UNIVERSITÄT BERN

CLASSIFICATION AND CLUSTERING OF GRAPHS BASED ON DISSIMILARITY SPACE EMBEDDING

Horst Bunke and Kaspar Riesen
{bunke,riesen}@iam.unibe.ch
Institute of Computer Science and Applied Mathematics
University of Bern, Neubrückstrasse 10, CH-3012 Bern (Switzerland)
http://www.iam.unibe.ch/∼fki/

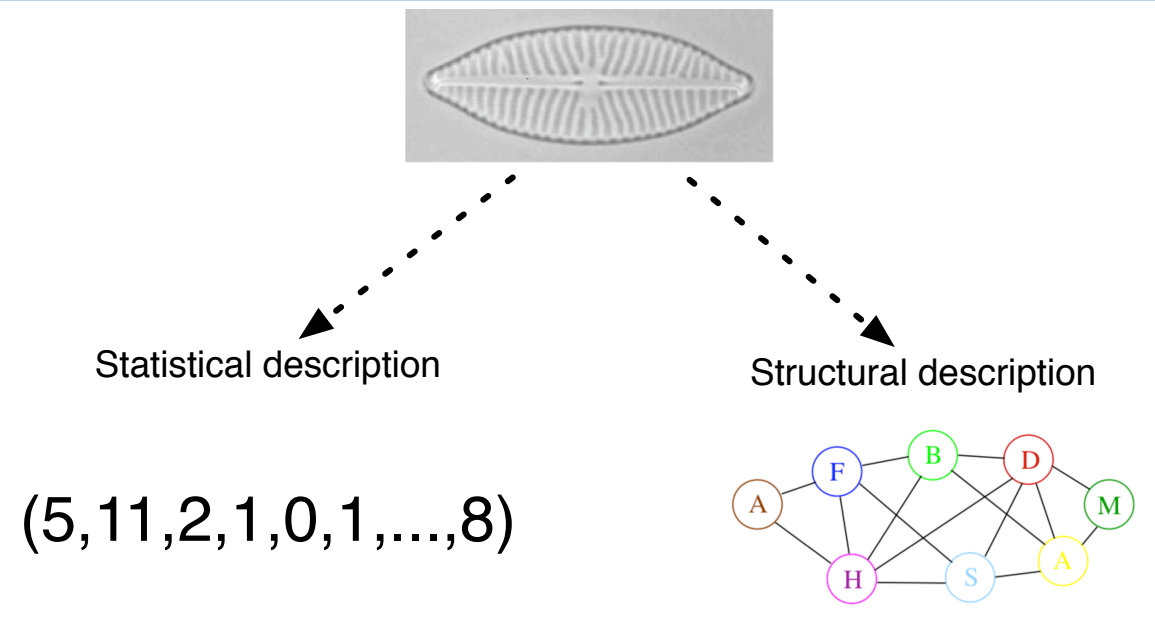
Acknowledgments

- Michel Neuhaus
- Swiss National Science Foundation
- Swiss National Science Foundation ’National Centers of Competence in Research’ Program
- Bob Duin, Ela Pekalska

Contents

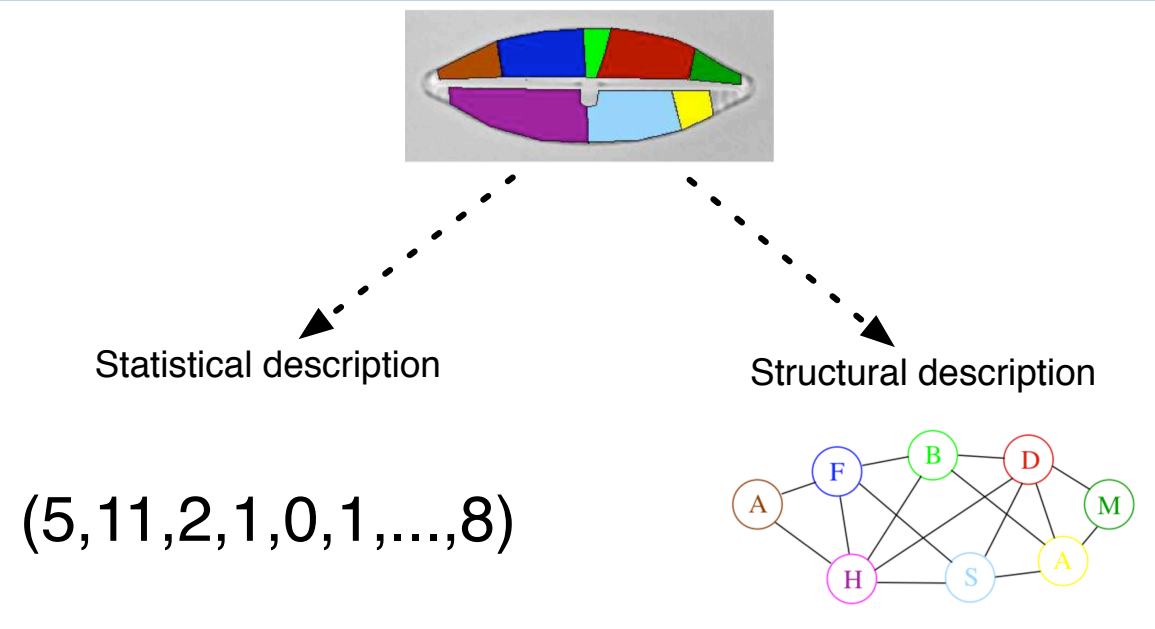
Introduction

- Traditional subdivision of Pattern Recognition:



