Configuration Manual

MSc Research Project
Programme Name

Kumar Chaturvedi
Student ID: x18181970

School of Computing
National College of Ireland

Supervisor: Dr. Catherine Mulwa

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Kumar Chaturvedi		
Student ID:	X18181970		
Programme:	MSc. DATA ANALYTICS	**Year:**	2019-2020
Module:	Research Project		
Lecturer:	17th August 2020		
Submission Due Date:	……………………………………………………………………………………………………….………		
Project Title:	Wolf and Dog Breed Classification using Deep Learning Techniques		
Word Count:	**665**	**Page Count: 10**	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Kumar Chaturvedi…

Date: 17/08/2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Kumar Chaturvedi
Student ID: x18181970

1 Introduction

This document tells the procedure step by step for any researcher to run the program code without any errors. This document also contains the hardware configuration and other minimum configuration required for running the models. After all the steps are followed results can be replicated and future work and research can be done without any problem.

2 System Specifications

2.1 Hardware requirements

The hardware configuration needed is described below:

Processor: intel(R) Core (TM) i3-7020U 2.30GHz
RAM: 8GB
Storage: 1 TB
Operating System: Windows 10

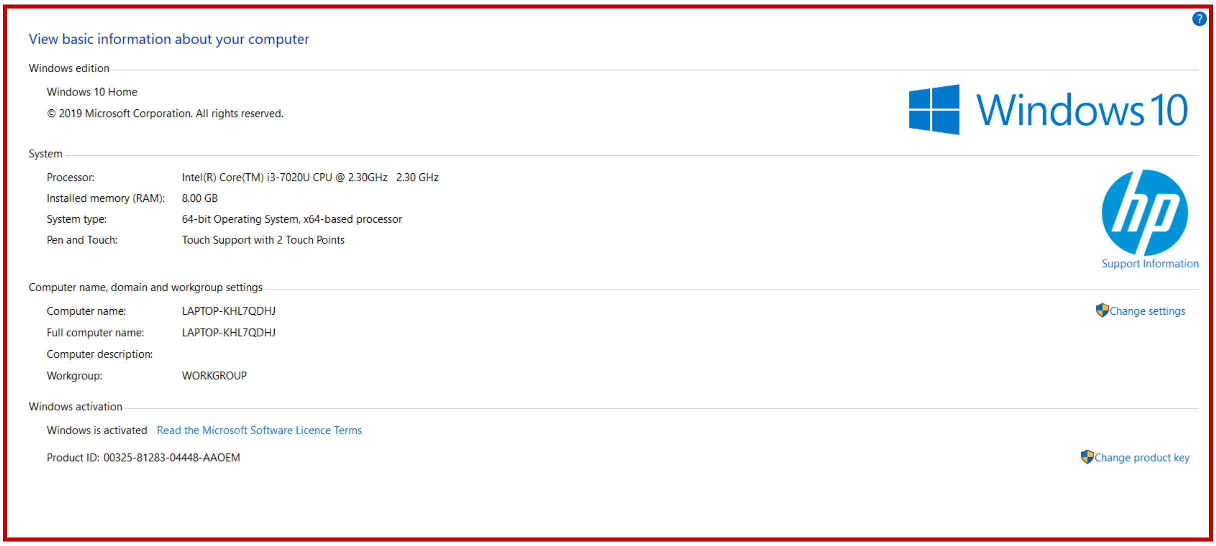


Figure 1: Hardware Requirements

2.2 Software requirements

Programming Tool: Google Colab (cloud-based environment)
Web Browser: Google Chrome
email: need one Gmail account for google drive access.

2.2.1 Environment setup for Google Colab

In this section google colaboratory environmental setup will be discussed for conducting the project easily. For better understanding screenshots are attached in the document.

