T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

IMAGE CLASSIFICATION WITH JPG DATA

ESRA ERYILMAZ

SUPERVISOR
PROF. DR. YUSUF SİNAN AKGÜL

GEBZE
JANUARY, 2023

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

IMAGE CLASSIFICATION WITH JPG DATA

ESRA ERYILMAZ

SUPERVISOR
PROF. DR. YUSUF SİNAN AKGÜL

JANUARY, 2023
GEBZE

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 26/09/2022 by the following jury.

JURY

Member
(Supervisor) : Prof. Dr. Yusuf Sinan Akgül

Member : Prof. Dr. İbrahim Soğukpınar

ABSTRACT

Image classification has drawn a lot of attention over past few years, and with the advent of deep learning impressive performances have been achieved with numerous industrial applications. Most of these deep learning models rely on RGB images to classify the image. However in some applications, images are compressed either for storage savings or fast transmission.

In this project, we passed time consuming image decompression step and trained commonly used deep learning models with compressed images. We train a neural network with a dataset based on the blockwise DCT coefficients (the JPG compression algorithm).

Keywords: Image classification, deep learning, compressed data, DCT method

ÖZET

Görüntü sınıflandırma, son birkaç yılda çok fazla dikkat çekti ve derin öğrenmenin ortaya çıkmasıyla birlikte, çok sayıda endüstriyel uygulamada etkileyici performanslar elde edildi. Bu derin öğrenme modellerinin çoğu görüntüyü sınıflandırmak için RGB görüntülerine dayanmaktadır. Ancak bazı uygulamalarda, görüntüler depolama tasarrufu veya hızlı iletim için sıkıştırılır.

Bu projede, zaman alan görüntü açma adımını atladık ve yaygın olarak kullanılan derin öğrenme modellerini sıkıştırılmış görüntülerle eğittik. Blok yönlü DCT katsayılarına (JPG sıkıştırma algoritması) dayalı bir veriseti ile sinir ağı eğittik.

Anahtar Kelimeler: Görüntü sınıflandırma, derin öğrenme, sıkıştırılmış veri, DCT metodu

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my supervisor Prof. Dr. Yusuf Sinan Akgül for his able guidance and support in completing my project. His guidance helped me in all the time of research and writing of this thesis.

I am also indebted to my family and friends for their invaluable support, advice and love during my education.

Esra Eryılmaz

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol or Abbreviation	:	**Explanation**
DCT	:	Discrete Cosine Transform
JPG (or JPEG)	:	Joint Photographic Experts Group
PNG	:	Portable Network Graphics
GIF	:	Graphics Interchange Format
RGB	:	Red Green Blue
MPEG-4	:	Moving Pictures Expert Group 4
CNN	:	Convolutional Neural Network
R-CNN	:	Region Based Convolutional Neural Network
DNN	:	Deep Neural Network
RNN	:	Recurrent Neural Network
LSTM	:	Long Short-Term Memory
AI	:	Artificial Intelligence
MS COCO	:	Microsoft Common Objects in Context
YOLO	:	You Only Look Once (Real-time Object Detection)
GUI	:	Graphical User Interface

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Project Description

Convolutional neural networks are now the reference models in image classification 1.1 or object detection tasks. The achieved levels of performance and robustness allow for usage of CNN models.
There are many applications for image classification; popular use cases include:

- Automated inspection and quality control
- Object recognition in driverless cars
- Detection of cancer cells in pathology slides
- Face recognition in security
- Traffic monitoring and congestion detection
- Retail customer segmentation
- Land use mapping

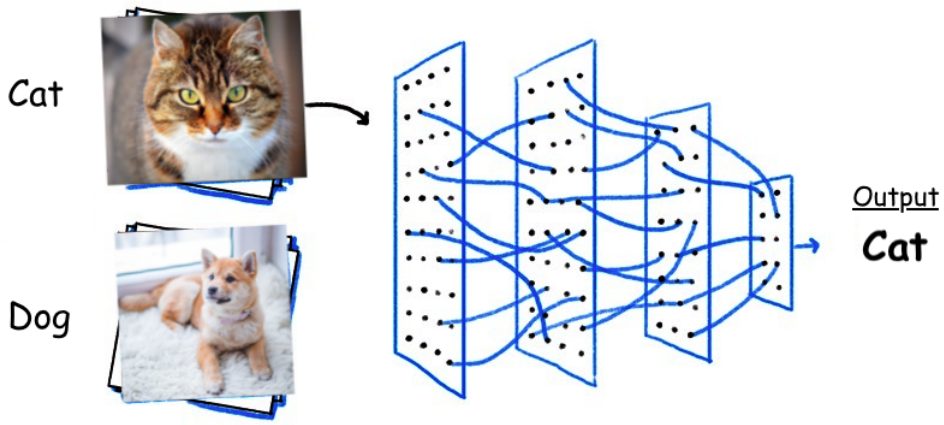

Figure 1.1: Image classification

In many contexts either for storage or fast transmission purposes, it is common to compress the images or videos in formats such as JPEG, PNG, GIF for images, and MPEG-4 Part2 or H.264 for videos.

Some of the most popular compression algorithms such as JPEG, mpeg4 part2 or H.264 share a common encoding strategy based on a block transformation of the images. Applying generic convolutional neural networks (CNN) would require a decoding step to RGB format, which is computationally costly and memory demanding. [1]

1.2. Project Purpose

In this project our aim is to study effects of JPG compression on image classification with CNNs. And using DCT method when performing image compression.

We want to shorten the training time by using the most common image classification networks on compressed images.

Additionally, our aim in accuracy result is that the difference in average loss of accuracy between RGB and JPG images should not be large.

