Using Convolutional Neural Networks to Identify Snake Species

Sam Myers

Computer Science

Truman State University

Kirksville Missouri United States of America

Sjm3253@truman.edu

ABSTRACT

The goal of this study was to create an artificial intelligence capable of correctly identifying the species of a snake greater than 90% of the time, with a secondary goal of analyzing the effect that the number of epochs has on accuracy, where an epoch is the number of times the program cycles through the full training dataset. To do so an artificial intelligence system (AI) was developed which utilized two convolutional neural networks capable of using a variable number of epochs. This system was trained with 104 images per species, sized 384x384. Using this system, across 5 levels of epochs (50, 100, 150, 200, 250), three trials were run, averaging 69%, 88.33%, 89.66%, 94.33%, and 98.33% respectively, at one point reaching 99% accuracy, and clearly showing how increasing the number of epochs increases the accuracy of the system, albeit with diminishing returns.

INTRODUCTION

In an average year anywhere from 81,000 to 138,000 people die as a result of a snake bite according to the World Health Organization [2021]. Around three times this number must undergo amputations or suffer other permanent disabilities, with most being children or agricultural workers. This is particularly problematic for agricultural workers who often depend on their ability to perform manual labor to support themselves and their families. With antivenoms being difficult to produce and a general lack of data making it difficult to accurately prepare treatments in areas where snakebites are common, one must ask if it is possible to prevent venomous snakebites with modern-day technology, namely with AI capable of image recognition.

While it is impractical for humans to survey large areas with the specific goal of finding venomous snakes, this act is possible for an AI working in combination with a camera system. This paper focuses on the differentiation of a nonvenomous species, and venomous species, specifically the Horned Rattlesnake (Crotalus cerastes) and the Eastern Garter Snake (Thamnophis sirtalis sirtalis). While this admittedly is a quite specific issue, it aims to show the feasibility of such a project and the ease with which one could accomplish such a task. This paper also features a user-accessible page in which one can submit their own images for consideration by the AI.

This study also covers the impact that the number of epochs has on the system's accuracy, with data being used from systems with 50, 100, 150, 200, and 250 epochs. While the number of epochs changes, the number of images iterated over stays constant at 104 for each species. This number was chosen for its ability to simultaneously provide sufficient data and accuracy and a realistic run time for the system. These images are a standardized 384x384 and are manipulated in-system (rescaled, zoomed, sheared, and flipped) to better train the system.

This study also has an accompanying webpage one may use to input their own images. The basics of this webpage were developed by Professor Donald J. Patterson of Westmont University and are designed to be integrated with a similar image recognition AI. This page has been modified to reflect the purpose of this study, however, all credit in the base webpage should be directed to Professor Patterson.

LITERATURE REVIEW

The field of image recognition as a facet of artificial intelligence is still relatively new, yet has a complex history full of groundbreaking studies and massive innovation. While the first AI-based image recognition systems have not yet reached a half-century in age, the idea of an artificial vision system has existed for thousands of years, first appearing in ancient Greek myth in the form of a bronze giant named Talos that was created by Hephaestus to patrol the island of Crete [Andreopoulos, 2013]. While the focus of image recognition has shifted from protecting Crete from potential attackers to more practical issues, the goal of automating a task to ease the work of humans has stayed the same. Modern computer vision began in the early 1960s, with the earliest applications being character recognition for automation-related tasks [Andreopoulos, 2013]. This work established the difficulty in image recognition, recognizing the challenges that come with illumination variability, time, cost and, sensor constraints. The first practical use of machine vision took place in the early 1970s, with the automated assembly of semiconductors. This production is arguably the most impactful application of machine vision, allowing the semiconductor revolution and the exponential rise in computing power over the last decades. Complementary to use in computer development is early recognition system's use in the biomedical field, namely in chromosome recognition, as well in the agricultural field to recognize various products [Andreopoulos, 2013]. From these successes the use of image recognition has only expanded, becoming available to almost any business, and in recent years, consumers as well. Despite its many successes, recognition outside of specifically tailored environments remains a difficult task even today.