1. Type Google Colab in google search.

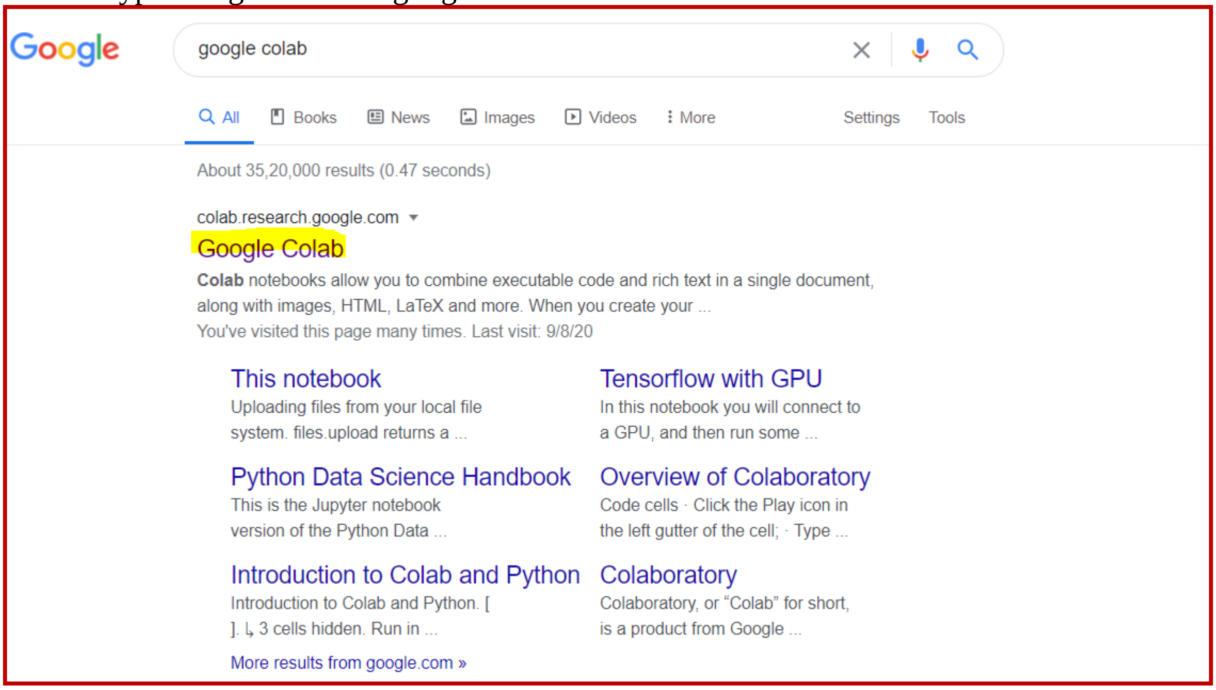


Figure 2: Searching Google Colab in Google Search

2: Click on Google Colab highlighted then a new pop-up will show for creating new Colab file and opening existing files.

