AARHUS
UNIVERSITET

Computer Vision & Machine Learning

Alexandros Iosifidis
@
Department of Electrical and Computer Engineering
Aarhus University

Image Classification

In standard (sample-based) Supervised Learning:

- We usually assume that each sample belongs to one category only

Camera

101 Object Categories dataset

A collection of other object datasets

Image Classification

In standard (sample-based) Supervised Learning:

- We usually assume that each sample belongs to one category only

Beaver

[101 Object Categories dataset]

[A collection of other object datasets]

Image Classification

For the above examples, standard image-based classification schemes that:

- Describe the image using local image descriptions (e.g. SIFT descriptors calculated in local neighborhoods of Interest Points)

- Represent the image using a feature vector (e.g. using the Bag of Words representation scheme)

can be used since the assumption that the image representation involves only descriptors of the (correct) class is true

More examples

What is the class of these images?

VOC 2005 Database

More examples

This is an image of class 'person'

What about this chair?

There is also a big table here

VOC 2005 Database

Multiple Instance Learning

In Multiple Instance Learning (MIL):

- Each sample (image) is followed by a label

- Each sample is represented by a set of feature vectors (e.g. SIFT vectors) called instances

- Not all instances describing a sample convey information related to the class of the sample (note that at least one of them must belong to the class of the sample label!)

We say that each sample is represented by a ‘bag of instances’.

Using this terminology, we can define several ML problems

Multiple Instance Learning

Using this terminology, we can define several problems:

- **Multiple Instance-based Classification**

- Multiple Instance-based Regression

- Multiple Instance-based Clustering

We will follow the taxonomy of:

J. Amores, “Multiple Instance Classification: review, taxonomy and comparative study”, Artificial Intelligence, 2013

Multiple Instance Learning

Notations:

- An image is represented by a bag (set) of N feature vectors $X = \{\vec{x}_1, \ldots, \vec{x}_N\}$ where $\vec{x}_i$ is the i-th instance of that bag.

- The number of instances representing each image may vary (different bags contain different number of vectors)

- All instances (of all images) are d-dimensional vectors (which define the instance space) $\vec{x}_i \in \mathbb{R}^d$

- We want to define (learn) a decision (classification) function $F(X) \in [0, 1]$ that can decide if a new/unknown image belongs to the positive class or not (for multi-class classification problems we use the One-versus-Rest scheme)

Multiple Instance Learning

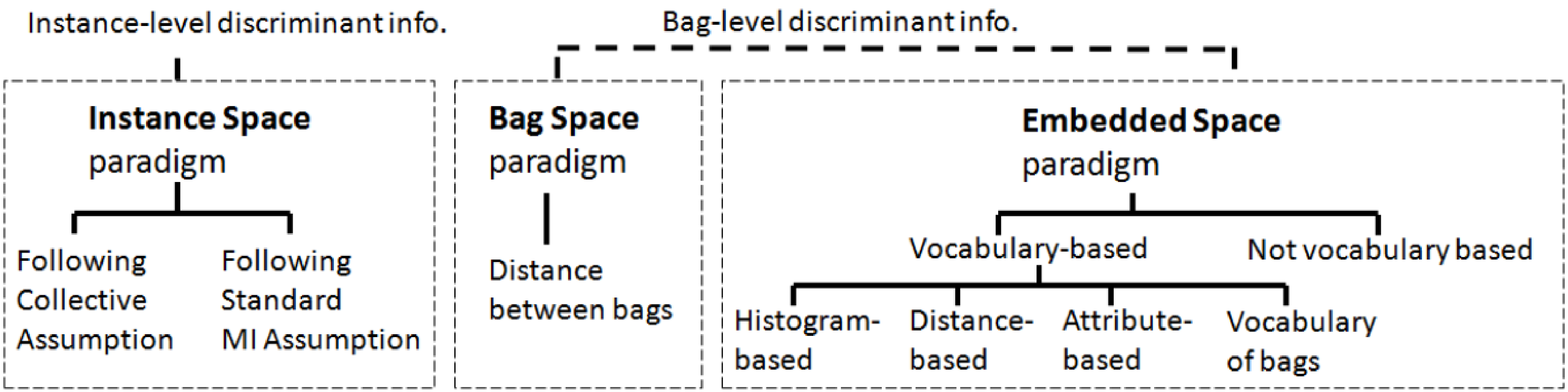
Notations:

- We want to define (learn) a decision (classification) function $F(X) \in [0, 1]$ that ca$\vec{x}_i$ decide if a new/unknown image belongs to the positive class or not (for multi-class classification problems we use the One-versus-Rest scheme)

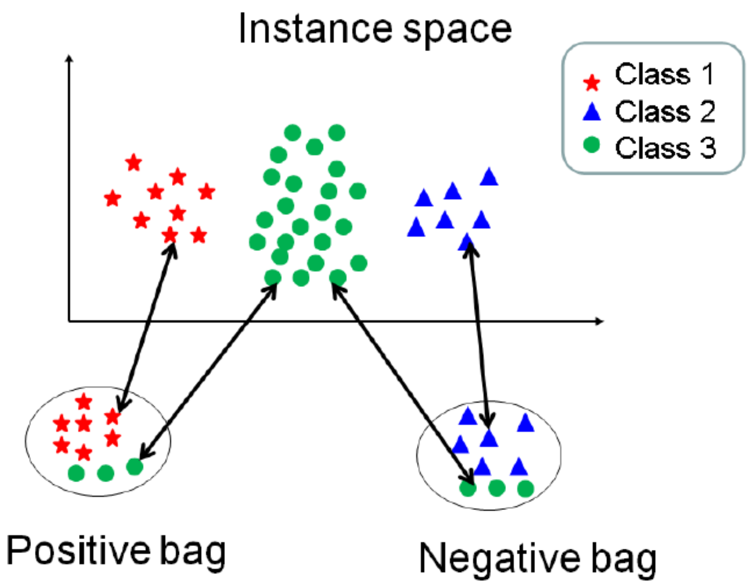
- To learn such a function *F(X)*, we use a set of M images (each represented by a bag) $\mathcal{T} = \{(X_1, y_1), \ldots, (X_M, y_M)\}$, where $y_i \in \{0, 1\}$ (depending if X_i is a positive or a negative image)

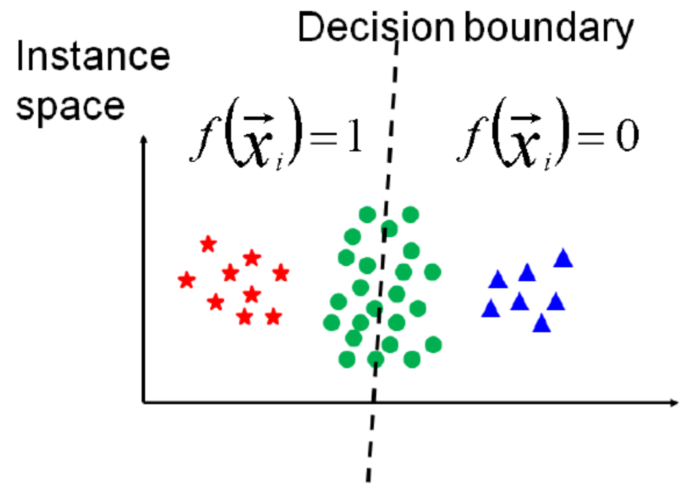
- *F(X)* is a classification function on the bag-level. We can also define instance-level classification function $f(\vec{x}_i)$

