CNNs in Practical Robotic Applications

Advanced topic I

Dimitris Chrysostomou

Associate Professor

Faculty of Engineering, Department of Materials and Production

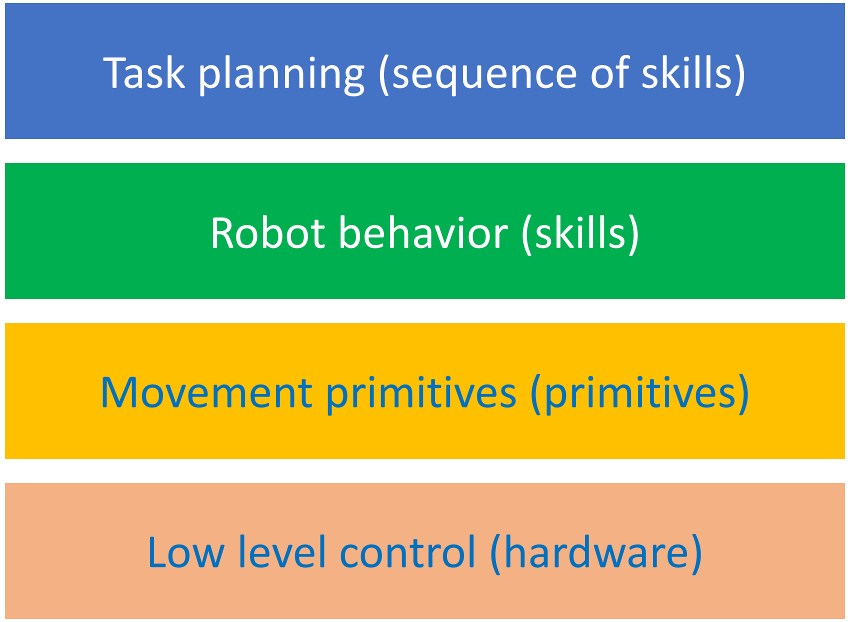
Spring 2020

AALBORG UNIVERSITY
DENMARK

Outline

- Previously in OMTP…
- A brief recap of CNNs
- Deep learning and robotic applications
- A glimpse into the deep learning platforms and frameworks
- A couple of practical examples of CNNs
- Assignments
- Recommendations for future endeavors

Previously in OMTP...



Task planning (sequence of skills)

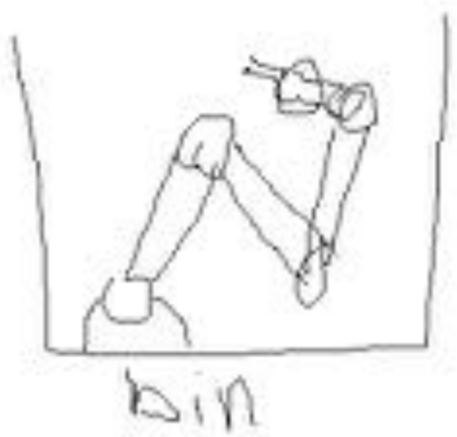
Robot behavior (skills)

Movement primitives (primitives)

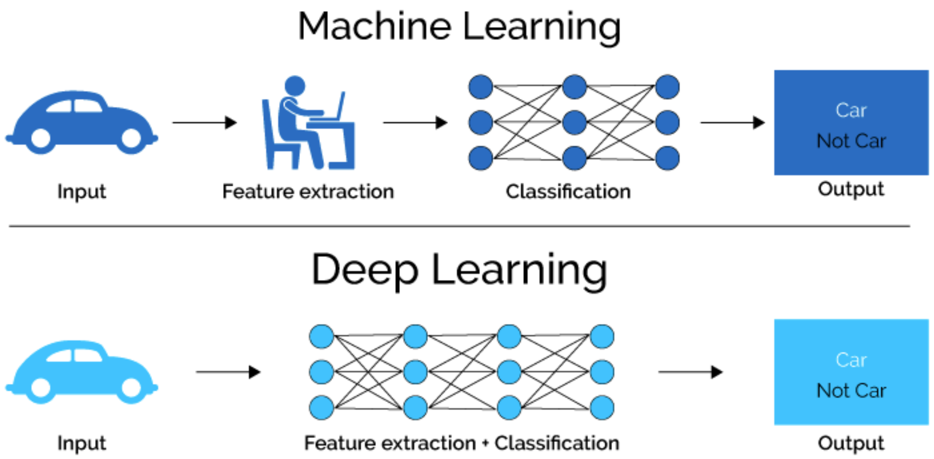
Low level control (hardware)

Quick recap of the course

- URDF & XACRO
- GIT and how to maintain your repository
- HRI and DMPs
- Robot control in contact with the environment
- Manipulate things with MoveIT
- Simple object detection and grasping
- Design of behaviors with state machines

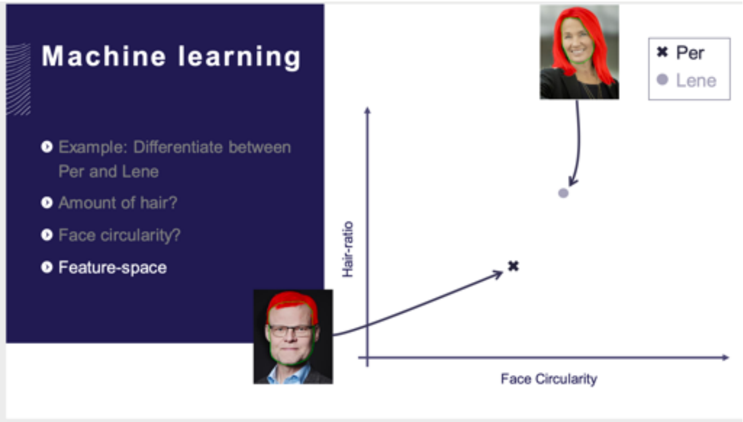


About CNNs and Deep Learning



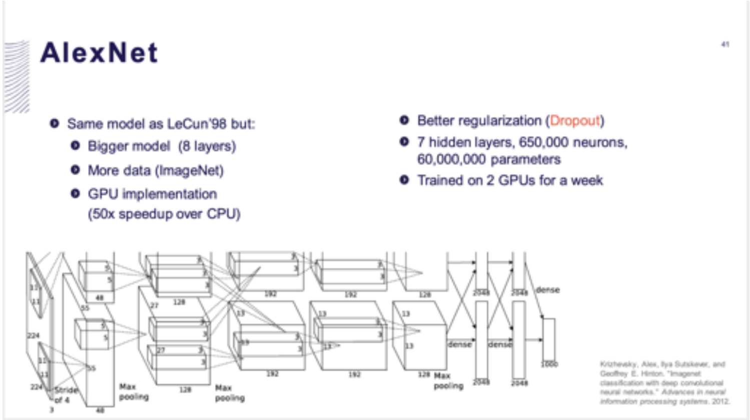
What about Deep Learning?

- You already have some expertise…in different architectures



Machine learning

- Example: Differentiate between Per and Lene
- Amount of hair?
- Face circularity?
- Feature-space

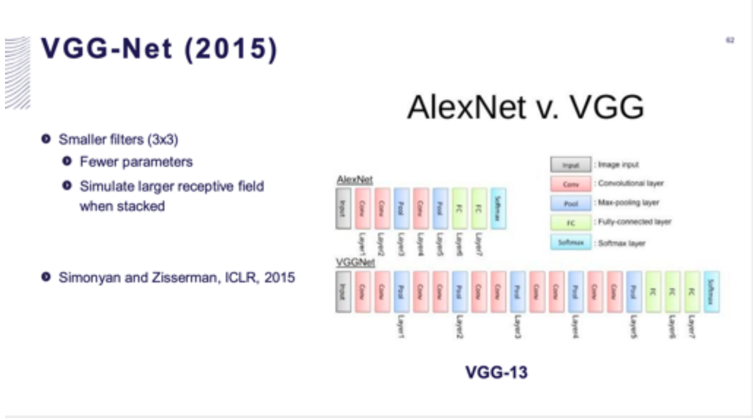


AlexNet

41

- Same model as LeCun'98 but:
  - Bigger model (8 layers)
  - More data (ImageNet)
  - GPU implementation (50x speedup over CPU)
- Better regularization (Dropout)
- 7 hidden layers, 650,000 neurons, 60,000,000 parameters
- Trained on 2 GPUs for a week

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012.



