Automatic Snake Classification using Deep Learning Algorithm

Lekshmi Kalinathan[1], Prabavathy Balasundaram[1], Pradeep Ganesh[1], Sandeep Sekhar Bathala[1] and Rahul Kumar Mukesh[1]

[1]*Department of CSE, SSN College of Engineering, Rajiv Gandhi Salai, Chennai, Tamil Nadu, India*

Abstract

Automatic snake classification is the process of identifying snake species using image processing techniques. This system is helpful in reducing the death by snake bites and to suggest appropriate antivenom for the victim in a short span of time. The previous works have built systems with relatively smaller databases using ML and older deep architectures. The systems were capable of identifying only a few snake species and/or had lower accuracies. However, the Snake classification can further be improved in order to make the system more robust. The proposed system is capable of identifying 772 classes of snake species and is built with a relatively larger dataset using newer deep learning architecture ResNeXt50-V2. An ensembled model is used to further improve the system to achieve an accuracy of 85.7% and F1-score of 0.68.

Keywords

Snake Classification, Deep Learning, ResNet-50, ResNeXt, Keras

1. Introduction

Death and amputation caused by snake bites is major cause of concern in health care institutions. There are approximately 1.8 to 2.7 million cases of envenoming each year of which 435,000 to 580,000 snake bites need treatment, for they can cause permanent disability and disfigurement.

Although only about 20% of snake species worldwide are medically important, identifying the biting snake is challenging, especially due to:

- The high diversity of snake species in snakebite endemic countries (e.g. 310 snake species in India)
- The limited herpetological knowledge of communities and healthcare providers confronted with snakebite
- Incomplete knowledge of their epidemiological importance

Often people who are bitten by snakes, are unable to get the required treatment immediately as they are ineffectual in neither identifying the snake nor giving adequate information about

CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania
lekshmik@ssn.edu.in (L. Kalinathan); prabavathyb@ssn.edu.in (P. Balasundaram); pradeep19077@cse.ssn.edu.in (P. Ganesh); sandeepsekhar19096@cse.ssn.edu.in (S. S. Bathala); rahulkumar19086@cse.ssn.edu.in (R. K. Mukesh)
0000-0003-3903-0410 (P. Balasundaram)

the snake that bit them. This leads to the physician to guess the type of snake involved in the situation. If the identification of snake determined by the medical practitioner based on the clues given is wrong, it will lead to further complications in the case, and, unfortunately can lead to the death of the patient. This is one of the reasons that decreases the survival rate of patients of snake bite cases.

Hence, if we are able to identify a snake using pictures taken in low resolution, it would improve the survival rate of the patients. Moreover, the system will also be useful for conservation of wildlife as it can be used to identify endangered species of snakes.

When snake species identification is automated, it could ameliorate public health as there are only a few initiatives that seek to identify snakes using computer vision techniques. So far only a handful of computer vision and machine learning algorithms specific to snakes have been developed. The best existing model as of 2020 achieved an F1-score of 0.625 using a pre-trained model, with Vanilla ResNet50-V2 architecture. However, none of these are yet usable in real-world situations where lives may be at stake. Hence, creating an automatic and robust system for snake species identification is an important goal for biodiversity, conservation, and global health.

2. Related Work

Alex James et al. [1] presented a parallel processed inter-feature product similarity fusion based automatic classification of kinds of snakes such as Spectacled Cobra, Russel's Viper and King Cobra to name a few. The authors used a database of 88 images of cobra and 48 images of viper for the initial feature taxonomy analysis and identified 31 different taxonomically relevant features from snake images for automated snake classification studies. In this paper, the authors use the nearest-neighbour classifier. This classifier identifies the class of unknown data sample from its nearest neighbour, whose class is already known. For automatic classification, the taxonomically relevant features are selected from the snake images and are normalized using mean-variance filtering. The histograms of gradients and orientation of these normalized features are used as feature vectors. These feature vectors are evaluated using a proposed minimum distance product similarity metric classifier. The proposed system was able to achieve an F-score of 0.91 when 5% of the class samples are used as gallery and remaining 95% of sample are used as test on snake image database. The authors analysed the scalability and real-time implementation of the classifier through GPU enabled parallel computing environment. The developed systems found their application in wild life studies, analysis of snake bites and in management of snake population.

