Bee Classification

Cat McQueen*, Amir Asdaghpour†
Department of Electrical and Computer Engineering, University of Arizona
Tucson AZ, USA
Email: *catmcqueen@email.arizona.edu, †asdaghpour@email.arizona.edu

***Abstract*—The Department of Agriculture is performing studies on the location and migration of bee species across the country. This is to study the affects of weather and climate change on bee populations, as well as to monitor bee populations for pollination purposes. Currently the data are being identified and labeled individually. As this requires expertise and significant time, this process is slow and arduous. In order to expedite and improve this process, a ResNet based classifier is presented. ResNet is a successful Deep Neural Net (DNN) that is commonly used for image classification problems. Using a ResNet50 model pre-trained on the ImageNet dataset, classification labels are learned effectively to identify bee Species and Genus classes.**

***Index Terms*—Classifier, Deep Neural Net, ResNet, Bee Identification**

I. Introduction

The Department of Agriculture is studying bee location and migration due to climate change and variable weather patterns. The current process is to collect specimens and hand label images to identify bee species location and patterns. As this style of classification requires expertise and is extremely time intensive, it does not scale appropriately to do rapid or large-scale bee identification. This process would be an ideal problem to solve with a neural net classifier, to reduce time to identification and expertise needed. To test this concept, a classifier of six unique bees is presented as a proof of concept. The species of bees are Megachile Campanulae, Melissodes Comptoides, Melissodes Communis, Megachile Petulans, Anthophora Abrupta, and Habropoda Laboriosa. By classifying by both Genus and Species, labels can be verified and data can be grouped by Genus or Species level classifications.

Using a ResNet architecture pre-trained on ImageNet, the images with their respective labels can be used to process the data. The desired result is a double classification of both Genus and Species. With a small dataset to train, the data were extrapolated using stretching and distortions, combined with Gaussian noise to fill out the dataset. More data are expected in the future, but this architecture is set to scale with the dataset and number of classes.

II. Methodology

A. Data Augmentation

As the dataset received through the Department of Agriculture is currently small due to the nature of the difficulty and time needed to label the data, data augmentation was used to expand the training and test datasets. Given that the input data for future use as a classifier will be of multiple sizes and distortions due to varying camera and lighting availability, the data augmentation includes shifts, crops, stretches, flips, and rotations. This augmentation allowed for the expansion of the dataset from approximately 350 to 5000 images. Using 5000 images to train on the data allows for better accuracy of the training model by providing diversity in the samples. Without individual samples, which is a very time intensive process, doing data augmentation allows for each element of the dataset to be unique, preventing overfitting.

As the training images were all taken from the same camera, the input dimensions of 2280x2160 were consistent for the input images. Using these values, random shift values were chosen for height and width. This allows the data to show different elements of the bees to learn all sides of the bees of that species or genus. Additionally, random widths and heights were chosen to stretch the images to fill the required sets. This stretching and shifting alleviates some of the overfitting that would be present when using the same image multiple times.

Additionally, in the model itself flipping the image vertically and horizontally, as well as adding Gaussian noise were used to add variation to the images. By varying the directionality of the image, weights become direction invariant, which is important as which side of the bee is photographed should not impact prediction performance. Adding Gaussian noise adds additional variation to each input and allows for generalization of learning patterns, keeping the model from overfitting on specific samples. In total, this leads to a variation of images, as is illustrated by the comparison found in Figure 1

Additionally, there were a few images that were held out and had the same augmentation performed to create a test dataset. The holdout dataset were combined with a set of unique generated images—using the above process—for each of the species of bee. This combined set became the test or validation dataset.

B. Network Architecture

The network, or model, used is based on a Tensorflow Keras ResNet model pre-trained on ImageNet [1]. Using the ResNet as the base of the model allows for the pre-trained weights to initialize in a manner that is pre-solved to search for identifying patterns in images. The rest of the model is set up to reduce the dimensionality of the output to get only the number of classes.

