SAPIENZA
UNIVERSITÀ DI ROMA

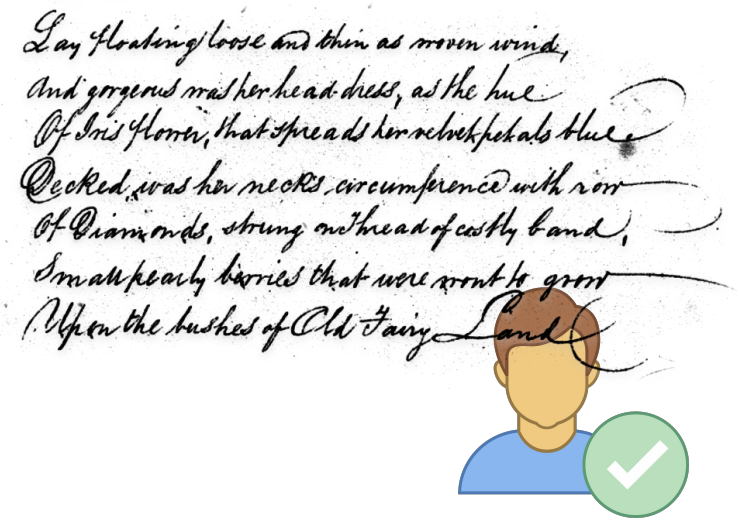
# Handwriting Verification

## In Biometric Systems

**Mohammadreza Shabani - 1966731**
**Hassan Teymoori - 1947458**

Thanks to Professor :
**Maria De Marsico**

June 2022



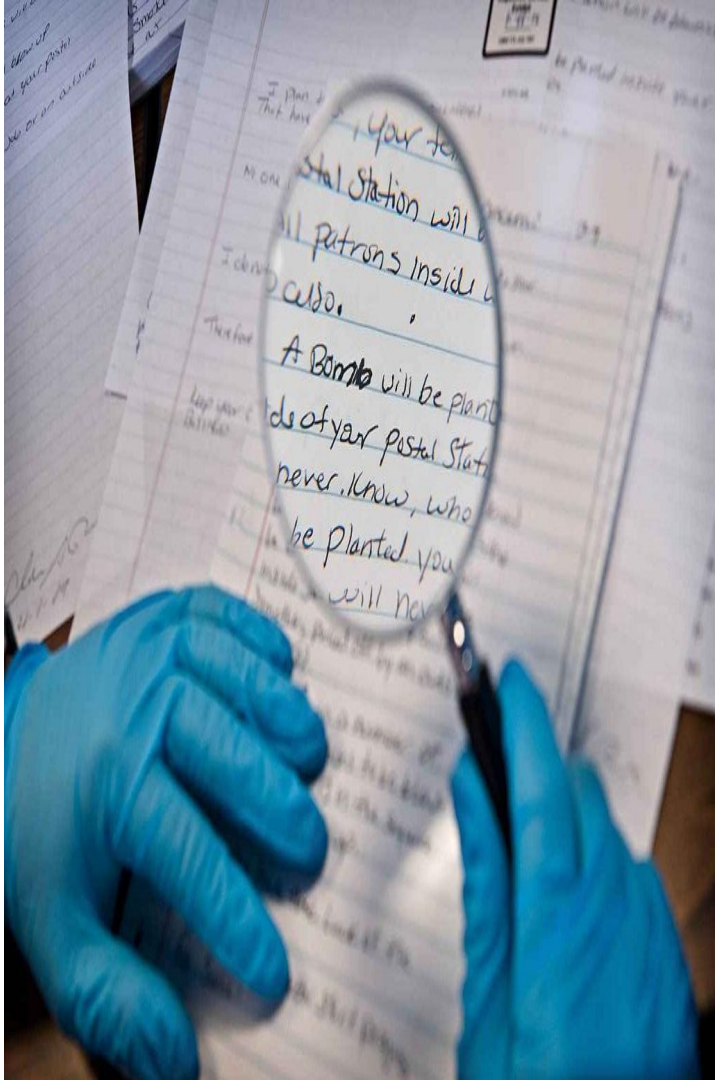
# TABLE OF CONTENTS

01

# Introduction

# Introduction: Motivation

- Help in determining the valid writer based on their handwritten text
- Used in various fields like security, literature analysis, and signature analysis
- Handwritten documents can be primarily divided into two classes:
  - Online → documents contains
    - the coordinates of pen movements,
    - pen-up and pen-down events.
    - etc..
  - Offline → documents are scanned images of handwritten documents.

