

Deep Metric Learning



- 
- The Why and the How of Deep Metric Learning: Diving deep into metric based deep learning.
 - Aakash Agrawal, Jan 11, 2021.
 - <https://towardsdatascience.com/the-why-and-the-how-of-deep-metric-learning-e70e16e199c0>
- 

Table of Contents

- **Prerequisites**
- **A tour of the basics**
 - *. Face verification, Face Recognition
 - *. ANNs (Training phase and Inference phase)
- **Image Classification for face verification?**
 - *. One-shot learning
- **Metric**
- **Metric Learning**
 - *. Mahalanobis Distance Metric
- **Deep Metric Learning**
 - *. Contrastive Loss — Siamese Networks
 - *. Triplet loss — Triplet Networks
 - *. Softmax loss
 - *. A-Softmax loss
 - *. Large Margin Cosine Loss (LMCL)
 - *. Arcface loss
- **References**

Prerequisites

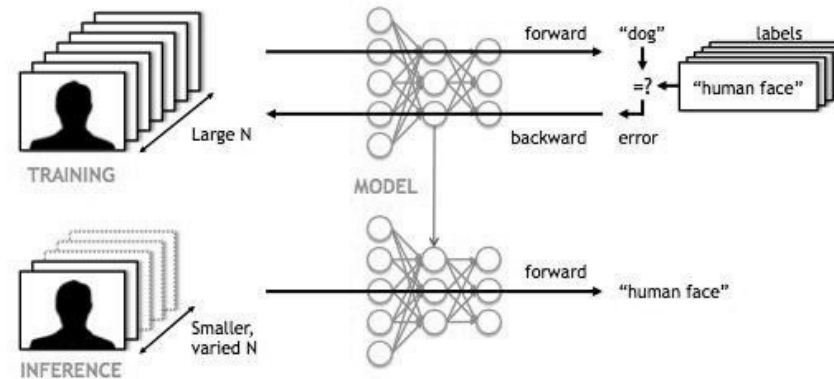
- a good knowledge of linear algebra
- familiarity with the basic concepts in machine learning
- **Keywords:**
 - Metric learning
 - Triplet loss
 - softmax loss
 - Face recognition
 - Face verification
 - DCNN
 - Arcface
 - Spheredface
 - Cosface

A tour of the basics

- Understand a few basic terminologies and establish a solid ground to enhance the understanding of deep metric learning.
- **Face verification** is the task of determining whether a given pair of images belong to the same person or not. In simple words, it is a task where given an image, we try to answer the question. **Is that you?**. A **1:1** authentication problem.
- **Face Recognition**, on the other hand, is a combination of two tasks: **Face identification** and **Face verification**. Face identification is the task of recognizing a person from a database of images. A task where given an image database (let's say a gallery of k persons) we try to answer the question, **who are you?**. So, Face recognition is a **1:k** authentication problem.
- It is quite helpful to know the basic training and inference process of a simple **Artificial neural network**.

Artificial neural network: TRAINING PHASE

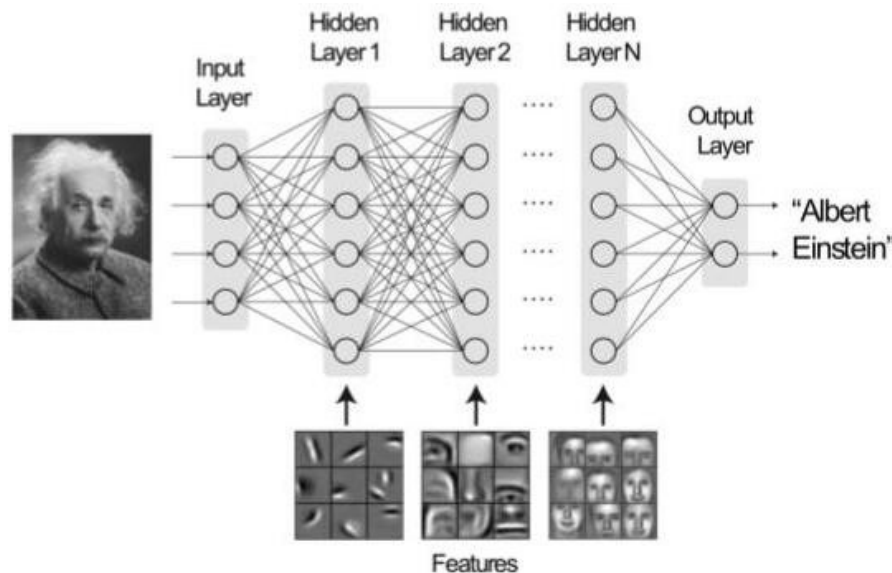
{please skip this part if you are familiar with the topic}.



