Introduction to Artificial Intelligence

ICPBL Project Outline

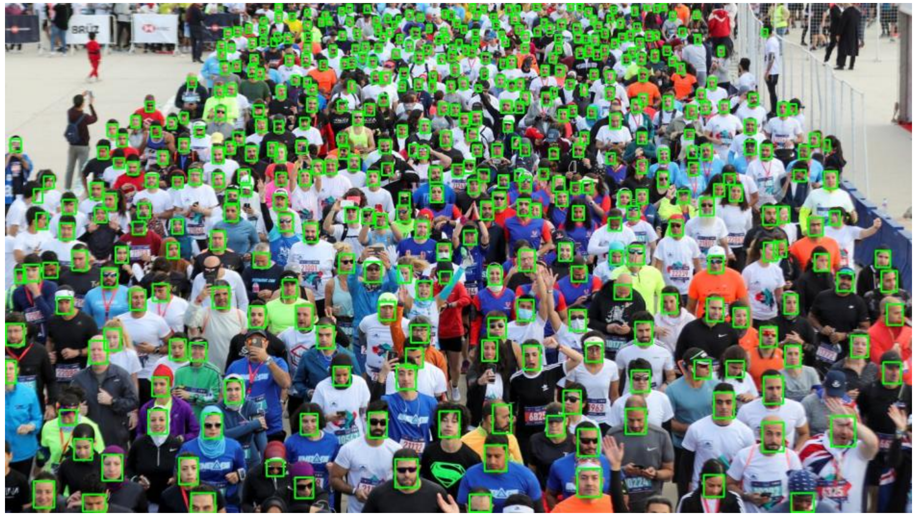
Je Hyeong Hong
25 May 2022
Department of Electronic Engineering

MOTIVATION

- Since the outbreak of COVID-19, many countries by law still require wearing facial masks in public indoor places.
- You are working for a company "Hanyang Corporations" making smart kiosks to check if a person is wearing a mask upon entrance to the building.
- You are to develop an AI model which classifies whether a person is wearing a mask or not.

SURROUNDING CONDITIONS

- Many public dataset exists for non-masked faces, but not so many for masked ones (at least not enough for training a deep network).
  - You need to mimic somewhat realistic masked faces.
- Fortunately, you have close (helpful) friends (Hyeonjeong Park and Jonggyu Park) who have already detected, cropped and aligned faces for you.



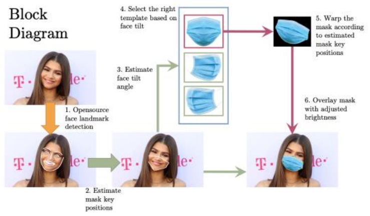
GOAL

- Given well cropped-and-aligned images of non-masked faces, develop a deep learning model that can classify whether a person in each image is wearing a mask or not.
  - Binary classification task
- Everything should work in Google Colab

IMPORTANT INFORMATION

- **Deadline**: 14th June 2022 11.59pm
  (1 day before the final exam)
- **Dataset**: click here
- **Submit 2 files to LMS**
  1. **A pdf report**
  2. **A zip file containing your code**
     1. **train.ipynb**: a notebook file for training (and saving) your model
        - Please include tensorboard visualization as well
        - Base skeletal code structure provided here
     2. **eval.ipynb:** another notebook file for loading your model and just testing with your own photo
        - Base skeletal code structure provided here (4. toy example)

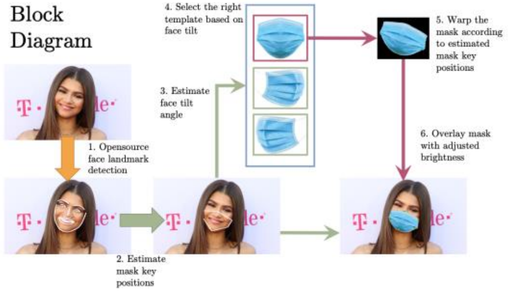
HOW TO PROCEED


Block Diagram
1. Opensource face landmark detection
2. Estimate mask key positions
3. Estimate face tilt angle
4. Select the right template based on face tilt
5. Warp the mask according to estimated mask key positions
6. Overlay mask with adjusted brightness

1. Download aligned-and-cropped face images
   - Cropped using RetinaFace
2. Use MaskTheFace tool to create augmented face images wearing virtual masks.
3. Import a Resnet-50 architecture from the model zoo in pytorch torchvision with a **single output**.
4. Train the model using dataset from above and using BCELossWithLogitsLoss.
   - Calculates binary cross entropy after adding a sigmoid function to the output. (avoid numerical issues with near-0 or near-1 activations from $\sigma(\cdot)$)
5. Test your model
   1. We will evaluate your model and weights.
   2. Try your own (or your favorite celebrity’s) photo(s).