2. LITERATURE REVIEW

Image classification is widely used and solved problem. A lot of work has been done on this subject and continues to be done. It can be solved on different image formats by using different algorithms.

2.1. Image Compression

Image compression is an application of data compression that encodes the original image with few bits. The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form. It affects the tiles of raster before storing them in geodatabase. It reduces the size of a file or database. Compression improves data handling, storage and database performance.

A variety of techniques ae available for image compression. Compression techniques can be lossless or lossy. 2.1

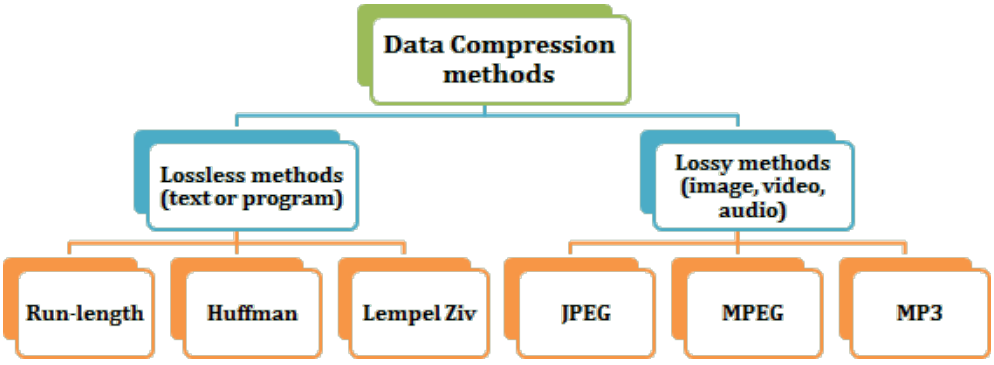

Figure 2.1: Data compression methods

2.1.1. JPG Compression

JPG (or JPEG) is a standard and widely-used image encoding and compression technique consists of the following steps: [2]

1. Converting the given image from RGB to YCbCr color space: this is done because the human visual system relies more on spatial content and acuity than it does on color for interpretation. Converting the color space isolates these components which are of more import.

2. Performing spatial subsampling of the chrominance channels in the YCbCr space: the human eye is much more sensitive to changes in luminance, and downsampling the chrominance information does not affect the human perception of the image very much.

3. Transforming a blocked representation of the YCbCr spatial image data to a frequency domain representation using Discrete Cosine Transform (DCT): this step allows the JPEG algorithm to further compress the image data as outlined in the next steps by computing DCT coefficients. 2.2

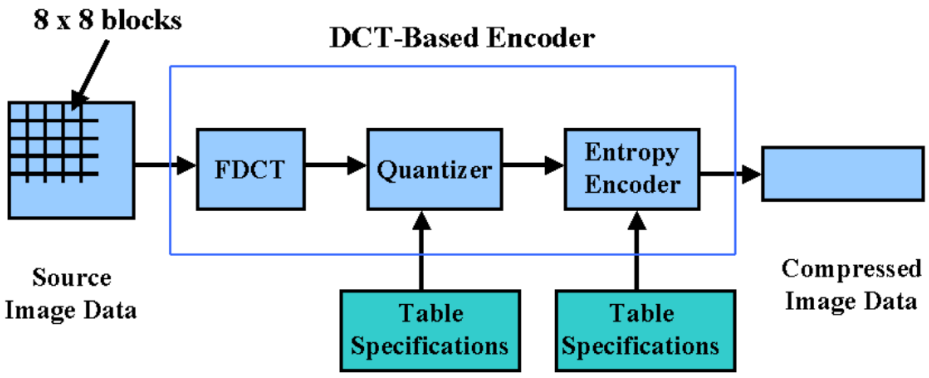

Figure 2.2: DCT

4. Performing quantization of the blocked frequency domain data according to a user defined quality factor: this is where the JPEG algorithm achieves majority of the compression, at the expense of image quality. This step suppresses higher frequencies more since these coefficients contribute less to the human perception of the image.

2.2. Image Classification without Deep Learning

There are many algorithms that can be used to perform classifications. Before the deep learning era, people did have image classifiers, which did not use neural networks because all the thing that makes deep learning possible were not invented yet.

People created feature extractors such as edge-detectors by hand, completely hardcoded, and used them to classify images. Even when people used machine learning techniques, all the focus was on crafting the feature detectors, and very less focus was put into the machine learning thing.

But such solutions were applicable to very narrow, limited fields, and lacked robustness. That changed with the application of deep learning in computer vision

problems.

2.3. Image Classification with Deep Learning

Image classification with machine learning leverages the potential of algorithms to learn hidden knowledge from a dataset of organized and unorganized samples (Supervised Learning). The most popular machine learning technique is deep learning, where a lot of hidden layers are used in a model. [3]

Deep learning brought great successes in the entire field of image recognition, face recognition, and image classification algorithms achieve above human-level performance and real-time object detection.

Additionally, there's been a huge jump in algorithm inference performance over the last few years.