Alex Pappachen James et al. [2] in his paper presented the automatic snake identification problem by developing a taxonomy-based feature, targeted for use by computer scientists and herpetologists. The feature database contained 38 taxonomically relevant features of each sample. Out of these 38 features, top features that have highest impact on classification were determined. In order to find the top features from the complete database, twelve attribute evaluators (ChiSquared AttributeEval and CfsSubsetEval to name a few) were used. Along with this, a combination of certain search methods like Genetic Search, Greedy Step-wise and Linear Forward Selection was used. The feature-subset analysis on the dataset indicated that

only 15 features are sufficient for snake identification. It was found that these features were almost equally distributed from the logical grouping of top, side and body views of snake images. Features from the bottom view of snakes had the least role in the snake identification. In order to perform automated snake classification, 13 classifiers (Bayes Net, Naïve Bayes and Multilayer perception to name a few) were used. Using these classifiers, the best F-score obtained was about 0.94.

Louise Bloch et al. [3] implemented a machine learning workflow that uses Mask Region-based Convolutional Neural Network (Mask R-CNN) for object detection, and EfficientNets for classification. The best model submitted by them in the SnakeCLEF 2020 [4] challenge achieved an F1-score of 0.404. After the expiration of this deadline, it was found that it could be improved to achieve an F1-score of 0.594. The main improvements in snake species classification presented by them were based on increasing the image size, combining location and image information as well as upscaled model architecture.

Moorthy Gokula Krishnan [5] wrote a report on the improvement of F1-score using pre-trained network on the large dataset for pre training. Vanilla ResNet50-V2 was used for pretraining of the database and comparison was drawn between F1-scores of the model which used pretrained network and model which did not use pre trained network. The conclusion of the comparison was that model with pretrained network had performed better, the F1-score improved from 0.5813 to 0.6018. The test dataset on which the final scores were calculated, follows a data distribution similar to validation dataset and the model achieves an F1-score of 0.625 when tested on AICrowd platform [6].

Patel A et al. [7] used deep learning methods to develop an application for smartphones which distinguishes images of 9 different snake species that dwell on the Galápagos Islands in Ecuador. In order to achieve this, object detection as well as classification algorithms have been used. The images from the dataset were collected from Tropical Herping collection of image and Web scraping images from Google and Flickr sites. Different combinations of architecture models such as Faster R-CNN, Inception V2, ResNet, MobileNet, and VGG16 have been tested for object detection and image classification. The model which was based on Faster R-CNN with ResNet achieved the best classification accuracy of 75 %.

3. Methodology

This section discusses about the dataset and base methodology utilized for implementing snake classification task.

3.1. Dataset

The data set provided by SnakeCLEF2021[8, 9] - Snake species Identification Challenge consists of a total of 412,537 images. This dataset contains images of 772 different snake species from 188 countries. The majority of data was gathered from online biodiversity platforms such as iNaturalist and HerpMapper. It was further extended by data scraped from Flickr and images collected from private collections and museums. The final dataset has a heavy long-tailed class distribution, where the most frequent species *(Thamnophis sirtalis)* was represented by 22,163 images and the least frequent by just 10 images *(Achalinus formosanus)*. The final data set was

split into training and validation sets consisting of 347,405 and 38,601 images respectively. Both subsets have the same class distribution, while the minimum number of validation images per class is one. A set of 26,531 images containing all 772 classes with a similar class distribution was used as the testing set. These images are associated with metadata that provides information about the continent and country of the place where the image has been taken. For some snake depictions, the information is not given and only "UNKNOWN" is provided in the metadata.

3.2. Deep Network Architecture used

A baseline model of Convolutional Neural Network (CNN) can be scaled up in hopes of achieving better performance and to solve complex problems. However, it has been found that adding more layers in the network potentially degrades the performance. This may be due to optimization function, initialization of the network and more specifically vanishing gradient problem. This performance degradation has been addressed by ResNet architecture [10].

