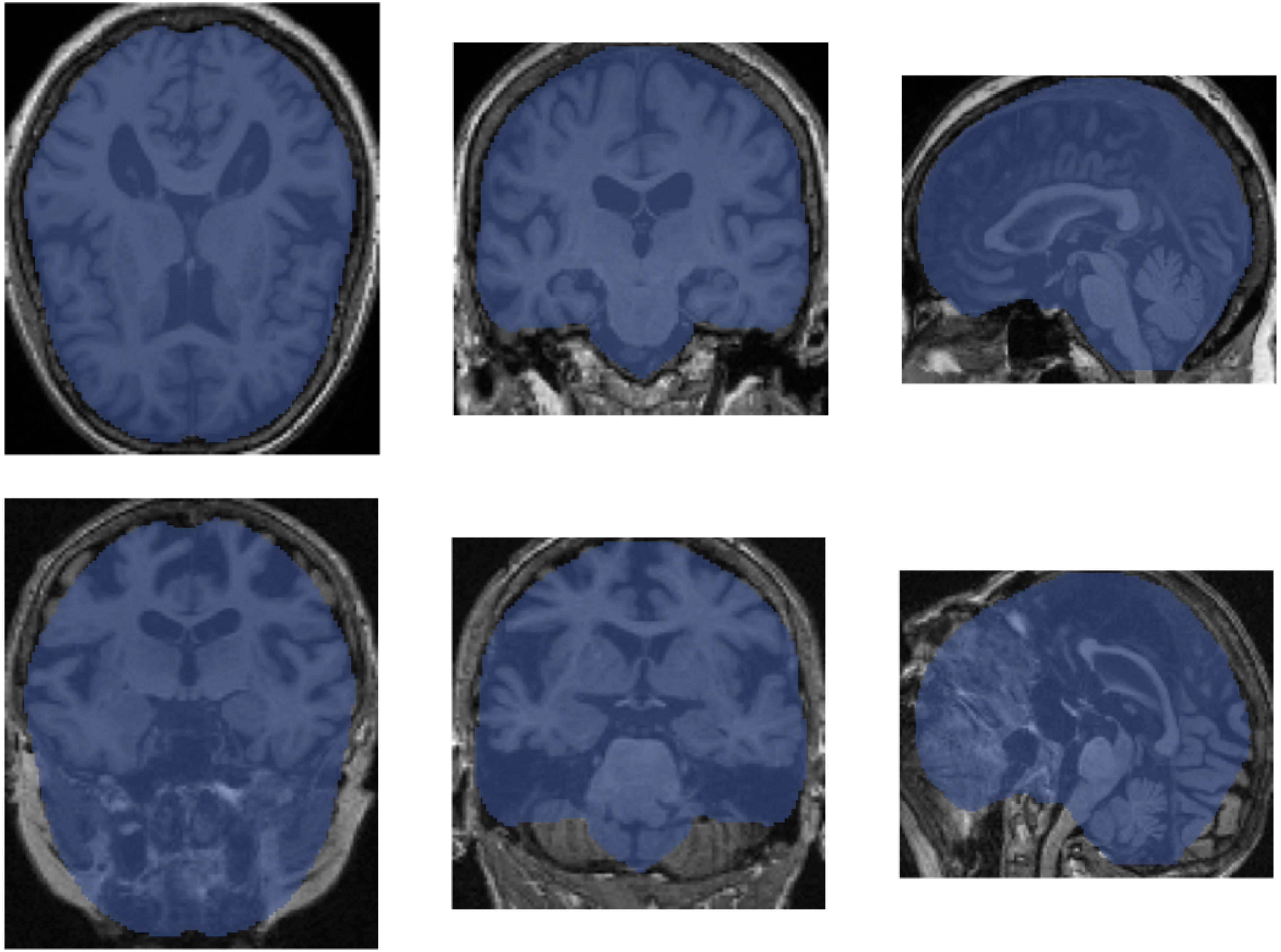


ADNI dataset

Alzheimer desease (AD) detection based on MR images through Random Forest and CNNs approaches

Data Analysis
The problem

- **ADNI (Alzheimer's Disease Neuroimaging Initiative) dataset:**
  - **3D** PET and MR images (labelled).
  - **AD** (107 MR images) and **HC** (346 RM images).
  - **Unbalanced** classes.
  - Common shape: (105 x 127 x 105)

- **Classification tasks:**
  - **Random Forest** classifier trained on **features extracted** by **PyRadiomics** and re-trained on the **5-most informative features** selected through **RFE** (*Recursive Feature Elimination*)
  - Classification based on a **Convolutional Neural Network**

Splitting data

- Delate useless columns of the AD and HC .csv and combine them

- Split the overall data into:
  60% train
  20% validation and
  20% test set

In a **fixed way** in order to select the same images for the different tasks execution (*Classification through RF* and *Classification through CNN*).

Random Forest Classifier
Put a mask and extract features

- Load the mask and set the **coordinates** in oder to **automatically center** the image and overlap it on the focused **brain area for each slice**

- Make a **list** of the files for healthy subjects and ill ones

- **Extract** images with the default setting and **apply mask:**
  ext = featureextractor.RadiomicsFeatureExtractor()
  ext.execute(healthFile, maskFile)

