An Analysis of Bull Elephant Footprints Using Footprint Identification Technology

Client: Dr. Marguerite Madden, Department of Geography, University of Georgia

Ted Woodsides, Taylor Last, Kat Smith

Department of Statistics
University of Georgia
May 10, 2021

1 Background

Protecting the dwindling elephant population has long been a focus of wildlife conservation groups in Africa, but few have been as successful as "Elephants for Africa". This group seeks to preserve elephant life by studying the human-elephant conflict and create solutions to this conflict that don't involve lethal action. Our client is Dr. Marguerite Madden and she works with this group by using her background in geospatial research to better understand the ecological and social requirements of elephants, specifically the African bull elephant. She and her team work primarily in areas of Zimbabwe where humans and elephants share the same land. Sharing land is common in Africa where 70% of the African elephant population lives outside of protected areas such as national parks and game reserves.

Much of the conflict previously mentioned stems from the aggression and size of the bull elephants. The bulls can weigh up to 14,000 pounds and are known to raid crop fields of farmers in Zimbabwe. They also cause destruction around villages, contaminate water treatment centers, and attack and kill humans. In the past, the villagers have been known to use lethal action on the elephants to solve these problems, but the sharp decline in elephant population over the last 90 years makes it pertinent that the villagers stop killing the elephants. In 1930 there were an estimated 10 million elephants roaming the world. Today, there is around 500,000 remaining (95% population decrease over 90 years). Dr. Madden and her team have made it their mission to reduce this conflict that is causing the loss of elephant life.

They began their mission by going to Zimbabwe and tagging 13 bull elephants. By tagging the elephants they could understand the movement patterns of these animals and monitor the frequency and type of destruction they were causing. However, the team found these elephants elusive and hard to track. They wanted to be able to pinpoint which specific elephant is causing an issue, and if certain conflict solutions are working. To do this, the team needs a way to identify the elephant's footprints. Like human fingerprints, the elephant footprints have unique ridges and grooves that can be used for identification. The footprints also have unique sizes, specifically circumferences, which is what we will focus on in our analysis.

Figure 1: Elephant Footprint

Figure 2: Elephant Foot

2 Description of the Data

In 2019, Dr. Madden and her team traveled to Zimbabwe to create an image repository of elephant footprints. During this trip, the team collected around 300 pictures of footprints. These 300 pictures of the data served as our preliminary dataset for our analysis. To get a final dataset, we had to inspect the pictures to ensure they were suitable for analysis. As described in later sections, the image processing software we used required a strict protocol for image selection.

We were able to find 20 pictures that fit the specifications our software required. We then took these 20 images and added randomly distributed noise at two levels (33% strength and 66% strength) to each picture using an online grayscale noise generator. Thus our final data set had 60 images consisting of the original picture of the footprint and two versions with added noise. Our goal was to develop software that will correctly cluster the original picture with the two noisy versions. If our software is able to do this, we believe it will be able to classify an individual elephant footprint into an existing cluster or declare that this elephants footprint has not yet been recorded-thus creating a new cluster.

3 Potential Problems with the Dataset

The first issue with the data was determining the size of each footprint. Since all of the pictures were taken from different heights, we had to scale each image to obtain a consistent estimate on the size of each footprint. To scale each image, we used Dr. Madden's foot as a reference object to have a ratio of human foot to elephant foot. We obtained a measurement of 4.5 inches across the sole of the show below the bottom lace.

The next major issues is that we are not certain if more than one footprint from the same elephant is included in our data set. Because many of the images were taken in the same territory, it is likely that some of the images are the same footprint. Thus, if the software creates clusters larger than 3 we will not be able to verify the accuracy other than visual affirmation.

The last issue with the data set is the size. Machine learning models like we need are "data hungry" which means performance is often dependent on sample size. We attempted to simulate the data by adding varying levels of noise to the original 20 pictures.

4 Exploratory Data Analysis

For the EDA, we focused on the length and width of five elephant footprints Dr. Madden visually suggested where from different elephants. We decided to use Adobe Acrobat Reader DC to create data points needed for our EDA. Adobe Acrobat Reader DC has a measuring tool that can be used to measure the footprint as well as our reference object, Dr. Maddens shoe.