Despite the difficulties, the idea of using image recognition to identify various animals is not an original one. Norouzzadeh et al.'s study "Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning" has a similar goal, identifying, counting, and describing the behaviors of 48 animals present in the Serengeti desert [2018]. Norouzzadeh et al.'s study utilizes the SS project, the world's largest camera-trap project, a collection of 225 camera traps located in the Serengeti National Park. The dataset holds over 1.2 million capture events of 48 different species, for a total of 3.2 million individual images, each labeled by volunteer citizen-scientists [2018]. These numbers however are limited due to the fact that over 75% of images contain no animal, as well as a strong bias towards certain animals and actions. These limits cut the total training dataset down to 1.4 million images and the test set to 105,000 images. These images were used across 9 models (AlexNet, NiN, VGG, GoogleNet, ResNet-18, Res-Net-34,ResNet-50, RestNet-101, and ResNet-152) with four goals in mind. The first goal was detecting images that contained animals, in which the architecture VGG recorded the accuracy, a rate of 96.8%. The second goal, identifying species, also found success, with the model ensemble being able to identify the animal present with 94.9% top-1 accuracy and 99.1 top-5 accuracy (where top-1 refers to the systems guess with the highest certainty, and top-5 refers to the system's top five highest certainty guesses). The third goal, counting animals, proved more difficult, with the ensemble of models only being able to achieve 63.1% top-1 accuracy, and 84.7% accuracy within a margin of one. Task four, identifying additional attributes, such as if an animal is standing, sitting, or whether young are present, proved similarly difficult due to the non-exclusivity of actions (For example, an animal could be standing, eating, and have a young present). As such, the ensemble of models was able to achieve a 76.2% accuracy [Norouzzadeh et al., 2018]. Based on these levels of accuracy the study predicted that their identification system could save roughly 8.4 years of 40-hour per week human labeling effort out of the original 14.6 years of effort needed.

Other studies have focused more specifically on the classification of snake species with computer vision as well, such as Durso et al.'s 2021 study "Supervised Learning Computer Vision Benchmark for Snake Species Identification From Photographs: Implications for Herpetology and Global Health". In this study 82,601 images were used, representing 45 of the most well-represented and medically significant snakes. Durso et al.'s study utilized the collaborative platform AICrowd, resulting in five algorithms with a top-1 accuracy of over 75%, and a top top-1 accuracy of 86.1%. The top algorithm employed incremental learning in neural networks, as well as parts of EfficientNet from Google Brain and a pre-trained network from ILSRVC, discriminative learning, cyclic learning rates, and automated image object detection [Durso et al, 2021]. This algorithm was built around a pretrained EfficientNet network, which retained information learned from the ImageNet Large Scale Visual Recognition Compendium, a compendium of over 1.2 million images sorted into 1,000 different classes. From this, a discriminative learning strategy was added, allowing different layers to capture different types of information, where initial layers are trained at a lower learning rate than the final layers are. The images were then automatically subsequently resized and cropped to provide the program with a clearer view of the snake itself before being run through a pre-trained EfficientNet, which balanced the depth and width of the architecture with Swish activation function, image resolution, as well as using appropriate dropouts [Durso et al, 2021].

A similar study comes from Petal et al. in "Revealing the Unknown: Real-Time Recognition of Galapagos Snake Species Using Deep Learning". This study, published in May 2020 was based around the nine Galapagos snake species and had two main tasks; object detection and classification of Pseudalsophis snake species. To detect the object in frame, the study employed a region-based convolutional neural network (R-CNN). The R-CNN has three main layers of operations: feature network, regional proposal network (RPN), and detection network. The R-CNN was implemented into four different architectures, these being Inception V2, ResNet, MobileNet, and VGG16 [Patel et al., 2020]. Data was collected via the Tropic Herping collection of images, as well as from a Google Images Web Crawler and website "Flickr". This resulted in

247 images across 9 species. The level of accuracy varied greatly depending on which architecture was used, with MobileNet reporting 10% accuracy, Inception V2 reporting 70% accuracy, and VGG16 and Resnet both reporting 75% accuracy [Patel et al., 2020]. Petal et al. go on to note the impact of the limited dataset, hypothesizing that with a greater collection of images, higher accuracy would be able to be reached.

