# CNN-Based Rotation Correction

Advanced Machine Learning
Università degli Studi di Milano-Bicocca

Matteo Breganni 869549

# Introduction

## **Quarter-Turn** Rotations

- Hand-crafted CNN
- Feature extraction
- Fine-tuning

## **Full-Range** Rotations

- Applying the best method of the first part

# Dataset Preparation

- **Dataset** created by manually sampling from:
  - Landscape Pictures [1] --> 110 images
  - Unpaired Day and Night cityview images [2] --> 190 images

- 60-20-20 Train/Val/Test split
- Each image is then **divided** into one or multiple squares
- Dataset becomes only square images, with **different resolutions**
- 359 train, 166 val, 117 test.
- Random rotations are applied while training, through a data loader

Figure 1: Image 1

Figure 2: Split Image 1

Figure 3: Image 2

Figure 4: Split Image 2

[1] https://www.kaggle.com/datasets/arnaud58/landscape-pictures
[2] https://www.kaggle.com/datasets/heonh0/daynight-cityview

# Hand-Crafted CNN Approach

# Hand-Crafted CNN Approach

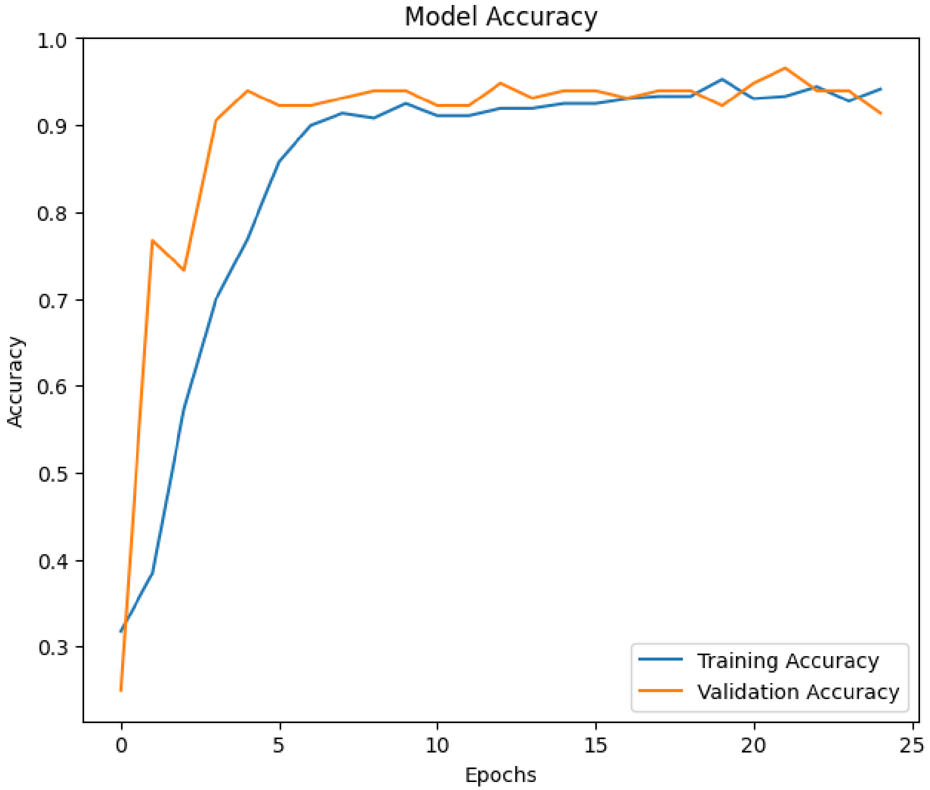
- Creating and training a **CNN from scratch**