After talking to Dr. Madden, we found her shoe size and measured the width of the shoe at the widest point in centimeters in order to use the shoe width as a reference to measure the size of the different elephants' footprints. We then used the measuring tool to measure the width of the shoe in each picture since each picture was taken from a different height. Then we also measured the widest and longest part of each of the elephants footprints in the same pictures. We then divided the width of the shoe in the picture (measured in centimeters) by the actual width of the shoe (which is 11.5 centimeters). This result gave us a different ratio for each image. We used the respective ratios to multiply the length and width of the footprints from the images to calculate the actual length and width of the elephants footprints.

The ratio used in each image can be seen in the last column of the 5 elephants data tables. These tables are in the Appendix, along with the measurements from the picture and the calculations of the actual length and width of the footprints. The thickness of the lines showing the measurements on the picture are thick to better show the lengths and all the measurements are from the bottom of the lines hanging off the connecting lines. Each measurement was taken from the widest part of the as well as the longest part of the footprint.

Table 1: Length and Width of Each Elephant's Footprint.

Footprint	Elephant 1	Elephant 2	Elephant 3	Elephant 4	Elephant 5
Length (cm)	39.04	38.35	39.06	39.18	37.15
Width (cm)	34.31	25.97	23.21	23.28	27.72

In Table 1, you can see the final data table with the calculated values for the length and width of the 5 elephants footprints from the chosen 5 representative pictures that are shown in the Appendix. Furthermore, in Table 2 we have calculated more statistics from the 5 elephants in our testing data set including the mean and standard deviation for both the width and length of their footprints. From Table 2, when looking at the standard deviations we can see that there is little variability in the elephant footprint lengths, while there is much more variability in the width. When looking at the means in Table 2, we can see that the length of the elephant footprints are larger than the width for these elephants.

Table 2: Summary Statistics of Identified Footprints

Statistic	Value (cm)
Mean of Footprint Lengths	38.56
Standard Deviation of Footprint Lengths	0.85
Mean of Footprint Width	26.90
Standard Deviation of Footprint Width	4.56

5 Footprint Identification Technique (FIT) Software

To accomplish our goal of clustering the footprints we needed image processing software with the ability to perform cluster analysis. Some research led us to WildTrack, a conservation company focused on using FIT for non-invasive wildlife monitoring. They have had success with FIT with many animals ranging from cheetahs to pandas to polar bears. They advised us that to properly train an image recognition algorithm, we need around 20 footprints from 20 different individual elephants. Additionally, these 20 footprints need to all be from the same foot, specifically the left hind foot. This will allow us to train the algorithm to pick up sex and individual ID. We communicated our data shortcoming with WildTrack, but ultimately decided the software could still be useful. WildTrack's software uses add-ins they have created in JMP to analyze the footprints. These scripts are written in JSL (JMP Scripting Language) which will be our primary language for this analysis. We will use and edit these scripts to outline the footprint and then divide the outlined portion into 44 segments. These segments will go across the width of the footprint, or from the edge to a bisecting line, depending on the number of segments desired. The area of these segments will then be analyzed for similarity across the 60 prints in our data set.

6 Detailed Methods for WildTrack FIT Software

Before running cluster analysis on the dataset, we decided to investigate if there was any correlation between the 44 variables we had for each footprint. Our hypothesis was that data points at different sections of the footprint would be correlated. A correlation heat map of every variable is shown in Figure 3.

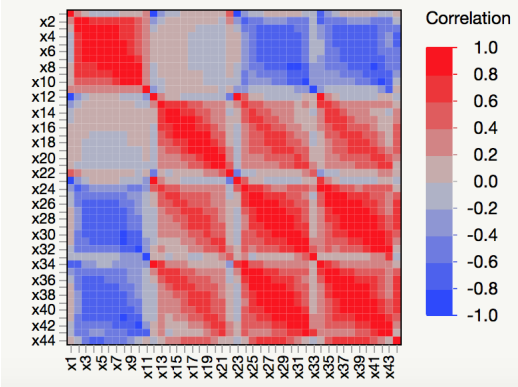

Figure 3: Correlation Heatmap

Figure 3 tells us our hypothesis was true and mutlicollinearity is an issue in our dataset. The x followed by a number on the axes represents the area of each of the 44 different segments. The numbers start at the top of the footprint (x1) and go chronologically to the bottom (x44). We know our hypothesis is true because the numbers that are close together (i.e close together on the foot) are highly correlated. To solve this issue we used Principal Component Analysis (PCA).

