VIDEO PROCESSING 2: FINDING OBJECTS AND SCENES

ACTIVITY 06

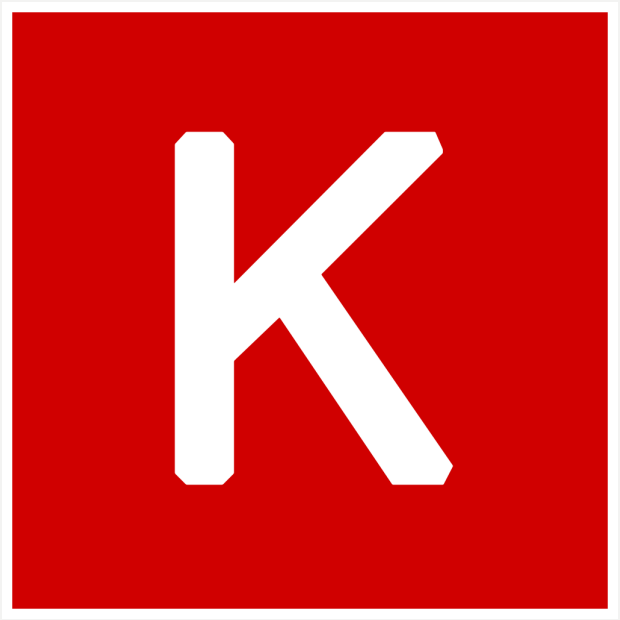
PHYSICS 301

KENNETH M. LEO

OBJECT DETECTION USING PRE-TRAINED NETWORKS

In Python, Pytorch and Keras has pre-trained neural network models which we will use in this activity. To be more specific, we will be using these three neural network models:

1. AlexNet
2. GoogleNet
3. MobileNetV2



ALEXNET

AlexNet was one of the convolutional neural networks (CNN) that used ImageNet to classify objects. What AlexNet does it it classifies an object into one of its 1000 classes that it created.

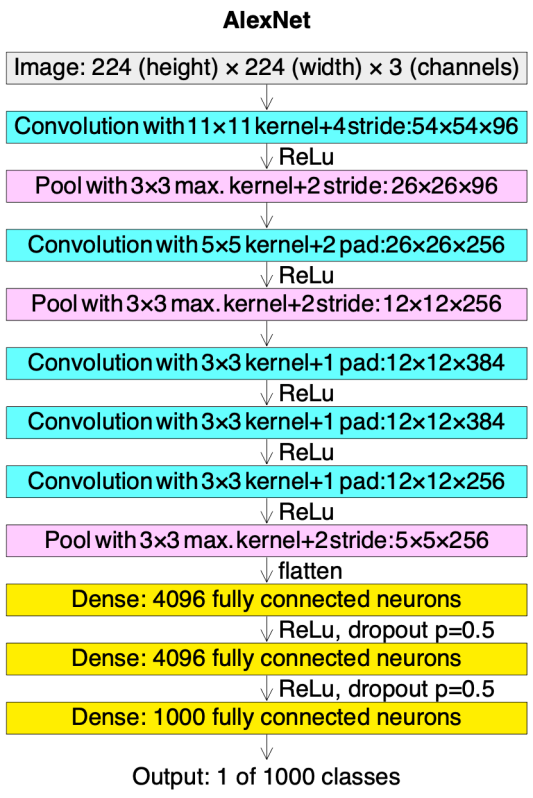


Image from: https://en.wikipedia.org/wiki/AlexNet

To use AlexNet in Python:

- load model from PyTorch using the command:

  alexnet = torchvision.models.alexnet(pretrained = True)

- input image must follow this size: 224 x 224

Drawback of AlexNet:

- Object classification is limited to 1000 classes* only

Here are 10 randomly selected classes from the 1000 possible classifications:

1. bagel, beigel
2. theater curtain, theatre curtain
3. bulbul
4. badger
5. cabbage butterfly
6. park bench
7. Doberman, Doberman pinscher
8. cheeseburger
9. affenpinscher, monkey pinscher, monkey dog
10. Bouvier des Flandres, Bouviers des Flandres