Introduction

- Traditional subdivision of Pattern Recognition:



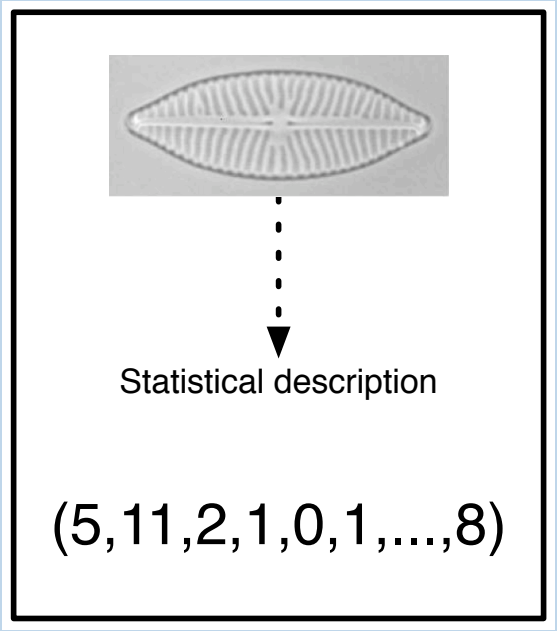
Statistical Approach

Advantages:

- Theoretically well founded
- Many powerful algorithms available

Disadvantages:

- Dimension of feature vectors fixed
- Only unary feature values, but no relations can be modeled



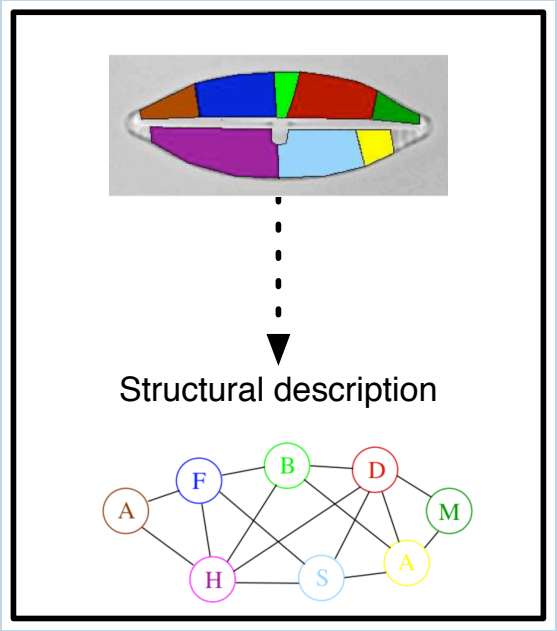
Structural Approach

Advantages:

- Representation size is variable
- Higher representational power (structural relationships)

Disadvantages:

- Lack of mathematical structure in the graph domain
- Lack of algorithmic tools



Vectors vs. Graphs

	vectors	graphs
representational power	-	+
available tools	+	-

Vectors vs. Graphs

UNIVERSITÄT BERN

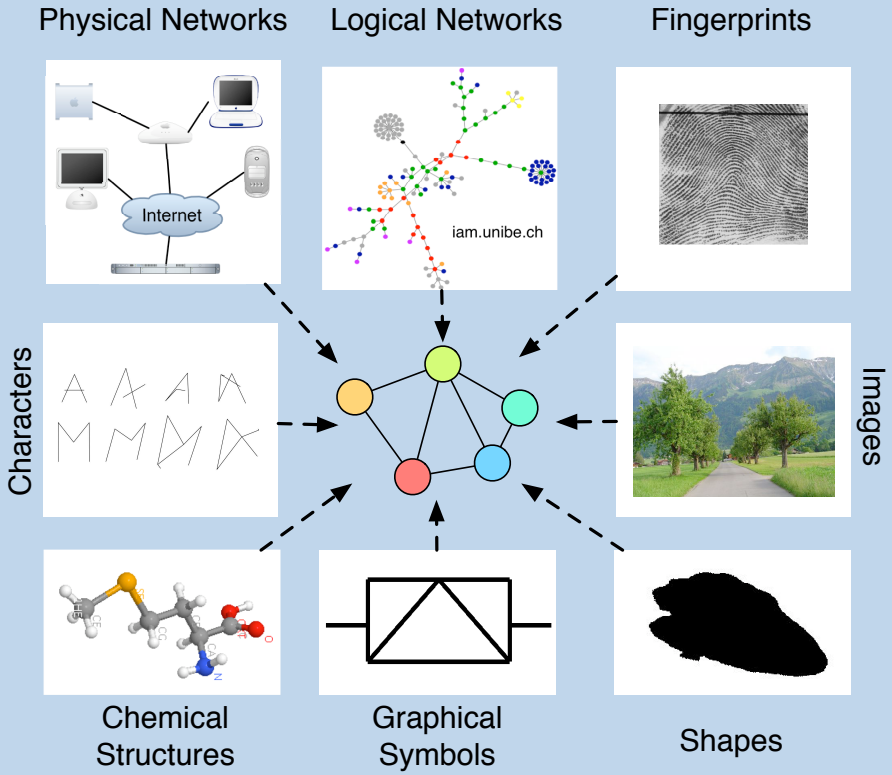
	vectors	graphs
representational power	-	+
available tools	+	+