6.1 Principal Component Analysis

Principal component analysis is a dimension reduction method that computes the eigenvalues and eigenvectors of the variance-covariance matrix of a data set. Then, we use the eigenvectors and eigenvalues as the components of PCA to determine how much variance is contained in each component. Once we have our desired number of components, we project our original data onto our principal components to construct our reduced data set. The components created by PCA are different linear combinations of the 44 variables in our dataset. In our case, the heat map tell us the components represent the different sections of the foot. To decide how many components to use, we created the scree plot shown in Figure 4.

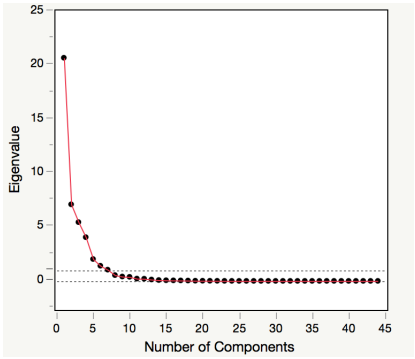

Figure 4: Scree Plot

The scree plot tells us there is minimal variance being explained after the 10th component. Thus, we decided to use 10 components in our analysis. The 10 components replaced the 44 variables we started with and gave us a final dataset to run our cluster analysis on.

7 Results of WildTrack Software

We elected to run Ward's agglomerative hierachial clustering on our dataset and use the corresponding dendrogram to visualize our results. This clustering method works by first treating each observation as singleton clusters. Then, it computes the Ward's distance of each pair and successively merge the most similar clusters. The merging is then repeated until the final optimal clusters are formed.

Because of the uncertainty of our labeling, we were focused on clustering together every image version (original picture, 33% noised, and 66% noised) for each of the 20 footprints. In Figure 5 one can see labels of the individual footprints on the left as well as instances where all three image versions were clustered together colored in yellow, and instances of two image versions clustered colored in orange.

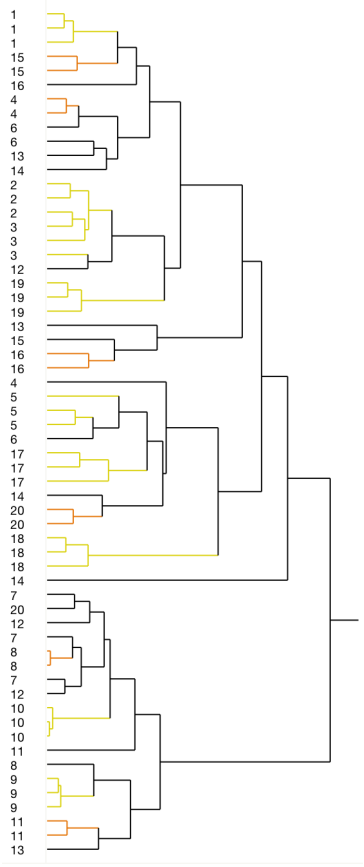

Figure 5: WildTrack Dendrogram

The dendrogram tell us that 27 out of 60 footprints were perfectly clustered and another 12 out of 60 were clustered with at least one noised match. These results are promising, but considering we used noised images we wanted more footprints to be correctly clustered.

We believe these mediocre results are because of the quality of the images and the quality of the scale. The image collection process for WildTrack's software is very strict, and we were not able to obtain images that followed their protocol. Specifically, we had 3 pictures per elephant instead of 15, did not have a metric scale in the picture, did not straddle the footprint when the picture was taken, and did not ensure the footprint completely fit the frame.

In an effort to increase clustering accuracy, we will try a new method called a Deep Convolutional Neural Network and compare the results to WildTrack. This method will be described in the next section.

