Metric Learning
and
Few Shot learning

https://shala2020.github.io/

Courtesy:
Some of the materials are adapted from : Moitreya Chatterjee, Yunan Luo, “Similarity Learning with (or without) Convolutional Neural Network”. University of Illinois Urbana-Champaign & Andrew Ng: Deeplearning.ai

Motivation for Metric Learning

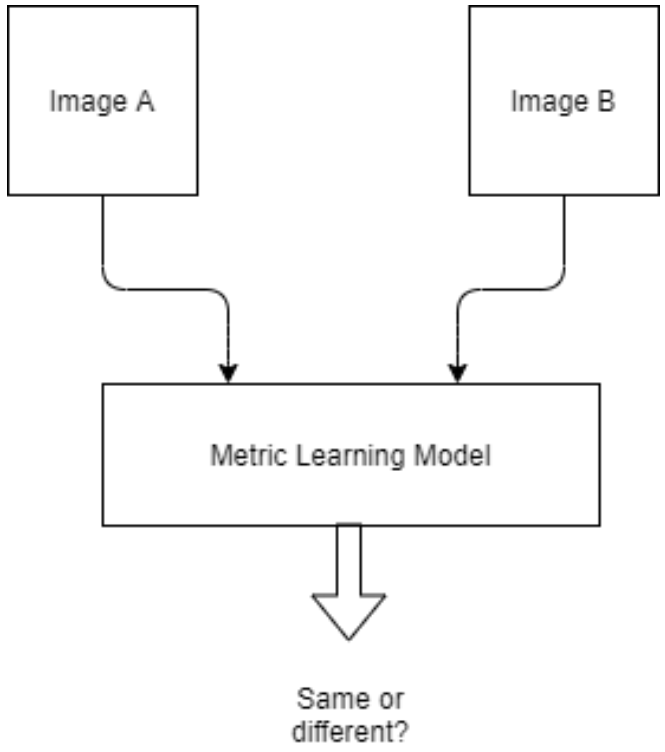


Fig : Overview of the problem

- We are asking the yes or no question whether A and B are similar or different.
- Metric learning model compares the images by defining a similarity measure (1- distance) between the images A and B

Defining Metric Learning

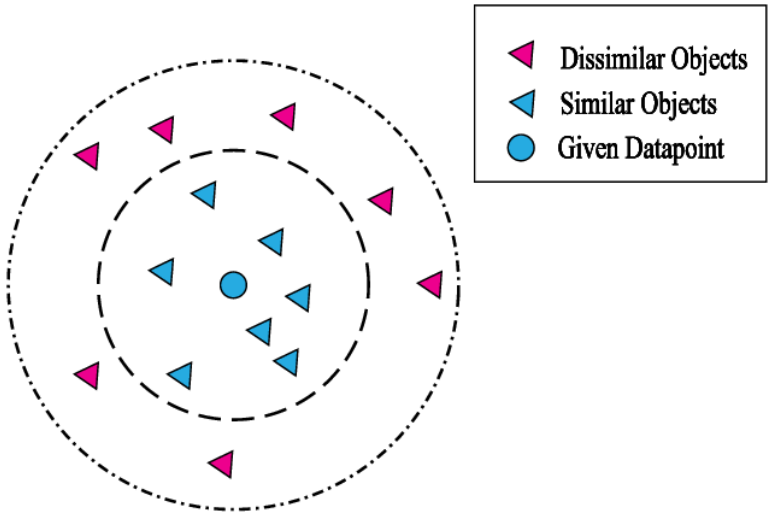


Fig : Objective of metric learning

- “Metric learning is an approach based directly on a distance metric that aims to establish similarity or dissimilarity between objects.”

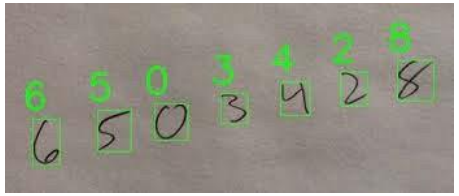
- Reduce “distance” between similar objects while increasing the distance between dissimilar objects.

Image Source: Google, PyImageSearch

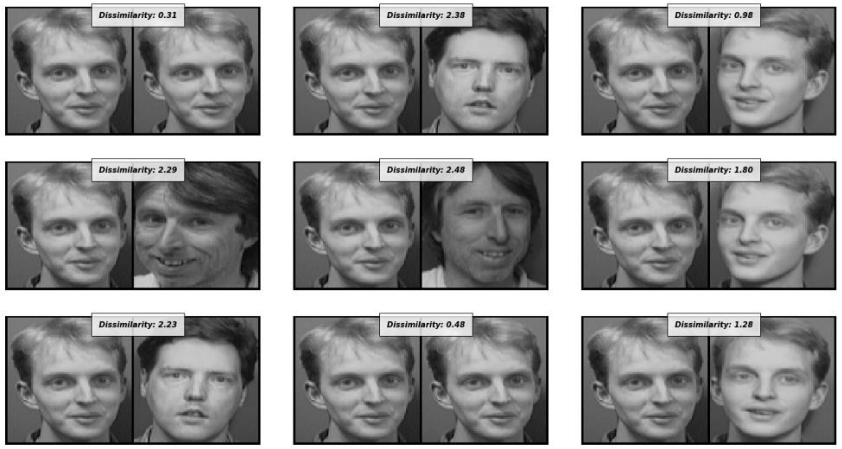
Applications of Metric Learning

Several applications exists in today's world:

