Dog Breed identification using CNN Architecture

Dr. Kanika Gupta[1], Sharankhala Goel[2], Bhavya Aggarwal[3], Riya Gupta[4]

[1] *Assistant Professor, Dept. of Information Technology, ABES Engineering College, Ghaziabad, Uttar Pradesh*

[2, 3, 4] *IT Graduate Students, ABES Engineering College, Ghaziabad*

ABSTRACT: In this forecourt competition, we are provided a stringently mongrel division of Image Net in order to exercise fine-grained image cataloguing. The dataset contains images of dogs of different breeds. Deep Learning is a technique by which a computer program learns statistical patterns within the data that will enable it to recognize or help us to distinguish between the different breeds of dogs. The model trains itself on the different features based on the images present and represent the data numerically, organizing the data in space. Initially, the image is divided into numerous lattices and a training batch size is set accordingly; then an algorithm is used to split and combine the descriptors, and the channel information of the image is extracted as the input of the convolutional neural network; and finally, we design a convolutional neural network-based to identify the dog species.

KEYWORDS: Convolutional neural network, Keras library, InceptionV3, InceptionResnetV2

I. INTRODUCTION

Computer vision is a field of the scientific study of algorithms which computer and other systems employ for the purpose of performing a specific task effectively and autonomously without using clear and detailed instructions from humans. Because of the fact that under such an algorithm, the computers can learn and make predictions themselves. Generally speaking, a training data is built in order to make predictions instead of explicitly programmed to per-form the task through machine learning algorithms based on sample data. The algorithms have been used in a wide variety of applications, such as stock prediction, recommendation systems, email filtering, object classification and computer vision, such as autonomous driving, where it is unfeasible to develop an algorithm of specific instructions to perform the corresponding task. Deep Convolutional Neural Network for Dog Breed classification is one of the best techniques to achieve this task. Popular CNN architectures have been used namely Inception-ResNetV2, and InceptionV3 all pre-trained on the Image Net dataset as robust feature extractors followed by the logistic regression classifier. The system is able to classify the crops into 12 different classes and weeds into 1 class. The models of InceptionRes-NetV2 and InceptionV3 models were loaded with a global average pooling layer added to the last convolution layer in each model. Performance analysis shows that using the Keras library with Inception Resnet V2 architecture drastically scored better than the other networks making it ideal for Breed classification. Based on top of PyTorch,

Keras contains popular algorithms for image classification and natural language tasks. However, we can also imagine the features (both low-level and higher-level) that are learned by the convolutional networks in the process of training. On the object CNNs are trying to categorize share many similar features, such as the breeds of dogs, it becomes hard to imagine the specific features that CNNs must learn in order to categorize these dogs correctly. This will be more clear if we take a look at sets of images such as Fig. 1 above, where the 3 dogs share almost all the same visible features, but belong to different classes. It is therefore interesting to see how well CNNs can perform on only dog breeds, compared to labels from all classes of objects in regular Image Net.

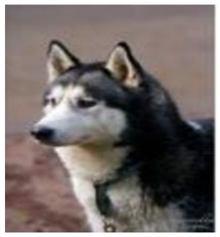
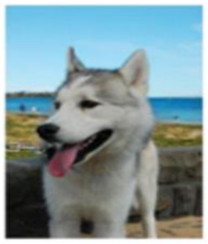
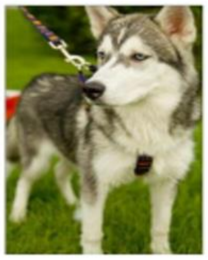




Fig. 1.Malamute Fig. 2.Eskimo Dog Fig. 3.Husky

However, we can also imagine the features(both low-level and higher-level) that are learned by the convolutional networks in the process of training. On the object CNNs are trying to categorize share many similar features, such as the breeds of dogs, it becomes hard to imagine the specific features that CNNs must learn in order to categorize these dogs correctly. This will be more clear if we take a look at sets of images such as Fig. 1 above, where the 3 dogs share almost all the same visible features, but belong to different classes. It is therefore interesting to see how well CNNs can perform on only dog breeds, compared to labels from all classes of objects in regular Image Net.