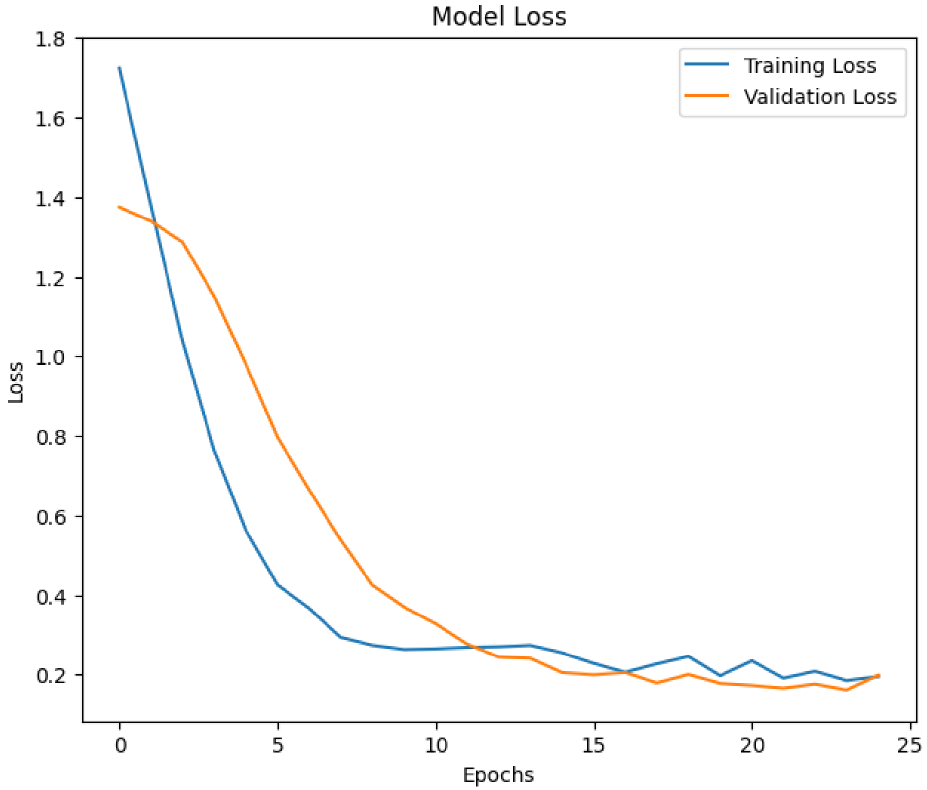
| Layer Type | Output Shape | Param # |
|---|---|---|
| Input | (224, 224, 3) | / |
| Conv, 16, 3x3 | (111, 111, 16) | 448 |
| Batch Normalization | (111, 111, 16) | 64 |
| Conv, 32, 3x3 | (55, 55, 32) | 4’640 |
| Batch Normalization | (55, 55, 32) | 128 |
| Conv, 64, 3x3 | (27, 27, 64) | 18’496 |
| Batch Normalization | (27, 27, 64) | 256 |
| Conv, 128, 3x3 + Dropout 40% | (13, 13, 128) | 73’856 |
| Conv, 128, 3x3 + Dropout 50% | (6, 6, 128) | 147’584 |
| Flatten | (4608) | 0 |
| Dense, 128 + Dropout 50% | (128) | 589,952 |
| Output | (4) | 516 |

- Total trainable parameters: 835,716
- **Data loader** that rotates input images randomly (0°, 90°, 180° 270°)

# Hand-Crafted CNN: Evaluation

- The model **converges** rapidly

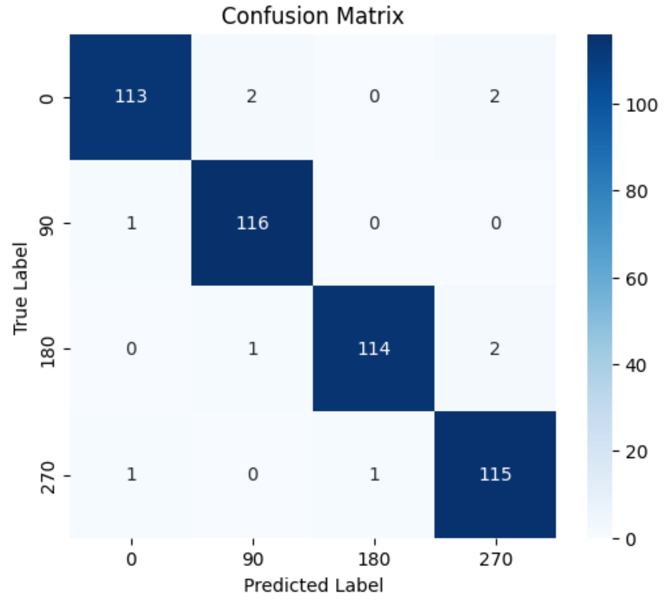


# Hand-Crafted CNN: Evaluation

- **Test set** created by rotating the test images by every possible quarter-rotation
- **Classification report:**

| | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| **0** | 0.98 | 0.97 | 0.97 | 117 |
| **90** | 0.97 | 0.99 | 0.98 | 117 |
| **180** | 0.99 | 0.97 | 0.98 | 117 |
| **270** | 0.97 | 0.98 | 0.97 | 117 |
| **Accuracy** | | | **0.98** | 468 |
| **Macro avg** | 0.98 | 0.98 | 0.98 | 468 |
| **Weighted avg** | 0.98 | 0.98 | 0.98 | 468 |