The core idea of ResNet is residual network block. This block consists of skip connection which skips some layers in the neural network by feeding the output of one layer to the subsequent layer not necessarily the adjacent layer. By using a skip connection, an alternative path is provided for the gradient. This helps us in avoiding the vanishing gradient problem.

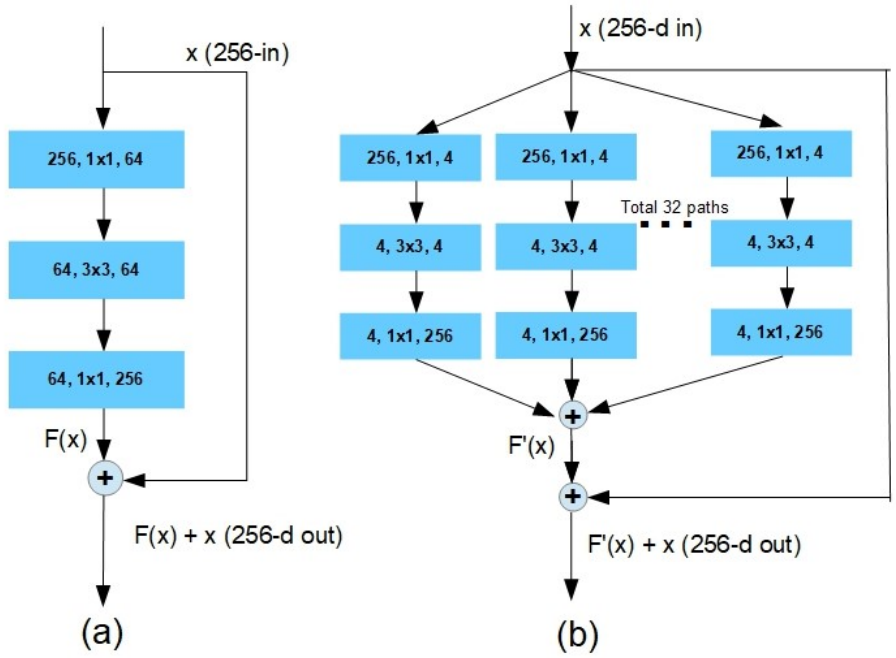


Figure 1: ResNet and ResNeXt Network Block

To improve the effectiveness of ResNet architecture without increasing the number of parameters, ResNeXt [11] architecture was proposed. In ResNeXt architecture, each residual network block shown in Figure 1 (a) is split into n number of paths. Number of paths inside

the ResNeXt block is defined as cardinality. In Figure 1 (b), the cardinality is 32. All the paths contain the same topology. Both the ResNet and ResNeXt network blocks have different width. Layer-1 in ResNet has one convolution layer with 64 width, while layer-1 in ResNeXt has 32 different convolution layers with 4 width (32*4 width). Validation error will be decreased with the increase of cardinality. Hence, the performance of the network will be improved.

4. Implementation

The proposed Automatic Snake Classification system was developed by following the steps:

- Preprocessing the data
- Batch Accumulation
- Building the model
- Ensemble Model for testing

The automatic snake classification system was developed using ResNeXt50-V2 model with Keras framework [12]. This model was trained using a machine with Intel Xeon Processor W-2145, 11M cache, 3.70GHz, 2*16GB DDR4 - 2666Mhz ECC REG DIMM 2TB SATA 7.2K RPM 3.5” HDDs, 240 GB SATA SSD 2.5" HDDs and NVIDIA GeForce RTX2080 Ti, 11 GB.

4.1. Preprocessing the data

Table 1

Details of SnakeClef 2021 dataset

Subset	# of images	% of data	min. # of images/class
Training data	347,405	84.21%	9
Validation data	38,601	9.36%	1
Test data	26,531	6.43%	1

In our implementation, as per the number of images given in Table 1, training data is used to train the model. Validation data is used to test the model obtained during every epoch to fine tune the weight parameters. Test data is used to test the final model obtained after the entire training process.

