Insect Identification Project for Agricultural Advancement

Archith Seshadri

10/1/22

Abstract:

Insects are complex creatures whose influence and characteristics can drive ecosystems or destroy plant populations. Therefore, it is imperative to recognize the differences between them and whether they pose harm or benefits, yet the majority of humans do not have a keen view for the differences in insect qualities. Through the use of image data, we developed an artificial intelligence system which is able to predict an insect's species based on a photo of a given insect. To assess the image data, we developed a convolutional neural network (CNN) which uses a series of neurons to output a result which we can interpret to identify the given insect. The highest validation accuracy we received was approximately 80%, which given the very subtle differences between the insects we used, the amount of categories we were defining from, and the limited number of images in each category, was a decently high validation accuracy. From this, we could conclude that we can not only identify a species, but with relatively high accuracy, giving us insight into how insect identification is a valid and applicable process which can be implemented into agricultural and pest control technology.

Introduction:

A program for insect identification would be able to help ecologists, exterminators, farmers, and many others in agricultural or insect related professions in identifying the insects around them on whether they are pesticidal or beneficial, dangerous or non-dangerous, and invasive or native to a certain area. For gardeners and farmers, this can help them determine the relationship the insects have with the plants in whether they are benefitting them or denying them resources. Our situation easily lends itself to a supervised problem with the purpose of classification. Classification models can be very effective, and also are very diverse in their methods for classification.

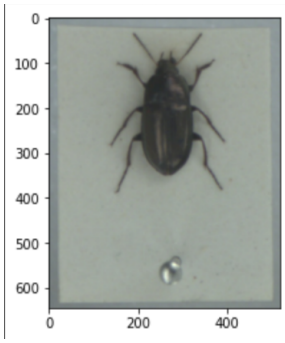
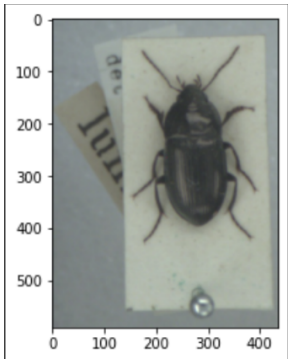
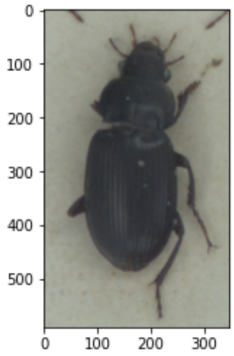
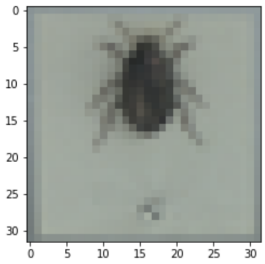
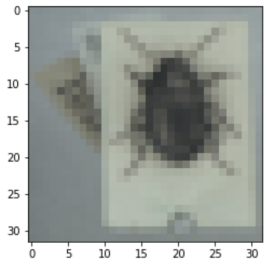
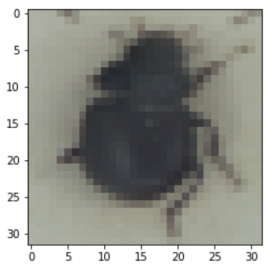
Other individuals have also delved upon the idea of insect identification. One specific research paper talked about how insect identification can be used for the documentation of insect biomass for researchers and animal preservers to make sure necessary species of insects don't go extinct (Oskar L. P. Hansen et al, 2019). Documentation of insect biomass is crucial as certain insects play vital roles in ecosystems, so their presence must be maintained and sustained over time. Despite this, the way they mention their process is only possible for the documentation of ground dwelling insects, which only accounts for a set percentage of the insect population. They also used the same process of a convolutional neural network, achieving an accuracy of around 56% while categorizing over the entirety of 291 insect species. In my research paper, I plan to recreate this setup of a convolutional neural network with this dataset, yet minimizing the scope to see what higher percentages can be achieved over a still adequate amount of categories.

Another research paper talks about the process of insect identification for the purpose of pest control, a similar topic I talked about in the applications of my neural network (Yong Ai et al., 2020). One fascinating idea this research paper presents is the process of identifying insects within stages of their growth, as certain insects are more or less dangerous at certain stages of their life. This idea requires more computational power

though and was therefore more expensive to implement. Finally, another drawback of this research paper is that they present the idea of pest control over the entire landscape of Chinese farms, yet the monitoring process would be tedious and expensive to implement. This is why I present the idea of insect identification for pest control at a smaller level such as in specified or sectional farming, ensuring that the process is done precisely.

