Knowledge Discovery in Databases II

Winter Term 2015/2016

Lecture 12:
Variety: Multi –Instance data

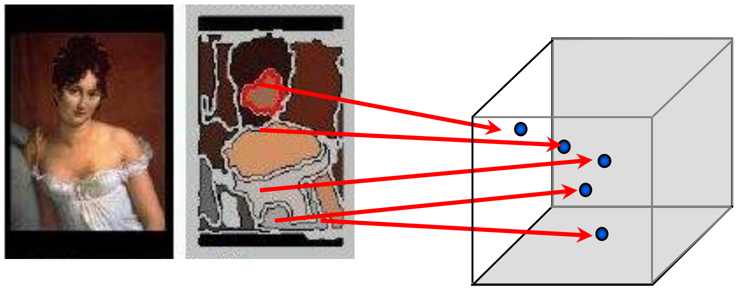
Lectures : Dr Eirini Ntoutsi, PD Dr Matthias Schubert
Tutorials: PD Dr Matthias Schubert

http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_II_(KDD_II)

Multi-Instance Data

- Multi-Instance Data
- Aggregation-based Methods
- Distance and Similarity Measures
- Multi-Instance Classification
- Clustering Multi-Instance Objects

What is Multi-Instance Data ?

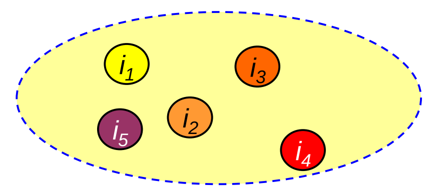


Bitmap   set of shape vectors

set of word vectors   Website

Multi-Instance objects describe:

- multiple components (e.g. CAD data)
- various appearances (e.g. proteins)
- set-valued objects (e.g. market baskets, teams)



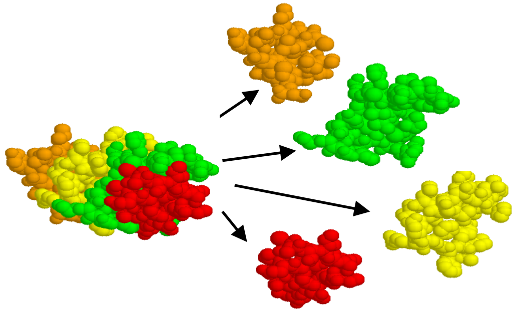
Differences to other structured objects:

1. All instances are elements of the same features space (vs. Multi-View data)
2. Multi-Instance objects do not have an order (vs. time-series, sequences, trajectories)

Examples for Multi-Instance Objects

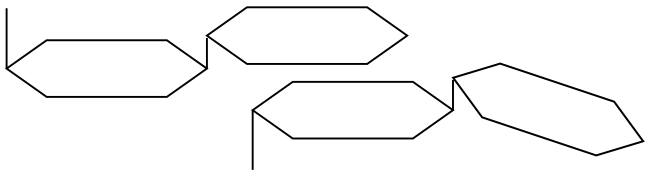
Proteins

- proteins consist of multiple amino acid sequences
- each sequences is an instance
- a protein is a set of its sequences



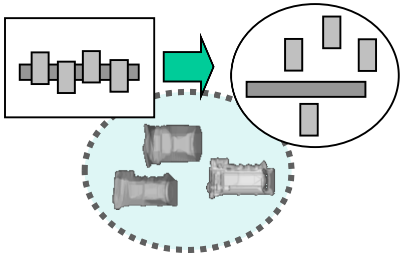
Macro-Molecules

- varying spatial conformations
- each conformation is an instance
- the molecule is described by
  a set of all possible conformations

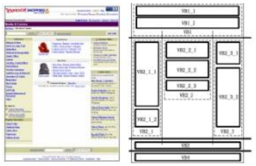


Further Examples

- **CAD-components:**
  set of spatial primitives



- **HTML documents:**
  set of layout blocks
  (dom tree structure is dropped)



- **Video data:**
  videos can be described by
  sets of shots (order is dropped)

Formally:

Object o is part of the power set of R: $o = \{r_1,..,r_n\} \in 2^R$
where R is the feature space of instance (shortly instance space)

Multi-Instance Data

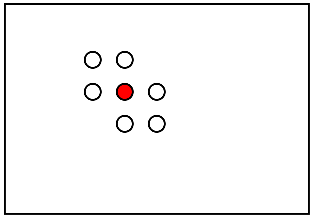
- Multi-Instance Data
- Aggregation-based Approaches
- Distance and Similarity Measures
- Multi-Instance Classification
- Clustering Multi-Instance Objects

Aggregation-based Approaches

Idea: Reduce the multi-instance object (i.e., *set* of instances) into a *single* representative instance.

E.g., build the centroid

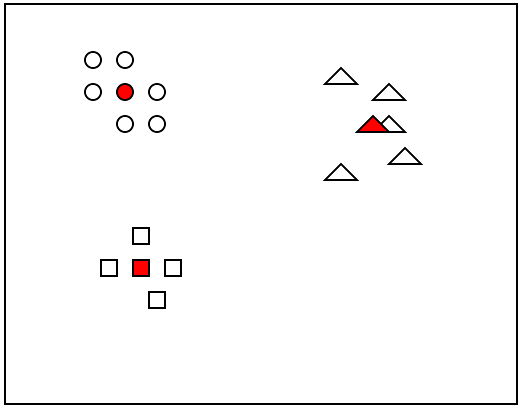
$\Rightarrow$ simple method describing a set by its componentwise means