Convolutional neural networks (CNN) are a type of deep learning neural networks that are commonly used to classify images. CNNs are known for their ability to reduce computational time and adapt to different variations of images (for example, a well-trained CNN can detect an object from an image even if it is smaller, larger, rotated, translated, etc. from the original image—this is what is known as translation invariance).

As with understanding how any type of neural network works, one needs to understand the theoretical/mathematical side and an application of it to a real-world example.

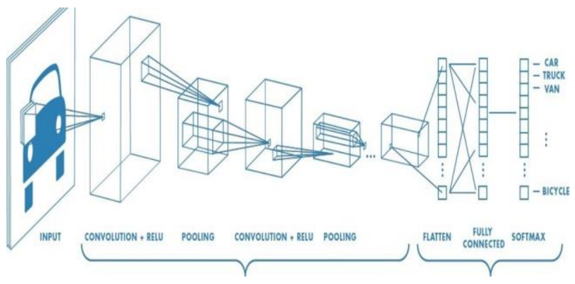


Fig. 4 CNN model

The first set in the architecture of a CNN is a set of convolution blocks, which have three components: convolution, ReLU, and pooling.

The first component (***convolution***) extracts features from the input image (shapes, curves, etc. that can help identify objects in an image). It does this by continuously applying a sliding filter to the image.

On a mathematical level, the convolution feature is derived from multiplying corresponding pairs of values between the current area being processed in the image (highlighted in purple in the animation above) and the kernel/filter (images are represented as a matrix of integer values (colors)). Given an input image ff and a filter/kernel hh, any cell's value with row mm and column nn can be computed by the following formula:

[Gm,n=(f∗h)m,n=∑j∑khj,kfm−j,n−kGm,n=(f∗h)m,n=∑j∑kh j,kfm−j,n−k]

The feature map is then fed to a is ***ReLU*** (Rectified Linear Unit). The goal of the ReLU layer is to introduce nonlinearity to the network (non-linearity increases the complexity that a neural network can detect). The derivation of the feature map is obviously a linear operation (**dot product**), so there needs to be some nonlinear activation within the CNN. The ReLU's mathematical function gg given an input zz is:

g(z)=max(0,z)

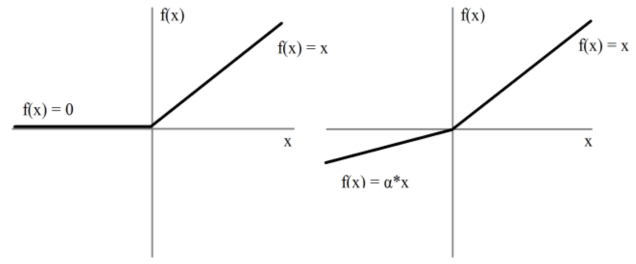


Fig. 5 Graphical Representation of ReLU function

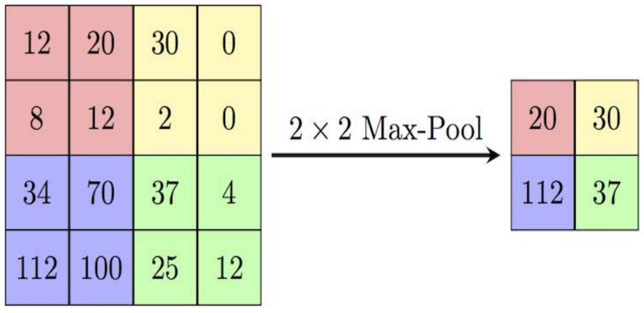
In the above picture, the left graph is the result of the ReLU function being applied.

