INDUSTRY INTERNSHIP

On

Image Classification using CNN with Keras

Submitted to

Department of Computer Science

Amity School of Engineering And Technology

Submitted By

Ankita Singh

B.Tech(CSE) 7^{th} Sem

A7605217051

under the guidance of

Dr. Pawan Singh

Department of Computer Science and Engineering

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY
AMITY UNIVERSITY UTTAR PRADESH
LUCKNOW (U.P.)
June 2020

AMITY UNIVERSITY

————UTTAR PRADESH————

DECLARATION BY THE STUDENT

I, **Ankita Singh**, student of B.Tech hereby declare that the report titled "**Image Classification with CNN using Keras**" which is submitted by me to Department of Computer Science and Engineering, **Amity School of Engineering and Technology**, Amity University Uttar Pradesh, Lucknow, in partial fulfillment of requirement for the award of the degree of BACHELOR OF TECHNOLOGY in Computer Science and Engineering has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition.

The Author attests that permission has been obtained for the use of any copy righted material appearing in the Project report other than brief excerpts requiring only proper acknowledgement in scholarly writing and all such use is acknowledged.

Date:

Ankita Singh
B.Tech (CSE)
A7605217051

AMITY UNIVERSITY

——————UTTAR PRADESH——————

CERTIFICATE

On the basis of declaration submitted by **Ankita Singh** student of B.Tech, I hereby certify that the report title **“Image Classification using CNN with Keras”** which is submitted to **Amity School of Engineering and Technology**, Amity University, Lucknow, Uttar Pradesh in partial fulfillment of the requirement for the award of the degree of BACHELOR OF TECHNOLOGY in Computer Science and Engineering, is an original contribution with existing knowledge and faithful record of work carried out by her under my guidance and supervision.

To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Lucknow

Date :

Dr.Pawan Singh
Associate Professor
Amity University

Prof. (Dr.) DeepakArora
Head – CSE &IT

External Examiner

Wg.Cdr (Dr.) Anil Kumar (Retd)
Asst. Pro. VC & Director ASET
Amity University, Lucknow

ACKNOWLEDGEMENT

First of all, I would like to thank Dr. Pawan Singh for steering me in the right direction in my beginnings.

When I look at the amount of knowledge I have gained till date in my B. Tech. journey, I realize how much time and effort spent toward the completion of this work; not only by me but also by key individuals whom I feel very indebted to.

I gratefully acknowledge the support and guidance of my supervisor Prof. Dr. Pawan Singh. I would not imagine completing this work without all his advices. I acknowledge the time he spent in our weekly meetings, even with his busy schedule and other commitments. I am deeply indebted to him for his patience. He assisted in all aspects of this work from discussing new ideas to writing and completing this project.

I would like to thank Amity University, Lucknow Campus for providing me a wonderful platform for this work. I would also like to thank the library for the knowledge I acquired with its help. I am highly obliged to my HOD, Dr. Deepak Arora and my HOI, Wg. Cdr. Anil Tiwari for helping me in the completion of this report. Lastly, I want to thank my family for supporting me entirely.

I would like to express my deep gratitude and appropriation to my family and friends who were very cooperative and kind in getting me on the road.

ABSTRACT

Image classification is a paramount topic in artificial vision systems which have drawn a notable amount of interest over the past years. This field aims to classify an image, which is an input, based on its visual content. Currently, most people relied on hand-crafted feature to describe an image in a particular way. Then, using classifiers which are learnable, such as SVM, random forest and decision tree were applied to the extract features to come to a final decision. The problem arises when large numbers of photos are concerned. It becomes too difficult problem to find feature from them. This is the one of reasons that deep neural network model has been introduced. Owing to the presence of deep learning, it becomes feasible to represent the hierarchical nature of features using various number of layers and corresponding weight with them. The existing image classification methods have been gradually applied in real world problems, but then there are various problems in its application process, such as un-satisfactory effect, very low classification accuracy or then and weak adaptive ability. The deep learning model has robust learning ability, which combines the feature extraction and the process of classification into a whole to complete the image classification test, which can improve the image classification accuracy effectively. Convolution Neural Networks are a powerful deep neural network technique. These networks preserve the spatial structure of a problem and were built for object recognition tasks such as classifying an image into respective classes. CNN are very known because people are achieving state of the art results on complex computer vision and natural language processing tasks. Convolutional neural networks have been extensively used for automatic image classification systems.