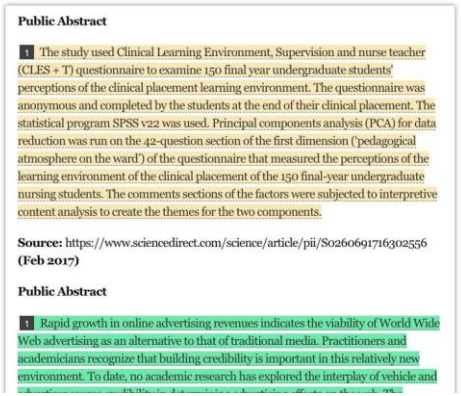
- Recognizing handwriting in checks.

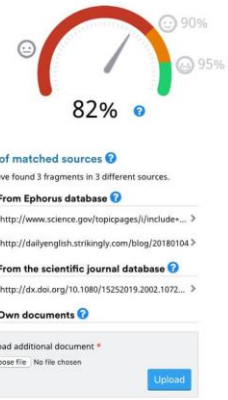


- Face recognition task.



- Plagiarism Checks





- Face detection task.

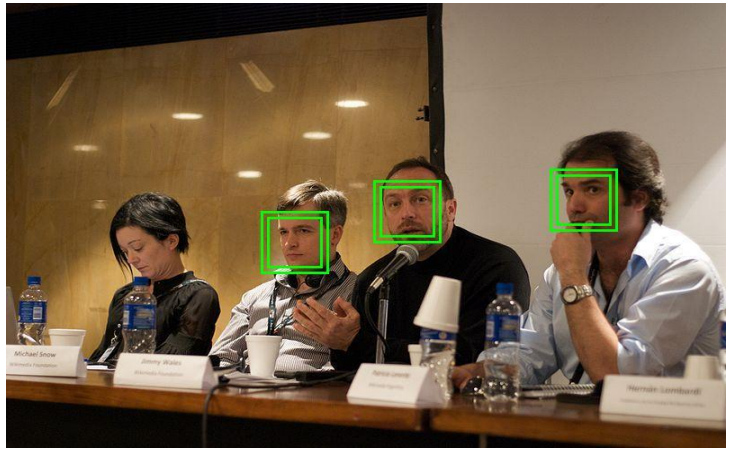


Image Source: Google, PyImageSearch

Defining Metric:

- A Metric is a function that induces a measure of similarity by quantifying a "distance" between every pair of elements in a set.

- If f(x,y) is a distance metric for all x, y, z belonging to the set, then:
    - Non-negativity: $f(x, y) \geq 0$
    - Identity : $f(x, y) = 0 \iff x = y$
    - Symmetry: $f(x, y) = f(y, x)$
    - Triangle Inequality: $f(x, z) \leq f(x, y) + f(y, z)$

Types of Metrics

In broad terms metrics are of two kinds:

- Pre-defined Metrics:

  E.g. Euclidian Distance:

$$f(x, y) = (x - y)^T(x - y)$$

- Learned Metrics:

  E.g. Mahalanobis Distance:

$$f(x, y) = (x - y)^T M(x - y) ;$$

  where M is estimated from the data.

Mahalanobis distance based learning is an example for unsupervised metric learning and also a linear metric learning scheme.