*some objects are grouped into one class

GOOGLENET

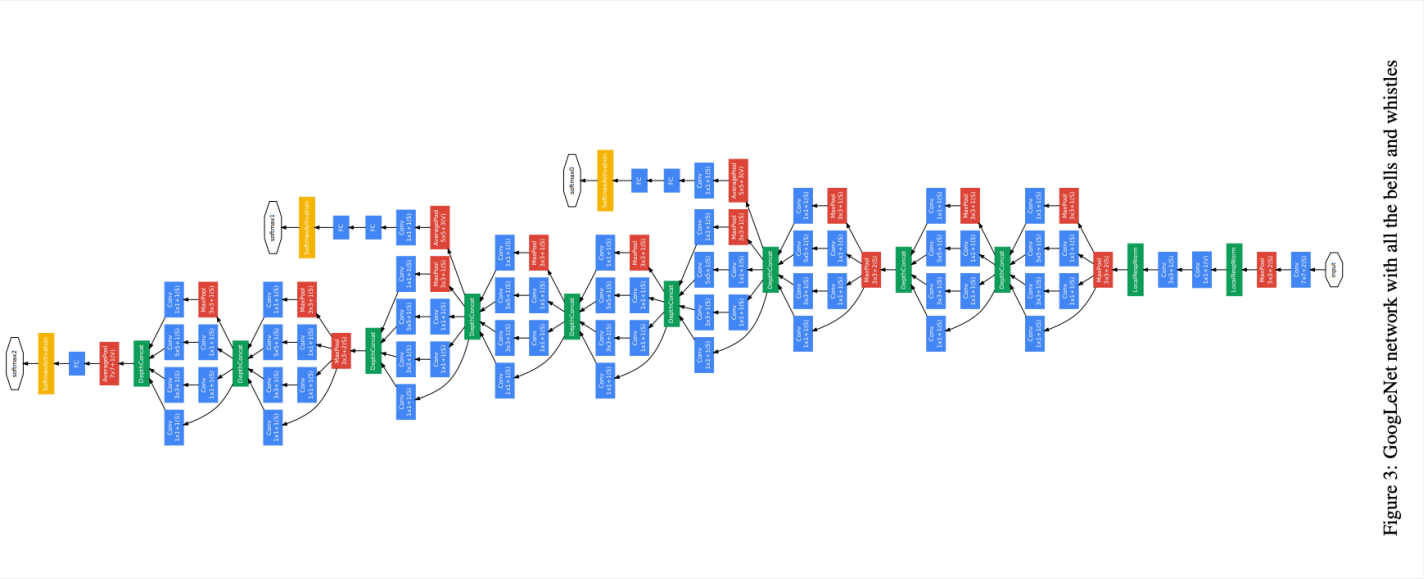


Figure 3: GoogLeNet network with all the bells and whistles

Image from: https://paperswithcode.com/method/googlenet

GoogleNet is a CNN based on a different concept (called inception modules) compared to AlexNet. One advantage of using GoogleNet is that it uses less parameters that AlexNet which means it is able to perform faster, which is actually in line with their goal to have a model that can be used on smart-phones.

To use GoogleNet, we use Keras and use the following command to load the pre-trained model:

```
googlenet = tensorflow.keras.applications.inception_v3.InceptionV3(weights='imagenet')
```

Notice that we are loading a model names inception_v3 since googlenet was based on the inception modules. Note also that for this model, we require our image to be 299 x 299 pixels. Also, similar to Alexnet, GoogleNet has a thousand classifications.

MOBILENET_V2

Mobilenet_v2 has a similar goal as googlenet which was to make it useful on mobile phones. Unlike Googlenet, mobilenet_v2 uses what you call *Depthwise separable convolution* instead of your standard convolution. This results in using lesser number of parameters, but with a slighter decrease in performance.

To use Mobilenet_v2, we use Keras and use the following command to load the pre-trained model:

```
googlenet = tensorflow.keras.applications.mobilenet_v2.MobileNetV2(weights='imagenet')
```

Size of input image: 224 x 224 pixels

Methods:

In this activity, we compare the three discussed models using different test images and then use webcam live-video (from Activity 05) to check classifications of other objects in real time.