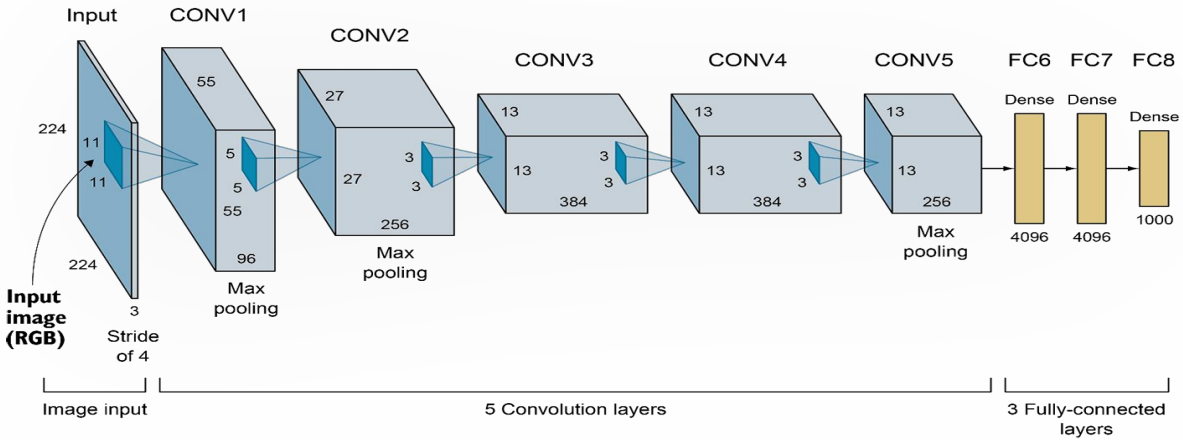


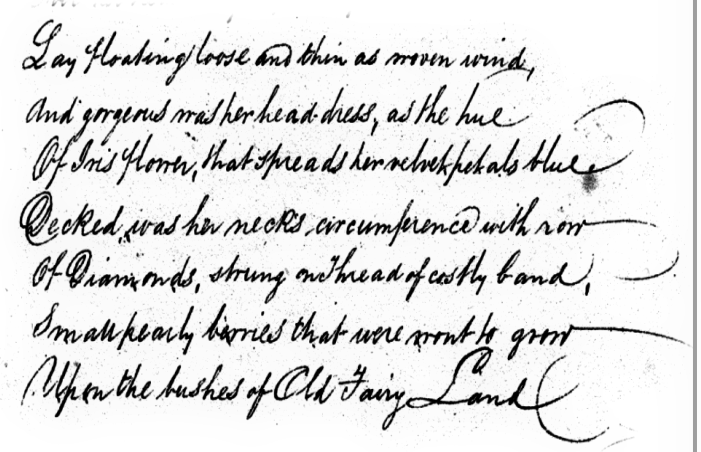
# Introduction: Architecture

The project is done in three stages:

1. Image pre-processing to remove noise
2. Feature extraction using AlexNet-based CNN architecture (We changed a bit during training)
   - ➢ 5 convolutional , 3 max-pooling, 2 normalization, 2 FC, and 1 softmax layers
   - ➢ Each convolutional layer consists of convolutional filters and a function ReLU
   - ➢ Input size is fixed due to the FC layers(224 x 224)