METHODOLOGY

This study uses the dataset "Pre-processed Snake Images" published by Sameeha Rahman on Kaggle.com. This dataset is cleaned up and preprocessed to remove overly noisy, blurry, small, and otherwise unusable images, ensuring only clear and uniform images are used in the architecture. Each image is 384*384 pixels in size, achieved via cropping and insertion of black bars where necessary. The original dataset contains images of the Northern Water snake (Nerodia sipedon), the Common Garter snake (Thamnophis sirtalis), Dekay's Brown snake (Storeria dekayi), the Black Rat snake (Patherophis obsoletus), and the Western Diamondback rattlesnake (Crotalus atrox), however, only images of the Common Garter snake and the Western Diamondback rattlesnake are used to train the neural network.

Before going into the specifics of this study's model, it is important to understand how a general image recognition AI works.

A typical image recognition AI will contain multiple layers, each with a different purpose. The primary layers are known as convolutional and pooling layers. Convolutional layers take the information from an image, convolve the information, and pass it to the next layer, much like a neuron does in the brain. An easy way to picture this is with a two-dimensional array of ones and zeroes. Each box in this array will represent an individual pixel. In a real image, there would be much more detail, as each pixel is made up of separate layers detailing the RGB values, however, this example will only contain one value. This array is processed by kernels, which serve as feature detectors. These kernels are smaller (typically 3x3 to 7x7) arrays that hold a pattern of their own. In this example, these values will be ones and zeroes as well. Visually, the array would appear as Figure 1, and the kernel as figure 2.

Figure 1: A Simple Image

0	1	1	0	1
1	0	1	0	1
0	0	0	1	1
0	1	1	1	0
1	1	0	1	1

Figure 2: A Simple Kernel

1	0	1
1	0	0
1	1	0

Dot Product multiplication is performed at each point where the kernel "fits" into the image, starting in the top left corner and finishing in the bottom right. Figure 3 represents what the first step in this process would look like, with Figure 4 representing the complete convolved feature.

Figure 3: First Dot Product Step

0x1	1x0	1x1	0	1
1x1	0x0	1x0	0	1
0x1	0x1	0x0	1	1
0	1	1	1	0
1	1	0	1	1

Figure 4: A Simple Convolved Feature

2	1	4
3	2	4
2	3	3

As this process repeats over a predetermined number of epochs, the system begins to be able to recognize repeating features, enabling the system to recognize individual objects.

A pooling layer is used to reduce the number of parameters needed to learn by summarizing the details of a convolved feature. This can be done in a variety of ways, from taking the maximum or averaging a part of a feature. This is performed in a way similar to convolving an image, with the region starting from the top left, and ending in the bottom right. For example, if a maximum pooling layer with a pool size of 2x2 was used on the convolved feature shown in Figure 4, the result would be Figure 5.

Figure 5: A Simple Pooled Layer

3	4
3	4

A neural network also needs an activation function, which is applied to the vector produced by either the convolutional layer or the pooled layer. In the case of this study, the activation function is applied after the convolutional layer. With this in mind, it is possible to look into the specifics of this study's network.

This model is built using Google's Tensorflow, which is broken down into three distinct parts: Preprocessing the data, building the model, and training and estimating the model. In this study's model, data preprocessing is limited, as the images are already preprocessed before entering the system. As such, the only necessary step is finding the order of the channels, which can vary from image type to image type. The dimensions are preloaded, and when combined with the channel order, give the input shape. From here, the model can be built.