- Embedding graphs in vector spaces makes available all algorithmic tools developed for feature based object representations
- Close relationship to graph kernels

Contents

Graph Based Representation

- A graph is defined by $g = (V, E, \mu, \nu)$, where
  - V is the finite set of nodes
  - $E \subseteq V \times V$ is the set of edges
  - $\mu : V \rightarrow L$ is the node labeling function
  - $\nu : E \rightarrow L$ is the edge labeling function



Graph Embedding

- Definition: Let $\mathcal{G}$ be a set of graphs. A **graph embedding** is a function $\varphi : \mathcal{G} \to \mathbb{R}^n$ mapping graphs to n-dimensional vectors, i.e.,

$$\varphi(g) = (x_1, \ldots, x_n)'$$

- Previous work in graph embedding:
  - Spectral methods [Kosinov, Caelli, 2004], [Luo, Wilson and Hancock, 2005], [Shokoufandeh et al. 2005]
  - Heat kernel [Bai and Hancock, 2005]
  - Riemannian approach [Robles-Kelly and Hancock, 2007]
  - Quantum commute times [Qiu and Hancock, 2007]
  - Graph characteristics using the Gauss-Bonnet Theorem [ElGhawalby and Hancock, 2008]
  - Characteristic polynomial analysis [Ren, Wilson and Hancock, 2009]

Graph Embedding

Characteristics of previous works:

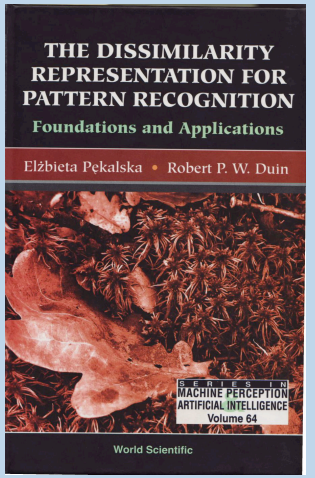
- Solid methodological foundation
- Often restricted to special classes of graphs
- Robustness to noise may be an issue

Contribution of this work:

- Applicable to any type of graphs
- Robust against various types of distortions, due to the use of graph edit distance

Background

- Method described in this paper is inspired by dissimilarity space embedding proposed by Duin and Pekalska
- Power of this approach mainly demonstrated on feature vector representations, although much more generally applicable
- Later investigated for string representations
- In this paper extension to graphs



Contents

Graph Edit Distance

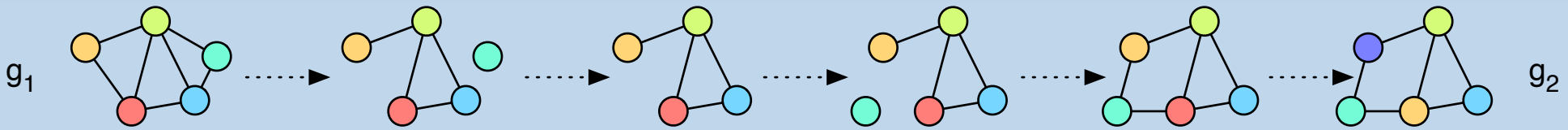
Define the dissimilarity of graphs by the minimum amount of distortion that is needed to transform one graph into another

- **Distortions** e_i**:** deletions, insertions, substitutions of nodes and edges
- **Edit path** $S = e_1, \ldots, e_n$**:** a sequence of edit operations that transform g_1 into g_2
- **Cost function:** Measuring the strength of a given distortion
- **Edit distance** $d(g_1, g_2)$**:** Minimum cost edit path between two graphs

Graph Edit Distance

Define the dissimilarity of graphs by the minimum amount of distortion that is needed to transform one graph into another

- **Distortions** e_i**:** deletions, insertions, substitutions of nodes and edges
- **Edit path** $S = e_1, \ldots, e_n$**:** a sequence of edit operations that transform g_1 into g_2
- **Cost function:** Measuring the strength of a given distortion
- **Edit distance** $d(g_1, g_2)$**:** Minimum cost edit path between two graphs