Key Terms: Machine Learning, Deep Learning, Deep neural Networks, Convolutional Neural Network, CIFAR-10 Dataset

INDEX

1. Introduction

1.1 Machine learning

Machine learning is an operation that requires teaching any computer system how to forge accurate predictions when any type of data has been fed into the computer system.These predictions could be the answer to whether a piece of vegetable in a photo is a broccoli or a beetroot, analyzing twitter comments as negative, positive or neutral, predicting stock prices of the market, whether an email is a spam or not, or recognizing a speech accurately so as to generate some captions for a video.The main difference between a traditional computer software and a machine learning model is that a human has not written any code that tells the computer system how to tell the difference between the broccoli and the Beetroot.Instead the machine-learning model has been taught how to accurately differentiate between the Vegetables by training the model on a large amount of data, for example, a large number of photos with vegetables.

1.2 Deep learning

Deep learning is a sub-divison of machine learning. It is a branch that is found on learning and ameliorating on its very own by inspecting certain computer algorithms. While machine learning utilizes easier concepts, here deep learning performs with various neural networks, that are constructed to emulate how a human learns and thinks. Until now, these neural networks were just restricted by the computing power and consequently were confined in complexity. Advancements in the big data analytics have allowed greater, sophisticated neural networks, allowin the computers to think, learn, and react to byzantine situations much faster than us human. Deep learning has assisted in image classification, translation of language etc.

2. Types of Machine learning

1. Supervised Learning

In machine learning, this type of learning is the type where we can view that the learning is assisted by a someone, for example, a professor. We have the dataset which act as the professor and the role of a dataset i.e. the professor is to train the models or the machine. After the model got trained, the model can then process makin a prediction, decision when the newly data is fed to it.

2. Unsupervised Learning

A model learn throughs the observations and find structure in those data. When the model is given the dataset, it can automatically find certain pattern and relationship in that dataset by constructing some clusters in it. What the model can not do is that it can't add label to the cluster, like it could not say that this a group of vegetables or fruits, however it can discriminate all the vegetables from the fruits. For instance we present some images of cars, trucks and planes to a model, so what it will do, depending on some patterns and relationship, will create some cluster and divide that particular data set in to those clusters. Later on, if a fresh data is put into the model as input, it puts it to one of the already built clusters.

3. Reinforcement learning

It is a potential of an agent, to inter-relate with its environment and figure out what would be the perfect possible outcome. The agent follow the notion of the hit and trial method. The agent is then guerdoned with one point for one correct or penalized by one point for one wrong answer, and on the basis of the positive reward points gained by the agent, the model will train itself. Then once again, when the model is trained, the model would get prepared to make predictions regarding the fresh data given to it.

3. Applications of Machine Learning

Machine learning is probably the most exhilarating technologies that one has come across. As obvious from its name, it makes a computer similar to a human being. Mahcine learning is used extensively all around. Some of its applications are

1. Online Fraud Detection
2. Spam detector
3. Predictions while computing
4. Virtual Personal Assistants
5. Image recognition
6. Search Engine result refining

4. Steps for machine learning

The process for machine learning occurs in 4 important steps-

1. **Gathering of data-** the first and foremost step is to fed some data to the computer through which our model learns. The data could be in any type of format such as text files or excel sheets.

2. **Data preparation-** this step involves taking only the useful information from the whole data set which is fed into the computer. The data which is really useful for the model for processing is only taken into consideration and the rest is discarded. This step also involves removing the unwanted data, checking for values which are missing and treatment of the outliers.

3. **Training set and testing set-** after the data is filtered in the second step, the data is then divided into 2 sets. One of the sets is called training set which is used for creating the model and the second set, i.e. test set is then used to check the accuracy of the model created.