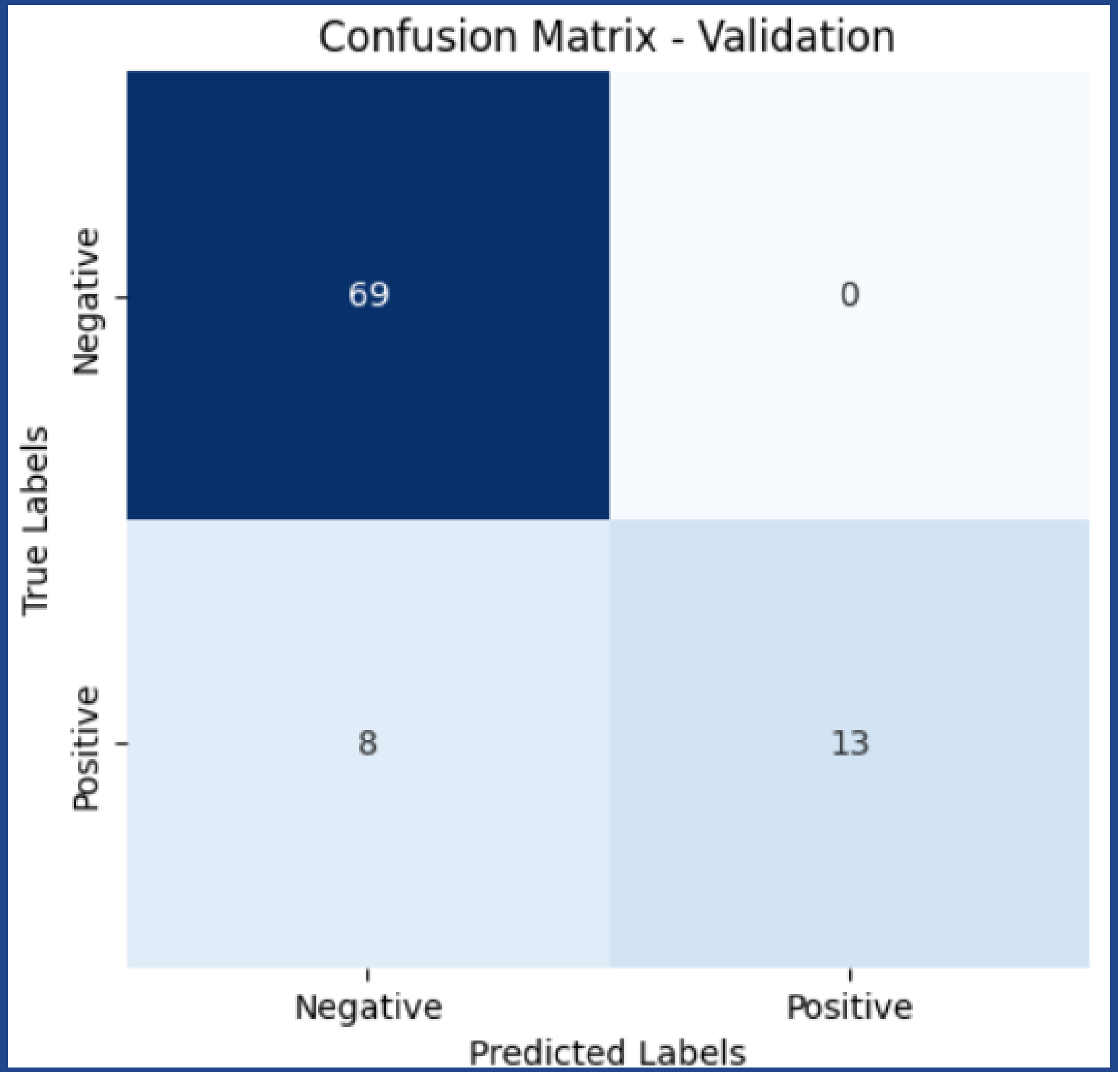
- Save the **features names** and the **extracted features (107x346/class)** lists as **arrays** and combine them

Random Forest

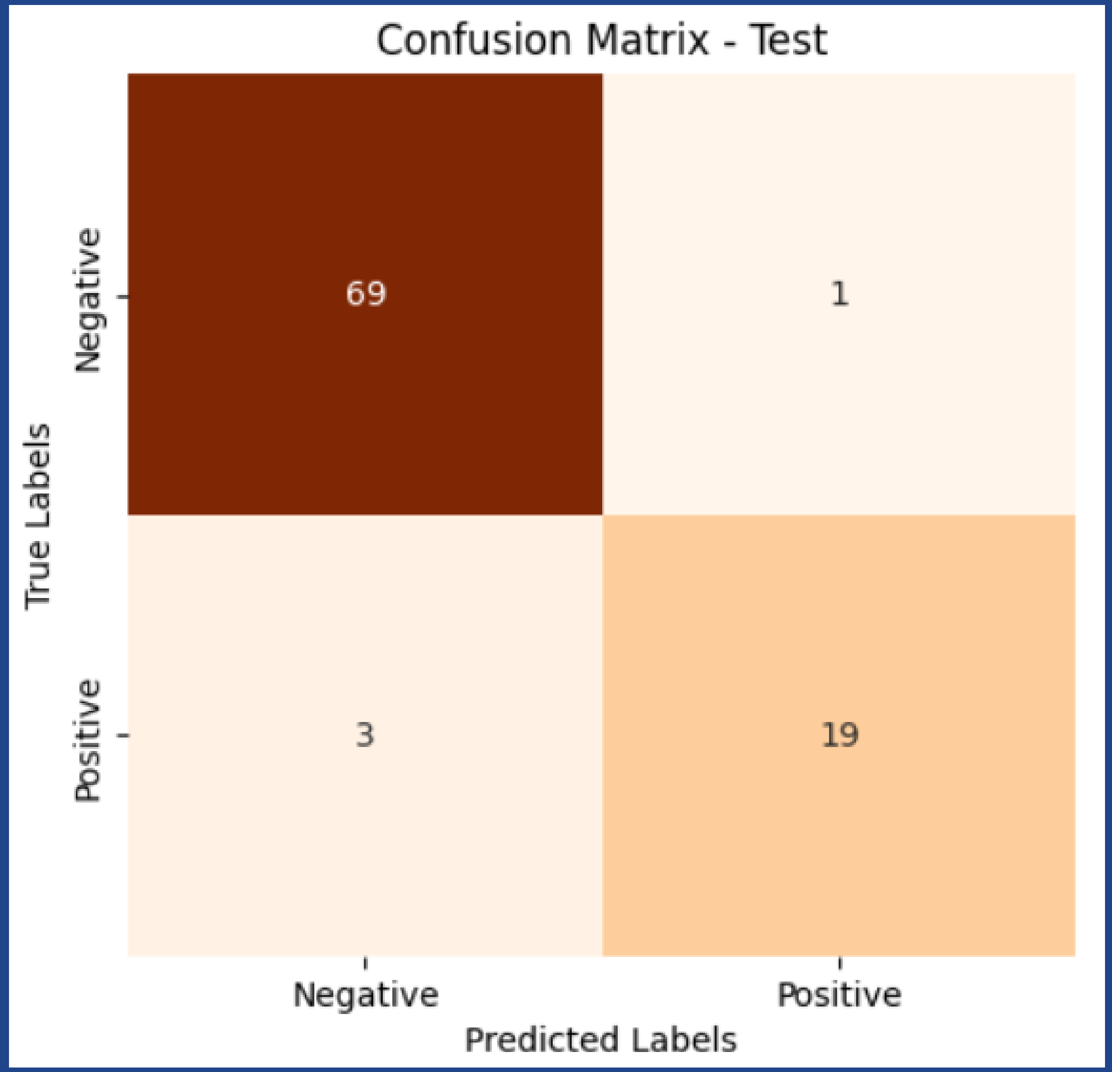
- Build **a RF Classifier** with:
  - **n_estimators** = 100 . Number of **decision trees** to include.
  - **max_depth** = 5 . This parameter **controls the maximum depth** of each decison tree. Each tree has at most 5 levels.
  - **random_state** = 42 . It sets the seed for the random number generator used by the RF. Ensures that the **same sequence of random numbers** --> reproducible results