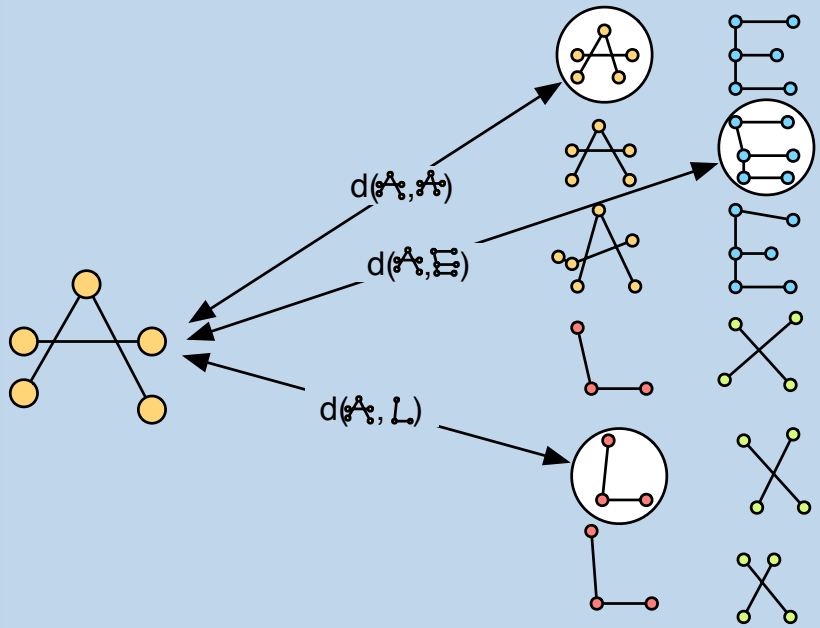
Graph Embedding

- Graph set: $\mathcal{G} = \{g_1, \ldots, g_t\}$
- Graph edit distance: $d(g_i, g_j)$
- Prototype set: $P = \{p_1, \ldots, p_n\}$
- The mapping

$$\varphi_n^P : \mathcal{G} \rightarrow \mathbb{R}^n$$

is defined as the function

$$\varphi_n^P(g) \mapsto (d(g, p_1), \ldots, d(g, p_n))$$



Distance in the Embedding Space

$$
\begin{aligned}
||\varphi_n^P(g) - \varphi_n^P(g')|| &= \langle \varphi_n^P(g), \varphi_n^P(g) \rangle + \langle \varphi_n^P(g'), \varphi_n^P(g') \rangle - 2\langle \varphi_n^P(g), \varphi_n^P(g') \rangle \\
&= \sum_{i=1}^{n} d(g, p_i)^2 + \sum_{i=1}^{n} d(g', p_i)^2 - 2\sum_{i=1}^{n} d(g, p_i) d(g', p_i) \\
&= \sum_{i=1}^{n} (d(g, p_i) - d(g', p_i))^2 \\
&\leq \sqrt{n} \cdot d(g, g')
\end{aligned}
$$

- If two graphs are close in the graph domain, their maps will be close in the vector space

Prototype Selectors

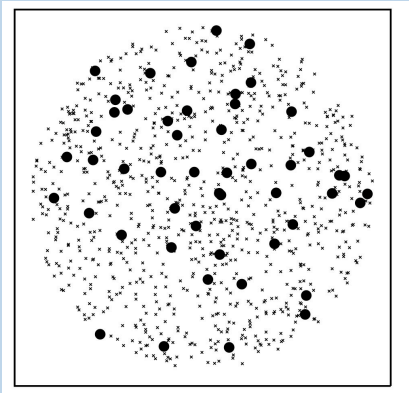
UNIVERSITÄT BERN

- Problem: Appropriate choice of the prototype set $P = \{p_1, \ldots, p_n\}$ that leads to a good performance of the resulting classifier in the vector space
- We distinguish between **class-wise** and **class-independent** selection:
  - class-independent: Selection is executed over the whole graph set to get n prototypes
  - class-wise: Selection is performed individually for each of the k different classes (l_i prototypes per class c_i such that $\sum_{i=1}^{k} l_i = n$)

Random Prototype Selector (*rps*)

UNIVERSITÄT BERN

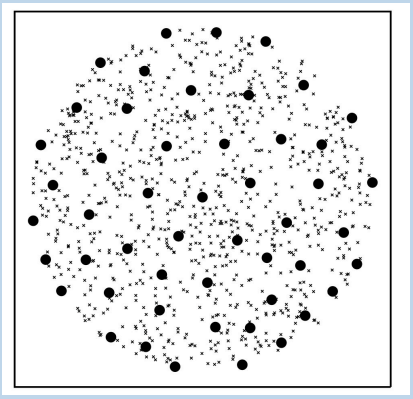
- Selects randomly n prototypes from the graph set $\mathcal{G}$



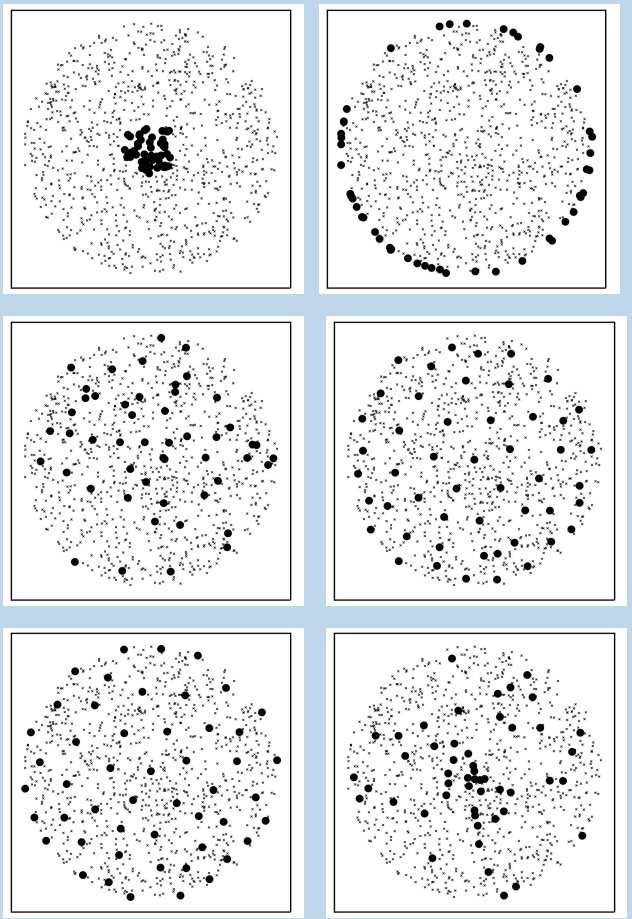
Spanning Prototype Selector (*sps*)

