Siamese Network & Triplet Loss

One shot learning

- Deep Convolutional Neural Networks have become the state of the art methods for image classification tasks.
- One of the biggest limitations is they require a lots of labelled data.
- In many applications, (eg: Building a face recognition model), collecting this much data is sometimes not feasible.

Reference - *https://towardsdatascience.com/one-shot-learning-with-siamese-networks-using-keras-17f34e75bb3d

Classification vs One shot learning

- Standard Classification task - the input image is fed into a series of layers, and finally at the output we generate a probability distribution over all the classes (typically using a Softmax).

- Two critical points to be noted:
  - Require large number of images for each class
  - If the network is trained only on, let’s say, 3 classes of images, then we cannot expect to test it on any other class.

- If we want our model to classify the images of other class as well, then we need to first get a lot of images for that particular class and then we must re-train the model again

Challenges in real world

- However, for some applications in real world, we neither have large enough data for each class and the total number classes is huge as well as dynamically changing.

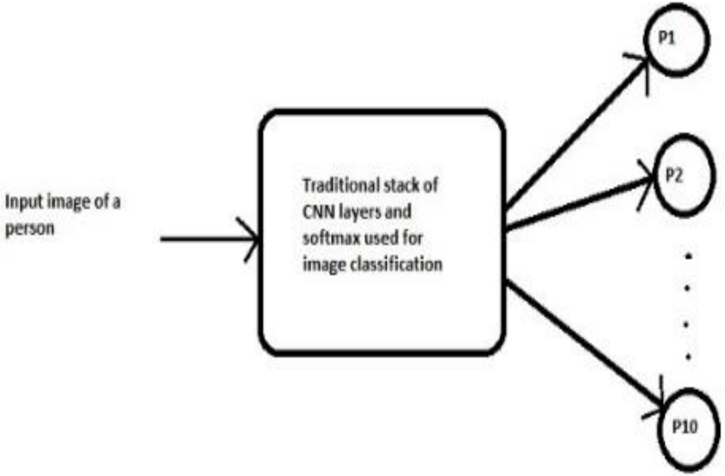
- The cost of data collection and periodical re-training is too high.

ONE SHOT LEARNING TO THE RESCUE!!

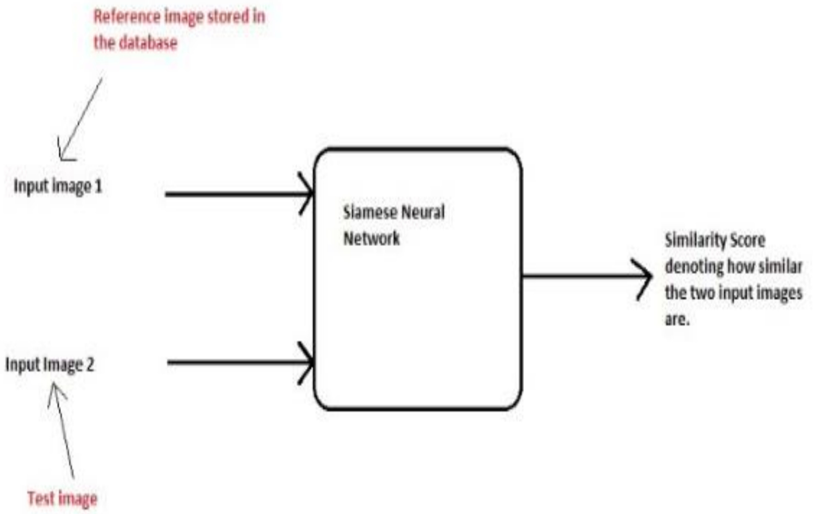
*

One shot learning

One shot learning - requires only one training example for each class.


Input image of a person
Traditional stack of CNN layers and softmax used for image classification
P1
P2
P10

Traditional CNN


Reference image stored in the database
Input image 1
Input Image 2
Test image
Siamese Neural Network
Similarity Score denoting how similar the two input images are.