- For example, in 2017, the Mask R-CNN algorithm was the fastest real-time object detector on the MS COCO benchmark, with an inference time of 330 ms per frame.
- In comparison, the YOLOR algorithm released in 2021 achieves inference times of 12 ms on the same benchmark, thereby overtaking the popular YOLOv3 and YOLOv4 deep learning algorithms.
- In July 2022, the release of YOLOv7 marked a new state-of-the-art that surpasses all previously known models, including YOLOR, in terms of speed and accuracy. 2.3

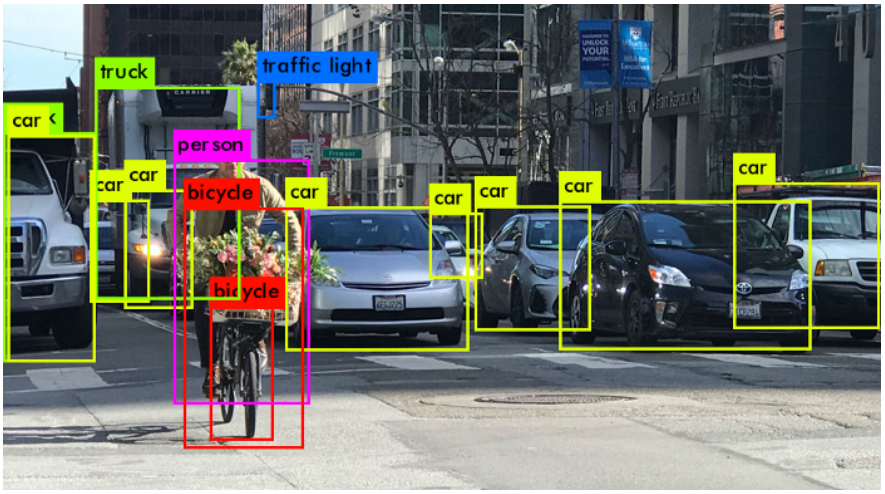

Figure 2.3: YOLO object detection example

2.3.1. Convolutional Neural Network (CNN)

A CNN is a framework developed using machine learning concepts. CNNs are able to learn and train from data on their own without the need for human intervention.

In fact, there is only some pre-processing needed when using CNNs. They develop and adapt their own image filters, which have to be carefully coded for most algorithms and models. CNN frameworks have a set of layers that perform particular functions to enable the CNN to perform these functions. [4]

2.3.2. Deep Neural Networks (DNN)

DNNs are just an umbrella term for a bunch of different neural network architectures, one of which is a CNN. DNN refers to any neural network that has more than one hidden layer. 2.4

While DNN uses many fully-connected layers, CNN contains mostly convolutional layers. The form of deep neural network known as ”convolutional” is particularly well-suited for image recognition because of its convolutional components. A convolutional neural network employs convolutional and pooling layers, which represent the translation invariant nature of most pictures. Because they inherently capture the structure of pictures, CNNs are a better fit for image classification task than generic DNNs. [5]

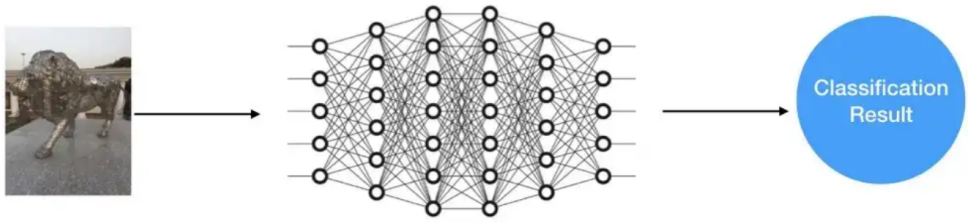

Figure 2.4: DNN

2.3.3. Long Short-Term Memory Networks (LSTM)

LSTMs are mainly used to learn, process, and classify sequential data because these networks can learn long-term dependencies between time steps of data. Common LSTM applications include sentiment analysis, language modeling, speech recognition, and video analysis. 2.5

The integration of the rudimentary Convolutional Neural Network (CNN) with Long Short-Term Memory (LSTM), resulting in a new paradigm in the well-explored field of image classification.

LSTM is one kind of Recurrent Neural Network (RNN) which has the potential to memorize long-term dependencies. It was observed that LSTMs are able to complement the feature extraction ability of CNN when used in a layered order. LSTMs have the capacity to selectively remember patterns for a long duration of time and CNNs are able to extract the important features out of it. This LSTM-CNN layered structure, when used for image classification, has an edge over conventional CNN classifier. The model which has been proposed is based on the sets of Artificial Neural Network like Recurrent and Convolutional neural network; hence this model is robust and suitable to a wide spectrum of classification tasks. [6]

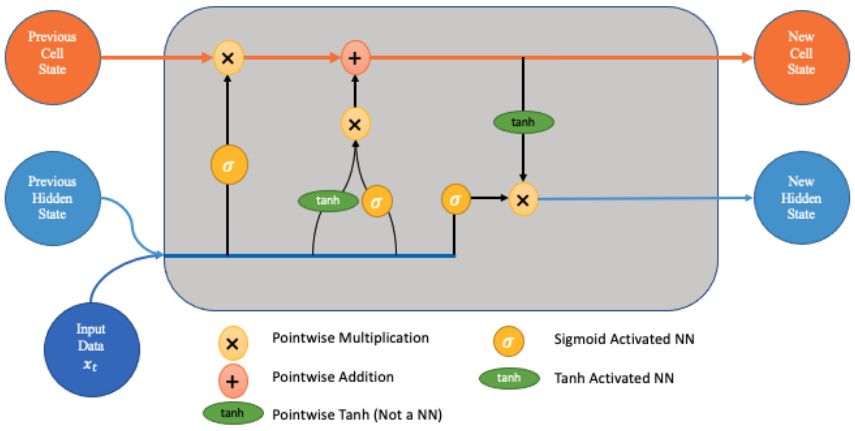

Figure 2.5: LSTM network illustration

2.3.4. Recurrent Neural Networks (RNN)

A RNN is a type of artificial neural network commonly used in speech recognition and natural language processing. Recurrent neural networks recognize data's sequential characteristics and use patterns to predict the next likely scenario. 2.6