VGG-Net (2015)

62

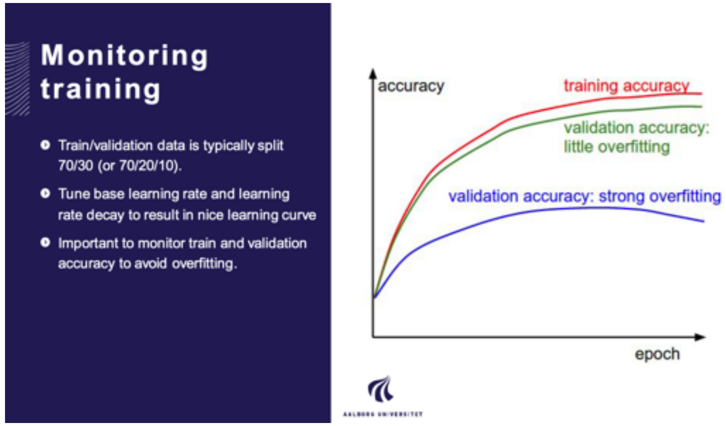
- Smaller filters (3x3)
  - Fewer parameters
  - Simulate larger receptive field when stacked
- Simonyan and Zisserman, ICLR, 2015

AlexNet v. VGG

VGG-13

What about Deep Learning?

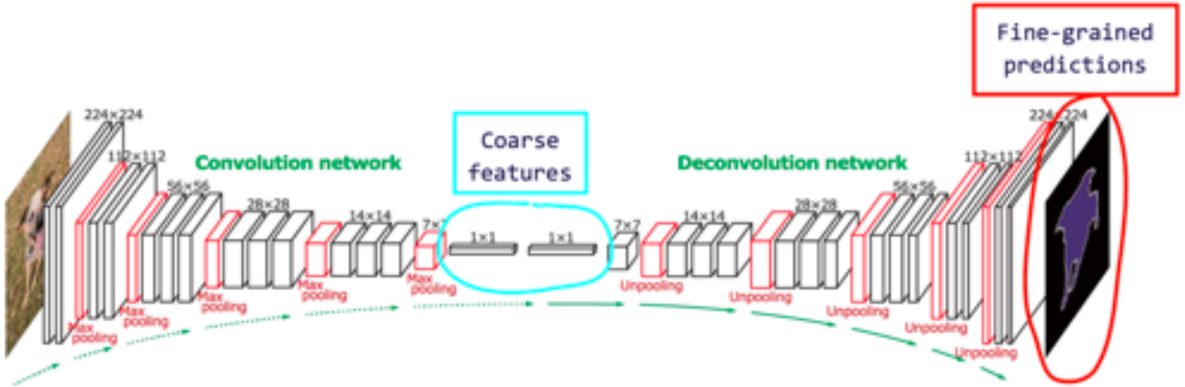
…CNNs, YOLO, evaluating and understanding training…



Monitoring training

- Train/validation data is typically split 70/30 (or 70/20/10).
- Tune base learning rate and learning rate decay to result in nice learning curve
- Important to monitor train and validation accuracy to avoid overfitting.

accuracy
training accuracy
validation accuracy: little overfitting
validation accuracy: strong overfitting
epoch
AALBORG UNIVERSITET



Fine-grained predictions

Coarse features

How do we use a neural network?

- Pre-trained network (as we will do today)
- Transfer learning
  - Fine tuning
  - Define new classes, use own training data
- Train a brand new network
  - Learning everything from scratch
  - Requires significant amount of data

What is the fuzz about? GPUs are cheap and plentiful they said…

https://www.youtube.com/watch?v=-P28LKWTzrI

What about Deep Learning and Robotics?

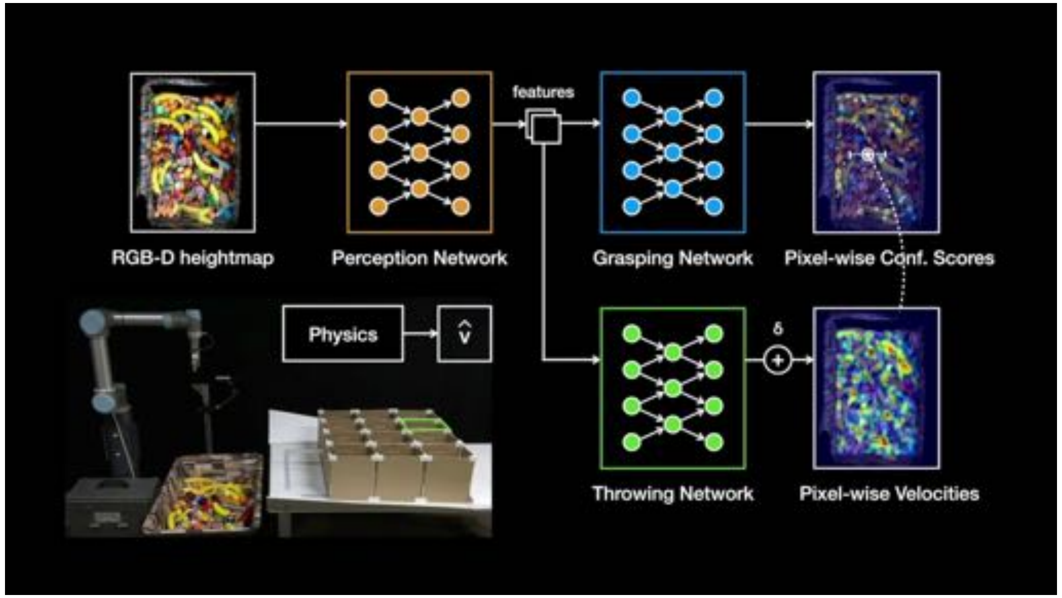
- Deep Learning dominates this field as well
- Robotic applications in every possible field use deep learning
- From predicting the consequences of their possible actions to self-assessing their own competence to complete a task
- Let’s explore some papers

Tossing bot project – Learn to throw stuff

https://www.youtube.com/watch?v=-O-E1nFm6-A

Zeng, A., Song, S., Lee, J., Rodriguez, A. and Funkhouser, T., 2019. Tossingbot: Learning to throw arbitrary objects with residual physics. *arXiv preprint arXiv:1903.11239*. https://arxiv.org/pdf/1903.11239.pdf

Throwing with a robot is VERY difficult!



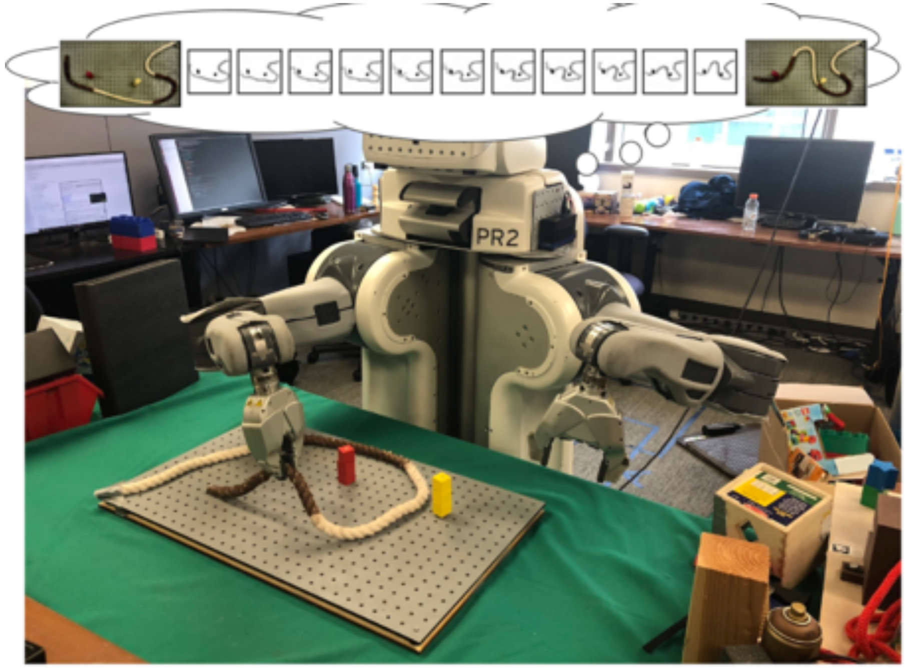
https://www.youtube.com/watch?v=f5Zn2Up2RjQ