1. We **randomly initialize** the **weights** and **biases** according to some probability distribution,
2. Feed the data set into the **untrained** neural network architecture.
3. **Forward propagation:** the hidden layer accepts the input data, applies an **activation** function, and passes the activations to the next successive layer. We propagate the results until we get the **predicted result**.
4. After generating the predictions, we calculate the **loss** by comparing the **predicted results** to the ground truth labels.
5. **Backward propagation:** We compute the **gradients** of the loss to weights and biases, and adjust them by subtracting a small quantity proportional to the gradient.
6. We repeat steps 1 to 5 for the entire training set — this is **one epoch**. Repeat for more epochs until eventually our error is minimized or our prediction score is maximized.

Artificial neural network: INFERENCE PHASE

{please skip this part if you are familiar with the topic}.

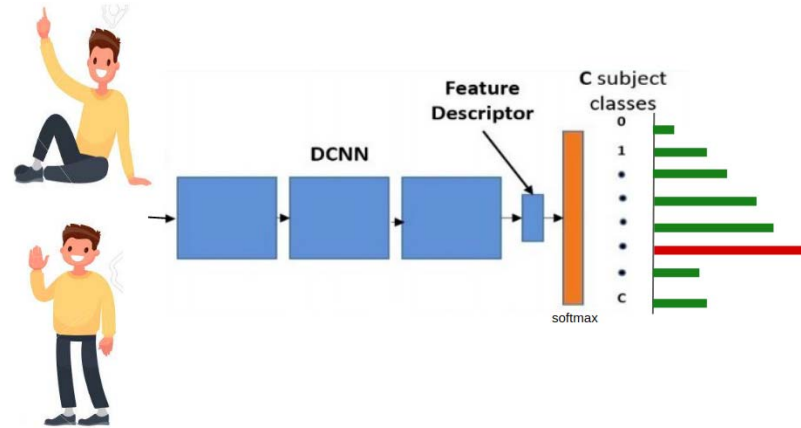


Deep Neural Networks learn Hierarchical Feature Representations. Source: Deep Learning, Journal of NeuroScience.

- In this phase, no calculation of gradients or adjustment of parameters takes place. It uses the same sets of weights and biases for the evaluation on an unknown test dataset which the network hasn't seen before.
- **Deep neural networks** are the go-to models for achieving the state of art performance when it comes to computer vision tasks.
- Learning **invariant** and **discriminative features** from data is the very fundamental goal for achieving good results for any computer vision task.
- Deep learning methods have been proven to be quite effective for **feature learning**, the very reason being the ability of deep learning methods to learn **hierarchical** feature representations by building high-level features from low-level ones.

Artificial neural network: IMAGE CLASSIFICATION

{please skip this part if you are familiar with the topic}.



- **Can we use Image Classification to solve the task of Image Verification?**
- We might be able to learn a quite robust Deep Convolutional Neural network that performs excellently on classifying all the employee images in an organization and also takes into account factors like poses, expression, and illumination.
- But, this is usually achieved when we have a **good amount of data**. By good amount of data, for example 1000's of examples for each class/employee.
- In Image Classification, if the number of **data points per class** is small, it might lead to **overfitting** and yield very poor results.
- Also, **Image Classification generally works well when the number of classes is small**.
- However, It is generally not the case with **Person/Image verification tasks**. In fact, it is quite opposite. Here, we usually have a **very large number of classes and the number of examples per class is quite small**.
- **And this is where one-shot learning comes into the picture.**

One-shot Learning

- **One-shot Learning** is a classification problem that aims to learn about object categories from **one/few** training examples/images. [Wikipedia].
- In simple words, **given just one example/image of a person, you need to recognize him/her.**
- To build a face recognition system, we need to solve this one-shot learning problem.
- **But, deep neural nets usually require vast amounts of data to train on** to excel at a particular task, which is not always available.
- **Deep learning models won't work well with just one training example**, {one-shot learning problem}.
- How to address this issue? We learn a **similarity function**, which helps us to solve the one-shot learning problem.
- Let us start to understand deep metric learning by understanding the basics.

One-shot Learning: METRIC

- A Metric is a **non-negative** function between two points x and y {say $g(x,y)$ } that describes the so-called notion of '**distance**' between these two points.
- There are several properties that a metric must satisfy:
 - **Non-negativity** $\Rightarrow d(x,y) \geq 0$ and $d(x,y) = 0$, iff $x = y$.
 - **Triangular inequality** $\Rightarrow d(x,y) \leq d(x,z) + d(z,y)$.
 - **Symmetry** $\Rightarrow g(x,y) = g(y,x)$.
- EXAMPLES:
 - **A. The Euclidean Metric:** In a ' d ' dimensional vector space, the metric is:

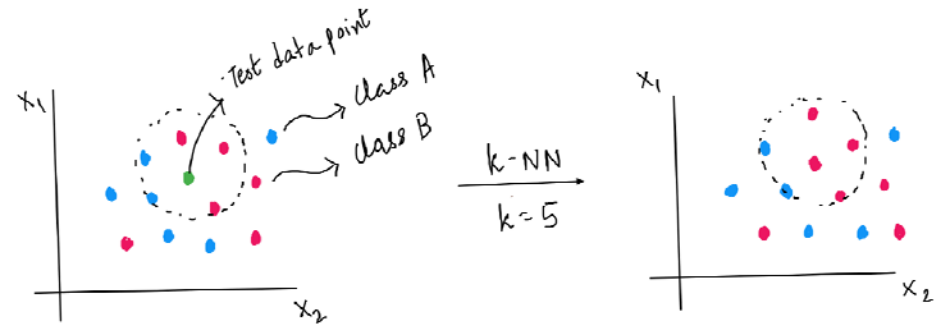
$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2}.$$

- **B. The Discrete Metric:** The metric is given by:

$$d(\mathbf{x}, \mathbf{y}) = \begin{cases} 1 & \text{if } x \neq y \\ 0 & \text{if } x = y. \end{cases}$$

One-shot Learning: METRIC LEARNING (1)

- What does a basic machine learning algorithm do?
- It is the task of learning a distance function over objects.
- In practice, it means that we can train a model that tells a number for any pair of given objects.
- This number should represent a degree or score of similarity between those given objects.
- For example, objects with a score of 0.9 could be more similar than objects with a score of 0.5. Actual scores and their direction could vary among different implementations.
- Given the data and the corresponding output labels, the goal is to come up with a set of **rules** or some **complex function** that maps those inputs to the corresponding output labels.
- One of the simplest machine learning algorithms where **distance** information is captured is the **KNN** (k-Nearest Neighbours) algorithm, where the idea is to find a neighborhood of **k** nearest data points for a new data point and assign this data point to the class to which the majority of the **k** data points belong.



A simple explanation of k-NN

One-shot Learning: METRIC LEARNING (2)

- Similarly, the goal of metric learning is to learn a **similarity function** from data.
- **Metric Learning aims to learn data embeddings/feature vectors in a way that reduces the distance between feature vectors corresponding to faces belonging to the same person and increases the distance between the feature vectors corresponding to different faces.**
- **Euclidean distances** are less meaningful in high dimensions. They fail to capture nonlinearity in data because of a number of reasons:
 - They represent an **isotropic** (same in every direction) distance metric.
 - These distances don't capture the class structure.
- One can use non-isotropic distances that use properties of the data to capture the structure of class relationships. **Mahalanobis Distance Metric** is one such metric.

In a 'd' dimensional space; $x, y \in \mathbb{R}^d$

$$D(x, y) = ((x - y)^T M (x - y))^{1/2}$$

Mahalanobis Distance Metric

One-shot Learning: METRIC LEARNING (3)

Mahalanobis distance

- Here, M is the inverse of the covariance matrix and acts as a weight term to the square Euclidean distance.
- The Mahalanobis distance equation represents a dissimilarity measure between two random vectors $\mathbf{x}$ and $\mathbf{y}$, following the same distribution with Covariance Matrix Σ .

The Covariance Matrix in ' d ' dimensional space:

$$M = \Sigma^{-1} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \dots & \sigma_{1d} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{d1} & \dots & \dots & \sigma_{dd} \end{bmatrix}^{-1}_{d \times d}$$

In a ' d ' dimensional space; $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$

$$D(\mathbf{x}, \mathbf{y}) = ((\mathbf{x} - \mathbf{y})^T M (\mathbf{x} - \mathbf{y}))^{1/2}$$

σ_{ij} is the covariance between variables i, j .
 $\therefore \sigma_{ij} = \sigma_{ji} \Rightarrow M = \text{Symmetric}$

Mahalanobis Distance Metric

M is an inverse of the Covariance Matrix.

One-shot Learning: METRIC LEARNING (4)

Mahalanobis distance

- If $M = I$ (Identity Matrix) we have the special case of Mahalanobis distance — Euclidean Distance, which proves an underlying assumption of the Euclidean Distance that the features are independent of each other.
- The equation(*) can be interpreted as a linear projection in euclidean space. So, W has a linear transformation property because of which Euclidean distance in transformed space is equal to the Mahalanobis distance in the original space.
- This distance metric provides a new data representation in the transformed space which is easily able to distinguish the items of different classes. This Linear transformation shows the inter-class separability power this approach possesses.
- So, Metric Learning is basically a goal to learn this transformation matrix W .