3. Verification using all-against-all multiple template approach

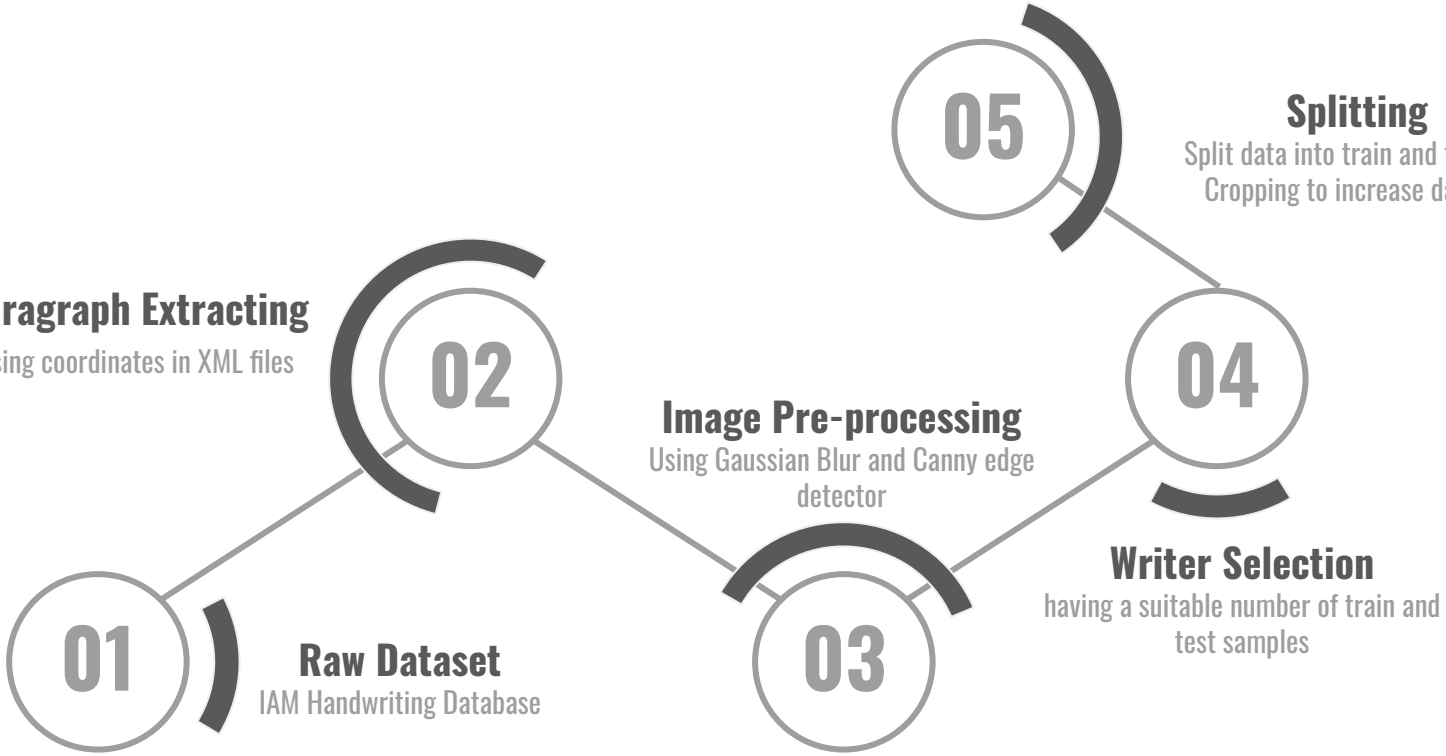





02

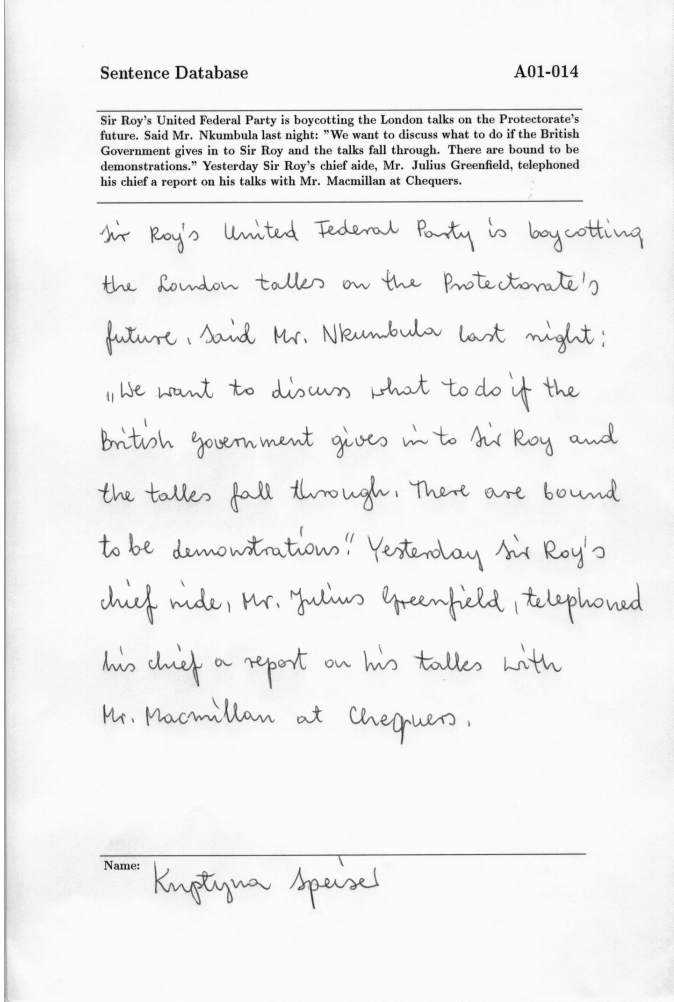
# Data Providing & Pre-Processing

# Data Providing and Image Pre-processing Procedure



# Data Providing: Raw Dataset

- The IAM Handwriting Database (Link)
- The database has some characteristics such as:
  - Images in resolution of 300dpi
  - PNG images with 256 gray levels
  - 657 different writers
  - English text
  - Writer ID appears on top of the forms
  - XML file contains metadata



Sentence Database A01-014

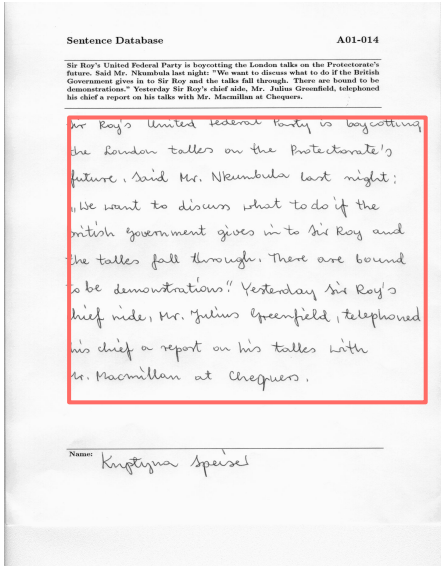
Sir Roy's United Federal Party is boycotting the London talks on the Protectorate's future. Said Mr. Nkumbula last night: "We want to discuss what to do if the British Government gives in to Sir Roy and the talks fall through. There are bound to be demonstrations." Yesterday Sir Roy's chief aide, Mr. Julius Greenfield, telephoned his chief a report on his talks with Mr. Macmillan at Chequers.

Sir Roy's United Federal Party is boycotting the London talks on the Protectorate's future. Said Mr. Nkumbula last night: „We want to discuss what to do if the British government gives in to Sir Roy and the talks fall through. There are bound to be demonstrations." Yesterday Sir Roy's chief aide, Mr. Julius Greenfield, telephoned his chief a report on his talks with Mr. Macmillan at Chequers.

Name: Krystyna Speisel

# Data Providing: Paragraphs Extracting

ONLY handwritten section was extracted by using the metadata in the XML file

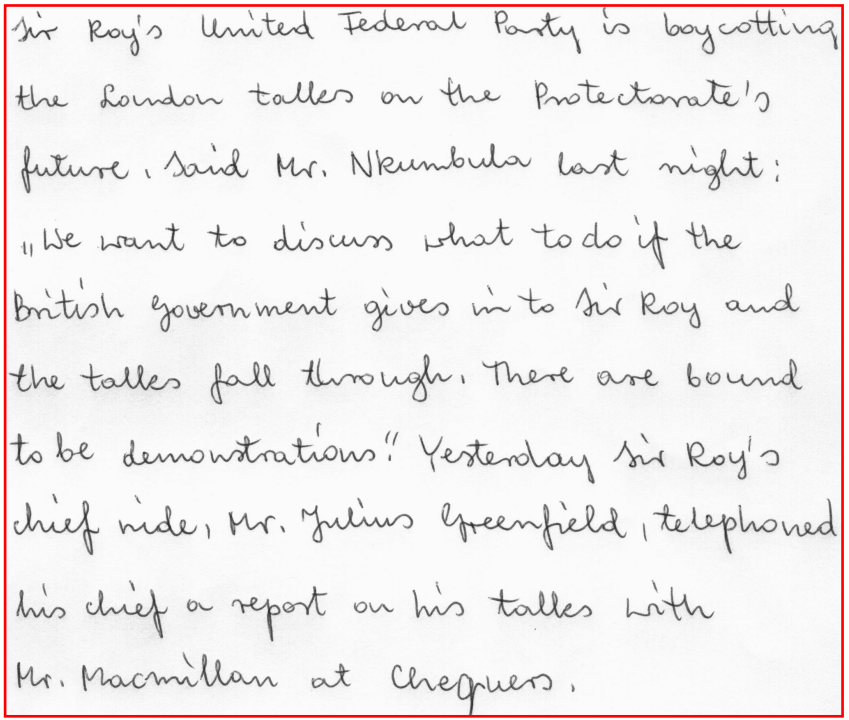


Sentence Database A01-014

Sir Roy's United Federal Party is boycotting the London talks on the Protectorate's future. Said Mr. Nkumbula last night: "We want to discuss what to do if the British Government gives in to Sir Roy and the talks fall through. There are bound to be demonstrations." Yesterday Sir Roy's chief aide, Mr. Julius Greenfield, telephoned his chief a report on his talks with Mr. Macmillan at Chequers.

Sir Roy's United Federal Party is boycotting the London talks on the Protectorate's future, said Mr. Nkumbula last night: „We want to discuss what to do if the British government gives in to Sir Roy and the talks fall through. There are bound to be demonstrations." Yesterday Sir Roy's chief aide, Mr. Julius Greenfield, telephoned his chief a report on his talks with Mr. Macmillan at Chequers.