Taxonomy of MIC methods



Instance Space Classification

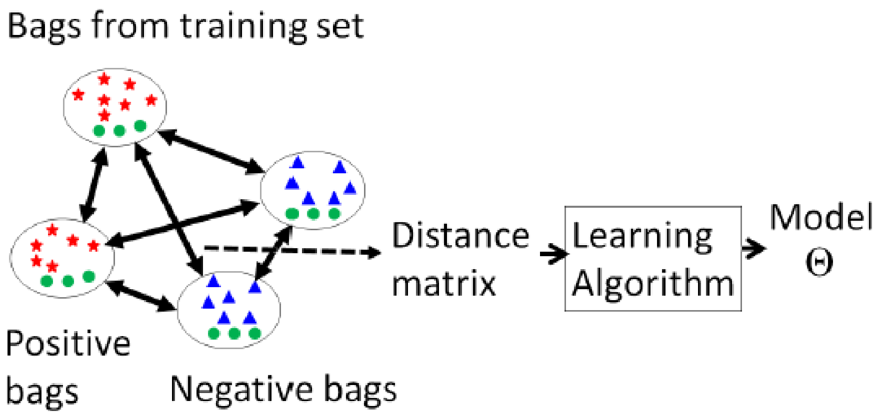




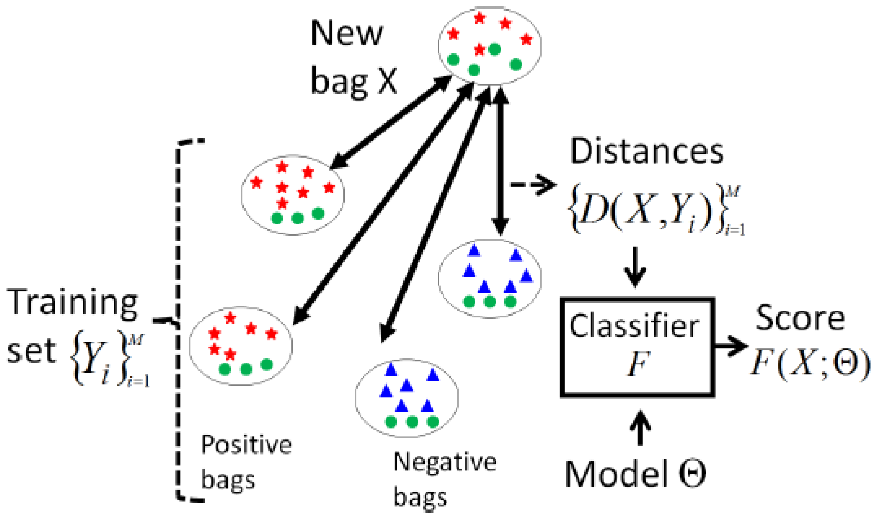
Learn discriminative classifier
f(x) on instance space