4. Evaluation- this step involves Evaluating the model. To verify the accuracy of the created model, the model is tested on some data which was not present in the data used for its creation. Here, the test data is used.

5. Convolutional Neural Network

A **Convolutional Neural Network also called ConvNet or CNN** is a Deep Learning algorithm which takes in an image as input and then assigns importance i.e. weights and biases to certain aspects or objects in the image and be able to differentiate them from the other. The pre-processing required in a Convolutional neural network is lower as compared to the other classification algorithms available. Technically, deep learning CNN models take each image as input and then pass it through a number of convolution layers with filters (also called Kernals), Pooling layers, fully connected layers (FC) and then apply activation functions to classify the object with probabilistic values between 0 and 1.

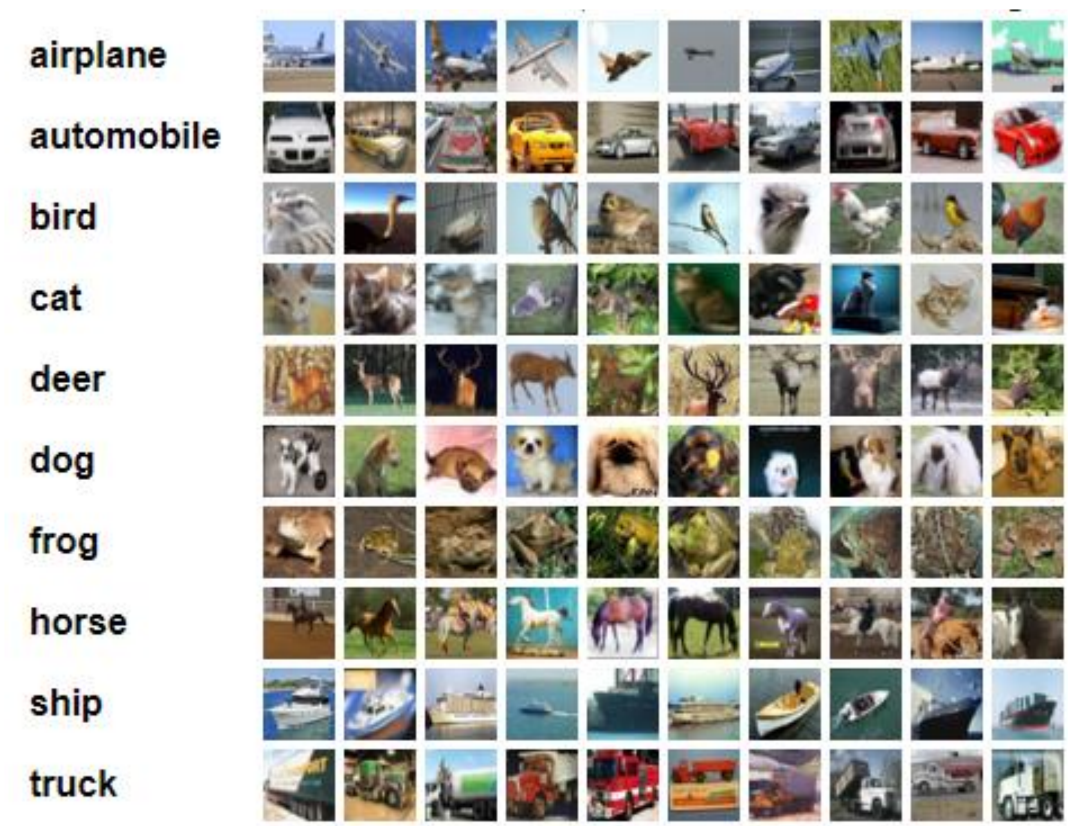
6. Steps for building a Convolutional neural network

1. Firstly we provide an input image into the convolution layer
2. Next, we choose some parameters, apply filters with paddings and strides, if required. We then perform convolution on the input image and apply activation functions.
3. We then need to perform pooling to reduce dimensionality size
4. Many convolution layers can be added as per need
5. Flatten the output and feed into a dense layer
6. Finally, train the model using the training set.