- The first prototype selected is the set median graph of $\mathcal{G}$
- Each additional prototype selected by the spanning prototype selector is the graph the furthest away from the already selected prototype graphs

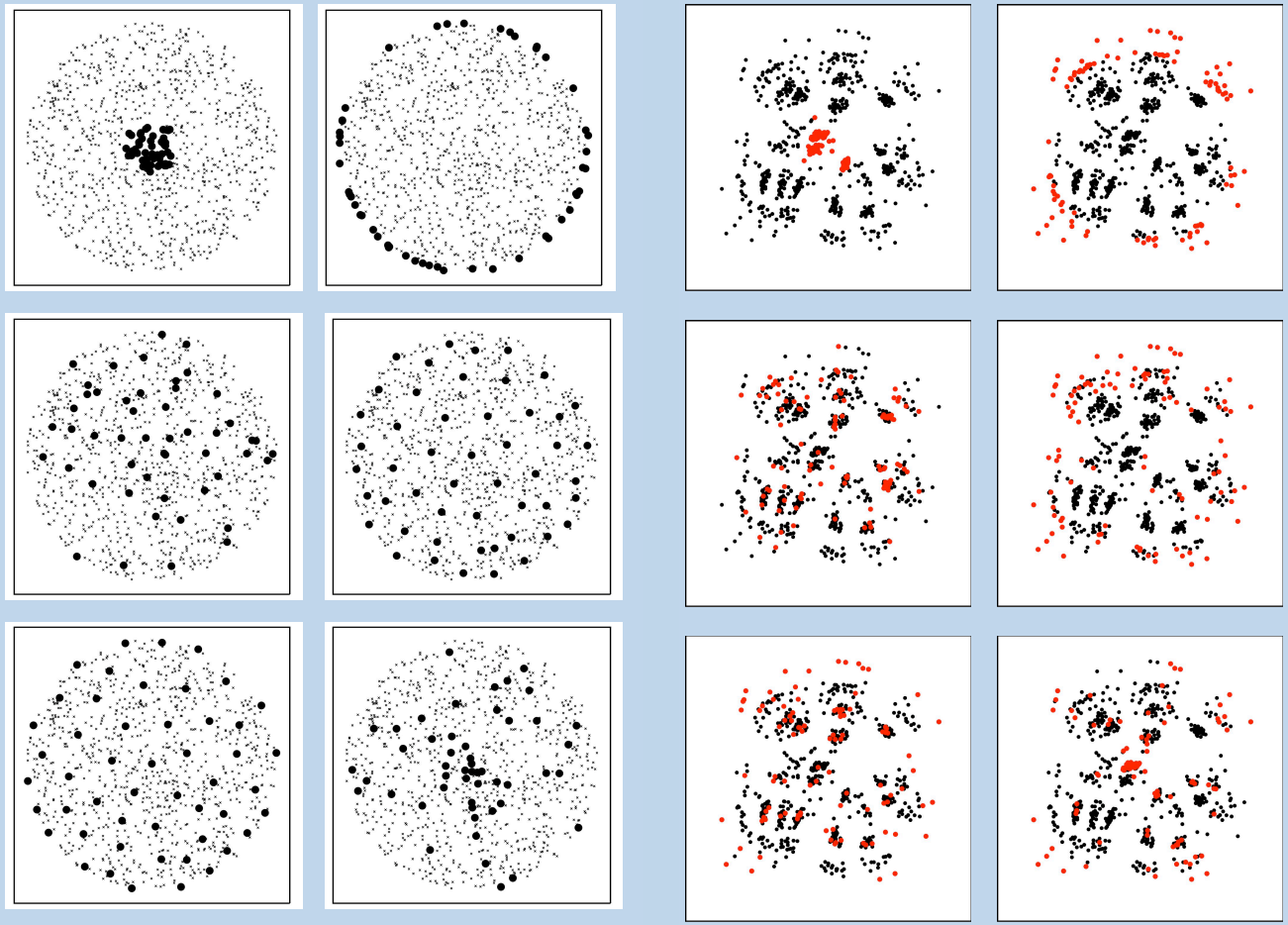
$$P_i = \begin{cases} median(\mathcal{G}) & \text{if } i = 1 \\ P_{i-1} \cup \{p_i\} & \text{if } 1 < i \leqslant n \text{ , where } p_i = \underset{g \in \mathcal{G} \setminus P_{i-1}}{\operatorname{argmax}} \min_{p \in P_{i-1}} d(g,p) \end{cases}$$