Bag Space Classification

Training phase

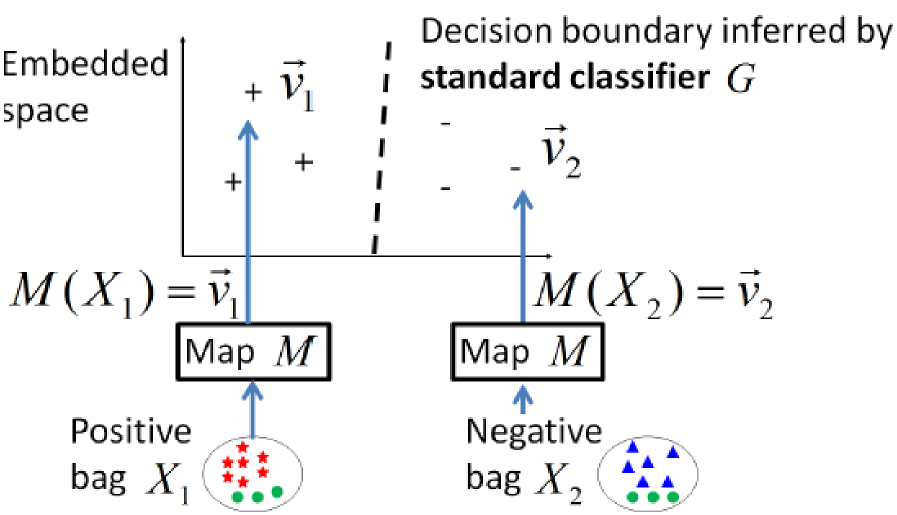


Test phase

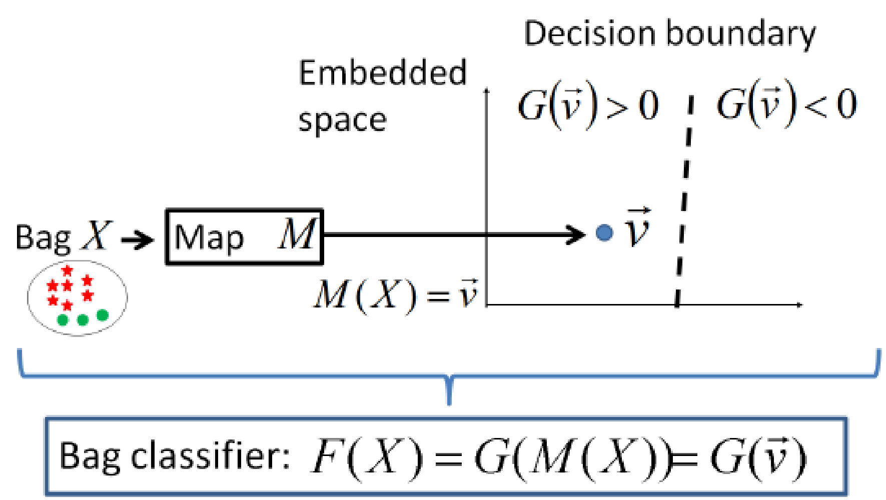


Embedded Space Classification

Training phase

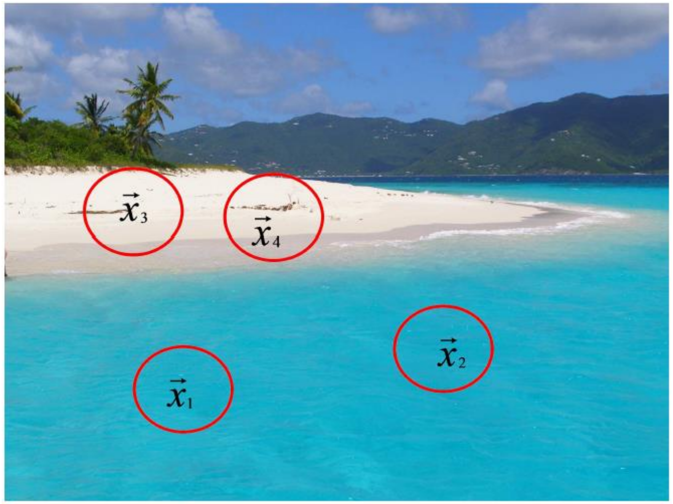


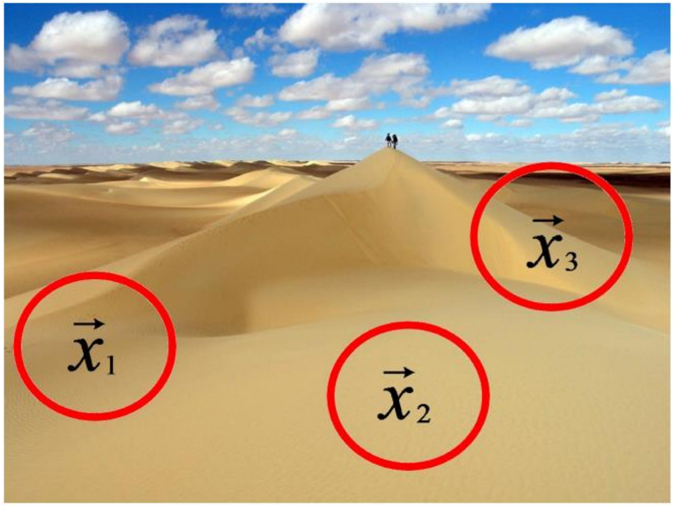
Test phase

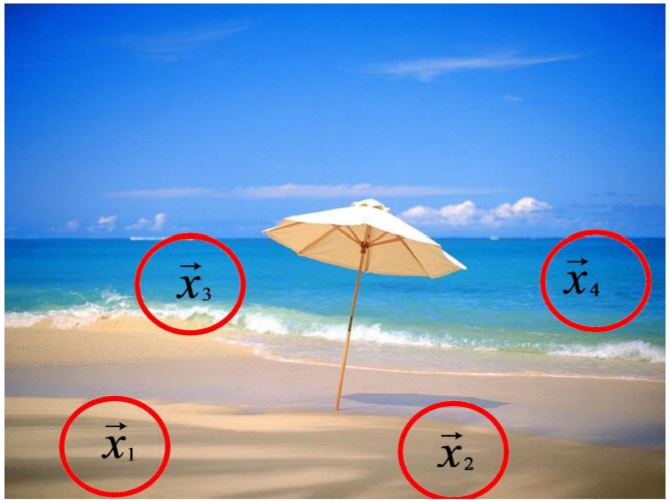


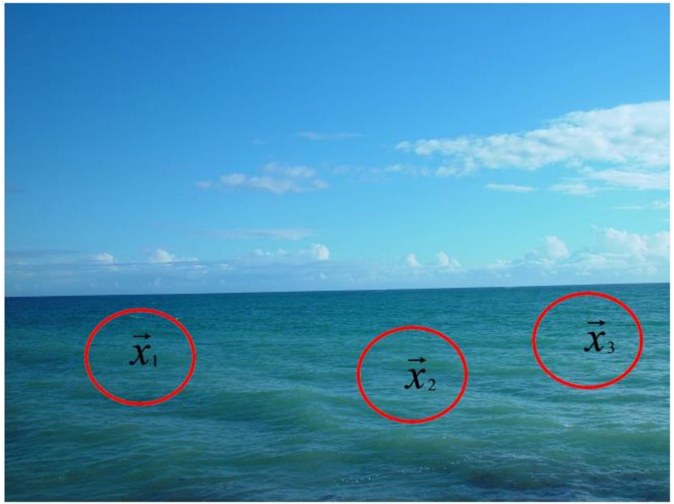
Some examples

Beach/No beach classification problem



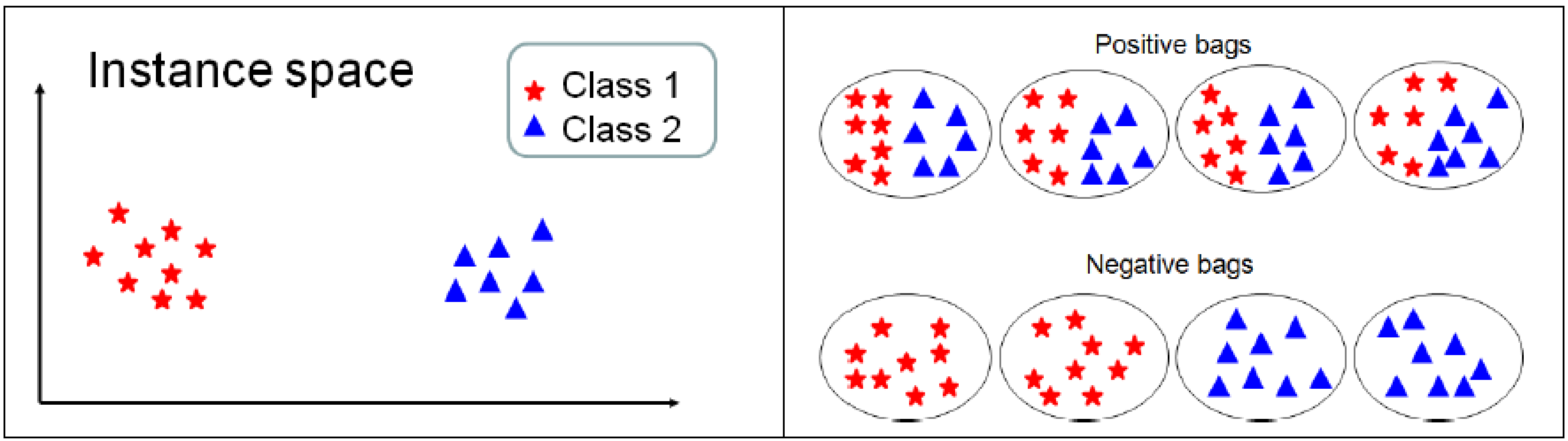




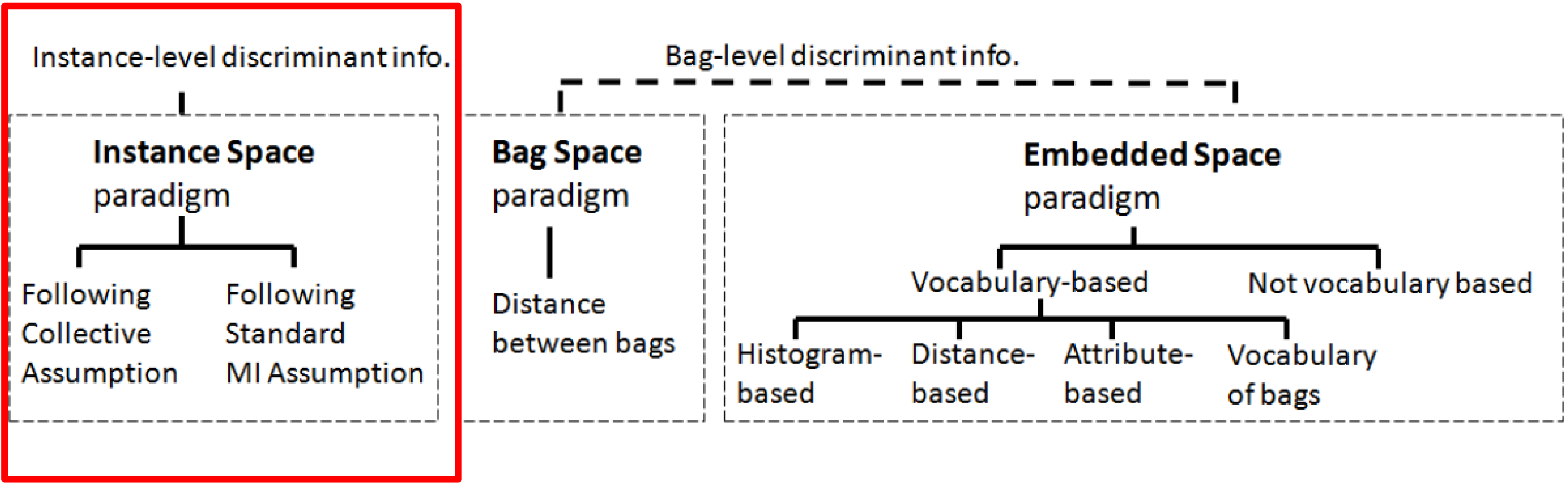


Some examples

Beach/No beach classification problem



Taxonomy of MIC methods



Instance space classification

Methods belonging to this category:

- Learn a classifier on the instance-level $f(\vec{x}) \in [0, 1]$

- This (instance-based) classifier is applied to all instances of a bag and the results are aggregated in order to obtain a bag-based decision

$$F(X) = \frac{f(\vec{x}_1) \circ f(\vec{x}_2) \circ \ldots f(\vec{x}_N)}{Z}$$

Instance space classification

Issue: Since we don't have instance-based labels, how can we train $f(\vec{x}_i)$?

- Standard MI assumption: Every positive bag contains at least one positive instance and every negative bag contains only negative instances.

- Collective assumption: all instances in a bag contribute equally to the bag's label

Instance space classification

Approaches following the standard MI assumption:

1. Axis-Parallel-Rectangle: train an instance-level classifier as:

$$f(\vec{x}; \mathcal{R}) = \begin{cases} 1 & \text{if } \vec{x} \in \mathcal{R} \\ 0 & \text{otherwise} \end{cases}$$

where *R* is an rectangle defined in the instance space.

R is optimized by maximizing the number of positive bags in the training set that contain at least one instance in *R*, while (at the same time) the number of negative bags not containing any instance in *R* is maximized.

Then, a bag-level classifier is obtained by: $F(X) = \max_{\vec{x} \in X} f(\vec{x})$

Instance space classification

Approaches following the standard MI assumption:

2. Support Vector Machine (SVM)-based instance-level classification:
 - Maximize the margin as in SVM,
 - Replace the constraints with:

$$
\begin{array}{llll}
f(\vec{x};\Theta) & \leq & -1+\xi_-, & \forall \vec{x} \in \mathcal{T}^- \quad (*) \\
f(\mu(X);\Theta) & \geq & (\frac{2}{|X|}-1)-\xi_+, & \forall X \in \mathcal{B}^+ \quad (**)
\end{array}
$$

where

$$\mathcal{T} = \mathcal{T}^+ \cup \mathcal{T}^-$$

Positive instances

Negative instances

$$
\begin{array}{lll}
\mathcal{T}^+ & = & \{\mu(X) : X \in \mathcal{B}^+\} \\
\mathcal{T}^- & = & \{\vec{x} : \vec{x} \in X \in \mathcal{B}^-\}
\end{array}
$$