Dataset:

The dataset we used involves 63,000 images of Beetles over 291 species, taken from the National History Museum of London's archives of insects (Oskar L. P. Hansen et al, 2019). This dataset was given to us with the Global Biodiversity Information Facility (GBIF) numbers attached as the labels for the image data. The GBIF number is a number used to index a species as easily as possible for the purpose of taxonomy. To convert these numbers to a readable species name, we had to import a library known as pygbif, which allowed us access to a dictionary of labels for a species once we indexed its GBIF number. From there, we simply got the value for the specific key in the dictionary, such as species name, and outputted that. For our dataset that we will use to train our model, we minimized it to include only 20 of the species in order to still identify over a decent range of images without sacrificing validation accuracy. After creating our dataset of images and image labels, we performed image preprocessing. This involved resizing all the images to a 32 by 32 grid of pixels and grayscaling the images so instead of 3 color schemes of red, green, and blue, we created just one color scheme. This is seen in the figure below, detailing the look of a regular image, the resized version, and then the gray scaled version:

Image Name	Amara Aenea	Amara Lunicollis	Amara Aulica
Regular Look			
Resized			

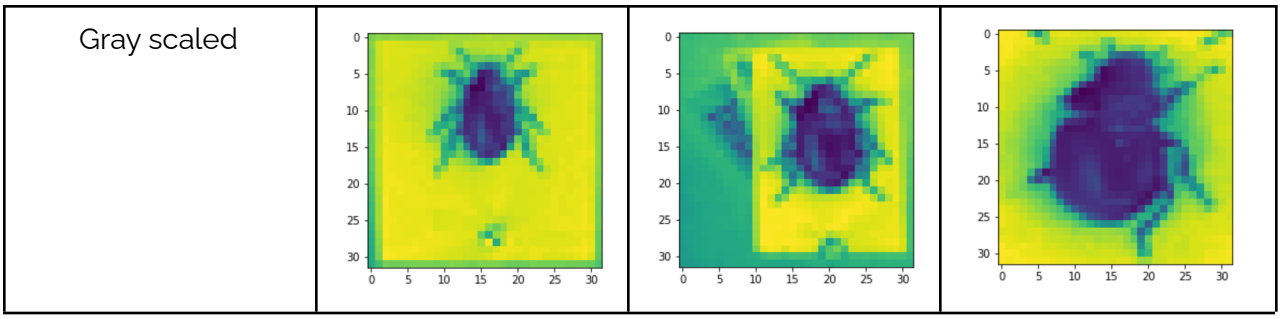


Figure 1: *How our dataset images look with and without the preprocessing.*

Finally, we normalized the images to make the pixel numbers from 0-255 to 0-1. All these operations facilitate the process of identification, making it easier for the neural network to identify patterns which it later uses to categorize the images.

Methodology / Models:

.As stated previously, a CNN works through a set of neurons, yet these neurons also have weights supplied to them which are tweaked as the model learns new characteristics about the images. As a part of our CNN, we then include kernels and poolings which facilitate our model in understanding these characteristics in our images. Kernels convert pixel numbers to certain values by multiplying each pixel by a set array of numbers and then adding all the values together. This helps the computer identify where the patterns are in an image in terms of location and intensity of pixel values. Poolings on the other hand help reduce our input by taking the largest value in a group of pixels and arraying through the pixels. This helps the model collect the most important information to therefore learn better. These are all seen in this diagram of our basic model:

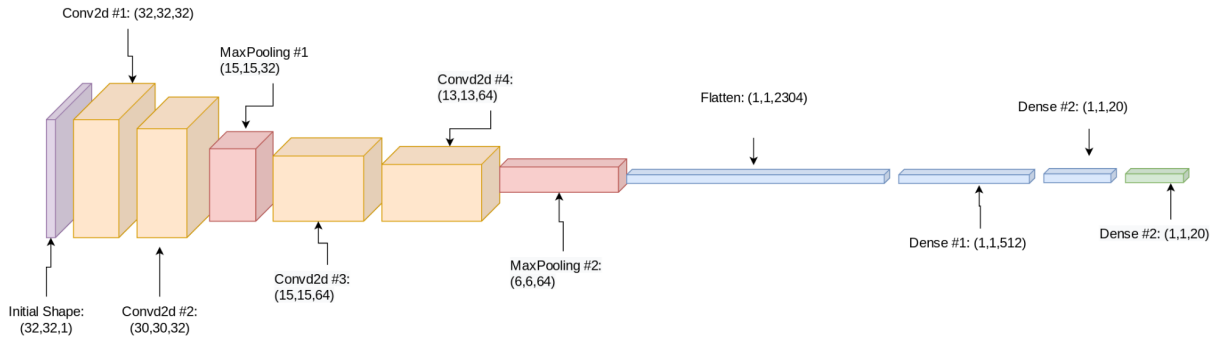


Figure 2: *Our basic CNN architecture*

This diagram illustrates how the size of the model's inputs, outputs, and shape changes as we implement more kernels, and pool, until we eventually flatten the model to create our 1 dimensional array of numbers which is inputted into our neurons. The first layer consists of 512 neurons, meaning each of our 2304 values gets attributed to 512 neurons,

