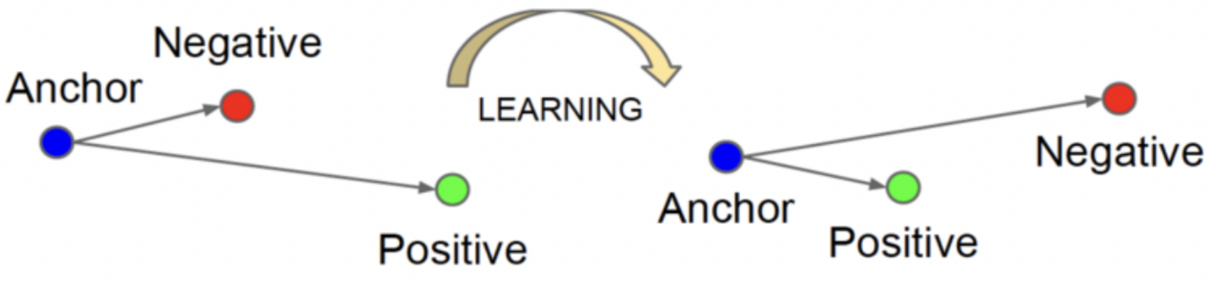
COMS4995: Deep Learning
Prof. Iddo Drori

Kinship Recognition competition

Je Seung You (jy2908)

☐ Base Network: Triplet Network



FaceNet: A Unified Embedding for Face Recognition and Clustering

- I approached this problem by using metric learning, which is to utilize a metric that calculates distances representing “identical” or “non-identical,” instead of classification.
- Also, I happened to find that there is a reference using Triplet network for facial recognition
    - https://arxiv.org/abs/1503.03832
    - https://github.com/tbmoon/facenet
- So, given my research, I chose to implement Triplet Network, which measures loss from anchor to positive and negative data points.
    - If it belongs to the same family, then it is positive
    - Otherwise it is negative
- Pro: each MID represents each person, and being able to see different age stages for a certain person would be beneficial to check kinship.
- Con: There are some cases that some individuals are included in a family even if they are not kinship, which results in label noise (difficulty towards learning)

☐ CNN Triplet Network (Why Triplet?)

- CNN is known to be effective for technologies with deep learning, such as autonomous driving, face recognition, and even big data healthcare systems.
- CNN performs in a way of learning the texture shape of each datapoint, after converting its datatype to a certain format
- CNN measures and learns the relationship among various data points based on Euclidean Distance

- However, it seems hard for CNN to classify the data points that share similar traits. Also, since Triplet emphasizes relationships compared to CNN, I decided to use Triplet for this competition.

☐ Methods applied to increase the performance of the Triplet network

- Preprocessing stage:
    - Face Super Resolution: https://github.com/ewrfcas/Face-Super-Resolution
        - Why?: Since resizing smaller images to larger scale (to 224 x 224) would make the photo blurry, I thought that super resolution would benefit in terms of increasing resolution.
        - I preprocessed the images in the given train dataset with the generator trained with 90000 and 200000 iterations, but it did not positively affect the performance.
    - Augmentation:
        - RandomHorizontalFlip
        - RandomAffine
        - ColorJitter
        - RandomGrayscale
        - Resize
        - Normalize
        - RandomCrop
- Model Structure:
    - Triplet network
    - Classifiers: family, identity
    - Vgg2face - https://github.com/cydonia999/VGGFace2-pytorch
        - Senet: https://arxiv.org/pdf/1710.08092.pdf
        - Download senet50_ft to run train.ipynb: https://drive.google.com/file/d/1YtAtL7Amsm-fZoPQGF4hJBC9ijjjwiMk/view
- Model Tuning:
    - Weight decay: Used it as a regularizer (to avoid overfitting)
    - Fine Tuning:
        - It ended up with overfitting - so I ended up not implementing this
        - It generated a better train pair accuracy, but it ended up not improving the performance of prediction (test accuracy x)
    - Ensemble:
        - I used Excel with the distance metrics
        - Generated better prediction results when merging several model predictions

☐ Training Execution Pipeline (with Google Colab)

- Procedure:
    - 1) SuperResolution.ipynb (optional)

    - 2) train.ipnb
    - 3) test.ipynb
    - 4) ensemble with best models (MS Excel)
- Run faceSuperResolution.ipynb
    - preprocess the images in our train set
    - The final submission did not use this (as the performance was not improved)
- Run train.ipynb
    - Generates `model.pth.tar`
- Run test.ipynb
    - Download `model.pth.tar`
    - Run test.ipynb, and submit predictions.csv to the competition