Image classification using GoogleNet

R. Saldanha, S. Chougale, A. Sharma

National College of Ireland

x18183034@student.ncirl.ie
x18192327@student.ncirl.ie
x18198821@student.ncirl.ie

***Abstract*—Image classification is an integral part of many industries. It is used in almost all industries such as healthcare, supply chains, autonomous driving in vehicles industry, face recognition in security industries and many other areas. Convolutional Neural networks are widely used when designing an image recognition software. Due to the dense architecture of convolutional neural networks which consists of a lot of layers, the network can extract a lot of features from the image which helps in classifying the images. There are many new types of convolutional neural networks with unique features such as ResNet, GoogleNet, VggNet, which are very efficient in image classification tasks. In this paper the working of GoogleNet CNN is discussed followed using GoogleNet in classifying the images. Due to the dense architecture of GoogleNet, a lot of images are required to train the model to get good accuracy, but the model is trained with less images, 25000, and still the accuracy obtained is 74%. This implies how good GoogleNet is in image classification tasks even the model is used in unfavorable conditions.**

INTRODUCTION

Machine learning is the protagonist of the current world which is dominated by technology. Image recognition is a major part of the current world where technology is growing at exponential rates. Image recognition has many applications such as self- driving cars, organizing large image databases, facial recognition, etc. Jetpac[1] created virtual city guides with the help of Instagram images by using image recognition and then users are able to figure out almost everything they wish to know about the place they want to visit. Image recognition is also used to enable autonomous driving by identifying objects on roads such pathways, vehicles, moving objects, people etc. which could significantly decrease the rate of road accidents. Image recognition is also used in security industry to recognize people by facial recognition.

Image recognition also helps people with impaired vision by text to speech process. Most of the text in the images too can be understood by people with impaired vision by text-to-speech software which uses image recognition. Iris recognition[2] is used to identify people which is based on image recognition. This process is improved and has much better accuracy and is used in smart phones too for various tasks such as unlocking the phone. Image recognition is also used extensively in gaming to use user's real location and create virtual adventures. Health care industry also uses image recognition for a lot of purposes such as identifying diseases such as pneumothorax, pneumonia, breast cancer etc. from X-ray images.

CNN (Convolutional neural networks) are very popular when it comes to image classification tasks due to their high accuracy. GoogleNet[3] was able to classify images in the ImageNet dataset with error rate of close to 6% due to its dense architecture and unique type of layers present in the architecture, which is inception layer, in which the convolutions and pooling operations are processed in parallel instead of following sequential path.

Out of the existing CNN networks, GoogleNet performed with best statistics when trained on ImageNet dataset, which has over 14 million images. GoogleNet outperformed all the state of the art CNNs AlexNet, ZFNet, VggNet in 2015. The introduction of the inception module by GoogleNet was revolutionary, which led to researchers try many variations of CNN.

In this paper, the use of GoogleNet to classify images is discussed. In the next section the objective of the paper is discussed followed by source of data. The related work is discussed after source of data. After that methodology is discussed where a brief description of KDD followed by the working of CNN in mentioned. After that, the preprocessing of data is discussed and then the design of GoogleNet is discussed. The implementation of the model is discussed next. After this the results of the implementation are discussed followed by future work.

OBJECTIVE

The objective of the paper is to create the GoogleNet network and then use it for image classification.

DATA SOURCE

The dataset used in this paper is CIFER-10 dataset. The dataset has 60,000 images [4]. The images are of dimensions 32x32. There are 10 categories in which the images could be classified. All the images are colored images. The categories in which the images can be classified are airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. The dataset can be obtained from python's Keras library which is also used to design the GoogleNet CNN.

RELATED WORK

Singla et al. [5] used GoogleNet along with Caffe library of the CNN with an aim to conduct categorization of different variety of food products based on the images under study and observation as well as to classify whether these images can differentiate between food and non-food images. Food-5K, Food-11 and IFD datasets are used in which the first two were used for training data and the third was used for evaluating the performance of the trained model and to compare with the results obtained after classification. Accuracy, Confusion matrix, F measure and Cohen's Kappa are the performance evaluation matrices to evaluate the performance of the proposed model. The results from the analysis showed that for classifying between food and non-food images they got around 99 percent and for categorization between different food products around 83 percent was achieved.