Due to having a singular input and output, the model is sequential, with three layers. The first of these is a two-dimensional convolutional neural network that uses the earlier defined input shape. This CNN is made up of a convolutional layer of 32 filters and a 3x3 kernel, followed by an activation layer using a rectified linear unit function (Or ReLu). A ReLu function is used to avoid vanishing gradient descent. Finally, there is a pooling layer, which downsizes the data, using a pool size of 2x2. The first CNN is followed up by another CNN with similar characteristics. This CNN takes the input size of the previous layer and applies the same two-dimensional convolutional layer with 32 filters and a 3x3 kernel. This layer once again uses a ReLu activation function, as well as a pooling layer with a 2x2 pool size. The third layer is nearly the same, using a 64 layer, 3x3 kernel convolutional layer, a ReLu activation function, and a pooling layer with a 2x2 pool size.
Finally, the data is flattened, a dense layer is added, and ran through another ReLu function. A dropout layer is added to prevent overfitting, which is when a model focuses too much on the statistical noise present in training images. The dropout layer prevents this by imitating the effect of an ensemble, a collection of different neural networks. This is done by randomly "dropping out" some number of layer outputs, which causes each layer to be treated like a layer with a different number of nodes, effectively causing each layer to be viewed differently. A different dense layer is added and ran through a new activation function, this time a sigmoid function in order to give a probability of success. Finally, the model is compiled with the goal of minimizing loss and returned, allowing it to be trained.
The training process begins with rescaling, shearing, zooming, and flipping the images. This ensures the program can recognize the object in an image, regardless of the effects or distortions applied to it (i.e., a raindrop or scratch covering a camera's lens. The same is done to the images used to test the AI, however this time less distortion is used. The model is fit and trained using Tensorflow's fit_generator, a function that performs typical backpropagation and adjusts the weights of the model, using the number of epochs available. fit_generator also tests the accuracy on the test dataset, returning the accuracy for the user.

Results

After the program was run, the following results were output. Each result represents the accuracy across three trials for each number of epochs.

Table 1: Accuracy Across 3 Trials

	Trial 1	Trial 2	Trial 3
50 Epochs	78%	62%	67%
100 Epochs	91%	91%	83%
150 Epochs	96%	86%	87%
200 Epochs	92%	95%	96%
250 Epochs	99%	97%	99%

This averages out to the figures shown in Table 2.

Table 2: Average Accuracy Across Three Trials

50 Epochs	69.00%
100 Epochs	88.33%
150 Epochs	89.66%
200 Epochs	94.33%
250 Epochs	98.33%

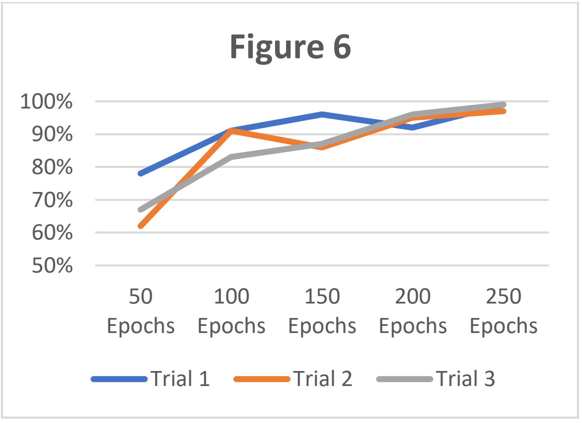
Table 3 notes the average difference between accuracy across each set of trials

Table 3: Average Difference in Accuracy Across Three Trials

50 Epochs	10.66%
100 Epochs	5.33%
150 Epochs	6.66%
200 Epochs	2.66%
250 Epochs	1.33%

Discussion

Overall, the results of the study paint a positive picture of the idea of using image recognition to categorize and label snakes. While some lower epoch systems struggled to accurately identify the differences between garter and rattlesnake, programs that used upwards of 200 epochs classified snake species correctly over 90% of the time, with some 250 epoch trials even reaching a 99% overall accuracy. The goal of this study was to reach a 90% accuracy, and as such, the systems utilizing 200 epochs and 250 epochs are considered to be a success, with the system using 250 epochs being the most successful. Such a system could be used in coordination with a camera system to pick out rattlesnakes and prevent potential injury and death with little human intervention.
While only differentiating between two species, this high rate of accuracy using a relatively low number of layers, epochs, and training images show that scanning for potentially harmful reptiles in a space where they are prevalent would be feasible and inexpensive, especially in regions that contain a limited number of venomous snakes.