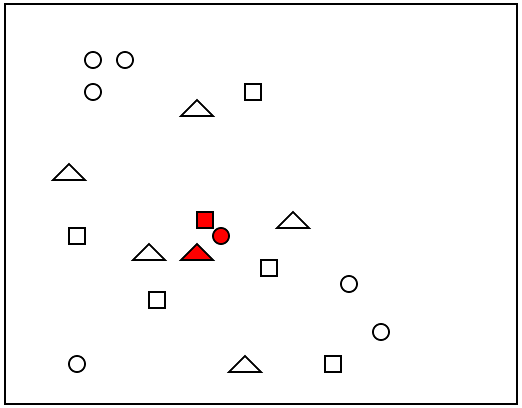
Problems:

- properties of the particular instances are lost
- cardinality of the set is lost
- outliers are not described well

Aggregation-based Approaches



1. case*: aggregation on suitable data*



2. case*: aggregation in unsuitable data*

Conclusion:

Aggregation depends on the distribution of the objects.

- If all instances are drawn from the same distribution aggregation makes sense.
- If instances might be drawn from different distributions, aggregation is not suitable.

Distance and Similarity Measures

Idea: Many data mining algorithms only need pairwise comparisons.

$\Rightarrow$ Define distances and kernel-functions on multi-instance objects

There are multiple ways to compare multi-instance objects:

- How many instances of should be similar?
- Does there have to be a bijective mapping between the sets ?

=> There are multiple similarity measures which might make sense in varying application areas.

General Problem of comparing sets

Multi-instance objects comparison yields an assignment task:

Which instance in object X has to be compared to which instance in object Y ?

Given two objects $X=\{x_1, x_2, x_3\}$ and $Y=\{y_1, y_2, y_3\}$:

- Each x_i can be compared to each y_j.
- To how many y_j has each x_i to be similar?
- To how many x_i has each y_j to be similar?

	X_1	X_2	X_3
Y_1			
Y_2			
Y_3			

- Usually: Each instance is assigned to at least one instance in the other object (often the closest).

Hausdorff Distance

Idea: Each instance is *covered* by the *closest* instance from the other object. The *maximum cover* distance describes the distance of the two objects.

- minimum distance = most similar instance (smallest radius to cover the instance)
- maximum distance over all row /columns (worst case cover)
- maximum of row and column maximums achieves symmetry

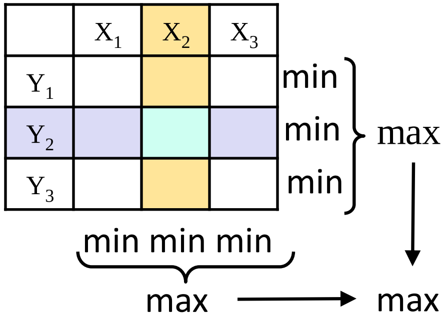
Definition: The Hausdorff Distance

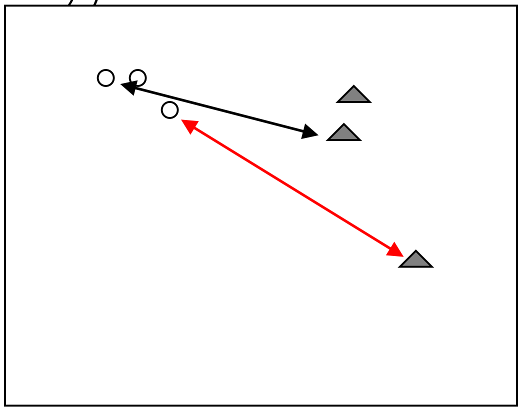
Let *X, Y be two* MI-objects and *d*(*x,y*) an instance distance measure over the feature space *R*, then the Hausdorff distance is defined as follows:

$$d_{Hausdorff}(X,Y) = \max\left(\max_{x_i \in X}\left(\min_{y_j \in Y}\left(d(x_i, y_j) \right)\right), \max_{y_i \in Y}\left(\min_{x_j \in X}\left(d(x_i, y_j) \right)\right)\right)$$

Informally, is the max distance out of the set of all distances between each point of a set to the closest point of a second set.

Complexity: $O(|X| \cdot |Y| \cdot d)$
(assuming *d*(*x,y*) is computable in *O*(*d*))





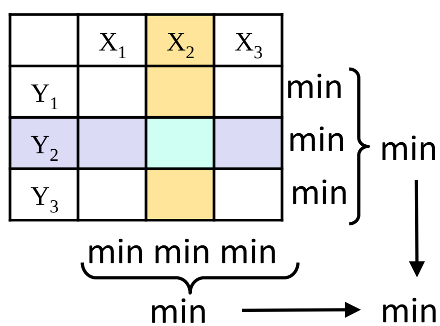
Single Link or Minimal Hausdorff Distance

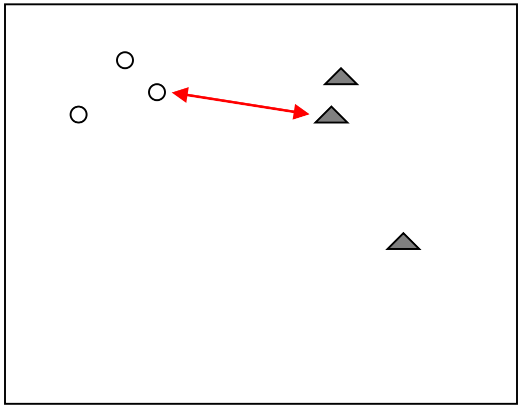
Idea: Use the closest pair of instances.