A. T. Vo et al. [6] proposed a model which has two defined parameters 'n' and 'm' where 'n' denotes the number of layers and 'm' defines the number of filters applied in the convolutional network. The proposed model is nLmF-CNN which is an improved version of the traditional CNN and which is formulated from ConvNetJS to which they have given advertisement images which are captured online and a binary response in the form of yes or no is produced in which if the advertisement is clearly displayed then it is 'yes' else 'no'. k-fold cross validation as the evaluation technique was used. In order to decide upon which pair of (n,m) values would deliver best performance accuracy for this purpose they did a comparative trials like for filters (m) they considered 'n(n+1)' which is even number selection and for layers considered 'n' natural number selection and with this technique they achieved a best accuracy rate for their proposed model of around 86 % for a pair of (1,4) which justified the viability of the proposed model.

Wang et al. [7] used GoogleNet Inception CNN to recognize the medication pills. For enhancing the training accuracy of the proposed model as well as to tackle the problems faced by original pill images they applied six different data augmentation techniques which are color casting, projective distortion, Gaussian filter, medium filter, random scaling positioning, fixed rotation and background learned from validation set due to which roughly around 1500 synthetic images were produced. With a mean average precession (MAP) of around 0.3 score and decent percentage accuracy rate by top-N classification accuracy matrix as well as evaluating the proposed model on dataset made freely available by NIH the deemed proposed methodology is suitable for the analysis.

Zhu et al. [8] used an optimized version of CNN architecture which is GoogleNet coupled with fine tuning strategy with an aim to recognize extreme weather conditions and for this purpose used ILSVRC-2012 dataset. They created a weather dataset for recognizing extreme weather patterns and furthermore fine tunned the pre-trained model for better performance. The results from the analysis proved that optimized version of GoogleNet model outperforms the original GoogleNet versions in various aspects of consideration like processing of images faster on CPU and GPU and recognition accuracy rate improved from around 94% to 95% also the proposed model size is smaller than the original pre-trained GoogleNet model.

Sabzi et al. [9] used GoogleNet and AlexNet to recognize handwritten words in Persian language. They applied data augmentation technique in order to have a standard dimension size and for overcoming issues like overfitting in the training model. They further pass on the pre-processed images with as well as without Batch normalization to GoogleNet and AlexNet. The results from the analysis proved that GoogleNet outperforms and delivers a high accuracy rate of around 99% when the pre-processed data with batch normalization is passed to the model.

H. Yanagisawa et al. [10] employed four different deep learning based algorithms for object detection and classification which are Fast R-CNN, Faster R-CNN in which R represents the regions with Convolutional Neural Network and Single Shot MultiBox Detector (SSD) for object detection and classification. They carried out a comparative analysis of the methodology to check which of the proposed methods were suitable for detecting the various objects. The results from the comparative analysis proved that Fast R-CNN outperformed the other three methods for detecting two objects under study which are panel layout and speech balloon while the objects like the face and the text spoken by the characters in the comic is well detected by Faster R-CNN.

G. Y. Son et al. [11] used three CNN architectures which are VGG-16, AlexNet and GoogleNet to investigate the smoke and flames through CCTV footages from three different location and types of images and retrieved all these images from 768 fire image dataset as their deep learning models. With enough images used for training, testing and validation datasets it proved that all the three models gave good accuracy and observed that VGG-16 net outperformed the other models.

T. Fang [12] used two pre-trained versions of CNN GoogleNet and AlexNet to detect cancer in lungs. Data augmentation on the LIDC-IDRI dataset was done to ensure that the training dataset performs efficiently well. After 300 epochs cycles it was observed that the trained model delivers more than 80% of accuracy and sensitivity rate and around 80% specificity rate. In order to get tri-dimensional images of CT scan medium intensity projection (MIP) was used which improved the accuracy of the proposed CNN models by around 12% and from the results of the analysis it was observed that GoogleNet outperformed by a small margin difference of around 2% to AlexNet.

M. Moran et al. [13] proposed a methodology to identify thyroid nodules in the infrared images and for this purpose employed three different architecture of CNN which are GoogleNet, AlexNet and VGG net and also used Accuracy measure for evaluation of performance accuracy and from the results of the analysis GoogleNet outperformed the other nets delivering an accuracy of around 86%.

Pelletier et al. [14] employed two pre-trained CNN architectures which are GoogleNet and AlexNet in order to classify the images of fishes which are captured in an non structured condition and with that they achieved a good accuracy rate in which GoogleNet delivered better accuracy rate more than 90% compared to AlexNet. This accuracy achieved by GoogleNet is high at both the rates when the entire image is considered for classification and when the images for fishing areas were cropped.