SAMPLE IMAGES USED







PRE-TRAINED MODEL RESULTS (1 OF 4)

Alexnet

Rank	Classification	Percentage
1	scabbard	21.46 %
2	nail	10.35 %
3	rifle	9.38 %
4	can opener, tin opener	6.48 %
5	assault rifle, assault gun	6.27 %

Rank	Classification	Percentage
1	sock	30.51 %
2	hand blower, blow dryer, blow drier, hair dryer, hair drier	10.79 %
3	sandal	8.42 %
4	running shoe	5.33 %
5	diaper, nappy, napkin	5.27 %

Googlenet

Rank	Classification	Percentage
1	scabbard	67.62 %
2	holster	6.18 %
3	rifle	2.18 %
4	revolver	1.7 %
5	can_opener	1.15 %

Rank	Classification	Percentage
1	running_shoe	98.25 %
2	sock	1.11 %
3	sandal	0.1 %
4	shoe_shop	0.05 %
5	doormat	0.04 %

Mobilenet V2

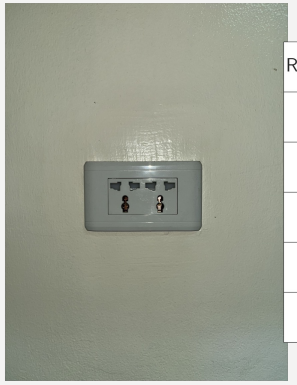
Rank	Classification	Percentage
1	platypus	38.09 %
2	American_alligator	7.24 %
3	scorpion	5.66 %
4	mitten	2.02 %
5	letter_opener	1.65 %

Rank	Classification	Percentage
1	water_bottle	18.85 %
2	running_shoe	16.53 %
3	sock	5.98 %
4	crutch	4.61 %
5	syringe	3.9 %

PRE-TRAINED MODEL RESULTS (2 OF 4)

Alexnet

Rank	Classification	Percentage
1	remote control, remote	60.81 %
2	computer keyboard, keypad	13.41 %
3	typewriter keyboard	8.0 %
4	space bar	4.9 %
5	hand-held computer, hand-held microcomputer	3.01 %



Rank	Classification	Percentage
1	switch, electric switch, electrical switch	69.78 %
2	wall clock	8.61 %
3	safe	3.77 %
4	pay-phone, pay-station	1.57 %
5	washer, automatic washer, washing machine	1.16 %

Googlenet

Rank	Classification	Percentage
1	remote_control	70.33 %
2	hand-held_computer	3.9 %
3	computer_keyboard	2.83 %
4	space_bar	1.38 %
5	cellular_telephone	1.13 %

Rank	Classification	Percentage
1	switch	88.19 %
2	safe	2.02 %
3	modem	1.15 %
4	mailbox	0.26 %
5	soap_dispenser	0.25 %

Mobilenet V2

Rank	Classification	Percentage
1	computer_keyboard	26.08 %
2	space_bar	4.8 %
3	modem	4.15 %
4	doormat	3.87 %
5	remote_control	3.81 %

Rank	Classification	Percentage
1	switch	75.38 %
2	digital_clock	2.29 %
3	mailbox	1.57 %
4	wall_clock	1.55 %
5	safe	1.38 %

PRE-TRAINED MODEL RESULTS (3 OF 4)

Alexnet

Rank	Classification	Percentage
1	water bottle	66.44 %
2	cocktail shaker	19.28 %
3	milk can	11.0 %
4	water jug	1.3 %
5	soap dispenser	0.69 %

Rank	Classification	Percentage
1	ashcan, trash can, garbage can, wastebin, ash bin, ash-bin, ashbin, dustbin, trash barrel, trash bin	40.47 %
2	space heater	19.06 %
3	loudspeaker, speaker, speaker unit, loudspeaker system, speaker system	6.32 %
4	soap dispenser	5.38 %
5	washer, automatic washer, washing machine	3.97 %