8 Deep Convolutional Neural Network (DCNN)

The deep learning model we chose to implement is a convolutional neural network, specifically the VGG16 (Visual Geometry Group - 16) model. Convolutional neural networks are often used in image recognition to cluster images based on similar patterns. The model requires the input image to be 224x224 dimensions (224 pixels long, 224 pixels wide). When the model runs, it loops through the pixels in groups of 3x3 matrices and takes the dot product of the 3x3 matrix and another 3x3 matrix that is specified as a hyperparameter of the VGG16 model. The model runs this loop 16 times, and each loop is called a convolutional layer. The purpose of each layer is to add different weights to pixels to determine edges and patterns in each image. After weights are designated at each layer, the model pools together similar features and predicts which group to assign an image to. The VGG16 is the state of the art model for image classification. It is computationally expensive and relatively slow, but it is the most accurate model right now for image recognition.

9 Detailed Methods for DCNN

In practice, the VGG16 model is used on very large, labeled data sets. The data set we analyzed contained 60 images. Also, the images were manually labeled, so we could test the effectiveness of the model. For the model to be suitable, we would need a data set that is large and has labeled images, so we could split the data into training and testing sets to assess its performance in a traditional manner. Since the data set we were provided didn't contain labels, we decided to use the DCNN model to extract the model features into a list, instead of predicting each cluster. The features we obtained from the model give information about each image, and they are arranged into a data frame of 4096 columns and 60 rows. Each column is a feature of each observation (row). Each row represents one image. Using the features from the model, we conducted principal component analysis to reduce the number of features. With the reduced dimensions, we were able to use K-means clustering to group the elephants based on similar features. K-means was used because we couldn't use supervised learning due to lack of labels.

9.1 Principal Component Analysis

Each image has 4096 features once it is processed by the VGG16 model. To handle the large number of dimensions, we needed to reduce our data set using principal component analysis. Each principal component is a linear combination of the 4096 features the data has.

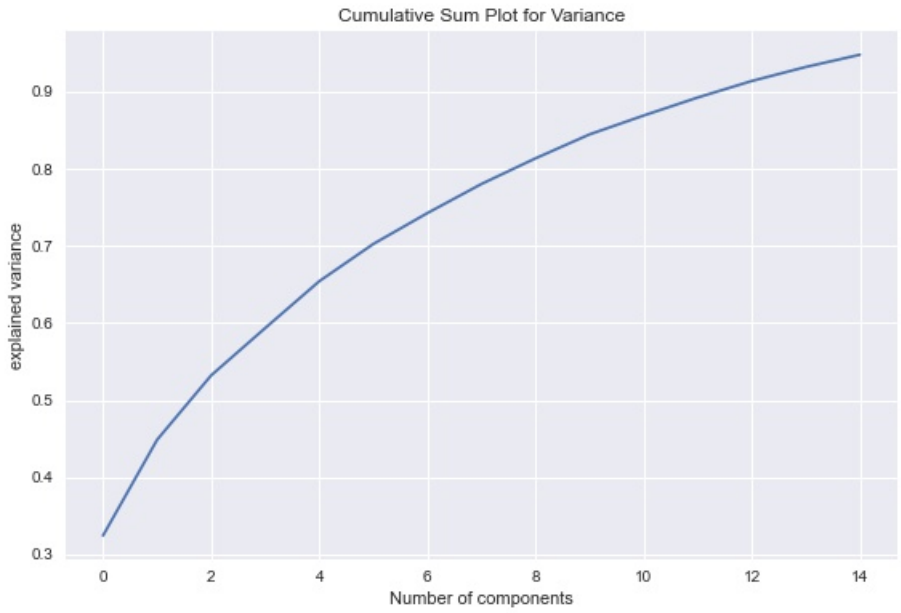

Figure 6: PCA Cumulative Sum

We specified the number of components as 15 since the first 15 components contained 94.7% of the variance from the original data. In the PCA Cumulative Sum plot, we see that we would need 15 components if we would like to maximize variance without taking too many components. The shape of our new data set after PCA was 60 rows by 15 columns.