Each layer included is significant and necessary to achieve high levels of classification. As stated above, Gaussian Noise is added to prevent overfitting. This is done by adding zero-centered Gaussian noise to the image. By doing this, we are

(a) Original Image

(b) Generated Image

Fig. 1: A comparison of the original image to a generated image

Algorithm 1 Classification Model

```
procedure MODEL(InputTensor, Label)
    Gaussian_Noise(stdev = 0.5)
    ResNet50(weights = trained_from_ImageNet)
    Flatten()
    Dense(size = 256, activation = ReLu)
    Dropout(0.5)
    num_class = NumberofClasses
    Dense(size = num_class, activation = Softmax)
end procedure
```

augmenting the data and creating artificial variation between training samples. The Gaussian Noise layer is only active on the training, or fit, functionality. It is not active during the predict functionality. As our data were generated based on relatively few images, this data variation allows for a unique set of data for each image.

The ResNet layer is discussed more below; however, it serves as the main training layer. Here the weights are determined for the classification. They are initialized with pre-trained weights based on a previous training using the ImageNet dataset [2].

As the output of the ResNet layer is 2D, the Flatten function takes each output results and flattens it into a 1D layer. This allows for a Dense layer to be run to combine and condense the data from the output of the ResNet layer. Since the number of classes is significantly less than 1000, which is the number of outputs from the ResNet layer, this combining is done in two steps. First, the layer is condensed to 256 elements, then half of the elements are removed using Dropout. Dropout prevents overfitting by eliminating reliance on any small subset of the elements. As the output layer is being condensed from 256 to 6 (or 4)—the number of classes in our dataset—a dropout factor of 0.5 is used, which means half the elements in this layer are set to 0 randomly for each iteration. The resulting data from the dropout layer is then condensed into a layer of 6, representing the classes.

The output scores here are given a softmax activation layer, giving an accuracy value to the confidence interval of each class. Softmax is a logistic regression used to give each score a value from (0,1) and forces the sum of the predictions to equal 1 as can be seen in Figure 2. The softmax function

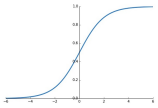


Fig. 2: The Softmax Activation Layer

can be represented by the following expression, with K as the number of classes:

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \text{ for } i = 1, ..., K \qquad (1)$$

The activation layer is done after the layer it is associated with. Therefore, the output of the model is the output of the Softmax of the final layer, and is a probability distribution over output classes. The maximum value of this output layer is then the predicted class. This is solved for using NumPy's argmax argument to get the most probable class.

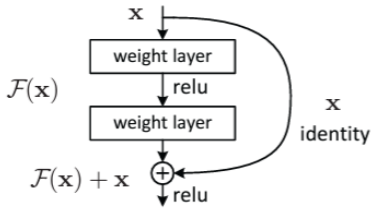


Fig. 3: Building Block of ResNet

C. ResNet

ResNet is a very generalizable classifier, and is commonly used for large scale image classification problems [1]. ResNet allows for extremely deep Convolutional Neural Nets (CNN) without vanishing gradients. This is because most deep networks have backpropogation that shrinks every layer as it transverses the path back to input; therefore having gradients that are non-existent by the initial layers. The solution to this problem in ResNet is done by adding an identity connection between layers. The building block of this network can be seen in Figure 3, from [1].

The addition of the $F(x)$, or the residual, with the x, or the identity mapping, gives a value that is variations on the identity. This means that instead of approximating the mapping from input to output, the ResNet is approximating the difference between the input and output with respect to the input. With $H(x)$ being the underlying mapping from input to output, instead of solving for $H(x)$ directly, it is solved for by using a residual approximation. $F(x) = H(x) - x$. This reduces the gradient vanishing problem by providing a dual path for the backpropogation to take, both the identity path and the residual path.

As each building block is approximated by $y = F(x) + x$ in an element-wise addition, there is no added computational complexity for each layer, or additional parameters. However, the identity mapping provides a path for backpropogation to travel without traversing any weight matrices. This allows

for the backpropogation gradients to reach the initial layers without vanishing gradients.