Positive bags

Negative bags

μ(X): mean vector in *X*

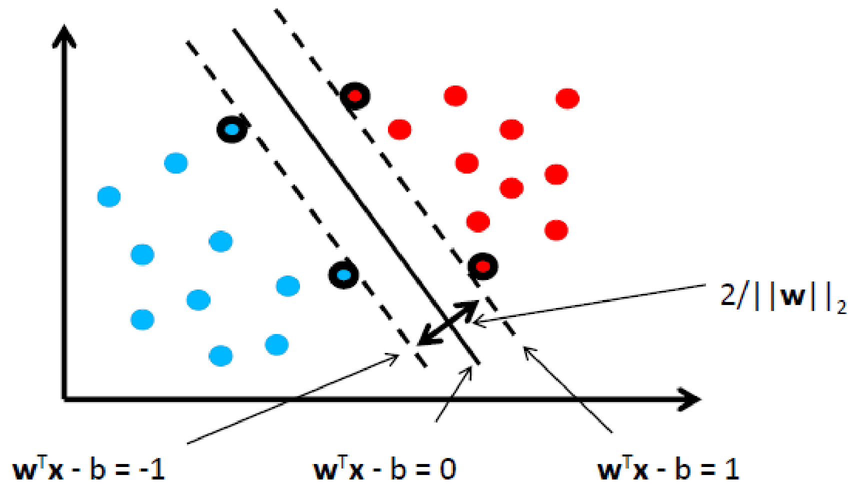
Instance space classification

Reminder of SVM

$$\mathcal{J}_{SVM} = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\xi_i,$$

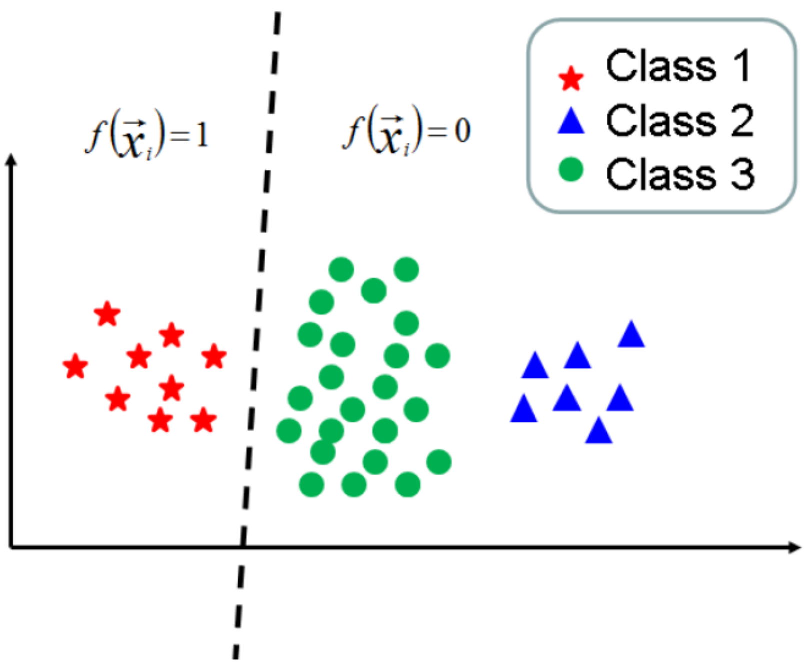
subject to the constraints:

$$\begin{aligned} l_i(\mathbf{w}^T\phi_i - b) &\geq 1 - \xi_i,\ i = 1, \ldots, N \\ \xi_i &\geq 0. \end{aligned}$$



Instance space classification

Type of solution of methods following the standard MI assumption:



Instance space classification

Issue: Since we don’t have instance-based labels, how can we train $f(\vec{x}_i)$?

- Standard MI assumption: Every positive bag contains at least one positive instance and every negative bag contains only negative instances.

- Collective assumption: all instances in a bag contribute equally to the bag’s label

Instance space classification

Approaches following the Collective assumption:

- All instances inherit the label of the bag. Then an instance-based classifier $f(\vec{x}_i)$ is trained.

- A bag-level classifier is obtained by:

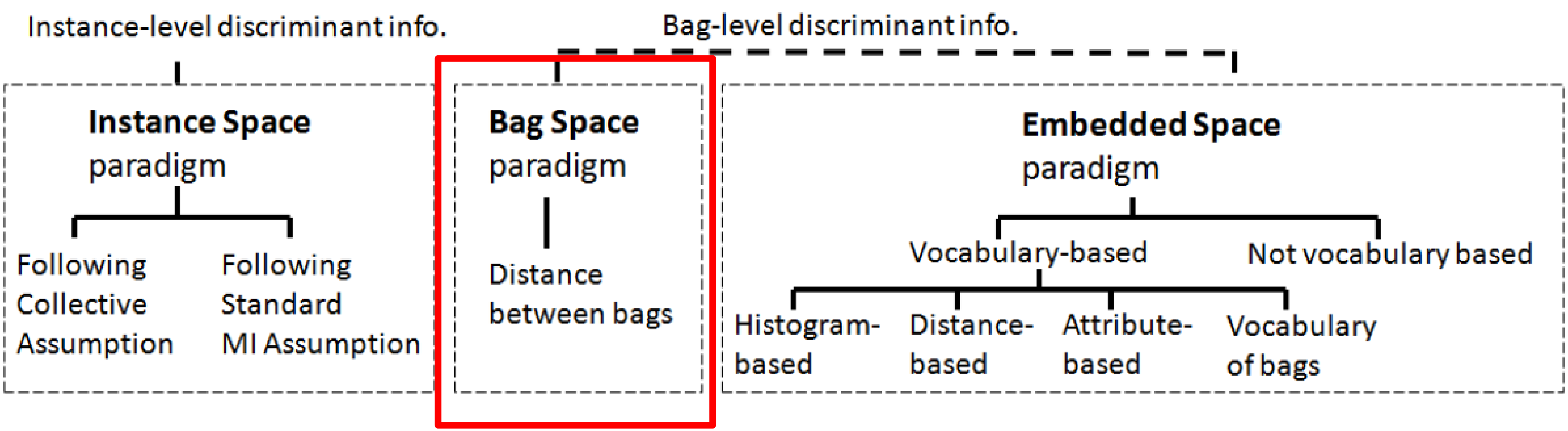
1. Averaging the instance-based classification results $F(X) = \frac{1}{|X|} \sum_{\vec{x} \in X} f(\vec{x})$

2. Applying a weighted average on the instance-based classification results

$$F(X) = \frac{1}{\sum_{\vec{x} \in X} w(\vec{x})} \sum_{\vec{x} \in X} w(\vec{x}) f(\vec{x})$$

Weights are optimized based on the training bag labels

Taxonomy of MIC methods



Bag space classification

Methods belonging to this category define the classification function using the entire bag *X* (global information). This allows the algorithm to exploit more information for defining *F(X)*

In order to define *F(X)* in the bag space, we define:

- A distance function *D(X,Y)* encoding the dissimilarity between two bags

- A kernel function *K(X,Y)* encoding the similarity between to bags

Note that distance functions can be used to define kernels and kernel functions can be used for distance calculation:

$$K(X,Y) = \exp(-\gamma D(X,Y)) \qquad D(X,Y) = \sqrt{K(X,X) - 2K(X,Y) + K(Y,Y)}$$

Bag space classification

Distance/kernel functions on the bag level:

1. Minimal Hausdorff distance: $$D(X,Y) = \min_{\vec{x}\in X, \vec{y}\in Y} \|\vec{x}-\vec{y}\|$$

2. EMD distance: $$D(X,Y) = \frac{\sum_i \sum_j w_{ij} \|\vec{x}_i - \vec{y}_j\|}{\sum_i \sum_j w_{ij}}$$

Weights are optimized based on the training bag labels

3. Chamfer distance: $$D(X,Y) = \frac{1}{|X|}\sum_{\vec{x}\in X} \min_{\vec{y}\in Y} \|\vec{x}-\vec{y}\| + \frac{1}{|Y|}\sum_{\vec{y}\in Y} \min_{\vec{x}\in X} \|\vec{x}-\vec{y}\|$$

4. Bag-space kernel: $$K(X,Y) = \sum_{\vec{x}\in X, \vec{y}\in Y} k(\vec{x},\vec{y})^p$$

The value of p is selected based on cross-validation

k(·, ·) is any kernel function, e.g. linear, RBF, polynomial

Bag space classification

Using the above-defined distance/kernel functions, we can apply standard classification algorithms, like:

1. (k-)Nearest Neighbor classifier

2. Support Vector Machine (SVM)

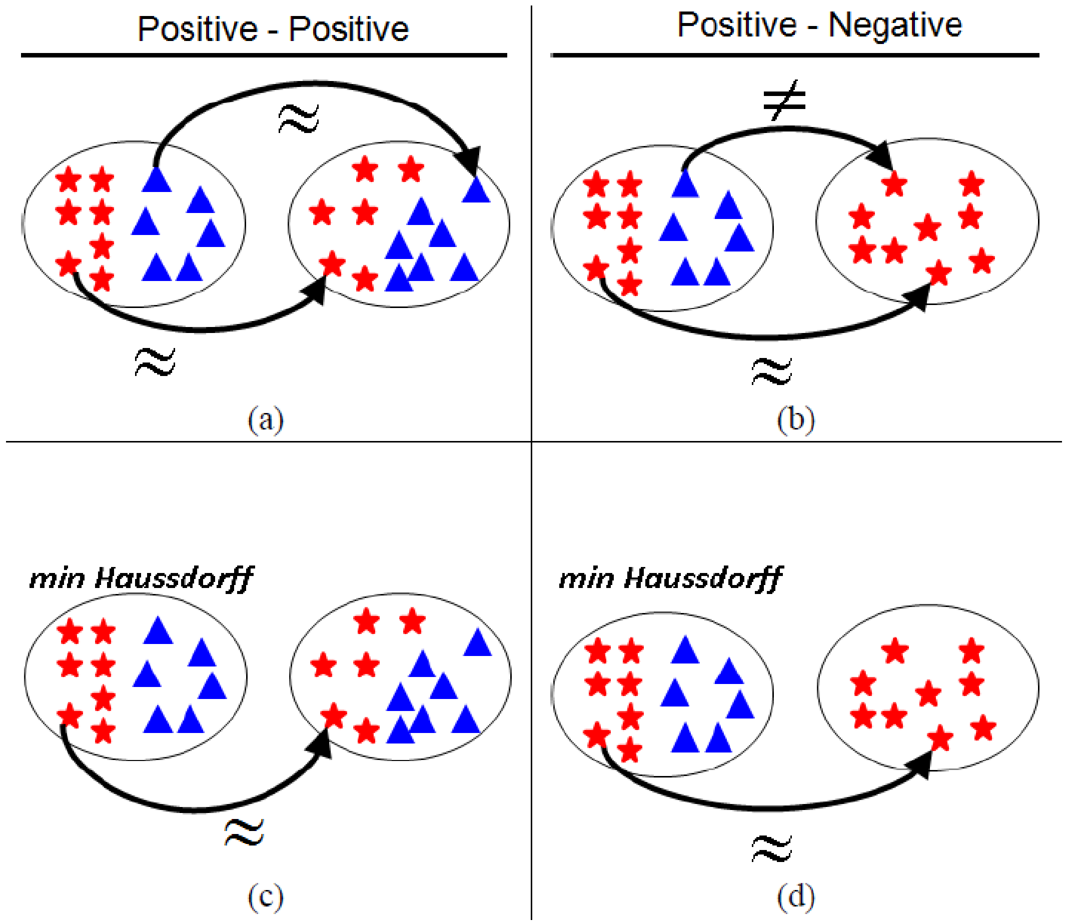
Bag space classification

(a) and (b): Chamfer and EMD distances
(c) and (d): minimal Haussdorf distance

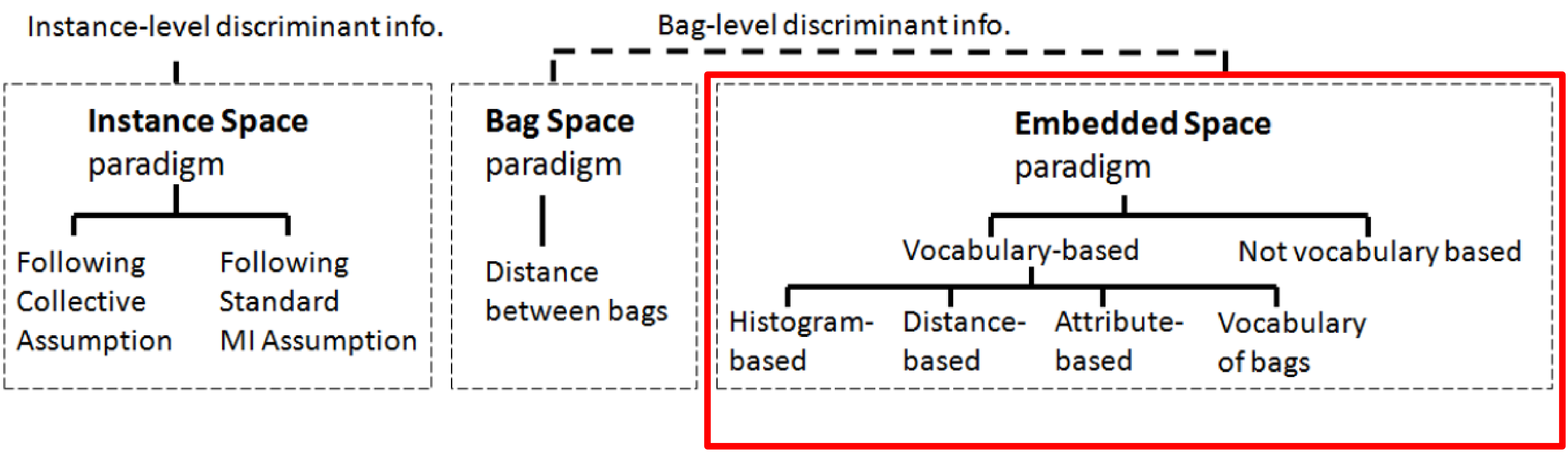
Minimal Hausdorff distance: $D(X,Y) = \min_{\vec{x}\in X, \vec{y}\in Y} \|\vec{x}-\vec{y}\|$

EMD distance: $D(X,Y) = \dfrac{\sum_i \sum_j w_{ij} \|\vec{x}_i - \vec{y}_j\|}{\sum_i \sum_j w_{ij}}$

Chamfer distance: $D(X,Y) = \dfrac{1}{|X|}\sum_{\vec{x}\in X} \min_{\vec{y}\in Y} \|\vec{x}-\vec{y}\| + \dfrac{1}{|Y|}\sum_{\vec{y}\in Y} \min_{\vec{x}\in X} \|\vec{x}-\vec{y}\|$



Taxonomy of MIC methods



Embedded space classification

Methods belonging to this category define a mapping $\boldsymbol{M}$: $X \rightarrow \vec{v}$ from the bag X to a feature vector $\vec{v}$ (which encodes the information of the bag). This is done by:

- Aggregating the statistics of all instance inside the bag

- Using a vocabulary (a set of prototypes) in order to encode similarity of instances in the bag with patterns/prototypes discovered in the training data

Embedded space classification

Methods aggregating the statistics of all instance inside the bag:

- Simple MI using the mean instance: $\mathcal{M}(X) = \frac{1}{|X|} \sum_{\vec{x} \in X} \vec{x}$

- Min-max instance vector: $\mathcal{M}(X) = (a_1, \ldots, a_d, b_1, \ldots, b_d)$

where $a_j = \min_{\vec{x} \in X} x_j$ and $b_j = \max_{\vec{x} \in X} x_j$

Embedded space classification

Methods using a vocabulary of prototypes:

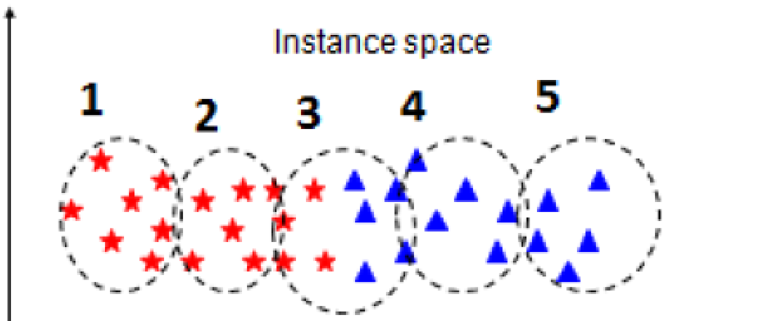
- Use a vocabulary defined as $V = \{(C_1, \theta_1), \ldots, (C_K, \theta_K)\}$ encoding a set of K 'concepts'. The j-th concept C_j has a set of parameters θ_j.