Zou et al. [15] proposed a methodology by employing deep neural networks which is Inception v1 CNN in order to classify the two tumour groups which are malignant and benign by considering the histopathological breast cancer images obtained after biopsy of around 82 patients. In order to import the images in its original aspect ratio format they utilized two pooling strategies which are Spatial Pyramid Pooling (SSP) and Global Average Pooling (GAP) along with an optimization algorithmic technique which is Adam along with fivefold cross validation technique which proved to be an effective way since the proposed model accurately classified the images into the respective categories. They also observed that despite employing two pooling strategies GAP outperformed SSP for categorizing breast cancer images.

Kamaron Arzar et al. [16] employed pre-trained GoogleNet CNN architecture with confusion matrix and Softmax classifier for identifying as well as classifying the images of butterflies into four categories. They considered around thirty images in the training and testing dataset for each type of butterfly and achieved an overall accuracy of around 97 % for the proposed model with a surprising cent percent accuracy for two specifies of butterfly.

Jasitha et al. [17] employed two different versions of CNN models which are fine-tuned GoogleNet and VGG-16 and also used support vector machine (SVM) classifier on three different datasets which are Dleaf, Flavia and Leafl with an aim to identify and classify a special category of plant leaves which are based on venation. Using confusion matrix along with five-fold cross validation technique they achieved a performance accuracy rate of approximately 99% for the fine-tuned GoogleNet which outperformed the other proposed model on the third dataset which is Leafl using SVM as the classifier which deemed to be a suitable methodology for fulfilling the scope of the project.

Arti et al. [18] had applied two pre-trained CNN models which are GoogleNet and AlexNet to classify X-Ray images. They pre-processed the images retrieved from IRMA database and split the datasets into training and testing datasets in a combination of 80:20 ratio as well as to evaluate the performance and accuracy of the model they used confusion matrix as the evaluation parameter and with that it was proved that AlexNet outperformed GoogleNet model having an accuracy rate of classifying the X-Ray images of around 93%.

Mittel et al. [19] used two pre-trained CNN models which are GoogleNet and AlexNet with an aim to detect cracks from images of metals taken during inspection and to prevent overfitting problems in the training model they pre-processed the images using methods like data augmentation and oversampling. With an F1 score test accuracy of around 0.8 and performance accuracy of around 99% GoogleNet outperformed AlexNet.

Suryawati et al. [20] had employed four deep learning neural network models which are CNN and its three pre-trained models which are GoogleNet, AlexNet and VGGNet with an aim to detect diseases in tomato using the images of the tomato plant from the PlantVillage dataset with that used around 80% for training and remaining 20% for testing and validation purpose. The results of the analysis proved that all the proposed models are deemed suitable for detecting diseases in the tomato plant with around 89% accuracy from which VGGNet outperformed and delivered an accuracy of around 95%.

I. K. E. Purnama et al. [21] had employed two pre trained deep learning models which are MobileNet v1 and Inception V3 to CNN with an intension to classify as well as detect seven different categories of diseases which are marked as cancerous that prevail on skin and which are examined through dermoscopic skin images retrieved from MNIST HAM10000 dataset. In order to increase the volume and to maintain a balance in the number of data for seven different classes of diseases they augmented the training dataset and in order to evaluate the accuracy of the performance of the proposed model confusion matrix was used with (K-1) first fold cross validation technique as well as used a classifier than runs on the teledermatology applications. The results from the analysis using the proposed models proved that Inception V3 model outperforms the MobileNet v1 which is suitable for their research work.

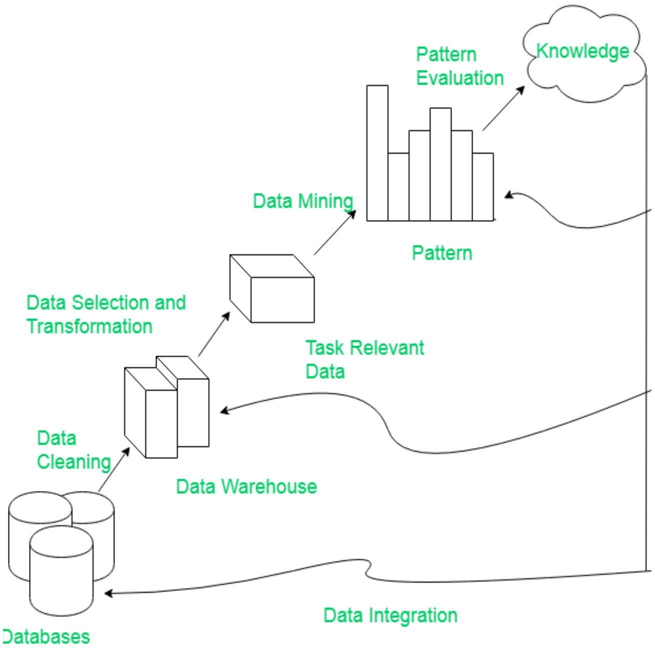
Arya et al. [22] used CNN and AlexNet to detect the diseases that may or may not persist in potatoes and the leaves of the mango tree and to conduct the analysis they used two different sources to retrieve images and used confusion matrix as the performance evaluation parameter. The results of the analysis proved that AlexNet outperformed with around 98% accuracy compared to CNN.