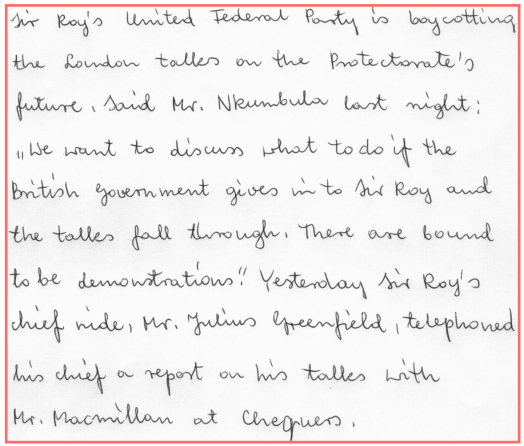
Name: Krystyna Speisel

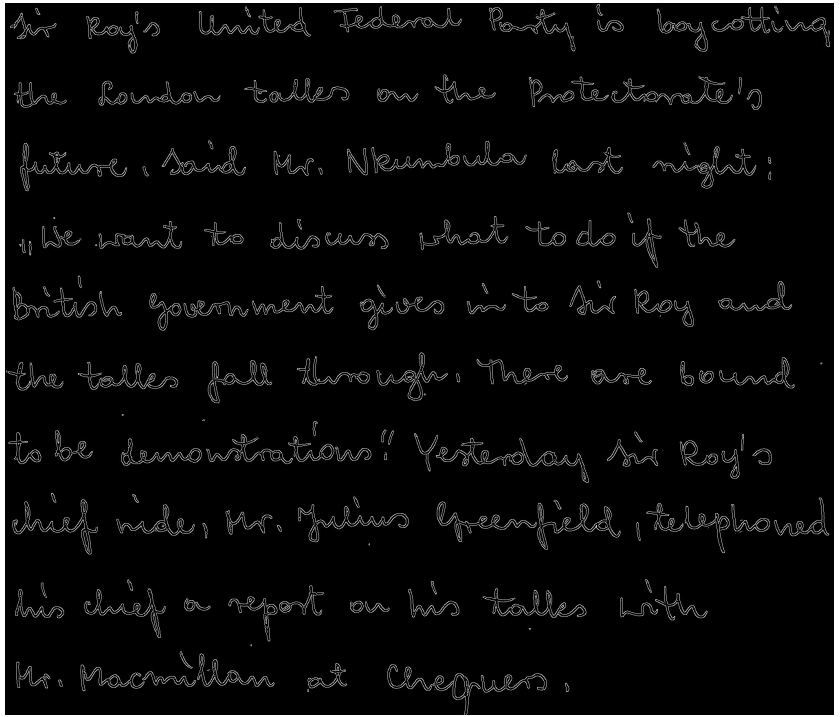


Sir Roy's United Federal Party is boycotting the London talks on the Protectorate's future, said Mr. Nkumbula last night: „We want to discuss what to do if the British government gives in to Sir Roy and the talks fall through. There are bound to be demonstrations." Yesterday Sir Roy's chief aide, Mr. Julius Greenfield, telephoned his chief a report on his talks with Mr. Macmillan at Chequers.

# Data Providing: Image Pre-processing

- Used to remove the noise and variations in background lighting.
- Opencv libraries used (in order):
  - Gaussian blur filter
  - Canny Edge Detector



# Data Providing: Writer Selection

- Aim is to have a suitable number of train and test samples
- The writers have been selected who have at least **9** different samples
- As the result, we have **340** forms with **31** different writers

# Data Providing: Splitting

- We must consider some criteria:
  - Each writer must have **at least one** sample in the test set to be tested.
    (unlike random splitting in machine learning)
  - A good number of training & testing samples
  - Since the task is to identify the writer, cropping the samples into multiple smaller ones (500 x 500) helps to increase the number of the training and testing samples
  - **Overlapping** between smaller samples:
    - In training set → is allowed → **8** different smaller images
    - In testing set → is not allowed at all → **3** different smaller images

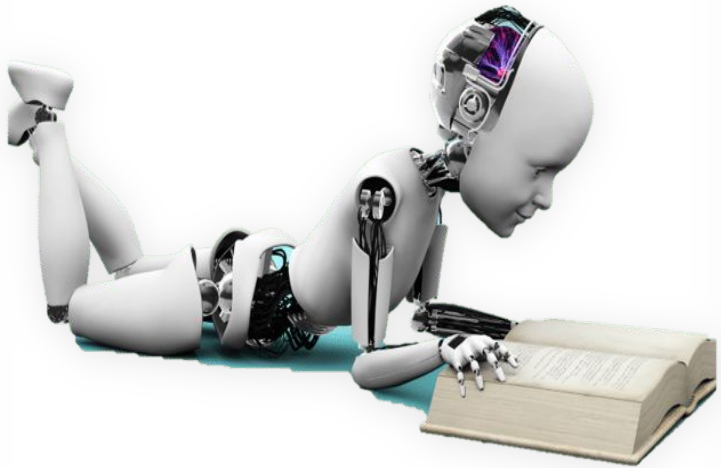
# Data Providing: Splitting

- To prevent having an unbalanced train set, and having an equal number of documents per each identity, we took 5 forms for each identity as the training samples (before cropping)

- Dataset Splitting Procedure:
  - Training Set → **5** out of **9** samples per each writer
  - Test Set → **4** out of **9** samples per each writer