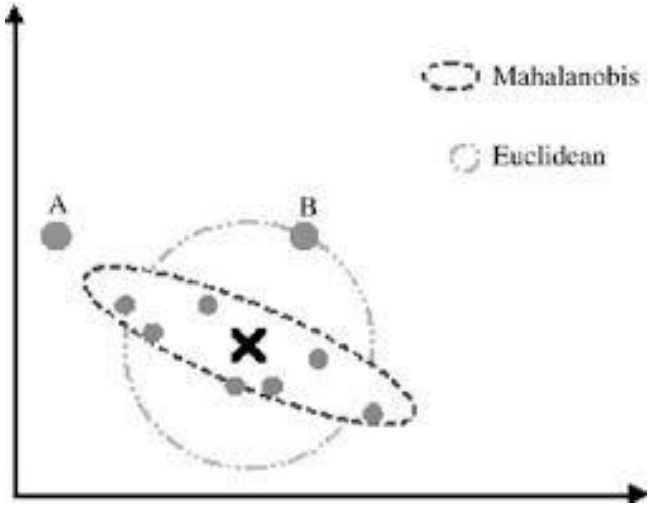


Fig: Mahalanobis vs Euclidean distance

Image Source: Google search

Supervised Metric Learning

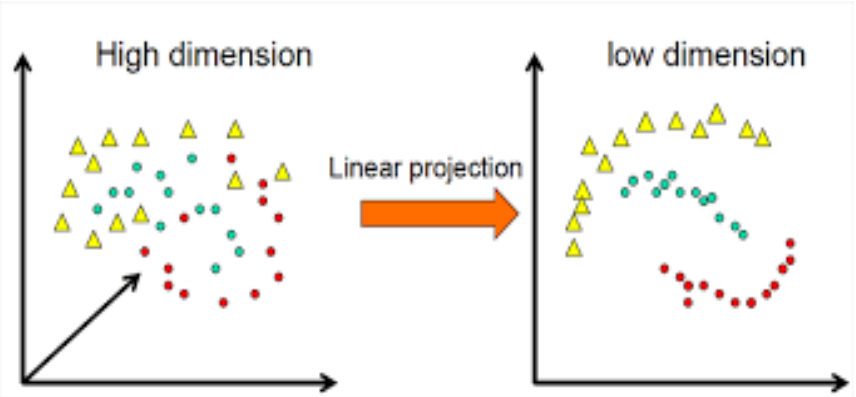
- In this setting, we have access to labeled data samples (z = {x, y}).

- Typical procedure is extract representative class specific features and then use one of the unsupervised metrics for performing a mapping.

The traditionally this done by :

1. Extract Features:

   Eg. color, texture of the input images.



2. Learn Similarity between the extracted features using the distance measure.

Bellet, A., Habrard, A. and Sebban, M., 2013. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*.

Supervised Metric Learning

The problem of traditional metric learning based methods is that the feature representation of the image is agnostic to the metric used in the framework.

CNNs resolves this issue by extracting the representation of the input data conditioned on the "similarity" measure being used, by an end-to- end learning.

Defining the Problem

We are interested in creating a model that has:

- Input: Given a pair of input images, we want to know how “similar” they are to each other.

- Output: The output can take a variety of forms:
  - Either a binary label, i.e. 0 (same) or 1 (different).
  - A Real number indicating how similar a pair of images are.