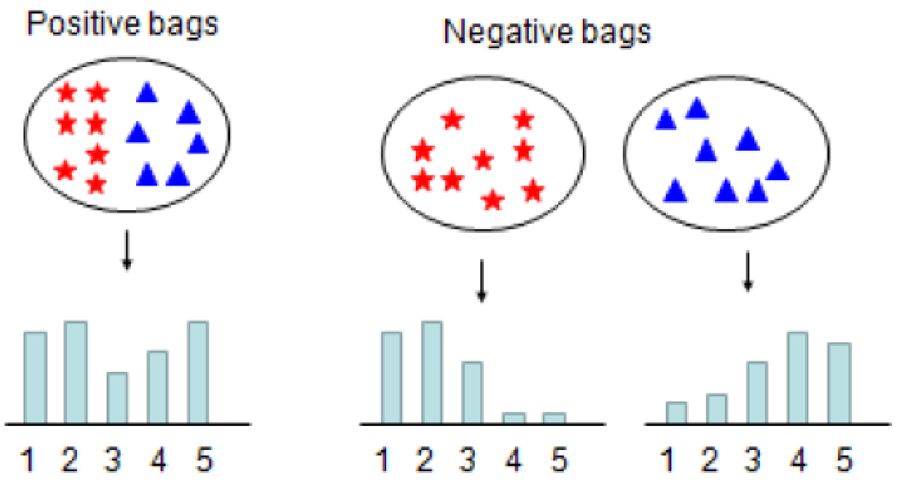
- A mapping function $\mathcal{M}(X, V) = \vec{v}$ mapping the bag X to a K-dimensional feature vector $\vec{v} = (v_1, \ldots, v_K)$. This mapping corresponds to an embedding of X to a K-dimensional feature space that takes into account the K prototypes/patterns

- A standard (vector-based) supervised classifier, like (k-)NN, SVM, etc.

Embedded space classification

$$V = \{(C_1, \theta_1), \ldots, (C_K, \theta_K)\}$$





Embedded space classification

A vocabulary-based method is the Bag-of-Words (BoWs) or Bag-of-Features (BoF) model, where:

- The vocabulary $V = \{(C_1, \theta_1), \ldots, (C_K, \theta_K)\}$ is obtained by clustering the instances in K groups (e.g. by applying K-Means algorithm). The j-th group C_j has parameters θ_j, which correspond to the cluster mean vector

- A mapping function $\mathcal{M}(X, V) = \vec{v}$ mapping the bag X to a K-dimensional feature vector $\vec{v} = (v_1, \ldots, v_K)$, where

$$v_j = \frac{1}{Z} \sum_{\vec{x}_i \in X} f_j(\vec{x}_i), \quad j = 1, \ldots, K$$

Depending on $f_j(\cdot)$ different BoWs models can be defined.

Embedded space classification

Histogram-based BoWs:

$$f_j(\vec{x}) = \begin{cases} 1 & \text{if } j = \arg\min_{k=1,\ldots,K} \|\vec{x} - \vec{p}_k\| \\ 0 & \text{otherwise} \end{cases}$$

Distance(Similarity)-based BoWs:

$$v_j = \max_{\vec{x}_i \in X} s_j(\vec{x}_i) \quad j = 1, \ldots, K$$

$$s_j(\vec{x}) = \exp\left(-\frac{\|\vec{x} - \vec{p}_j\|^2}{\sigma^2}\right)$$

Embedded space classification

In the above BoWs models, the vocabulary (also called codebook) is obtained by applying an unsupervised approach (K-Means clustering).

BoWs models where the vocabulary is <u>optimized</u> by exploiting the bag-level labels are possible

Such methods:

- Initialize the vocabulary by using an unsupervised approach (e.g. by applying K-Means clustering)

- Update (fine-tune) the vocabulary in order to achieve better classification performance at the bag-level

Embedded space classification

Discriminant Bag-of-Words model. Slightly different notation:

N_T training bags

feature vectors $\mathbf{p}_{ij}, \in \mathbb{R}^D,\ i = 1, \ldots, N_T,\ j = 1, \ldots, N_i$

Codebook $\mathbf{V} \in \mathbb{R}^{D \times K}$ $\mathbf{v}_k \in \mathbb{R}^D,\ k = 1, \ldots, K$

Similarity function $d_{ijk} = \|\mathbf{v}_k - \mathbf{p}_{ij}\|_2^{-g}$

Membership (normalized similarity) $\mathbf{u}_{ij} = \dfrac{\mathbf{d}_{ij}}{\|\mathbf{d}_{ij}\|_1}$

Bag representation $\mathbf{q}_i = \dfrac{1}{N_i} \sum_{j=1}^{N_i} \mathbf{u}_{ij}$ and normalized one $\mathbf{s}_i = \dfrac{\mathbf{q}_i}{\|\mathbf{q}_i\|_2}$

Embedded space classification

After defining the bag representations $\mathbf{s}_i$, i=1,…, N_T , we apply LDA-based classification:

- We standardize $\mathbf{s}_i$'s to obtain $\mathbf{x}_i$'s
  (the training set will have zero mean and unit variance)

- We map $\mathbf{x}_i$'s to $\mathbf{z}_i$'s by: $\mathbf{z}_i = \mathbf{W}^{*T}\mathbf{x}_i$

- We classify bags using the representations $\mathbf{z}_i$'s and the Nearest Class Centroid classifier.

Embedded space classification

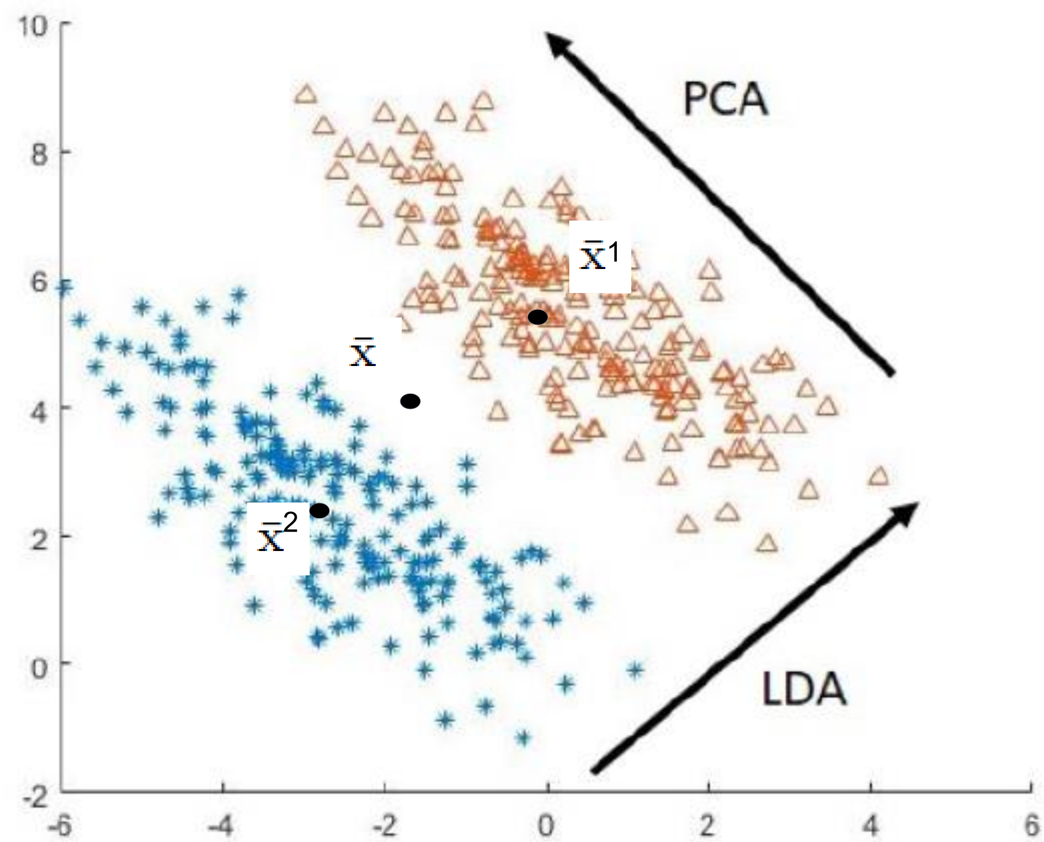
Reminder of Linear Discriminant Analysis (LDA)

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \frac{trace\{\mathbf{W}^T\mathbf{A}\mathbf{W}\}}{trace\{\mathbf{W}^T\mathbf{B}\mathbf{W}\}}$$

$$\mathbf{W}^* = \underset{\mathbf{W}^T\mathbf{W}=\mathbf{I}}{argmax}\ Tr\left[\mathbf{W}^T\left(\mathbf{B} - \lambda^*\mathbf{A}\right)\mathbf{W}\right]$$