# Data Providing: Splitting

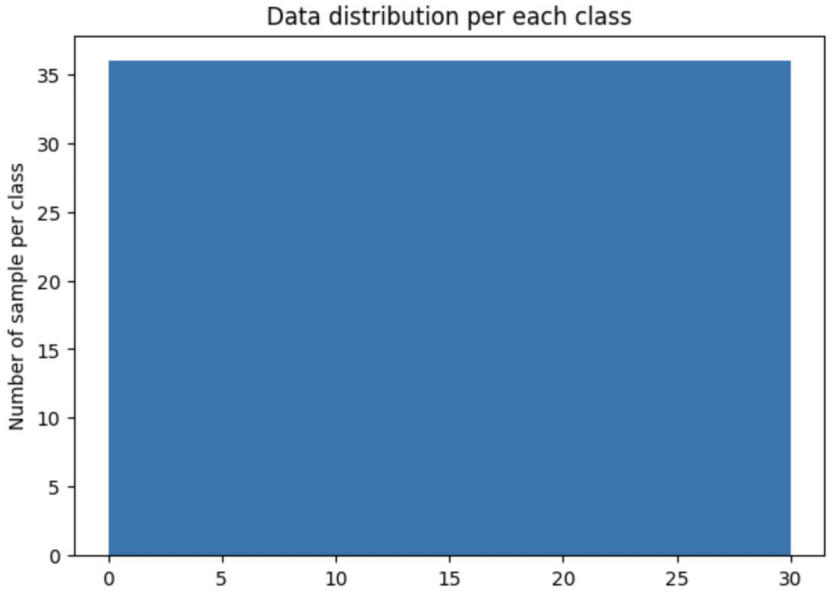
- In overall, in our dataset there are:
  - **31** identities, with **9** documents for each
  - **5** documents per each identity in the training set
  - **4** documents per each identity in the test set
  - Each of the training set images cropped into **8** sub-images
  - Each of the test set images cropped into **3** sub-images
  - **31** * **5** * **8** = **1240** training samples
  - **31** * **4** * **3** = **372** test samples



# 03 Training Phase

# Training Phase: Image Generation

- We have 3 different models w.r.t. 3 different approaches using **Keras Image Generator**:
  - Without any augmentation
  - With only rotation
  - With more augmentation such as Rotation, shift, brightness, zooming, rescale

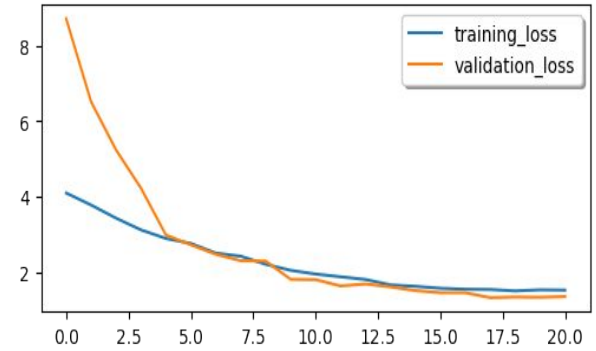


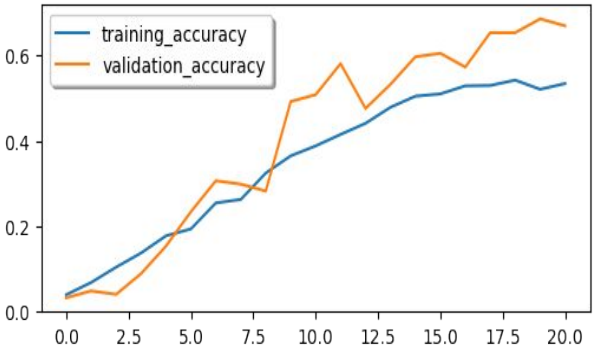
# Training Phase: Implementation Part

- The network is constructed by referencing AlexNet with some configurations:
  - 4 layers instead of 5 convolutional layers (complexity)
  - **Optimizer:** Stochastic Gradient Descent (SGD)
  - **Loss Function:** categorical_crossentropy
  - **Number of Epochs: 150** (But we used " early stopping ")
  - **Evaluation Metric used:** accuracy
- Also, callback functions of **ModelCheckpoint**, **EarlyStopping**, and **ReduceLROnPlateau**
- 124 samples from the train set were used as a validation set

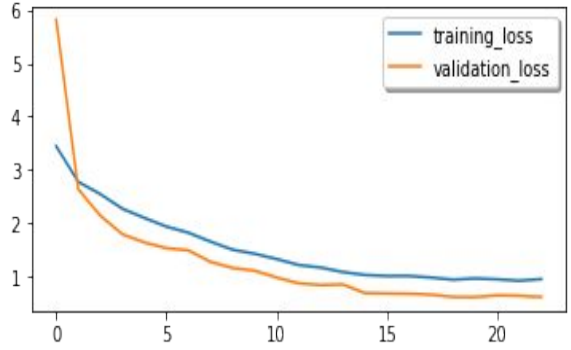
# Training Phase: Accuracy and Loss Visualization

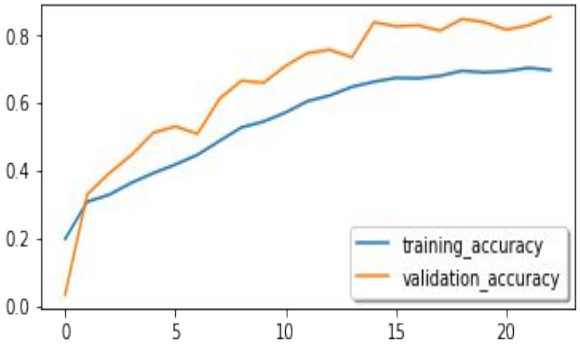
*Only rotation*




*more augmentation*



## more augmentation

```
[ ] test_on_a_image('/content/test_set/0/a01-007u-0.png')

    author id: 0 ----------> score: 99.947%

[ ] test_on_a_image('/content/test_set/344/g06-011m-0.png')

    author id: 346 ----------> score: 78.269%
    author id: 153 ----------> score: 11.905000000000001%
    author id: 634 ----------> score: 7.295999999999999%

[ ] test_on_a_image('/content/test_set/155/c03-000f-1.png')

    author id: 155 ----------> score: 86.467%
    author id: 0 ----------> score: 6.039%

    test_on_a_image('/content/test_set/336/g06-011e-2.png')

    author id: 0 ----------> score: 47.239%
    author id: 346 ----------> score: 21.458%
    author id: 336 ----------> score: 20.237%
```

- higher score of similarity
- More mistakes :(

## Only rotation

```
[ ] test_on_a_image('/content/test_set/0/a01-007u-0.png')

    author id: 0 ----------> score: 79.88199999999999%
    author id: 346 ----------> score: 9.014999999999999%
    author id: 348 ----------> score: 4.9%

[ ] test_on_a_image('/content/test_set/344/g06-011m-0.png')

    author id: 153 ----------> score: 63.247%
    author id: 344 ----------> score: 12.092%
    author id: 634 ----------> score: 12.011%
    author id: 346 ----------> score: 4.833%

[ ] test_on_a_image('/content/test_set/155/c03-000f-1.png')

    author id: 155 ----------> score: 60.33%
    author id: 150 ----------> score: 7.619%
    author id: 384 ----------> score: 6.5009999999999994%
    author id: 342 ----------> score: 4.027%

    test_on_a_image('/content/test_set/336/g06-011e-2.png')

    author id: 336 ----------> score: 37.208000000000006%
    author id: 333 ----------> score: 10.807%
    author id: 634 ----------> score: 9.17%
    author id: 346 ----------> score: 8.997%
    author id: 347 ----------> score: 6.76%
    author id: 342 ----------> score: 5.318%
    author id: 153 ----------> score: 5.108%
    author id: 635 ----------> score: 4.531000000000001%
```

- the score of similarity is not quite high.
- The true identity is in the returned list

First interpretation: Only rotation performs better BUT..