Constructing a Siamese Network

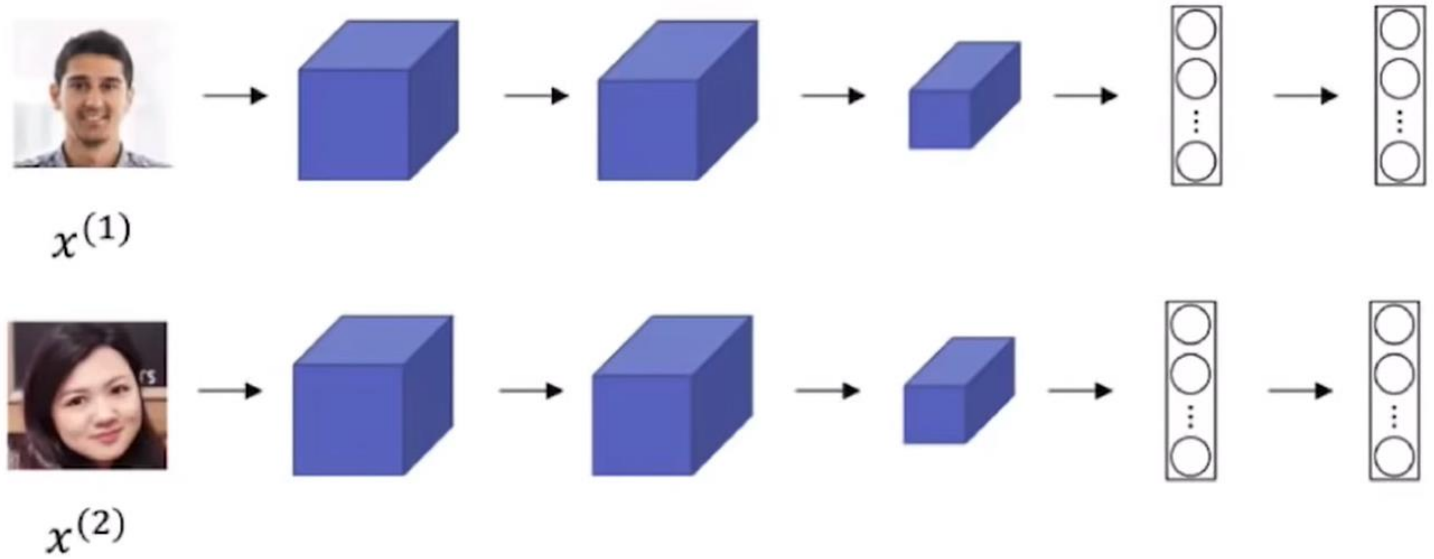


Image Credits : Andrew Ng , Deeplearning.ai

Siamese Network as a Binary Classifier

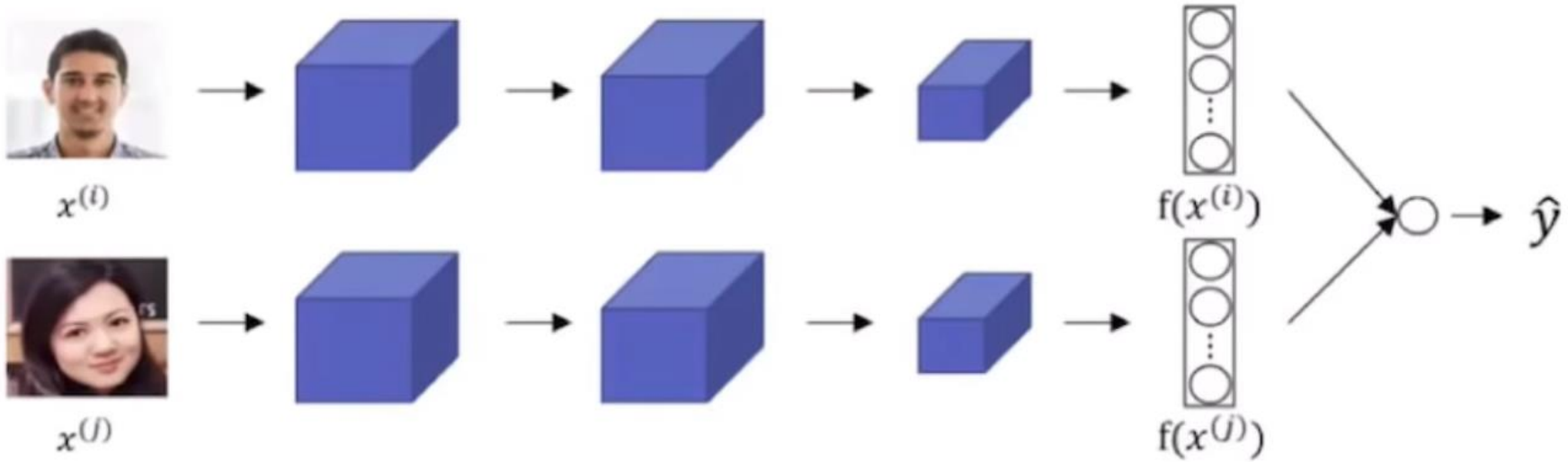
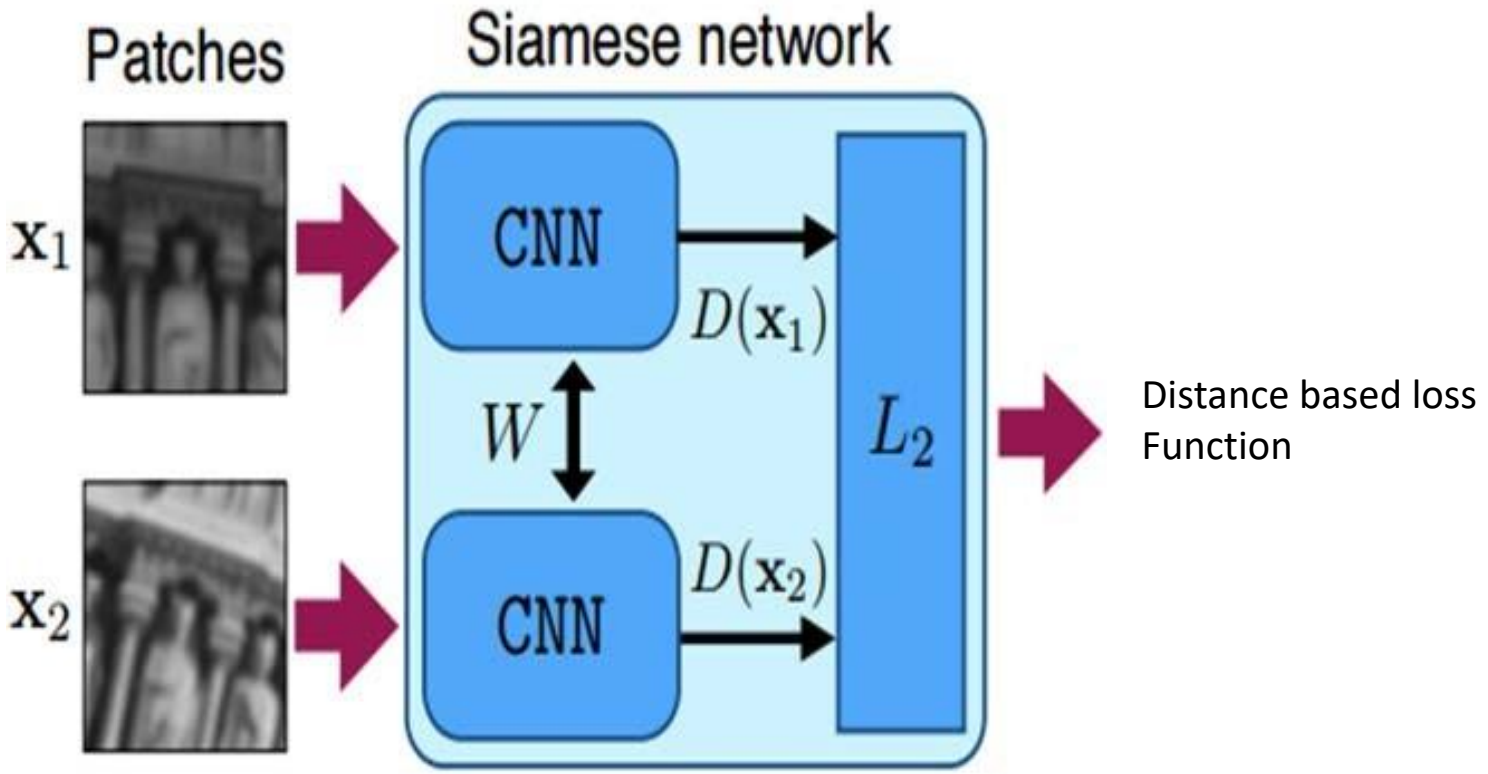