Combined with CNNs, the model called CNN-RNN makes image classification task more efficient. Image data can be viewed as two-dimensional wave data, and convolution calculation is a filtering process. It can filter non-critical band information in an image, leaving behind important features of image information. The CNN-RNN model can use the RNN to Calculate the Dependency and Continuity Features of the Intermediate Layer Output of the CNN Model, connect the characteristics of these middle tiers to the final full-connection network for classification prediction, which

will result in better classification accuracy. [7], [8]

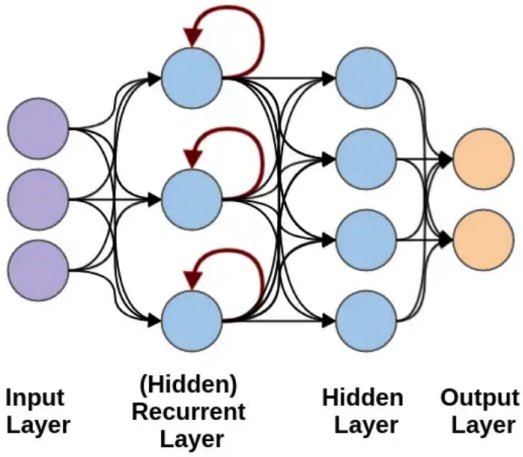

Figure 2.6: RNN

2.4. Image Classification on Compressed Data

There have been several works in recent years which attempt to merge deep learning with low-level JPEG primitives : [9]

- Ghosh and Chellappa [10] include a DCT as part of their initial layer and show a performance improvement on classification.
- Gueguen et al. [11] read JPEG DCT coefficients directly into their network and show that the DCT representation requires fewer parameters to learn comparable results to pixels, yielding a speed improvement.
- Ehrlich and Davis [12] formulate a fully JPEG domain residual network and again show a speed improvement.
- Lo and Hang [13] show a method for semantic segmentation on DCT coefficients and show both a performance and speed improvement over using pixels.
- Deguerre et al. [1] show a method for object detection on DCT coefficients and again show performance and speed improvement over pixels.
- Ehrlich et al. [14] formulate a JPEG artifact correction network on DCT coefficients and attain state-of-the-art results for color images.
- Choi and Han use a network to learn task-guided quantization matrices that maximize task performance after JPEG compression [15].

3. METHOD AND SYSTEM ARCHITECTURE

3.1. System Requirements

Software and hardware requirements :

1. High resolution image display,
2. Sufficient storage space,
3. Sufficient computer powering,
4. Memory transfer bandwidth,
5. TensorFlow version 2.9.2 [16]
6. Python version 3.6 (any Python version 3.x will be fine).[17]

3.2. Architecture

The CNN model has been used in the project. The CNN process includes different steps. If we summarize this process in general, the model given in Figure 3.1 can be used.

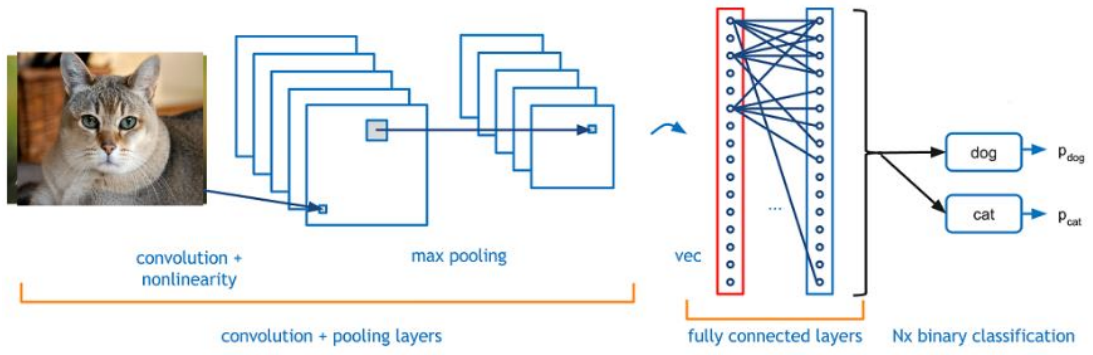

Figure 3.1: Cat vs Dog Image Classifier using CNN

3.2.1. Data Acquisition

The dataset was found on the internet (from the ImageNet website) [18]. It contains two classes (cat and dog) inside.

3.2.1.1. Dataset

The dataset containing 30.000 images is shown in the figure 3.2.

Figure 3.2: Some examples of Dogs vs Cats Dataset

3.2.2. Discrete Cosine Transform

DCT method was used to compress the images. The most well-known image compression format is JPEG.

In JPEG Compression, DCT operation starts with nonoverlapping 8×8 image data blocks to generate 64 DCT coefficients representing various horizontal, vertical and composite frequencies. Out of the 64 coefficients obtained, the less important frequencies are discarded during the quantization step and the important frequencies are used in the decompression step to retrieve the image. As DCT processes each of the three input channels separately, so in terms of convolution they are considered as three separate applications of convolution, and the information obtained from separate input channels stay separate. The forward DCT coefficients are obtained with the help

of this equation 1. [19]

$$F_{uv} = \frac{c_u c_v}{4} \sum_{i=1}^{7} \sum_{j=1}^{7} f(i,j) cos(\frac{(2m+1)u\pi}{16}) cos(\frac{(2n+1)v\pi}{16}) \quad (1)$$

3.2.3. Network Architecture

This network architecture includes the description of the layers used in our DCT-CNN model. [20] 3.3