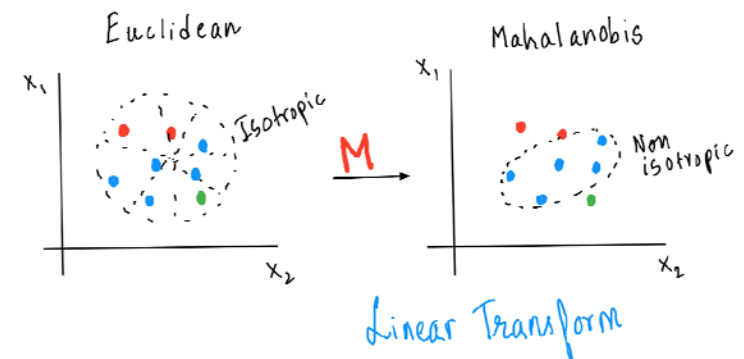
Matrix M can be decomposed as:

$$M = W^T W$$

$\therefore$ Mahalanobis Distance becomes-

$$D(x, y) = ((x - y)^T W^T W (x - y))^{\frac{1}{2}}$$
$$\Rightarrow D(x, y) = ((W(x - y))^T (W(x - y)))^{\frac{1}{2}}$$
$$\Rightarrow D(x, y) = \|Wx - Wy\|_2 \sim eq(x)$$

Linear Transformation property of the Mahalanobis distance metric



Isotropic Euclidean distance V/S Non-isotropic Mahalanobis distance metric

One-shot Learning: METRIC LEARNING (5)

- **Metric learning** is an approach based directly on a distance metric that aims to establish similarity or dissimilarity between images.
- **Deep Metric Learning** on the other hand uses Neural Networks to automatically learn discriminative features from the images and then compute the metric.
- **Why there is a need for deep metric learning?**
 1. Faces of the same person when presented in different poses, expressions, illuminations, etc. might be able to fool a face verification/face recognition system very efficiently.
 2. The task of building an efficient, large scale Face Verification system is radically a **task of designing appropriate loss functions that best discriminate the classes under study**. In recent years, Metric/Distance learning using Deep learning has been shown to output highly satisfying results for many computer vision tasks such as face recognition, face verification, image classification, Anomaly detection, etc.
 3. Metric Learning only has a limited capability to capture non-linearity in the data.
- Deep Metric Learning helps capture Non-Linear feature structure by learning a non-linear transformation of the feature space.

DEEP METRIC LEARNING

There are two ways in which we can leverage deep metric learning for the task of face verification and recognition:

- 1. Designing appropriate loss functions for the problem.**

Most widely used **loss functions** for deep metric learning are **the contrastive loss** and **the triplet loss**.

- 1. Adding a custom layer to an easy to learn classification model.**

DEEP METRIC LEARNING:

1. Designing appropriate loss functions for the problem (1)

** Contrastive Loss — Siamese Networks:

- One of the very fundamental ideas where **explicit metric learning** is performed is **the siamese network model**.
- Siamese network is a Symmetric neural network architecture consisting of two identical subnetworks that share the same sets of parameters (hence computing the same function) and learns by calculating a metric between highest level feature encodings of each subnetwork each with a distinct input.

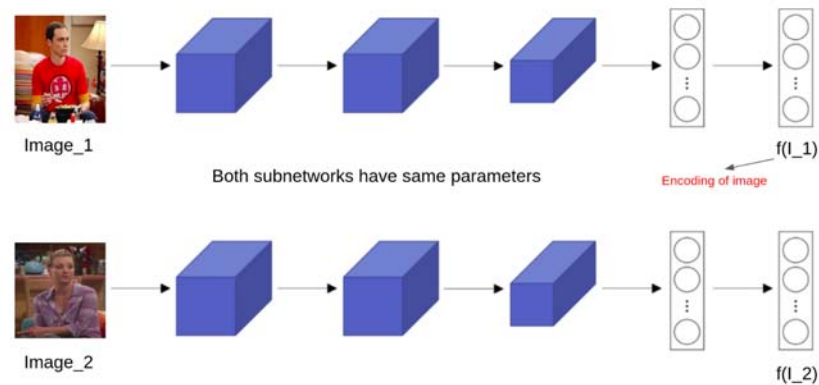


Image by Author: Subnetworks in a siamese net share the same sets of parameters

DEEP METRIC LEARNING:

1. Designing appropriate loss functions for the problem (2)

** Contrastive Loss — Siamese Networks:

- Mathematically, $d(\mathbf{x}, \mathbf{y}) = ||\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})||^2$, Here, “ $\mathbf{x}$ ” and “ $\mathbf{y}$ ” represent the training examples/images. “ $d(.)$ ” represents the metric distance between two images and “ $\mathbf{f}$ ” represents the encoding of the input images. The goal is to learn parameters so that:
 - If $\mathbf{x}$ and $\mathbf{y}$ are the same person — $d(\mathbf{x}, \mathbf{y})$ is **small**,
 - If $\mathbf{x}$ and $\mathbf{y}$ are a different person — $d(\mathbf{x}, \mathbf{y})$ is **large**.
- Siamese Networks are mainly used in combination with a **Contrastive Loss function aka Pairwise ranking Loss**.
- The loss function is mainly used to learn embeddings (feature vectors) in a way that **the metric distance between two examples from the same class is small and that between different classes is large in a metric space**.

