SSRG International Journal of Computer Science and Engineering
ISSN: 2348 – 8387 /doi:10.14445/23488387/IJCSE-V8I5P102
Volume 8 Issue 5, 12-16, May 2021

Automatically Identifying Wild Animals In Camera-Trap Images With Deep Learning

Kalletla Sunitha[1]
Assistant Professor, Computer Science, and Engineering Department
Mahatma Gandhi Institute of Technology(Affiliated to JNTUH),
Gandipet, Hyderabad, TS-500075. India.

Received Date: 12 April 2021
Revised Date: 19 May 2021
Accepted Date: 25 May 2021

Abstract - *Human vision will comprehend and examine the pictures effectively than automatic investigation by the framework. To conquer this issue in the existing order framework, a few examinations have been done, but the output has been given uniquely for low-level picture natives. Be that as it may, the existing methodology needs the exact arrangement of pictures. AI empowers the examination of enormous amounts of information. The principle point of the AI is to cause the PCs to adapt naturally without human intercession or help and change their activities appropriately. Motion- sensor "camera traps" gather fauna pictures reasonably and furthermore regularly. In any case, the data extricated from these photos will be a costly, tedious, manual errand. In this paper, we show that such data can be consequently removed by deep learning, which is a cutting-edge kind of artificial intelligence. Convolutional neural networks(CNN) are utilized for this reason.*

The model for image classification utilizing CNN is developed. This model is trained with a dataset of various creature pictures to characterize pictures. The Relu activation function is utilized for fault distinguishing proof and adjustment. The trained model is tried with various datasets of creature pictures. This model yields the creature pictures with the label of the creature's name.

Keywords*: Camera Traps, Convolutional Neural Networks(CNN), Deep learning, image classification, wildlife.*

I. INTRODUCTION

"Spy cameras" or "camera traps" are being used as an increasingly popular tool for monitoring wildlife monitoring and it is effective and reliable in collecting wildlife information constantly and enormous in volume. Nonetheless, it is a costly and tedious manual errand to extricate data from these photos. We show that such data can be consequently separated by deep learning, which is a cutting-edge type of artificial intelligence. To tackle this issue, the convolutional neural network is the most reasonable methodology. In this research, we collect the datasets of animals. By converting this data into knowledge, we train deep convolutional neural networks to classify the animal images.

There are several models available to classify images using image processing and support vector machines. Most of the previous models used datasets in which only the front view of the animals was considered. They classified images based on the front view of animals only. Even though those models are highly accurate, they may not work in a real-time environment because the camera may capture animals from side view also.

The proposed model is trained using convolutional neural networks, which is one of the best algorithms for image classification. The model is trained with a large dataset(25000 images for training and 12500 images for testing) which leads to improving the accuracy of the model. The dataset consists of data of animal images by considering different views(i.e., side, top, back) of the animal. The model is developed such that it shows the name of the animal in the image. Now, we have trained the model with two animal classes(i.e., cat and dog) to check the performance of the model.

II. RELATED WORK

A. Shahram Taheri ; Önsen Toygar(26 July 2018)

In this paper, they have developed various computer vision approaches for animal classification. A strategy called the score-level fusion of convolutional neural network (CNN) features and appearance-based descriptor features are used. In this strategy, a score-level fusion with two unique methodologies is used. One methodology utilizes CNN, which can automatically extract features, learn them and classify them, and another methodology utilizes kernel Fisher Analysis (KFA) for its feature extraction phase. The results of the experiments shown that automatic feature extraction in CNN is better than other simple feature extraction techniques (both local- and appearance-based features), and also the appropriate score-level combination of CNN and simple features can achieve even higher accuracy than applying CNN alone. The score-level fusion of CNN extracted features and the appearance-based KFA method has a positive effect on classification accuracy.[1]

B. Hung Nguyen, Thin Nguyen, Sarah J. Maclagan, Tu Dinh Nguyen, Paul Flemons, and Kylie Andrews(18 January 2018)

In this paper, they have shown how to train a computational framework capable of separating creature pictures and recognizing species, consequently utilizing a single-labeled dataset from the Wildlife Spotter project, which is finished by resident researchers, and the state-of-the-art deep CNN architectures. Their experimental results shown by them are having an accuracy at 96.6% for the errand of recognizing pictures containing creatures and 90.4% for distinguishing the three most basic species among the set of images of wild animals taken in Southcentral Victoria, Australia, demonstrating the feasibility of building fully-automated wildlife observation. These outcomes have accelerated research discoveries, built more productive resident science-based checking frameworks and ensuing administration choices, and had huge effects on the universe of environment and trap camera pictures analysis.[2]