One can also note the diminishing returns present in using a greater number of epochs. For example, the difference in average accuracy from 50 epochs to 100 epochs is 19.33%. From 100 epochs to 150 epochs, it is 1.33%, from 150 to 200 it is 4.66%, 200 to 250 a change of 4%. Simply raising the number of epochs from 50 to 100 causes a greater change in accuracy than changing the number of epochs by a further 150. Figure 6 demonstrates this well.

While this higher level of accuracy may be worth the extra processing power and time required in this case, it is important to recognize that in cases where less accuracy is needed, it would be prudent to employ a lesser number of epochs in order to reduce the time and resources needed.

Table 3 also notes the average difference across trials. As shown, with the exception being the 150-epoch system, this range decreases as the epochs increase. This is likely due to a variety of reasons, the first simply being the inability to increase in accuracy. As systems get more accurate, they approach an 100% rate that cannot be passed. This serves as a barrier, limiting the difference between systems. While a less accurate system may have trials that are both less effective and more effective, a more accurate system will have only lesser effective systems, as a more effective system is impossible. The average difference also is likely due to how stochastic gradient descent works. A system with fewer epochs is much more likely to get “caught” in local minimums when a lower global minimum exists elsewhere. In cases where there are a greater number of epochs, there are more chances for the program to recognize this error and correct it, finding the true local minimum.

While multiple systems were able to reach 90% accuracy, there were limitations in place that prevented higher accuracy, as well as higher accuracy in lower epoch systems. First is the processing power of the machine on which systems were run. Due to concerns of overheating the machine, the number of layers present in the system as well as the number of training images was purposely limited. Similar steps were taken in order to lessen the amount of time needed to run the systems. A system run on a more powerful machine, or a system run with more time available could undoubtedly reach a higher level of accuracy. Time restraints also limited the number of trials, as well as the spread of epochs available. Had a greater period of time been available, a more complete study could be performed on the impact that the number of epochs has on accuracy.

Conclusion

Overall, this study has proven successful, proving the feasibility of an image recognition system based around the classification of species of snakes. However, there is still a long way to go before a system can classify multiple species of snake with a high degree of accuracy, as well as potentially do it in an active environment. Future studies may look into classifying multiple species or utilize images in which other potentially harmful animals are also recognized (stinging insects, or animals that may be aggressive when disturbed). Other studies may also look into recognizing if snakes are potentially harmful in a live video, which could be used in coordination with a camera system to scan a potentially infested area before any humans are put at risk. A particularly interesting study would also be recognizing signs of recent snake habitation, such as markings left by a snake, or decarded snakeskin to determine if any potentially harmful reptiles were in the area recently.

Outside of animal-specific studies, other studies could focus more directly on the effect of varying the number of epochs, determining what number is best based on accuracy, and processing power/time required. Other studies could change the number of layers, the size of images, or the number of images present.

If this study were to be redone, a greater emphasis would be spent on determining the exact effect increasing the number of epochs has, using a greater number of trials as well as a greater number of epochs in order to provide a general equation. More focus would also be placed on the tradeoffs that come with a greater number of epochs, such as the time required to run the program, and the amount of energy expended to do so.

References

Andreopoulos, Alexander & Tsotsos, John. (2013). 50 Years of object recognition: Directions forward. Computer Vision and Image Understanding. 117. 827–891. 10.1016/j.cviu.2013.04.005.

Durso, A. M., Moorthy, G. K., Mohanty, S. P., Bolon, I., Salathé, M., &amp; Ruiz de Castañeda, R. (1AD, January 1). Supervised learning computer vision benchmark for snake species identification from photographs: Implications for herpetology and global health. Frontiers. https://www.frontiersin.org/articles/10.3389/frai.2021.582110/full.

Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C., Clune, J. (2018, June 19). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. PNAS. https://www.pnas.org/content/115/25/E5716.

Patel, A., Cheung, L., Khatod, N., Matijosaitiene, I., Arteaga, A., &amp; Gilkey, J. W. (2020, May 6). Revealing the unknown: Real-time recognition of galápagos snake species using deep learning. MDPI. https://www.mdpi.com/2076-2615/10/5/806.