DEEP METRIC LEARNING:

1. Designing appropriate loss functions for the problem (3)

The Contrastive loss is given by:

$$L = L(f(x), f(y), z) = z \underbrace{\|f(x) - f(y)\|_2}_{\text{distance bw (+) pairs}} + (1-z) \underbrace{\max(0, m - \|f(x) - f(y)\|_2)}_{\text{distance bw (-) pairs}}$$

Here, $z \in \{0, 1\}$; $z = 0 \Rightarrow x, y$ are positive pairs
 $z = 1 \Rightarrow x, y$ are Negative pairs

Contrastive Loss

** Contrastive Loss — Siamese Networks:

- Now, let's understand the maths.
- Here, $m > 0$ is the margin, z is a boolean which is 1 if x, y are positive pairs and 0 otherwise, $\|\cdot\|$ is the L2 norm, other notations are the same as above.
- Let's understand this equation. If x and y are actually positive pairs, **z is 1**, So, the second term vanishes (**margin m** has no effect). If the network computes a large distance between x and y , the loss will be high as the first term has a significant contribution to the loss.
- Now, if x and y are actually negative pairs, **z is 0**. So, the first term vanishes. If the network computes a small distance (~ 0) between x and y , we will have, **$L = m$** , again a significant contribution to the loss.
- **Role of margin m :** It is to be noted that the representations of negative pairs will only contribute to the loss if the estimated distance $\|f(x) - f(y)\| < m$. Meaning that it will no longer care how far the negative pairs x and y are once this limit reaches. So, it can focus more on the difficult to embed points.
- Note, Siamese Network is mainly used when the number of data points per class is very small — **One-Shot Learning**.

DEEP METRIC LEARNING:

1. Designing appropriate loss functions for the problem (4)

**Triplet Loss — Triplet Networks:

- Triplet network is also a Symmetric neural network architecture but consists of three identical subnetworks that share the same sets of parameters.
- The learning is performed as a set of three images
 - 1) The **Anchor** image (The baseline image),
 - 2) The **positive** image (truth images of a specific person belonging to the same class as an anchor),
 - 3) The **negative** image (images not belonging to the anchor class).
- The goal of triplet loss is to ensure that the anchor/baseline image of a specific person is **closer** to all the positive sets of images (Mathematically, meaning that the euclidean distance between an anchor and a positive image is small) and far away from the negative sets of examples.

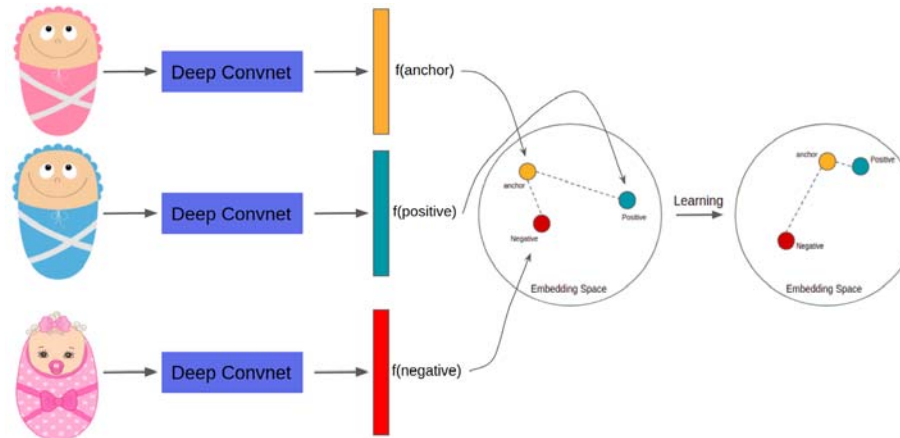


Image by Author: An intuition behind triplet loss

DEEP METRIC LEARNING:

1. Designing appropriate loss functions for the problem (5)

**Triplet Loss — Triplet Networks:

- Here, a = anchor image, p = positive image, n = negative image, m = margin. The embedding is represented by a ' d ' dimensional vector $f(\cdot)$ {the encoding}. Let, $d(a, p) = \|f(a) - f(p)\|$.
- Let's understand this equation by asking when the loss will be 0. The second term inside $\max(\cdot)$ should be < 0 , i.e. $d(a, n) > d(a, p) + m$. If this happens, the triplet network will be quite lucky as there will be no learning. These triplets are known as Easy triplets.
- When the loss will be high? When, $d(a, n) < d(a, p)$, which has a significant contribution to the loss. This will be a hard to train on triplet. Such triplets are known as Hard triplets. Another Case: $d(a, p) < d(a, n) < d(a, p) + m$, Again, the loss will be positive.
- Issues with triplet loss & contrastive loss — These require a time consuming and performance-sensitive pair/triplet mining procedure to train the model.