Results

- 107 extracted features from **unbalanced classes** (AD and HC)
- More attention on: **Recall** $\frac{TP}{TP+FN}$, **Precision** $\frac{TP}{TP+FP}$, **F1 score** $2\frac{Precision \cdot Recall}{Precision+Recall}$



Precision: **1**
Recall: **0.62**
F1 score: **0.76**
Acc: **0.91**



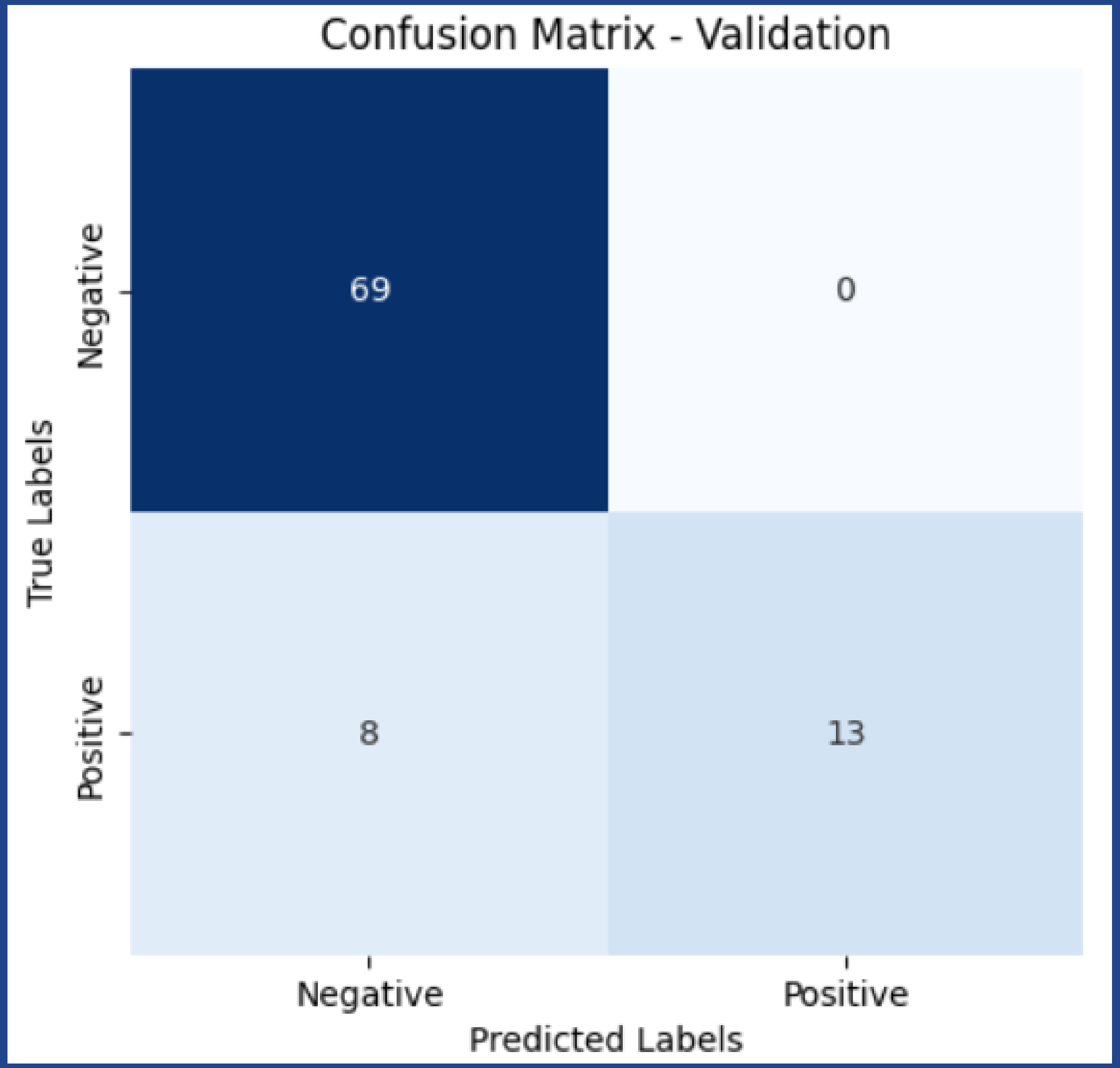
Precision: **0.95**
Recall: **0.86**
F1 score: **0.90**
Acc: **0.95**

