Multi-view Linear Discriminant Analysis Network (MvLDAN)

Sawan Patidar (0801CS171070)

December 13, 2020

Contents

Chapter 1

Introduction

1.1 CCA:

Canonical correlation analysis finds collections linear combinations of pairs of multivariate data vectors that are maximally correlated with each other, and uncorrelated with other pairs. Simultaneously observed data pairs can be analyzed to diagnose relationships between the pairs, and CCA on time-lagged pairs can be used for forecasting. MCA is a related method that maximizes covariance.

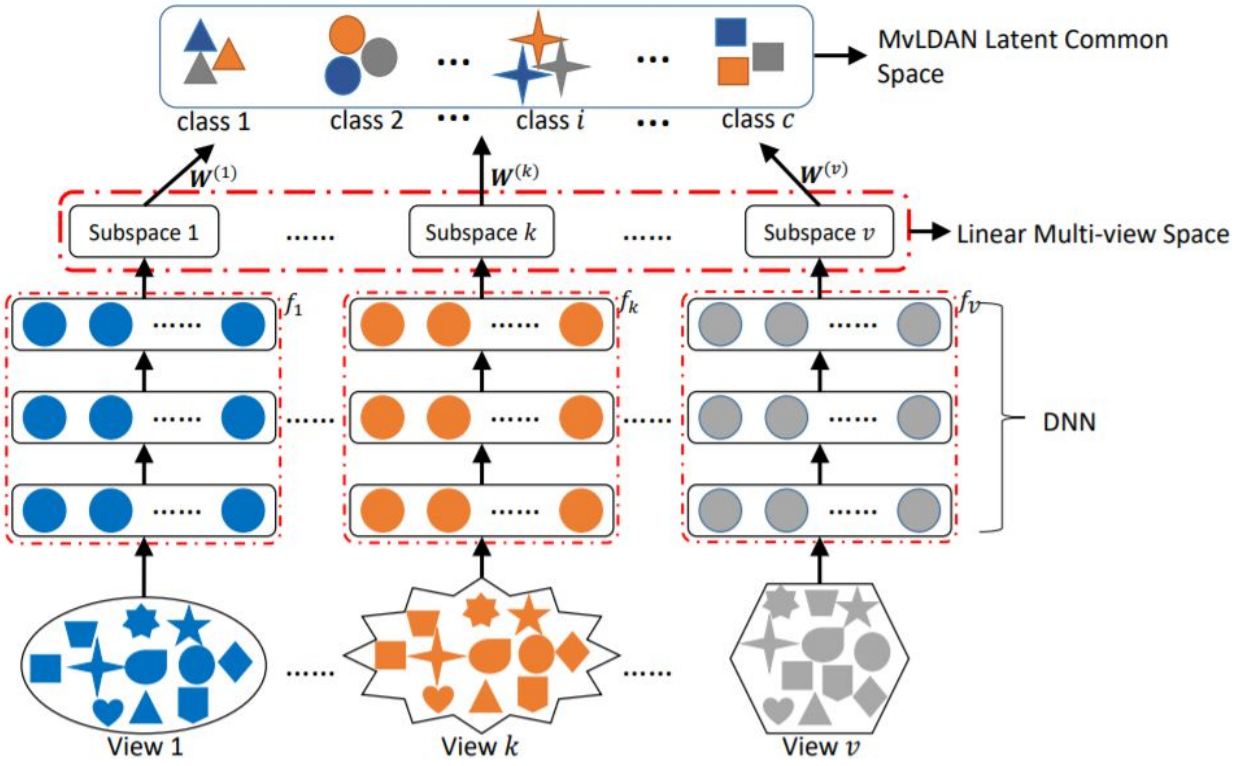
1.2 LDA

Linear Discriminant Analysis or Normal Discriminant Analysis or Discriminant Function Analysis is a dimensionality reduction technique which is commonly used for the supervised classification problems. It is used for modeling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.