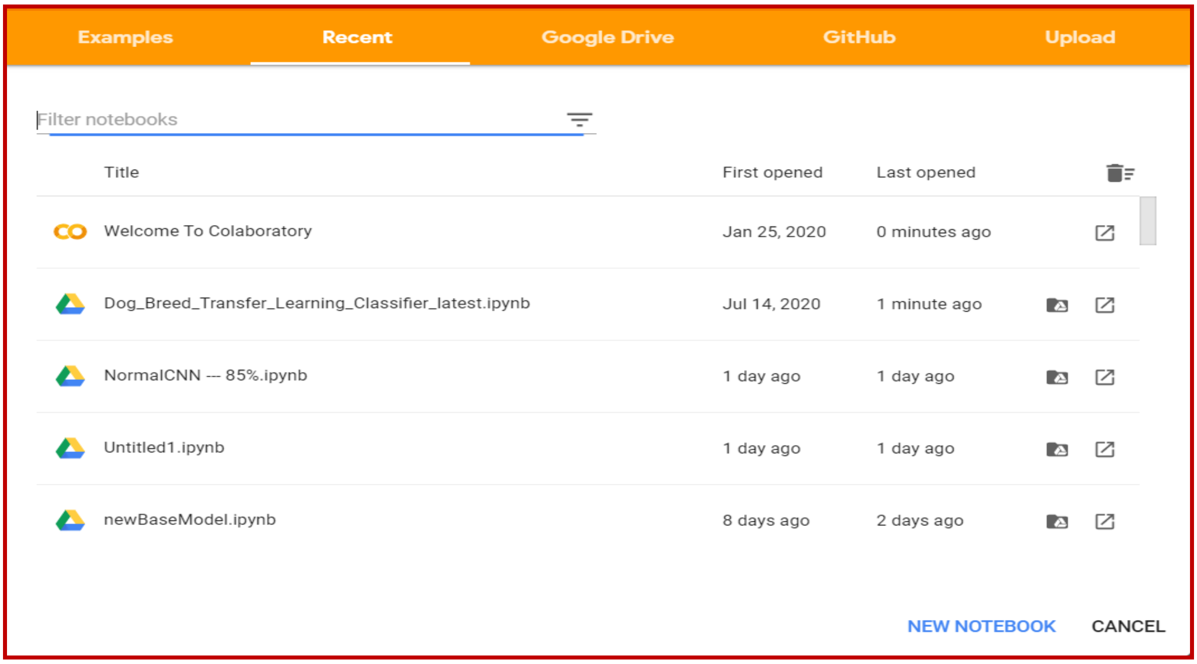


Figure 3: Creating New Notebook

3: Data collected from below two sources are merged and uploaded to google drive.