R. K. Mohapatra et al. [23] used GoogleNet or Inception v1 CNN in order to differentiate between original and forged signature. CEDAR, BHSig260 and UTSig were used to collect genuine and forged signatures. In order to fine tune the performance and learning capability of the proposed model they pre-processed the input samples as filtered and grayscale images. The results from the analysis proved that the proposed model InceptionSVGNet outperforms the other models for classifying between original and forged handwritten signatures.

Ma et al. [24] aimed at recognizing the potatoes which are in sprouting conditions. The limitation found by using the traditional image recognition technique in recognizing the complex features of potatoes encouraged them to use deep neural network models and due to this reason, they employed three different CNN models which are LeNet, AlexNet and GoogleNet. The results from the analysis proved that GoogleNet CNN model outperformed compared to the other two models and achieved an accuracy rate of around 80 % which deemed to be a potential contributor in recognizing sprouting potatoes.

METHODOLOGY

In this research KDD(knowledge discovery in dataset) methodology is used. KDD is an iterative process in which the focus is entirely on the data used in the research. In KDD the goal is to extract knowledge from the database by recognizing the patterns. In this process the data is cleaned, then analyzed to get knowledge from data and then use this knowledge to predict the data. The below image[25] is a representation of KDD.

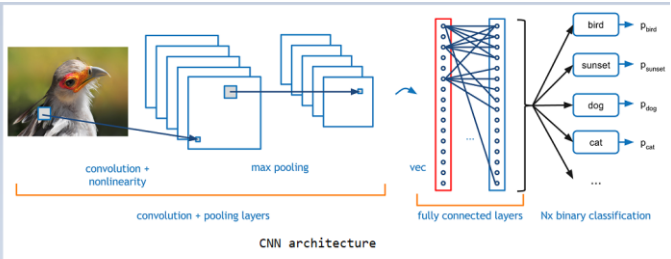


The first step in KDD is to get data from databases and then clean the data. Cleaning of data means removing the outliers, removing the missing values or replacing them with statistics like mean. After the data is transformed, for example, attribute transformations such as converting the categorical values into 0 and 1, or dimensionality reduction and then further removing all the data which is not required in the research. The next step is the extraction of knowledge from the data by various methods such as visualizing the data to find out patterns.

Back propagation is an important part in KDD methodology. In every step of KDD process the knowledge that is gained is used again to gain more insights from the data when the steps are repeated. Back propagation is the most important part of CNN process, this will be explained later in the description of CNN.

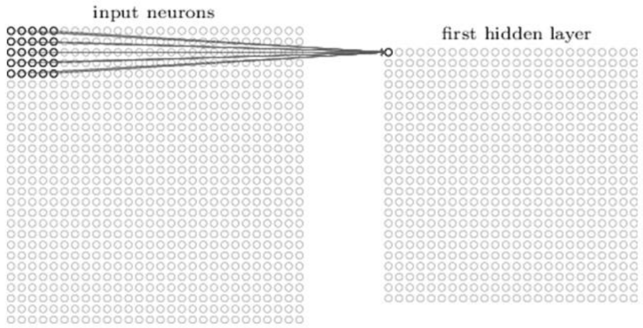
CNN Overview

Image classification is the categorization of the images, for example, if an image is of house or a vehicle. CNN is one of the best technologies when it comes to classifying images. An image is an array of numbers of dimensions YxYx3, where Y is in range 0 to 255 which is pixel density and number 3 implies different RGB values when image is colored.
CNN is a neural network which classifies the image by identifying various patterns in images such as edges, shapes etc. The image below describes a general CNN network[26].



CNN architecture

CNN has many layers; convolutional layer is always the first layer in the network. The input image which is an array of numbers is passed to this layer and then a filter, which is also an array of numbers, passes across the entire image and outputs a number every time it moves from one position to another. This process is best described in the image below[26].



Visualization of 5x5 filter convolving around an input image

The filter starts at the top left position of the image and all pixel values are multiplied by filter values and added to output a number, and then the filter moves to different position, right, as per the value of stride, if the stride is 1 then filter will move by one unit in right direction. This process is repeated till the filter has moved across the entire image. Each filter identifies a feature which could be anything such as any curve or an edge.