9.2 K-Means

K-means clustering is an algorithm that is very popular among unsupervised learning techniques. The algorithm works by choosing k random points in the data set and choosing the observations around it with the smallest Euclidean distance. Once a point is added to a cluster, the mean for each cluster is recalculated. The algorithm outputs a result when the mean of each cluster doesn't change between iterations, which indicates the algorithm has converged on the mean of each cluster.

We chose to use K-means for this data set because the intuition behind the model is that it can cluster images without labels. We manually added labels to each file as a way to verify model performance. Due to the accuracy of the VGG16 model features, K-means was a perfect choice for clustering our elephants into a predefined set of 20 clusters. The K-means algorithm constructs the clusters based on the mean of the PCA-transformed data set.

10 Results for DCNN

To test the model, we used our data set of 20 images with added noise at a 33% level and 66% level, and we converted all of the images from RGB to Grayscale to standardize the pixels in the image and take out any variation due to color. We read the 60 images into our script and set the number of clusters (k) equal to 20. We set $k = 20$ because we wanted to ensure the VGG16 and KMeans models could correctly identify the same image three times.

Cluster	0	1	2
0	Elephant 1 copy 2.JPG	Elephant 1 copy.JPG	Elephant 1.JPG
1	Elephant 11 copy 2.JPG	Elephant 11.JPG	Elephant 11 copy.JPG
2	Elephant 17 copy.JPG	Elephant 17.JPG	Elephant 17 copy 2.JPG
3	Elephant 9 copy 2.JPG	Elephant 9.JPG	Elephant 9 copy.JPG
4	Elephant 15.JPG	Elephant 15 copy.JPG	Elephant 15 copy 2.JPG
5	Elephant 5.JPG	Elephant 5 copy 2.JPG	Elephant 5 copy.JPG
6	Elephant 13 copy 2.JPG	Elephant 13.JPG	Elephant 13 copy.JPG
7	Elephant 8copy 2.JPG	Elephant 8.JPG	Elephant 8 copy.JPG
8	Elephant 2.JPG	Elephant 2 copy 2.JPG	Elephant 2 copy.JPG
9	Elephant 10.JPG	Elephant 10 copy.JPG	Elephant 10 copy 2.JPG
10	Elephant 6 copy.JPG	Elephant 6 copy 2.JPG	Elephant 6.JPG
11	Elephant 20.JPG	Elephant 20 copy 2.JPG	Elephant 20 copy.JPG
12	Elephant 14.JPG	Elephant 14 copy 2.JPG	Elephant 14 copy.JPG
13	Elephant 7 copy.JPG	Elephant 7.JPG	Elephant 7 copy 2.JPG
14	Elephant 4 copy 2.JPG	Elephant 4.JPG	Elephant 4copy.JPG
15	Elephant 16 copy.JPG	Elephant 16 copy 2.JPG	Elephant 16.JPG
16	Elephant 19 copy 2.JPG	Elephant 19.JPG	Elephant 19 copy.JPG
17	Elephant 12.JPG	Elephant 12 copy 2.JPG	Elephant 12 copy.JPG
18	Elephant 3.JPG	Elephant 3 copy 2.JPG	Elephant 3 copy.JPG
19	Elephant 18.JPG	Elephant 18 copy 2.JPG	Elephant 18 copy.JPG

Figure 7: Clustered Images

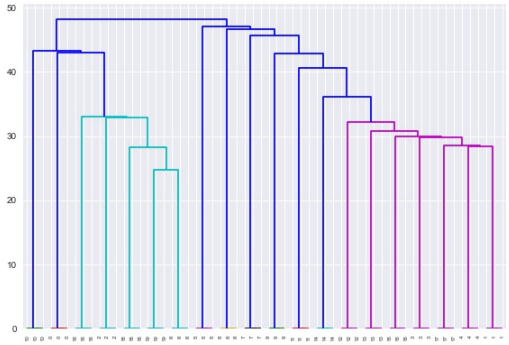

Figure 8: Dendrogram Cluster

The model correctly classified 3 images into 20 different clusters with each cluster containing the original picture and the two duplicates with noise (Figure 7). Using the minimum Euclidean distance with single linkage, our dendrogram illustrates how similar each cluster of elephants are (Figure 8). The first layer of the dendrogram correctly classifies all elephants to their appropriate cluster. The succeeding layers display which clusters have similar features. The purpose of Figure 8 is not to show which cluster an elephant belongs to (numbers on the x axis) but is meant to show that the clustering is well behaved and accurate.