- **Great performance**: 98% accuracy

- Very **few misclassifications**, we can retrieve and plot them to further analyze the model

# Hand-Crafted CNN: Evaluation

- **Misclassified images:**


- **Repeated mistakes** on the same test images
  - Indicates a **lack of generalization** in the model, due to the small dataset's size

# Features Extraction Approach

# Feature Extraction Approach

- **Feature extraction** from MobileNetV2
- Feature arrays used on a **SVM Classifier**
- Could be a good approach considering the **small dataset's size**

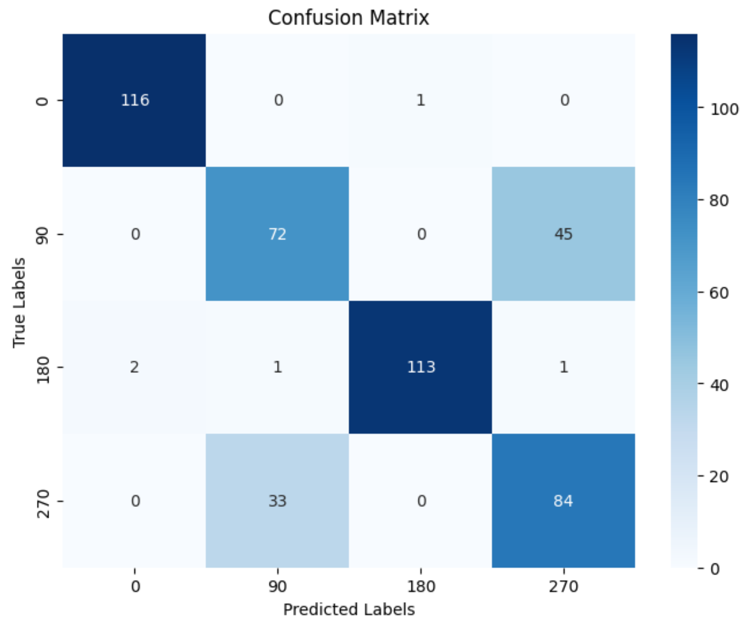
- Images **rotated by every possible rotation** (0°, 90°, 180° 270°) to give it the best chance of success (for all train, val and test images)
  - The function made can be set to fewer unique rotations, by changing a parameter

- SVM fitted and **hyperparameter optimization** done, using the validation set

# Feature Extraction: Evaluation

- Used the best parameters on a new SVM, fitted with both train and validation data, to attempt to increase the performance
  - The performance increase was very **minor**

| Class | SVM Best Parameters | | | SVM with Added Val Data | | |
|---|---|---|---|---|---|---|
| | **Precision** | **Recall** | **F1-Score** | **Precision** | **Recall** | **F1-Score** |
| **0** | 0.98 | 0.98 | 0.98 | 0.98 | 0.99 | 0.99 |
| **1** | 0.64 | 0.60 | 0.62 | 0.68 | 0.62 | 0.65 |
| **2** | 0.98 | 0.97 | 0.98 | 0.99 | 0.97 | 0.98 |
| **3** | 0.63 | 0.68 | 0.65 | 0.65 | 0.72 | 0.68 |
| **Accuracy** | | | **0.81** | | | **0.82** |
| **Macro Avg** | 0.81 | 0.81 | 0.81 | 0.82 | 0.82 | 0.82 |
| **Weighted Avg** | 0.81 | 0.81 | 0.81 | 0.82 | 0.82 | 0.82 |




- **0° and 180°**’s performances are comparable to the previous method
- **90° and 270°** are greatly **confused with each other**
- Most likely it’s due to a **lack of information** about these type of rotations in the extracted features

# Feature Extraction: Evaluation

- Plotting all the **mistakes** made on **0° and 180°** classes:
  - (images not rotated)