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Example of detection of a curve by a filter

In the above image[26], a filter is detecting a curve, which is a part of a bigger image and then outputs a number which corresponds to the curve. As the network goes deeper, more complex features of images such as shape of a leg are determined by filters. The output produced by these filters are then passed as an input to another convolution layer. At the end of the network there is a fully connected layer which outputs an N dimensional vector where N is the number of categories and the layer checks all previous layers and then determines which all features belongs to which category, for example, if the output vector is of the form [0.1, 0.15, 0.75], then it implies that there are 10%, 15% and 75% chances that image belongs to category A, B or C.

CNN backpropagation and other parameters

The filters are first set to initial values and they are not able to detect the patterns correctly and therefore the image could be classified wrong during the training of the model. When the image is classified incorrectly then with the help of loss function and learning rate, the weights, which are values of filter matrix, are adjusted so that the next time the patterns are detected better. This process keeps repeating and the accuracy of the training model keeps increasing.

A brief description of the other parameters related to the CNN architecture are explained below.

Activation function – This function is used to get output from the convolutional layer, it adds non-linearity to the input to output process. In a way, the activation function decides whether there should be an output from the filter or not.

Pooling function – After the output is obtained from the convolutional layer, the pooling function is used to reduce the dimensions of the output to reduce the computations of the network.

Filter size – This is the dimension of the filter used. If the filter size is 2x2, then the filter is a matrix of dimensions 2x2.

Stride – This is a number which determines how much the filter should move from one location to another when convolving across the image. If the stride is 2 then the filter will by move by 2 units right after the filter and pixel values are multiplied and added.

Zero padding – This is used to add zeros across the borders of the input array so that when the output is obtained after the filter has passed across entire image, the dimensions of the input and output are same.

Dropout – This function is used during training of the model to randomly drop the weights to avoid overfitting in the model.

Optimization function – This function determines the adjustment of weights during backpropagation. It updates the weight values of the filters to improve the model.

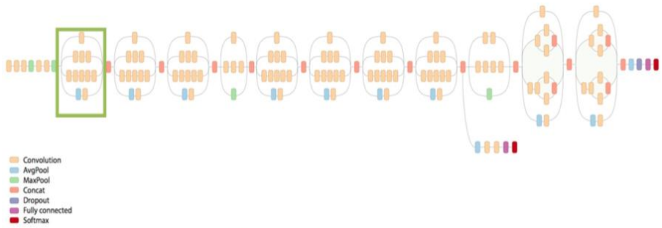
Learning rate – This is the rate by which weights should be updated during the backpropagation. If the learning rate is less then there will be more less change in the magnitude of the weights.

Data preprocessing

The input images are of the size 32x32, but the dimensions of the images are needed to be 224x224 to use as input in the GoogleNet CNN. Therefore, the images are resized into 224x224x3 array of numbers with the help resize function of CV2 library of python. This library has many functions which are required for image processing. The type of image array has been changed from 'int' to 'float'. This operation is performed by 'astype' function which is numpy array function used to change the datatype of an array. This is performed to normalize the image input array. After the datatype of the image array is transformed, the image array is divided by 255 to produce a normalized image array, the input images are now ready to be fed into the GoogleNet CNN.

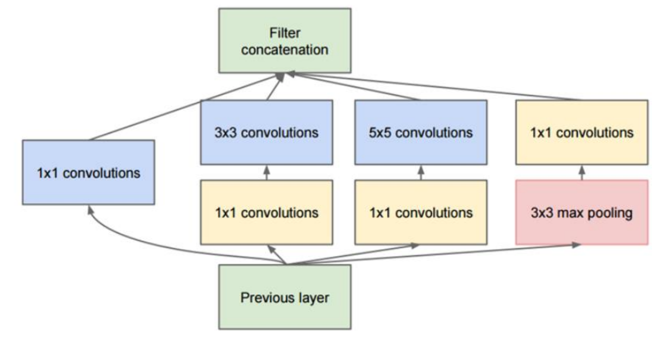
GoogleNet design

GoogleNet was the first CNN which introduced the concept the all the CNN layers do not have be sequential. In GoogleNet, there is an inception layer which is a unique feature of GoogleNet.



GoogleNet architecture

The above image [3]is an architecture of GoogleNet. The green box in the image is an inception layer. The inceptions layer performs the convolutional and pooling work simultaneously which detects a lot of useful patterns. The below image[3] clearly explains how the inception module works.



Inception Layer

The 1x1 convolutions in the inception layer are used as a dimensionality reduction to reduce the extremely large number of inputs. So together when the large sized convolution, medium sized convolution and the pooling operation are performed, the layer can extract very fine grain patterns in the image. All these operations are performed wile remaining computationally considerate.