7. About Dataset

In this project, I have used the CIFAR-10 dataset. This dataset consists of 60 thousand 32x32 rgb images with 10 classes, each class having 6000 images per class. There are 50 thousand training images and 10 thousand test images. This dataset was collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton.

The classes are completely mutually exclusive. There is no overlap between the classes, automobiles and trucks.



8. Methodology

In this image classification task, I have used my knowledge of convolutional neural networks and with the help of TensorFlow and Keras, have created a model to classify the images in the CIFAR-10 dataset using the jupyter notebook.

Importing libraries

In [2]:

```
import tensorflow as tf
import os
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline
if not os.path.isdir('models'):
    os.mkdir('models')
print('Tensorflow version:', tf.__version__)
print('Is using gpu', tf.test.is_gpu_available())
```

```
Tensorflow version: 2.1.0
Is using gpu False
```

In the first code block, I have imported some libraries necessary for this task.

1. **TensorFlow-** TensorFlow is a Python library which is imported for fast numerical computing. It was created and released by Google. TensorFlow is a foundation library which can is used to build models of deep learning directly or by using some wrapper libraries that can simplify the process built on top of TensorFlow.

2. **OS-** The OS library in python provides useful functions for interacting with one's operating system. The OS module comes under the Python's standard utility modules. This module provides a portable way of using the operating system's dependent functionality. The 'os' and 'os.path' modules subsumes various functions to interact with the file system.

3. **Numpy-** The Numpy library is the core library used for scientific computing in Python. This library provides a high-performance multidimensional array object, and various useful tools for working with such arrays.

4. **Matplotlib-** this library is a visualization library. One of the many benefits of visualization is that it allows us visual access to a large amount of data in easily digestible visuals such as line, bar, scatter, histogram etc.

Pre-process our data

```
In [3]: def get_three_classes(x,y):
            indices_0,_=np.where(y==0.)
            indices_1,_=np.where(y==1.)
            indices_2,_=np.where(y==2.)
            indices=np.concatenate([indices_0,indices_1,indices_2],axis=0)
            x=x[indices]
            y=y[indices]
            count=x.shape[0]

            indices=np.random.choice(range(count),count,replace=False)

            x=x[indices]
            y=y[indices]
            y=tf.keras.utils.to_categorical(y)
            return x,y
```

In the next block of code, I have created a function called get_three_classes to pre-process the CIFAR-10 dataset. For simplification, I have used only the first 3 classes in the dataset i.e. ‘Airplane’, ‘Bird’ and ‘Automobiles’. After selecting the first three classes, it is necessary to remove the unwanted data i.e. the images for the other classes as they will be superfluous to our model. Using functions from the Numpy module, images corresponding to the 3 selected classes are selected and then are shuffled randomly and stored. Lastly, the labels are converted to their numerical representation using One-Hot encoding. Machines understand numbers and not text. Hence, we need to convert each text category i.e. our classes, to numbers in order for the machine to process them using mathematical equations.

Loading the CIFAR-10 dataset

```
In [4]: (x_train,y_train),(x_test,y_test)=tf.keras.datasets.cifar10.load_data()
        x_train,y_train=get_three_classes(x_train,y_train)
        x_test,y_test=get_three_classes(x_test,y_test)
        print(x_train.shape,y_train.shape)
        print(x_test.shape,y_test.shape)

        (15000, 32, 32, 3) (15000, 3)
        (3000, 32, 32, 3) (3000, 3)
```

My next step was to load the CIFAR-10 dataset from Keras. A Deep neural network, being the current passion, the convolution of its extensive frameworks have always been a hurdle for their application for all the fledglings to machine learning. Also, there were proposals for up-graded and simplifed high level API for building neural network models. Keras, one of the many prime high level neural networks API, is written in the Python language and aids many back-end neural network computation engines. On loading the dataset, the dataset was divided into the training set and test set. Using the get_three_classes function defined previously, we select the first classes and get their samples and put them in the train and test set. On printing the summary of our train and test sets using the shape function, we see that our train set has 15,000 examples, 32x32 is the size of the images and 3 means rgb i.e. the pictures are colored. Our test set has 3,000 examples.