Image: 66_1.jpg
True: 2
Pred: 0

Image: 60_2.jpg
True: 0
Pred: 2

Image: 60_2.jpg
True: 2
Pred: 1

Image: 66_2.jpg
True: 2
Pred: 0

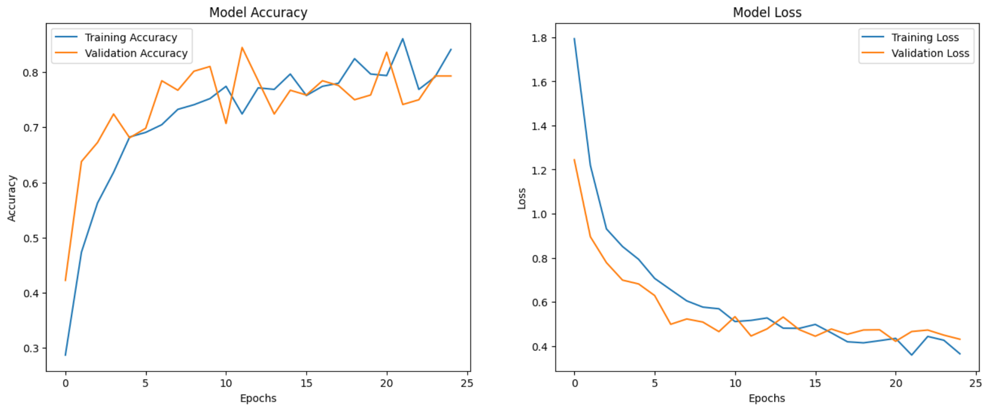
Image: 246_2.jpg
True: 2
Pred: 3

- Two images are misclassified for both rotations
  - **Different** from the images of the previous method
  - **This cannot be a sure indication of possible better generalization**

# Fine-Tuning Approach

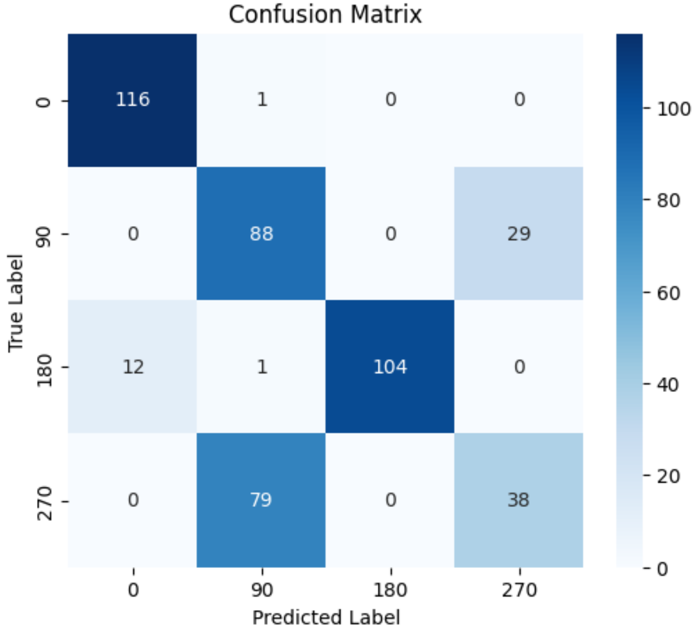
# Fine-Tuned CNN: Evaluation

- **Fine-tuning** of MobileNetV2 pre-trained on ImageNet
  - **Freezing** the base model's weights at the start
  - Adding a **dense layer** (512) with 0.6 dropout and the output layer
  - Using MobileNetV2's preprocessing pipeline to prepare the images




| | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| **0** | 0.91 | 0.99 | 0.95 | 117 |
| **90** | 0.52 | 0.75 | 0.62 | 117 |
| **180** | 1 | 0.89 | 0.94 | 117 |
| **270** | 0.57 | 0.32 | 0.41 | 117 |
| **Accuracy** | | | **0.74** | 468 |
| **Macro avg** | 0.75 | 0.74 | 0.73 | 468 |
| **Weighted avg** | 0.75 | 0.74 | 0.73 | 468 |