End-to-end learning of semantic grasping a.k.a the Google Show-off

https://www.youtube.com/watch?v=iaF43Ze1oeI

Jang, E., Vijayanarasimhan, S., Pastor, P., Ibarz, J. and Levine, S., 2017. End-to-end learning of semantic grasping. *arXiv preprint arXiv:1707.01932*. https://arxiv.org/pdf/1707.01932.pdf

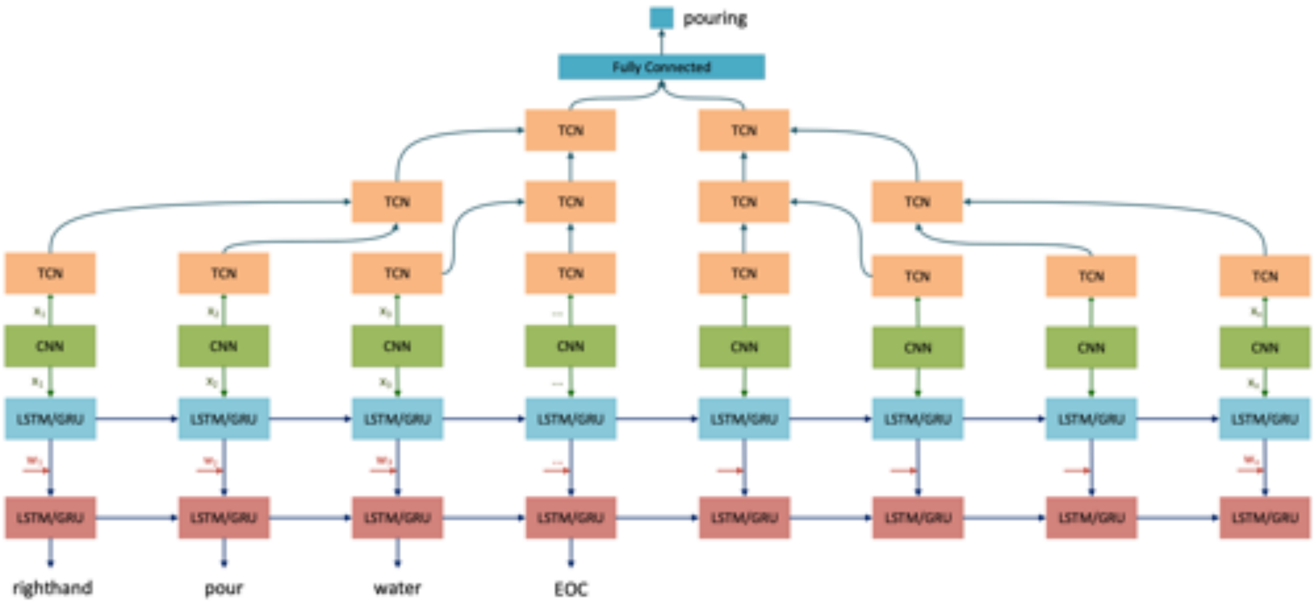
Learning robotic manipulation



https://www.youtube.com/watch?v=XzL1cf8Esfk

Wang, A., Kurutach, T., Liu, K., Abbeel, P. and Tamar, A., 2019. Learning robotic manipulation through visual planning and acting. arXiv preprint arXiv:1905.04411. https://arxiv.org/pdf/1905.04411.pdf

Translate videos to robotic manipulation



https://www.youtube.com/watch?v=h0Ffcx_Otfo

Nguyen, A., Do, T.T., Reid, I., Caldwell, D.G. and Tsagarakis, N.G., 2019. V2cnet: A deep learning framework to translate videos to commands for robotic manipulation. arXiv preprint arXiv:1903.10869.

Grasp planning and learning grasp policies

DexNet 3.0

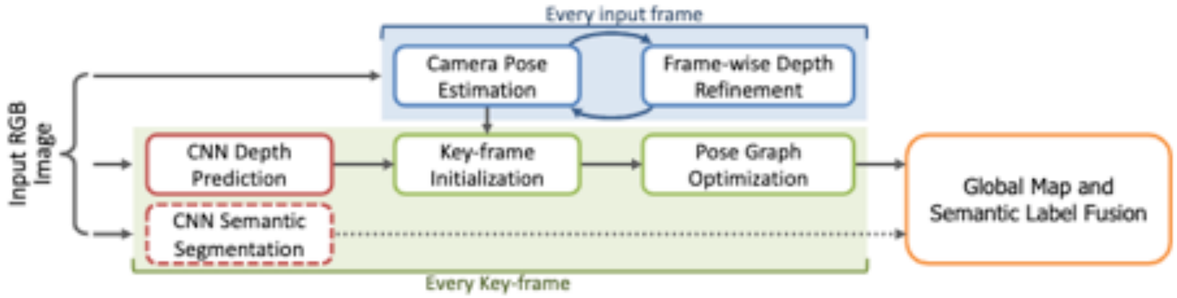
https://www.youtube.com/watch?v=tEoTxvD8AeM

DexNet 4.0

https://www.youtube.com/watch?v=r-0PKne9e_w

Mahler, J., Matl, M., Liu, X., Li, A., Gealy, D. and Goldberg, K., 2018, May. Dex-Net 3.0: Computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning. In 2018 IEEE International Conference on Robotics and Automation (ICRA) (pp. 1-8). IEEE.

CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction

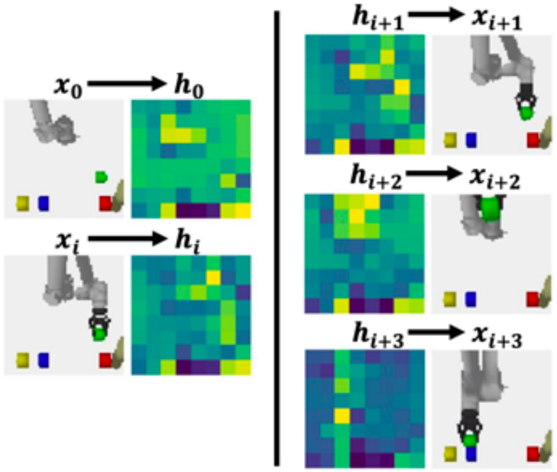


https://www.youtube.com/watch?v=z_NJxbkQnBU

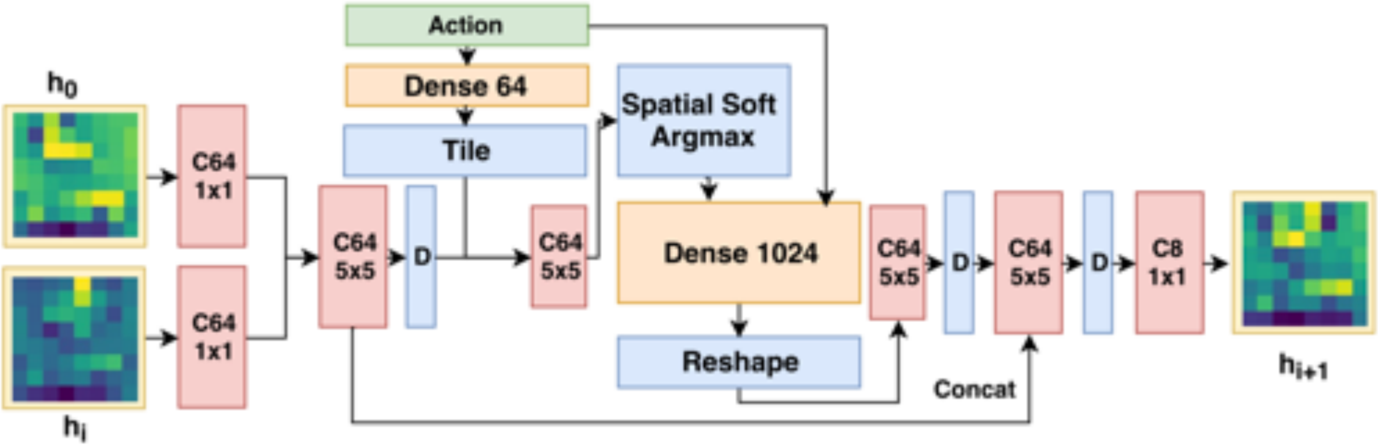
Tateno, K., Tombari, F., Laina, I. and Navab, N., 2017. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 6243-6252).