ReLU is a non-linear function, because negative values of z are mapped to 0. However, for positive values of z, ReLU Is a linear function. This is what is known as a piecewise linear function (a hinge function), and this makes the ReLU function ideal:

- It introduces non-linearity (z<0z<0), which increases the level of complexity the CNN can detect.
- The ReLU is linear if z>0z>0, which preserves the speed advantage gradient-based optimization (i.e. gradient descent) has on linear models.

To summarize: ReLU increases the capabilities of the CNN model while still making it fast enough to train.

After ReLU, the model's data is sent to a ***pooling*** layer. The purpose of the pooling layer is to reduce the computational complexity of the CNN by reducing the feature map's spatial size, as well as to reduce overfitting my selecting the feature map's most important components. For example: the following 4x4 matrix was reduced to a 2x2 matrix through max pooling:



The most commonly used pooling function is called max pooling. Given a filter size and a stride (how far the filter moves horizontally and vertically) (dx,dy)(dx,dy), the max pool function takes the maximum amount from each element in the input (e.x. max(12, 20, 8, 12) = 20, max(30, 0, 2, 0) = 30, max(34, 70, 112, 100) = 112, max(37, 4, 25, 12) = 37)). The three components (convolution, ReLU, and pooling) are continuously applied to the feature map over and over—CNNs generally have multiple convolution blocks (like a stacked sandwich):

The standard softmax function (most commonly used) is the following:

Given an output vector yy: S(yi)=eyi∑jeyj

II. Literature Review

This section will depict the problem faced by researchers in this field. Before individuals figured out how to utilize the intensity of deep learning, the best-realized methodology was to cautiously physically structure the features. Commonly, these highlights must be connected to somewhere in the range of explicit limited arrangement of issues. To handle the issue of picture classification utilized a different kernel to consolidate kernels in a manner that is most discriminative for each class. Additionally, utilized the structure as a major aspect of an intelligent framework for winged animal species identification. A less complex issue that is like dog classification, is the issue of cat versus dog classification, and more work has been done on it. In a classifier on color features got 56.9% accuracy on the Asirra ”dog versus Cat” dataset. Golle et al accomplished a precision of 82.7% utilizing a Support Vector Machines classifier dependent on a mix of texture features and color. analysts utilized the SIFT (Scale-Invariant Feature Transform) feature to train a classifier lastly got an exactness of 92.9% on the Dog versus Cat issue. With progressions of CNNs, the precision rates expanded altogether. Accordingly, B. Liu et al accomplished the exactness of 94% by utilizing CNNs to learn highlights of pictures and SVMs for arrangement [13]. On the ”cat versus Dog” Kaggle rivalry most astounding consequence of 98.9% was accomplished by P. Sermanet in 2013 [4]. However, an issue of mutts classification has various real contrasts that make it significantly all the more testing. Right off the bat, it is a multi-class characterization issue, in which the calculation would need to distinguish a pooch breed out of 133 decisions. Besides, significantly more fine-grained photograph order is required, on the grounds that canines of various breeds look more like each other than to felines. At long last, Dogs versus Cats dataset had a few requests of size more information accessible for each classification: 12,500 versus 60. In [7] Liu et al attempted to handle the canine breed recognizable proof issue utilizing part confinement. They assembled model based geometric and appearance models of pooch breeds and their face parts. Their approach additionally included a pecking order of parts and breed-explicit part confinement. An acknowledgment rate of 67% was accomplished on a huge genuine informational index. Additionally, they tentatively exhibited that precise part confinement fundamentally expands order execution. Tremendous advancement has been made in fine-grained order with the headway of profound CNNs. [8] talks about information enlargement as a successful sys-tem for improving the exhibition of CNNs when managing constrained datasets. [9] brings up that profound learning models are ordinarily very repurposable, and accordingly a method called exchange picking up (moving took as a rule information from one, generally bigger, issue to another) is conceivable. It is underlined that it is a successful strategy for situations when just a little dataset is accessible for training

III. Methodology

This is a multiclass classification problem with 120 classes representing different breeds of dogs. Input is given as an image and the goal is to classify it to its appropriate class. The problem is tackled as an Image Classification problem using Deep Convolutional Neural Network. Further on, trans-fer learning technique is used to improve the accuracy.