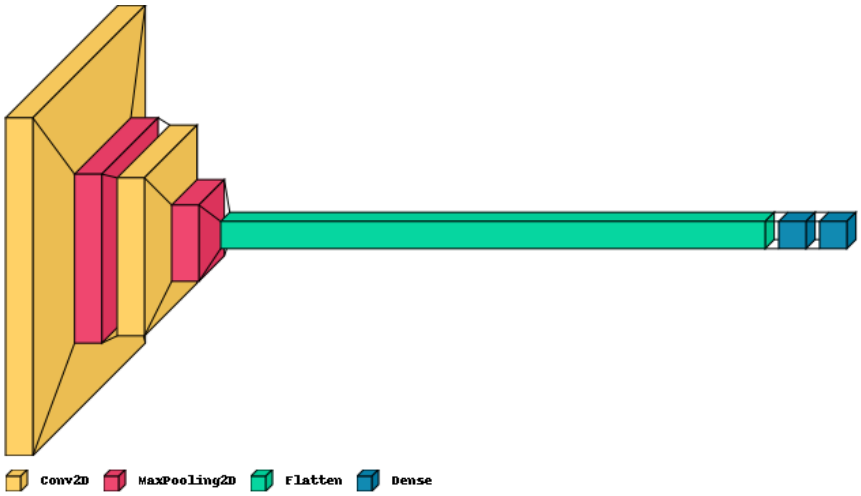

Figure 3.3: Layers visualization of cnn architecture

3.2.3.1. Convolutional Layer

The convolutional layer consists of various filters with weights and bias. The convolutional layer is responsible for extracting various features such as edges, texture, colour, shapes etc. Our model consists of 2 convolutional layers. The first layer learns the lower level features and has 32 filters. The higher level representation and the new association between the feature map are learned by the other hidden layer which has 32 filters. The kernel size of each filter is 3×3 and a default stride is used. We used rectified linear units (ReLU) as an activation function to make neurons respond selectively to its input and to bring non-linearity so that our model learn the required relation between input and output.

3.2.3.2. Max Pooling

At all the 2 convolutional layers, max polling is applied, it reduces the spatial dimensions of features and speed up the whole training. Any typical complex image convolution needs the more filters to find various significant patterns in the image. As a result of that output dimension increases. However, sometimes it is necessary to use more parameters, it may also result in overfitting that reduces the model generality. So, the problem can be solved by reducing the spatial dimension and that's the role of max pooling. However max pooling attains the maximum value within its neighbourhood and makes the computation of larger problem easier.

3.2.3.3. Fully Connected Layer

All the convolution layers before fully connected layers hold information about the local features of an image. All the output of these layers are mixed and fed as input to classification block consisting of fully connected neural network. Fully connected layer extracts the beneficial information from them and uses it for higher representation. The fully connected layer represents the feature vector of an image, where feature vector holds all the essential information of the image. During training these features determines the loss and help the model to train well. Similarly, during testing also as a trained model, these features are used for classification of the images. In our model all the last layer are fully connected layers. 3.4 [21]

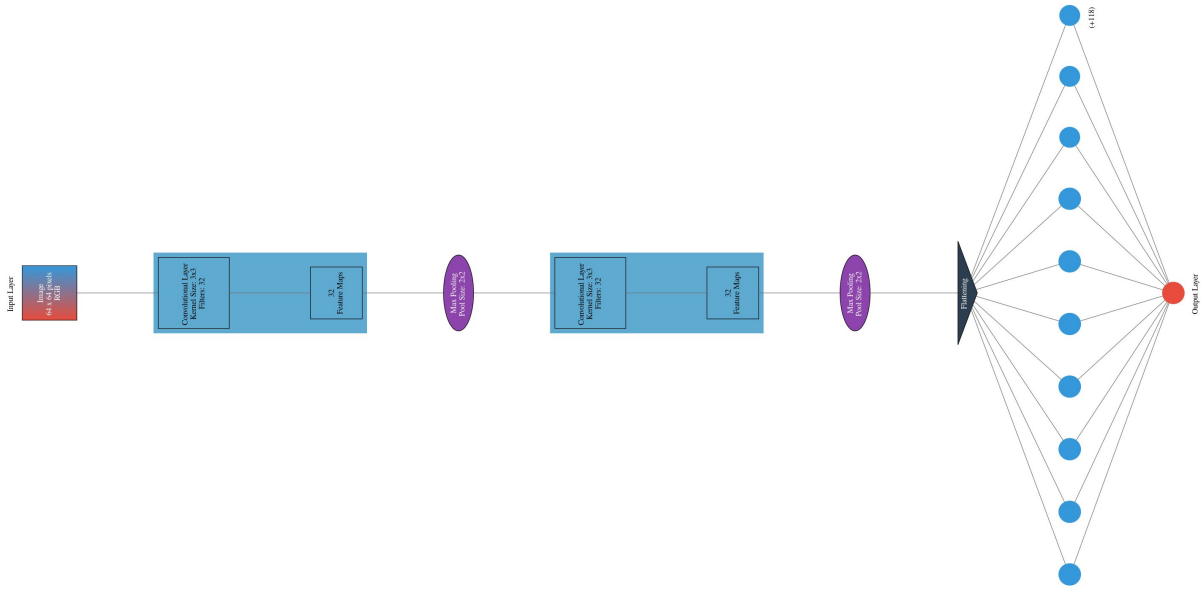

Figure 3.4: Network summary visualization

3.3. Implementation Details

I used CNN for image classification model. CNN model implementation source code is here :

```
# Initialising the CNN
cnn = tf.keras.models.Sequential()

# Step 1 - Convolution
cnn.add(tf.keras.layers.Conv2D(filters=32, kernel_size=3, activation
    ='relu', input_shape=[64, 64, 3]))

# Step 2 - Pooling
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))

# Adding a second convolutional layer
cnn.add(tf.keras.layers.Conv2D(filters=32, kernel_size=3, activation
    ='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))

# Step 3 - Flattening
cnn.add(tf.keras.layers.Flatten())

# Step 4 - Full Connection
cnn.add(tf.keras.layers.Dense(units=128, activation='relu'))

# Step 5 - Output Layer
cnn.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))
```