One shot

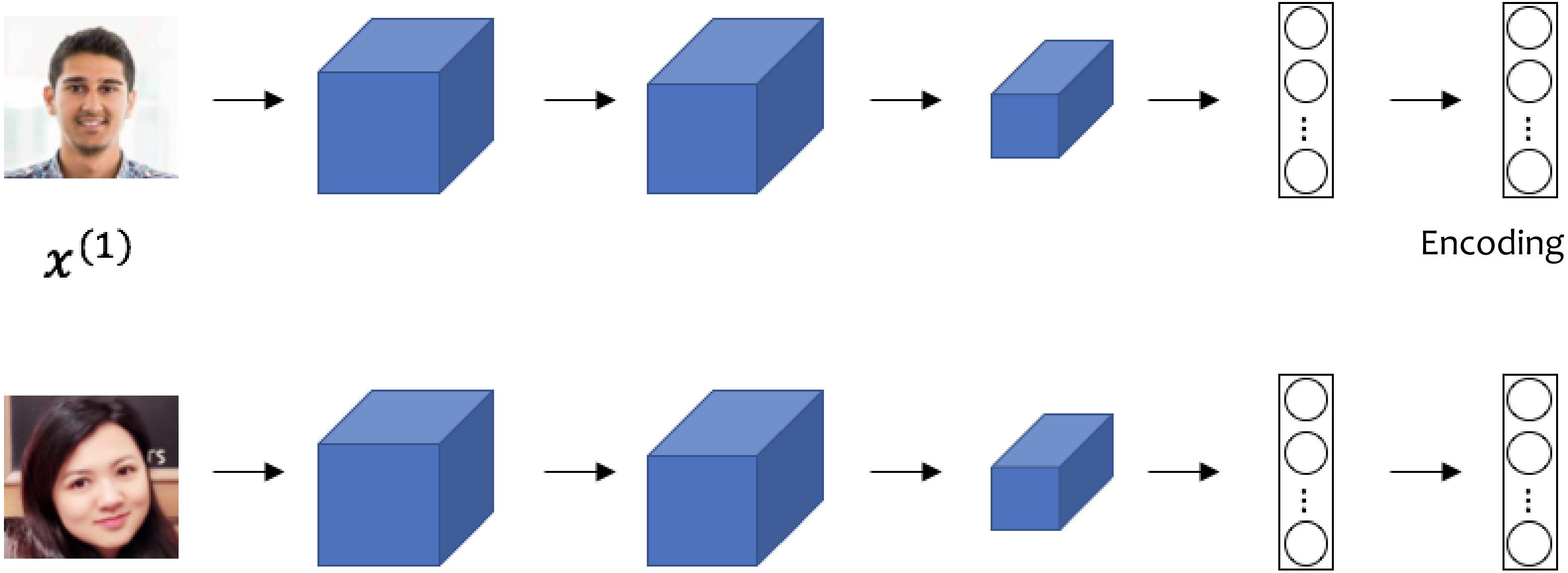
*

Siamese networks

- The objective of a Siamese network is to find how similar two comparable things are.
- For eg. verification of signature, face recognition..).
- Any siamese network has two identical subnetworks, which share common parameters and weights

*Reference - https://towardsdatascience.com/siamese-network-triplet-loss-b4ca82c1aec8

Siamese Network



$x^{(1)}$

Encoding of x(1)

$x^{(2)}$

Encoding of x(2)

Siamese networks

- To compare both the images x1 and x2,
  - Calculate the distance between their encoding f(x1) and f(x2).
  - If it is less than a threshold (a hyperparameter), it means that the two pictures are the same person, if not, they are two different persons.

Siamese networks

$$d(x^{(1)}, x^{(2)}) = \|f(x^{(1)}) - f(x^{(2)})\|_2^2$$

distance function between two encoding of x1 and x2.

And this is working for any two images *xi* and *xj*.

If $x^{(i)}, x^{(j)}$ are the same person, $\|f(x^{(i)}) - f(x^{(j)})\|^2$ is small.
If $x^{(i)}, x^{(j)}$ are different persons, $\|f(x^{(i)}) - f(x^{(j)})\|^2$ is large.

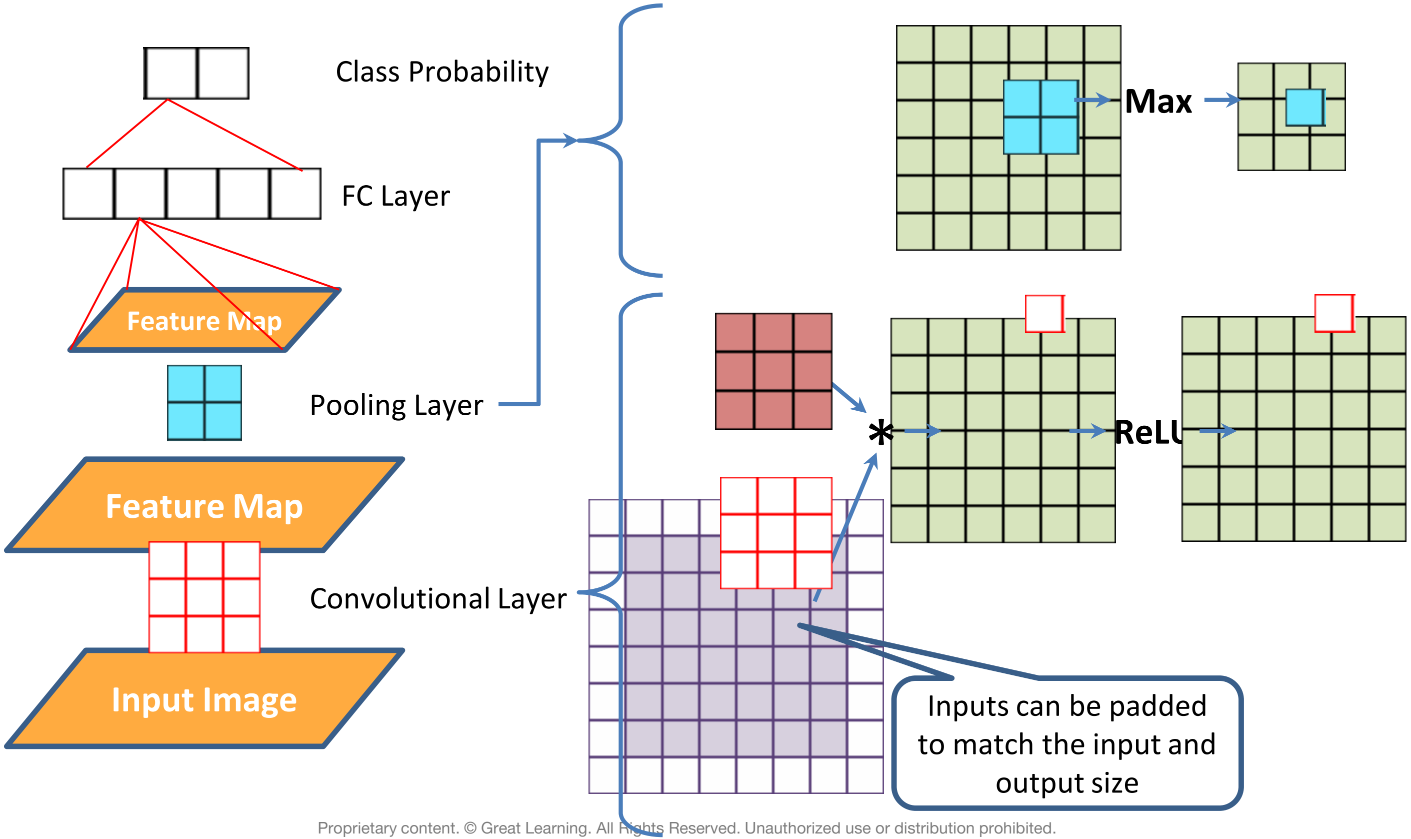
In Siamese networks,

- we take an input image of a person and find out the encodings of that image,
- then, we take the same network without performing any updates on weights or biases and input an image of a different person and again predict it’s encodings.

