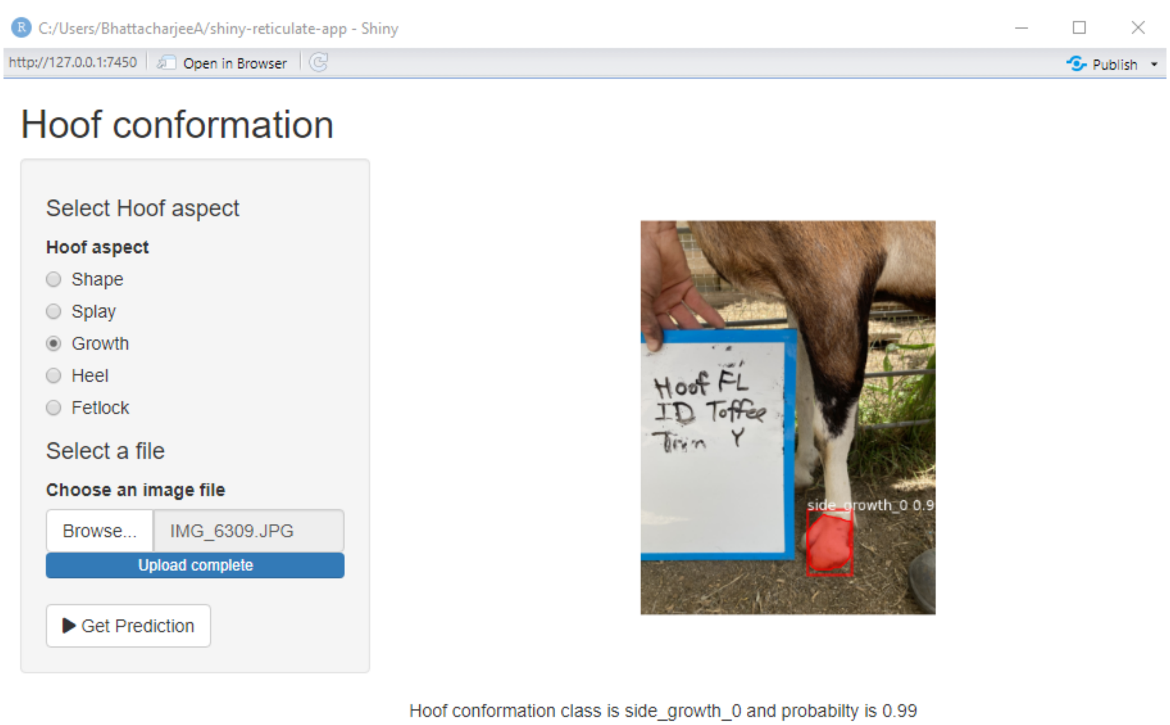


Figure 10b Hoof classified as a side_growth_0 with a probability of 0.99 (refer to Appendix G)

Results

The best model for each hoof aspect was selected where the Mean Average Precision (mAP) [9] was highest (Table 6).

Hoof aspect	mAP validation	mAP test	Detected on test	Accuracy on test
Claw shape	0.4	0.43	51%	85%
Claw splay	0.73	0.72	97%	84%
Toe length	0.71	0.48	78%	100%
Heel shape	0.52	0.46	91%	71%
Fetlock shape	0.91	0.91	87%	100%

Table 6 Summary of model performance

Claw Shape model

The loss curve is fluctuating and the lowest value of validation loss was considered for each model. Mean Average Precision[9] on the test set was used as a criterion to select the best model. The configuration with a learning rate of 0.001 and batch size of 2 (without image augmentation) was found to have the highest mAP on the 23rd epoch (Figure 4a). The best model was found to be mask_rcnn_hoof_shape_cfg_coco_0023 (refer to Appendix F and supplementary folder) with mAP = 0.43 (Table 6). The model can detect and classify hoof claw shapes (Figure 11).