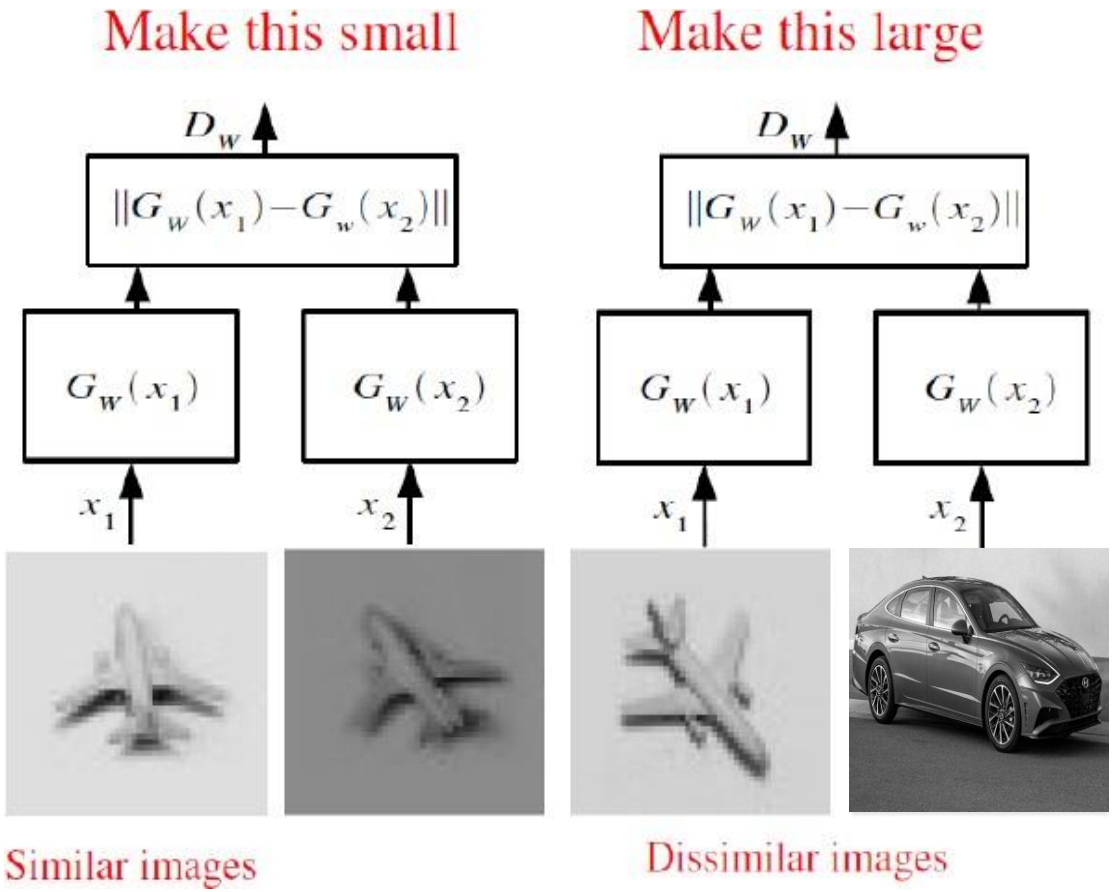
Image Credits : Andrew Ng , Deeplearning.ai

Siamese CNN with distance based loss



Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P. and Moreno-Noguer, F., 2015. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 118-126).

Siamese CNN – Loss Function



The final loss used is :

$L = \sum \text{loss of positive pairs} + \sum \text{loss of negative pairs}$

Chopra, S., Hadsell, R. and LeCun, Y., 2005, June. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (Vol. 1, pp. 539-546). IEEE.

Contrastive Loss function

$$L = \begin{cases} \min d_w(x_1, x_2) & if \quad PositivePair \\ \max d_w(x_1, x_2) & if \quad NegativePair \end{cases}$$

To maximise d(x1,x2) is equivalent to:

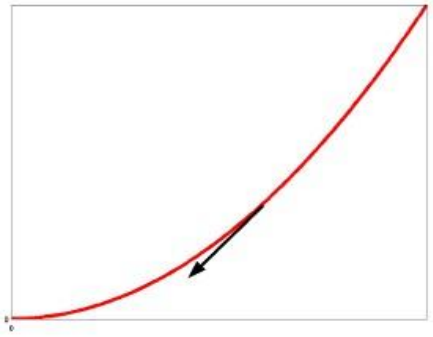
$$\min(-d_w(x_1, x_2) + \infty)$$

We model the “infinity” using a margin resulting in a hinge loss: $\max(0, m - d_w)$

The contrastive loss is :