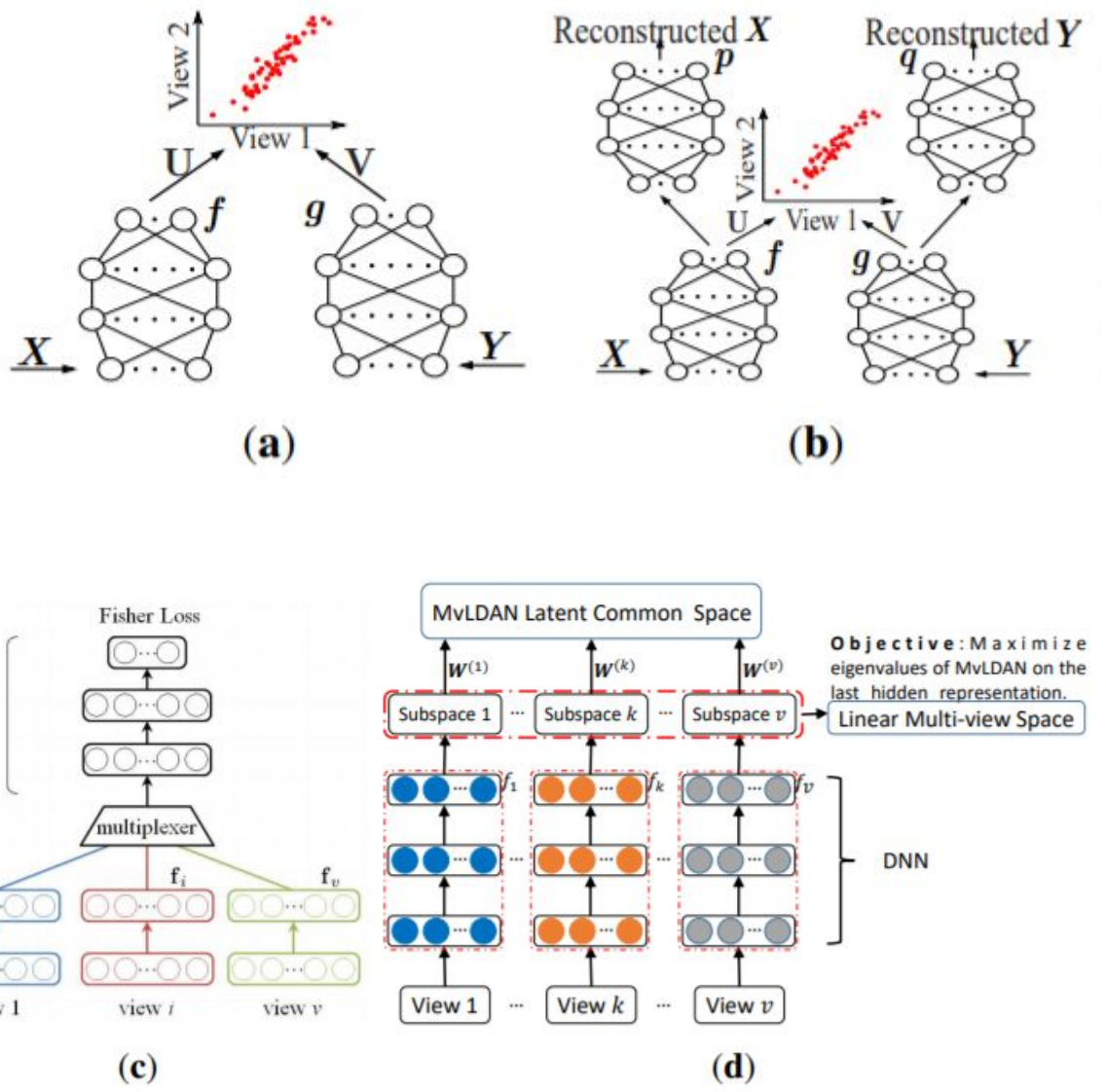
1.3 MvLDAN

To eliminate the complicated (usually highly nonlinear) view discrepancy for favorable cross-view recognition and retrieval, Multi-view Linear Discriminant Analysis Network (MvLDAN) seeks a nonlinear discriminant and view-invariant representation shared among multiple views. Unlike existing multi-view methods which directly learn a common space to reduce the view gap, MvLDAN employs multiple feedforward neural networks (one for each view) and a novel eigenvalue-based multi-view objective function to encapsulate as much discriminative variance as possible into all the available common feature dimensions. With the proposed objective function, the MvLDAN could produce representations possessing:

a) low variance within the same class regardless of view discrepancy,
b) high variance between different classes regardless of view discrepancy,
c) high covariance between any two views.