5-most informative features

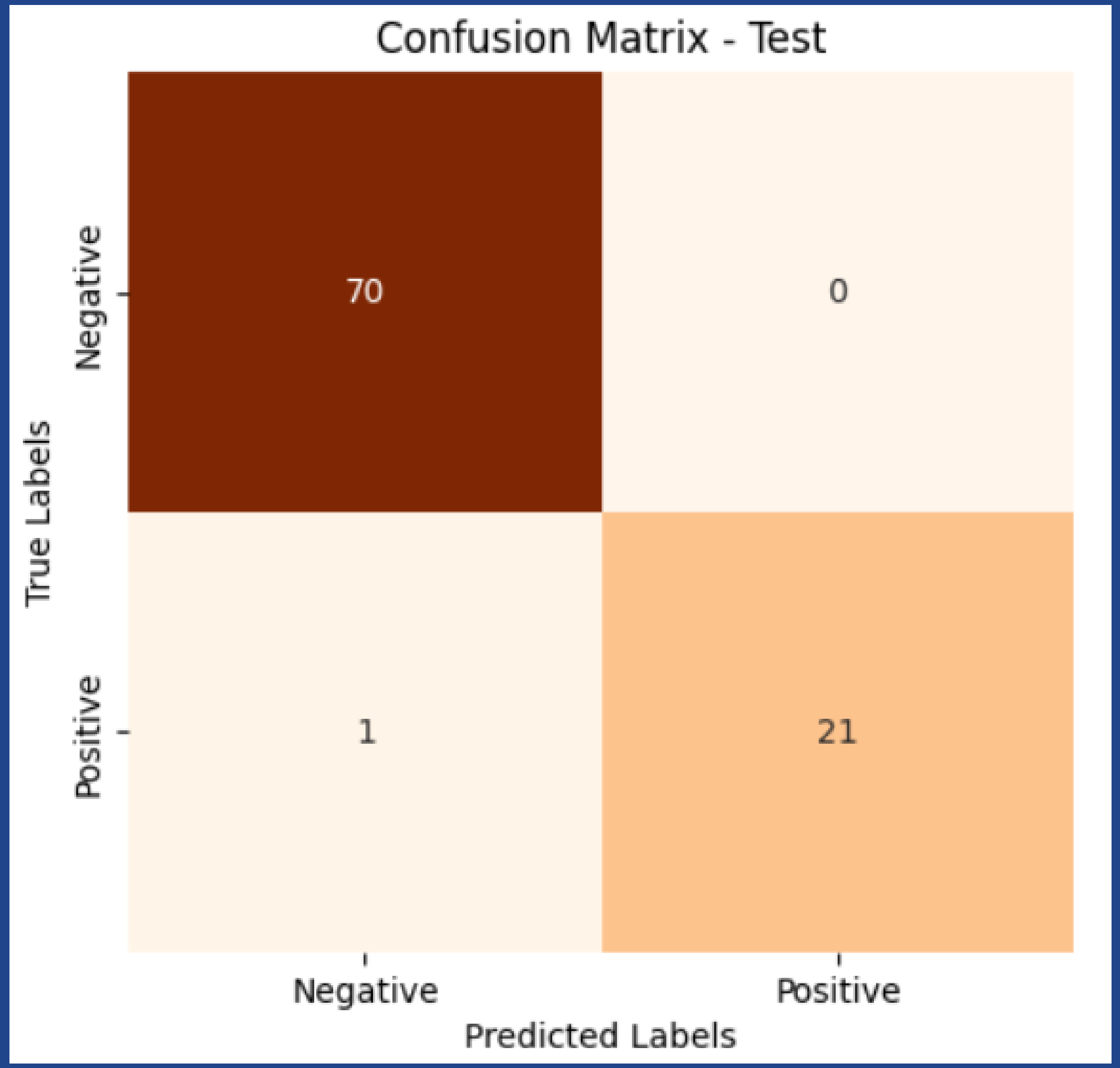
- **RFE *(Recursive Feature Elimination)***
  - Build RF model using **all available features**
  - **Rank the importance** of each feature based on the model performance, and then remove the least important feature(s) from the model.
  - The importance of each feature **is then calculated based on the weights or coefficients assigned to the features** by the model (with mean absolute error, R-squared, or F1-score)
  - Can help **to reduce overfitting, improve model accuracy,** and **reduce computational complexity**
- **Extracted features (5):** "firstorde Minimum", "firstorder Skewness", "glcm Imc2", "gldm SmallDependenceLowGrayLevelEmphasis", "ngtdm Coarseness"

Random Forest re-trained results

- Performance metrics **increase on the test** set
- It can be explained due to the presence of the most informative features selected by RFE. Using less features = more generalized algorithm