The GoogleNet is designed with the help of python's Keras library. Keras is a very popular and easy to use machine learning library, many complex and deep CNNs can be created easily by using Keras. At the start of the CNN, a 2d convolutional layer with 7x7 filter is present in which the input with dimensions 224x224x3 is given. The padding used is the first convolution layer is ‘same’ with the stride of (2,2) and activation function used is ReLU(rectified linear unit). To set the initial random weights Glorot uniform initializer is used in the first layer. After the first layer, to reduce the dimensions of the output, maxpooling function is used. After this another convolution layer with 3x3 filter and a stride of 2 is added followed by a maxpooling function. Two similar inception layers are added then, having four 1x1 convolutions and one 3x3 and 5x5 convolution layers. The output of all the convolutions is concatenated and then maxpool function is added to reduce the input dimensions. After that five inception modules are added followed by a maxpool function. Two more inception modules are added then followed by a global average pooling function which helps to reduce the overfitting of the model. After the global average pool, a dense layer with softmax activation function is added to obtain the output. The dropout value used in the model 0.4.

Model: "GoogleNet_Inception_v1"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 224, 224, 3)	0	
conv_1_7x7/2 (Conv2D)	(None, 112, 112, 64)	9472	input_1[0][0]
max_pool_1_3x3/2 (MaxPooling2D)	(None, 56, 56, 64)	0	conv_1_7x7/2[0][0]
conv_2a_3x3/1 (Conv2D)	(None, 56, 56, 64)	4160	max_pool_1_3x3/2[0][0]
conv_2b_3x3/1 (Conv2D)	(None, 56, 56, 192)	110784	conv_2a_3x3/1[0][0]
max_pool_2_3x3/2 (MaxPooling2D)	(None, 28, 28, 192)	0	conv_2b_3x3/1[0][0]
conv2d_2 (Conv2D)	(None, 28, 28, 96)	18528	max_pool_2_3x3/2[0][0]
conv2d_4 (Conv2D)	(None, 28, 28, 16)	3088	max_pool_2_3x3/2[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 28, 28, 192)	0	max_pool_2_3x3/2[0][0]
conv2d_1 (Conv2D)	(None, 28, 28, 64)	12352	max_pool_2_3x3/2[0][0]
conv2d_3 (Conv2D)	(None, 28, 28, 128)	110720	conv2d_2[0][0]
conv2d_5 (Conv2D)	(None, 28, 28, 32)	12832	conv2d_4[0][0]
conv2d_6 (Conv2D)	(None, 28, 28, 32)	6176	max_pooling2d_1[0][0]
inception_3a (Concatenate)	(None, 28, 28, 256)	0	conv2d_1[0][0] conv2d_3[0][0] conv2d_5[0][0] conv2d_6[0][0]
conv2d_8 (Conv2D)	(None, 28, 28, 128)	32896	inception_3a[0][0]
conv2d_10 (Conv2D)	(None, 28, 28, 32)	8224	inception_3a[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 28, 28, 256)	0	inception_3a[0][0]
conv2d_7 (Conv2D)	(None, 28, 28, 128)	32896	inception_3a[0][0]
conv2d_9 (Conv2D)	(None, 28, 28, 192)	221376	conv2d_8[0][0]
conv2d_11 (Conv2D)	(None, 28, 28, 96)	76896	conv2d_10[0][0]
conv2d_12 (Conv2D)	(None, 28, 28, 64)	16448	max_pooling2d_2[0][0]

The above image is the summarization of the GoogleNet model obtained from model. Summary. The model has nine inception layers, but the above image is the summary till the first layer only, this gives us the idea how big this CNN is. The whole summary of the architecture would be at least five such images. The network has over 100 layers.

Implementation

The model is trained on Google colab. The input images used in training the model are 25,000 and the number of images for validation are 2,000. The model was trained for 25 epochs. The batch size used is 256. The learning rate in the model is 0.01. The loss function used in the model is categorical crossentropy, the loss function is used to calculate the prediction error which is used later in the adjustment of weights during the backpropagation. The optimization function used is SGD (Stochastic gradient descent), the optimizer function is used in updating the weights during the backpropagation. The number of input images used is 25,000 although there are more images present which could be used, this is due to the RAM issue, during the data preprocessing when the image data is transformed, a RAM error is issued which states that more RAM is required. The limit of free available RAM is 25 GB; hence a smaller number of images are used to train the model. The alternative method that was tried to use a greater number of images was import the images in batches and then later concatenate all the image arrays, but the method was unsuccessful as the RAM was not enough for the concatenation process.