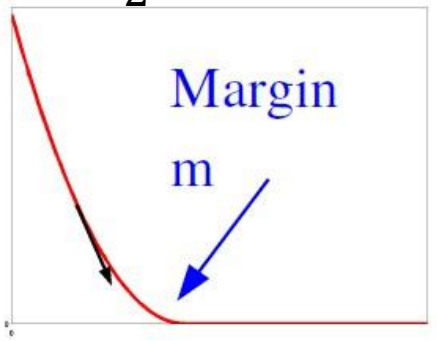
$$L = \frac{1}{2} y d_w + \frac{1}{2}(1 - y)\max(0, m - d_w)$$

minimize $\frac{1}{2} d_w$



Similar images

minimize $\frac{1}{2} max(0, m - d_w)$



Dissimilar images

Image Credit :Learning Hierarchies of *Invariant Features*. Yann LeCun
helper.ipam.ucla.edu/publications/gss2012/gss2012_10739.pdf

SiameseNetwork : Cosine Loss Functions

- Maximize cosine value for positive pairs to reduce the angle in-between
- Reduce the cosine value to be less than a margin value for dissimilar pairs

$$l(x_1, x_2, y) = \begin{cases} max(0, cos(x_1, x_2) - m) & \text{For y =0} \\ 1 - cos(x_1, x_2) & \text{For y=1} \end{cases}$$

Yi Sun, etc. Deep Learning Face Representation by Joint Identification-Verification. NIPS 2014

Draw back of Siamese Architecture

- For hard problems, the network most will settle for a median distance for positive and negative pairs.
- We do not provide the reference of “how much” to increase or decrease
- Loss function of Siamese doesn’t have the global context to which it should work.
- We require more information from a single iteration for effective training.

Triplet Network

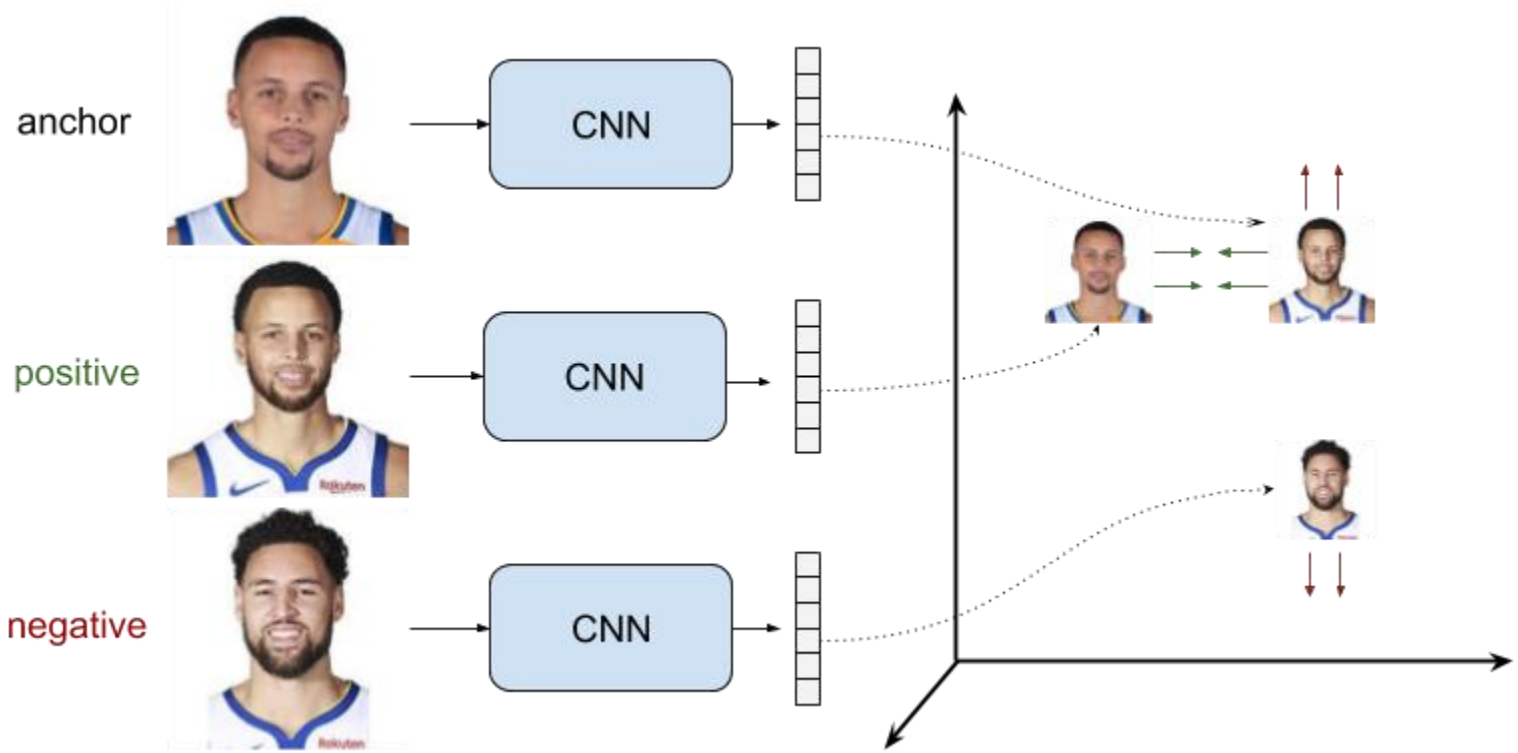
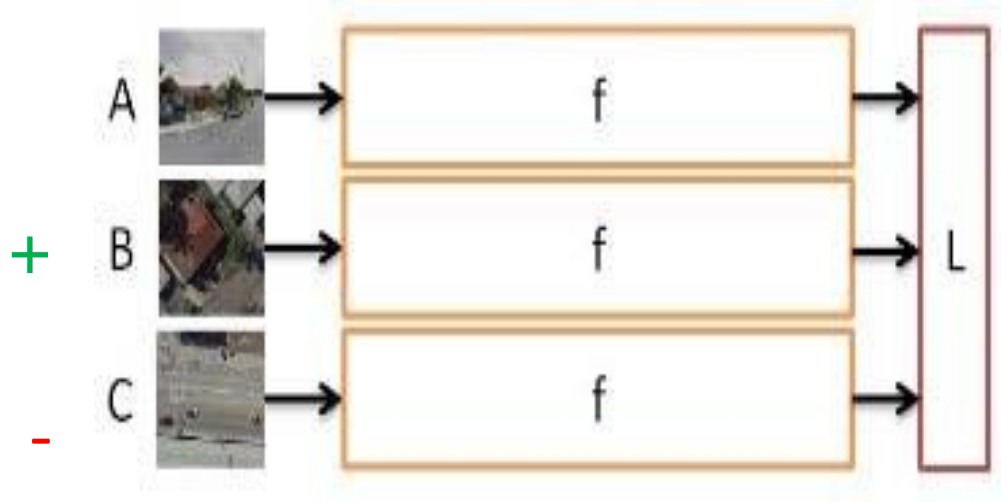