III. Data Feedthrough For Multiple Networks

As the point of this series of networks is to have two labels, Genus and Species, the Keras data feedthrough is set up to read only the species level data, as that is how the data are labeled. Therefore, a modification to the feedthrough is needed to map all items in the same Genus together. By creating a new data generation function that reads in each label and maps it to the corresponding genus, the feedthrough is able to be used, while relabeling the data.

This is done by taking the dictionary of the feedthrough labels from the classification and using the dictionary values to collect the genus name from the folder label. Taking these dictionary values and keys, it is possible to map the keys from the species level dataset and re-map the data to fit the Genus level classification. By using this remapping, the genus level labels and mapping can be done without any changes to the original data structure. This permits both labels to be calculated in sequence without replication of data.

A. Results

1) Genus: Using the data feedthrough method mentioned above, the data are combined to map the species levels together. There are 4 total Genus classifications: Anthophora, Habropoda, Megachile, and Melissodes. Both Anthophora and Habropoda each have only a single species represented, while Melissodes and Megachile each have two species represented in this dataset.

With this set of labels, the confusion matrix shown in Table I takes a sample of validation images and predicts the outcome, the confusion matrix shows the number of predictions and the true labels. This gives an idea of which classifications are most often confused with others. From this we can see that the images from the Melissodes Genus is the most likely to be classified correctly.

The training and validation accuracy Figure 5 shows higher accuracy on the training set than the test dataset, with more erratic results on the validation data. The loss values are low on the training set, but there is a large difference between the training and test set losses. Though both are in the same scale of learning. This shows some overfitting on the Genus level, but could potentially be avoided by continuing to train as the validation loss is decreasing over time. As more data are available, this can be made more robust, and there will be more variation in both the training and test sets, leading to a higher correlation between the training and test accuracy values.

In Figure 4, the softmax values are all approximately 1.0, meaning the classifier is very certain of that prediction. This is promising as each of the validation images in the dataset have more variation than those of the test set. This shows visually that the labels can be correctly predicted with high confidence even with partial images of bees.

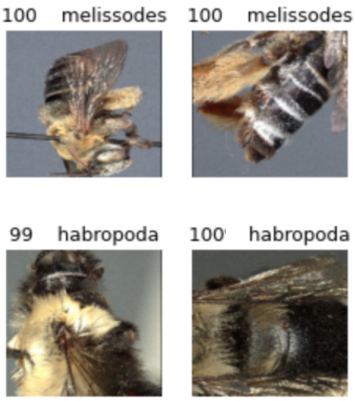
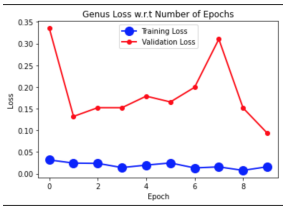
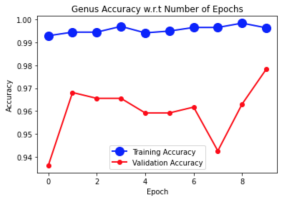


Fig. 4: Image with Softmax confidence values $\times$ 100



(a) Loss Values



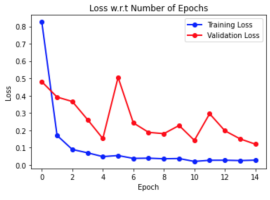
(b) Classification Accuracy

Fig. 5: The Genus Loss and Accuracy per Epoch through Training

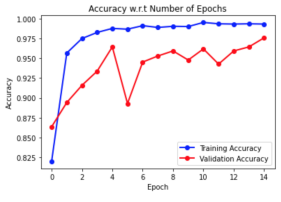
2) Species: For the Species level classification, there were 6 classes: Campanulae, Comptoides, Communis, Petulans, Abrupta, and Laboriosa. With this set of labels, the confusion matrix shown in Table II takes a sample of validation images and predicts the outcome, the confusion matrix shows the number of predictions and the true labels. This gives an idea of which classifications are most often confused with others.