The above figure shows the Framework of MvLDAN, where different shapes denotes different classes, and different colors represent different views.



The above figure shows the Schematic diagram of DNN-based multi-view learning models. Fig. (a) shows the network model of DCCA. Fig. (b) shows the network model of DCCAE . Fig. (c) shows the network model of MvDN . Fig. (d) shows the network model of MvLDAN. In the figures, X and Y represent different views; f, g, gc and fi are the corresponding sub-networks; U, V and W(k) are the view-specific transforms.

Chapter 2

Mathematical Formulation

2.1 Formulation

Let $\boldsymbol{X}^{(k)} = \{\boldsymbol{x}_{ij}^{(k)} \in \mathbb{R}^{d_k \times 1} | i = 1, \cdots, c; j = 1, \cdots, N_i^{(k)}; k = 1, \cdots, v\}$ be the samples from the k-th view, where $\boldsymbol{x}_{ij}^{(k)}$ denotes the j-th sample from the k-th view of the i-th class with dk dimensionality, c is the number of classes and $N_i^{(k)}$ is the number of samples from the k-th view of the i-th class. As shown in Fig. 1, the sub-network for the k-th view can be represented as a nonlinear function $f_k(\cdot; \theta_{f_k}) \in \mathbb{R}^{p \times 1}$, where θ_{f_k} is the parameter of the k-th view. Therefore, the output of the k-th view is formulated as

$$\boldsymbol{y}_{ij}^{(k)} = f_k(\boldsymbol{x}_{ij}^{(k)}) \quad \text{--- Eq.1}$$

for the sample $\boldsymbol{x}_{ij}^{(k)}$. The proposed method seeks to find a set of linear transformations $\{\boldsymbol{W}^{(1)}, \cdots, \boldsymbol{W}^{(k)}, \cdots, \boldsymbol{W}^{(v)}\}$ to project the samples from v views into a latent common space, where $\boldsymbol{W}^{(k)} \in \mathbb{R}^{p \times q}$ denotes the mapping of the i-th view, and q = c − 1 (since Sb is of rank c − 1 at most).

For ease of representation, we denote the projection results by

$$\boldsymbol{Z} = \{\boldsymbol{z}_{ij}^{(k)} = \boldsymbol{W}^{(k)^T} \boldsymbol{y}_{ij}^{(k)} | i = 1, \cdots, c; j = 1, \cdots, N_i^{(k)}; k = 1, \cdots, v\}$$

Unlike the traditional LDA-like methods we use the pairwise-view information to further eliminate the view discrepancy. Mathematically, the objective function is formulated as:

$$\underset{\substack{f_1, f_2, \cdots, f_v \\ \boldsymbol{W}^{(1)}, \cdots, \boldsymbol{W}^{(v)}}}{\arg\max} \frac{\mathrm{Tr}(\boldsymbol{S}_b) + \lambda \mathrm{Tr}(\boldsymbol{S}_c)}{\mathrm{Tr}(\boldsymbol{S}_w) + \beta \sum_{k=1}^{v} \|\boldsymbol{W}^{(k)}\|_F^2}, \quad \text{--- Eq. 2}$$

where Tr(·) is the trace operator, S_b and S_w correspond to the between-class and within-class scatter matrices from the classes of all views, S_c is the pairwise-view covariance matrix over all views.

Moreover, the between-class scatter S_b, within-class scatter matrix S_w, and the covariance matrix S_c are defined by:

$$\boldsymbol{S}_b = \sum_{i=1}^{c} N_i(\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T,$$

$$\boldsymbol{S}_w = \sum_{i=1}^{c}\sum_{k=1}^{v}\sum_{j=1}^{N_i^{(k)}}(\boldsymbol{z}_{ij}^{(k)} - \boldsymbol{\mu}_i)(\boldsymbol{z}_{ij}^{(k)} - \boldsymbol{\mu}_i)^T,$$

$$\boldsymbol{S}_c = \sum_{k<l}^{v} \boldsymbol{Z}^{(k)}\boldsymbol{Z}^{(l)^T},$$

---- Eq 3,4,5

Where $\boldsymbol{\mu}_i = \frac{1}{N_i^{(k)}}\sum_{k=1}^{v}\sum_{j=1}^{N_i^{(k)}} \boldsymbol{z}_{ij}^{(k)}$ is the mean of all samples in the i-th class from the k-th view, $\mu = \frac{1}{N}\sum_{k=1}^{v}\sum_{i=1}^{c}\sum_{j=1}^{N_i^{(k)}} \boldsymbol{z}_{ij}^{(k)}$ is the mean of all samples from all views.