Claw Splay model

The loss curve is fluctuating and the lowest value of validation loss was considered. Mean Average Precision[9] on the test set was used as a criterion to select the best model. The configuration with a learning rate of 0.001 and batch size of 2 was found to have the highest mAP on the 39th epoch (Figure 6). The best model was found to be mask_rcnn_hoof_splay_cfg_coco_0039 (refer to Appendix F) with mAP = 0.72 (Table 6). The model can detect and classify hoof claw splay(Figure 12).

Toe length/Growth model

The loss curve is fluctuating and the lowest value of validation loss was considered. Mean Average Precision [9] on the test set was used as a criterion to select the best model. The configuration with a learning rate of 0.001 and batch size of 2 was found to have the highest mAP on the 24th epoch (Figure 7). The best model was found to be mask_rcnn_hoof_growth_cfg_coco_0024 (refer to Appendix F and supplementary folder) with mAP = 0.48 (Table 6). The model can detect and classify hoof toe length (Figure 13).

Heel Shape model

The loss curve is fluctuating and the lowest value of validation loss was considered. Mean Average Precision [9] on the test set was used as a criterion to select the best model. The configuration with a learning rate of 0.001 and batch size of 2 was found to have the highest mAP on the 40th epoch (Figure 8). The best model was found to be mask_rcnn_hoof_heel_cfg_coco_0040 (refer to Appendix F) with mAP = 0.46 (Table 6). The model can detect and classify hoof heel shapes (Figure 14).

Fetlock Shape model

The loss curve is fluctuating and the lowest value of validation loss was considered. Mean Average Precision [9] on the test set was used as a criterion to select the best model. The configuration with a learning rate of 0.001 and batch size of 2 was found to have the highest mAP on the 11th epoch (Figure 9). The best model was found to be mask_rcnn_hoof_fetlock_cfg_coco_0011 (refer to Appendix F) with mAP = 0.91 (Table 6). The model can detect and classify hoof heel shapes (Figure 15).

Figure 11 Hoof classified as a front_shape_0 with a probability of 0.8 (refer to Appendix G)

Figure 12 Hoof classified as a front_splay_0 with a probability of 0.99 (refer to Appendix G)

Figure 13 Hoof classified as a side_growth_0 with a probability of 0.95 (refer to Appendix G)

Figure 14 Hoof classified as a side_heel_1 with a probability of 0.94 (refer to Appendix G)

Figure 15 Hoof classified as a side_fetlock_0 with a probability of 1 (refer to Appendix G)

Discussion

The result suggests that the models can detect 51% and correctly classify 85% of the claw shape. Detect 97% and correctly classify 84% of the claw splay. Detect 78% and correctly classify 100% of the toe length. Detect 91% and correctly classify 71% of the heel shape, and detect 87% and correctly classify 100% of the fetlock shape (Table 6). The fetlock shape model looks the most efficient, and the claw shape model looks the least efficient. However, these observations can also be because of class imbalance in the training dataset (Table 1) and the test dataset (Table 3). Image augmentation was tried while training the claw shape model (Table 5), but the best value of mAP achieved was lower than that achieved without augmentation. So, the selected set of image augmentation was not effective. More configurations, image annotations, and class weights couldn't be tried due to time constraints and GPU unavailability. GPUs were in high demand, so getting a 2-3 hours window was difficult in the last few days.

Moreover, it was observed that the validation loss curves were fluctuating (Figure 4a, 6, 7, 8, 9). Lowering the learning rate(Figure 4c, 5a) and increasing the batch size (Figure 4b) didn't affect the curves. The noisy movement of the learning curve could be because of an unrepresentative validation dataset. The validation dataset is imbalanced and has few examples (Table 2), so it couldn't have been able to provide sufficient information to evaluate the model [10].