RESULTS

The validation accuracy obtained after 25 epochs is about 74%. The accuracy of 65% was achieved after 18 epochs, which suggests that a tuning of parameters could result in better accuracy with less epochs. The mean absolute error obtained is 6%. The results of the model are good enough if the input data is considered. GoogleNet CNN has a very dense architecture which needs a lot of data to train. The accuracy would have been improved if more data was used. The training time of the model was much less on Google colab GPU Tesla K80 if compared to the laptop that was used to test on 10,000 images. Google colab with the hardware accelerator selected as TPU (Tensor processor unit) took around 6 hours to train and the when the model was trained on 10,000 images on laptop, it took 7 hours to complete. The laptop mentioned here has 16 GB of RAM with AMD Ryzen 7 processor with inbuilt Radeon Vega graphics card.

FUTURE WORK

The future work includes using more images data and fine tuning the training parameters. The GoogleNet performs well when there is a huge amount of input data for training, otherwise such a dense network of over 100 layers is not used to its full potential. Instead of using SGD optimizer, new optimizers could be used such as Adam, Adamax or Nadam.
There are updated versions of GoogleNet available now, the latest GoogleNet is Inception V-4, this model could be used to check the difference in the accuracy. Looking at the potential scope of other CNN pre-trained architecture like VGGNet and AlexNet which is evident from the past research work done by many researchers worldwide the future roadmap would be to use these three CNN model on our dataset as well as preprocess the training datasets using image data augmentation methodology like cropping, rotation, translation in order to overcome problems liked overfitting and with that perform a comparative analysis of the performance and accuracy of the model in order to classify the images using performance evaluation matrices like confusion matrix, F1 score and many others.

REFERENCES

[1] Golemanova, R., 2019. *7 Image Recognition Uses Of The Future - Imagga Blog*. [online] Imagga Blog. Available at: https://imagga.com/blog/7-image-recognition-uses-of-the-future/.

[2] A. Choudhury, "8 Uses Cases Of Image Recognition That We See In Our Daily Lives", *Analytics India Magazine*, 2019. [Online]. Available: https://analyticsindiamag.com/8-uses-cases-of-image-recognition-that-we-see-in-our-daily-lives/.

[3] A. Deshpande, "The 9 Deep Learning Papers You Need To Know About &#40&#85nderstanding C&#78&#78s Part 3&#41", *Adeshpande3.github.io*, 2016. [Online]. Available: https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html.

[4] Krizhevsky, A., 2009. *Learning Multiple Layers Of Features From Tiny Images*. [online] Cs.toronto.edu. Available at: http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[5] A. Singla and L. Yuan, "Food / Non-food Image Classification and Food Categorization using Pre-Trained GoogLeNet Model," pp. 3–11.

[6] A. T. Vo, H. S. Tran, and T. H. Le, "Advertisement image classification using convolutional neural network," *Proc. - 2017 9th Int. Conf. Knowl. Syst. Eng. KSE 2017*, vol. 2017-January, pp. 197–202, 2017, doi: 10.1109/KSE.2017.8119458.

[7] Y. Wang, J. Ribera, C. Liu, S. Yarlagadda, and F. Zhu, "Pill Recognition Using Minimal Labeled Data," *Proc. - 2017 IEEE 3rd Int. Conf. Multimed. Big Data, BigMM 2017*, pp. 346–353, 2017, doi: 10.1109/BigMM.2017.61.

[8] Z. Zhu, J. Li, L. Zhuo, and J. Zhang, "Extreme Weather Recognition Using a Novel Fine-Tuning Strategy and Optimized GoogLeNet," *DICTA 2017 - 2017 Int. Conf. Digit. Image Comput. Tech. Appl.*, vol. 2017-December, pp. 1–7, 2017, doi: 10.1109/DICTA.2017.8227431.

[9] R. Sabzi et al., "Recognizing Persian handwritten words using deep convolutional networks," 2017 Artificial Intelligence and Signal Processing Conference (AISP), Shiraz, 2017, pp. 85-90.

[10] H. Yanagisawa, T. Yamashita, and H. Watanabe, "A study on object detection method from manga images using CNN," *2018 Int. Work. Adv. Image Technol. IWAIT 2018*, pp. 1–4, 2018, doi: 10.1109/IWAIT.2018.8369633.

[11] G. Y. Son, J. S. Park, B. W. Yoon, and J. G. Song, "Video Based Smoke and Flame Detection Using Convolutional Neural Network," *Proc. - 14th Int. Conf. Signal Image Technol. Internet Based Syst. SITIS 2018*, pp. 365–368, 2018, doi: 10.1109/SITIS.2018.00063.