Visualize our data

In [6]:
```
class_names=['Airplane','Car','Bird']
def show_random_examples(x,y,p):
    indices=np.random.choice(range(x.shape[0]),10,replace=False)


    x=x[indices]
    y=y[indices]
    p=p[indices]
    plt.figure(figsize=(10,5))
    for i in range (10):
        plt.subplot(2,5,1+i)
        plt.imshow(x[i])
        plt.xticks([])
        plt.yticks([])
        col='green' if np.argmax(y[i])==np.argmax(p[i]) else 'red'
        plt.xlabel(class_names[np.argmax(p[i])],color=col)
    plt.show()
show_random_examples(x_train,y_train,y_train)
```

Visualizing data can be helpful in many ways. Here, I have created another function show_random_exmaples to visualize my new data using the matplot library. After creating the model, this function will be used for displaying the predictions. Currently, I have displayed randomly any 10 images with their labels just to verify if up till now the code is working fine. Using the functions from matplot library, this function would display those 10 random examples i.e. the images from the dataset and their labels either in green or red color. Green color will signify that the model classified the image correctly and the red color will signify that the model classified the image wrongly.


```
In [7]: show_random_examples(x_test,y_test,y_test)
```

My next step was to create the model. In the first two lines, I simply imported different layers from keras.

1. **Convolutional layer** is the layer where the required filters are applied to the original input.
2. A **pooling layer** is used to reduce the spatial size of the representation to reduce the amount of parameters and computation required in the network.
3. **Batch normalization** is a technique used for improving the performance, speed, and stability of the deep neural networks. As the name suggests, it used to normalize the input layer by re-centering and re-scaling.

4. **Dropout** is simply a regularization technique. A simple way to avoid over-fitting. Using this, randomly selected neurons are ignored during training.
5. **Flattening** is used to transform a 2-dimensional array of features to a 1-D array that can be used as input to a fully connected neural network layer. It used before the FC layer.
6. **Dense layer** is basic deeply connect neural network layer.. This layer performs the operation below with the input and then returns one output.

output = activation(dot(input, kernel) + bias

Creating our model (architecure of our model)

In [8]:

```
from tensorflow.keras.layers import Conv2D, MaxPooling2D, BatchNormalization
from tensorflow.keras.layers import Dropout, Flatten, Input, Dense
def create_model():
    def add_conv_block(model, num_filters):
        model.add(Conv2D(num_filters,3,activation='relu',padding='same'))
        model.add(BatchNormalization())
        model.add(Conv2D(num_filters,3,activation='relu'))
        model.add(MaxPooling2D(pool_size=2))
        model.add(Dropout(0.5))
        return model
    model=tf.keras.models.Sequential()
    model.add(Input(shape=(32,32,3)))
    model=add_conv_block(model,32)
    model=add_conv_block(model,64)
    model=add_conv_block(model,128)
    model.add(Flatten())
    model.add(Dense(3,activation='softmax'))
    model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
    return model
model=create_model()
model.summary()
```