C. N Manohar, Y H Sharath Kumar, G Hemantha Kumar (03 November 2016)

In this paper, Supervised and Unsupervised techniques are contrasted with arranging the creatures. At first, Maximal Region Merging Segmentation algorithm is used to segment animal images. At that point, the Gabor features are extracted from these segmented images. Further, Supervised and Unsupervised techniques are utilized to decreased extricated features. In the supervised method, a Linear Discriminate Analysis (LDA) dimension reduction technique is used to reduce the features. In the unsupervised method, a Principle component analysis (PCA) dimension reduction technique is used to reduce the features. The reduced features are then fed into the K-means algorithm for the purpose of grouping. A dataset of 2000 animal images consisting of 20 different categories of animals with various percentages of training samples is used for experimentation. In this model, it is observed that a supervised classification system performs better compared to an unsupervised method.[3]

D. Hayder Yousif, Roland Kays, Jianhe Yuan, Zhihai He(28 September 2018)

In this paper, a method for human-animal identification utilizing joint background modeling and deep learning classification is created. An effective background modeling and deduction plan to produce locale recommendations for the closer view objects are explicitly evolved. Complexity-accuracy analysis of deep convolutional neural networks (DCNN) is performed to build up a quick deep learning classification method to classify these region proposals into three categories which are human, animals, and background patches. They have tracked down that the upgraded DCNN was able to maintain a high level of accuracy while reducing the computational complexity by 14 times.[4]

III. METHODOLOGY

A. System Architecture

System architecture describes the steps in constructing a model for animal image classification. As shown in figure 3.1, the first data is retrieved from datasets. The data has to be prepared for training. So, the data is labeled and converted into Gray Scale images. And the images are reshaped. Now, the CNN model is trained with prepared data. The model is validated and tuned to improve the classification accuracy model. Finally, the model is tested with another dataset to know whether the model is classifying images correctly or not.

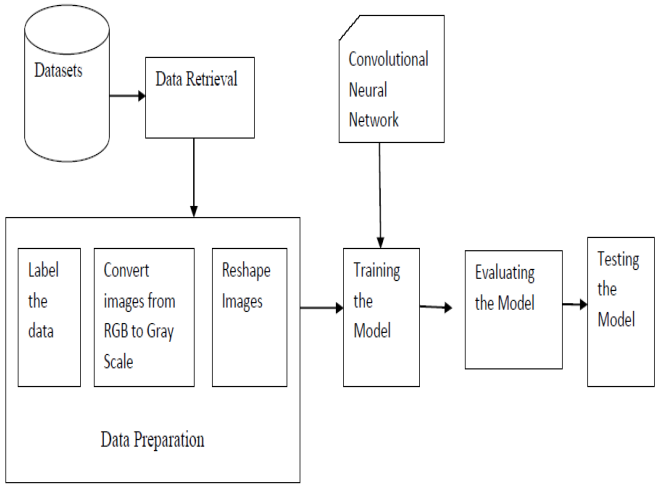


Figure 1: System Architecture

B. CNN Architecture

CNN is a Deep Learning algorithm; it takes an input image, assigns some importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing which is required in a CNN is much lower when compared to other classification algorithms. To be more specific, the image is passed through a series of convolutional, nonlinear, pooling layers and fully connected layers, and then it generates the output as shown in figure2 below.

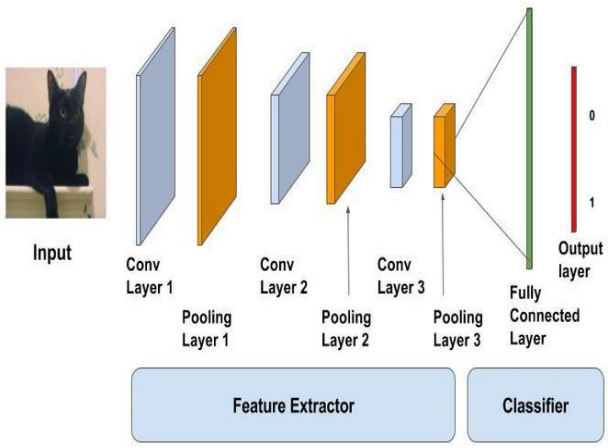


Figure 2: CNN Architecture

a) Convolutional Layers:

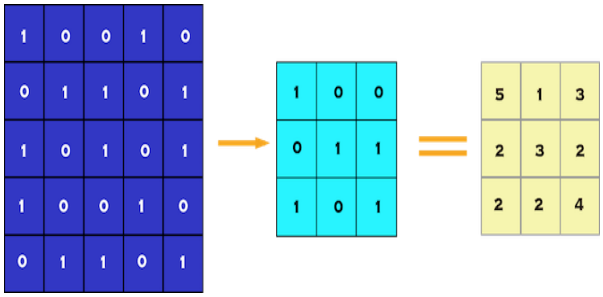


Figure 3: Convolution of 5X5 pixel image with 3X3 pixel filter (stride=1X1 pixel)

The above figure describes the convolutional layer. It is the very first layer where the features from the images are extracted in our datasets. The pixels are just related with their neighboring and close pixels; convolution permits to save connection between various pieces of a picture. Convolution is fundamentally used to filter the picture with a more modest pixel filter to diminish the size of the picture without losing the connection between pixels, at the point when we apply convolution to 5X5 picture by utilizing a 3X3 filter with 1X1 stride (1 pixel move at each progression). The yield is 3X3 (which is 64% diminishing in intricacy).

b) Non-linear Layers:

This is added after every convolution activity. It carries nonlinear property with an activation function. A network would not be sufficiently intense and will not be able to model the response variable (as a class label) without this property.

c) Pooling Layer:

While building CNNs, it is basic to embed pooling layers after every convolution layer to decrease the size of the spatial of the representation to reduce the counts of the parameter, which diminishes the computational complexity. What's more, the overfitting issue is settled with pooling layers. Essentially, the maximum, average, or sum values are chosen inside these pixels to lessen the parameters by choosing a pooling size. One of the common pooling techniques called Max Pooling is demonstrated as follows:

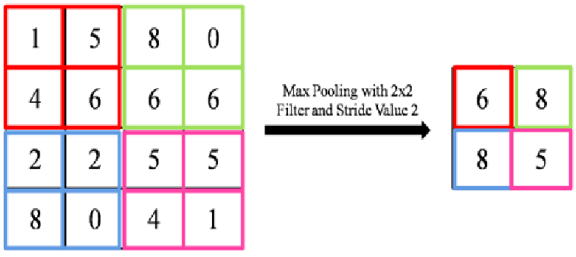


Figure 4: Max Pooling by 2X2

The above figure describes the max-pooling technique in the pooling layer of convolutional neural networks.

d) Set of Fully Connected Layers:

It is important to append a fully connected layer after completing a series of convolutional, nonlinear, and pooling layers. A fully connected layer takes the output information from convolutional networks. The network brings about an N-dimensional vector by attaching a fully connected layer to the end of the network, where N is equivalent to the number of classes that choose the desired class from the model.

III. IMPLEMENTATION

A. Importing Train data and Test data:

Datasets that are used in training and testing consist of 24,000 and 12,500 images, respectively. In the training dataset, each image consists of a label. The testing dataset does not contain labels. Since the datasets are too large, they are converted into a zip file. They are mounted at drive. Now, the path is created for the datasets. We can import datasets From drive.

B. Labeling the data:

By default, our dataset comes with a Label of "cat" or "dog," but we can't feed in string or characters into our Neural Network. So, we have to convert them into *vectors*, and we can do it by the following code. If the label is "cat," then the code returns [1,0] vector. If the label is "dog," then the code returns the vector[0,1].

C. Creating Testing and Training sets:

We have images of cats and dogs, but we cannot feed- in those images directly. So, we have to convert them into matrixes. Images are the combination of RBG(Red, Green, and Blue). So basically, there are three color channels that we have to feed into our ConvNet, we can do that, but it is computationally too costly (i.e., too many complex computing processes that might require high-performance GPU and kinds of stuff). But for this particular task, we may do things in a simpler and efficient manner. So we are changing RGB images into Gray Scale images with the help of OpenCV, and it consists of only one color channel. As we have done Gray scaling to our image, we have to reshape the image into 50 by 50 images, also for lowering our computational expenses, and then we are appending it to our list.