In order to determine the correct amount of clusters (k), we iterated over all possible values of k, starting at 3 and ending at 30, to ensure we are choosing the correct number of clusters. We limited the clusters to 30 instead of 60 due to the computational cost of running 60 iterations. For each iteration, we obtained the sum of squared error(SSE) from a predefined function in Sci-Kit Learn.

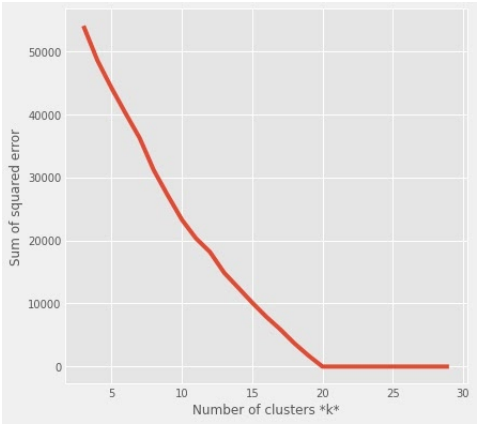

Figure 9: SSE vs. K

Looking at the plot in Figure 9 of SSE vs. k, we chose 20 clusters since it was the lowest k that minimized SSE.

The initial results obtained from the VGG16 model are very promising, but it is only a test on data that simulated to obtain expected results. We ran the test knowing that the optimal k value would be 20 since we have 20 different images duplicated two times with noise. To scale the model, we tested it on a list of 22 identified elephants to tune the hyperparameters. If Dr. Madden, our client, wishes to use VGG16 model for elephant identification, she can upload any specified data set and it will cluster elephants together with high level of accuracy.

11 Recommendations

11.1 Sample Size

The two models perform differently for varying sample size, and there is a clear distinction on when to use one model over the other. If the data set is relatively small, WildTrack's software is a better solution because it is simpler and more intuitive when dealing with small data sets. Since you have to trace each image, a small sample size is ideal. Also, the DCNN is a model that was built for a lot of data. If we had a large enough sample size, we would recommend using the DCNN because it is scalable and easy to implement. A data set with over 100 images would take a long time to trace in WildTrack, but with the DCNN model, we could simply read in the file and receive output to perform K-means clustering.

11.2 Accuracy

As seen in the results for the two methods both have success correctly clustering the footprints, but DCNN easily beat the hierarchical clustering used with WildTrack's software. DCNN perfectly clustered 20 of 20 footprints while WildTrack perfectly clustered only 9 of 20. Although these results are decisive, it is worth noting that the WildTrack models success would be improved by using pictures that directly followed the protocol set forth by the software creators.

11.3 Ease of Use

A big difference in the two software is the users ability to understand what exactly the software is doing. The Wildtrack software is very intuitive. One traces the outline of the footprint and the algorithm slices this outline into data points that are compared for similarity in size. On the contrary, the DCNN is a "black box" deep learning algorithm that is very difficult to understand. If a user is willing to sacrifice some time and possible accuracy to understand what exactly they are doing in their analysis, the WildTrack software would be a better option for them. However, if one is comfortable with nuances that come with Deep Learning and want to maximize accuracy while minimizing time, the DCNN is a better option.

12 Appendix

12.1 Workflow for WildTrack FIT Software

The method used for our analysis is rooted in taking professional pictures and understanding how to use WildTrack's FIT African Elephant add-in. There is a two part process to using the add-in, extracting the footprint from the image and carrying out the cluster analysis. Both will be explained step by step in this section, as well as how to properly take pictures of the footprints.

Part 1: Taking Pictures of the Footprints:

1. Find a trail of at least 15 prints from the same elephant with good definition.
2. Highlight the position of the left hind foot by drawing a circle around each one in the trail.
3. Place a metric sale about 1 cm below and to the left of the footprint.
4. Straddle the print and point the camera lens directly overhead the footprint.
5. Ensure that the footprint, and ruler completely fill the frame.
6. Repeat steps 3-6 for every print in the trail.