# Training Phase: Test Set Accuracy

- Predicting the identity of the given image using a function based on Machine Learning on Testing set
- Accuracy over the test set in the case of:
  - solo rotation model → 44.89%
  - With more augmentation → 63.17%
- More augmentation model has a higher accuracy over all images, means that performs better than solo rotation one

```
[ ] THRESHOLD = 0.04

def apply_threshold(dic, threshold=THRESHOLD):

    for (key, value) in dic:
        if value >= threshold:
            print(f'author id: {key} ----------> score: {str(value*100)}%')
```

```
[ ] import numpy as np
from keras.preprocessing.image import img_to_array, load_img

classes = val_set.class_indices

def test_on_a_image(path):
  test_image = img_to_array(load_img(path, target_size = (IMAGE_SIZE, IMAGE_SIZE)))
  result = model.predict(np.expand_dims(test_image, axis = 0))
  dic = {}
  for (key, value) in classes.items():
      dic[key] = float(format(result[0][value],'.5f'))

  return apply_threshold(sorted(dic.items(), key=lambda x: x[1], reverse=True)[:10])
```

| Model | Loss | Accuracy |
|---|---|---|
| Solo Rotation | 1.8915 | 44.89% |
| More Augmentation | 1.3178 | 63.17% |

# 04 Evaluation

# Evaluation:

- Remove last layer(classification) to access the feature layer
- The goal is to create a similarity matrix (372 x 372) which:
  - Has **31 identities**, each of them has 12 templates(31 x 12 = 372)
  - Each row (we have 372 rows) is a **31 recognition operation**
  - In the column, we have **372 templates**
  - Each row contains **1 genuine claim**
  - Each row contains **N_identities - 1 imposters (30 operations)**
  - For each operation we considered a **group of templates** with the same identity with maximum similarity. Then this similarity was compared with the given threshold. This is known as **multiple-template verification** in the literature.

# Evaluation:

- To compare feature vectors with each other we defined **two** measures:
  - **Cosine Similarity** → had a better result in general
  - **Correlation**

- As we had 372 feature vector, the matrix of the correlation and similarity is **372x372**

- The matrix gives us the ability to perform an all-against-all biometric evaluation task

```
[ ] cosine_similarity       = cosine_similarity_matrix(feature_test_set)
    correlation_similarity  = correlation_similarity_matrix(feature_test_set)

[ ] print(cosine_similarity.shape)
    print(correlation_similarity.shape)

    (372, 372)
    (372, 372)
```

Similarity Matrix

```
# each tamplate of the identity in the matrix has a column index 372 * 372
'''
    id_0 template_0   0 1 2 3 4 5 6 7 8 9 10 11 __ 12 13 14 15 16 17 18 19 20 21 22 23 __ ... 370 371
    id_0 template_1   0 1 2 3 4 5 6 7 8 9 10 11 __ 12 13 14 15 16 17 18 19 20 21 22 23 __ ... 370 371
    id_0 template_3   0 1 2 3 4 5 6 7 8 9 10 11 __ 12 13 14 15 16 17 18 19 20 21 22 23 __ ... 370 371
            ...
    id_0 template_11  0 1 2 3 4 5 6 7 8 9 10 11 __ 12 13 14 15 16 17 18 19 20 21 22 23 __ ... 370 371
    id_1 template_0   0 1 2 3 4 5 6 7 8 9 10 11 __ 12 13 14 15 16 17 18 19 20 21 22 23 __ ... 370 371
            ...
    id_30 template_11 0 1 2 3 4 5 6 7 8 9 10 11 __ 12 13 14 15 16 17 18 19 20 21 22 23 __ ... 370 371
'''
```

```python
for probe in range(0, similarity_matrix.shape[0]): # for each probe of the matrix ===> probe
    for identity in range(N_IDENTITY): # loop over the columns based on the identities

        identity_templates_indexes = np.arange(identity * N_TEMPLATES, (identity + 1) * N_TEMPLATES)
        identity_templates_indexes_excluded = identity_templates_indexes[np.where(identity_templates_indexes!=probe)]
        identity_templates_scores  = similarity_matrix[probe, identity_templates_indexes_excluded]

        if (identity_templates_scores.max() > threshold): # possible genuine acceptance
            if(identity == probe//N_TEMPLATES): # check to see whether the identity id is the same probe identity
                confusion_matrix[0, 0] +=1      # Genuine Match or Genuine Acceptance (GM, GA)
            else: confusion_matrix[1, 0] +=1.   # False Match or False Acceptance (FM, FA)
        else: # possible imposters
            if(identity == probe//N_TEMPLATES): # check to see whether the identity id is the same probe identity
                confusion_matrix[0, 1] += 1     # False Non-Match or False Rejection (FNM, FR)
            else: confusion_matrix[1, 1] += 1   # Genuine Non-Match or Genuine Rejection(GNM, GR)

return confusion_matrix
```

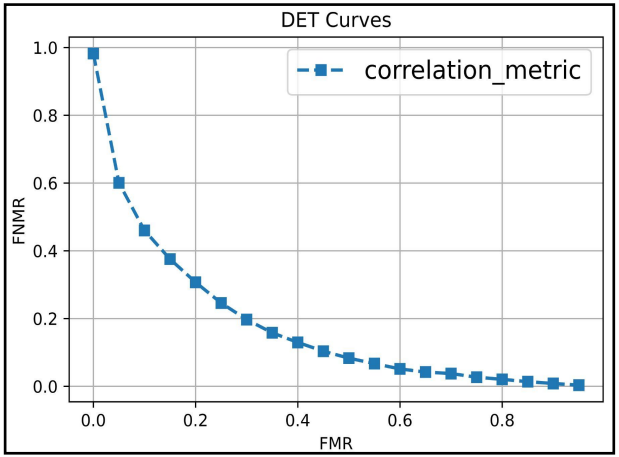
# Evaluation:

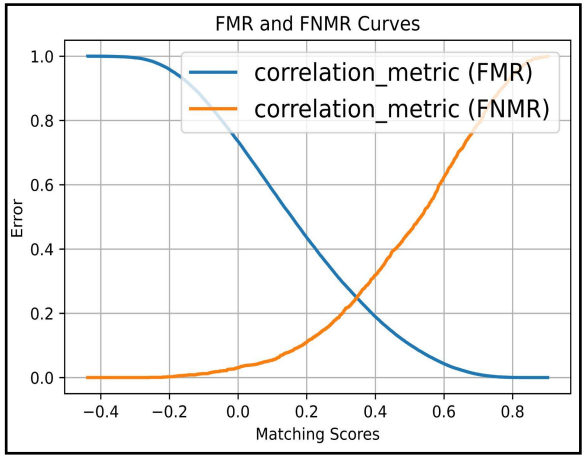
- We need to identify all the thresholds to perform a full evaluation of the biometric
- **pyeer** library:
  - is using two different scores as the input:
    - One corresponds to imposter scores
    - The other is the genuine scores

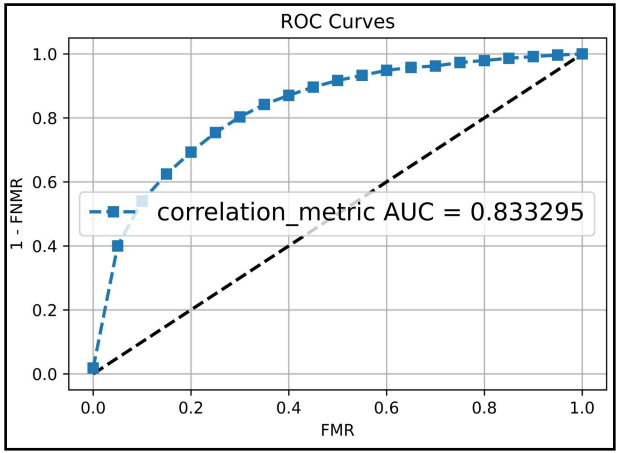
```
def separated_imposters_genuine_scores(similarity_matrix):
    genuine_scores = []
    imposter_scores = []
    for probe in range(0, similarity_matrix.shape[0]): # for each probe of the matrix ===> probe
        for identity in range(N_IDENTITY): # loop over the columns based on the identities

            identity_templates_indexes = np.arange(identity * N_TEMPLATES, (identity + 1) * N_TEMPLATES)
            identity_templates_indexes_excluded = identity_templates_indexes[np.where(identity_templates_indexes!=probe)]
            identity_templates_scores  = similarity_matrix[probe, identity_templates_indexes_excluded]

            if(identity == probe//N_TEMPLATES):
                genuine_scores.append(identity_templates_scores)
            else:
                imposter_scores.append(identity_templates_scores)

    return np.array(genuine_scores), np.array(imposter_scores)
```

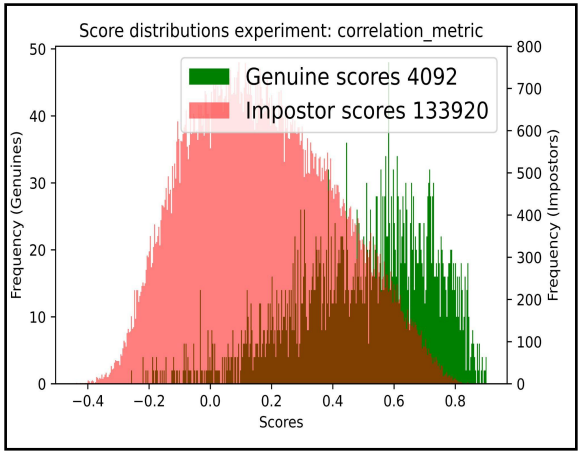
**Model with no augmentation:**

- Results are not much satisfactory
- According to the distribution figure, the overlap part is where we have wrong scores