After importing the layers from keras, I created two more functions called add_conv_block and create_model, using which the model is created. Then the summary of the model is displayed.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896       
_________________________________________________________________
batch_normalization (BatchNo (None, 32, 32, 32)        128       
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 30, 30, 32)        9248      
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 15, 15, 32)        0         
_________________________________________________________________
dropout (Dropout)            (None, 15, 15, 32)        0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 15, 15, 64)        18496     
_________________________________________________________________
batch_normalization_1 (Batch (None, 15, 15, 64)        256       
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 13, 13, 64)        36928     
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 6, 6, 64)          0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 6, 6, 64)          0         
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 6, 6, 128)         73856     
_________________________________________________________________
batch_normalization_2 (Batch (None, 6, 6, 128)         512       
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 4, 4, 128)         147584    
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 2, 2, 128)         0         
_________________________________________________________________
dropout_2 (Dropout)          (None, 2, 2, 128)         0         
_________________________________________________________________
flatten (Flatten)            (None, 512)               0         
_________________________________________________________________
dense (Dense)                (None, 3)                 1539      
=================================================================
Total params: 289,443
Trainable params: 288,995
Non-trainable params: 448
_________________________________________________________________
```

Training our model

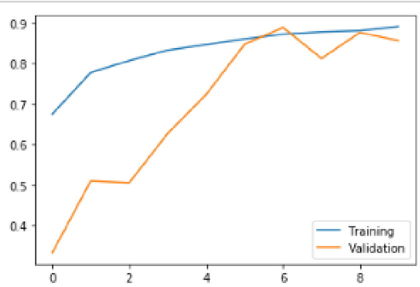
In [9]:
```
h=model.fit(
  x_train/255.,y_train,
  validation_data=(x_test/255.,y_test),
  epochs=10,batch_size=128,
  callbacks=[tf.keras.callbacks.EarlyStopping(monitor='val_accuracy',patience=3),
            tf.keras.callbacks.ModelCheckpoint('models/model_{val_accuracy:.3f}.h5',
                                               save_best_only=True,save_weights_only=False,
                                               monitor='val_accuracy')])
```

After the creation of the model, I had to fit the model on the validation data using the validation set. For testing, I normalized the set. I set the epoch value to 10 and batch size to 128. I used EarlyStopping callback to stop the training when my monitored metric i.e accuracy has stopped improving. The Patience parameter was set to 3 which mean that after 3 numbers of epochs, if there is no improvement in the val_accuracy, the training will be halted. In addition to the EarlyStopping callback, I also used ModelCheckpoint to save my model or weights in a different file at some interval, so the model or weights can be loaded later again. I saved my model in a separate file called “models”.

```
Train on 15000 samples, validate on 3000 samples
Epoch 1/10
15000/15000 [==============================] - 67s 4ms/sample - loss: 0.8761 - accuracy: 0.6753 - val_loss: 2.1125 - val_accura
cy: 0.3333
Epoch 2/10
15000/15000 [==============================] - 65s 4ms/sample - loss: 0.5437 - accuracy: 0.7777 - val_loss: 1.9272 - val_accura
cy: 0.5103
Epoch 3/10
15000/15000 [==============================] - 70s 5ms/sample - loss: 0.4787 - accuracy: 0.8071 - val_loss: 1.6110 - val_accura
cy: 0.5057
Epoch 4/10
15000/15000 [==============================] - 70s 5ms/sample - loss: 0.4250 - accuracy: 0.8327 - val_loss: 0.8738 - val_accura
cy: 0.6273
Epoch 5/10
15000/15000 [==============================] - 71s 5ms/sample - loss: 0.3914 - accuracy: 0.8465 - val_loss: 0.7443 - val_accura
cy: 0.7233
Epoch 6/10
15000/15000 [==============================] - 68s 5ms/sample - loss: 0.3652 - accuracy: 0.8603 - val_loss: 0.4366 - val_accura
cy: 0.8473
Epoch 7/10
15000/15000 [==============================] - 67s 4ms/sample - loss: 0.3292 - accuracy: 0.8722 - val_loss: 0.2902 - val_accura
cy: 0.8887
Epoch 8/10
15000/15000 [==============================] - 67s 4ms/sample - loss: 0.3199 - accuracy: 0.8780 - val_loss: 0.6152 - val_accura
cy: 0.8123
Epoch 9/10
15000/15000 [==============================] - 66s 4ms/sample - loss: 0.3096 - accuracy: 0.8811 - val_loss: 0.3315 - val_accura
cy: 0.8763
Epoch 10/10
15000/15000 [==============================] - 71s 5ms/sample - loss: 0.2856 - accuracy: 0.8907 - val_loss: 0.4041 - val_accura
cy: 0.8563
```

After fitting my model, using the matplot library, I plotted a graph showing the validation data and training data with the the val_accuracy and accuracy.