A. The set of data mentioned

The set of data utilized in this study which is Stanford Dogs set of data, which contains 120 unique dog breeds and 10222 and 10357 images for training and testing respectively. This datasets is the small part of ImageNet challenging datasets. The training directory of images provided at the beginning is labelled by their id, rather than with the name of different Dog breeds. Exploratory data visualization was performed by plotting the number of samples per class and by plotting random images from each class in an image-grid. The data obtained was given in a discrete manner with various range of images in a particular breed. Working and training with the real-world categorical data is difficult to implement, especially when the relationship between the different classes is not ordinal. The various images needed to be segregated in a fixed set of categories according to their classes with their particular name of breeds. Individually each of the images were searched from the label.csv file and into their respective id, breeds and classes according to the train path. However, for each labels, the number of samples got, was skewed and varied in range from the minimum being around 65 to the maximum being more than 125 as shown in Figure 1.

B. Algorithms and techniques used

Convolutional neural networks (CNN) are used for feature extraction and they have became very famous due to different architectures present in some frameworks. The Convolution part of the CNN is built using two main components, namely Convolution layers and Pooling Layers. The convolution layers increase depth by computing output of neurons connected to local layers. Pooling layers perform down-sampling. CNN is used to extract the feature vectors from input images. An ANN model classifies the input based on the feature vector provided by CNN.
Transfer learning is an approach in DL where pre-trained models are used for building the model. Pre-trained architecture and weights are initialized. This idea became famous because the computation power required for training Deep Learning models is not available with everyone

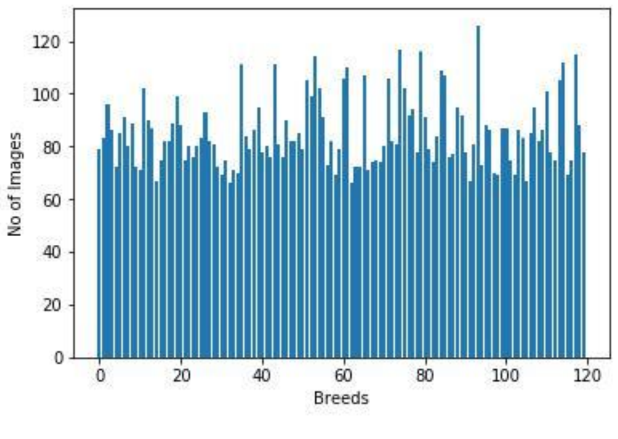


Fig. 6. Images per class

already pre-trained models are available which did some exceptionally good performance in the Imagenet competitions. These models can be loaded from libraries like Tensorflow, Keras, Pytorch and can be trained further according to the problem statement. So it is very common to use a pre-trained architecture and initialize them with imageNet weights and use the architecture by adding dense layers for classification as well as regression problems.
Output Softmax Activation function Layer is used for the multiclass classification.

IV. IMPLEMENTATION AND EXPERIMENT

A. Data Pre-processing and Augmentation

Images were reshaped to 224x224 pixels while using Keras library and 299x299 while Keras library.
Data was separated into training set and validation set in the ratio of 80:20 by making an Image Data Generator using Keras.

Moreover the images were augmented by flipping, zooming, shifting and rescaling by 255.It is always considered better to feed a different set of training data for each epoch of the CNN model. This makes up for the lack of training data and also present better scores, eliminating chances of overfitting.

The actions performed on the training data are Rotation,changing of brightness randomly, translation of the images and shearing of the pictures while using Keras.

B. Data Exploration

There are total 120 categories. Total Images are 20579. There are 10222 total training images. There are 10357 total testing images.The resolution of the images were quite variable with higher resolution ones being 4000x3000 pixels and smaller ones being 400x300 pixels.