Definition: Minimal Hausdorff Distance or Single Link Distance

Let *X, Y* be two MI-objects and let *d*(*x,y*) be an instance distance measure in the underlying feature space R, then the minimal Hausdorff or single link distance is defined as follows:

$$d_{\text{singlelink}}(X,Y) = \min_{x_i \in X}\left(\min_{y_j \in Y}\left(d(x_i, y_j) \right) \right)$$





Complexity: $O\left(|X| \cdot |Y| \cdot d\right)$
(assuming *d*(*x,y*) is computable in *O*(*d*))

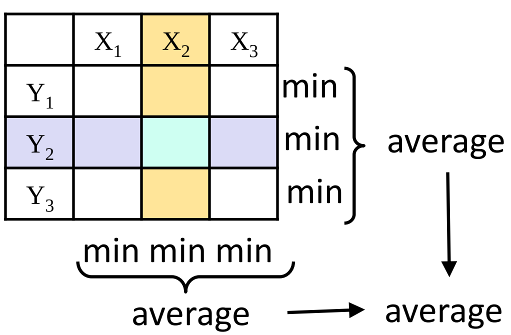
Sum of Minimum Distances

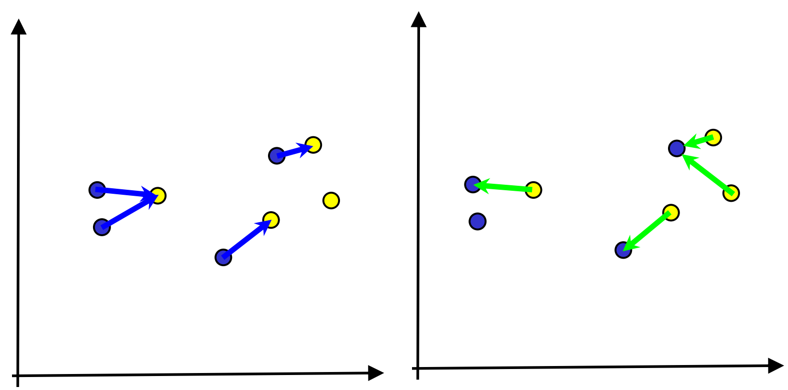
Idea: Use the average distance of the closest pairs.

Definition: Sum of Minimum distances (SMD)

Let *X, Y be two* MI-objects and *d*(*x,y*) an instance distance measure over the feature space *R*, then the SMD distance is defined as follows:

$$d_{SMD}(X,Y) = \frac{1}{2}\left(\frac{1}{|X|}\sum_{x_i \in X}\left(\min_{y_j \in Y}\left(d(x_i, y_j)\right)\right) + \frac{1}{|Y|}\sum_{y_j \in Y}\left(\min_{x_i \in X}\left(d(x_i, y_j)\right)\right)\right)$$





Complexity: $O\left(|X| \cdot |Y| \cdot d\right)$
(assuming *d*(*x,y*) is computable in *O*(*d*))

Minimal Matching Distance (MMD)

Idea: The distance between two sets is described by a cost-minimal bijection.

Definition*:*

Let O_1, O_2 *be two* MI-objects and let $d(x,y)$ be an instance distance measure over the feature space R, then the Minimal Matching Distance is defined as follows:

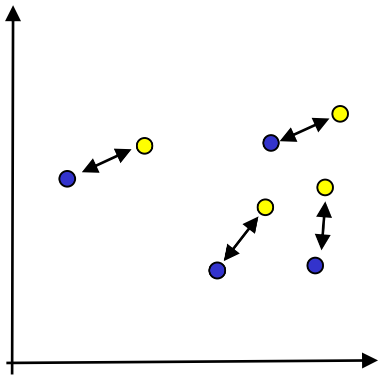
$$d_{MM}(O_1, O_2) = \min_{\pi_i \in \Pi(O_1)} \left(\sum_{k=1}^{|O_2|} d\left(o_{1,\pi(k)}, o_{2,k}\right) + \sum_{l=|O_2|+1}^{|O_1|} w\left(o_{1,\pi(l)}\right) \right)$$

w.l.o.g. let $|O_1| > |O_2|$. $\Pi(O_1)$ *represents the set of all* permutations of the instances in O_1 und $w(o_{i,j})$ is a weighting term penalizing instances without a match.

Remark:

MMD is metric if $w(o_{i,j})$ is large enough to prevent unmatched instances, i.e., $w(o_{i,j})$ *has to be larger than any distance to* any other instance.

=> Not matching any object is always worse than matching it



Computing MMD

Method: Solve a minimum weight perfect matching problem, e.g. with the *Hungarian method* (runtime complexity $O(n^3)$).

Input

The cost matrix: built upon the instances of the compared objects. The entries are the distances of the corresponding instances.

The algorithm requires a square cost matrix: fill missing entries with $w(o_{i,j})$ value.

Algorithm:

1. Subtract the minimum entry from each row
2. Subtract the minimum entry from each column
3. Draw lines through appropriate rows and columns so that *all* the zero entries of the cost matrix are covered and the minimum number of such lines is used
4. Test for optimality: If the min number of covering lines is *n,* an optimal assignment of zeros is possible and we are finished. Otherwise, proceed to Step 5.
5. Determine the smallest entry not covered by any line. Subtract this entry from each uncovered row, and then add it to each covered column. Return to Step 3.