Precision: **1**
Recall: **0.62**
F1 score: **0.76**
Acc: **0.91**

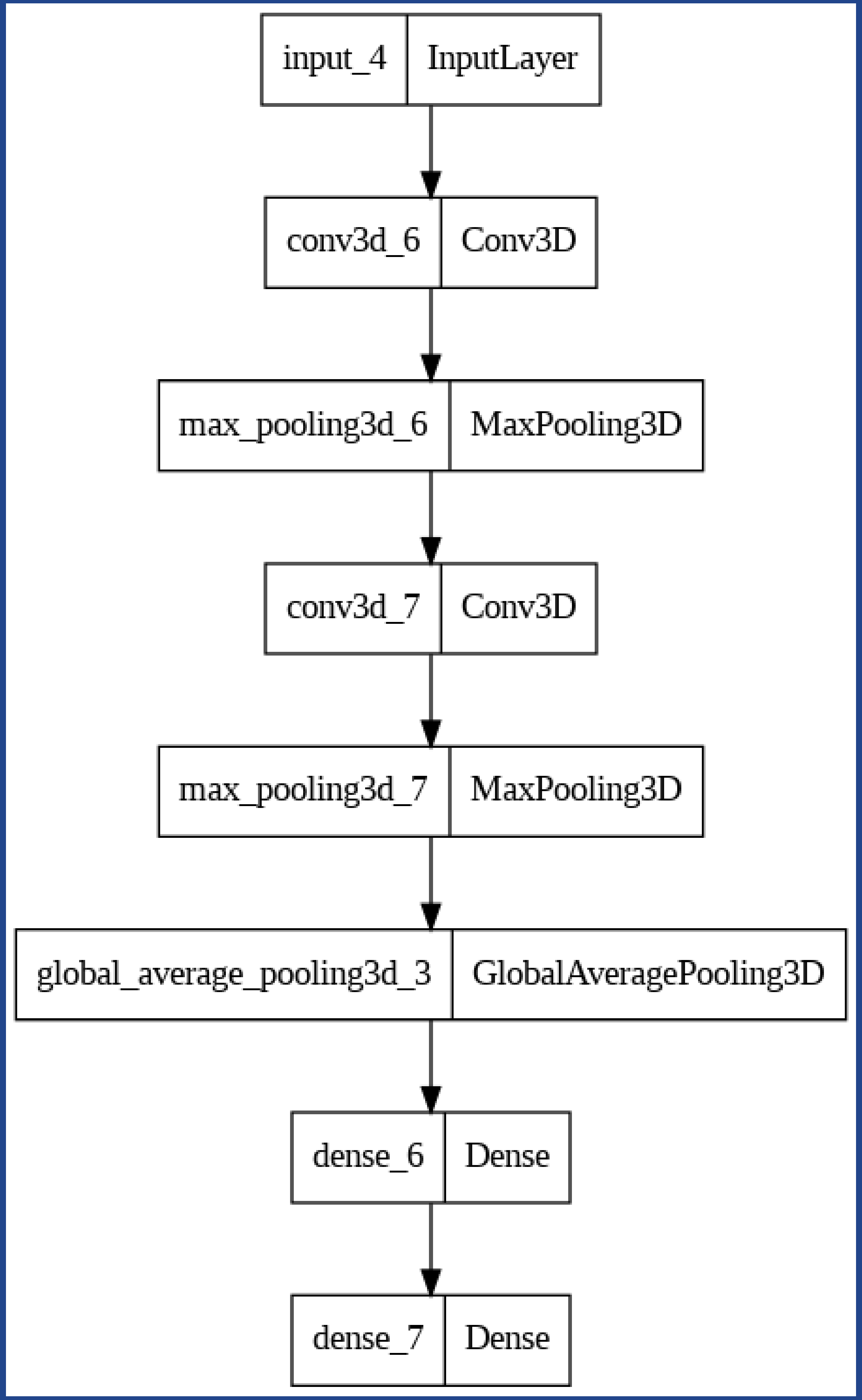


Precision: **1**
Recall: **0.95**
F1 score: **0.98**
Acc: **0.99**

3-Dimensional CNN

- **IDEA:** Since Convolutional Neural Networks are **state of the art** in image classification, it is important to try this kind of approach

- Start with a very **simple** architecture:
  - 2 Convolutional layers, composed by 32 and 64 filters, with ReLU activation
  - 1 Hidden dense layer (128 units)



Problems

- Too **few images** (Only 107 for the positive class)
- **Unbalanced** data (23% - 77%)
- Very heavy images (1 400 175 voxels each)
- Don't have a powerful **GPU**, Google colab has usage limitations

↓

- Need to resize images at (64,64,64) shape, losing data quality
- Can't perform undersampling because we have few images
- Can't perform oversampling because images are very similar (position and colors)