Listing 3.1: CNN model

3.4. User Interface

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications.
[22][23]

The interface was created using tkinter (Tk). This interface allows to upload an image with ”Upload Image” button. With ”DCT”, ”DCT (dim/4)”, ”DCT (dim/8)” buttons it allows upload an DCT applied image. With ”INFO” button it shows information about images. With ”CLASSIFY” button it writes whether the uploaded image

is a cat or a dog on the label. 3.5

Also you can access the trailer video showing the interface at the URL : `https://youtu.be/tLzQcWJJSxQ`

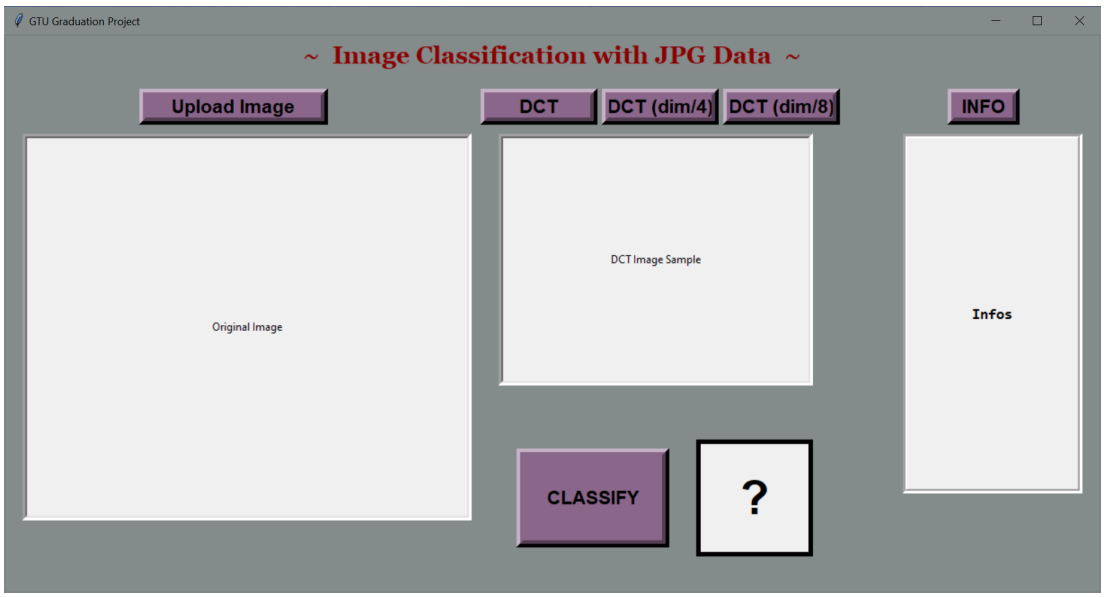

Figure 3.5: User Interface

The sample output is shown in Figure 3.6

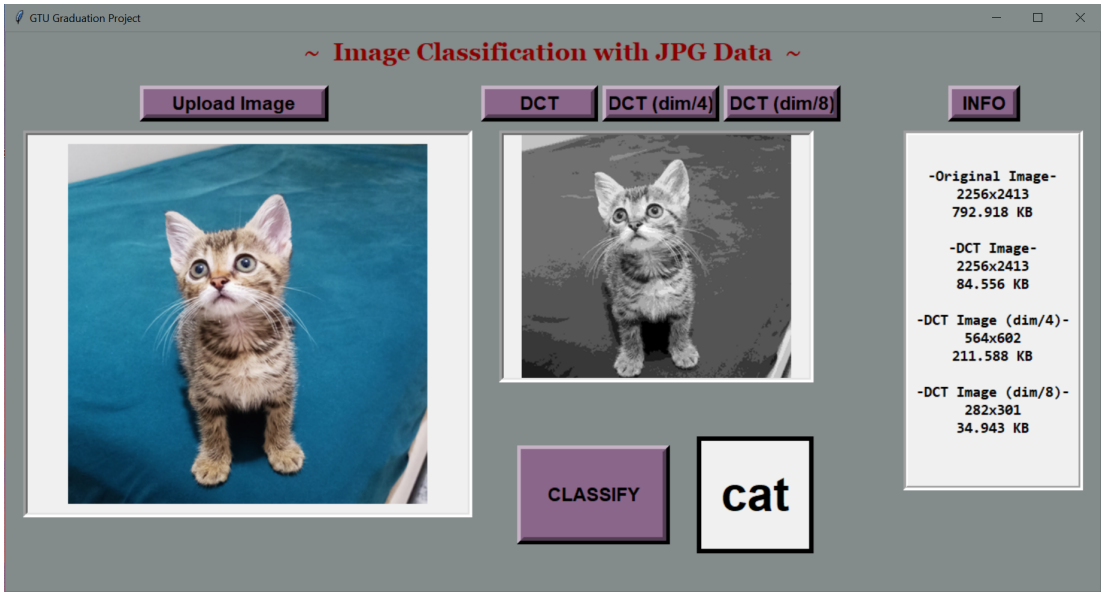

Figure 3.6: User Interface Output Sample

4. EXPERIMENTS AND RESULTS

- On the first practices, I was training a deep learning model for image classification. I was trying to classify two-classes data: happy and unhappy.
- Then I decided to change my dataset. I found a dataset contains 30.000 images of cats and dogs. And I trained a CNN model for it. But in the beginning, I tried on a small part of the dataset because the dataset was large and took a long time to train.
- Then, I was keeping my images in 5 different folders with 5 different qualities and trained that way. In general as we can see here 4.1 the accuracy results almost the same.