Siamese Network

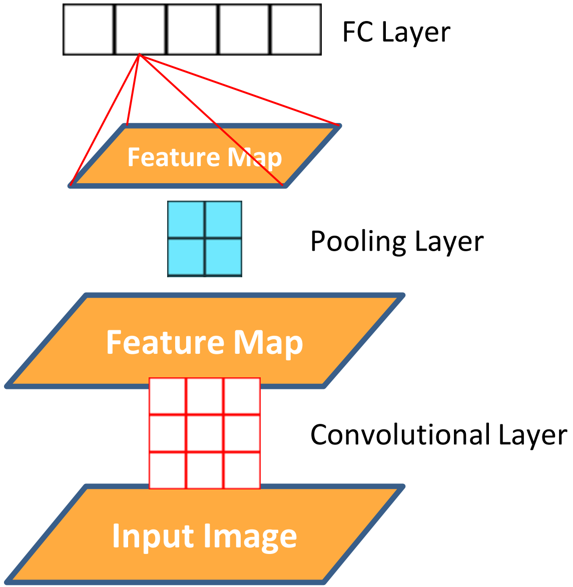
- Now, we compare these two encodings to check whether there is a similarity between the two images.
- These two encodings act as a latent feature representation of the images.
- Using this, we compare and tell if the two images have the same person or not.

CNN Revisited

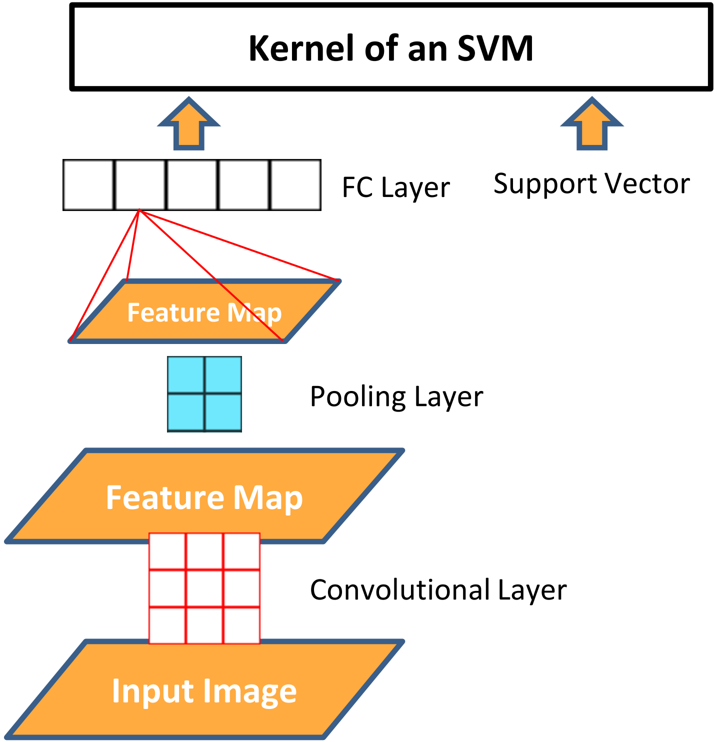


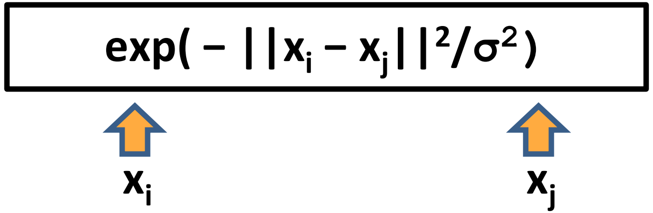
Inputs can be padded to match the input and output size

The last FC layer gives good features



These features are transferable and can be used in an SVM, for example





Yosinski, Jason, Clune, Jeff, Bengio, Yoshua, and Lipson, Hod. "How transferable are features in deep neural networks?"