C. Complications

The dataset provided was very unbalanced and minimum 10% of the images included human interference which concluded to less validation accuracy. Another reason for overfitting is that the Inception V3 and all other models which we used were built on the imageNet library and we are transferring that knowledge to a new problem.

V. RESULTS

Across all the various models, InceptionResNetV2, and InceptionV3 were tested. The validation Multiclass Loss Score varied from 1.85 (for Inception Resnet-V2) to 5.11947 (for InceptionV3).

TABLE I
TRAIN AND VAL ACCURACY TABLE

Model	Val Accuracy	Train Accuracy
Resnet101	71.63%	90.26%

Resnet50	63.78%	87.89%
InceptionResnetV2	40.72%	58.04%
InceptionV3	34.84%	52.49%

All the configurations used the same input size and no additional training on the current dataset. The models were trained for a maximum of 10 epochs and all achieved convergence. The table 1 shows the Accuracy and table 2 shows the Loss obtained across all the experimental configurations.

TABLE II
VALIDATION AND TEST LOSS TABLE

Model	Test Loss	Val Loss
Resnet101	0.34708	.4904
Resnet50	0.41525	.8306
InceptionResnetV2	2.6044	2.6044
InceptionV3	5.11947	6.4802

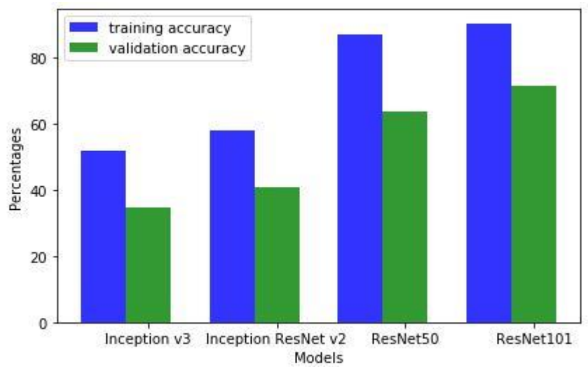
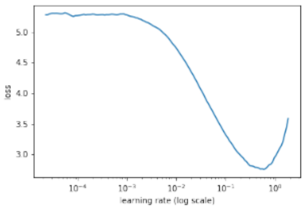


Fig. 8.Graphical Comparison of losses

magnitude and selected that as a learning rate (1e-1 in the example above).
Learning rate finder sits on top of other optimizers (e.g. momentum, Adam, etc) and help you to select the good learning rate given what other jerks you are using (such as advanced optimizers but not limited to optimizers).



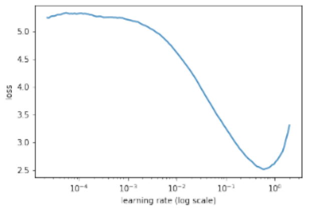


Fig. 7.Graphical Comparison of accuracies

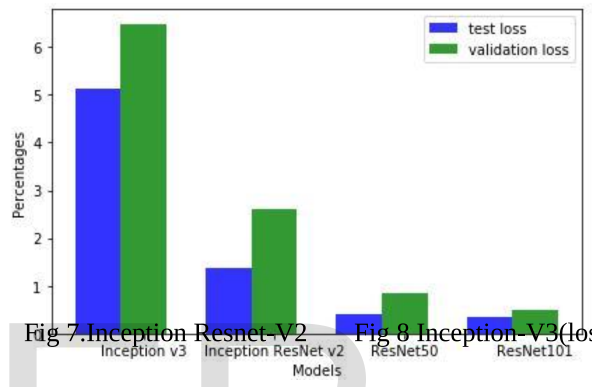


Fig 7.Inception Resnet-V2 Fig 8 Inception-V3(loss vs lr)

From the above graph (Learning rate VS Loss), if we zoom into the bell curve, we can observe the following things:
Whenever rate of learning is very limited, it will take enough time to reach to the bottom.
Whenever rate of learning is very large, it could get fluctuate from the bottom. Ultimately, the learning rate is very high due to which loss will get worse.
We looked for scam of learning rate against loss ,and predispose the lowest point and go back by one

VI. CONCLUSION AND FUTURE SCOPE

The current work relies on configurations with Inception Resnet V3 architecture as feature extractor and Transfer learning as the chosen training mechanism with a Softmax Activation function as a classifier to predict the final class.