D. Construction of CNN Model:

Now, the task is to build a convolutional neural network. We have used seven layers to build convolutional neural networks. We have declared the input function that takes our format of the feed-in data, and then we have declared five convolution layers with each varying inputs units, and we have used Relu as an activation function. At layer six, for classification, a convolution layer is converted into a fully connected layer. At layer seven, we have added the Softmax function.

a) RELU(Rectified Linear Unit):

It is an activation function. It gives input value for all positive inputs and gives zero for negative inputs. It is

used here to make all negative values as zero when we encounter any negative values in image pixels so that the model can take less time to train or run.

$$y=\max\{0,x\} \quad \text{-------- (1)}$$

The above equation (1) depicts the relu activation function. Here, y is output and x input. Here, x takes pixel values. If x is less than zero, then the y value is zero; otherwise, the y value equals x.

b) SOFTMAX:

It is an activation function that is often placed as the output layer of a neural network. It is commonly used in multi-class learning problems where a set of features can be related to one-of-K classes. It is also used to compute the normalized exponential function for all the units in the layer.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K)$$

----------(2)

The above equation(2) depicts the softmax function. Here, K is a number of classes, and zi is output from the previous layer for respective class K. Intuitively, what the softmax does is that it squashes a vector of size K between 0 and 1. Furthermore, because it is a normalization of the exponential, the sum of this whole vector equates to 1. We can then interpret the output of the softmax as the probability that a certain set of features belongs to a certain class.

c) Loss and Accuracy:

After the model parameters are learned and fixed, the accuracy of the model is determined. After the comparison is on true targets, then the number of mistakes (0-1 loss) the model makes are recorded by feeding the test samples, after compared is on to the true targets.

The percentage of misclassification is calculated. Here, we are using cross-entropy measures to calculate the loss.

$$\text{Loss}=-(y\log(p)+(1-y)\log(1-p)) \quad \text{---------- (3)}$$

The above equation (3) is used to calculate the loss of the model. Here, y is a binary indicator, i.e., 0 or 1. Here, y is 1 if the class label is the correct classification for a given observation. Otherwise, y is 0. In the above equation, p is the predicted probability observation of class.

Accuracy = (Correctly classified samples / total number of samples)X 100 ---------- (4)

The above equation (4) is used to calculate the accuracy of the model. If the number of test samples is 1000 and model 952 classifies correctly, then the model's accuracy is 95.2%.

d) Splitting test and train data:

It is important to split the data into training and validating set for training and validating our model. We are converting it into a numpy array for convenience and storing in x and y variables for actual data and classes.

e) Training CNN Model:

Now, we fit our data into the convolutional neural network model that we have built previously. We are saying our model to train for *10 epoch* and with validation sets of test_x and test_y variables. In this module, the CNN model improves its accuracy by training the model for 10 epochs.

IV. RESULTS

A. Accuracy of model:

After training the CNN model, the model is ready to test. When we run the model, it gives the accuracy of the model by validating the model. We have obtained 91% accuracy during validation.

Figure 5. Accuracy of CNN model

Above figure 5 shows the accuracy obtained during the validation of the convolutional neural network model. In figure 4, the 'loss' parameter depicts the loss of the model, and the 'acc' parameter depicts the accuracy of the model. Here, loss and accuracy are calculated using formulae given in equation 1 and equation 2, respectively.

B. Testing the model:

For testing the model, another dataset that is not used in training is taken. After running the testing code block, it creates a numpy array in which the label of each image in the testing dataset is stored after the classification of that image.

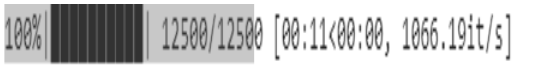


Figure 6: Testing the model.

The above figure6 shows that the CNN model is tested with 12500 images, and the labels of those images are stored on numpy arrays.

a) Testing the model with correct data

Now, the CNN model is tested with animal images. The images are correctly classified. The label of the image is shown on the top of the image. The labels of the images are correct.