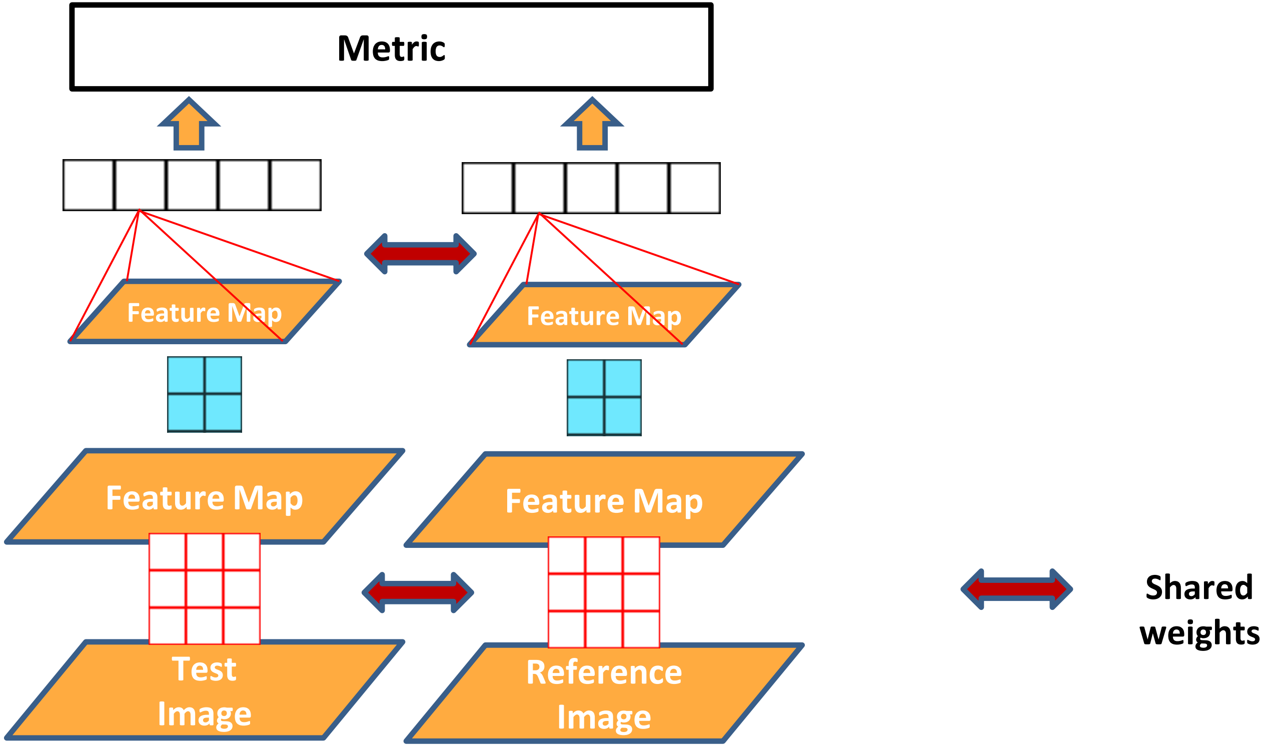
Properties of a kernel

- Similarity metric
- High value for similar pairs of inputs
- Low value for dissimilar inputs
- Positive semi-definite

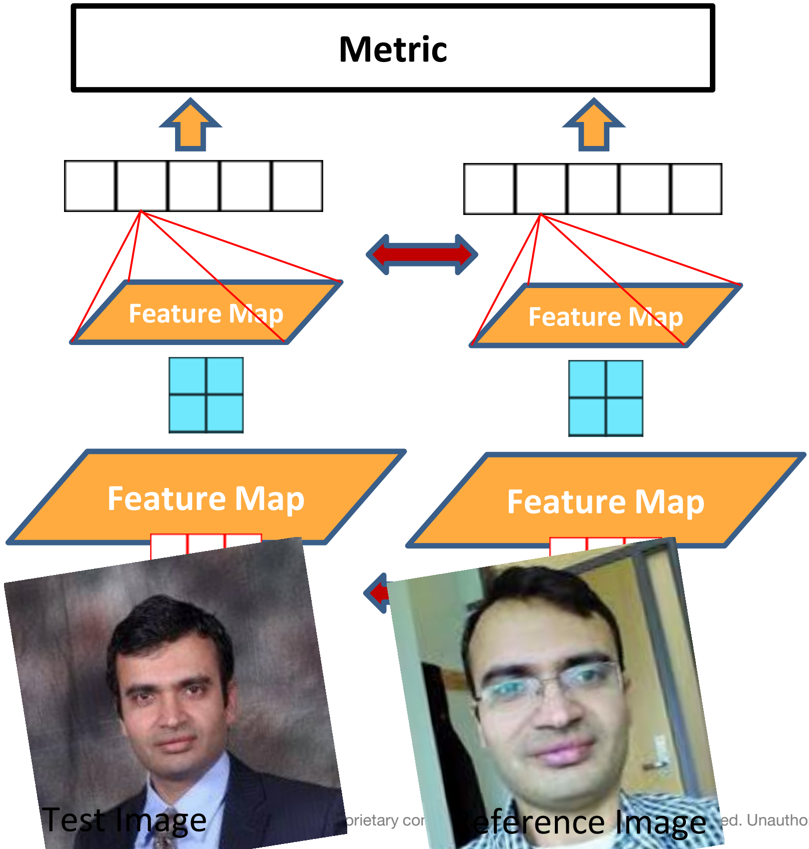
Learning the kernel is called metric learning

- A metric is like a distance
- Inverse of similarity
- It is symmetric
- It follows triangle inequality

Siamese network as metric learning

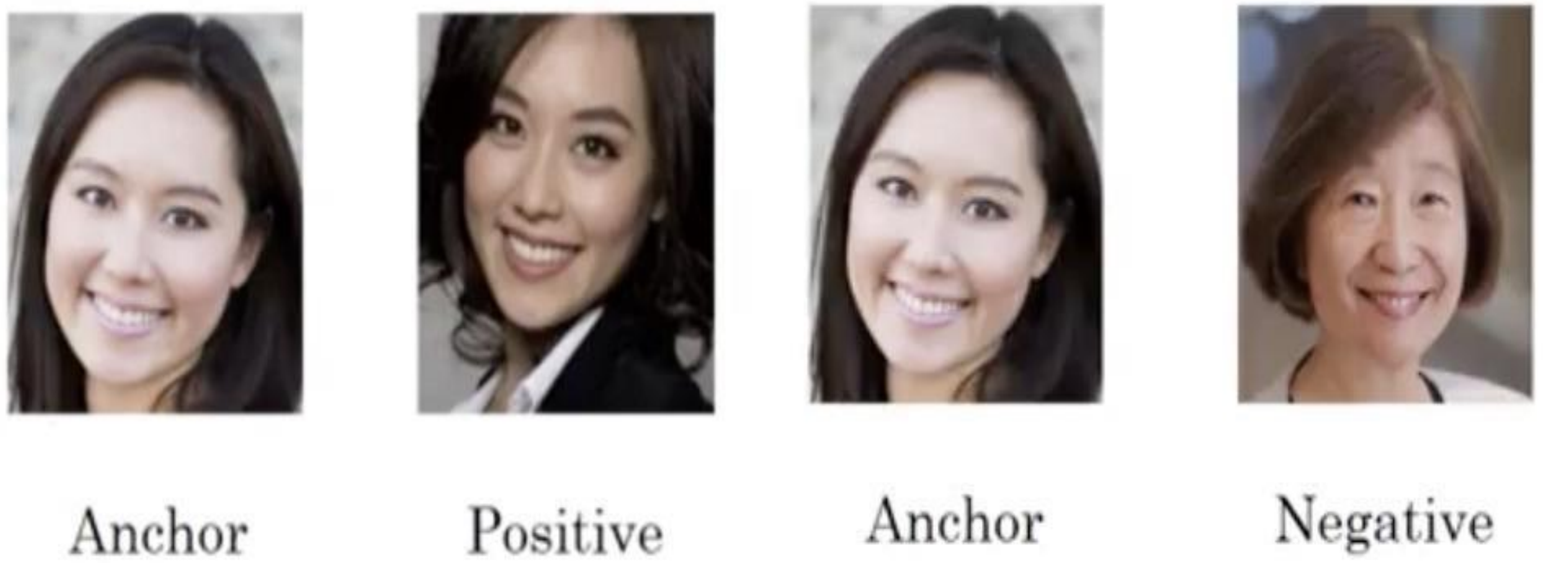


For example, face verification



⟷ **Shared weights**

Triplet Loss



Triplet Loss

- You can train the network by taking an anchor image and comparing it with both a positive sample and a negative sample.
- The dissimilarity between the anchor image and positive image must low and the dissimilarity between the anchor image and the negative image must be high.