The following are some changes that can be done to increase the performance further
As an improvement and future work, data masking can be done on the train set. Noise from background of the images (especially from barcodes) can be canceled by masking images. Without the background noise and restricting the visibility to the dogs, the model can be trained better, and performance can be improved significantly.
Various dimensionality reduction techniques can be applied on the image data.

The configuration can be changed to rely on different deep learning architecture for feature extraction.
The current work only initializes the weights with that of the model trained on the ImageNet. The model can be trained on the current dataset post initialization to achieve better performance.
Different proportions of layers can be frozen, unfrozen or reinitialized to further improve performance.

REFERENCES

[1] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, Rethinking the Inception Architecture for Computer Vision, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.
[2] O. Russakovsky et al., ImageNet Large Scale Visual Recognition Challenge.
[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Adv. Neural Inf. Process. System
[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, Return of the Devil in the Details: Delving Deep into Convolutional Nets, arXiv Prepr. arXiv
[5] Zhang Y, Zang K, Hua Z, et al. Research on algorithm of dog recognition in elevator based on video. Mech Des Manuf Eng 2018;
[6] Abdel-Hamid, A. r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, Convolutional neural networks for speech recognition, IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 22, no. 10, pp. 15331545, Oct 2014.
[7] Sermanet, A. Frome, and E. Real, Attention for fine-grained categorization, CoRR, vol. abs/1412.7054, 2014.
[8] Simon and E. Rodner, Neural activation constellations: Unsupervised part model discovery with convolutional networks, in Proceedings of the IEEE
[9] Zhang, J. Donahue, R. Girshick, and T. Darrell, Part-based r-cnns for fine-grained category detection, in Computer Vision ECCV 2014, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 834849.
[10] Duan, D. Parikh, D. Crandall, and K. Grauman, Discovering localized attributes for fine-grained recognition, in 2012 IEEE Conference on Computer Vision and Pattern Recognition, June 2012, pp. 34743481.
[11] Chen, J. Yang, H. Jin, E. Shechtman, J. Brandt, and T. X. Han, Selective pooling vector for fine-grained recognition, in 2015 IEEE in 2005 IEEE Winter Conference On Appplication of Computer Vision, Jan 2015, pp, 860867.
[12] medium.com/@marnieboyer/dog-breed-classification-using-a-pre-trained-cnn-model-84d0af72b4fc
[13] medium.com/@hiromi$_s$uenaga=deep learning 2 part 1 lesson 2 eeae2edd2be4
[14]towardsdatascience:com=fastai multi label image classification 8034be646e95
[15]www:kaggle:com=c=dog breed identif ication=discussion=46201latest 276907
[16]medium:com=datadriveninvestor=optimizers for training neural networks e0196662e21e
[17]towardsdatascience:com=fastai image classification 32d626da20
[18]towardsdatascience:com=fastai multi label image classification 8034be646e95
[19]www:freecodecamp:org=news=how i used deep learning to classif y medical images with fast ai cc4cf d64173c=
[20]joshvarty:com=2019=01=29=image classification with fastai=
[21]heartbeat:fritz:ai=training an image classification convolutional neural net to detect plant disease using fast ai a9f5315ed59
[22]medium:com=@taggatle=image classification with fastai v1 2908b6e06273
[23]medium:com=@tif a2up=image classification using deep neural networks a beginner friendly approach using tensorflow 94b0a090ccd4
[24]selimamrouni:github:io=portfolio=dog breed:html