Visual Robot Task planning

Based on learned and artificially generated policies to predict the consequences of future actions

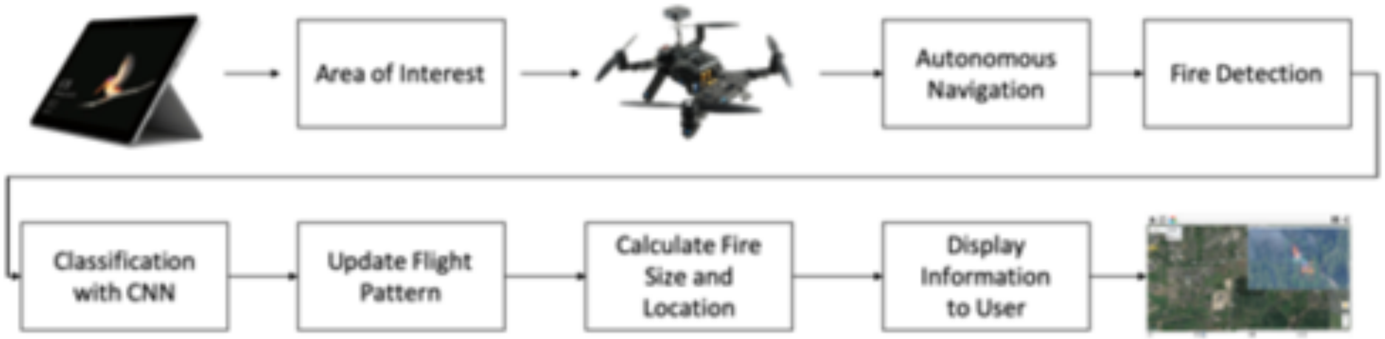


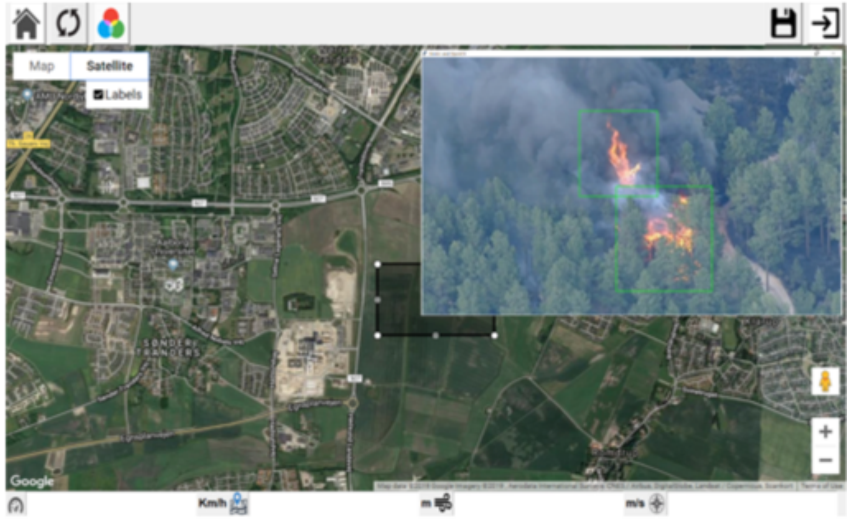
$x_0 \rightarrow h_0$
$x_i \rightarrow h_i$
$h_{i+1} \rightarrow x_{i+1}$
$h_{i+2} \rightarrow x_{i+2}$
$h_{i+3} \rightarrow x_{i+3}$



Paxton, C., Barnoy, Y., Katyal, K., Arora, R. and Hager, G.D., 2019, May. Visual robot task planning. In *2019 International Conference on Robotics and Automation (ICRA)* (pp. 8832-8838). IEEE.

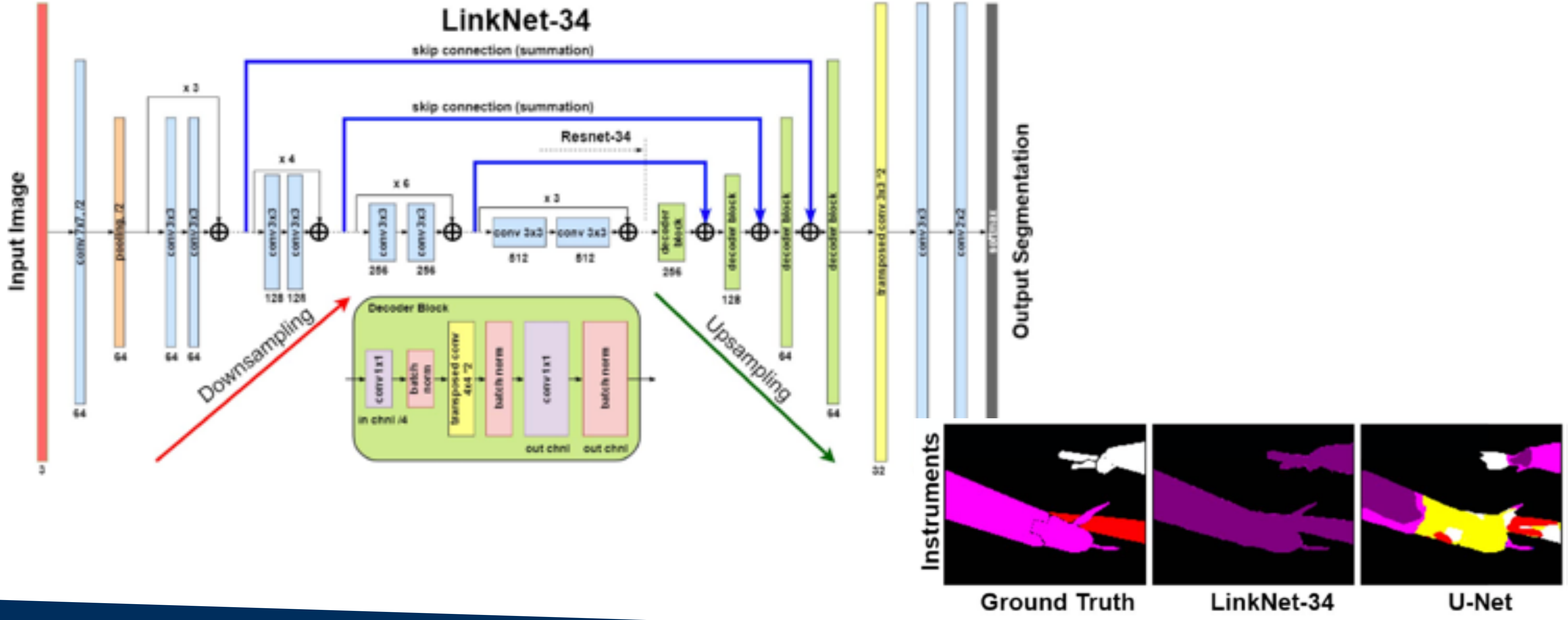
Monitoring wildfires with quadcopters





Novac, I., Geipel, K.R., de Domingo Gil, J.E., de Paula, L.G., Hyttel, K. and Chrysostomou, D., 2020, January. A Framework for Wildfire Inspection Using Deep Convolutional Neural Networks. In *2020 IEEE/SICE International Symposium on System Integration (SII)* (pp. 867-872). IEEE.

Instrument segmentation in robot-assisted surgery



Shvets, A.A., Rakhlin, A., Kalinin, A.A. and Iglovikov, V.I., 2018, December. Automatic instrument segmentation in robot-assisted surgery using deep learning. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 624-628). IEEE.