Image Quality	Accuracy
%100	0.893
%50	0.890
%30	0.888
%10	0.880
%1	0.902

Table 4.1: In different quality accuracy results

- After these image classification exercises, I moved on to the image compression part. I used the DCT method for this. I compressed all images using this method. There was no change in the dimension size of the output of the DCT method. Experiments were also made by reducing the dimension of the images.
  The dimension size changes made as follows:

  In the DCT method, some mathematical operations are applied to each 8x8 block and put back in the same place. At the end of this process, the numbers in the upper right corner of each 8x8 block matrix are valuable. For this reason while reducing the size by 8 times, the number in the upper right corner of each 8x8 matrix was taken and converted into an image. 4.1
- After applying the DCT method, I gave the images of different dimensions to the CNN model that I used. Then I recorded the results. You can see the results in this 4.2 table.

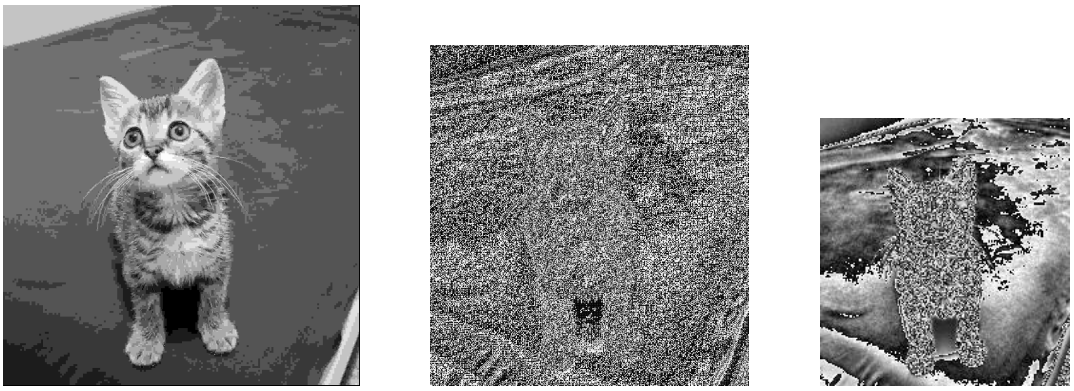

Figure 4.1: Images with DCTs of different sizes

	Accuracy	Training time
Original images	30 epoch → 0.92 | 0.91	9540 second | 12055 second
	20 epoch → 0.89	10068 second
Images with DCT (dimension/8)	30 epoch → 0.77 | 0.68 | 0.76	5309 second | 5953 second | 6530 second
	25 epoch → 0.70	1474 second
	20 epoch → 0.67	1162 second
Images with DCT (dimension/4)	30 epoch → 0.71 | 0.70	6901 second | 10036 second
	20 epoch → 0.68	9187 second

Table 4.2: Result table

4.1. Performance Measurements

In order to measure performance as mentioned above, deep learning results were considered. For this, accuracy and loss values were taken into account. The graphs of the experiment results are shown in the figures below.