Image Credits: https://gombru.github.io/2019/04/03/ranking_loss/

Triplet Network



- Compare triplets in one go.
- Check if the sample in the **topmost** channel anchor, is more similar to the one in the middle than the one in the bottom.
- Allows us to learn ranking between samples.

Vo, N.N. and Hays, J., 2016, October. Localizing and orienting street views using overhead imagery. In European Conference on Computer Vision (pp. 494-509).

Triplet Network: Loss Function

Condition:

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2$$

$$\forall \left(f(x_i^a), f(x_i^p), f(x_i^n)\right) \in \mathcal{T} \ .$$

Hinge Loss:

$$L = \max\{0, m + D(f(x_i^a), f(x_p^a))) - D(f(x_i^a), f(x_n^a))\}$$

Jiang Wang, etc. Learning Fine-grained Image Similarity with Deep Ranking. CVPR 2014

Sampling in Triplet Network

- Sampling of triplet is a critical aspect in Triplet networks
- Random sampling will most likely corresponds to easy examples
- Hard negative mining is required for learning efficiently.
- We settle for semi hard negative samples to account for robustness to outliers

Triplet Generation

- **Easy Triplets**: d(xa,xn)>d(xa,xp)+m. The negative sample is already sufficiently distant to the anchor sample respect to the positive sample in the embedding space. The loss is 00 and the net parameters are not updated.
- **Hard Triplets**: d(xa,xn)<d(xa,xp). The negative sample is closer to the anchor than the positive. The loss is positive (and greater than mm).
- **Semi-Hard Triplets**: d(xa,xp)<d(xa,xn)<d(xa,xp)+m. The negative sample is more distant to the anchor than the positive, but the distance is not greater than the margin, so the loss is still positive (and smaller than mm).

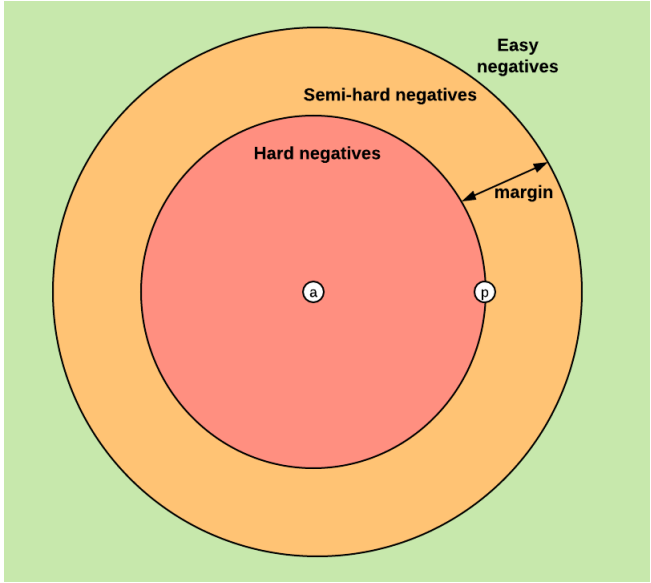


Image Credits: https://gombru.github.io/2019/04/03/ranking_loss/

Few Shot Learning

Why Few Shots:

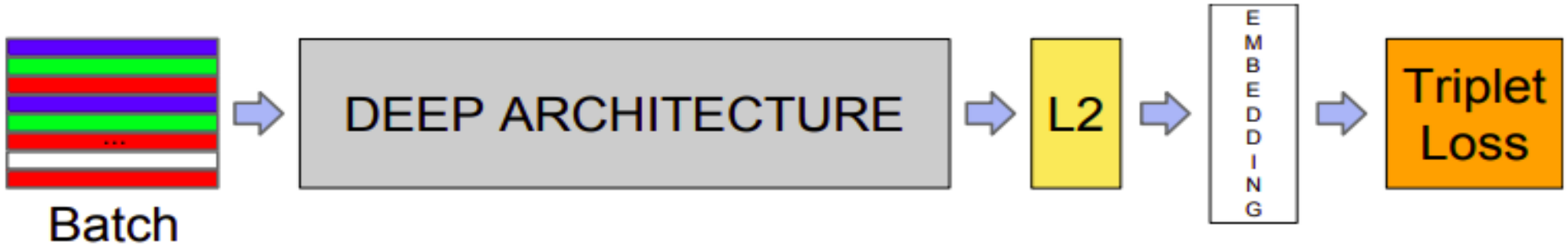
Learn from One or Few training examples (Less Data)

Generalizability with minimum exposure to changes

Image Source: Google, PyImageSearch

Face Recognition with One Shot Learning

Model Structure



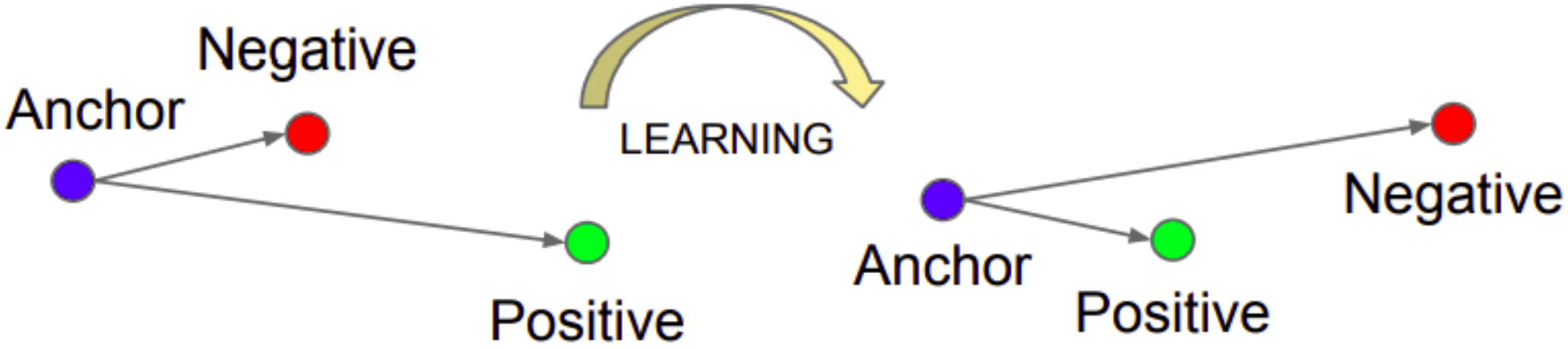


Image Source: Google, PyImageSearch

Triplet Loss Revisited

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2$$

$$\forall \left(f(x_i^a), f(x_i^p), f(x_i^n)\right) \in \mathcal{T} \ .$$

α is a margin that is enforced between positive and negative pairs

Minimization Objective

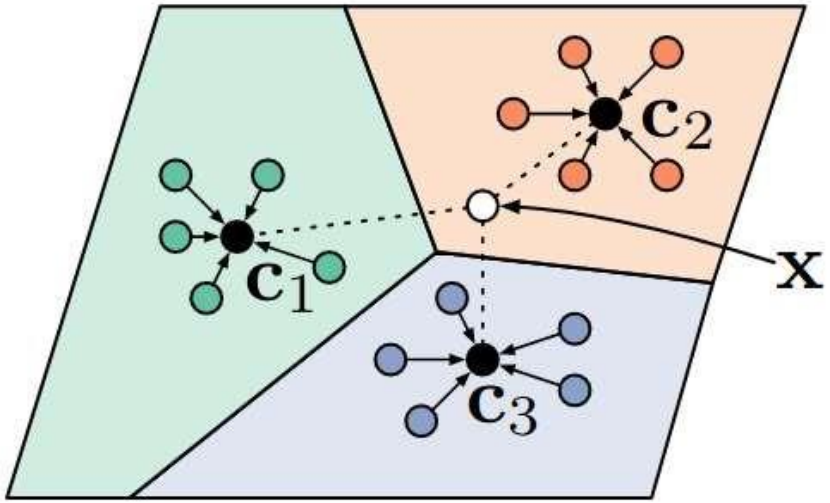
$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\right]$$

Prototypical Networks

- In Prototypical Networks Snell et al. apply a compelling inductive bias in the form of class prototypes to achieve impressive few-shot performance — exceeding Matching Networks without the complication of FCE. The key assumption is made is that there exists an embedding in which samples from each class cluster around a single **prototypical representation** which is simply the mean of the individual samples

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$$

Equation (1) from Prototypical Networks—calculating class prototypes. S_k is the support set belon class k and f_phi is the embedding function.



Class prototypes c_i and query sample x.

Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." *Advances in neural information processing systems*. 2017.

Thank You