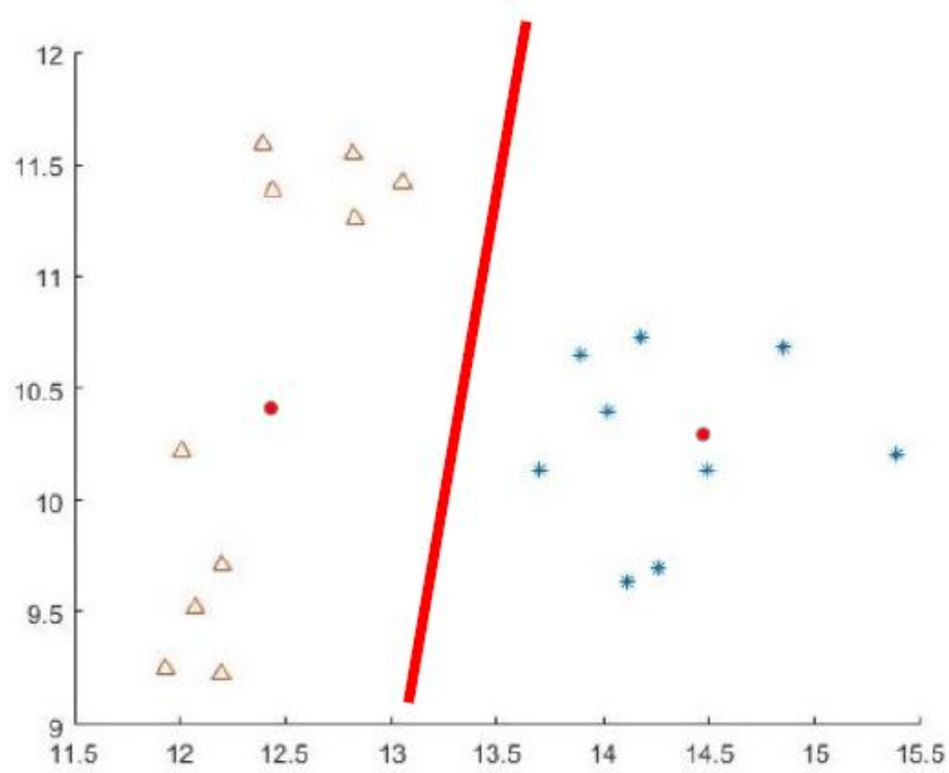
$$\mathbf{A} = \sum_{\alpha=1}^{C}\sum_{i=1}^{N_T} b_i^{\alpha}\left(\mathbf{x}_i - \bar{\mathbf{x}}^{\alpha}\right)\left(\mathbf{x}_i - \bar{\mathbf{x}}^{\alpha}\right)^T$$

$$\mathbf{B}_t = \sum_{\alpha=1}^{C}\left(\bar{\mathbf{x}}^{\alpha} - \bar{\mathbf{x}}\right)\left(\bar{\mathbf{x}}^{\alpha} - \bar{\mathbf{x}}\right)^T$$



Embedded space classification

Reminder of Nearest Class Centroid classifier



Embedded space classification

After initializing the codebook vectors (using K-Means), we use the bag labels in order to optimize them using the LDA optimization criterion.

This is done by applying an iterative optimization process, where at each step t, the codebook vectors are updated by following the gradient of LDA criterion:

$$\mathbf{v}_{k,t+1} = \mathbf{v}_{k,t} - \eta \frac{\partial \mathcal{J}_t}{\partial \mathbf{v}_{k,t}}$$

$$\frac{\partial \mathcal{J}_t}{\partial \mathbf{v}_{k,t}} = \frac{\partial \mathcal{J}_t}{\partial x_{ik,t}} \frac{\partial x_{ik,t}}{\partial q_{ik,t}} \frac{\partial q_{ik,t}}{\partial d_{ijk,t}} \frac{\partial d_{ijk,t}}{\partial v_{k,t}}$$

Embedded space classification

After initializing the codebook vectors (using K-Means), we use the bag labels in order to optimize them using the LDA optimization criterion.

This is done by applying an iterative optimization process, where at each step t, the codebook vectors are updated by following the gradient of LDA criterion:

$$\begin{aligned}\frac{\partial \mathcal{J}_t}{\partial \mathbf{v}_{k,t}} &= \left(a\tilde{\mathbf{W}}_{t(i,:)}(\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t^{\alpha}) - c\tilde{\mathbf{W}}_{t(i,:)}\bar{\mathbf{x}}_t^{\alpha}\right) \\ &\cdot \left(\frac{1}{\tilde{s}_{k,t}} - \frac{s_{ik,t} - \bar{s}_{k,t}}{\tilde{s}_{k,t}^3}\right)\left(\frac{1}{\|\mathbf{q}_{i,t}\|_2} - \frac{q_{ik,t}^2}{\|\mathbf{q}_{i,t}\|_2^3}\right) \\ &\cdot \frac{N_T - 1}{N_T N_i}\left(\frac{1}{\|\mathbf{d}_{ij,t}\|_1} - \frac{d_{ijk,t}}{\|\mathbf{d}_{ij,t}\|_1^2}\right) \\ &\cdot -g\|\mathbf{v}_{k,t} - \mathbf{p}_{ij}\|_2^{-(g+2)}(\mathbf{v}_{k,t} - \mathbf{p}_{ij})\end{aligned}$$

$$a = \frac{2b_i^{\alpha}}{trace(\mathbf{W}_t^T\mathbf{B}_t\mathbf{W}_t)}$$

$$c = \frac{2b_i^{\alpha}trace(\mathbf{W}_t^T\mathbf{A}_t\mathbf{W}_t)}{trace(\mathbf{W}_t^T\mathbf{B}_t\mathbf{W}_t)^2}$$

$$\tilde{\mathbf{W}}_t = \mathbf{W}_t\mathbf{W}_t^T$$

Embedded space classification

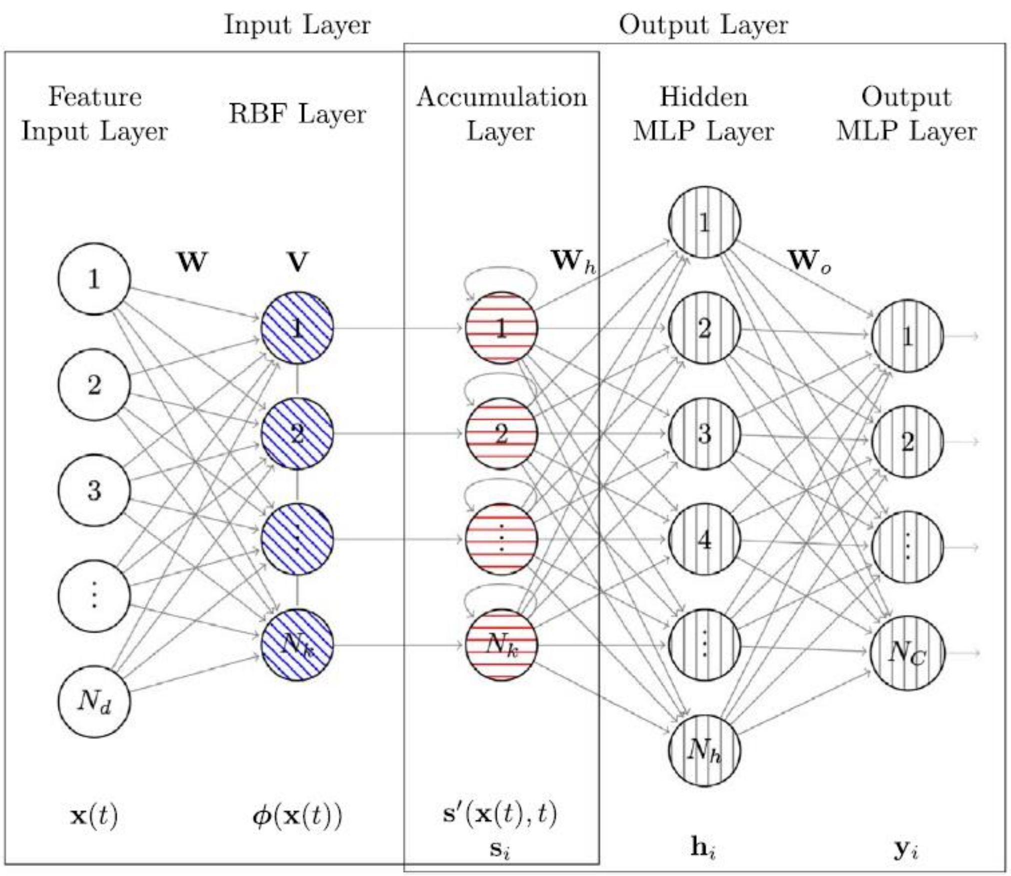
Neural Bag-of-Words (Bag-of-Features) model:

- Similar to the Discriminant BoWs idea, but using a neural network-based topology

- This allows to jointly optimize the Codebook and the parameters of a non-linear classifier

Embedded space classification

Neural Bag-of-Words (Bag-of-Features) model:



Embedded space classification

Neural Bag-of-Words (Bag-of-Features) model:

N training bags

feature vectors $\mathbf{x}_{ij} \in \mathbb{R}^D \ (j = 1 \ldots N_i)$, i=1,…,N

Codebook $\mathbf{V} \in \mathbb{R}^{D \times K}$ $\mathbf{v}_k \in \mathbb{R}^D, \ k = 1, \ldots, K$

Similarity function $[\mathbf{d}_{ij}]_k = \exp\left(\frac{-\| \mathbf{v}_k - \mathbf{x}_{ij} \|_2}{g}\right)$

Membership (normalized similarity) $\mathbf{u}_{ij} = \frac{\mathbf{d}_{ij}}{\|\mathbf{d}_{ij}\|_1}$

Bag representation $\mathbf{s}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \mathbf{u}_{ij}$

Embedded space classification

Neural Bag-of-Words (Bag-of-Features) model:

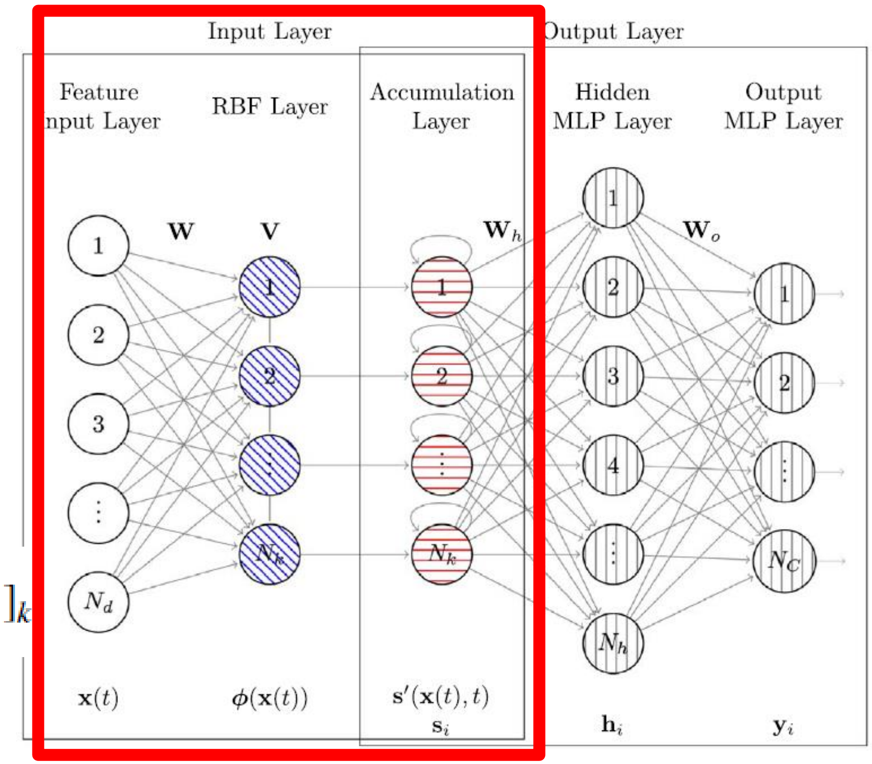
- The (normalized) output of the RBF layer is:

$$[\boldsymbol{\phi}(\mathbf{x})]_k = \frac{\exp(-\|(\mathbf{x} - \mathbf{v}_k) \odot \mathbf{w}_k\|_2)}{\sum_{m=1}^{N_K} \exp(-\|(\mathbf{x} - \mathbf{v}_m) \odot \mathbf{w}_m\|_2)}$$

- Outputs of the RBF layer are accumulated as follows:

$$[\mathbf{s}'(\mathbf{x}(t), t)]_k = \frac{1}{t}[\boldsymbol{\phi}(\mathbf{x}(t))]_k + \frac{t-1}{t}[\mathbf{s}'(\mathbf{x}(t-1), t-1)]_k$$

leading to $\mathbf{s}_i = \frac{1}{t}\sum_{j=1}^{t} \boldsymbol{\phi}(\mathbf{x}_{ij})$



Embedded space classification

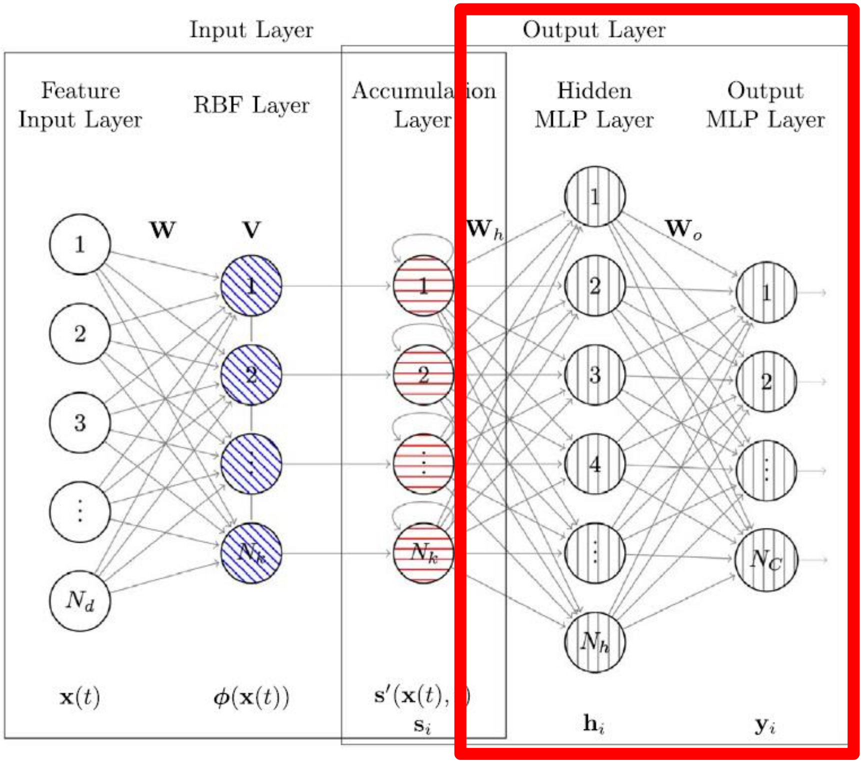
Neural Bag-of-Words (Bag-of-Features) model:

- Multi-layer Perceptron (MLP) layers:

$$\mathbf{h}_i = \phi^{(s)}(\mathbf{W}_H \mathbf{s}_i + \mathbf{b}_H)$$

$$\mathbf{y}_i = \phi^{(s)}(\mathbf{W}_O \mathbf{h}_i + \mathbf{b}_O)$$

$$\phi^{(s)}(x) = \frac{1}{1 + e^{-x}}$$



AARHUS
UNIVERSITET

Department of Electrical and
Computer Engineering

Embedded space classification

Neural Bag-of-Words (Bag-of-Features) model:

- Update all parameters using error Back-propagation:

$$\Delta(\mathbf{W}_O, \mathbf{W}_H, \mathbf{b}_O, \mathbf{b}_H, \mathbf{V}, \mathbf{W}) = -\left(\eta_{MLP}\frac{\partial L}{\partial \mathbf{W}_O}, \eta_{MLP}\frac{\partial L}{\partial \mathbf{W}_H}, \eta_{MLP}\frac{\partial L}{\partial \mathbf{b}_O}, \eta_{MLP}\frac{\partial L}{\partial \mathbf{b}_H}, \eta_V\frac{\partial L}{\partial \mathbf{V}}, \eta_W\frac{\partial L}{\partial \mathbf{W}}\right)$$