Faculty of Computers and Artificial Intelligence

Department: Computer Science

2022/2023

CS 396 Selected Topics in CS-2

Research Project

Team ID No. 52

	ID	Name	Grade
1.	201900597	مارك فايز وديع قسطنطين	
2.	201900938	هدير محمد حنفي محمود	
3.	201900915	نورهان محمد المهدي مختار مرجان	
4.	201900960	ياسمين عزت إبراهيم غريب	
5.	201900973	يوسف إبراهيم أحمد الجريدلي	
6.	201900980	يوسف رمضان عبد العزيز عبد الحافظ	

Paper Details:

- **Paper Name:**

  "Deep Network Ensemble Learning applied to Image Classification using CNN Trees"

- **Authors Name:**
  - Abdul Mueed Hafiz
  - Ghulam Mohiuddin Bhat

- **Dataset USED:**
  - ImageNet
  - Natural Images

- **Implemented Algorithms:**
  - CNN
  - Ensemble learning Algorithms

- **Results:**

  The classification accuracies of the proposed approach are higher than that of the baseline in all experiments using ImageNet subsets. The accuracy is also almost equal for the Natural Image subset.

- *[Submitted on 23 Jul 2020], Cornell University*

Project Description:

Image classification is a complex procedure which may be affected by many factors

Dataset used:

1-Natural Images

https://www.kaggle.com/datasets/prasunroy/natural-images

- **Samples:**
  It includes 4338 files for (Training, Validation & Testing)
- **Number of Classes and their labels:**
  - 6 classes
  - Labels:
    (Airplane, Car, Dog, Flower, Motorbike, Person)

- **Implementation Details:**

- Resizing to 224 images
- In All the CNNs we use Adam Optimizer for training, with mini-batch size of 32 for larger image categories and of 16 for smaller ones,

- The CNNs are trained using early stopping. We use loss function in binary cross-entropy for binary-class CNNs and categorical cross-entropy for multiclass CNNs.
- We divide the experiments into two sets based on the number of classes used for classification : (four and six, respectively)

1. **Four-class** :

uses a single binary-class CNN (2 categories per class) i.e. [(c1, c2) v/s (c3, c4)] in first stage, and two binary-class CNNs in second stage i.e. [c1 v/s c2] and [c3 v/s c4]

2. **Six-class :**

using two different ensemble structures. When final layer neuron count is 2, we use sigmoid activation. For more that 2 final layer neuron count, we use softmax activation

Classification using Ensemble Structure #1

Here , we use a three-class CNN (3 categories per class) i.e. [(c1, c2) v/s (c3, c4) v/s (c5, c6)], in first stage, and three binary-class CNNs in second stage i.e. [c1 v/s c2], [c3 v/s c4], and [c5 v/s c6]

Classification using Ensemble Structure #2

Here we use a single binary-class CNN (2 categories per class) i.e. [(c1, c2, c3) v/s (c5, c6, c4)], in first stage, and two three[1]class CNNs in second stage i.e. [c1 v/s c2 v/s c3] and [c5 v/s c6 v/s c4]. Figure 5 shows the architecture of the six-class classification ensemble#2 for the proposed technique.

- **(for the first dataset):**

  Ratio& number of images:

  -Training: 2298 (53%)

  -Validation: 1063 (24%)

  -Test: 977 (23%)

2- 10 Monkey Species

https://www.kaggle.com/datasets/slothkong/10-monkey-species?resource=download

- **Samples:**

  It includes 2488 files for (Training, Validation & Testing)

- **Number of Classes and their labels:**
  - 6 classes
  - Labels : (“patas_monkey”, “bald_uakari”, “japanese_macaque”, “white_headed_capuchin”,

“silvery_marmoset”,
“black_headed_night_monkey”)

- **(for the second dataset):**

Ratio& number of images:

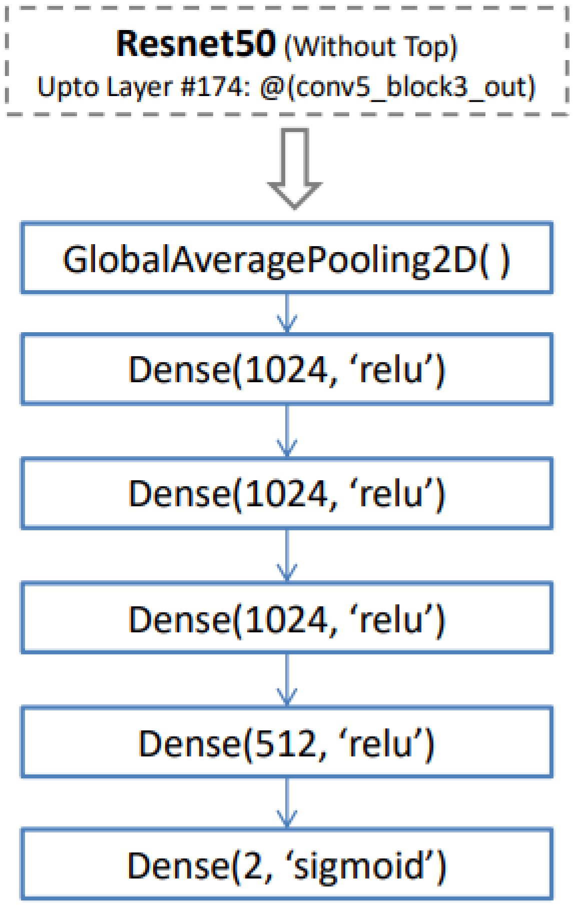
-Training: 1847 (75%)

- validation: 498 (20%)

- Test: 143(5%)

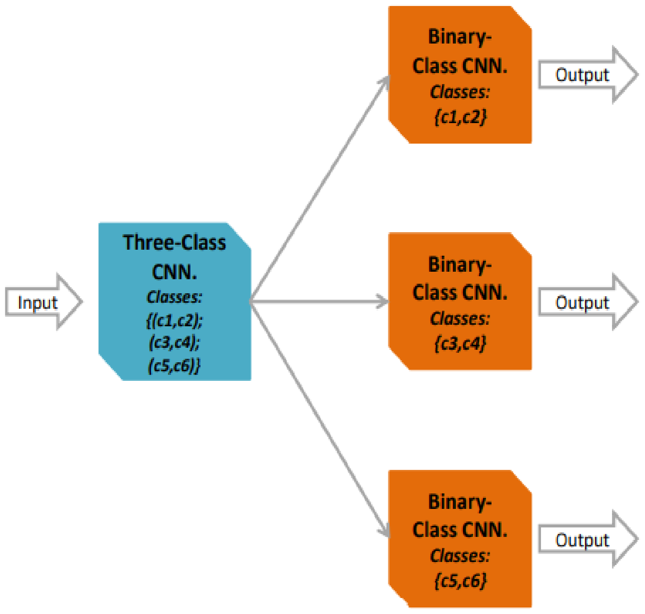
Block Diagram:

- Four-Class Classification:


Resnet50 (Without Top)
Upto Layer #174: @(conv5_block3_out)
GlobalAveragePooling2D( )
Dense(1024, ‘relu’)
Dense(1024, ‘relu’)
Dense(1024, ‘relu’)
Dense(512, ‘relu’)
Dense(2, ‘sigmoid’)

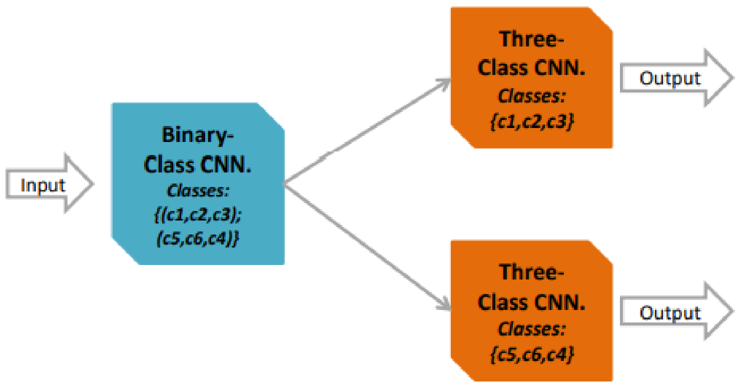
- **Six-class classification**

1- three-class CNN:



Input
Three-Class CNN. Classes: {(c1,c2); (c3,c4); (c5,c6)}
Binary-Class CNN. Classes: {c1,c2}
Output
Binary-Class CNN. Classes: {c3,c4}
Output
Binary-Class CNN. Classes: {c5,c6}
Output