The images provided by the SnakeClef2021 dataset are of varied sizes. The training images are first resized to $224 \times 224 \times 3$ dimensions using a random crop resized operation. The images were also horizontally and vertically flipped with a probability of 0.5. Further, the images were scaled and rotated with a probability of 0.5. The images were also normalized with a standard deviation and mean of 0.23 and 0.5. Similarly, the images of validation and the testing dataset were resized to $224 \times 224 \times 3$ dimensions using resize function. The images were also normalized with a standard deviation and mean of 0.23 and 0.5 respectively. Both the training and testing images were augmented using the *Albumentations* [13] python library with the parameters of augmentation method such as *RandomResizedCrop, Transpose, Resize, HorizontalFlip. VerticalFlip, ShiftScaleRotate and Normalize.*

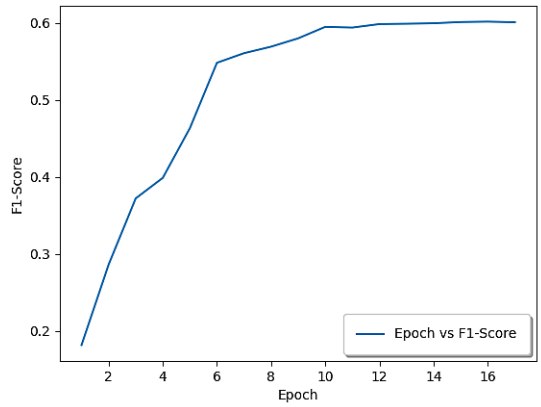
4.2. Batch Accumulation

Table 2

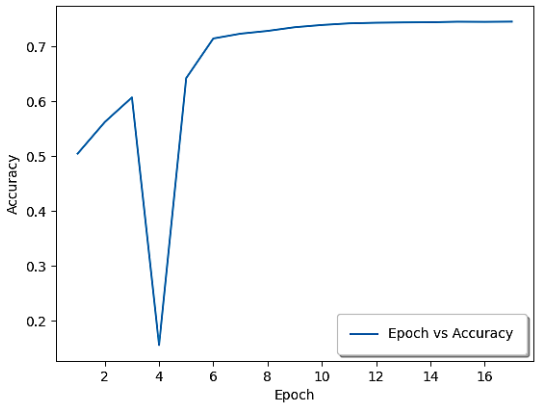
Model Summary of ResNeXt50-V2 Architecture

Conv1	L1 - [224, 7 x 7, 64]		
Conv2	**3 Stacked Residual Blocks**		
	L1- [64, 1×1, 128] L2- [128, 3×3, 128] L3- [128, 1×1, 256]	L1- [256, 1×1, 128] L2- [128, 3×3, 128] L3- [128, 1×1, 256]	L1- [256, 1×1, 128] L2- [128, 3×3, 128] L3- [128, 1×1, 256]
Conv3	**4 Stacked Residual Blocks**		
	L1- [256, 1×1, 256] L2- [256, 3×3, 256] L3- [256, 1×1, 512] Downsampling L0 -[256, 1×1, 512]	L1- [512, 1×1, 256] L2- [256, 3×3, 256] L3- [256, 1×1, 512]	
	L1- [512, 1×1, 256] L2- [256, 3×3, 256] L3- [256, 1×1, 512]	L1- [512, 1×1, 256] L2- [256, 3×3, 256] L3- [256, 1×1, 512]	
Conv4	**6 Stacked Residual Blocks**		
	L1- [512, 1×1, 512] L2- [512, 3×3, 512] L3- [512, 1×1, 1024] Downsampling L0 -[512, 1×1, 1024]	L1- [1024, 1×1, 512] L2- [512, 3×3, 512] L3- [512, 1×1, 1024]	L1- [1024, 1×1, 512] L2- [512, 3×3, 512] L3- [512, 1×1, 1024]
	L1- [1024, 1×1, 512] L2- [512, 3×3, 512] L3- [512, 1×1, 1024]	L1- [1024, 1×1, 512] L2- [512, 3×3, 512] L3- [512, 1×1, 1024]	L1- [1024, 1×1, 512] L2- [512, 3×3, 512] L3- [512, 1×1, 1024]
Conv5	**3 Stacked Residual Blocks**		
	L1- [1024, 1×1, 1024] L2- [1024, 3×3, 1024] L3- [1024, 1×1, 1024] Downsampling L0 -[1024, 1×1, 2048]	L1- [2048, 1×1, 1024] L2- [1024, 3×3, 1024] L3- [1024, 1×1, 2048]	L1- [2048, 1×1, 1024] L2- [1024, 3×3, 1024] L3- [1024, 1×1, 2048]
Softmax	L1 – [2048, - , 772]		