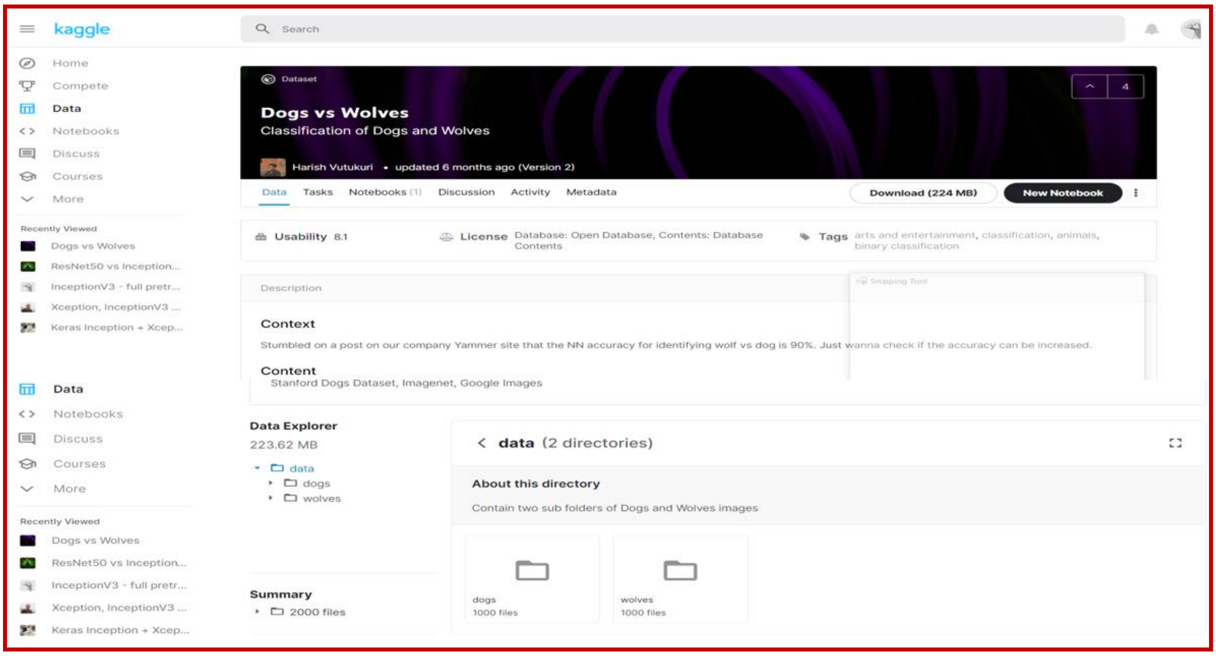


Fig4: Wolf Images

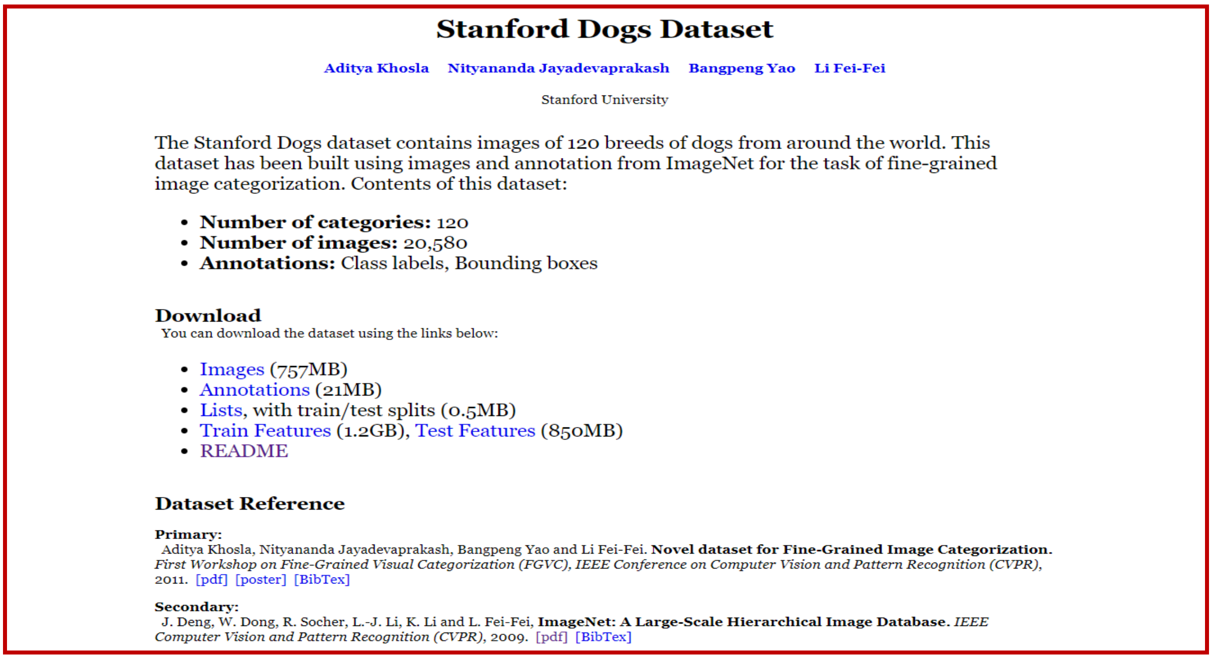


Fig:5 Dog Breed Images

4: Once data is uploaded in google drive and mounted, Colab notebook will look like in below figure