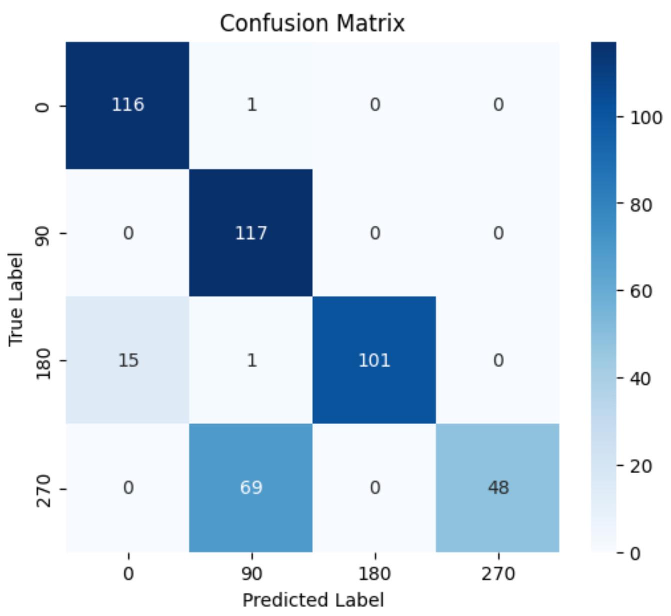
- Seems to be suffering from the **same issue** as the feature extraction method
- The **issue** stands in the **pre-trained network**



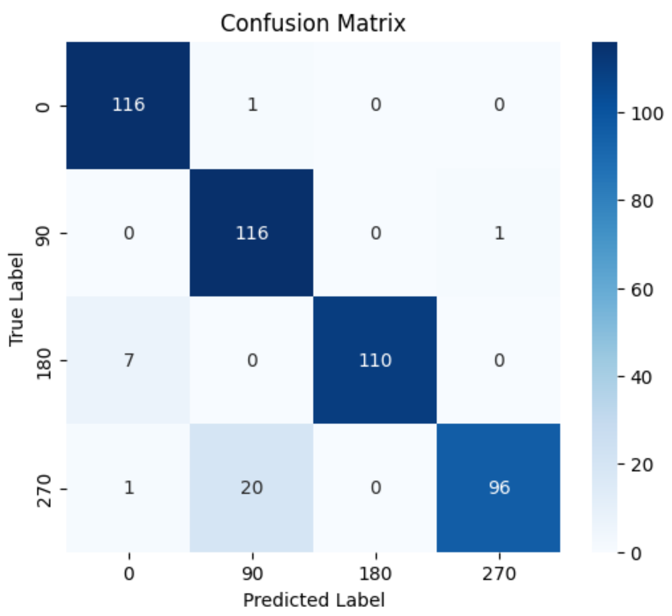
# Further Fine-Tuning (unfrozen)

- **Further fine-tuning** un-freezing the base model's weights, with a lower learning rate

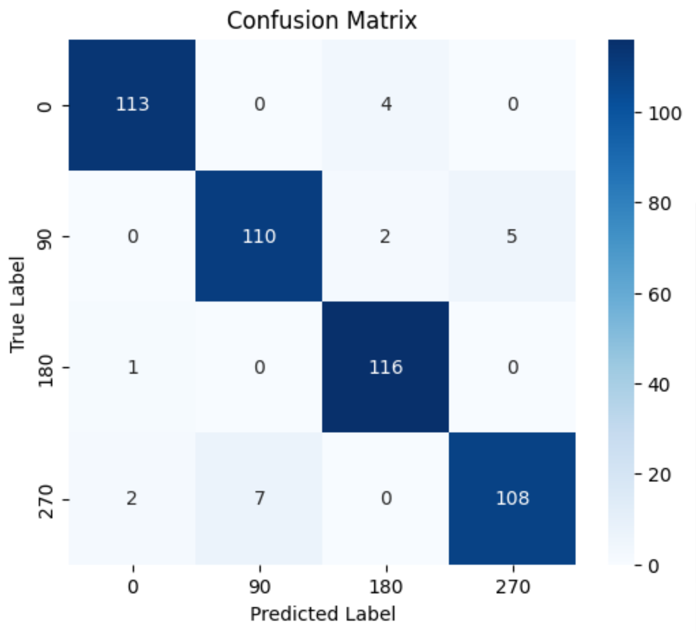
**Model 2** accuracy: 82%




**Model 3** accuracy: 94%




**Model 4** accuracy: 96%




- **Increasingly more training** on the previous model
- The issue on the **classes 90° and 270°** slowly gets «fixed»
  - Not reaching the hand-crafted performance (could be fine-tuned more though)
- The classification task for which MobileNetV2 was trained is **not compatible with this task**, as it does not capture **lateral-rotation information** well

