Identifying Friend or Foe in Tampa Snake Species using Convolutional Neural Networks

Darri Stuber, Julia Piper, and Dr. Binod Rimal*

Data Science Program, Department of Mathematics

The University Of TAMPA

Abstract

The fear of snakes is notably one of the most widespread phobias among individuals. While rooted in primal instincts, this fear is further amplified by contemporary cultural beliefs and superstitions. Florida is home to approximately fifty distinct snake species, with six of them presenting venomous threats to humans. In response to this concern, we have developed a Convolutional Neural Network (CNN) model tailored to distinguish between seven prevalent snake species in Tampa, Florida. These seven species comprise five relatively non-threatening snakes—North American Racer, Ringneck snake, Eastern Rat Snake, Banded Water Snake, and Eastern Hognose Snake—and two hazardous pit-vipers, namely the Cottonmouth and Diamondback Rattlesnake. The significance of our model lies in its ability to differentiate visually similar snakes that significantly vary in their potential danger to humans. For example, discerning between the Diamondback Rattlesnake and Eastern Hognose Snake, or the Cottonmouth and Banded Water Snake, can pose a challenge to the untrained eye. Our CNN model holds particular promise as the initial component of a snake identification application, assisting individuals in recognizing encountered snakes and providing them with vital safety information.

Common Snake Species in Tampa Bay

DiamondBack RattleSnake
(*Crotalus atrox*)

Eastern Hognose Snake
(*Heterodon platirhinos*)

Eastern Rat Snake
(*Pantherophis alleghaniensis*)

Ringneck Snake
(*Diadophis punctatus*)

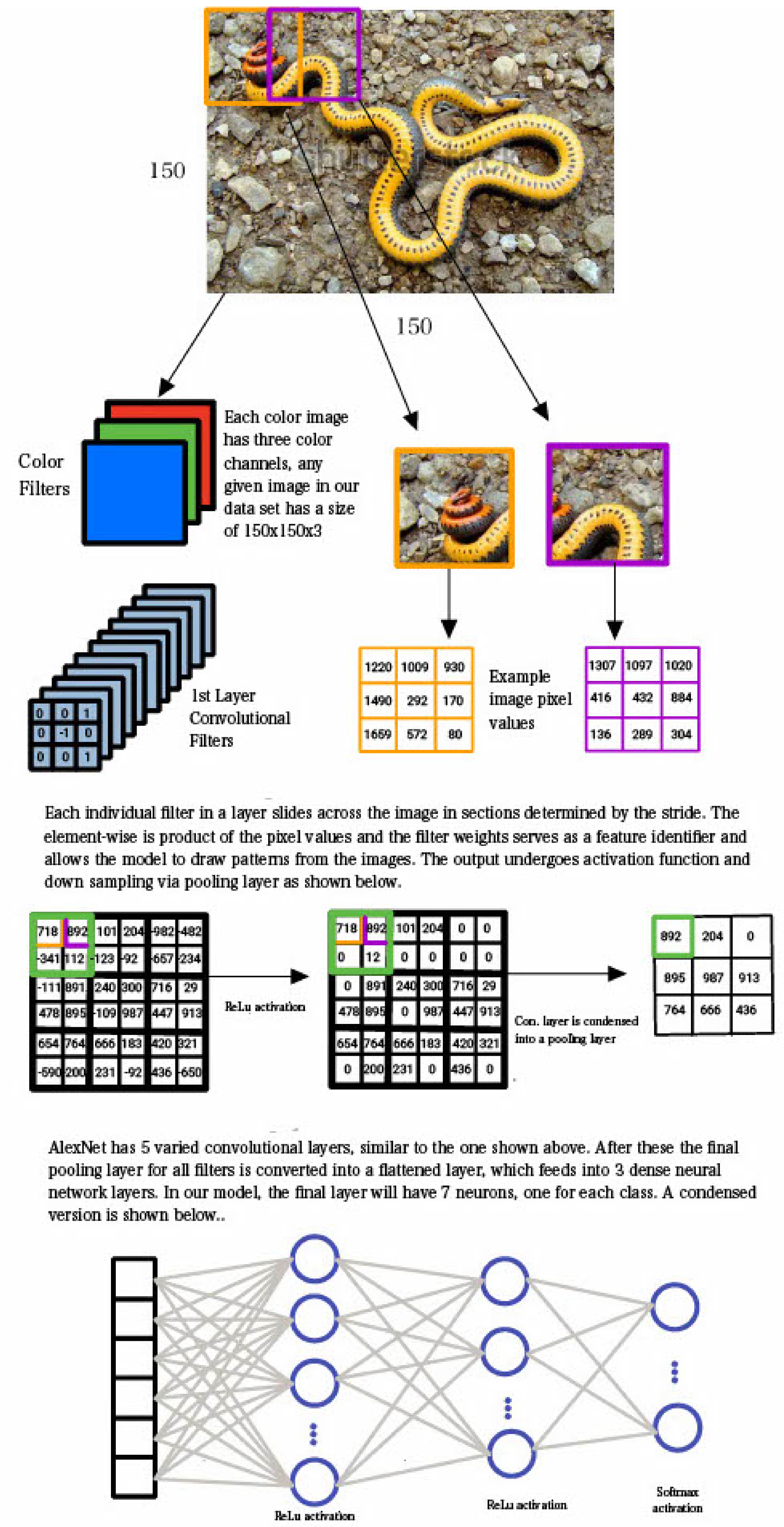
North American Racer
(*Coluber constrictor*)