The Triplet Loss is given by:

$$L = L(a, p, n) = \max(0, m + \|f(a) - f(p)\| - \|f(a) - f(n)\|)$$

Triplet Loss

DEEP METRIC LEARNING:

2. Adding a custom layer to an easy to learn classification model (1):

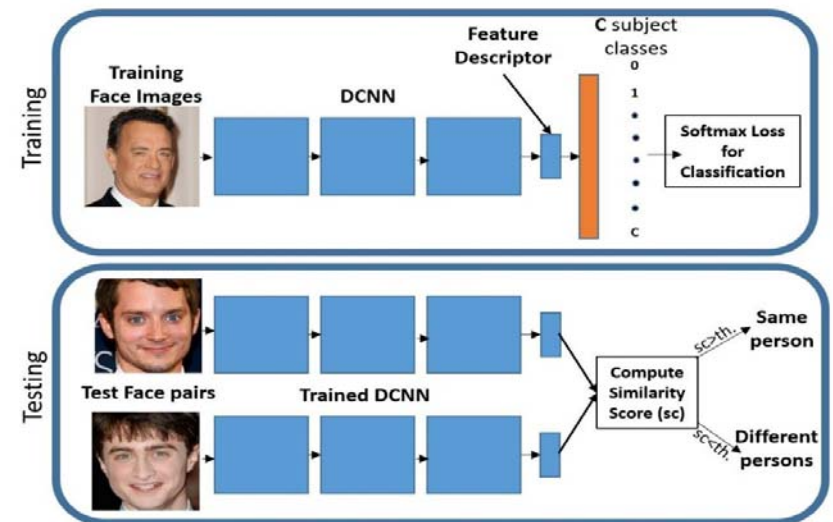
One can achieve distance learning simply by learning the usual image classification problem.

What we need to do is just add an additional layer on top of an easy to learn classification model.

General pipeline for training and testing a Face verification system using DCNN.

1. A DCNN is trained as an image classification task using the ground truth training dataset face images and the corresponding labels.
2. A Softmax loss function is used to optimize the parameters of the DCNN model.
3. At inference time, feature embeddings are extracted for the test face pairs using the trained DCNN.
4. We then compute the cosine similarity between these feature embeddings. If the score is greater than a set threshold, we say that the test pairs are the person and otherwise different.

This is a very simple face verification pipeline without the need for any metric learning.



[source]: arXiv:1703.09507v3 [cs.CV] 7 Jun 2017

- 
- A Beginners Guide to Deep Metric Learning
 - Vijaysinh_Lendave, November 6, 2021
 - <https://analyticsindiamag.com/a-beginners-guide-to-deep-metric-learning/>

Learning the similarity between objects has a dominant role in human cognitive processes and artificial systems for recognition and classification.



Table of Contents

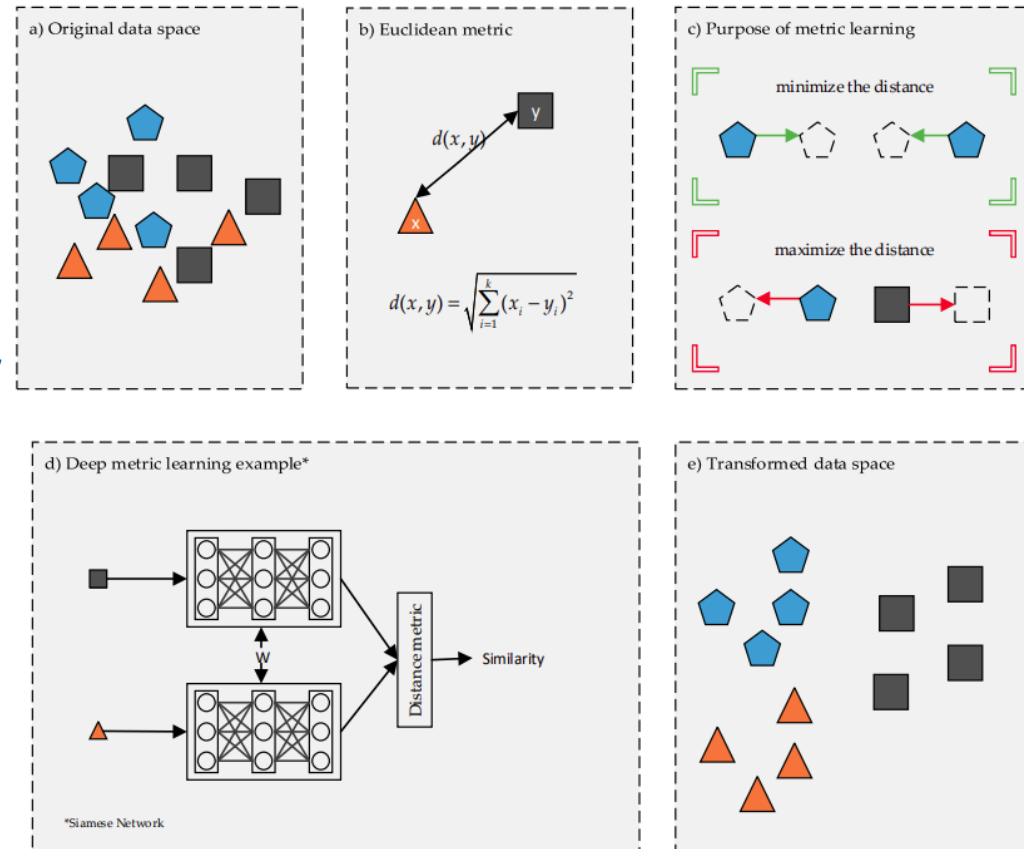
- What is Metric Learning?
- Deep Metric Learning
- Important Points of Deep Metric Learning