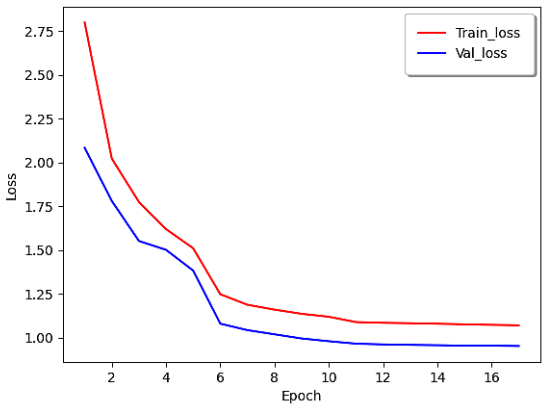
Training data is split into 3860 batches where each batch consists of 5 folds or mini-batches. Training will be inefficient due to noisy gradients, if the mini-batch is small and consists of a set of images without the inclusion of all the classes. Hence, the images in mini-batch have been chosen in such a way that it covers all the classes. A stratified k-fold cross-validation where k = 5, has been adapted to enforce the class distribution in each split of the data to match the distribution in the complete training dataset. Cosine annealing with warm restarts scheduler has been used to update the learning rate. Learning rate of 1e-4 and Adam optimizer [14] were used to fine-tune the weight matrices of the architecture.



(a) Epoch vs F1-Score



(b) Epoch vs Accuracy



(c) Epoch vs Loss

Figure 2: Performance Measures of ResNeXt50-V2 Architecture

4.3. Building the model

Architecture of the ResNext model consists of *Conv1* layer with 64 filters of each 7×7. The output of *Conv1* which is 64 feature maps were the input for 3 stacked *Conv2* block. The output of *Conv2* which is 256 feature maps were the input for 4 stacked *Conv3* block. The output of *Conv3* which is 512 feature maps were the input for 6 stacked *Conv4* block. The output of *Conv4* which is 1024 feature maps were the input for 3 stacked *Conv5* block. The output of *Conv5* which is 2048 feature maps were the input for soft max layer whose output is 772 classes. Model summary of the implemented ResNeXt50-V2 is shown in Table 2. The following hyper parameters were used to train the model:

- The input image size was kept as $224 \times 224 \times 3$
- Image Augmentations used during training process were as follows:

RandomResizedCrop, Transpose, Resize, HorizontalFlip. VerticalFlip, ShiftScaleRotate and Normalize methods from Albumentations Python library
- The model was trained for 17 epochs
- CrossEntropyLoss with default parameters
- Learning rate of 1e-4 with "CosineAnnealingWarmRestarts" scheduler (T_max=10, eta_min=1e-6, last_epoch=-1)
- Adam optimizer was used

5. Results and Analysis

This section discusses the analysis of training and testing processes through various performance measures.

5.1. Analysis of training process

The training process was analysed in order to determine whether the model is stable and free from overfitting.

During the training process, F1-Score and training accuracy were measured for every epoch and plotted as shown in Figure 2a and 2b respectively. It is inferred from Figure 2a that the F1-Score is increasing right from the very first epoch and get stabilized to 0.6008 from 15^{th} *epoch.* From Figure 2b, it is inferred that the training accuracy is stabilized from 11^{th} *epoch.* Similarly, the training and validation losses were also measured for every epoch and plotted as shown in Figure 2c. It is clearly found from the graph that the validation loss for every epoch is lesser when compared to the training loss. From this it is inferred that the *generated model* is free from overfitting.