Models can be improved by changing the training strategy, increasing the amount of training data, and further tuning the hyperparameters. Another factor that can improve the result is the placement of the camera while taking the hoof photographs. If the camera was not placed squarely in front of the hoof, the angle of the photograph may make it more difficult to accurately score [1]. Surrounding conditions such as, whether the goat is standing on concrete or ground, shadows falling on the hoof also affected the result. To reduce the number of misclassifications, it is important to consider a balanced dataset or implement class weights while training the model.

The Shiny App was found to be a reliable way to display summary outputs of the goat hoof image. The user interface of the Shiny App needs to be improved, but couldn't be taken up because of time constraints. Once published, it can give the farmer an overview of their animal health.

Conclusion

We successfully implemented Shiny App and Mask R-CNN to detect the goat hoof in the uploaded image and classify the hoof based on its deformity. The model was trained on NeSI infrastructure, downloaded locally, and then integrated with the Shiny App. The process of training and plotting the loss curve was time-consuming and required GPU availability. Training with more images and balanced class weights should be considered to achieve higher Mean Average Precision. Further work is required to improve the model and to publish the Shiny App.

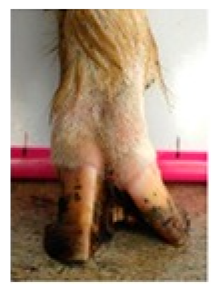
References

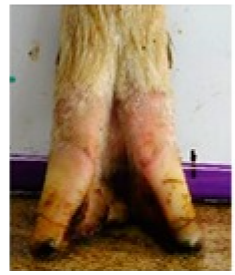
1. Deeming, Laura E et al. "The Development of a Hoof Conformation Assessment for Use in Dairy Goats." *Animals : an open access journal from MDPI* vol. 9,11 973. 14 Nov. 2019, doi:10.3390/ani9110973
2. AgResearch, Linkedin, https://nz.linkedin.com/company/agresearch, Accessed 19 Dec. 2022
3. Musyarofah et al 2020 IOP Conf. Ser.: Earth Environ. Sci. 500 012090, doi:10.1088/1755-1315/500/1/012090
4. Waleed Abdulla, Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow, 2017, GitHub repository, https://github.com/ahmedfgad/Mask-RCNN-TF2, Accessed 30 Nov. 2022
5. COCO dataset, https://cocodataset.org/#home, Accessed 9 Jan. 2023
6. Rani Powers, Tutorial: using Shiny + reticulate to create apps with R and Python 3, 2019, GitHub repository, https://github.com/ranikay/shiny-reticulate-app, Accessed 9 Jan. 2023
7. Abhishek Dutta and Andrew Zisserman. 2019. The VIA Annotation Software for Images, Audio and Video. In Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3343031.3350535.
8. Waleed Abdulla, Splash of Color: Instance Segmentation with Mask R-CNN and TensorFlow, March 20, 2018, Medium, https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46, Accessed 9 Jan. 2023

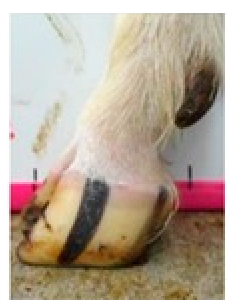
9. Ahmed Gad, Evaluating Object Detection Models Using Mean Average Precision, March 3, 2021, Kdnuggets, https://www.kdnuggets.com/2021/03/evaluating-object-detection-models-using-mean-average-precision.html, Accessed 16 Jan. 2023
10. Jason Brownlee, How to use Learning Curves to Diagnose Machine Learning Model Performance, February 27, 2019, Machine Learning Mastery, https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/, Accessed 16 Jan. 2023
11. He, Kaiming et al. “Mask R-CNN.” 2017 IEEE International Conference on Computer Vision (ICCV) (2017): 2980-2988.
12. Sreenivas Bhattiprolu, python_for_microscopists/286-Object detection using mask RCNN - end to end, 2022, GitHub repository, https://github.com/bnsreenu/python_for_microscopists/tree/master/286-Object%20detection%20using%20mask%20RCNN%20-%20end%20to%20end, Accessed 5 Dec. 2022