Part 2: Extracting the footprint from an image:

1. Open JMP, choose the FIT African Elephant add-in, and select the Image Feature Extraction option from the main menu.
2. Launch Application.
3. Drag and Drop the first footprint image into the feature extraction window.
4. Click the 'resize' button to ensure the footprint image is inside the graphics window.
5. Select 'Drawing Mode' under Drawing Palette and outline the circumference of the footprint.
6. Select 'Top/Bottom markers' under reference markers and manually place a marker at the highest point and lowest point of the outline of the footprint. Use the image in the bottom left of the screen as a guide.
7. Select 'Ruler Markers' under reference markers and place a marker on 10 cm and 20 cm in the ruler in the picture. If using a smaller ruler, the ruler span can be changed to represent the difference between the ruler markers (i.e if a 10 cm ruler ruler span can be changed to 5 cm thus reference markers need to be at 5 cm and 10 cm).
8. Click 'Append row' to send 84 scripted variables (distances, areas) to a row in a database.
9. Repeat steps 3-8 for each picture in the data set.
10. Divide each animal's number of footprints in half to create two data sets per animal. This will serve as the different trails and help us with the cluster analysis to follow.

Part 3: Pairwise Data Analysis:

1. Open JMP, choose the FIT African Elephant add-in, and select the Pairwise Data Analysis option from the main menu.
2. Select Elephant as 'input x, model category' and trails as 'input trails'. The y columns (footprint measurements) are automatically populated.
3. Select 'Run'. A data table will appear showing the pairwise comparison of trails.
4. Select the 'clusters' button at the base of the table. The first shows the distances between any two trails. The second is a 'cluster dendrogram'.
5. Test the accuracy of the classification by verifying that trails from the same elephant were clustered together in the dendrogram.

12.2 Workflow for DCNN

Deep Convolutional Neural Network: This method involves using Python and the VGG16 model from Keras to manipulate images and cluster them together based on a predefined set of clusters (k).

1. Convert all of the images to grayscale, so the model performs better.
2. Import the required packages (e.g. Keras, Sklearn, Pandas, Numpy, etc.).
3. Designate a path on your system where the images are stored.
4. Loop through the file designated in step 2 and extract the file names using os.scandir().
5. Import the model and specify model hyperparameters.
6. Define the function that reshapes the image and uses the model to predict the feature vector for each image, and store the vectors for each image.
7. Reshape the features to provide the correct amount of dimensions to the KMeans model (amount of image, 4096).
8. Use Principal Component Analysis to reduce the amount of dimensions in the features vectors. (We chose 20 Principal Components)
9. Fit the principal components to the KMeans algorithm.
10. View each cluster to assess how the model performed and which pictures are in each cluster.

12.3 Example Table from EDA

Table 3: Elephant 1 Measurements

Label	Type	Picture Value	Unit	Actual [x(6.216)]
Shoe Width	Distance	1.85	cm	11.5
Length	Distance	6.28	cm	39.036
Width	Distance	5.52	cm	34.312

Table 4: Elephant 2 Measurements

Label	Type	Picture Value	Unit	Actual [x(5.928)]
Shoe Width	Distance	1.94	cm	11.5
Length	Distance	6.47	cm	38.354
Width	Distance	4.38	cm	25.965

Table 5: Elephant 3 Measurements

Label	Type	Picture Value	Unit	Actual [x(6.805)]
Shoe Width	Distance	1.69	cm	11.5
Length	Distance	5.74	cm	39.061
Width	Distance	3.41	cm	23.205

Table 6: Elephant 4 Measurements

Label	Type	Picture Value	Unit	Actual [x(6.886)]
Shoe Width	Distance	1.67	cm	11.5
Length	Distance	5.69	cm	39.181
Width	Distance	3.38	cm	23.275

Table 7: Elephant 5 Measurements

Label	Type	Picture Value	Unit	Actual [x(6.461)]
Shoe Width	Distance	1.78	cm	11.5
Length	Distance	5.75	cm	37.151
Width	Distance	4.29	cm	27.718