Example: Computing MMD

Matrix of pairwise distances

10	12	20	21
10	12	21	24
14	17	28	30
16	20	30	35

Subtract row min

0	2	10	11
0	2	11	14
0	3	14	16
0	4	14	19

Subtract column min

0	0	0	0
0	0	1	3
0	1	4	5
0	2	4	8

Mark all 0s

0	0	0	0
0	0	1	2
0	1	4	5
0	2	4	8

Add and subtract unmarked m in

1	1	0	0
0	0	0	1
0	1	3	4
0	2	3	7

Mark

1	1	0	0
0	0	0	1
0	1	3	4
0	2	3	7

add and subtract min

2	1	0	0
1	0	0	1
0	0	2	3
0	1	2	6

compute result

2	1	0	0
1	0	0	1
0	0	2	3
0	1	2	6

$$D_{mmd}(O_1,O_2)=21+21+17+16 = 75$$

10	12	20	21
10	12	21	24
14	17	28	30
16	20	30	35

Multi-Instance Data

- Multi-Instance Data
- Aggregation-based Approaches
- Distance and Similarity Measures
- Multi-Instance Classification
- Clustering Multi-Instance Objects

Multi-Instance Classification

Setting:

Training set D={(*O, c*)} where $O \in DB$ and $c \in C$.

Each object *O* is a MI object, i.e, it consists of a *set of instances* or *bag of instances* : $O_i=\{o_j,\ldots,o_k\}$

! Each object O_i has a class label, but the instances (o_j) themselves are not explicitly labeled.

Goal:

Learn a model that predicts the class labels for unseen objects (i.e., sets or bags of instances)

Example:

Simple jailer problem (Chevaleyre & Zucker, 2001):

"Imagine there is a locked door and we have N keychains, each containing a bunch of keys. If a keychain (i.e., bag/set of keys) contains a key (i..e, instance) that can unlock the door, the keychain is useful. The learning goal is to build a model that can predict whether a given keychain is useful or not."

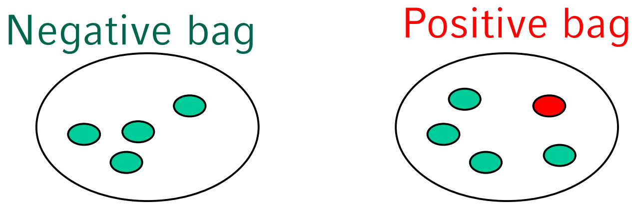
Multi-Instance Classification

Challenge:

Which instances $\{o_j, .., o_k\}$ of O_i *are responsible for the membership of* O_i *in class c?*

Classic multi instance learning (Dietterich et al, 1997):

- Binary class: *positive*, *negative*
- Assumption: instances have hidden/unobserved class labels: *positive* or *negative*
- Assumption: An object O_i is labeled as *positive*, if and only if contains at *least one positive* instance, otherwise it is *negative.*
- Important to define which class is the positive one, during application



General multi instance learning:

- Multi-class: arbitrary number of classes
- Instances can be relevant to multiple classes
- Class membership for object *O* might depend on any instance-subset of *O*

General Multi Instance Classification

Problem:

MI objects from the same class need not be completely similar (similar w.r.t to each instance). => Classes can be described in multiple different ways

General approach to multi-instance classifiers:

- Classes can be defined by “concepts” on the instances (football team = 1 goal keeper and 10 regular players)
- Each concept describes a group of instances
- Concepts might occur in a class or be completely absent
- The cardinality of the concepts in the class might play a role (5 goal keepers and 1 regular player is not a football team)

MI classification with given concepts

Input:

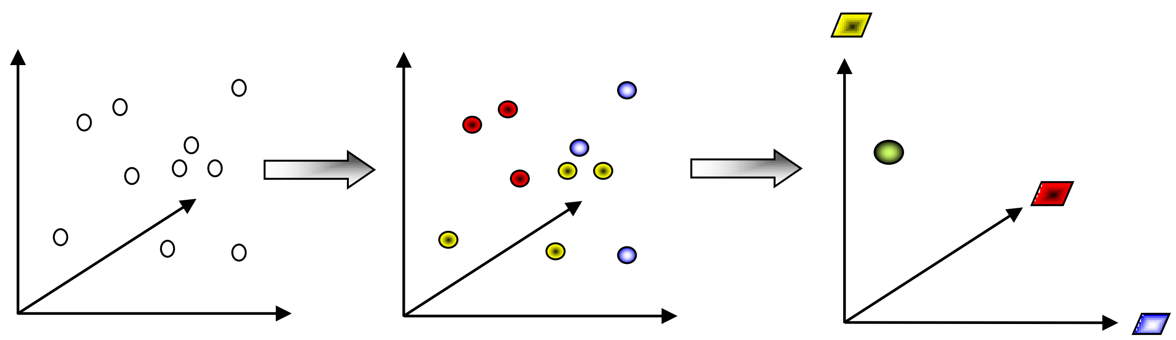
- C: The class attribute
- DB: The set of multi-instance objects O being labelled with class labels from C
- K: the set of *instance concepts*

Solution: Two Stage Classification.

1. Classifier 1: Learns a mapping KL from instance o_j to concept K_l : $KL(o_j) = K_l \in K$

=> Each multi-instance object O can be mapped to a distribution over the concepts K

2. Classifier 2: Learns a mapping CL from concept distribution to class $CL(O) = c_i \in C$



MI classification with unknown concepts

Input:

- *C*: The class attribute
- *DB*: The set of multi-instance objects *O* being labelled with class labels from *C*

Problem: The concepts for defining a class are unknown => training a classifier to predict instance concepts is not possible

Solution approaches:

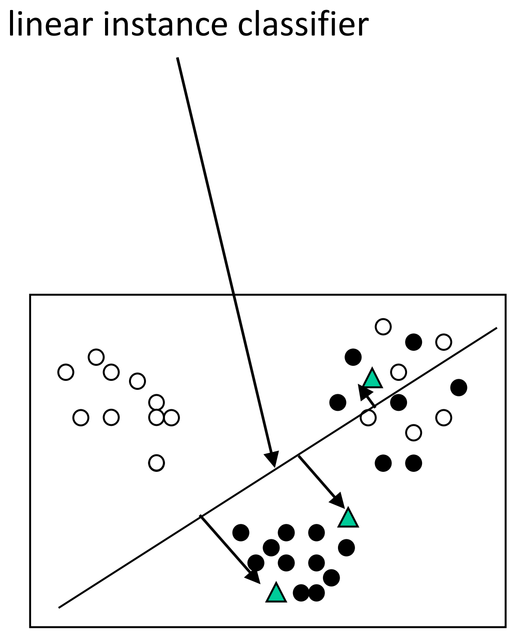
- Train an *instance classifier* predicting the likelihood that instance o_i is element of any multi-instance object *O* having a class c_j.
- Aggregate the prediction over all instances in O
  (assumption: O was generated by drawing n times with replacement)

Remark:

- methods depend on reliability of the confidence values
- method assume the independency of the instances (multinomial distribution)

General Multi-Instance Classification

Example: 2 classes, 3 “unknown“ concepts



- Trainings set for instance classifier

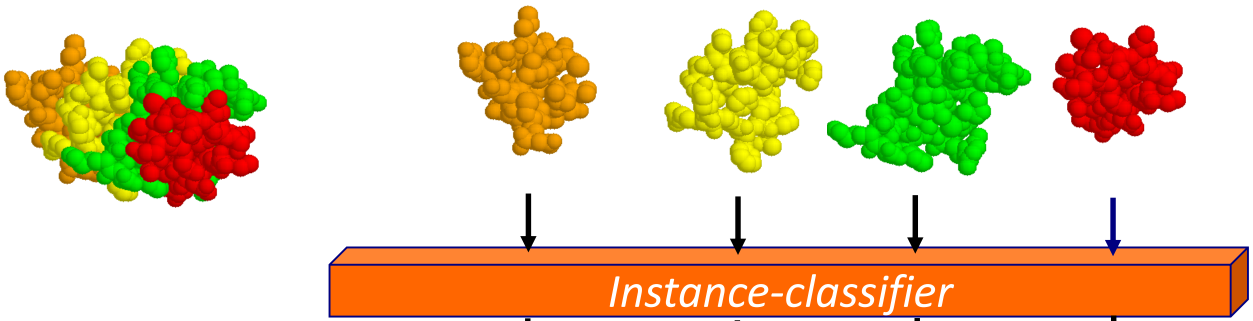
$$TR_A = \bigcup_{O_i \in DB} \{ o_j \in O_i \wedge CL(O_i) = A \}$$

- instances in concepts being typical for a class should be classified with a high confidence
- instances in ambiguous concepts should be classified with smaller confidence values

- the classifier often needs rather complex class borders (small bias but larger likelihood of overfitting)

General Multi-Instance Classification

Example: Combination of the instance predictions



conf. for instance	C1	0.6	0.05	0.4	0.2
	C2	0.4	0.95	0.6	0.8
conf. for cmpl. object	C1	**0.6**	**0.073**	**0.05**	**0.013**
	C2	**0.4**	**0.927**	**0.95**	**0.987**

0.987 C2

Confidence of *O* for class C_k: $\Pr[C_k \mid O] = \dfrac{\Pr[C_k] \cdot P[O \mid C_k]}{\sum_{i \in C} \Pr[C_i] \cdot P[O \mid C_i]}$ (Bayes theorem)

where $\Pr[O \mid C_k] = \prod_{I_i \in O} \Pr[I_i \mid C_k]$

Classical Multi-Instance Learning

Setting: There is one relevant concept K_{rel}. All objects containing at least one instance $o_i \in O$ *with* $K(o_i)=K_{rel}$ *belong to class "relevant".*

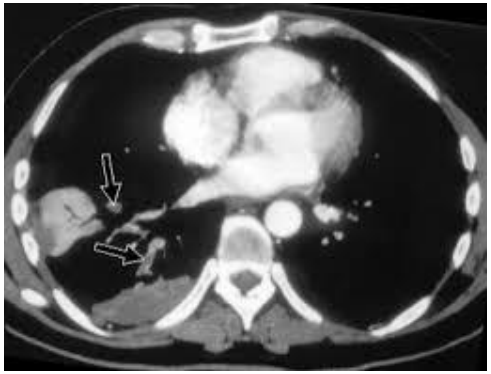
Examples:

1- Does a molecule smell like musk? [Dietterich et al. 1998]

Molecules are described as sets of spatial conformations. If the molecule has a spatial conformation matching the musk receptor, it has a musky smell.

2- Search for lung embolisms

CT scanner generates a set of suspicious areas in the lung. If a least one of them is a lung embolism the patient needs treatment.