Triplet Loss Function

$$\mathcal{L} = max(d(a,p) - d(a,n) + margin, 0)$$

“a” - represents the anchor image

“p” - represents a positive image

“n” - represents a negative image

margin - is a hyperparameter. It defines how far away the dissimilarities should be

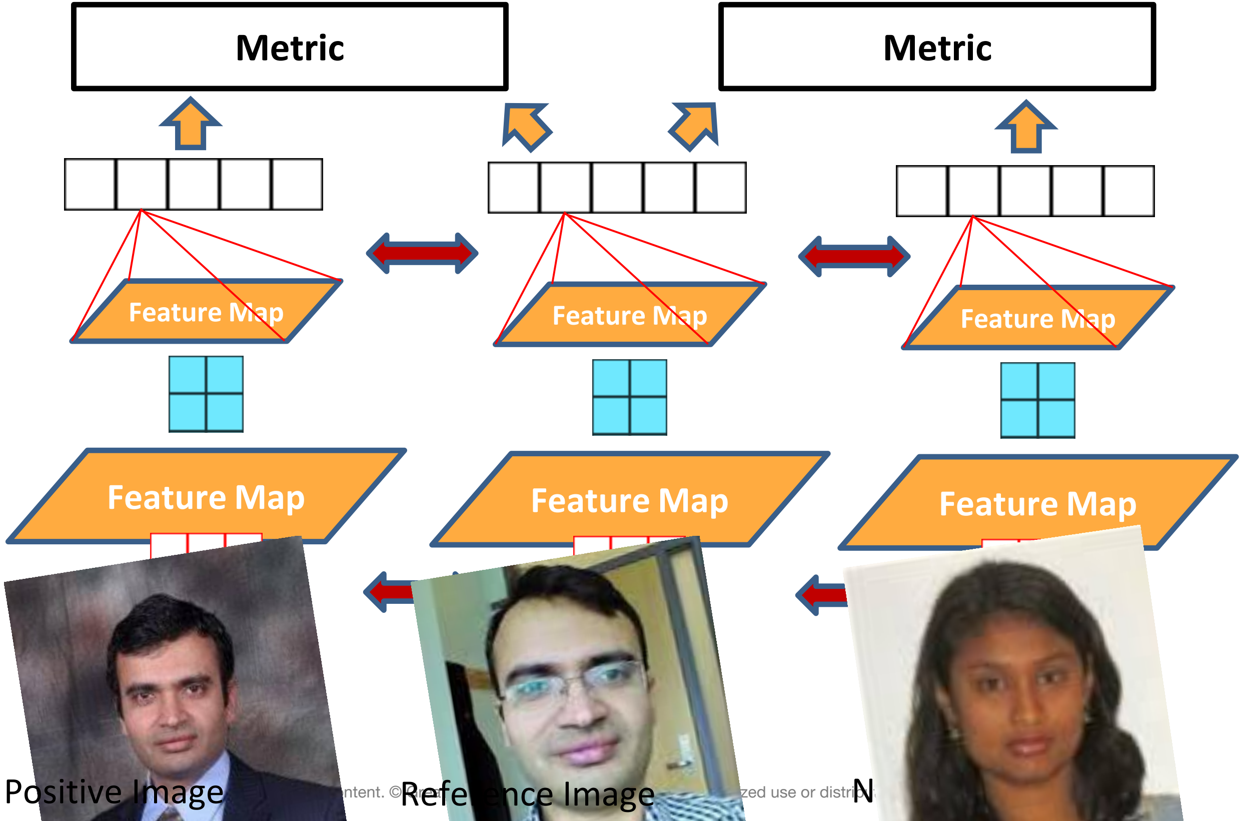
Triplet Loss Function

By using this loss function we calculate the gradients and with the help of the gradients, we update the weights and biases of the siamese network.