5.2. Analysis of testing process

Table 3

Performance measures of different models

ResNeXt50-V2 Models	F1-Country Score	F1-Score	Accuracy
Model-13	0.42	0.38	0.66
Model-14	0.51	0.40	0.69
Model-15	0.44	0.40	0.68
Model-16	0.40	0.38	0.68
Model-17	0.37	0.37	0.68
Ensembled Model	0.67	0.68	0.86

During the testing process, instead of generating predictions with the recent model, our proposed ensemble model has been utilized. In order for ensembling, the models corresponding to the last 5 epochs i.e models 13, 14, 15, 16 and 17 were considered. These models were chosen to be the top models as the accuracy and F1-Score were stabilized during the corresponding epochs. In the process of ensembling, the predictions generated from every model are averaged.

Experiments were conducted to generate the predictions from every individual and ensembled model as shown in Table 3. Models 13, 14, 15, 16 and 17 generate different predictions for each image due to the fact that they have different weight matrices. Each model may have specialised in predictions for distinct set of images. In the ensembled model, the average of the predictions generated from all the models in consideration is taken. This increases the accuracy of the model manifold. The same can be deduced from the table. The ensembled model generated an *Accuracy* and *F1-Country Score* of 85.77% and 0.724 respectively.

6. Conclusions and Future Work

In conclusion, it can be stated that snake species identification is a challenging task, primarily because of the high diversity of snake species and the data set having a heavy long-tailed class distribution. Automatic classification has been built using ResNeXt50-V2 Architecture and the model achieved an F1-country score of 0.724. In future, accuracy of the system can be enhanced through preprocessing of the images. Subsequently, alternate deep architectures can be implemented and tested for better accuracies.

7. Acknowledgement

We thank the CSE department of SSN College of Engineering for letting us utilize the GPU server machine extensively to implement this task.

References

[1] A. James, Snake classification from images, PeerJ Preprints 5 (2017).

[2] A. P. James, B. Mathews, S. Sugathan, D. K. Raveendran, Discriminative histogram taxonomy features for snake species identification, Human-Centric Computing and Information Sciences 4 (2014).

[3] L. Bloch, A. Boketta, C. Keibel, E. Mense, A. Michailutschenko, O. Pelka, J. Rückert, L. Willemeit, C. Friedrich, Combination of image and location information for snake species identification using object detection and efficientnets, in: CLEF working notes, 2020.

[4] L. Picek, R. Ruiz De Castaneda, A. M. Durso, P. Sharada, Overview of the snakeclef 2020: Automatic snake species identification challenge, in: Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, 2020.

[5] M. G. Krishnan, Impact of pretrained networks for snake species classification., in: Working Notes of CLEF 2020 - Conference and Labs of the Evaluation Forum, 2020.

[6] A. M. Durso, G. K. Moorthy, S. P. Mohanty, I. Bolon, M. Salathé, R. Ruiz De Castañeda, Supervised learning computer vision benchmark for snake species identification from photographs: Implications for herpetology and global health, Frontiers in Artificial Intelligence 4 (2021) 17.

[7] A. Patel, L. Cheung, N. Khatod, I. Matijosaitiene, A. Arteaga, J. W. Gilkey, Revealing the unknown: real-time recognition of galápagos snake species using deep learning, Animals 10 (2020) 806.
[8] L. Picek, A. M. Durso, R. Ruiz De Castañeda, I. Bolon, Overview of snakeclef 2021: Automatic snake species identification with country-level focus, in: Working Notes of CLEF 2021 - Conference and Labs of the Evaluation Forum, 2021.
[9] A. Joly, H. Goëau, S. Kahl, L. Picek, T. Lorieul, E. Cole, B. Deneu, M. Servajean, R. Ruiz De Castañeda, I. Bolon, H. Glotin, R. Planqué, W.-P. Vellinga, A. Dorso, H. Klinck, T. Denton, I. Eggel, P. Bonnet, H. Müller, Overview of lifeclef 2021: a system-oriented evaluation of automated species identification and species distribution prediction, in: Proceedings of the Twelfth International Conference of the CLEF Association (CLEF 2021), 2021.
[10] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 31, 2017.
[11] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[12] A. Gulli, S. Pal, Deep learning with Keras, Packt Publishing Ltd, 2017.
[13] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, A. A. Kalinin, Albumentations: fast and flexible image augmentations, Information 11 (2020) 125.
[14] D. Kingma, J. Ba, Adam: A method for stochastic optimization, in: International Conference on Learning Representations, 2014.