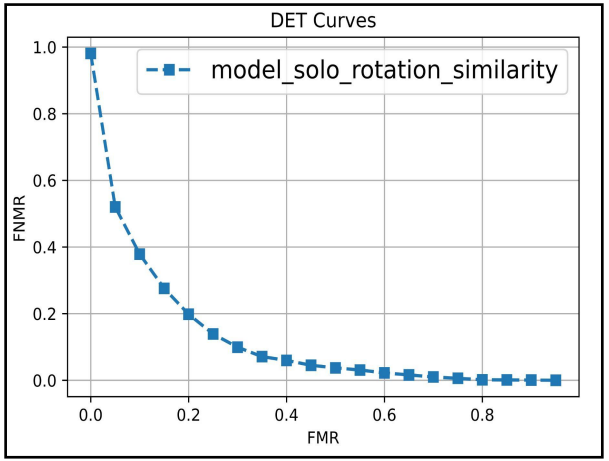



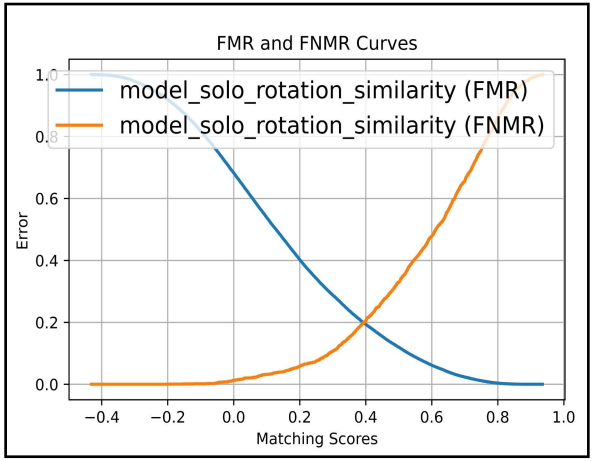



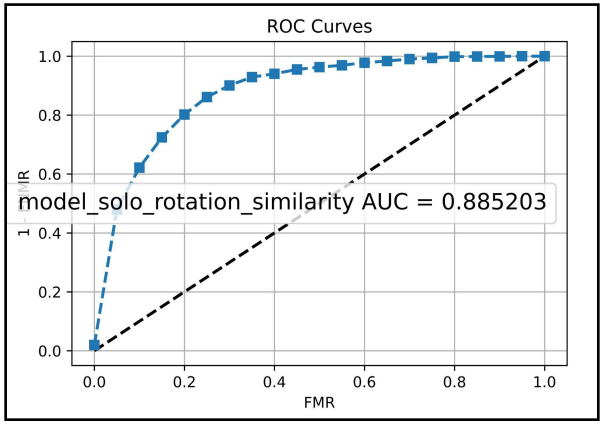



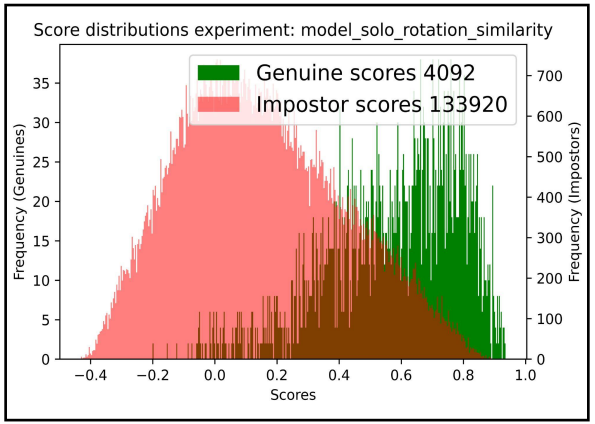
**Model with solo rotation:**

- Better result w.r.t. the previous one both in cosine and correlation measures
- Cosine similarity had a better result in this case
- Adding more augmentation options could be effective to increase the performance
- the EER in the right picture is equal to 0.2 which is better than the previous model
- AUC: 0.83 → 0.88



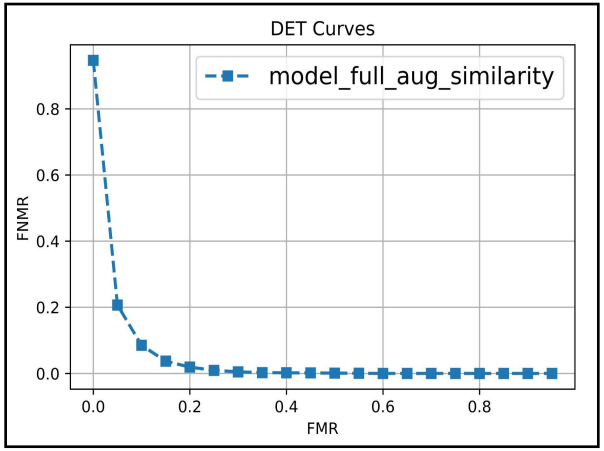



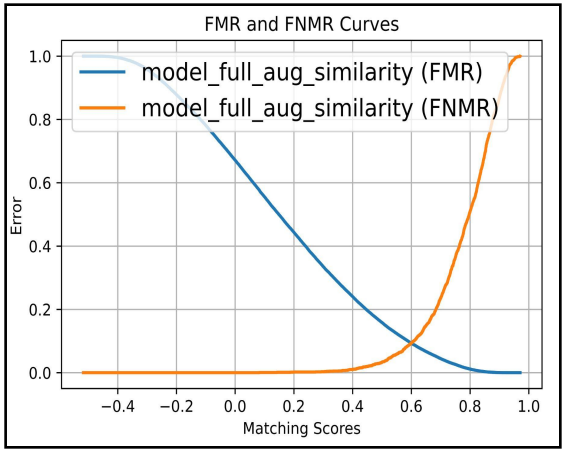



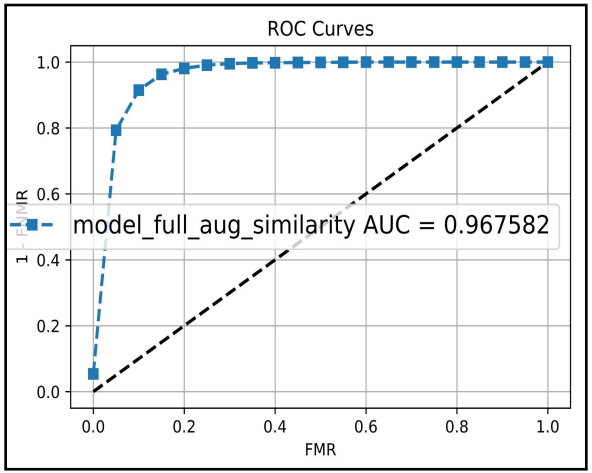



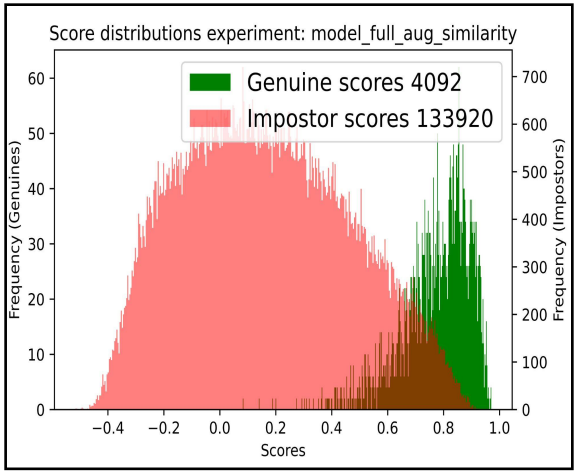
**Model with more augmentation:**

- The best model based on the evaluation result
- The EER also dropped drastically to a lower value (almost 0.1)
- The AUC is now equal to almost 0.97 which is quite impressive due to the nature of the task
- The overlap part also reduced dramatically compared to the previous ones



# Thanks for your attention!

## Any Questions?

teymoori.1947458@studenti.uniroma1.it
shabani.1966731@studenti.uniroma1.it

June 2022