# Full Range Rotations

# Full Range Rotations

- Expanding the previous methods, while applying the best one (**hand crafted CNN**) to this task
- Images can now be **rotated any amount** from 0° to 360°.
- Customs functions were necessary to rotate the images
    - **Black fill** must be deleted, otherwise the model would use it to learn
    - The **biggest square** that fits inside the rotated image is cropped [2]

[2] https://math.stackexchange.com/questions/828878/calculate-dimensions-of-square-inside-a-rotated-square

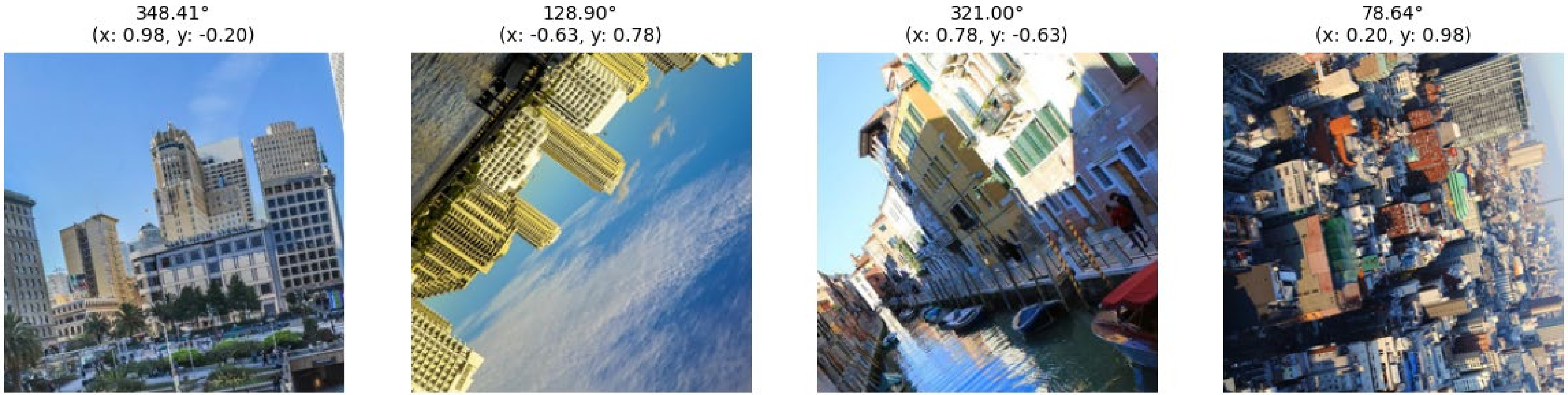
# Full Range Rotations

- The **output** cannot be one-hot encoded anymore, but it also shouldn't be the rotation amount in degrees.
  - If the rotation amount was used, rotations like **1°** and **259°** would be seen as **polar opposites**, not close rotations.

- The rotations were converted to two ***x* and *y* values**:

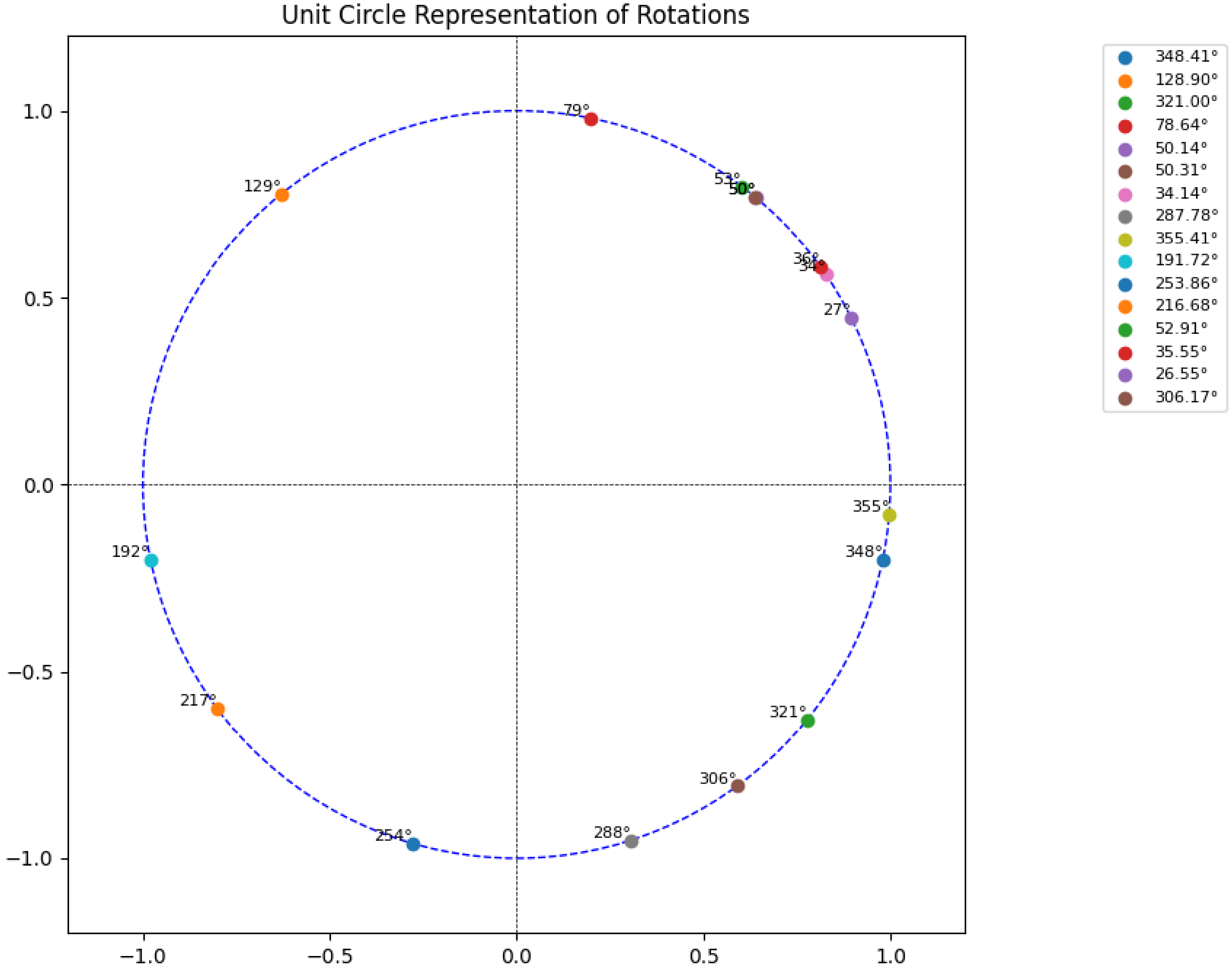
$$x = \cos(\theta) \qquad y = \sin(\theta)$$

  - where θ is the rotation angle in radians.