↓

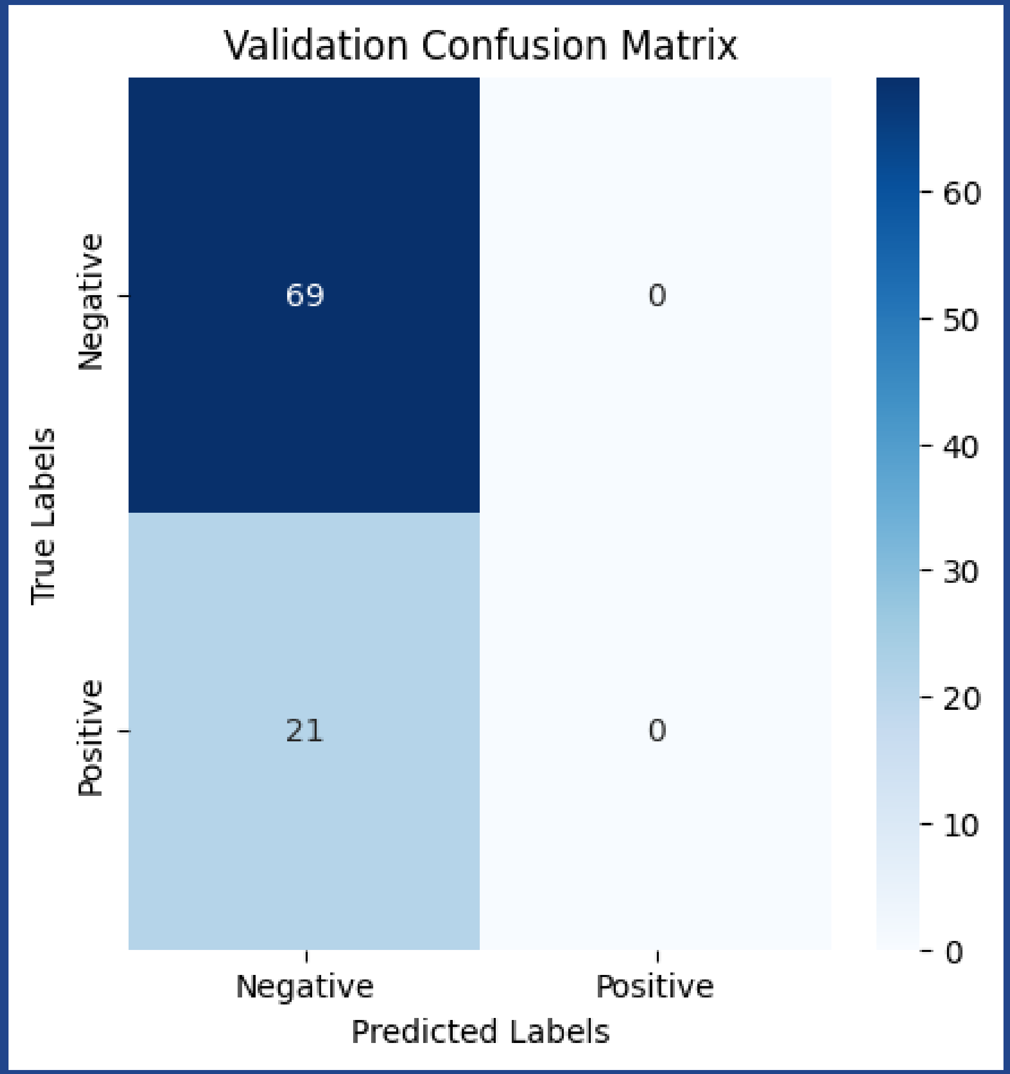
- Very **bad performance**

Results

- Only classify images as negative, since it is the most **numerous** class

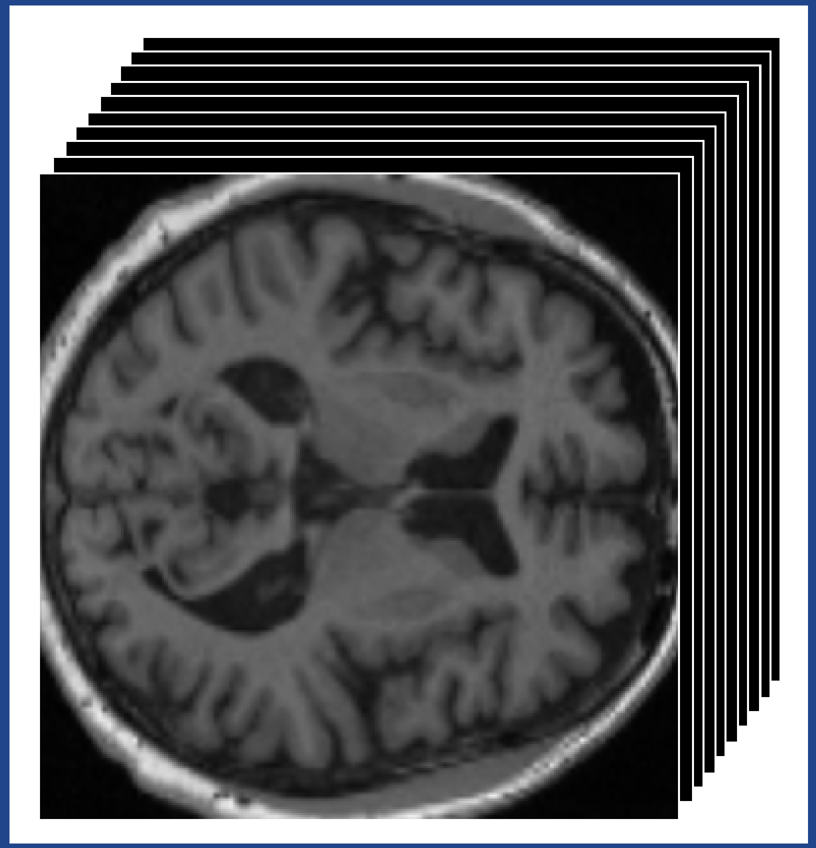
- Many modifications to the network architecture have been tried, however the computationally sustainable ones have not produced different results from the situation shown in this confusion matrix

- Decided **not to continue** with the 3D approach



2-Dimensional CNN

- **IDEA:** Divide the original 3D image in a lot of slices, cutting on a given dimension, and learn a classifier to predict them.
- All slices of the same original MR are put **in the same dataset split**, so there are not too similar images in different partitions
- After many attempts the best dimension was found to be the **transverse** plan
    - Each image is divided in 105 slices
- **Simple CNNs** perform better than complex ones
- Not possible to perform **data augmentation** for the previous mentioned reasons