http://medicalpicturesinfo.com/pulmonary-embolism/

Classical Multi-Instance Learning

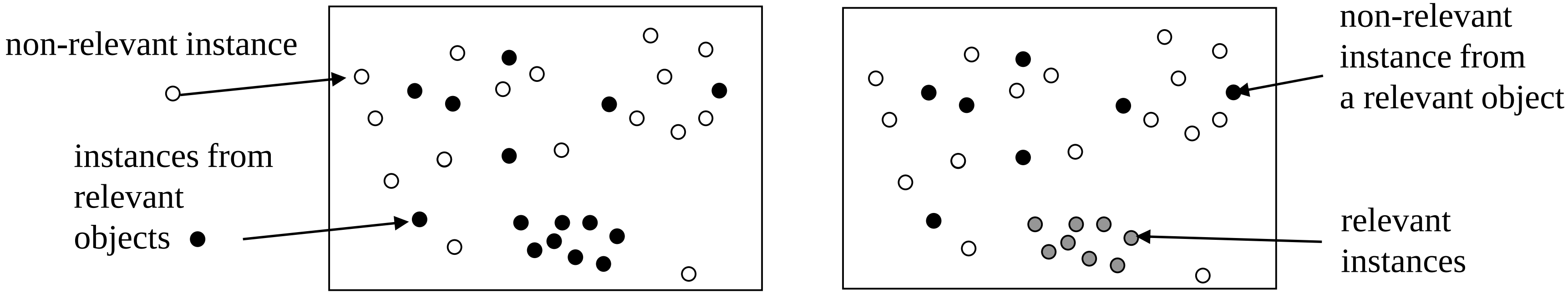
Approach: Classify all single instances

=> if one is relevant, the complete object is relevant as well.

Problem: Labeled instances are only reliable for the non-relevant class.

Remark: Multi-instance learning corresponds to learning a classifier for the relevant concept

- all instances of objects in the non-relevant class cannot be part of the relevant concept
- instances of objects from the relevant class can belong to both concepts
- at least one instance for each object has to belong to the relevant concept

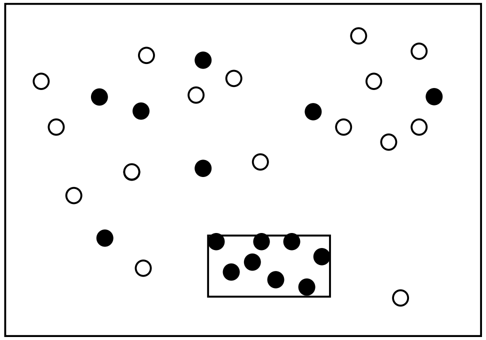


Classical Multi-Instance Learning

Approaches to classical multi-instance learning

Find a region in the feature space which contains only relevant instances (no negative samples) and contains at least one instance from each relevant object.

- Solution space is constructed by all sets of instances containing one instance from each objects. (assume: k objects having n instances => n^k solutions)



- Each solution can be used to demark the relevant area of the feature space
- It cannot be guaranteed that there is one area without any non-relevant samples
- Irrelevant features, learning bias etc. also influence the quality

Classical Multi-Instance Learning

Expectation Maximization Diverse Density classification (EM-DD)

Idea: Describe the relevant concept by an instance *h* and weights s_d for weighting the influence of the features $D=\{d_1,..,d_m\}$.

Predicting the object class is done by the max confidence of any instance in O:

$$Label(O_1 \mid h, \vec{s}) = \max_j \left\{ \exp\left[-\sum_{i=1}^{m} \left(s_i \left(o_{j,i} - h_i \right) \right)^2 \right] \right\}$$

where l=0 codes „relevant“ and l=1 codes „irrelevant“

The quality of the classifier for set DB can be described by the Negative Logarithmic Diverse Density (*NLDD*) :

$$NLDD(h, \vec{s}, DB) = \sum_{i=1}^{|DB|} \left(-\log\left(\left| l_i - Label\left(O_i \middle| h, \vec{s}\right) \right| \right) \right)$$

EM-DD training algorithm:

init h //e.g. centroid of a samples of the relevant instances, s_i = 0.1

While($NLDD_{new} < NLDD_{old}$)

FOR ALL O_i in DB mit CL(O_i) = „relevant“ DO

$$o_i^* = \arg\max_{o_{ij} \in O_i} \left(Label\left(O_i \middle| h, \vec{s} \right) \right)$$

$$h' = \arg\max_{h \in H} \prod_{i=1}^{n} \Pr\left(l_i \middle| h, \vec{s}, o_i^* \right)$$

// optimization of weights
// by gradient descent

$NLDD_{old} = NLDD_{new}$

$NLDD_{new}$ = NLDD(h‘,D)

h = h‘

return *h*

Remark: $\Pr\left(l_i \middle| h, \vec{s}, o_i^* \right) = \exp\left[-\sum_{i=1}^{m} \left(s_i \left(o_i^* - h_i \right) \right)^2 \right]$

Multi-Instance Classifier

Conclusions:

general Multi-Instance Classification

- only a view dedicated approaches are published
- most approaches are based on distance measures or kernels

Classical Multi-Instance Learning

- Large effort in the research community
  - Citation-kNN and Bayes-kNN (nearest neighbor-based approaches)
  - Multi-Instance decision trees and rule-based classifiers
  - Neural Networks for multi-instance objects

  $\Rightarrow$ EM-DD (showed most promising results without any meta-learning)
- General benchmark is the musk use case !!

  More practical results showed good results for general MI-learners

Multi-Instance Data

- Multi-Instance Data
- Aggregation-based Approaches
- Distance and Similarity Measures
- Multi-Instance Classification
- Clustering Multi-Instance Objects

Multi-Instance Clustering

- MI-Objects can be clustered based on distance-based methods such as k-medoid, DBSCAN, OPTICS, etc.
  - only applicable to purely distance-based methods (cluster model ?) (e.g., k-Means cannot be used due to the lack of centroids)
  - selecting a well-suited distance measure is very important
  - This approach does not yield expressive cluster models