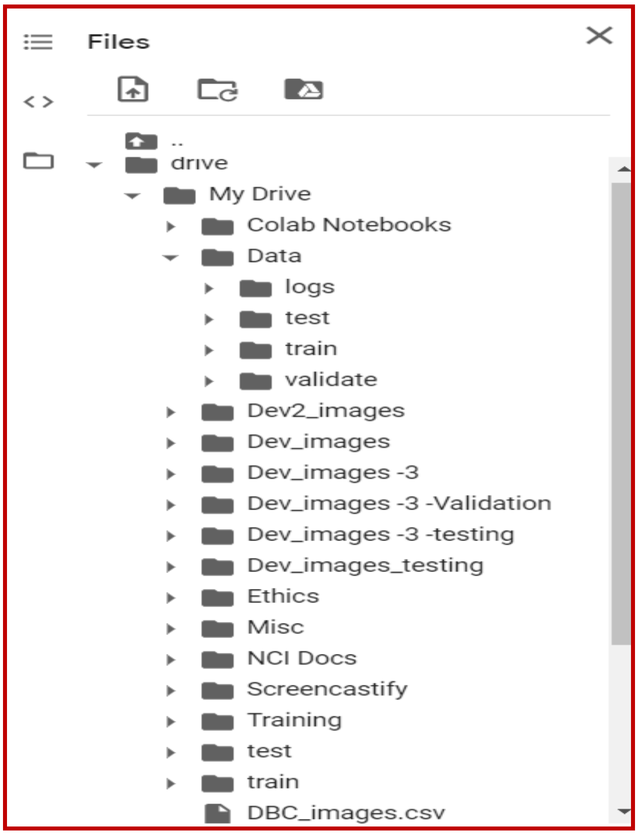
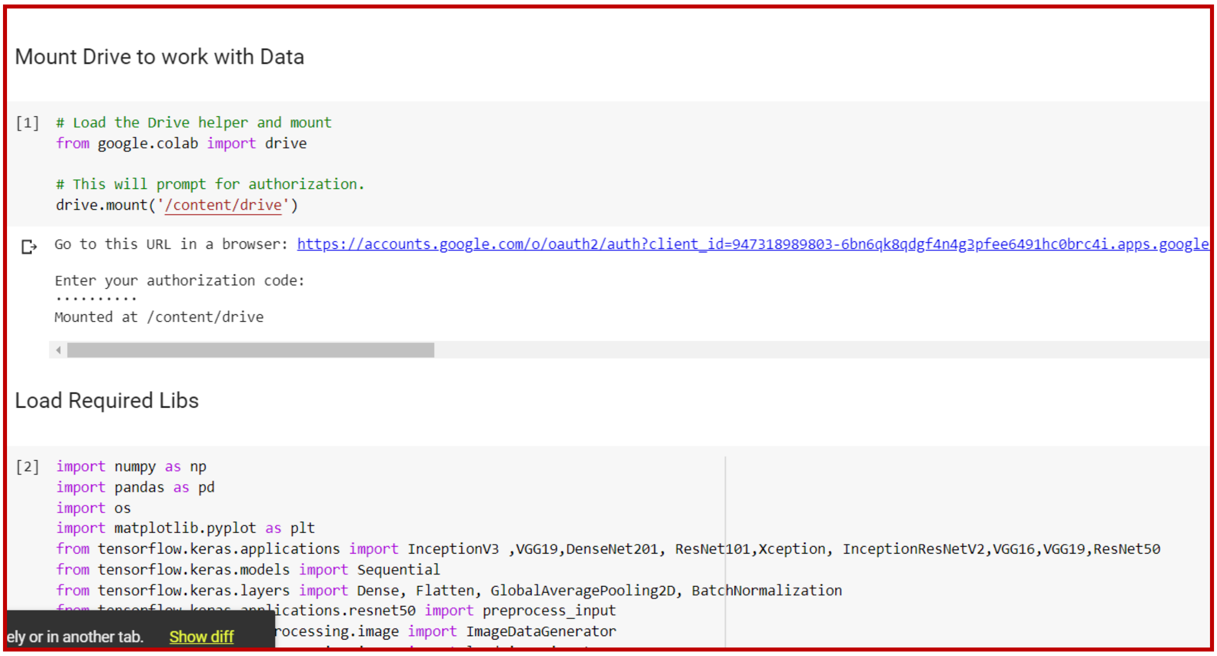


Fig6: Data Repository

3 Deep Learning Model Execution



Mount Drive to work with Data

```
[1] # Load the Drive helper and mount
    from google.colab import drive

    # This will prompt for authorization.
    drive.mount('/content/drive')

    Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.google

    Enter your authorization code:
    ··········
    Mounted at /content/drive
```

Load Required Libs

```
[2] import numpy as np
    import pandas as pd
    import os
    import matplotlib.pyplot as plt
    from tensorflow.keras.applications import InceptionV3 ,VGG19,DenseNet201, ResNet101,Xception, InceptionResNetV2,VGG16,VGG19,ResNet50
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.layers import Dense, Flatten, GlobalAveragePooling2D, BatchNormalization
    ...ications.resnet50 import preprocess_input
    ...rocessing.image import ImageDataGenerator
```


Figure 7. Loading Libraries

```
    from keras.layers import Dropout
```

Data Prepration

Splitting data into Training,Validation and Testing Subsets

```
[3] import os
    import pandas as pd
    data = []
    for filename in os.listdir('/content/drive/My Drive/Dev_images/'):
      for f in os.listdir('/content/drive/My Drive/Dev_images/' + filename):
        data.append(['/content/drive/My Drive/Dev_images/' + filename+'/'+f, filename])

    df = pd.DataFrame(data, columns = ['image_path', 'type'])
    df.to_csv("/content/drive/My Drive/DBC4_images.csv")
    dir = os.listdir('/content/drive/My Drive/Dev_images/')
    result = []

[ ] for i in range(len(dir)):
      df_mod = df[(df['type']==dir[i])]
      train, validate, test = np.split(df_mod.sample(frac=1), [int(.6*len(df_mod)), int(.8*len(df_mod))])
      result.append([train, validate, test])
```