```
In [11]: accs=h.history['accuracy']
         val_accs=h.history['val_accuracy']
         plt.plot(range(len(accs)),accs,label='Training')
         plt.plot(range(len(accs)),val_accs,label='Validation')
         plt.legend()
         plt.show()
```



As the final step, I loaded the best model i.e. the one with the highest accuracy and then using the predict function, my model made the predictions and with the help of the show_random_examples, the predictions are displayed.

Predicitons

```
In [15]: model=tf.keras.models.load_model('models/model_0.892.h5')
```

```
In [16]: preds=model.predict(x_test/255.)
```

```
In [19]: show_random_examples(x_test,y_test,preds)
```

9. Conclusion

Machine learning is rapidly developing field in Computer Engineering. It has applications in about each other field of study and is as of now being actualized industrially in light of the fact that machine learning can take care of issues excessively troublesome or tedious for people to explain. An advantage to utilizing a convolution neural system is that it is intended to all the more likely handle picture and speech recognition tasks. Rather than hidden layers, convolutional neural networks have a convolutional and pooling layer. It is a result of these layers that convolutional neural networks are favored for Image classification tasks.

10. References

[1] Manohar Swamynathan. "Mastering Machine Learning with Python in Six Steps", Springer Science and Business Media LLC, 2017

[2] Sheryl May D. Lainez, Dennis B. Gonzales. "Automated Fingerlings Counting Using Convolutional Neural Network", 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), 2019

[3] "Classification and Stage Prediction of Lung Cancer using Convolutional Neural Networks", International Journal of Innovative Technology and Exploring Engineering, 2019

[4] Jinjing Zhang, Fei Hu, Li Li, Xiaofei Xu, Zhanbo Yang, Yanbin Chen. "An adaptive mechanism to achieve learning rate dynamically", Neural Computing and Applications, 2018

[5] machinelearningmastery.com

[6] www.coursehero.com

[7] blog.kaggle.com

B Tech CSE NTCC 7 Sem 2020-21

by Ankita Singh

Submission date: 13-Jun-2020 02:00AM (UTC+0530)

Submission ID: 1342762945

File name: Ankita_Singh_NTCC_7_Sem.docx (1.67M)

Word count: 2279

Character count: 11438

B Tech CSE NTCC 7 Sem 2020-21

ORIGINALITY REPORT

15%	11%	8%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.coursehero.com Internet Source	2%
2	wp.wwu.edu Internet Source	2%
3	machinelearningmastery.com Internet Source	2%
4	towardsdatascience.com Internet Source	1%
5	Manohar Swamynathan. "Mastering Machine Learning with Python in Six Steps", Springer Science and Business Media LLC, 2017 Publication	1%
6	Sheryl May D. Lainez, Dennis B. Gonzales. "Automated Fingerlings Counting Using Convolutional Neural Network", 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), 2019 Publication	1%
7	blog.kaggle.com	

Internet Source

1%

8	Keita Kinjo, Takeshi Ebina. "Case-based decision model matches ideal point model:", Journal of Intelligent Information Systems, 2017 Publication	1%
9	Mohammad Farukh Hashmi, Aashish Kumar, Avinash G. Keskar. "chapter 6 Subjective and Objective Assessment for Variation of Plant Nitrogen Content to Air Pollutants Using Machine Intelligence", IGI Global, 2020 Publication	1%
10	tel.archives-ouvertes.fr Internet Source	1%
11	"Classification and Stage Prediction of Lung Cancer using Convolutional Neural Networks", International Journal of Innovative Technology and Exploring Engineering, 2019 Publication	1%
12	pure.aber.ac.uk Internet Source	1%
13	Jinjing Zhang, Fei Hu, Li Li, Xiaofei Xu, Zhanbo Yang, Yanbin Chen. "An adaptive mechanism to achieve learning rate dynamically", Neural Computing and Applications, 2018 Publication	1%

14	repository.ihu.edu.gr Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		