Googlenet

Rank	Classification	Percentage
1	water_bottle	64.82 %
2	water_jug	17.91 %
3	saltshaker	1.15 %
4	pill_bottle	1.01 %
5	nipple	0.67 %

Rank	Classification	Percentage
1	hamper	78.49 %
2	space_heater	12.98 %
3	ashcan	2.01 %
4	radio	0.88 %
5	loudspeaker	0.6 %

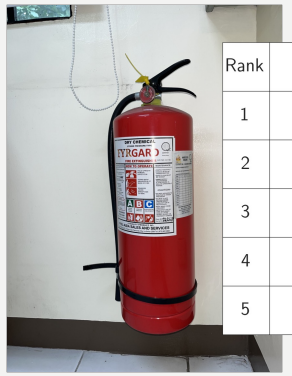
Mobilenet V2

Rank	Classification	Percentage
1	water_bottle	40.0 %
2	water_jug	17.56 %
3	milk_can	10.81 %
4	ashcan	4.35 %
5	rain_barrel	3.32 %

Rank	Classification	Percentage
1	space_heater	19.69 %
2	ashcan	16.79 %
3	soap_dispenser	7.66 %
4	hamper	4.53 %
5	iPod	3.95 %

PRE-TRAINED MODEL RESULTS (4 OF 4)

Alexnet



Rank	Classification	Percentage
1	gas pump, gasoline pump, petrol pump, island dispenser	43.62 %
2	wine bottle	21.2 %
3	red wine	16.57 %
4	punching bag, punch bag, punching ball, punchball	5.51 %
5	water bottle	5.14 %

Rank	Classification	Percentage
1	bucket, pail	39.02 %
2	milk can	24.07 %
3	cup	15.16 %
4	Crock Pot	6.05 %
5	candle, taper, wax light	2.44 %

Googlenet

Rank	Classification	Percentage
1	punching_bag	19.47 %
2	milk_can	1.61 %
3	oil_filter	1.57 %
4	whiskey_jug	1.45 %
5	vacuum	1.42 %

Rank	Classification	Percentage
1	coffee_mug	55.97 %
2	cup	23.6 %
3	coffeepot	2.09 %
4	pitcher	0.84 %
5	mixing_bowl	0.74 %

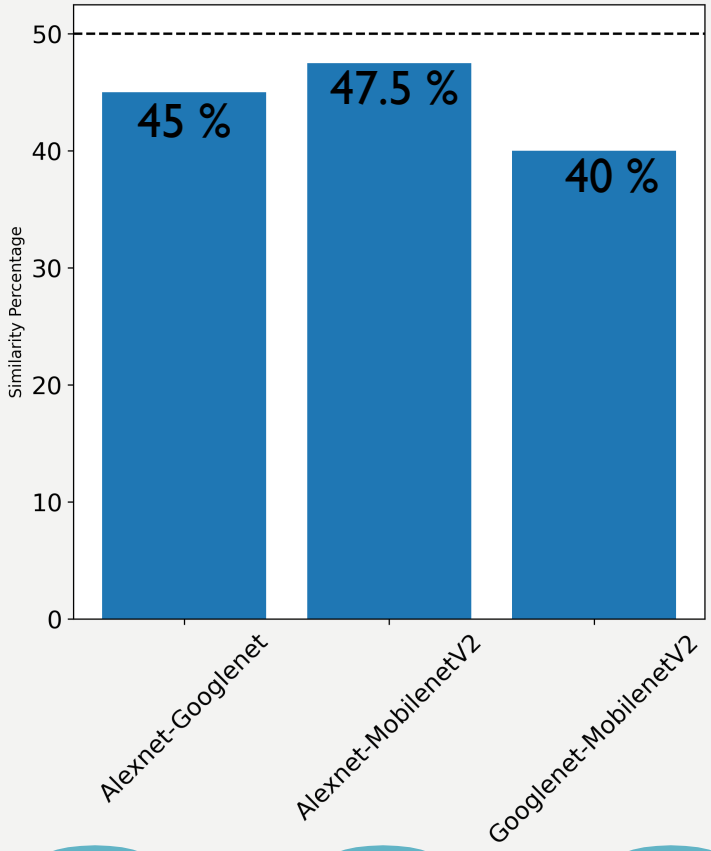
Mobilenet V2

Rank	Classification	Percentage
1	punching_bag	78.22 %
2	beer_bottle	2.22 %
3	gas_pump	1.37 %
4	water_bottle	1.07 %
5	shield	0.8 %