Cottonmouth
(*Agkistrodon piscivorus*)

Banded Water Snake
(*Nerodia fasciata*)

This project utilized the public domain dataset "Identifying different Breeds of Snakes" created by Debadri Dutta [2020]. The original dataset contained classification images for 35 snake species but for the sake of our project seven species common in Tampa Bay, Florida were chosen. These 7 classes were extracted from the main dataset and used within our CNN. Flaws we encountered in this dataset were a variety of backgrounds and potential misclassifications. To account for this, we personally verified the data within our 7 classes and removed incorrect images. We also resized all images to 150x150 pixels for uniformity. Images from of Sciences National Geographic Society [2020].

Convolutional Neural Networks



Hyperparameter Tuning

For machine learning models, tuning for hyperparameters such as optimizer, batch size, and learning rate can improve the overall accuracy. Optimizers determine how model parameters are updated during backpropagation. Batch size is the number of samples processed prior to each model update. Learning rate denotes the cadence at which the error function of the model is minimized. After tuning,our final hyperparameters are Adagrad optimizer, a batch size of 16, and learning rate of 0.001. Results of hyperparameter tuning shown below [Rimal, 2022].

Optimizer	Accuracy
Adam	0.15
Nadam	0.15
Adagrad	0.16

Table 1. Best optimizer accuracy for 2 epochs and 2 replicates.

		Learning Rate		
		0.001	0.01	0.05
Batch Size	16	0.329	0.168	0.178
	32	0.320	0.181	0.213
	64	0.319	0.188	0.224

Table 2. Batch size and learning rate hyperparameter tuning with adagrad optimizer. Best accuracies for 2 replicates and 3 epochs shown.

Coding an AlexNet CNN

1st Convolutional layer with maxpooling

```
alex_model.add(Conv2D(96, (11, 11), activation='relu', strides=(4, 4), input_shape=(150, 150, 3)))
alex_model.add(MaxPooling2D(pool_size=(3, 3), strides=(2, 2)))
alex_model.add(BatchNormalization())
```

2nd Convolutional layer with maxpooling

```
alex_model.add(Conv2D(256, (5, 5), padding='same', activation='relu'))
alex_model.add(MaxPooling2D(pool_size=(3, 3), strides=(2, 2)))
alex_model.add(BatchNormalization())
```

3rd Convolutional layer without maxpooling

```
alex_model.add(Conv2D(384, (3, 3), padding='same', activation='relu'))
```

4th Convolutional layer without maxpooling

```
alex_model.add(Conv2D(384, (3, 3), padding='same', activation='relu'))
```

5th Convolutional layer with maxpooling

```
alex_model.add(Conv2D(256, (3, 3), padding='same', activation='relu'))
alex_model.add(MaxPooling2D(pool_size=(3, 3), strides=(2, 2)))
alex_model.add(BatchNormalization())
```

Flattened Layer

```
alex_model.add(Flatten())
```

6th dense layer

```
alex_model.add(Dense(256, activation='relu'))
alex_model.add(Dropout(0.5))
```

7th dense layer

```
alex_model.add(Dense(256, activation='relu'))
alex_model.add(Dropout(0.5))
```

8th dense layer

```
alex_model.add(Dense(7, activation='softmax'))
```

Model Results

	Train	Test
Acc.	0.559	0.593
Loss	1.196	1.300

Table 3. Best accuracy and loss for train and test (validation) data when running AlexNet model using final hyperparameters for 10 epochs and 1 replicate.

	Train	Test
Acc.	0.923	0.725
Loss	0.243	0.875

Table 3. Best accuracy and loss for train and test (validation) data when running AlexNet model using final hyperparameters for 10 epochs and 3 replicates.

Using the final parameters, we ran our CNN with three replicates and 10 epochs each. Using multiple replicates drastically improved the models classification accuracy, as shown above. The best accuracy after 3 replicates was 0.9230. This is promising for future applications, such as a interface or app for users to classify their own snake images. Below are the results for our final model and a graph showing the improvement in accuracy for each for each epoch.

References

Debadri Dutta. Identifying different breeds of snakes, 2020. URL https://www.kaggle.com/datasets/duttadebadri/identifying-different-breeds-of-snakes.

California Academy of Sciences National Geographic Society. inaturalist, 2020. URL https://www.inaturalist.org.

Binod Rimal. *Financial Time-Series Analysis with Deep Neural Networks*. PhD thesis, Florida Atlantic University, 2022.

Contact: Dr. Binod Rimal at brimal@ut.edu