[12] T. Fang, "A Novel Computer-Aided Lung Cancer Detection Method Based on Transfer Learning from GoogLeNet and Median Intensity Projections," *2018 IEEE Int. Conf. Comput. Commun. Eng. Technol. CCET 2018*, pp. 286–290, 2018, doi: 10.1109/CCET.2018.8542189.

[13] M. B. H. Moran *et al.*, "Identification of thyroid nodules in infrared images by convolutional neural networks," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2018-July, 2018, doi: 10.1109/IJCNN.2018.8489032.

[14] S. Pelletier, A. Montacir, H. Zakari, and M. Akhloufi, "Deep Learning for Marine Resources Classification in Non-Structured Scenarios: Training vs. Transfer Learning," *Can. Conf. Electr. Comput. Eng.*, vol. 2018-May, 2018, doi: 10.1109/CCECE.2018.8447682.

[15] W. Zou, H. Lu, K. Yan, and M. Ye, "Breast cancer histopathological image classification using deep learning," *Proc. - 10th Int. Conf. Inf. Technol. Med. Educ. ITME 2019*, pp. 53–57, 2019, doi: 10.1109/ITME.2019.00023.

[16] N. N. Kamaron Arzar, N. Sabri, N. F. Mohd Johari, A. Amilah Shari, M. R. Mohd Noordin, and S. Ibrahim, "Butterfly Species Identification Using Convolutional Neural Network (CNN)," *2019 IEEE Int. Conf. Autom. Control Intell. Syst. I2CACIS 2019 - Proc.*, no. June, pp. 221–224, 2019, doi: 10.1109/I2CACIS.2019.8825031.

[17] P. Jasitha, M. R. DIleep, and M. DIvya, "Venation Based Plant Leaves Classification Using GoogLeNet and VGG," *2019 4th IEEE Int. Conf. Recent Trends Electron. Information, Commun. Technol. RTEICT 2019 - Proc.*, pp. 715–719, 2019, doi: 10.1109/RTEICT46194.2019.9016966.

[18] P. Arti, A. Agrawal, A. Adishesh, V. M. Lahari, and K. B. Niranjana, "Convolutional neural network models for content based X-Ray image classification," *2019 5th IEEE Int. WIE Conf. Electr. Comput. Eng. WIECON-ECE 2019 - Proc.*, pp. 1–4, 2019, doi: 10.1109/WIECON-ECE48653.2019.9019943.

[19] D. Mittel and F. Kerber, "Vision-Based Crack Detection using Transfer Learning in Metal Forming Processes," *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, vol. 2019-September, pp. 544–551, 2019, doi: 10.1109/ETFA.2019.8869084.

[20] E. Suryawati, R. Sustika, R. S. Yuwana, A. Subekti, and H. F. Pardede, "Deep structured convolutional neural network for tomato diseases detection," *2018 Int. Conf. Adv. Comput. Sci. Inf. Syst. ICACSIS 2018*, pp. 385–390, 2019, doi: 10.1109/ICACSIS.2018.8618169.

[21] I. K. E. Purnama *et al.*, "Disease Classification based on Dermoscopic Skin Images Using Convolutional Neural Network in Teledermatology System," pp. 1–5, 2020, doi: 10.1109/cenim48368.2019.8973303

[22] S. Arya and R. Singh, "A Comparative Study of CNN and AlexNet for Detection of Disease in Potato and Mango leaf," no. Dl, pp. 1–6, 2020, doi: 10.1109/icict46931.2019.8977648.

[23] R. K. Mohapatra, K. Shaswat, and S. Kedia, “Offline Handwritten Signature Verification using CNN inspired by Inception V1 Architecture,” pp. 263–267, 2020, doi: 10.1109/iciip47207.2019.8985925.

[24] J. Ma, J. Rao, Y. Qiao, and W. Liu, “Sprouting Potato Recognition Based on Deep Neural Network GoogLeNet,” *2018 IEEE 3rd Int. Conf. Cloud Comput. Internet Things*, pp. 502–505, 2020, doi: 10.1109/cciot45285.2018.9032562.

[25] Rajput, A., n.d. *KDD Process In Data Mining - Geeksforgeeks*. [online] Geeksforgeeks. Available at: https://www.geeksforgeeks.org/kdd-process-in-data-mining/.

[26] A. Deshpande, "A Beginner's Guide To Understanding Convolutional Neural Networks", *Adeshpande3.github.io*, 2016. [Online]. Available: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

[27] "Home - Keras Documentation", *Keras.io*. [Online]. Available: https://keras.io/.