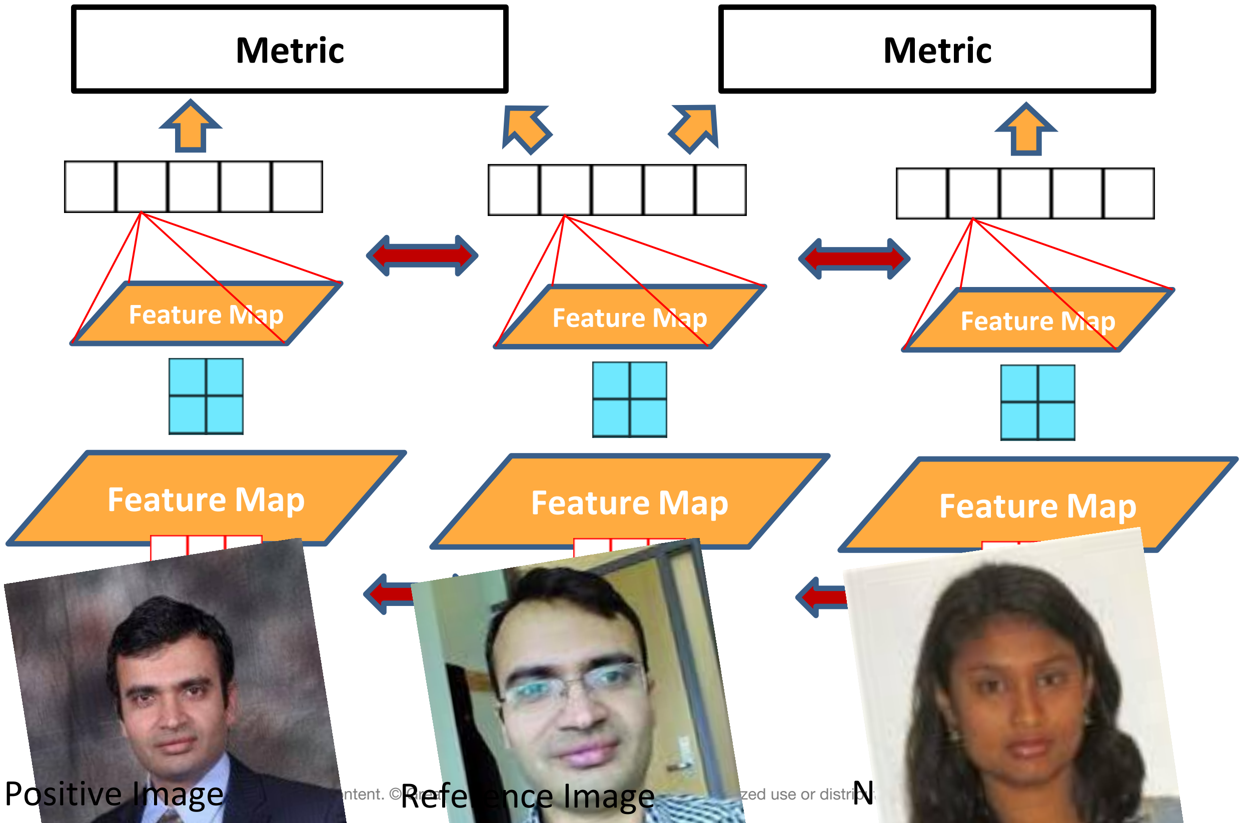
Paper on triplet loss - https://arxiv.org/abs/1703.07737

Interesting Read - https://bamos.github.io/2016/01/19/openface-0.2.0/

Target values differ for similar and dissimilar pairs



Or, the relative values are different



Metric

Metric

Feature Map

Feature Map

Feature Map

Feature Map

Feature Map

Feature Map

Positive Image

Reference Image

N

Two ways of viewing a metric

- Absolute terms (Regular Siamese training)
  - Distance (x_{ref}, x_{+}) = Low; Distance (x_{ref}, x_{-}) = High
  - Similarity (x_{ref}, x_{+}) = High; Similarity (x_{ref}, x_{-}) = Low
- Relative terms (Triplet Siamese training)
  - Distance (x_{ref}, x_{-}) – Distance (x_{ref}, x_{+}) > Margin
  - Similarity (x_{ref}, x_{+}) – Similarity (x_{ref}, x_{-}) > Margin
- Class probability was based on a single input
  - ClassProb (x,c) = High when $x \in c$; otherwise low

Some distance and similarity measures

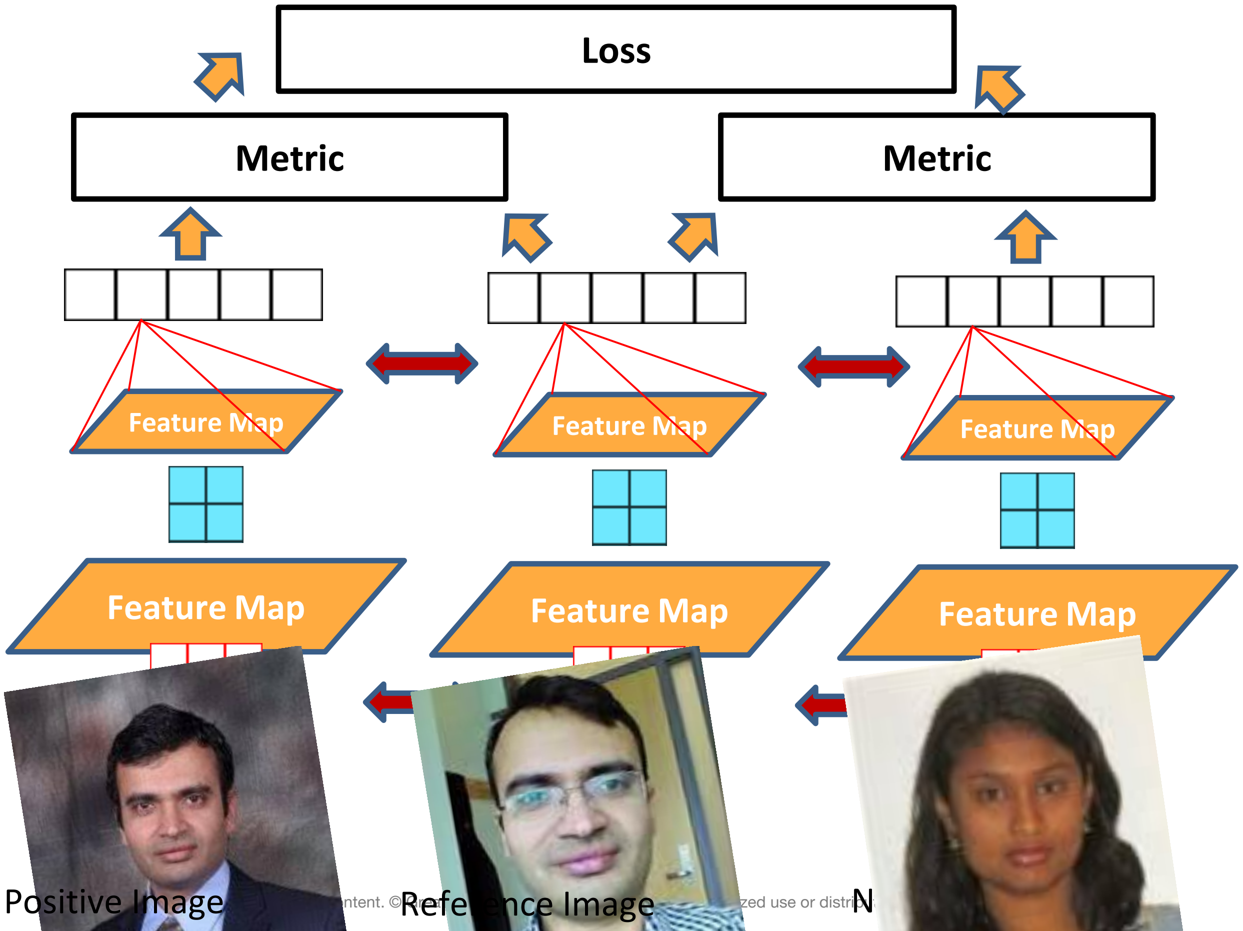
- Distances examples
  - L2 norm of difference (Euclidean distance)
  - L1 norm of difference (City-block/Manhattan dist.)
- Similarity examples
  - Dot product
  - Arc cosine
  - Radial basis function (RBF)