Rank	Classification	Percentage
1	coffee_mug	10.55 %
2	bucket	9.34 %
3	caldron	4.81 %
4	cup	4.3 %
5	beaker	3.37 %

COMPARING RESULTS FROM TEST IMAGES

DISCLAIMER: For this part, note that I only have 8 test images so the statistics here may not represent the actual statistics so in the future, more test images are needed to confirm this.



Comparing the similarity percentage of each pairwise models, we see that almost all of them have the same similarity percentage, with alexnet and mobilenetv2 having the highest similarity percentage

	Top-1 Accuracy	Top-5 Accuracy
Alexnet	35 %	75 %
Googlenet	50 %	75 %
MobilenetV2	62.5 %	75 %

Now, we see that when we use the top-5 predictions of the 3 models, we get 75% accuracy but notice that only the mobilenetv2 has a high accuracy if we only consider the top predictions

TESTING MODELS USING WEBCAM-VIDEO

Now, in this part, we use videos taken by our webcam (so its live object detection!). The top 5 predictions will also be printed live at the bottom right of the webcam video

Sample-gif of how it works

TESTING MODELS USING WEBCAM-VIDEO

Here are (more) random objects that I classified while walking around the lab :D

Alexnet

TESTING MODELS USING WEBCAM-VIDEO

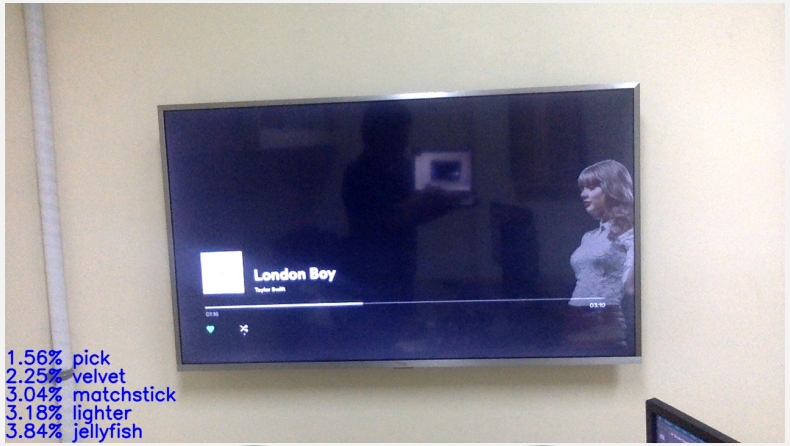
Here are (more) random objects that I classified while walking around the lab :D

GoogleNet

TESTING MODELS USING WEBCAM-VIDEO

Here are (more) random objects that I classified while walking around the lab :D

MobileNetV2



* Across all models, I noticed that the switch is almost always accurate. Maybe the reason for this is that the shape of switches are universal unlike other objects

SUMMARY PLUS REFERENCES

Summary

Similar to Activity 05, I enjoyed this a lot because I got to work with different CNN models and apply that to live video taken by webcam. I also was able to successfully implement the pre-trained models in Python using PyTorch and Tensorflow Keras. If I had more time, I guess I could have made my own model and compare its accuracy to the premade ones.

Score:

Technical Correctness – 30
Quality of Presentation – 30
Reflection – 30
Ownership – 10

Total – 100/100

References

[1] https://www.quora.com/What-are-the-differences-among-AlexNet-GoogleNet-and-VGG-in-the-context-of-convolutional-neural-networks
[2] https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/deep_learning/googlenet
[3] https://stackoverflow.com/questions/50624496/what-is-the-difference-between-tensorflow-inception-and-mobilenet
[4] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.C. "MobileNetV2: Inverted Residuals and Linear Bottlenecks." In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4510-4520). IEEE.