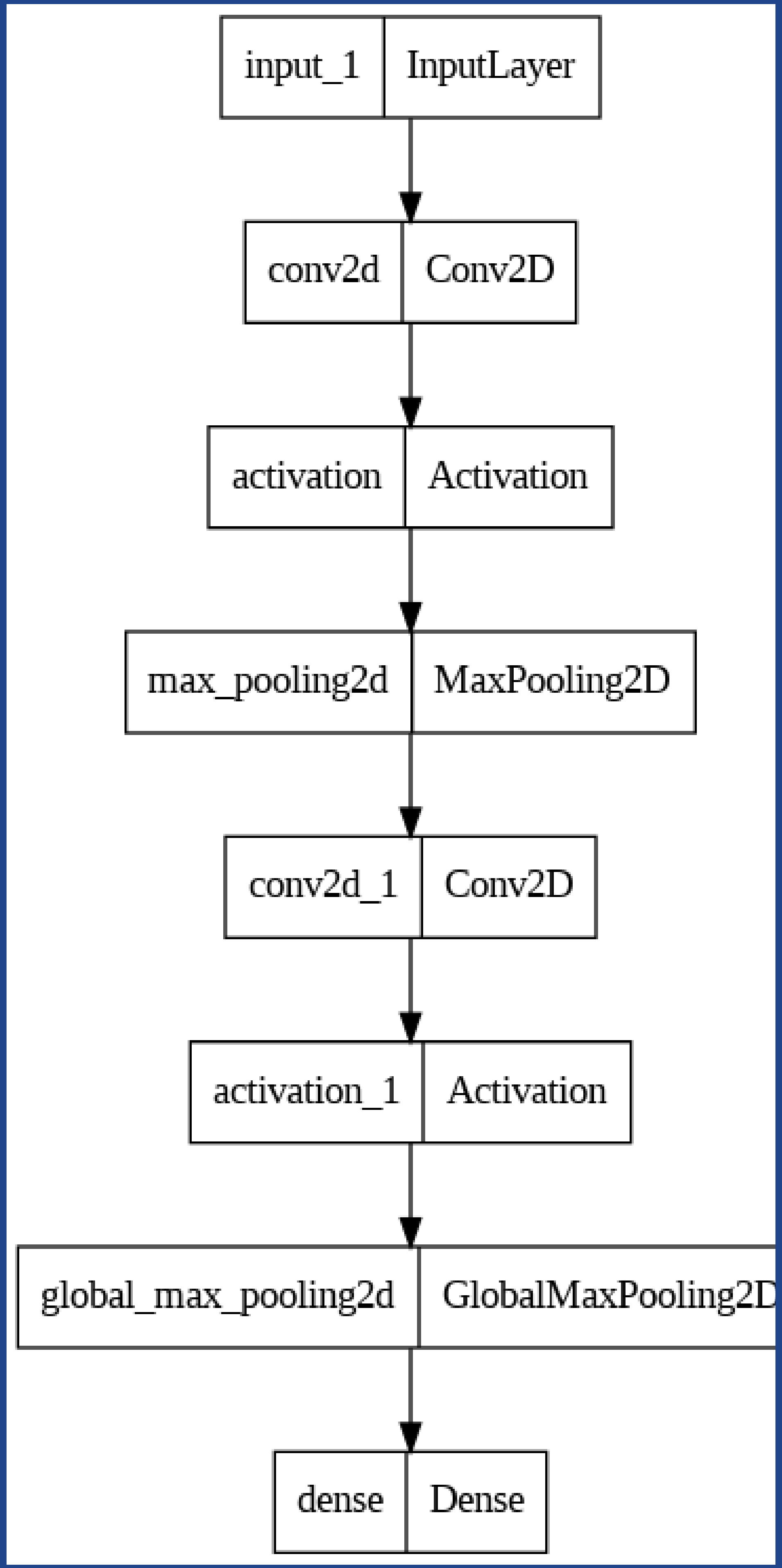


Architecture

- Different possible **architecture** have been tried
  - Dimension and number of convolutional filters
  - Optimizer
  - Activation function
  - Normalization
  - Regularization

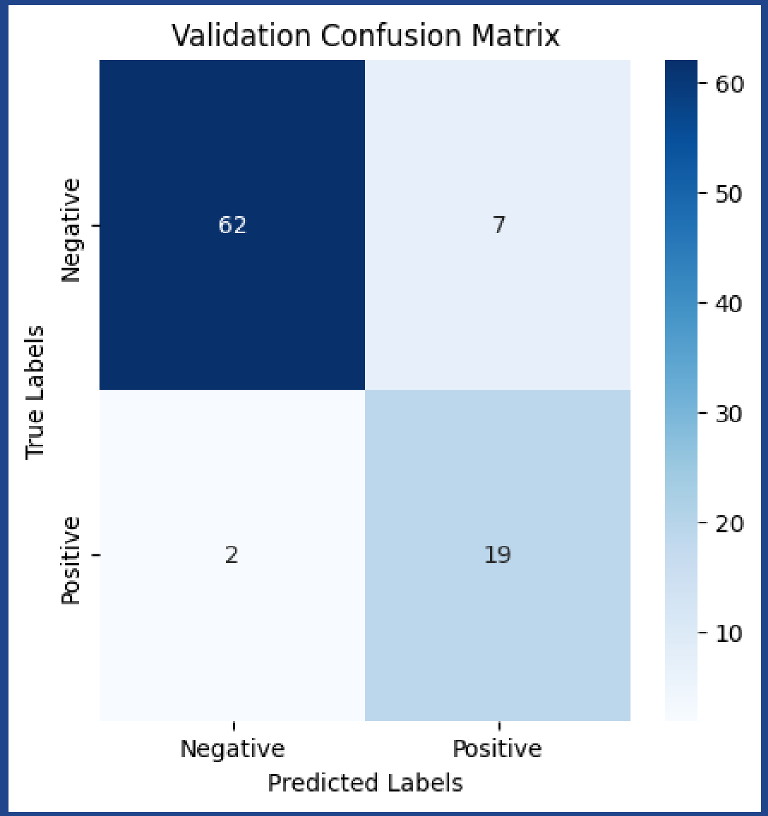
- Best result are provided by the most **simple** one:
  - 2 Convolutional layers, 32 and 64 filters, with ReLU activation
  - ReLU activation

- The netwok have been trained with the following **parameters**:
  - Adam optimizer
  - Learning Rate = 0.001
  - 40 epochs