Useful literature

- Pierson, H.A. and Gashler, M.S., 2017. Deep learning in robotics: a review of recent research. *Advanced Robotics*, *31*(16), pp.821-835.
- Canziani, A., Paszke, A. and Culurciello, E., 2016. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*.
- Nassif, A.B., Shahin, I., Attili, I., Azzeh, M. and Shaalan, K., 2019. Speech recognition using deep neural networks: A systematic review. IEEE Access, 7, pp.19143-19165.
- Burghouts, G.J., Huizing, A. and Neerincx, M.A., 2020. Robotic Self-Assessment of Competence. arXiv preprint arXiv:2005.01546.
- Otter, D.W., Medina, J.R. and Kalita, J.K., 2020. A Survey of the Usages of Deep Learning for Natural Language Processing. IEEE Transactions on Neural Networks and Learning Systems.
- Zacharias, J., Barz, M. and Sonntag, D., 2018. A survey on deep learning toolkits and libraries for intelligent user interfaces. arXiv preprint arXiv:1803.04818.
- Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S. and Dean, J., 2019. A guide to deep learning in healthcare. Nature medicine, 25(1), pp.24-29.

What about Deep Learning frameworks?

Too many…

Choose your adventure…

TensorFlow

Keras

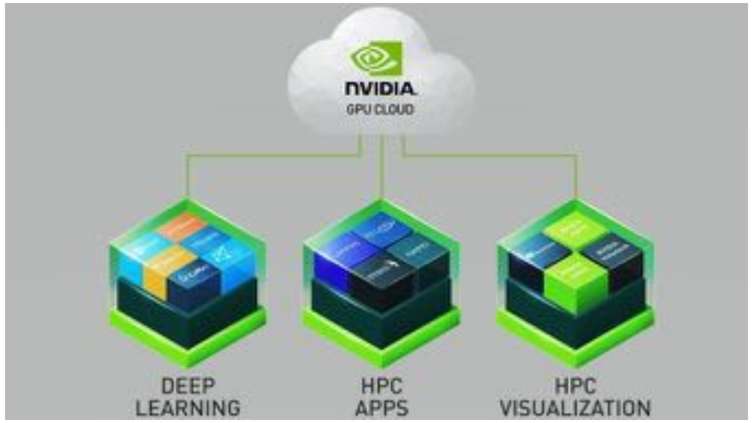
PyTorch

Caffe

theano

https://en.wikipedia.org/wiki/Comparison_of_deep-learning_software

…and your online playground



Amazon Machine Learning

colab

Time for some practical applications!

Let’s produce some data

& fight against COVID-19

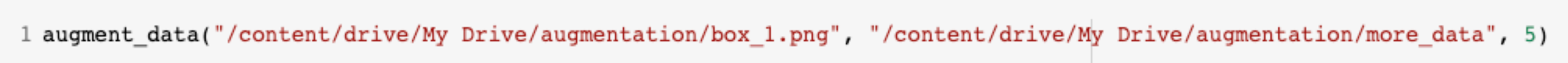
Let’s produce some data

- What is data augmentation?
  - Answer on Teams

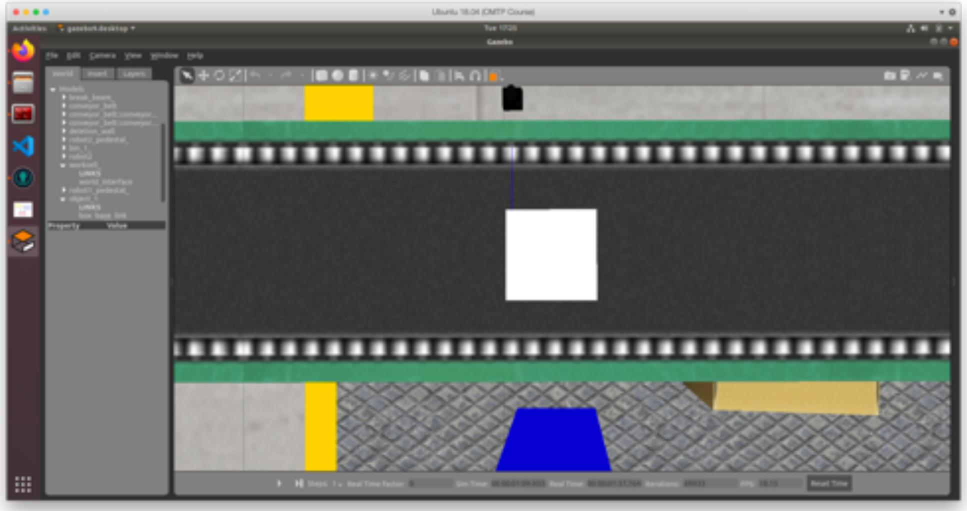
Easy setup

1. From moodle -> Download box_generation.zip -> Unzip
2. Upload the contents of the zip file to your Google drive *(will take some time depending on your network speed)*
3. Login to Google Colab
4. Mount your Google Drive
5. File -> Upload/Open Notebook
6. Select box_generation.ipynb from your Google Drive or from hard drive
7. Edit -> Notebook Settings -> Hardware acceleration -> GPU
8. Run the script and generate new image data

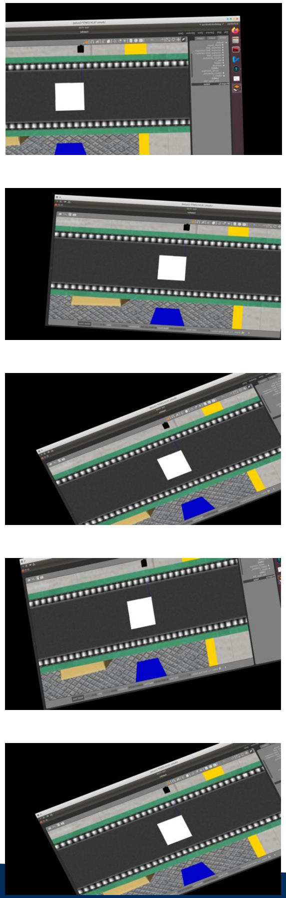
Lots and lots of data



```
augment_data("/content/drive/My Drive/augmentation/box_1.png", "/content/drive/My Drive/augmentation/more_data", 5)
```