SYNTHETIC DATA GENERATION (5 %)

- Download data.zip - Google Drive
- The folder is divided into training (train) and validation (val) set.
- In each folder,
    - “not_wearing_mask” contains images not wearing masks.
    - “wearing_mask” is empty.
- You need to apply the MaskTheFace tool to generate images to be included in the “wearing_mask” directory for both training and validation sets.


Block Diagram
1. Opensource face landmark detection
2. Estimate mask key positions
3. Estimate face tilt angle
4. Select the right template based on face tilt
5. Warp the mask according to estimated mask key positions
6. Overlay mask with adjusted brightness

DATA AUGMENTATION (15 %)

The original image is 128x128. Use **pytorch dataloader (ImageFolder)** for both training and validation data to perform the followings:

- **Add random rotation**
- **Add random flip**
- **Add random scaling**
- Add any other augmentation you wish.
- **Crop to 112x112**

Please check this guide: Transforming and augmenting images — Torchvision main documentation (pytorch.org)

- Note: transforming an image to tensor using transforms.ToTensor() normalizes values from between 0 to 255 to between 0 and 1.

MODEL (5%)

- Import a Resnet-50 architecture from model zoo in Torchvision.
  - torchvision.models — Torchvision 0.8.1 documentation (pytorch.org)
- Define the output dimension to be 1.
  - Not_wearing_mask: 0
  - Wearing_mask: 1

TRAIN

- Use SGD optimizer
- Use BCEWithLogitsLoss $l_n = -w_n \left[y_n \cdot \log \sigma(x_n) + (1 - y_n) \cdot \log(1 - \sigma(x_n)) \right]$
- Run for at least 20 epochs
- Save your model after training
  - Saving and Loading Models — PyTorch Tutorials 1.11.0+cu102 documentation

VISUALIZATION (10 POINTS)

- Show training loss and training accuracy using Tensorboard
  - Accuracy: how many times you get things correct over training samples.
- Show validation loss and validation accuracy using Tensorboard:
  - Accuracy: how many times you get things correct over validation samples.

- Follow the **linear regression coding example from week 10**!

SCORES

- Synthetic data generation: Applying MaskTheFace (5 points)
- Data augmentation: Designing your dataloader in pytorch for loading training and validation data as well as data augmentation (15 points)
- Importing a correct Resnet-50 model from Torchvision model zoo and drawing/describing the Resnet-50 architecture in your report (5 points)
- Training (30 points)
  - Define SGD optimizer and BCEWithLogitsLoss
- Visualization (10 points)
- Qualitative evaluation (15 points): try your custom images
- Discussions (15 points)
- Evaluation on the test set (5 points)

FOR YOUR REPORT (MAX 3 PAGES, 11PT FONT)

Please include:

- A diagram of the Resnet-50 architecture and detailed descriptions.
- Some samples of augmented images using MaskTheFace.
- Plots
  - Training loss vs epoch
  - Average training set accuracy vs epoch
  - Verification loss vs epoch
  - Average verification set accuracy vs epoch
- Discussions
  - Choose 2 ablation studies from below:
    - Changing hyperparameters (learning rate, batch size, etc.)
    - Turning on or off random data augmentation
    - Turning on or off regularization (weight decay, early stopping, etc.)
    - Changing model architecture (Resnet 152, etc.)
    - Changing initialization procedure for weights and biases.
  - Also discuss limitations of the current model.
- Qualitative evaluation: classification results on some of your (or your favorite person’s) photos

IMPORTANT REMINDERS

- You’re allowed to discuss with us or friends but please don’t just copy each other’s code or report.
  - **PLAGIARISM WILL NOT BE TOLERATED**
  - This exercise is designed to help you understand the training process.
  - When you get stuck in some errors whilst coding, the quickest way to try is search for a similar post on Google. (it’s a good practice)

MISCELLANEOUS POINTS

- Follow an example on MNIST data:
  - examples/main.py at main · pytorch/examples · GitHub
- Need to learn how to load images to pytorch tensors using dataloaders and torch.transforms.
- Need to know how to move data between CPU and GPU

Your question or feedback is always welcome

Too fast or too slow? Too difficult or too easy?

Send message via e-mail or Zoom

융합전자공학부