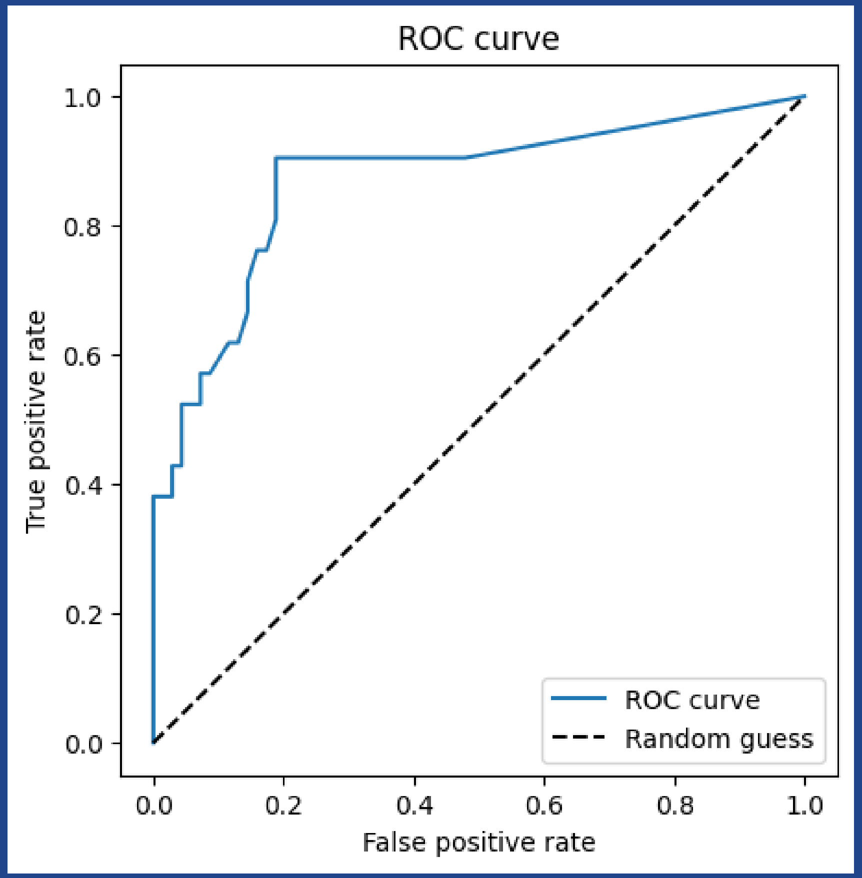


Results

- **Good Performances** on single slices
- To classify a complete image, all its slices are classified and the final prediction is based on the **percentage** of positive/negative images



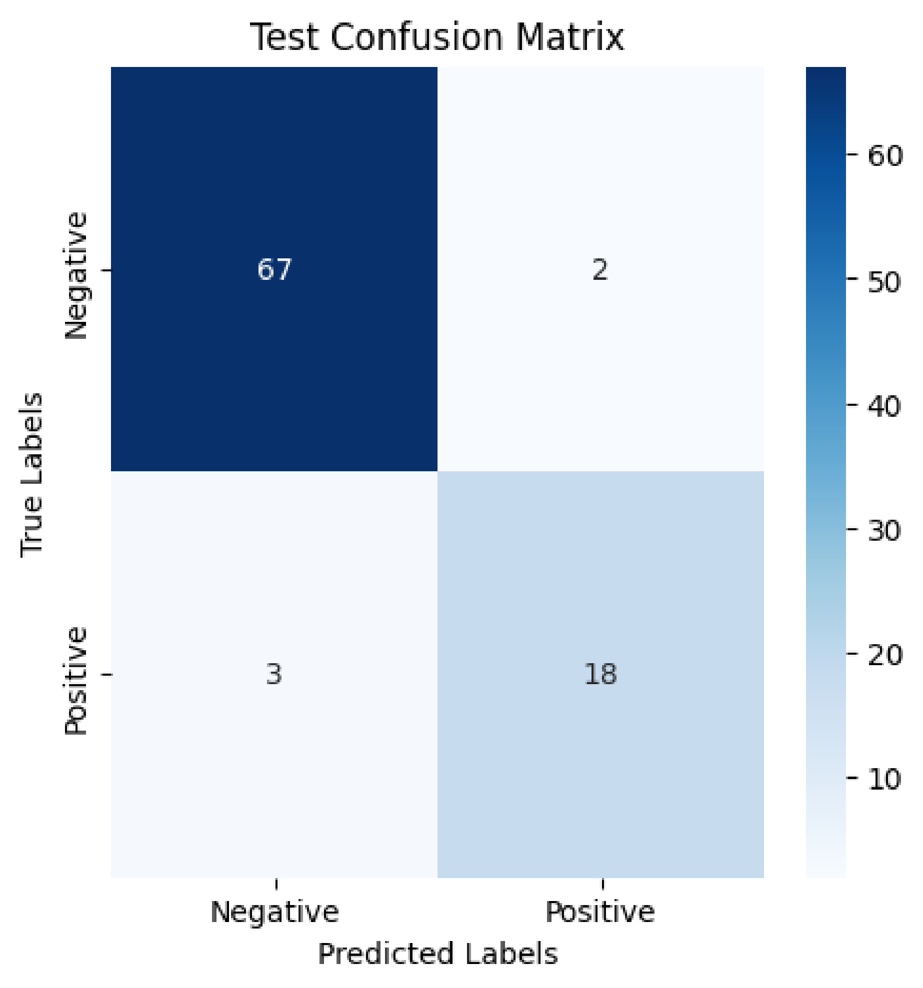
Precision: **0.73**
Recall: **0.90**
F1 score: **0.81**
Acc: **0.90**



- Look at the ROC curve to find the **optimal treshold**
- Good performance on Validation Set
- Need a final proof on Test set, there may be **overfitting**

Results on Test Set

- Performance metrics **increase on the test** set
- It is possible that Test set contains **easier images** to classify



Precision: **0.90**
Recall: **0.86**
F1 score: **0.88**
Acc: **0.94**

Alberto Gadda (ID 824029)
Greta Gravina (ID 881470)

Thanks for your attention

a.gadda@campus.unimib.it
g.gravina8@campus.unimib.it