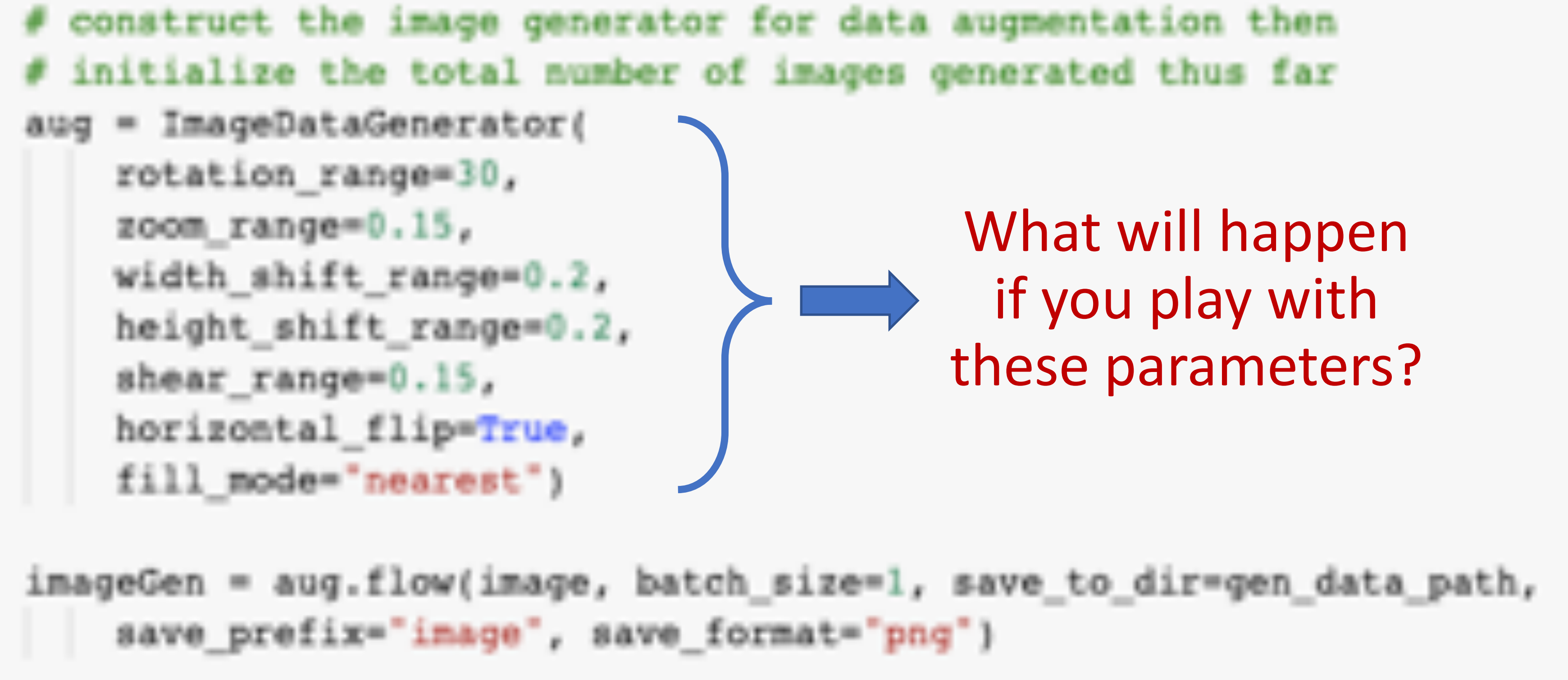


Your original image



Number of augmented images

Experiment with the parameters

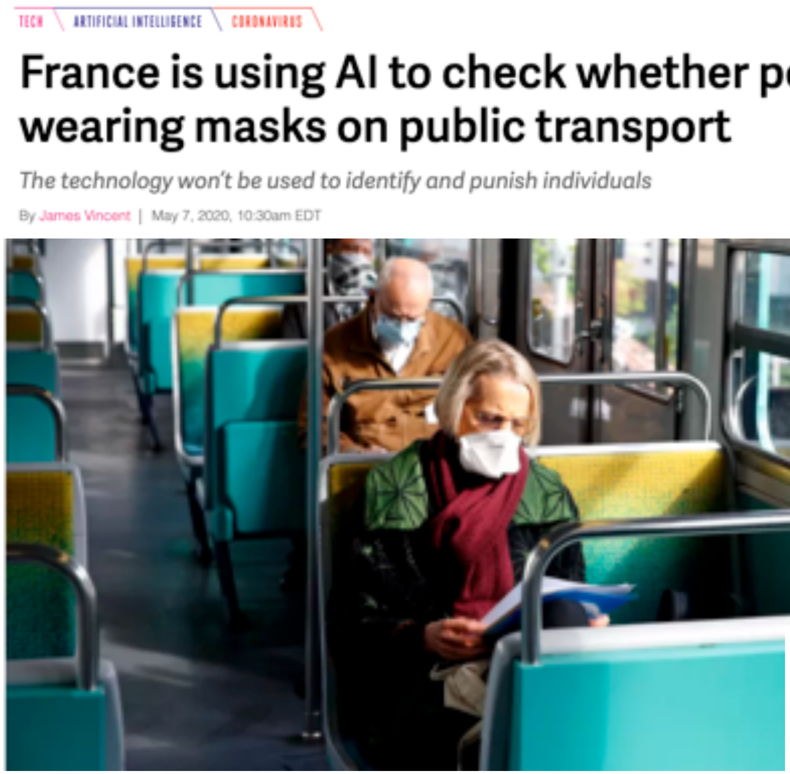


```
# construct the image generator for data augmentation then
# initialize the total number of images generated thus far
aug = ImageDataGenerator(
        rotation_range=30,
        zoom_range=0.15,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.15,
        horizontal_flip=True,
        fill_mode="nearest")

imageGen = aug.flow(image, batch_size=1, save_to_dir=gen_data_path,
        save_prefix="image", save_format="png")
```

What will happen if you play with these parameters?

Hint: https://keras.io/api/preprocessing/image/



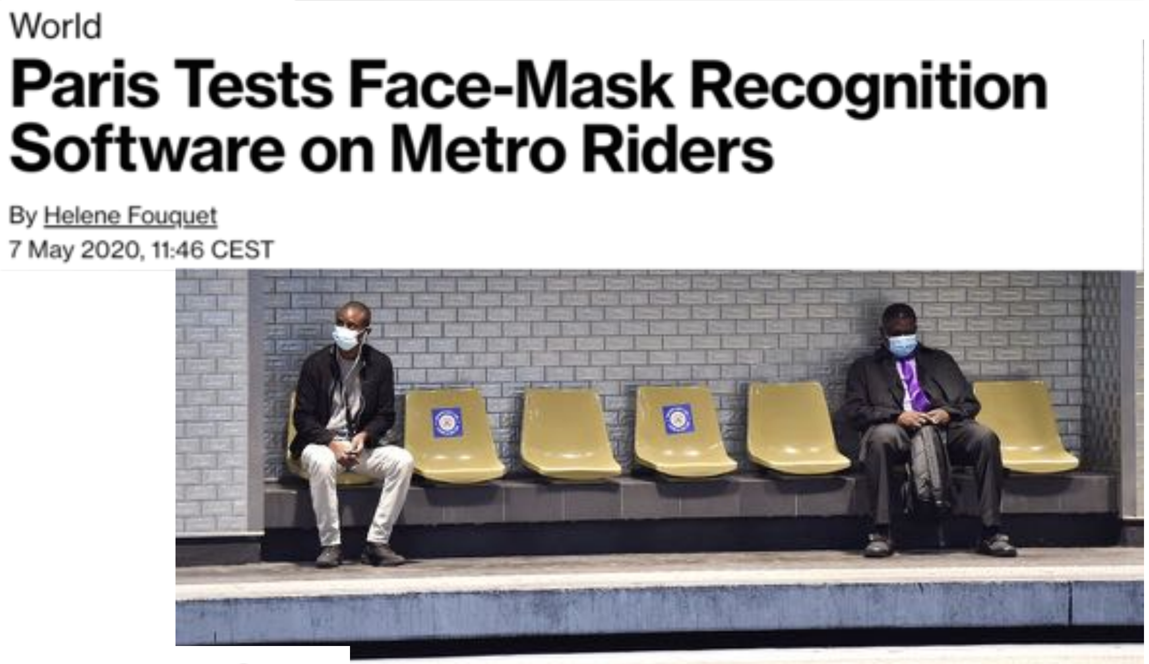
TECH \ ARTIFICIAL INTELLIGENCE \ CORONAVIRUS

France is using AI to check whether people are wearing masks on public transport

The technology won't be used to identify and punish individuals

By James Vincent | May 7, 2020, 10:30am EDT

https://www.theverge.com/2020/5/7/21250357/france-masks-public-transport-mandatory-ai-surveillance-camera-software?_hsenc=p2ANqtz-9L-Z1IQRtr0eXEuDxN6VLANb5tU0fUtvRmSptURnrYwQVvT0V-AoKO2K40cP8yyG2fEclsBkU1zjT0Rvw1QwJa9zB-gg&_hsmi=87877396



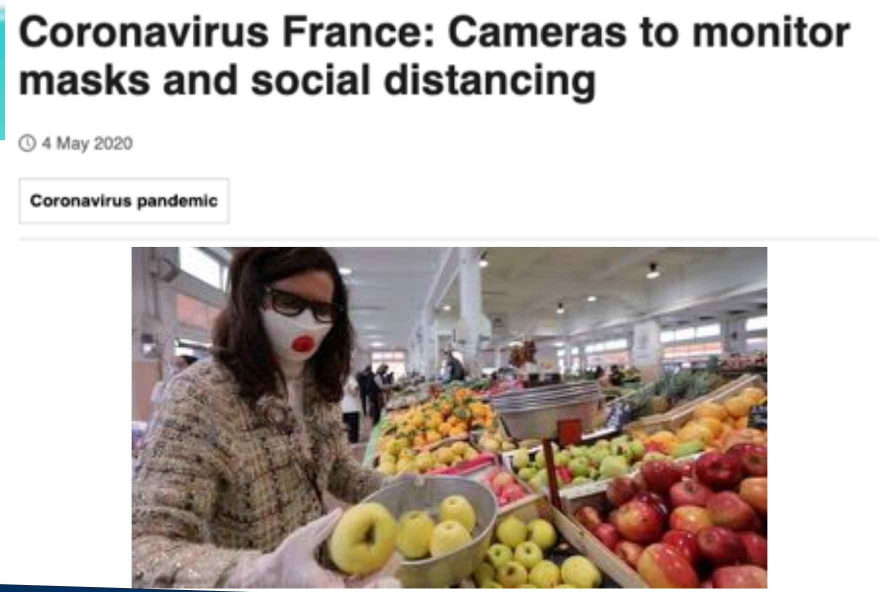
World

Paris Tests Face-Mask Recognition Software on Metro Riders

By Helene Fouquet

7 May 2020, 11:46 CEST

https://www.bloomberg.com/news/articles/2020-05-07/paris-tests-face-mask-recognition-software-on-metro-riders