Prototype Selection



Prototype Selection on Real Data

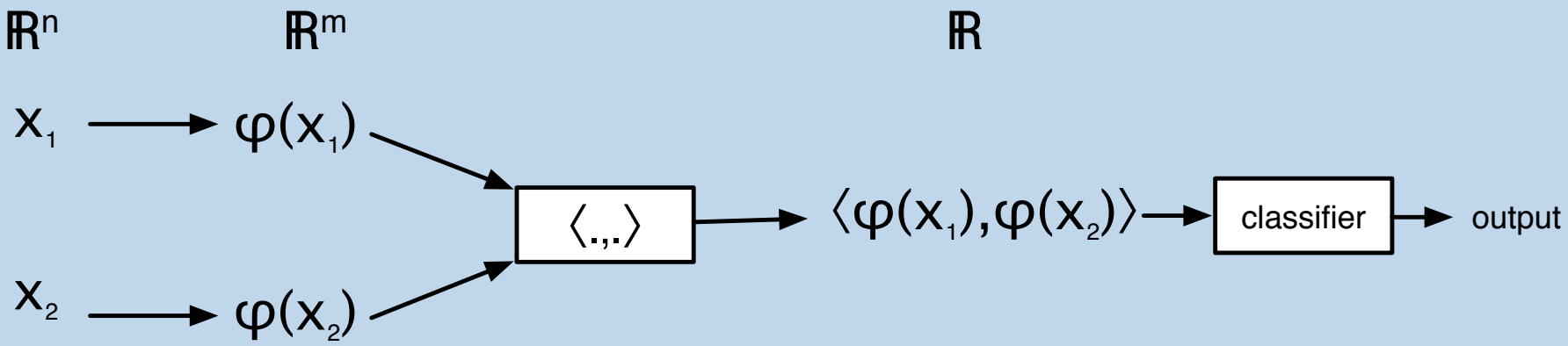


Relation to Graph Kernels

- Kernel methods have emerged as a promising new research direction
- Kernel methods have been known for long, but their potential for intelligent data analysis has been discovered only about 10 years ago
- Originally kernel methods have been developed for feature vectors
- Recently it has been demonstrated that they are applicable to graphs as well
- Graph kernel methods make available a large spectrum of methods from computational intelligence, pattern recognition, machine learning, etc.

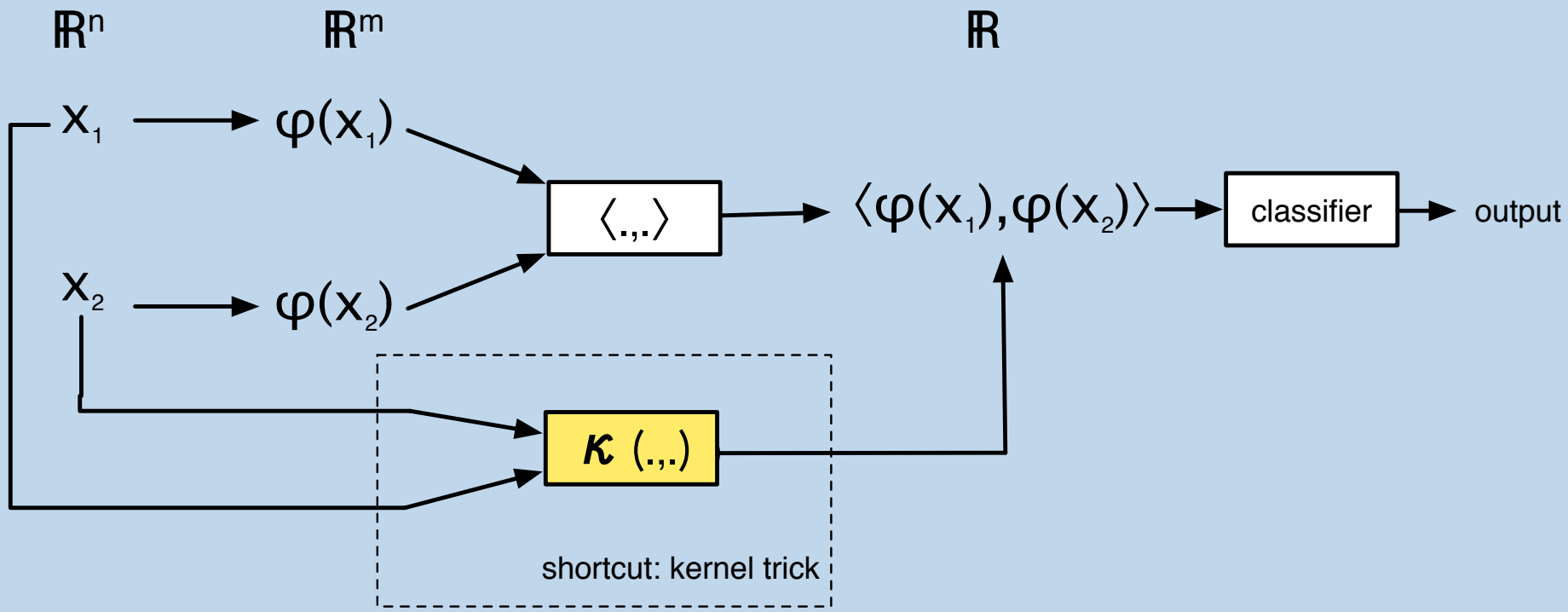
Kernel Trick

- Illustration of the *kernel trick*:



Kernel Trick

- Illustration of the *kernel trick*:



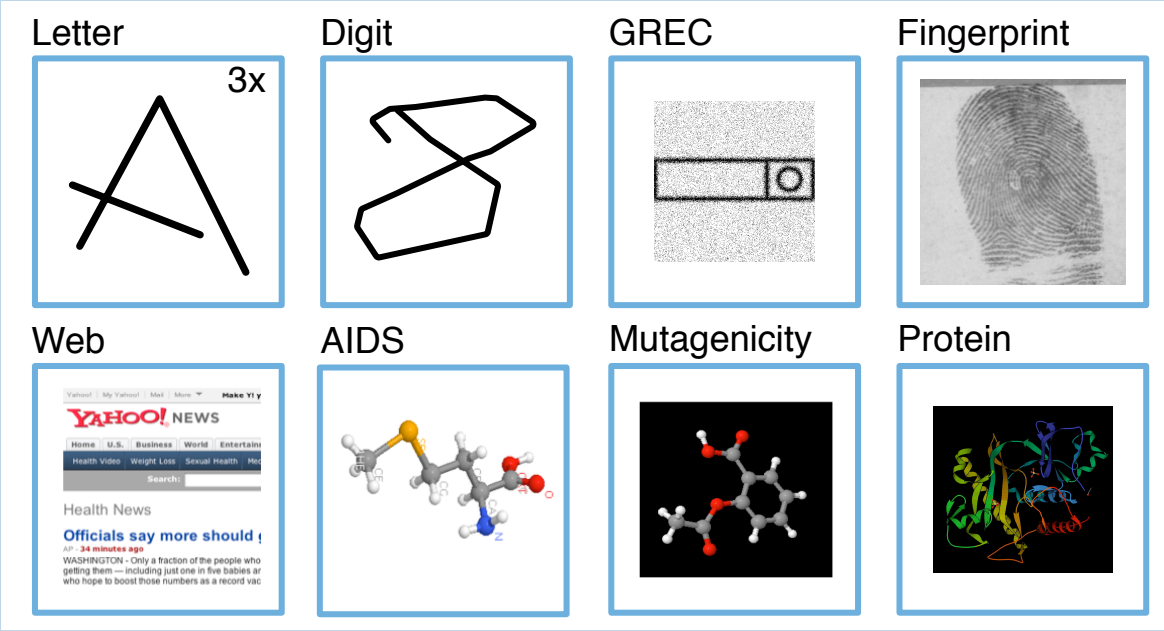
Graph Kernel

- **Definition**: Let $g, g' \in \mathcal{G}$ be graphs and $\varphi : \mathcal{G} \to \mathbb{R}^n$ a function with $n \in \mathbb{N}$. A graph kernel function is a mapping $\kappa : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ such that $\kappa(g, g') = \langle \varphi(g), \varphi(g') \rangle$. □
- If a classifier can be kernelized, it can be run with scalar products resulting from graphs
- Consequently, all kernel machines can be applied to graphs
- The proposed graph embedding is even more general: it makes not only all kernel machines, but all algorithms from statistical pattern recognition available for graphs

Experimental Evaluation: Classification

- Aim: compare classifiers in the embedding space with classifiers in the graph domain
- Use data sets with a wide range of different characteristics (number of classes, size of graphs, type of labels, . . .)
- Reference classifiers:
  - k-NN classifier in graph domain (k-NN)
  - SVM on graph edit distances: $\kappa(g, g') = \exp\left(-\gamma d(g, g')\right)$ with $\gamma > 0$ (Sim)
- Classifier in the vector space:
  - SVM
- Meta parameters of all systems optimized on a validation set: number of prototypes, prototype selection procedure

Data Sets



- Each set is divided into three disjoint subsets for training, validation, and testing.
- All data sets have been made publicly available under `www.iam.unibe.ch/fki/databases/iam-graph-database`.

Data Sets Characteristics

Data set	size (*tr*, *va*, *te*)	$\lvert\Omega\rvert$	node labels	edge labels
Letter (*low*)	750, 750, 750	15	(x, y) coord.	none
Letter (*medium*)	750, 750, 750	15	(x, y) coord.	none
Letter (*high*)	750, 750, 750	15	(x, y) coord.	none
Digit	1,000, 500, 2,000	10	(x, y) coord.	none
GREC	836, 836, 1,628	22	Type/(x, y) coord.	Type
Fingerprint	500, 300, 2,000	4	none	Angle
AIDS	250, 250, 1,500	2	Chem. symbol	none
Mutagenicity	500, 500, 1,500	2	Chem. symbol	none
Protein	200, 200, 200	6	Type/AA-seq.	Type/Length
Webpage	780, 780, 780	20	Word/Frequency	Type(s)

Data Sets Characteristics (cont'd)

Data set	$\varnothing\ \|V\|/\|E\|$	max $\|V\|/\|E\|$	balanced
Letter (*low*)	4.7/3.1	8/6	yes
Letter (*medium*)	4.7/3.2	9/7	yes
Letter (*high*)	4.7/4.5	9/9	yes
Digit	8.9/7.9	17/16	yes
GREC	11.5/12.2	25/30	yes
Fingerprint	5.4/4.4	26/25	no
AIDS	15.7/16.2	95/103	no
Mutagenicity	30.3/30.8	417/112	yes
Protein	32.6/62.1	126/149	yes
Webpage	186.1/104.6	834/596	no