Appendices

A. Figures displaying different hoof conformation in goats. Source : Deeming, Laura E et al. "The Development of a Hoof Conformation Assessment for Use in Dairy Goats." *Animals : an open access journal from MDPI* vol. 9,11 973. 14 Nov. 2019, doi:10.3390/ani9110973

Hoof aspect	Description	Subjective score	Photo example
Claw shape	Both claws are straight	0	
Claw shape	One claw is bent/twisted	1	
Claw shape	Both claws are bent/twisted	2	

Claw splay	Claws are not splayed	0	
Claw splay	Claws are moderately splayed	1	
Claw splay	Claws are severely splayed	2	
Toe length	Toe is not overgrown	0	
Toe length	Toe is moderately overgrown	1	

Toe length	Toe is severely overgrown	2	
Heel shape	Heel is upright	0	
Heel shape	Heel is moderately dipped	1	
Heel shape	Heel is severely dipped	2	
Fetlock shape	Fetlock is upright and straight	0	

Fetlock shape	Fetlock is dipped towards the ground	1	

B. H5 file documentation. https://docs.fileformat.com/misc/h5/#:~:text=H5%20File%20Viewer-,What%20is%20an%20H5%20file%3F,for%20quick%20retrieval%20and%20analysis.

C. Shiny App https://shiny.rstudio.com/

D. Reticulate package documentation https://rstudio.github.io/reticulate/

E. Directory structure for NeSI system

```
—Root folder (Mask_RCNN-TF2)
    |
    |-------hoof_claw_shape.ipynb
    |-------hoof_claw_splay.ipynb
    |-------hoof_fetlock_shape.ipynb
    |-------hoof_heel_shape.ipynb
    |-------hoof_toe_length.ipynb
    |-------hoof_shape
    |       |--------train
    |       |        |--------labels
    |       |        |        |--------train JSON file
    |       |        |--------train_images.JPG
    |       |--------val
    |       |        |--------labels
    |       |        |        |--------validation JSON file
```

```
|          |          |--------validation_images.JPG
|          |--------test
|                     |--------labels
|                     |          |--------test JSON file
|                     |--------test_images.JPG
|-------hoof_splay
|          |--------train
|          |          |--------labels
|          |          |          |--------train JSON file
|          |          |--------train_images.JPG
|          |--------val
|          |          |--------labels
|          |          |          |--------validation JSON file
|          |          |--------validation_images.JPG
|          |--------test
|                     |--------labels
|                     |          |--------test JSON file
|                     |--------test_images.JPG
|-------hoof_side
|          |--------train
|          |          |--------labels
|          |          |          |--------train JSON file
|          |          |--------train_images.JPG
|          |--------val
|          |          |--------labels
|          |          |          |--------validation JSON file
|          |          |--------validation_images.JPG
|          |--------test
|                     |--------labels
|                     |          |--------test JSON file
|                     |--------test_images.JPG
```

|-------hoof_test.py

|-------hoof_test.sl

|-------mask_rcnn_coco.h5

|-------logs

|--------model_1

|--------model_2

|--------model_3

:

|--------model_n

F. The directory structure in the local system (Shiny app and Python integration)

—Root folder (Shiny_Reticulate_App)