Coronavirus France: Cameras to monitor masks and social distancing

🕓 4 May 2020

Coronavirus pandemic

https://tinyurl.com/yare5b2r

Let’s do something useful…

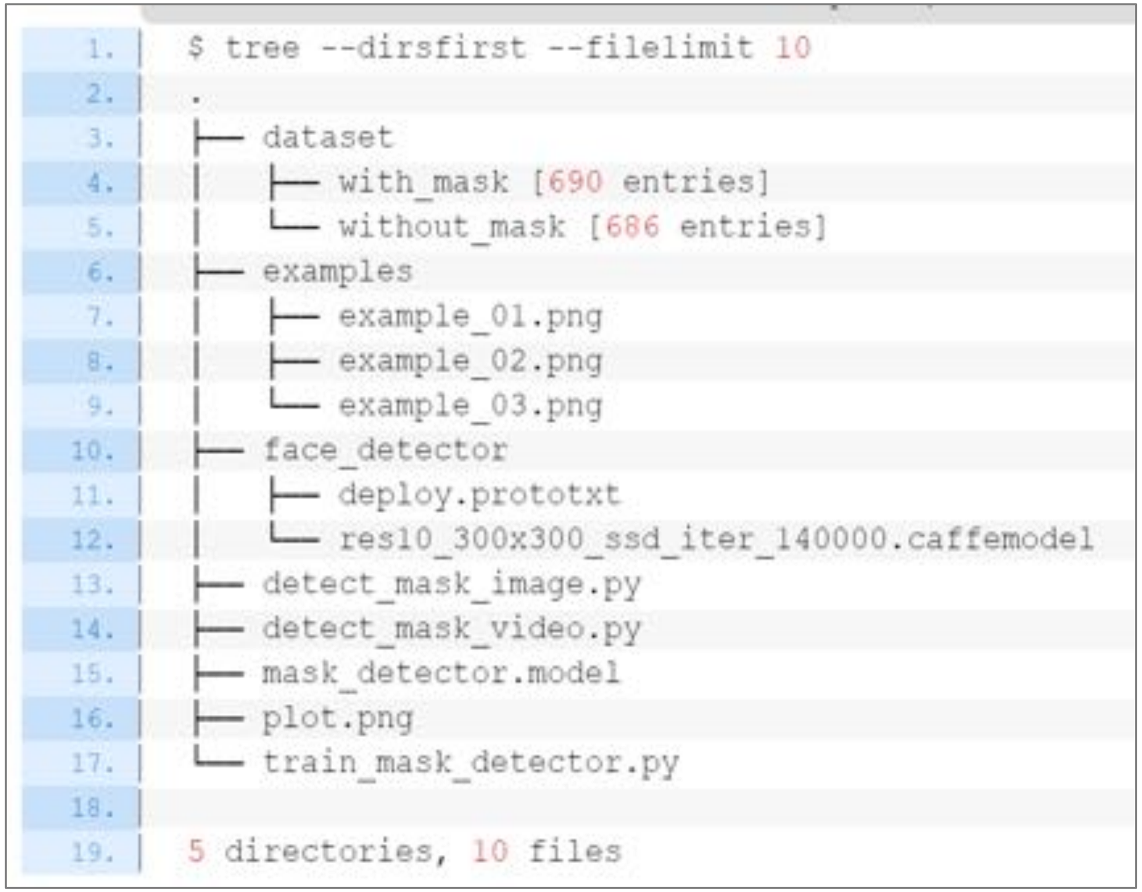
Face Mask Detector with OpenCV, Keras/TensorFlow, & Deep Learning

1. From moodle -> Download covid19_colab.zip -> Unzip
2. Upload the contents of the zip file to your Google drive *(will take some serious time depending on your network speed)*
3. Login to [Google Colab]()
4. Mount your Google Drive
5. File -> Upload/Open Notebook
6. Select covid19_mask.ipynb from your Google Drive or from hard drive
7. Edit -> Notebook Settings -> Hardware acceleration -> GPU

Project structure

- **Sequence**
  - Train_mask_detector.py
  - Detect_mask_image.py
  - Detect_mask_video.py
- **Examples**

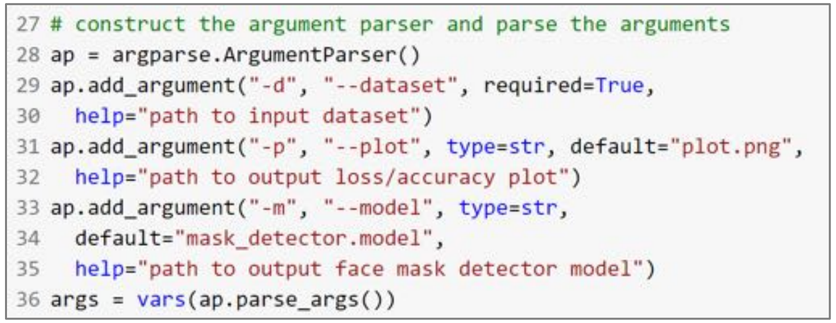
It contains three example images for testing the trained model. You can also upload more images to this folder to do your own tests.



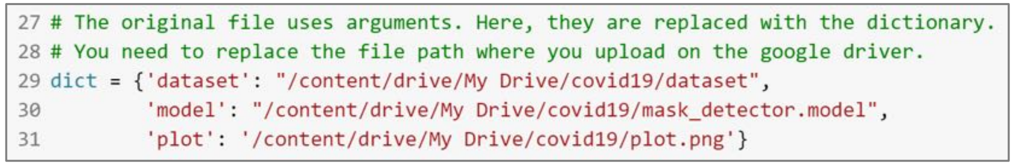
```
$ tree --dirsfirst --filelimit 10
.
├── dataset
│   ├── with_mask [690 entries]
│   └── without_mask [686 entries]
├── examples
│   ├── example_01.png
│   ├── example_02.png
│   └── example_03.png
├── face_detector
│   ├── deploy.prototxt
│   └── res10_300x300_ssd_iter_140000.caffemodel
├── detect_mask_image.py
├── detect_mask_video.py
├── mask_detector.model
├── plot.png
└── train_mask_detector.py

5 directories, 10 files
```

Some modifications - Train_mask_detector.py

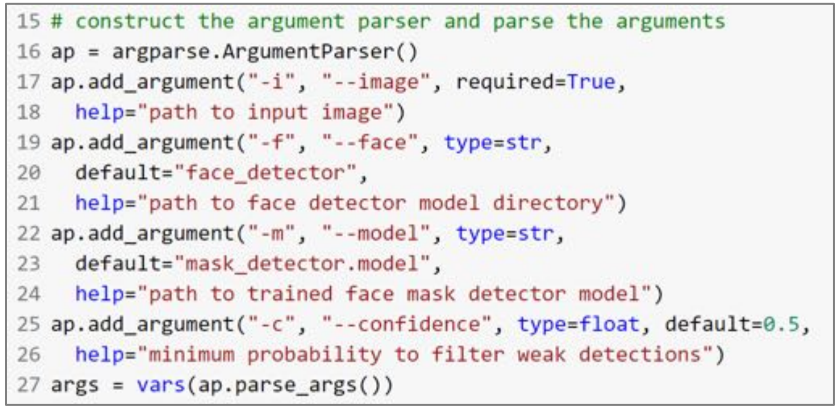


```python
# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-d", "--dataset", required=True,
	help="path to input dataset")
ap.add_argument("-p", "--plot", type=str, default="plot.png",
	help="path to output loss/accuracy plot")
ap.add_argument("-m", "--model", type=str,
	default="mask_detector.model",
	help="path to output face mask detector model")
args = vars(ap.parse_args())
```