The between-class scatter matrix in the common space can be further written as

$$S_b = \boldsymbol{W}^T \begin{bmatrix} \boldsymbol{B}_{11} & \cdots & \boldsymbol{B}_{1v} \\ \vdots & \ddots & \vdots \\ \boldsymbol{B}_{v1} & \cdots & \boldsymbol{B}_{vv} \end{bmatrix} \boldsymbol{W} = \boldsymbol{W}^T\boldsymbol{B}\boldsymbol{W},$$

--- Eq. 6

where $\boldsymbol{W} = \left[\boldsymbol{W}^{(1)^T}\ \boldsymbol{W}^{(2)^T}\ \cdots\ \boldsymbol{W}^{(v)^T}\right]^T$ is constructed from the transformation matrices for all views and B_{kl} is defined as:

$$\boldsymbol{B}_{kl} = \sum_{i=1}^{c} \frac{1}{N_i}\boldsymbol{s}_i^{(k)}\boldsymbol{s}_i^{(l)^T} - \frac{1}{N}\boldsymbol{s}^{(k)}\boldsymbol{s}^{(l)^T},$$

--- Eq. 7

Where $\boldsymbol{s}_i^{(k)} = \sum_{j=1}^{N_i^{(k)}} \boldsymbol{y}_{ij}^{(k)}$ is the sum of all samples in the i-th class from the k-th view; $\boldsymbol{s}^{(k)} = \sum_{i=1}^{c} \boldsymbol{s}_i^{(k)}$ is the sum of all samples from the k-th view.
Similarly other equations can be re written as $S_w = W^TCW$ and $S_C = W^TDW$, where C and D are partitioned matrices like B, the (k,l)th submatrices of C and D and are defined as:

$$\boldsymbol{C}_{kl} = \sum_{i=1}^{c}\left((k == l)\sum_{j=1}^{N_i^{(k)}}\boldsymbol{y}_{ij}^{(k)}\boldsymbol{y}_{ij}^{(l)^T} - \frac{1}{N_i}\boldsymbol{s}_i^{(k)}\boldsymbol{s}_i^{(l)^T}\right)$$

$$\boldsymbol{D}_{kl} = \begin{cases} \boldsymbol{0} & k = l \\ \boldsymbol{Y}^{(k)}\boldsymbol{Y}^{(l)^T} & k \neq l \end{cases},$$

--- Eq. 8,9

where k == l is a Boolean equation, whose value is 1 if k = l and 0 otherwise; 0 is a zero matrix. Thus, our objective function can be rewritten as

$$\underset{\substack{f_1,f_2,\cdots,f_v \\ \boldsymbol{W}^{(1)},\cdots,\boldsymbol{W}^{(v)}}}{\arg\max} \frac{\mathrm{Tr}\left(\boldsymbol{W}^T\left(\boldsymbol{B}+\lambda\boldsymbol{D}\right)\boldsymbol{W}\right)}{\mathrm{Tr}\left(\boldsymbol{W}^T\left(\boldsymbol{C}+\beta\boldsymbol{I}\right)\boldsymbol{W}\right)},$$

--- Eq. 10

where I denotes the identity matrix.

The equation can be relaxed as the following tractable ratio trace (determinant ratio) form:

$$\underset{\substack{f_1,f_2,\cdots,f_v \\ \boldsymbol{W}^{(1)},\cdots,\boldsymbol{W}^{(v)}}}{\arg\max} \mathrm{Tr}\left(\left(\boldsymbol{S}_w+\beta\boldsymbol{W}^T\boldsymbol{W}\right)^{-1}\left(\boldsymbol{S}_b+\lambda\boldsymbol{S}_c\right)\right) = \underset{\substack{f_1,f_2,\cdots,f_v \\ \boldsymbol{W}^{(1)},\cdots,\boldsymbol{W}^{(v)}}}{\arg\max} \frac{\left|\boldsymbol{W}^T\left(\boldsymbol{B}+\lambda\boldsymbol{D}\right)\boldsymbol{W}\right|}{\left|\boldsymbol{W}^T\left(\boldsymbol{C}+\beta\boldsymbol{I}\right)\boldsymbol{W}\right|},$$