4.24.34.4

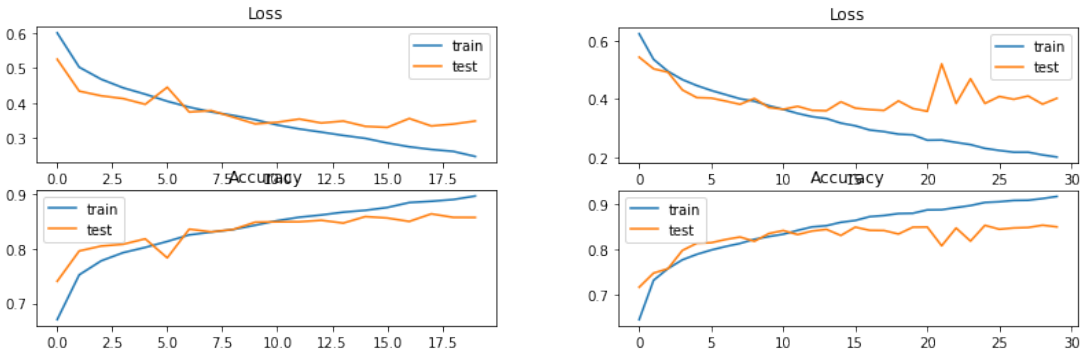

Figure 4.2: Graphics of original dimension CNN results

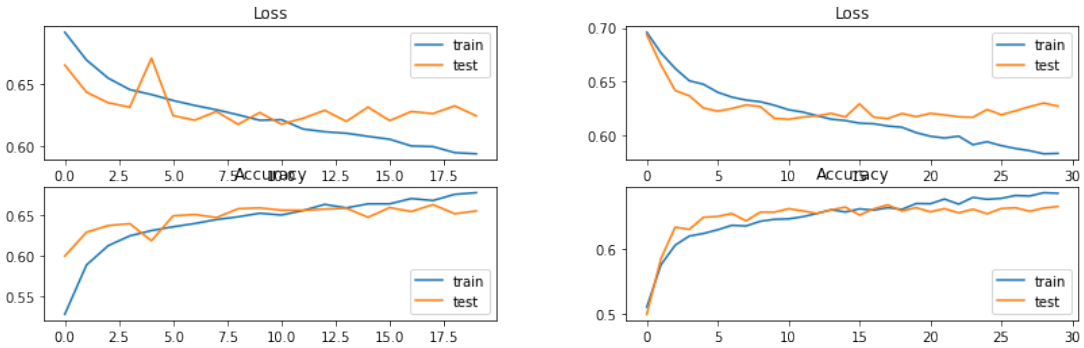

Figure 4.3: Graphics of CNN results of DCT applied images with dimension/8

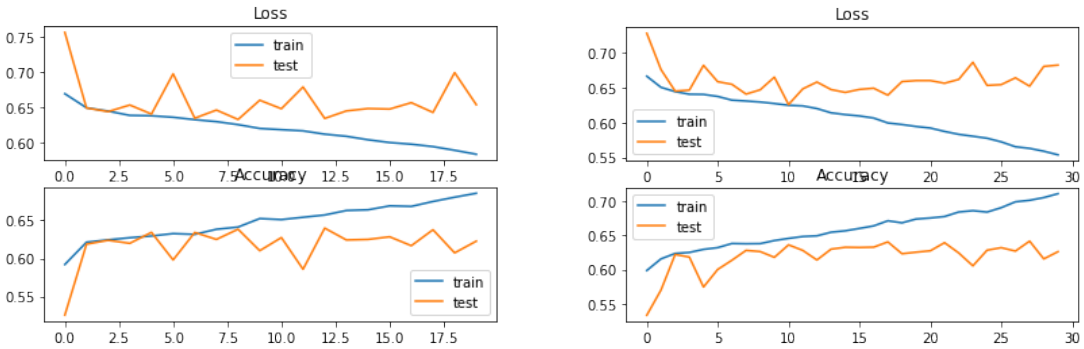

Figure 4.4: Graphics of CNN results of DCT applied images with dimension/4

5. EVALUATION OF SUCCESS CRITERIA

1. The first success criterion : The average accuracy difference between RGB and JPG models should be no more than -5% loss.
   This success criterion has not been achieved. The accuracy results of the compressed images decreased by -15%. This decrease changes according to the epoch values of the training.

2. The second success criterion : The model with JPG is about 1.5× faster than regular RGB images.
   A successful result was obtained in terms of speed. The training of compressed images is at least 1.5 times faster than the original images. This speed was even up to 8 times faster sometimes.

3. The third success criterion : At least 30.000 data will be used.
   I used exactly 30.000 cats and dogs images which I found on the internet.

6. DISCUSSION AND CONCLUSION

In this work, we have created a database of compressed images and studied the efficiency of this method by a CNN, due to compression.

Experiments show that on average, images can be compressed and a deep learning model can learn using these compressed images. This study also showed that compressed image trainings are much faster than original image trainings.

To future studies, in order to increase the accuracy, epoch values can be more manipulated. The number of compressed images can be increased more. Or a different CNN model can be used. [24]

BIBLIOGRAPHY

[1] B. Deguerre, C. Chatelain, and G. Gasso, "Fast object detection in compressed jpeg images," pp. 333–338, 2019.

[2] A. B. Watson *et al.*, "Image compression using the discrete cosine transform," *Mathematica journal*, vol. 4, no. 1, p. 81, 1994.

[3] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," *Pattern Recognition Letters*, vol. 141, pp. 61–67, 2021.

[4] L. Alzubaidi, J. Zhang, A. J. Humaidi, *et al.*, "Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, no. 1, pp. 1–74, 2021.

[5] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital signal processing*, vol. 73, pp. 1–15, 2018.

[6] M. K. Aditi and E. Poovammal, "Image classification using a hybrid lstm-cnn deep neural network," *Int. J. Eng. Adv. Technol.(IJEAT)*, vol. 8, no. 6, pp. 1342–1348, 2019.

[7] Q. Yin, R. Zhang, and X. Shao, "Cnn and rnn mixed model for image classification," vol. 277, p. 02 001, 2019.

[8] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," pp. 2285–2294, 2016.

[9] M. Ehrlich, L. Davis, S.-N. Lim, and A. Shrivastava, "Analyzing and mitigating jpeg compression defects in deep learning," pp. 2357–2367, 2021.

[10] A. Ghosh and R. Chellappa, "Deep feature extraction in the dct domain," pp. 3536–3541, 2016.

[11] L. Gueguen, A. Sergeev, B. Kadlec, R. Liu, and J. Yosinski, "Faster neural networks straight from jpeg," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[12] M. Ehrlich and L. S. Davis, "Deep residual learning in the jpeg transform domain," pp. 3484–3493, 2019.

[13] S.-Y. Lo and H.-M. Hang, "Exploring semantic segmentation on the dct representation," pp. 1–6, 2019.

[14] M. Ehrlich, L. Davis, S.-N. Lim, and A. Shrivastava, "Quantization guided jpeg artifact correction," pp. 293–309, 2020.

[15] J. Choi and B. Han, "Task-aware quantization network for jpeg image compression," pp. 309–324, 2020.

[16] B. Pang, E. Nijkamp, and Y. N. Wu, "Deep learning with tensorflow: A review," *Journal of Educational and Behavioral Statistics*, vol. 45, no. 2, pp. 227–248, 2020.

[17] . [Online]. Available: `https://www.python.org/`.

[18] . [Online]. Available: `https://www.image-net.org/`.

[19] S. Yu and E. E. Swartzlander, "Dct implementation with distributed arithmetic," *IEEE transactions on Computers*, vol. 50, no. 09, pp. 985–991, 2001.

[20] B. Rajesh, M. Javed, S. Srivastava, *et al.*, "Dct-compcnn: A novel image classification network using jpeg compressed dct coefficients," pp. 1–6, 2019.

[21] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," 2016, `http://www.deeplearningbook.org`.

[22] . [Online]. Available: `https://www.geeksforgeeks.org/python-gui-tkinter/?ref=lbp`.

[23] . [Online]. Available: `https://docs.python.org/3/library/tkinter.html`.

[24] M. Dejean-Servières, K. Desnos, K. Abdelouahab, W. Hamidouche, L. Morin, and M. Pelcat, "Study of the impact of standard image compression techniques on performance of image classification with a convolutional neural network," 2017.