and then finally to 20 neurons. Softmax finally pools these final 20 numbers into probabilities which we can then use to assess which category our input belongs to.

Though our initial model ran successfully, it produced a relatively low accuracy. Most CNN models follow a pattern in which after the initial trials or epochs of running the testing data, the validation accuracy increases drastically, yet after later trials, the accuracy plateaus. To see how our model improves and consequently maxes out over time, we looked at where our testing accuracy flattened out, running for more and more epochs. Our model's accuracy consistently increased until around 60-75 epochs, in which it plateaued. This can be seen in the graph detailing the changes in accuracy as the epochs increase:

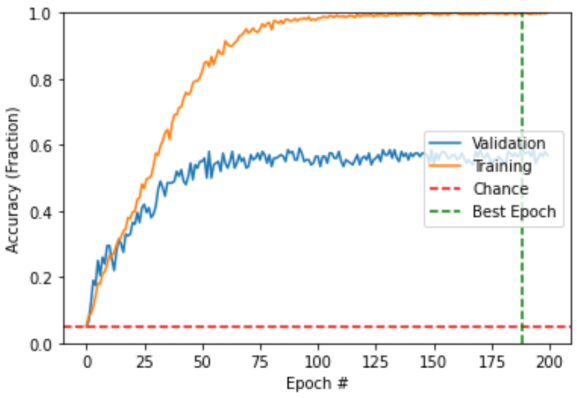


Figure 3: *Graph of our basic model's accuracy as we increase the amount of epochs*

Once we knew where our model plateaued, our next step to achieve a more successful model was to run hyperparameter tuning, effectively deciding which parameters should be tweaked to give us the best result. This involved changing the number of images per category, changing both the pooling and the kernel size, and changing the size of the first dense layer, which were all documented through running the model each time under the same conditions, and simply tweaking the focus, or our parameter. Other parameters were also tweaked through the process of hyperparameter tuning, in which the model will emulate this process on its own, selecting for a random value in the parameter and seeing how that affects the accuracy. We executed this hyperparameter tuning for certain characteristics such as the output size of the 2nd and 3rd convolutional layers, the activation function of the first hidden layers, and the learning rate of the optimizer.

Results and Discussion:

From our hyperparameter tuning we were able to notice patterns evolving in each of the focus points as we altered them. These patterns could be seen in this graphs shown below:

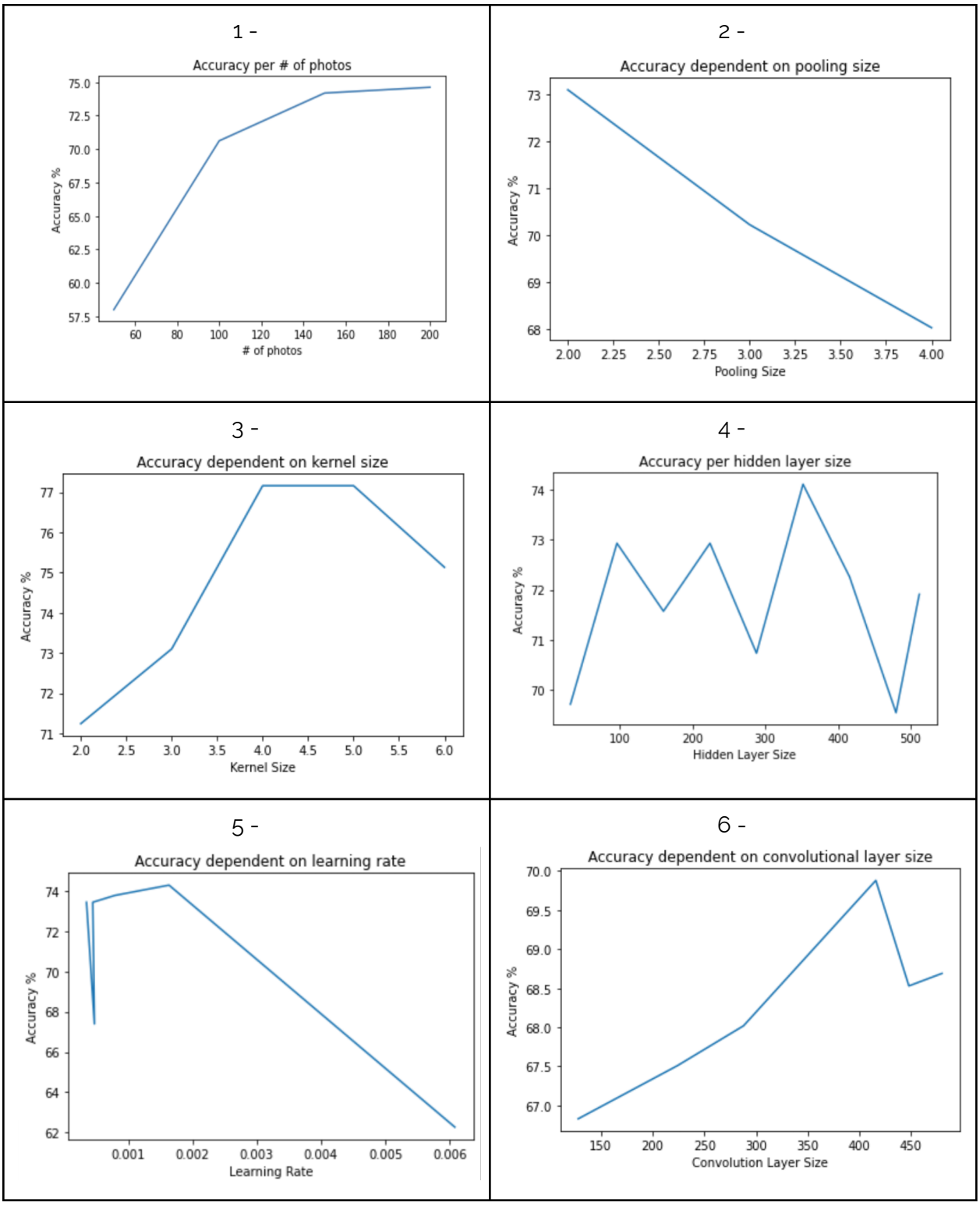


Figure 4: *Graphs of accuracy over our hyperparameters*

In the first graph, we can explicitly see how as we increase the number of images, the accuracy follows suit. This pattern also would plateau, making us account for the fact that at a certain point, it's not worth adding more images as the processing power would be greatly increased for a minimal increase in accuracy. Next, the graph for pooling size highlighted that as the pooling size increased, the validation accuracy decreased. We also didn't include a pooling size of 1 as we can assume the effect would be the same as no pooling size at all as we are taking the highest value of a single pixel which is simply the pixel value itself. Graph number 3 then shows that kernel size is nearly parabolic, in that we can choose the best kernel size to be the peak of the function. After that, graph number 4 depicts how accuracy behaves for different hidden layer sizes. The erratic shape of this graph shows us that since no clear pattern is depicted, the most ideal value is determined by so many different factors that its path can't be fitted. Graph number 5 illustrates the effect of learning rate on accuracy, which we can generalize its pattern to say that as its value increases, the validation accuracy decreases. Finally, graph 6 shows how the size of a convolutional layer influences peaks in accuracy which is our main determinant for the value we should use.

In determining which hyperparameters we should use, we concurrently ran hyper parameter tuning which was not only selecting for one trait, but selecting for combinations of traits. This generally proves to be more effective in determining which value to use, as the parameters are dependent and rely on one another, so the best parameter value when selected on its own might not still be the best value when in tangent with other parameters, From this hyperparameter tuning, we get the following to adjust from basic model:

Trial summary
Best Hyperparameters:
number of layers: 3
2nd conv2d: 320
3rd conv2d: 64
Dense Layer: 64
Activation Type: tanh
Learning Rate: 0.00031096248743627106
Score: 0.7817258834838867

Using this data, we adjusted the hyperparameters as such. We also combined this data with what our best value was for pooling size and our best value for kernel size in what we considered as our best model. This gave us a model architecture which looked like this:

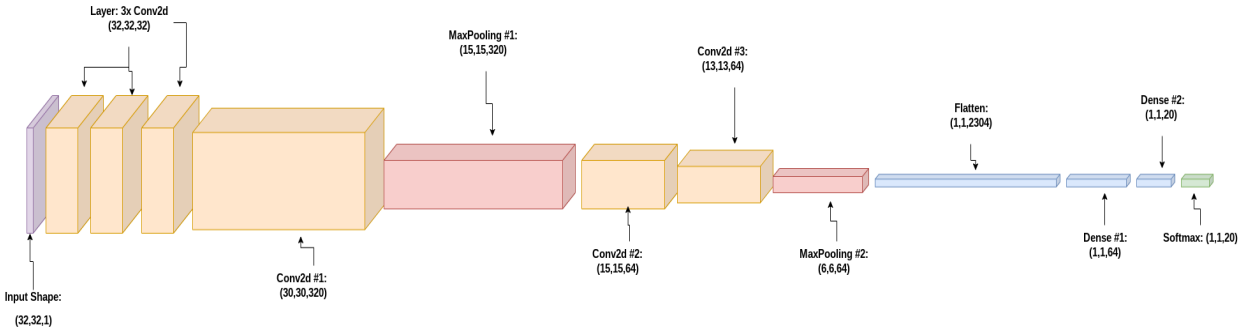


***Figure 5**: CNN architecture of our best model*

As we ran this model it produced an accuracy of 80.71%, 2.54% higher than any previous model we obtained from our hyperparameters, giving us our finalized model for this dataset.

Conclusions:

Through a convolutional neural network, we were able to predict the species of beetles among 20 species types. We even tested this model with an image not from our dataset, testing how different color backgrounds or different sizes of the same beetle affect how well our model predicts. It was able to predict it correctly, illustrating that our model can be functional for real world images. This is still premature though given the input of only one image, so a further task would be to continue this investigation with multiple backgrounds, especially to those taken from real world scenarios with, for example, vegetation in the background. We also accounted for several tradeoffs in the development of our best model, one of them being how much processing power the model uses in comparison to the accuracy it outputs. This is given the limited access to processing power available for our model, so if a hyperparameter only gives a slight increase in accuracy, it might not be worth implementing it for the increased processing power it takes to apply it. If we were to apply this model to further applications, it might be necessary to use this processing power where accuracy is more crucial, such as in identifying pests for farmland, where economic risks are at stake.

This research could also be extended for further use, in which additional steps can be taken to determine the most ideal model or application. This involves tweaking model characteristics, such as the size of our input image, as it would let us assess more qualities in the image. This would also come at the cost of processing power. Our CNN architecture also falls under this category, as if we could assess under more hyperparameters, we could select for different characteristics which affect the model differently, letting us know definitively what has the largest impact on our model's accuracy. Finally, we could expand the scope of our data, identifying over more types of beetles, or even expanding towards different insects. By expanding our data, this would let farmers, gardeners, exterminators , and insect enthusiasts to be able to identify the insects around them, letting them utilize this data to create flourishing ecosystems and environments while also educating on insect variety.

Acknowledgments:

This research paper was created with the guidance of Barbie Duckworth, an MIT alumni and employee of the Inspirit AI scholars program. Resources and information for this project were also endowed by the Inspirit AI program.

References

Kaloudis, S., et al. "Insect Identification Expert System for Forest Protection." *Expert Systems with Applications*, Pergamon, 6 Jan. 2005, https://www.sciencedirect.com/science/article/abs/pii/S0957417404001496 .

Larios, Natalia, et al. "Automated Insect Identification through Concatenated Histograms of Local Appearance Features: Feature Vector Generation and Region Detection for Deformable Objects - Machine Vision and Applications." *SpringerLink*, Springer-Verlag, 7 July 2007, https://link.springer.com/article/10.1007/s00138-007-0086-y.

"Research on Recognition Model of Crop Diseases and Insect Pests Based on Deep Learning in Harsh Environments." *IEEE Xplore*, https://ieeexplore.ieee.org/abstract/document/9201298 .

Oskar L. P. Hansen, et al. "Species-level image classification with convolutional neural network enables insect identification from habitus images." 24 December 2019, https://onlinelibrary.wiley.com/doi/full/10.1002/ece3.5921

Rosebrock, Adrian. "Keras conv2d and Convolutional Layers." *PyImageSearch*, 17 Apr. 2021, https://pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/.

Sharma-, ByPalash, et al. "Keras Dense Layer Explained for Beginners." *MLK - Machine Learning Knowledge*, 20 Oct. 2020, https://machinelearningknowledge.ai/keras-dense-layer-explained-for-beginners/.

Link to the AI found here:

Steps to create the model:

https://github.com/archith-s/Insect-Identification/blob/main/Insect_Identification.ipynb

Best model in use:

https://github.com/archith-s/Insect-Identification/blob/main/Best_Model.ipynb