--- Eq. 11

where | · | is the determinant operator. The above eq is s equivalent to the following generalized eigenvalue decomposition (GED) problem

$$\left(\boldsymbol{B}+\lambda\boldsymbol{D}\right)\boldsymbol{w}_i=\gamma_i\left(\boldsymbol{C}+\beta\boldsymbol{I}\right)\boldsymbol{w}_i,$$

--- Eq. 12

Where $\gamma_i|_{i=1}^{c-1}$ is the i-th largest eigenvalue of the GED with the corresponding eigenvector wi , and wi constitutes the i-th column vector of the matrix W.

Moreover $\gamma_i|_{i=1}^{c-1}$ and $\boldsymbol{w}_i|_{i=1}^{c-1}$ are the eigenvalues and eigenvectors of $\left(\boldsymbol{C}+\beta\boldsymbol{I}\right)^{-1}\left(\boldsymbol{B}+\lambda\boldsymbol{D}\right)$, respectively.

Theorem 1: : Let γi be the i-th largest eigenvalue of the GED Eq given below , then the DNN's $\boldsymbol{f}^*=\{f_1^*,f_2^*,\cdots,f_v^*\}$ learnt by MvLDAN will be the solution to Eq 11.

$$\underset{f_1,f_2,\cdots,f_v}{\arg\max} \mathrm{Tr}\left(\left(\boldsymbol{S}_w+\beta\boldsymbol{W}^T\boldsymbol{W}\right)^{-1}\left(\boldsymbol{S}_b+\lambda\boldsymbol{S}_c\right)\right).$$

---Eq. 13

Equivalently, $\boldsymbol{f}^*$ is also solution to

$$\underset{f_1,f_2,\cdots,f_v}{\arg\max} \sum_{i=1}^{c-1}\gamma_i.$$

--- Eq. 14

A certain threshold is used to filter the relatively larger eigenvalues when optimizing the parameters of the DNNs, i.e. we just consider the m eigenvalues that do not exceed the threshold for variance maximization:

$$\arg\min_{f_1,f_2,\cdots,f_v} -\frac{1}{m}\sum_{i=1}^{\cdots}\gamma_i \quad \text{with}$$
$$\{\gamma_1,\cdots,\gamma_m\} = \{\gamma_i|\gamma_i < \min\{\gamma_1,\cdots,\gamma_{c-1}\} + \epsilon\}.$$

--- Eq. 19

2.2 Optimisation

1) Feedforward and calculate the loss: For the k-th view, the samples of a batch of $X^{(k)}$ are fed forward to the MvLDAN as in Eq. (1), and the output of the MvLDAN is denoted as $Y^{(k)}$, with $\boldsymbol{Y}^{(k)} = \left\{\boldsymbol{y}_{ij}^{(k)}|i=1,\cdots,c;j=1,\cdots,N_b\right\}$ Where N_b is the batch size. The loss of the whole network is calculated as in Eq. (19), denoted as $\mathcal{J} = -\frac{1}{m}\sum_{i=1}^{m}\gamma_i$.

2) Gradient of the loss layer: As the loss J is directly calculated with the outputs of the network, there is no parameter involved. We firstly need to calculate the gradient of $\mathcal{J}$ with respect to $Y^{(k)}$. If $\boldsymbol{w}_i^T(\boldsymbol{C}+\beta\boldsymbol{I})\boldsymbol{w}_i = 1$, then,

$$\frac{\partial\gamma_i}{\partial\boldsymbol{Y}^{(k)}} = \boldsymbol{w}_i^T\left(\frac{\partial(\boldsymbol{B}+\lambda\boldsymbol{D})}{\partial\boldsymbol{Y}^{(k)}} - \gamma_i\frac{\partial(\boldsymbol{C}+\beta\boldsymbol{I})}{\partial\boldsymbol{Y}^{(k)}}\right)\boldsymbol{w}_i.$$