TABLE I: Genus Confusion Matrix

Prediction	**True Label**			
	Anth.	*Habr.*	*Mega.*	*Mel.*
Anthophora	206	4	1	0
Habropoda	5	206	1	0
Megachile	2	0	249	0
Melissodes	1	2	0	107



(a) Loss Values



(b) Classification Accuracy

Fig. 6: The Species Loss and Accuracy per Epoch through Training

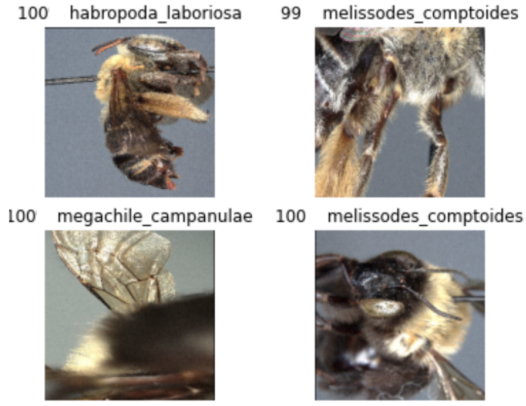


Fig. 7: Image with Softmax confidence values × 100

From this we can see that the images are mostly classified correctly, and that much of the confusion happens between bees that belong to different Genus classes.

The training validation chart 6 shows higher accuracy on the training set than the test dataset. This is expected, as is the more erratic nature of the test accuracy values. There is convergence on the training and test shapes though, which is the ideal case. With more varied test and validation data, the difference between the training and validation datasets should decrease, and the loss should converge more quickly. However, the results look promising, with 98% precision and recall. The Species level classifier looks promising and accurate.

By looking at the outputs of the validation set with the softmax scores, such as in Figure 7, it is apparent that the species level identification returns Softmax values that mark most classifications as very certain. This softmax output score can be used to threshold the values to mark some images as ones an expert needs to revisit to confirm the classification. However, in the case for these images, the classifications are fairly certain.

TABLE II: Confusion Matrix

	True Label					
Prediction	*Abr.*	*Lab.*	*Camp.*	*Pet.*	*Comm.*	*Comp.*
Abrupta	206	4	1	0	0	0
Laboriosa	5	206	1	0	0	0
Campanulae	1	0	229	1	0	0
Petulans	1	0	1	18	0	0
Communis	0	1	0	0	29	0
Comptoides	1	1	0	0	1	77

IV. Future Work

For each sample, there are data representing the location, date, time, health, and bee type (worker, queen, drone, etc.) and other relevant data for each sample. Integrating these elements into the training will allow for more accurate labeling of the data based on location patterns. Alternately, comparing expected bee types according to location and adding flags to Genus and Species level classifications that are unexpected for the location in the data. These are improvements that would help with the purpose of the data collection for identifying the location and migration of bee populations in the United States of America.

In this project, a goal is to label by gender as well as Species and Genus. As the project currently stands, there are no image samples for male bees to integrate into the dataset. Once these samples are identified and can be integrated, the goal is to take each input image and related data and label it with Genus, Species, and gender of the bee. These three labels are the most important three labels for this project as they are the most intensive to identify when labeling by hand. This will then be tied to the location data to help with bee migration research.

V. Conclusion

Data modifications can be used to expand minimal datasets to provide additional data images. Using the expanded dataset, it is possible to effectively train a classifier of Bees using images from labeled image samples. These samples can be collected from varying locations and image sizes, and will still produce viable results. Using both the Genus and Species to classify the images, the classification of bee information can be made more accurate, and account for variation with Genus classes. By automating bee identification, Entomologists may spend more time identifying bee location and migration patterns.

References

[1] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition", 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1512.03385

[2] R. Meudec, A. Hundt, Y. Xie, S. Schott, G. Marimiesse, C. Rauch, "resnet", 2019. [Online]. Available: https://github.com/keras-team/keras-contrib/blob/master/keras_contrib/applications/resnet.py

[3] C. McQueen, A. Asdaghpour, "BeeClassifier", (2022), [Source code]. [Online]. Available: https://github.com/CatMcQueen/ECE523_ML/blob/main/FinalProject_BeeClassifier.ipynb