Figure 8: Splitting Data

Modeling - ***Preparing Data classifier***

```
[7] base_model= ResNet101(include_top=False, pooling='avg', weights='imagenet')
    model = Sequential()
    model.add(base_model)
    model.add(Flatten())
    model.add(BatchNormalization())
    model.add(Dense(2048, activation='relu'))
    model.add(BatchNormalization())
    model.add(Dense(1024, activation='relu'))
    model.add(BatchNormalization())
    model.add(Dense(64, input_dim=64, kernel_regularizer=regularizers.l2(0.0001)))
    model.add(Dropout(0.5))
    model.add(Dense(num_classes, activation='softmax'))

    model.layers[0].trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet101_weights_tf_dim_ordering_tf_kernels_not
171450368/171446536 [==============================] - 1s 0us/step
```

```
[8] model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Model Training

```
[ ] initial_epochs =50
```

Figure9: Modelling the Classifier model

Plotting the results

```
[9] ###PLOTING
    # Training Accuracy Vs Validation Accuracy
    # Training Loss Vs Validation Loss
    # Precision, Recall, F1 Score, AUC, ROC Curves

    def plot_confusion_matrix(cm, classes,
                              normalize=False,
                             title = 'Confusion Matrix',
                             cmap=plt.cm.Blues):

      plt.imshow(cm, interpolation = 'nearest', cmap=cmap)
      plt.title(title)
      plt.colorbar()
      tick_marks = np.arange(len(classes))
      plt.xticks(tick_marks, classes, rotation = 45)
      plt.yticks(tick_marks, classes)

      if normalize:
        cm = cm.astype('float') / cm.sum(axis=1) [:, np.newaxis]
        print("Normalized Confusion Matrix")
      else:
        print("Confusion Matrix without normalization")

      print(cm)

      thresh = cm.max() / 2.

ly or in another tab.  Show diff   oduct(range(cm.shape[0]), range(cm.shape[1])):
```

Figure 10: Plotting the results

Exploratory Data Analysis

```
[12] data_labels = pd.read_csv('/content/drive/My Drive/DBC_images.csv')
     data_labels.head()
```

	Unnamed: 0	image_path	type
0	0	/content/drive/My Drive/Dev_images/American_St...	American_Staffordshire_terrier
1	1	/content/drive/My Drive/Dev_images/American_St...	American_Staffordshire_terrier
2	2	/content/drive/My Drive/Dev_images/American_St...	American_Staffordshire_terrier
3	3	/content/drive/My Drive/Dev_images/American_St...	American_Staffordshire_terrier
4	4	/content/drive/My Drive/Dev_images/American_St...	American_Staffordshire_terrier

```
[13] data_labels.describe()
```

	Unnamed: 0
count	1924.000000
mean	961.500000
std	555.555278
min	0.000000

ely or in another tab. Show diff

Figure 11: Exploratory Data Analysis

References

A. Khosla, N. Jayadevaprakash, B. Yao, F.-F. Li (2011). Novel dataset for fine-grained image categorization: Stanford dogs, *IEEE Conference on Computer Vision and Pattern Recognition*

A. Krizhevsky, I. Sutskever and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks, *Advances in neural information processing systems*

A. A. Hammam, M. M. Soliman, A. E. Hassanein (2018). DeepPet: A Pet Animal Tracking System in Internet of Things using Deep Neural Networks, *2018 13th International Conference on Computer Engineering and Systems (ICCES)*

B. Yao, G. Bradski, and L. Fei-Fei (2012). A codebook-free and annotationfree approach for fine-grained image categorization, *IEEE Conference on Computer Vision and Pattern Recognition*

C. Li, Q. Song, Y. Wang, H. Song2, Qi Kang, Jian Cheng, H. Lu (2016). LEARNING TO RECOGNITION FROM BINGCLICKTURE DATA, *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*