--- Eq. 20

Therefore the gradient of $\mathcal{J}$ w.r.t. $Y^{(k)}$ can be calculated as follows

$$\frac{\partial\mathcal{J}}{\partial\boldsymbol{Y}^{(k)}} = \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{w}_i^T\left(\frac{\partial(\boldsymbol{B}+\lambda\boldsymbol{D})}{\partial\boldsymbol{Y}^{(k)}} - \gamma_i\frac{\partial(\boldsymbol{C}+\beta\boldsymbol{I})}{\partial\boldsymbol{Y}^{(k)}}\right)\boldsymbol{w}_i.$$

--- Eq. 21

3. MvLDAN update via gradient descent: Let us denote the parameters of the k-th sub-network as θ_{f_k}, then θ_{f_k} is updated by descending the stochastic gradient as follows:

$$\nabla_{\theta_{f_k}}\mathcal{J} = \frac{\partial\mathcal{J}}{\partial\boldsymbol{Y}^{(k)}}\cdot\frac{\partial\boldsymbol{Y}^{(k)}}{\partial f_k}.$$

--- Eq. 22

Chapter 3

Algorithm

This section we will come to know about the algorithm used in Multi-view Linear Discriminant Analysis Network.

3.1 MvLDAN Algorithm:

Algorithm 1 Optimization procedure of MvLDAN

Input: Training set $\mathcal{X} = \left\{\boldsymbol{X}^{(k)}\right\}_{k=1}^{v}$, the corresponding class label set, batch size N_b, and learning rate α.

1: **while** not converge **do**
2: Randomly select N_b samples for each view from $\mathcal{X}$ to construct a multi-view mini-batch.
3: Compute the outputs of each view-specific sub-network on the multi-view mini-batch according to Eq. (1) as $\boldsymbol{y}_{ij}^{(k)} = f_k(\boldsymbol{x}_{ij}^{(k)})$.
4: Compute $\boldsymbol{B}$, $\boldsymbol{D}$ and $\boldsymbol{C}$ on the obtained outputs according to Eq. (7), Eq. (8) and Eq. (9), respectively.
5: Compute the eigenvalues according to Eq. (12).
6: Update the parameters of each sub-network by minimizing $\mathcal{J}$ in Eq. (19) by descending their stochastic gradient according to Eq. (22) as $\theta_{f_k} = \theta_{f_k} - \alpha\nabla_{\theta_{f_k}}\mathcal{J}$.
7: **end while**
8: Compute the transformations $\left\{\boldsymbol{W}^{(k)}\right\}_{k=1}^{v}$ on $\mathcal{X}$ according to Eq. (1), Eq. (7), Eq. (8), Eq. (9) and Eq. (12).

Output: Optimized MvLDAN model.

Chapter 4

Documentation of API

4.1 Package Organisation:

from MvLDAN import MvLDAN_general

loss_out = Lambda(MvLDAN_gneral, output_shape=(1,), name='ctc')(net_output + net_labels)

4.2 Methods:

load_noisyMNIST():

Load the dataset in training, validating and testing sets.

create_mnist_full_model((input_size, output_size, value_l2, learning_rate)

Create a model from the mnist dataset:

Parameters:

input_size = ndarray(n*p)

Where p is feature and n is number of samples

output_size = ndarray(n*p)

Where p is feature and n is number of samples

Value_l2 = constant with value 1e-5

learning_rate = constant with value 1e-3

th_MvLDAN_Sw_Sb(data, labels):

Method to find out the within and between scatter matrix.

Parameters:
data = training data (50000 inputs)
labels = labels associated with training data

th_MvLDAN(data_inputs, labels):

Method returns the mean, standard deviation, the proposed set of linear transformations and the eigan values.

Parameters:
data = training data (50000 inputs)
labels = labels associated with training data

Chapter 5

Learning Outcomes

- Learned about CCA and LDA
- Feedforward Neural Networks.
- Limitations of various unsupervised and supervised approaches and advantages of MvLDAN over them.
- Eliminating various complicated view discrepancy for favourable cross view recognition and retrieval.
- How to study a research paper

A. References

- Deep Learning: Feedforward Neural Network https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7
- Multi-view Linear Discriminant Analysis Network https://ieeexplore.ieee.org/document/8704986