Some distance and similarity measures

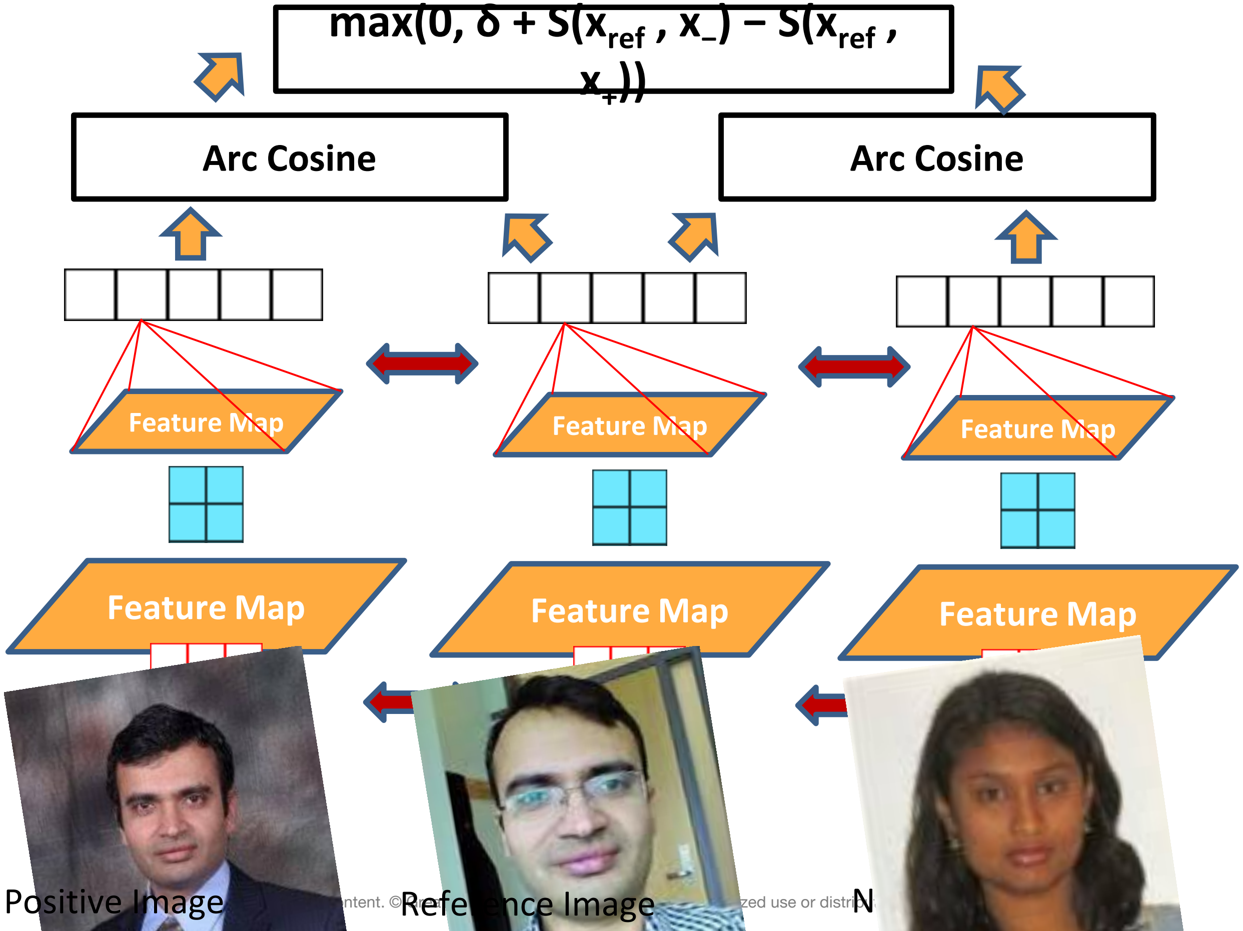
- Distances examples
  - $||(f(x_i) - f(x_j)||_2^2$
  - $|(f(x_i) - f(x_j)|_1$

- Similarity examples
  - $f(x_i)^T f(x_j)$ or $f(x_i) \cdot f(x_j)$
  - $f(x_i) \cdot f(x_j) \;/\; (||f(x_i)|| \; ||f(x_j)||)$
  - $\exp(- ||x_i - x_j||^2/\sigma^2)$