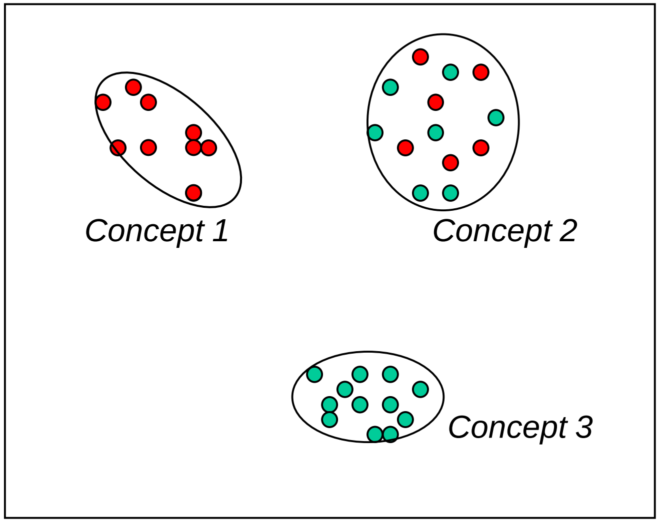
- **Idea:** Use the concept model from classification → Concept-based multi-instance clustering
  - Instances belong to certain concepts
  - MI objects can be described by distribution over the different concepts

  => clusters can be composed by objects having similar concept distributions

Concept-Based Multi-Instance Clustering

Idea:

- Each instance $o_i \in O$ belongs to a concept.
- Multi-instance (MI-)clusters are distributions over the set of concepts.



MI-Cluster1 contains instances from concept 1 and concept 2.

MI-Cluster2 contains instances from concept 2 and concept 3.

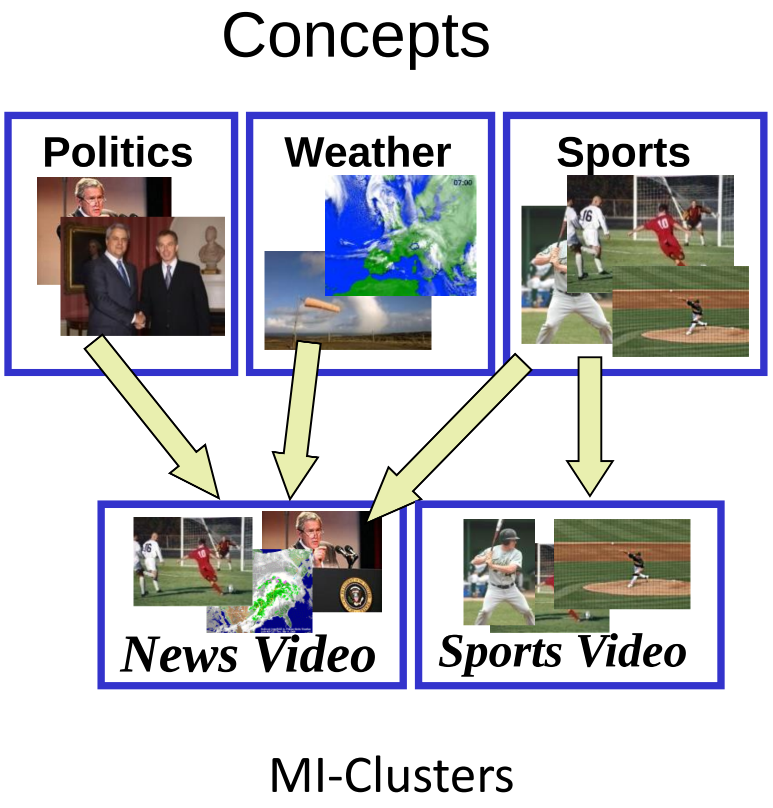
Description of a MI-cluster = cluster description of the contributing concepts.

Concept-Based Multi-Instance Clustering

Example: Video Data

- Videos are represented as sets of Shots/Scenes (MI objects)
- Shots belong to a concept (e.g. sports, weather,..)

- An MI-cluster contains video with shots belonging to the same concepts:
  - Sport-videos contain sports shots.
  - Weather-videos contain weather shots.
  - News videos contains sports, weather, politics,...-shots.



A MI-EM Algorithms (Kriegel et al, 2006)

Instance set:

- *DB*: a set of MI-objects $o = \{i_1, \ldots, i_k\}$

- I_{DB} : the instance set of *DB*, is the union of all multi-instance objects

$$I_{DB} = \bigcup_{DB} o$$

Instance Model:

An Instance Model *IM* for the instance set I_{DB} is given by a mixture model of *k* statistical processes that can be described by:

- a prior probability $\Pr[k_j]$ for each process k_j.
- the necessary parameters for each process k_j, e.g. a mean vector μ_j and a covariance matrix Σ_j for Gaussian processes.

These k processes correspond to the *concepts*.

A MI-EM Algorithm

Multi-Instance Cluster Model M

- A set C of clusters over the instance model IM.

Each MI-cluster $c \in$ C is described as follows:

- a prior probability $Pr\,[c]$,
- a cardinality distribution $Pr\,[Card(o)|c]$
- a conditional distribution of concepts $Pr\,[\,i \in k \mid i \in o \in c\,]$ (shortly: $Pr\,[k|c]$) for each concept k in IM.