2- Single binary-class CNN (2 categories per class):



Input
Binary-Class CNN. Classes: {(c1,c2,c3); (c5,c6,c4)}
Three-Class CNN. Classes: {c1,c2,c3}
Output
Three-Class CNN. Classes: {c5,c6,c4}
Output

Hyperparameters:

Batch size.

```
target_size=(img_rows, img_cols),batch_size=32,class_mode='binary')
```

Number of neurons, Activation Function.

```
x = tf.keras.layers.Dense(1024, activation='relu')(x)
```

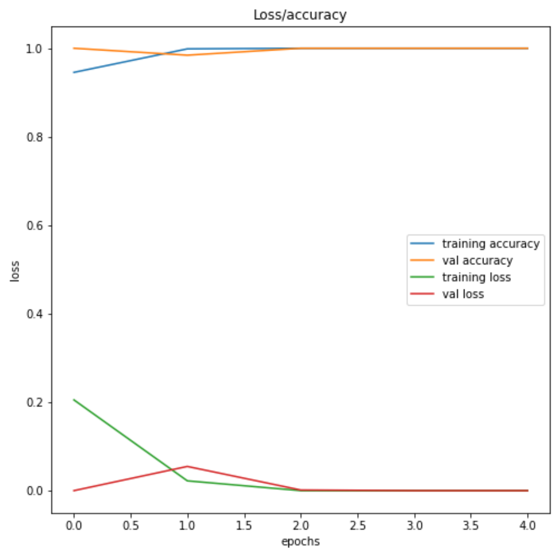
Optimizer, Loss

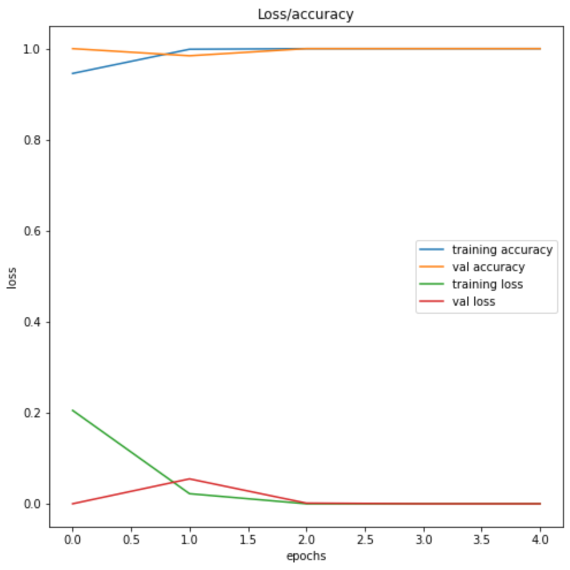
```
model21.compile(optimizer='Adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Number of epochs, validation steps.

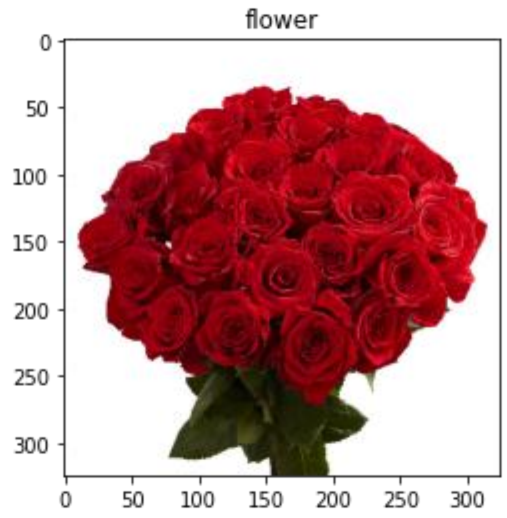
```
history = model1.fit(
      train_generator1,
      steps_per_epoch=train_generator1.n//train_generator1.batch_size,
      epochs=25,