Loss gradient is propagated back



Loss gradient is propagated back



$\max(0, \delta + S(x_{ref}, x_{-}) - S(x_{ref}, x_{+}))$

Arc Cosine

Arc Cosine

Feature Map

Feature Map

Feature Map

Feature Map

Feature Map

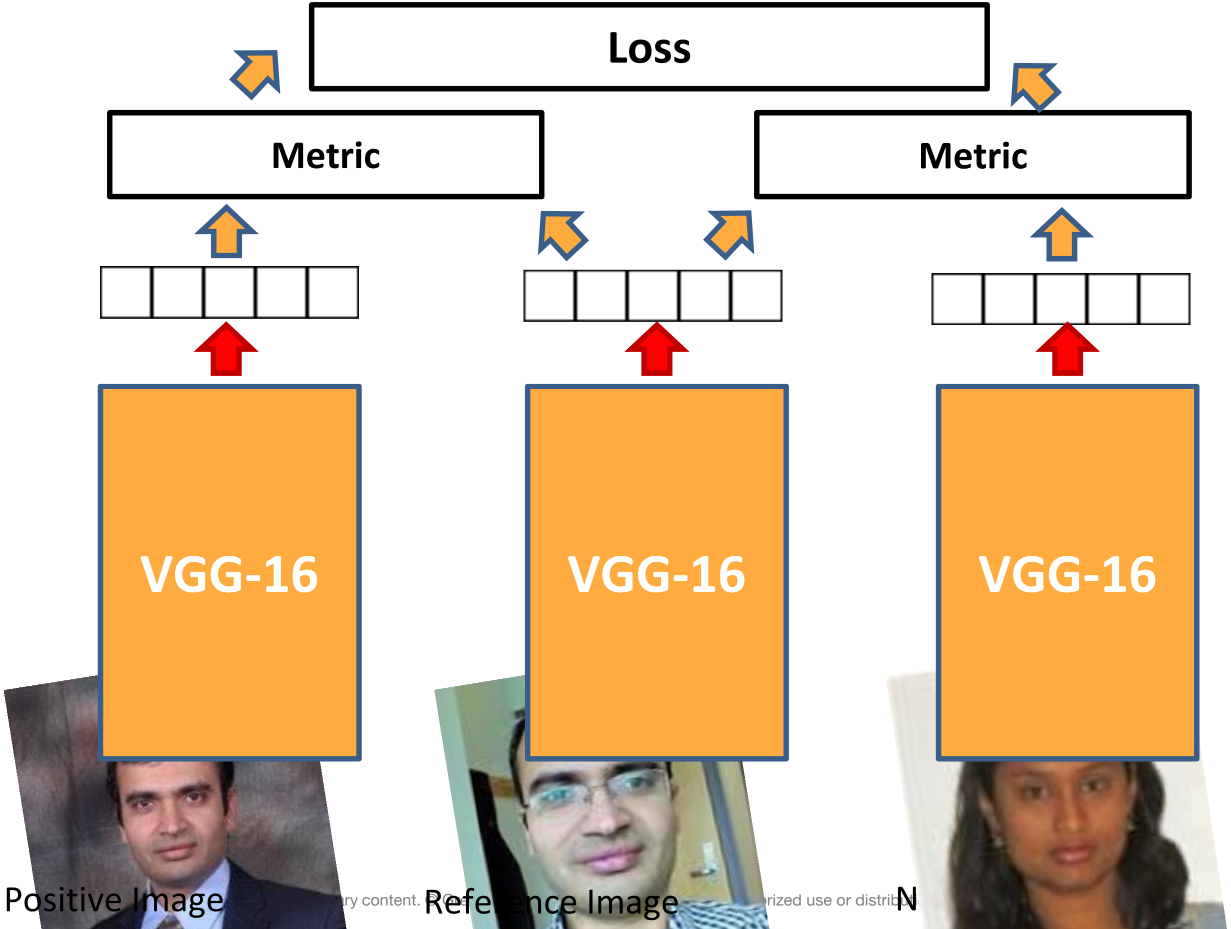
Feature Map

Positive Image

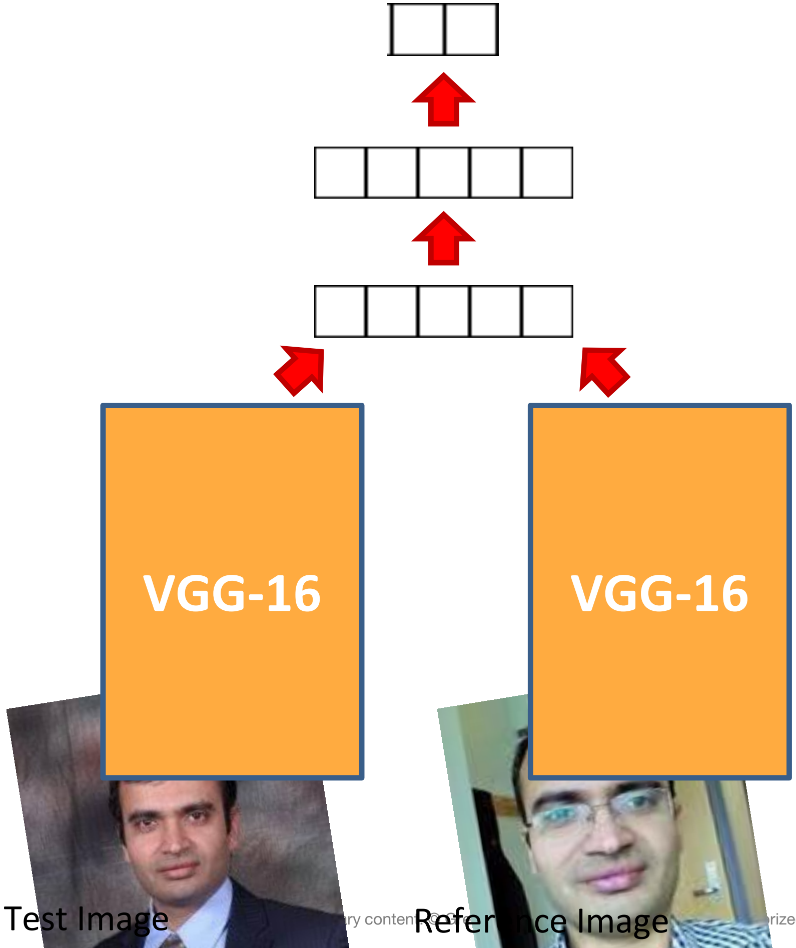
Reference Image

N

Pre-trained networks can be used



Some joint layers can also be added


VGG-16
VGG-16
Test Image
Reference Image