The probability of an object o in the model M is computed as follows:

$$\Pr[o] = \sum_{c \in C} \Pr[c] \cdot \Pr[Card(o) \mid c] \cdot \prod_{i \in o} \prod_{k \in IM} \Pr[k \mid c]^{\Pr[k|i]}$$

the a-posteriori probability of o and cluster c is given as:

$$\Pr[c \mid o] = \frac{1}{\Pr[o]} \Pr[c] \cdot \Pr[Card(o) \mid c] \cdot \prod_{i \in o} \prod_{k \in IM} \Pr[k \mid c]^{\Pr[k|i]}$$

A MI-EM Algorithm

Example: 2 MI-Cluster

Cluster 1: ▲

50 % prior probability

expected number if instances: 2 3

concept1	concept2	concept3
0.2 1	0.01	0.79 2

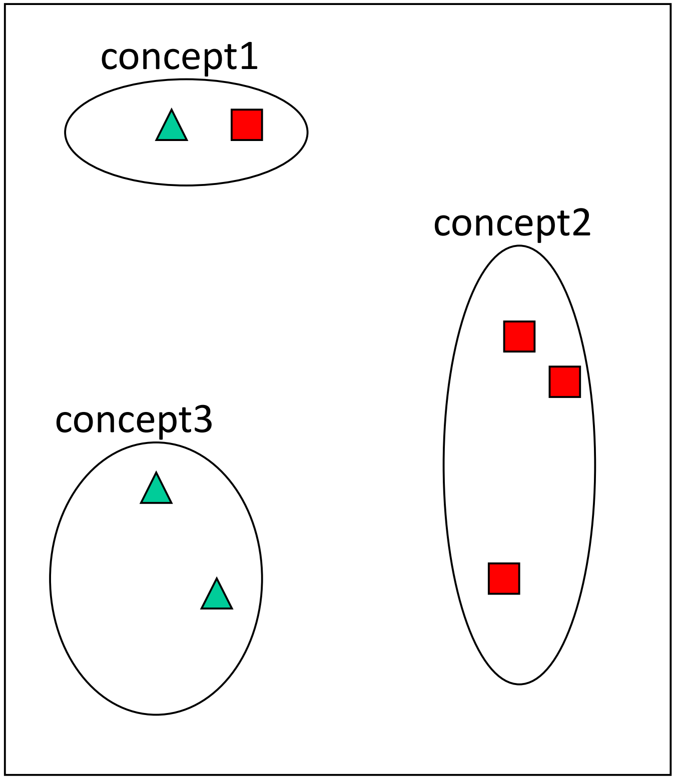
Cluster 2: ■

50 % prior probability

expected number if instances : 5 4

concept1	concept2	concept3
0.1 1	0.89 3	0.01

Instance Model *IM*



A MI-EM Algorithm

Overview of the algorithm:

- **Step 1:** Compute a mixture model (*IM*) on the instance set I_{DB} *(build concepts)*

- **Step 2:** Compute an initial model for clustering MI objects based on their concept distribution

- **Step 3:** Use EM to optimize the cluster model

A MI-EM Algorithm

Step 1: Derive a mixture model for the instance set

Build I_{DB} and use EM-clustering to derive *IM* (the concepts).

Step 2: Find a start partitioning of MI-objects

- For each MI-object *O* in DB build a "Confidence Summary Vector" *CSV*(*O*).
  - it is a k-dimensional vector, k=#concepts
  - the *j*-th component of CSV (*O*) is defined as:

$$CSV_j(O) = \sum_{i \in O} \Pr[k_j] \cdot \Pr[i \mid k_j]$$

- Use k-means to group the *CSVs* to an initial cluster model

A MI-EM Algorithm

Step 3: Optimize the partitioning through EM

The start partitioning (step 2) is optimized using EM

E-Step: Compute the log-likelihood of the current model M.

$$E(M) = \sum_{o \in DB} \log \sum_{c_i \in M} \Pr[c_i \mid o]$$

M-Step: apply the following updates:

update prior probability of MI-cluster c_i: $W_{c_i} = \Pr[c_i] = \frac{1}{Card(DB)} \sum_{o \in DB} \Pr[c_i \mid o]$

update cardinality distribution: $l_{c_i} = \frac{\sum_{o \in DB} \Pr[c_i \mid o] \cdot Card(o)}{Card(DB)} \cdot \frac{1}{MAXLENGTH}$

update concept distribution: $P_{k_j, c_i} = \Pr[k_j, c_i] = \frac{\sum_{o \in DB} \left(\Pr[c_i \mid o] \cdot \sum_{u \in o} \Pr[u \mid k_j] \right)}{\sum_{o \in DB} \Pr[c_i \mid o]}$

Summary: Multi-Instance Data Mining

- Aggregation is useful for homogeneous sets
- Multiple distance and similarity function for MI objects
- Distance measures can be plugged into various algorithms
- Selecting the right distance measure is essential to the success
- Concept-based approaches abstract from sets of instances to concepts and apply data mining to the concept distribution
- Concept-based approaches rely on a suitable set of concepts and methods to assign instances to these concepts

References

- Kriegel H.-P, Pryakhin A., Schubert M. : *An EM-Approach for Clustering Multi-Instance Objects,* Proc. 10th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD 2006), Singapore, 2006.
- Dietterich T.G., Lathrop R.H., Lozano-Perez T. : *Solving the Multiple Instance Problem with Axis-Parallel Rectangles,* Artificial Intelligence, vol. 89, num.1-2, Seiten 31-71, 1997
- Weidmann N., Frank E., Pfahringer B. : *A Two-Level Learning Method for Generalized Multi-instance Problems*. ECML 2003: S. 468-479
- Gärtner T., Flach P.A., Kowalczyk A., Smola A.j. : *Multi-Instance Kernels*, Proceedings of the 19th International Conference on Machine Learning, p. 179-186, 2002
- Zhang Q., Goldman S. : *EM-DD: An improved multiple-instance learning technique*. Neural Information Processing Systems 14, 2001.
- Eiter T., Mannila H. : *Distance Measures for Point Sets and Their Computation*. Acta Informatica, 34(2):103-133, 1997.
- Brecheisen S, Kriegel H.-P., Kröger P., Pfeifle M., Schubert M. : *Using Sets of Feature Vectors for Similarity Search on Voxelized CAD Objects*, Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'2003), San Diego, CA, 2003