What is Metric Learning? (1)

- **Metric learning** is a method of determining similarity or dissimilarity between items based on a distance metric.
- Metric learning seeks to increase the distance between dissimilar things while reducing the distance between similar objects.
- As a result, there are ways that calculate distance information, such as k-nearest neighbours, as well as approaches that transform data into a new representation.
- Deep learning and metric learning have been combined in recent years to create the notion of deep metric learning.
- The principle of similarity between samples underpins deep metric learning.

What is Metric Learning? (2)

- By evaluating data, metric learning creates a new distance metric.
- A metric learning strategy that performs the learning process on data will be able to distinguish the sample data better.
- **Metric learning's main goal is to learn a new metric that will minimize distances between samples of the same class while increasing distances between samples of different classes.**
- While metric learning seeks to bring similar things closer together, it increases the gap between dissimilar objects.



What is Metric Learning? (3)

- When it comes to metric learning studies in the literature, it's clear that they're all linked to Mahalanobis' distance metric.
- The training samples are $X=[x_1, x_2, \dots, x_N] \in R^{d \times N}$ where $x_i \in R^d$ is the i th training example and N is the total number of training samples.
- The following formula is used to compute the distance between x_i and x_j :

$$d_M(x_i, x_j) = \sqrt{(x_i - x_j)^T M (x_i - x_j)}$$

Data for Metric Learning

- Unlike classifiers, a metric learning training does not require specific class labels.
- All that is required are examples of similar and dissimilar objects.
- We would call them positive and negative samples.
- At the same time, it could be a relative similarity between a pair of objects.
- For example, twins look more alike to each other than a pair of random people.
- And random people are more similar to each other than a man and a cat.
- A model can use such relative examples for learning.

Training the Model

- There are several ways to train a metric learning model.
- Among the most popular is the use of Triplet or Contrastive loss functions.

Deep Metric Learning (1): Introduction

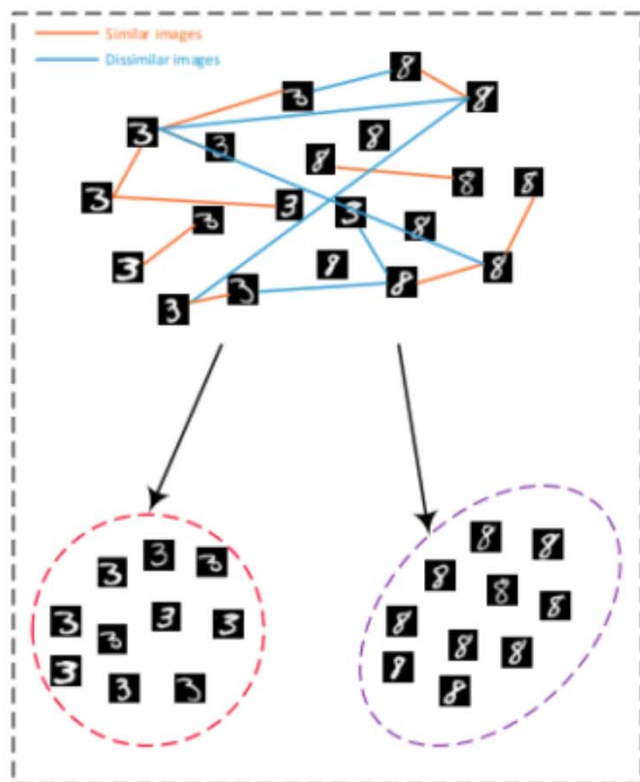
- Deep metric learning aims to train a similarity metric that uses samples to compute the similarity or dissimilarity of two or more objects.
- The Euclidean, Mahalanobis, Matusita, Bhattacharyya, and Kullback-Leibler distances are basic similarity metrics used for data classification.
- These pre-defined measures, on the other hand, offer restricted data classification capabilities.
- To address this challenge, a [strategy based on the Mahalanobis metric was developed to categorize the data into traditional metric learning](#).