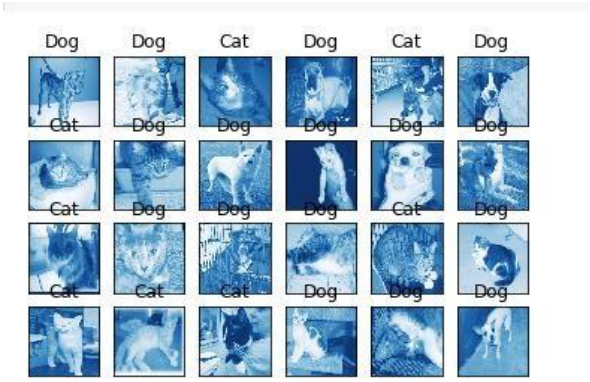
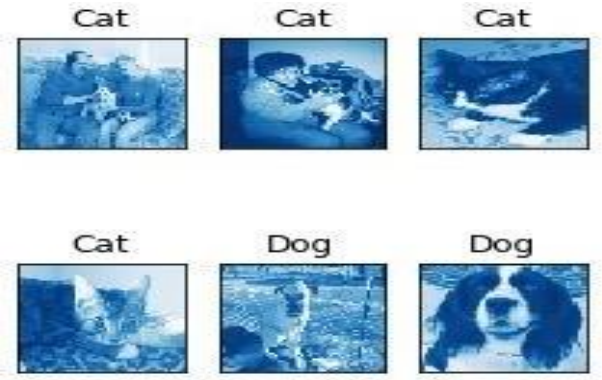


Figure 7:Output of the model with correct data

b) Testing the model with multi-variant images

Multi variant images are images that consist not only



of cat or dog classes but also other classes like humans. When the model is tested with these images, it is classifying the images correctly. It is labeling the images correctly.

Figure 8: Output of model with multi-variant data

The above figure shows the correct output of the model when it is tested with multi-variant data.

V. CONCLUSIONS

The main objective of this research of "Image Classification Using Convolutional Neural Network" is to classify the animal images considering all views (i.e., side, top, back), which takes a lot of time for manual classification. The proposed convolutional neural network architecture classified the images at an accuracy of 91%. When we test the model with another dataset, it gives an accurate label of the image. When the model is tested with the multi-variant dataset, it is giving correct results. Since the model is coded in Google colab using python, it takes less time for training and validating the model. Now, the proposed architecture classified only a single animal image. In the future, we can create a model which classifies multiple animals in a single image. We can take a real-time dataset from the covert cameras and validate the model. The dataset consists of not only images but also other things like trees, ground, water, etc. By considering this dataset, we have to develop a model which differentiates between animals and other things. And also, it has to classify all animals in that image.

References

[1] Shahram Taheri, Önsen Toygar, Animal classification using facial images with score-level fusion, 26 July 2018.

[2] Hung Nguyen, Sarah J. Maclagan, Tu Dinh Nguyen, Thin Nguyen, Paul Flemons, Kylie Andrews. Animal Recognition with Deep Convolutional Neural Networks for Automated Wildlife Monitoring, 18 January 2018.

[3] N Manohar, Y H Sharath Kumar, G Hemantha Kumar, Supervised and unsupervised learning in animal classification, 03 November 2016.

[4] Hayder Yousif, Jianhe Yuan, Roland Kays, Zhihai He, Fast human-animal detection from highly cluttered camera-trap images using joint background modeling and deep learning classification, 28 September 2018.

[5] Tom Mitchell, Machine Learning, Mc Graw Hill.

[6] Akshaya B, Kala M T, Convolutional Neural Network Based Image Classification And New Class Detection, Power Instrumentation Control and Computing (PICC) 2020 International Conference on, (2020) 1-6.

[7] Maissa Hamouda, Karim Saheb Ettabaa, Med Salim Bouhlel, Image and Signal Processing, vol. 10884, pp. 310, 2018.

[8] Y. Bengio, Learning deep architectures for ai, Foundations and Trends in Machine Learning, (2009) 1-127.

[9] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proceedings of Artificial Intelligence and Statistics (AISTATS), 2010.

[10] A. Krizhevsky, I. Sutskever, and G. Hinton, Imagenet classification with deep convolutional neural networks, Neural Information Processing Systems 25, 2012.

[11] Daichi Ozaki, Hiroshi Yamamoto, Eiji Utsunomiya, Kiyohito Yoshihara, Harmful Animals Detection System Utilizing Cooperative Actuation of Multiple Sensing Devices, Information Networking (ICOIN) 2021 International Conference on, (2021) 808-813.

[12] S Madhava Prabhu, Seema Verma, A Comprehensive Survey on Implementation of Image Processing Algorithms using FPGA, Recent Advances and Innovations in Engineering (ICRAIE) 2020 5th IEEE International Conference on, (2020) 1-6.

[13] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint, 2014.