|

|-------best_model_claw_shape.h5

|-------best_model_claw_splay.h5

|-------best_model_fetlock_shape.h5

|-------best_model_heel_shape.h5

|-------best_model_toe_length.h5

|-------python_functions_rcnn_fetlock.py

|-------python_functions_rcnn_growth.py

|-------python_functions_rcnn_heel.py

|-------python_functions_rcnn_shape.py

|-------python_functions_rcnn_splay.py

|-------server.R

|-------ui.R

G. Class description for annotation and labeling

Hoof aspect	Description	Subjective score	Id on VIA	Description on VIA

Claw shape	Both claws are straight	0	1	front_shape_0
Claw shape	One claw is bent/twisted	1	2	front_shape_1
Claw shape	Both claws are bent/twisted	2	3	front_shape_2
Claw splay	Claws are not splayed	0	1	front_splay_0
Claw splay	Claws are moderately splayed	1	2	front_splay_1
Claw splay	Claws are severely splayed	2	3	front_splay_2
Toe length	Toe is not overgrown	0	1	side_growth_0
Toe length	Toe is moderately overgrown	1	2	side_growth_1
Toe length	Toe is severely overgrown	2	3	side_growth_2
Heel shape	Heel is upright	0	1	side_heel_0
Heel shape	Heel is moderately dipped	1	2	side_heel_1
Heel shape	Heel is severely dipped	2	3	side_heel_2
Fetlock shape	Fetlock is upright and straight	0	1	side_fetlock_0
Fetlock shape	Fetlock is dipped towards the ground	1	2	side_fetlock_1

Id 0 is for background in VIA

H. JSON file example

{
  "info": {
    "year": 2023,
    "version": "1.0",
    "description": "VIA project exported to COCO format using VGG Image Annotator (http://www.robots.ox.ac.uk/~vgg/software/via/)",
    "contributor": "",
    "url": "http://www.robots.ox.ac.uk/~vgg/software/via/",
    "date_created": "Tue Jan 31 2023 16:23:08 GMT+1300 (New Zealand Daylight Time)"
  },
  "images": [
    {
      "id": 1,
      "width": 3024,
      "height": 4032,
      "file_name": "IMG_1841.jpeg",
      "license": 0,
      "date_captured": ""
    }
  ],
  "annotations": [
    {
      "segmentation": [
        [
          1639,
          2687,
          1805,

2858,
1764,
2909,
1764,
2950,
1764,
3029,
1727,
3093,
1653,
3121,
1505,
3121,
1482,
3098,
1459,
3029,
1482,
2969,
1473,
2941,
1487,
2885,
1491,
2844,
1551,
2798,
1574,
2747
]
],

"area": 150164,
"bbox": [
1459,
2687,
346,
434
],
"iscrowd": 0,
"id": 1,
"image_id": 1,
"category_id": 1
}
],
"licenses": [
{
"id": 0,
"name": "Unknown License",
"url": ""
}
],
"categories": [
{
"supercategory": "Hoof",
"id": 1,
"name": "side_heel_0"
},
{
"supercategory": "Hoof",
"id": 2,
"name": "side_heel_1"
},

```
      {
        "supercategory": "Hoof",
        "id": 3,
        "name": "side_heel_2"
      }
    ]
  }
```


I. What is NeSI - https://www.nesi.org.nz/

J. Issues with the NeSI infrastructure
   a. Tensorboard (GUI) is currently not supported on the cluster, so evaluating the loss curve is a challenge. One way is to plot it manually, but it is time-consuming.
   b. P100 are in high demand so jupyter sessions running on P100 fail to start most of the time. Running the code as a Slurm batch job is an option but training and validation loss are not displayed.
   c. Currently, NeSI doesn't support outward-facing app. So, the model needs to be downloaded every time and integrated with an outward-facing app on AgResearch systems. This increases the development time.
   d. The storage allocated for the project was on the lower side. After every couple of training, manual deletion of previous files was required to make up space for new training.

K. Table displaying the performance of models on test data

	Test image count	**Detected count**	**Detected %**	**Correct Classification count**	**Correct Classification%**
Claw shape	39	20	51.28%	17	85.00%
Claw splay	39	38	97.44%	32	84.21%
Toe length	23	18	78.26%	18	100.00%

Heel shape	23	21	91.30%	15	71.43%
Fetlock shape	23	20	86.96%	20	100.00%
Total	147	117		102	
%		79.6%		87.2%	