Deep Metric Learning (2): Introduction

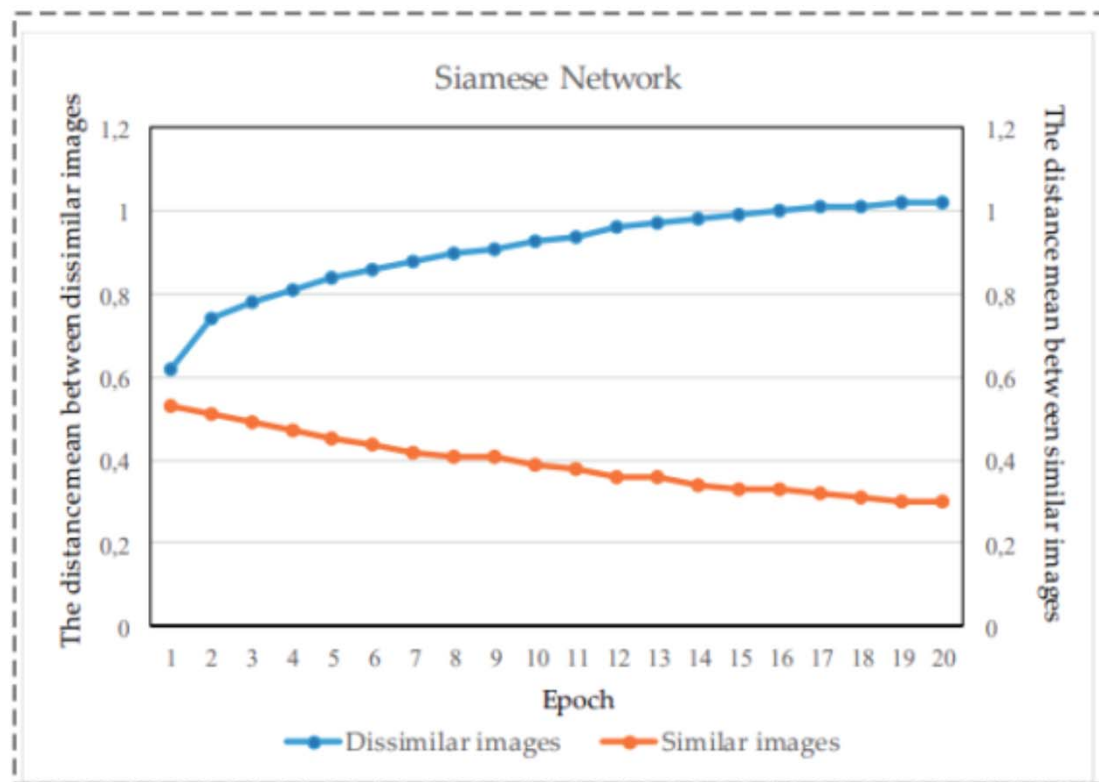
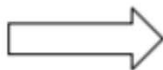
- Deep metric learning provides problem-based solutions that are caused by learning from raw data by utilizing deep architectures and finding embedded feature similarity through nonlinear subspace learning.
- When it comes to the extent of deep metric learning, it includes everything from :
 - video comprehension to human re-identification,
 - medical difficulties,
 - three-dimensional (3D) modelling,
 - face verification and recognition, and
 - signature verification.

Deep Metric Learning? (3)

The metric loss function has been used in deep learning to carry out this process. It pushes samples from different classes apart from each other while striving to bring examples from the same classes closer together



(a)



(b)

Deep Metric Learning (4)

- Some experiments on the MNIST picture collection were conducted while utilizing contrastive loss to show this approach
- The graph shows the distance values reflect the average of distances between similar and different images.
- The distance value for similar photos dropped step by step after each epoch,
- The distance value for different photos, on the other hand, increased at the same time.
- For comparable or different images, the Siamese network distance relationship has been successfully used in each epoch (Figure b).
- This experiment demonstrates that the approach's goal can be achieved successfully.

Important Points of Deep Metric Learning

- The three basic components of deep metric learning:
 - Informed input samples,
 - the topology of the network model, and
 - a metric loss function
- Although deep metric learning focuses on metric loss functions, informative sample selection is also critical in classification and clustering.
- In the contrastive loss, the simplest technique to determine train samples is to use randomly generated positive or negative pairs of objects.

References

- Aakash Agrawal, The Why and the How of Deep Metric Learning: Diving deep into metric based deep learning, Jan 11, 2021.
<https://towardsdatascience.com/the-why-and-the-how-of-deep-metric-learning-e70e16e199c0>
- Vasnetsov Andrey, Metric Learning Tips & Tricks: How to train object matching model with no labeled data and use it in production, May 15, 2021,
<https://towardsdatascience.com/metric-learning-tips-n-tricks-2e4cf6e6b75b>
- Vijaysinh_Lendave, A Beginners Guide to Deep Metric Learning, November 6, 2021, <https://analyticsindiamag.com/a-beginners-guide-to-deep-metric-learning/>