      validation_data=valid_generator1,callbacks=[cb1],
      validation_steps=1)
```

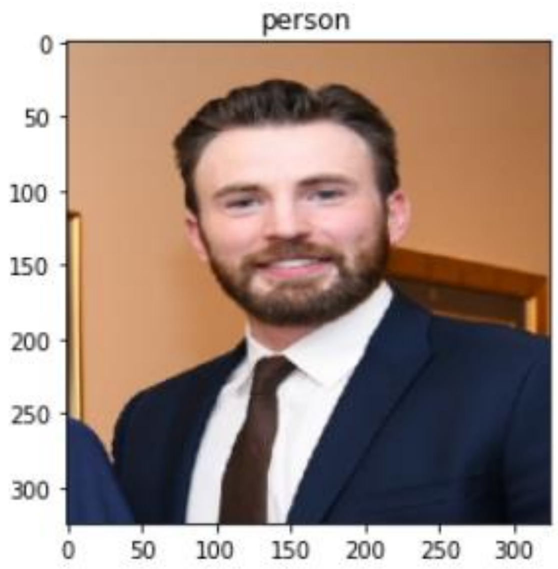
Testing results

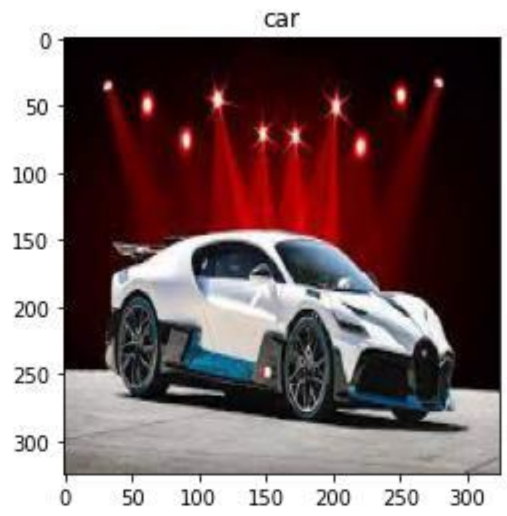


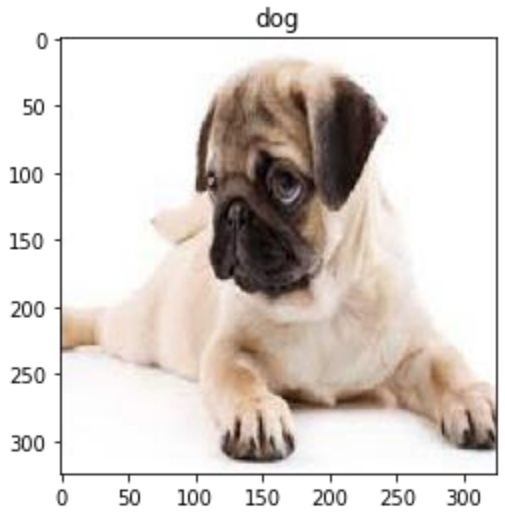


<u>Outputs :</u>

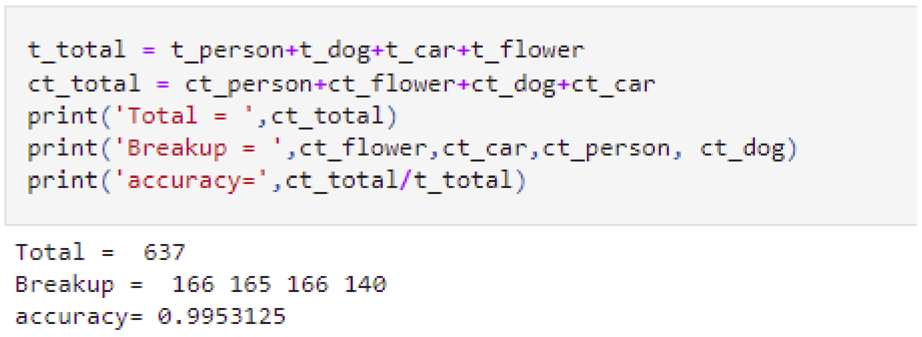








Accuracy: 99%



```
t_total = t_person+t_dog+t_car+t_flower
ct_total = ct_person+ct_flower+ct_dog+ct_car
print('Total = ',ct_total)
print('Breakup = ',ct_flower,ct_car,ct_person, ct_dog)
print('accuracy=',ct_total/t_total)
```

```
Total =  637
Breakup =  166 165 166 140
accuracy= 0.9953125
```

Results noted:

The classification accuracies of the proposed approach are higher than that of the baseline in all experiments using ImageNet subsets. The accuracy is also almost equal for the Natural Image subset.