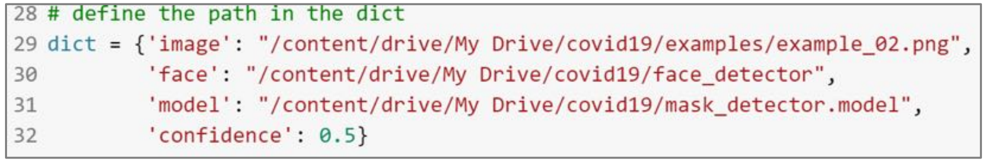


```python
# The original file uses arguments. Here, they are replaced with the dictionary.
# You need to replace the file path where you upload on the google driver.
dict = {'dataset': "/content/drive/My Drive/covid19/dataset",
        'model': "/content/drive/My Drive/covid19/mask_detector.model",
        'plot': '/content/drive/My Drive/covid19/plot.png'}
```

Some modifications - detect_mask_image.py



```python
# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=True,
  help="path to input image")
ap.add_argument("-f", "--face", type=str,
  default="face_detector",
  help="path to face detector model directory")
ap.add_argument("-m", "--model", type=str,
  default="mask_detector.model",
  help="path to trained face mask detector model")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
  help="minimum probability to filter weak detections")
args = vars(ap.parse_args())
```



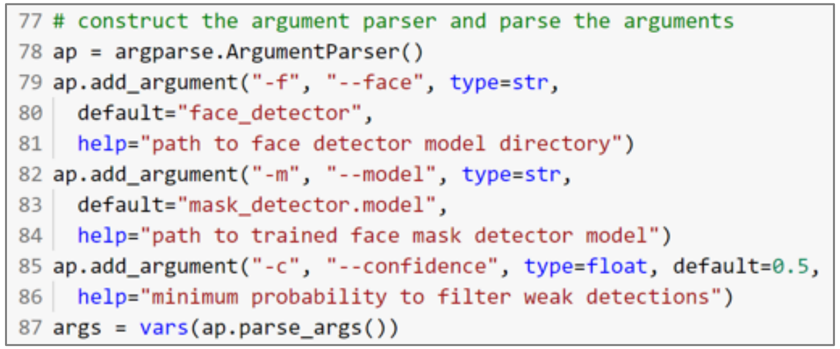
```python
# define the path in the dict
dict = {'image': "/content/drive/My Drive/covid19/examples/example_02.png",
        'face': "/content/drive/My Drive/covid19/face_detector",
        'model': "/content/drive/My Drive/covid19/mask_detector.model",
        'confidence': 0.5}
```

Some modifications - detect_mask_image.py

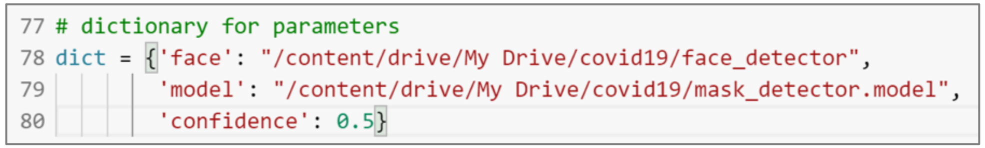
Show image

- Since cv2.imshow() is not working in Google Colab, we need to change it
- from google.colab.patches, **“import cv2_imshow”** is added to the “detect_mask_image” script
- Replace cv2.imshow() to cv2_imshow()
- Or just use the edited file from the lecture material…

Some modifications - detect_mask_video.py



```
# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-f", "--face", type=str,
    default="face_detector",
    help="path to face detector model directory")
ap.add_argument("-m", "--model", type=str,
    default="mask_detector.model",
    help="path to trained face mask detector model")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
    help="minimum probability to filter weak detections")
args = vars(ap.parse_args())
```



```
# dictionary for parameters
dict = {'face': "/content/drive/My Drive/covid19/face_detector",
        'model': "/content/drive/My Drive/covid19/mask_detector.model",
        'confidence': 0.5}
```

Some modifications - detect_mask_video.py

Run the Sequence as is

1. Train_mask_detector.py (~10 mins depending on no. of epochs and local hardware)
2. Detect_mask_image.py
3. Detect_mask_videopy

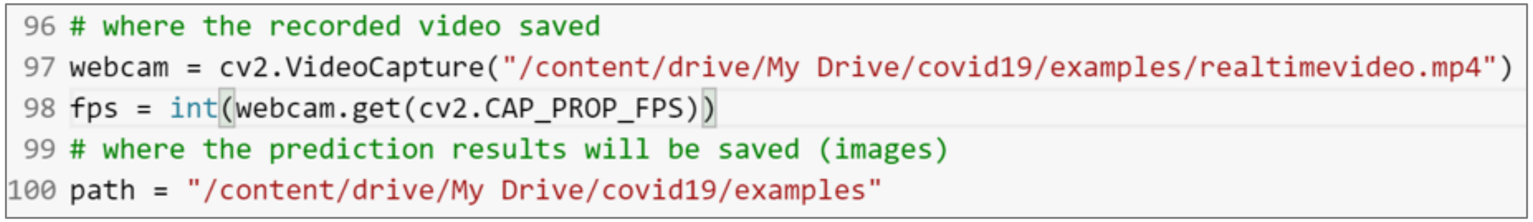
The original *detect_mask_video.py* cannot be run directly from the Colab.

Since the code is actually running on Google VM, you don't have a camera there but only locally on your computer

There is an extra script to use your webcam to record a realtime video and upload to your Google Drive.

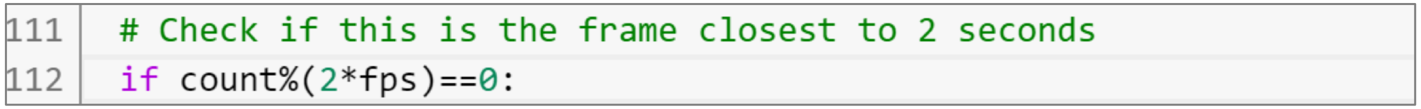
The last part (*detect_mask_video.py*) can detect if you are wearing a mask in the video or not 😷

Some modifications - detect_mask_video.py



```python
# where the recorded video saved
webcam = cv2.VideoCapture("/content/drive/My Drive/covid19/examples/realtimevideo.mp4")
fps = int(webcam.get(cv2.CAP_PROP_FPS))
# where the prediction results will be saved (images)
path = "/content/drive/My Drive/covid19/examples"
```

1. Change the video path - **“/content/drive/My Drive/covid19/examples/realtimevideo.mp4”** to where your video is saved
2. Change the prediction result path – **“/content/drive/My Drive/covid19/examples”** to where you want to save the prediction results.



```python
    # Check if this is the frame closest to 2 seconds
    if count%(2*fps)==0:
```

3. takes sample image **every 2 seconds** from the video, you can change to other time interval here.

Recap

- Why Deep Learning and CNNs are important
- How CNNs are used in robotics
- What type of Deep Learning frameworks are out there
- Put everything together for an application for a face mask detection in the COVID-19 era

Lecture material

- Play with the code and try different aspects of it. How does it perform?

- Download the papers and keep them for further study

Assignment for all

- Make the code run on your own computer using Google Colab or locally.

- Record a (very short) video of yourself detecting your face…do you wear a mask or not?

- As always, submit assignment to git repo – one short video per team member

Optional assignment #1

- Exchange MobileNetv2 with your favorite network (surprise me!) and show how it performs 🤓
- Bonus question: can we just use whichever network for this application?
- As always, submit assignment to git repo and post on MS Teams

Optional assignment #2

Use it in your OMTP environment and train your simulated camera to identify two classes of (any kind of) objects 😎

Recommendations for future endeavors

- Explore transfer learning
- Always evaluate your performance (IoU, mAP, …)
- Explore TensorFlow and other DL possibilities in ROS
- Choose your territory – Computer Vision, NLP, Speech Recognition, RL – and be an expert in it
- Follow your experts in your field of choice
- Follow the flagship conferences in the field and arXiv.org

- Don’t be too religious on specific frameworks, platforms and/or tools.

https://youtu.be/cJMwBwFj5nQ