Siamese Neural networks for Face Recognition

contact: *cursifrancesco@gmail.com*

Introduction

Face Recognition consists in identifying a person from their face's pictures.
For instance, given the picture below, the goal would be to recognise if the person in the figure is Jessica Alba or not.

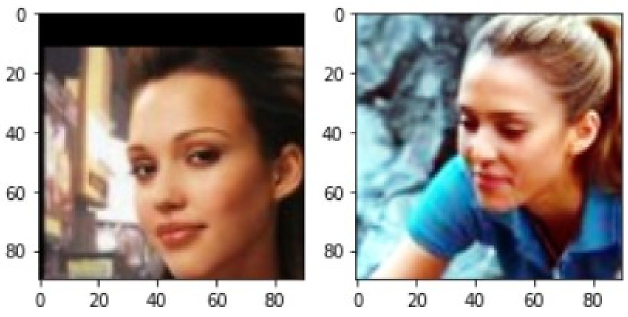

Fig. 1: Input pictures for face recognition

One approach to solve it would be to have a **Neural Network Classifier**, with as may classes as the people we would like to identify.

Even though neural networks (or other approaches) work very well in classification, this approach is very cumbersome because it requires a priori knowledge of the number of people we are going to classify.

Siamese Neural Networks

To overcome those limitations and have a more flexible framework, **Siamese Neural Networks (SNN)** can come very useful.

SNN consists of two identical Convolutional Neural Networks. The two networks in SNN have the same identical structure and weights. The only difference is in the inputs. SNN, in fact,have as inputs two images. The feature extractor learns and extracts features from both images and then a comparison between these features ($\mathbf{\Psi}_1, \mathbf{\Psi}_2$) is made.

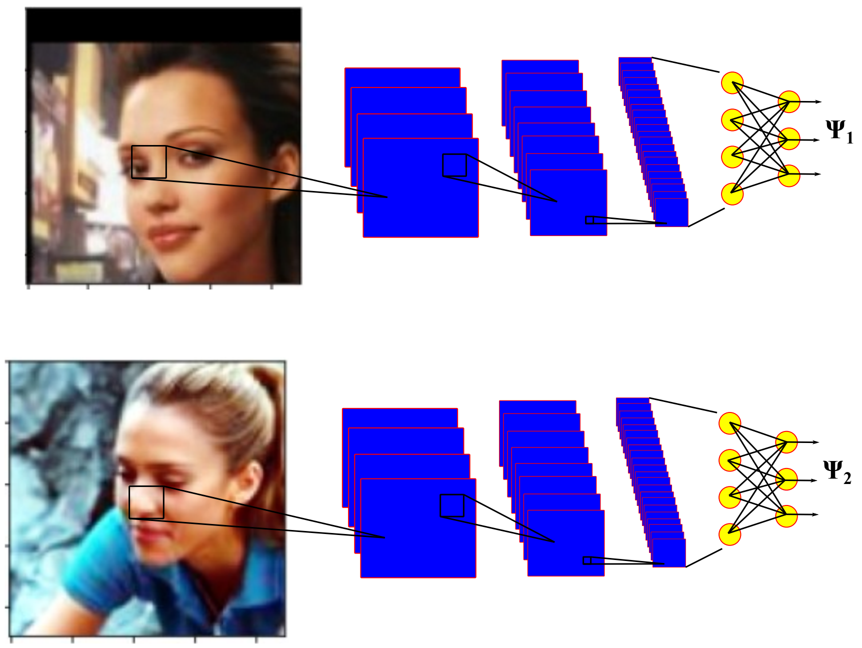

Fig. 2: Siamese Neural Network architecture

Depending on the metrics used, the closer the feature vectors are, the more similar the input pictures would be, meaning that they are of the same person.

Training SNN:

Generally SNN consist of a sequence of convolutional layers (conv+pooling) and fully connected layers to extract the features and map them in $\mathbb{R}^n$, where n is the dimension (number of nodes) of the output layer. *Remember that it is the same network that is used to extract features from the two images.*

A metrics which is generally used is the **Euclidean Distance** $\underset{\cdot}{=}||\mathbf{\Psi}_1 - \mathbf{\Psi}_2||$ and it can be employed with different training losses:

- **Binary Cross Entropy Loss**: in this case the distance d is input to a logistic regressor, which outputs the probability of the two images of being similar. The predicted probability would be $\hat{y} = \sigma(wd+b)$, where w, b are additional networks weight and bias to learn.
  The loss to use for training the networks's weights will just be $L \;=\; -(\;\tilde{y}\;\log(\hat{y})\;+\;(1\;-\;\tilde{y})\;\log(1\;-\;\hat{y}))$, where $\tilde{y}$ is the ground truth label (0 if the images are of different people, 1 for the same person).
- **Contrastive Loss**: in this case the distance metrics is directly used to minimize $L \;=\;\; \tilde{y}\; d^2 \,+\, (1\,-\,\tilde{y})\;\max(m-d, 0)^2$, where m is a margin value used for reducing the effect of misclassification. This means that, when the pictures belong to the same person ($\tilde{y} = 1$), the aim is to minimize the distance between the two feature vectors. If they belong to different people ($\tilde{y} = 0$)and they have already a large distance, the loss would be 0. If they are of different people and the loss is small (misclassified by the network), then it would be penalized and pushed towards the margin.

When training the network, the training data must consists of pairs of pictures containing both similar (of the same person) and dissimilar pictures.