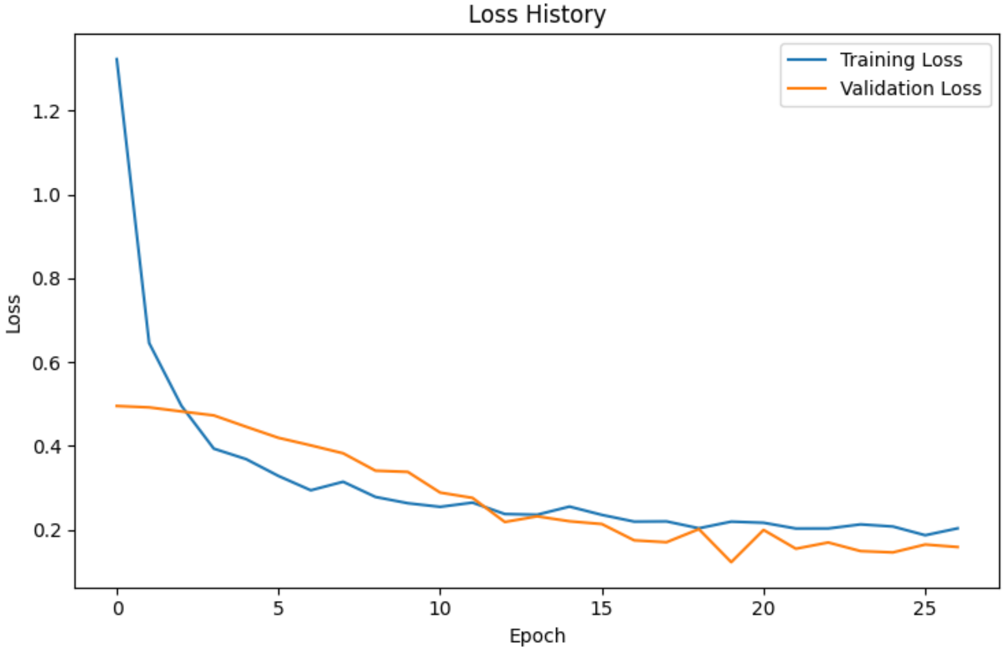
# Full Range Rotations

- Rotations like 0° and 259° are **next to each other**
- Rotations like 0° and 180° are **opposite of each other**



# Full Range Rotations: Evaluation

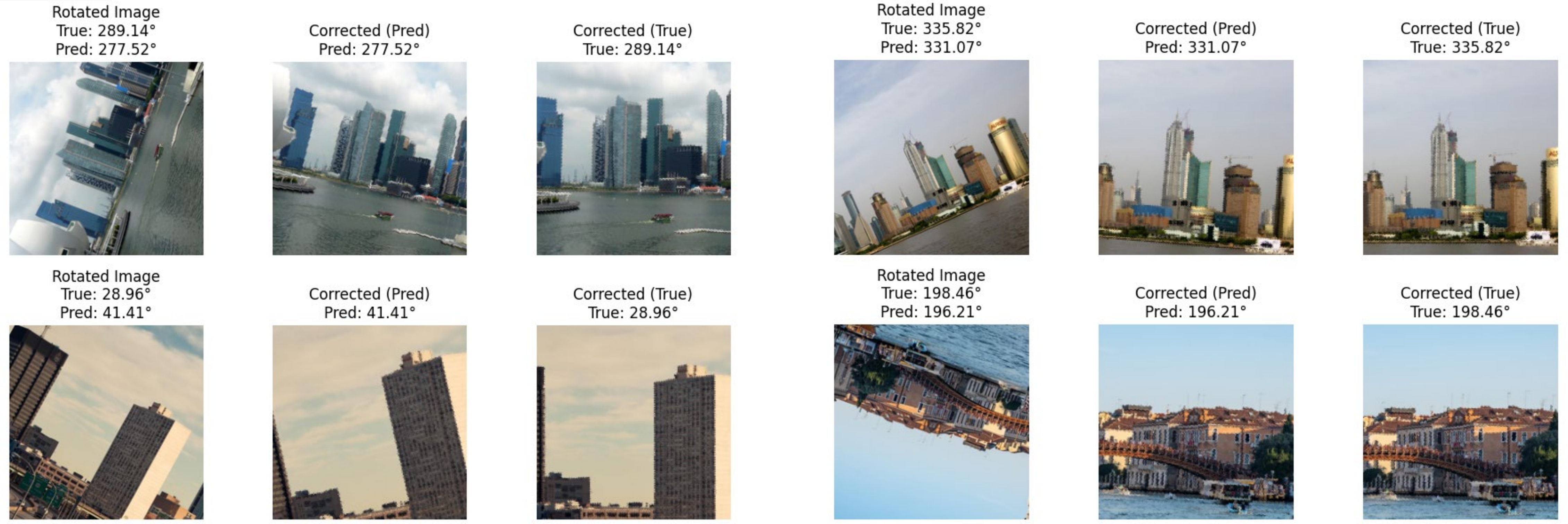
- The **same hand-made model architecture** was used
  - With slightly higher learning rate




- Test Loss (mse): 0.15
- **Average error in degrees**: 27.08°
  - Fairly **acceptable** considering that the images in the dataset are **not necessarely perfectly level**. A more curated dataset could improve the performance.
  - The model's **architecture could be improved**, for example with a bigger dense layer before the output, 128 neurons might be too little

# Full Range Rotations: Evaluation

- To better evaluate the results, let's **correct the rotations**:




- **Small rotation errors** are very noticeable to the human eye

# Thanks for your attention

# Thanks for your attention

## CNN-Based Rotation Correction

Advanced Machine Learning
Università degli Studi di Milano-Bicocca

Matteo Breganni 869549