Results using Prototype Selection

	ref. systems		embedding method
Data Set	k-NN	sim	sps-c
Letter low	99.3	99.6	99.3
Letter medium	94.4	94.9	94.9
Letter high	89.1	92.9	92.3 ①
Digit	97.4	98.1	98.6 ①
Fingerprint	79.1	82.0	82.0 ①
GREC	82.2	71.6	92.4 ①
AIDS	94.9	97.0	98.1 ①
Mutagenicity	66.9	68.6	71.6 ①
Protein	68.5	68.5	73.0
Webpage	80.6	82.9	82.7 ①

- Compared to k-NN: 9 improvements (7×①), 0 deteriorations (0×❶).

Results using Prototype Selection

	ref. systems		embedding method
Data Set	k-NN	sim	sps-c
Letter low	99.3	99.6	99.3
Letter medium	94.4	94.9	94.9
Letter high	89.1	92.9	92.3
Digit	97.4	98.1	98.6 ②
Fingerprint	79.1	82.0	82.0
GREC	82.2	71.6	92.4 ②
AIDS	94.9	97.0	98.1 ②
Mutagenicity	66.9	68.6	71.6 ②
Protein	68.5	68.5	73.0
Webpage	80.6	82.9	82.7

- Compared to sim: 5 improvements (4×②), 3 deteriorations (0×❷).

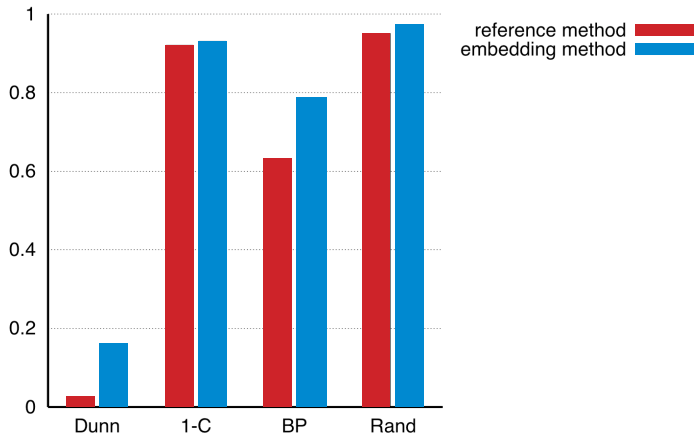
Experimental Evaluation: Clustering

- Aim: compare k-means clustering in the embedding space and in the graph domain
- Reference system:
  - k-medians clustering in graph domain (k-med)
- System in the vector space:
  - k-means algorithm
- Meta parameters optimized similarly to classification experiment

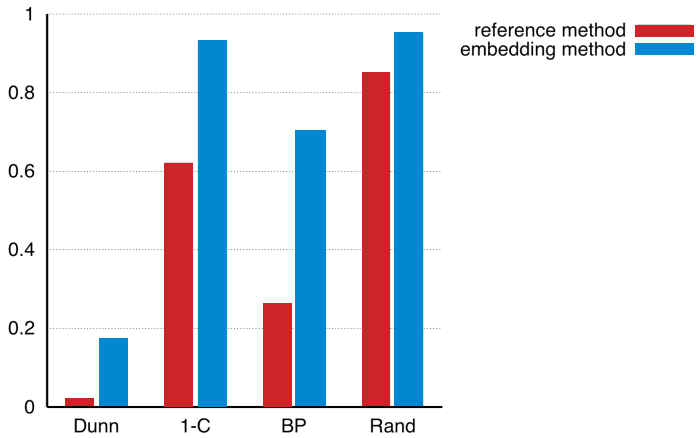
Evaluation of Clustering Algorithms

- Dunn- and C-Index (no ground truth needed)
  - Measure both the compactness of each individual cluster and the seperation of different clusters.
- Rand- and Bipartite-Index (if ground truth is available)
  - Compare the produced clusters with the classes in the ground truth.

Clustering Results

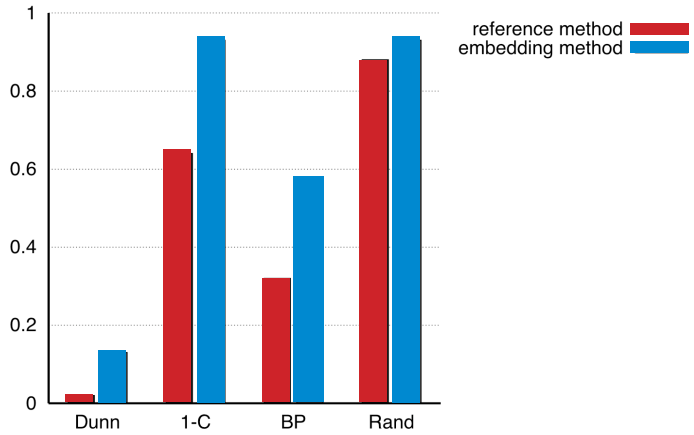


Letter low

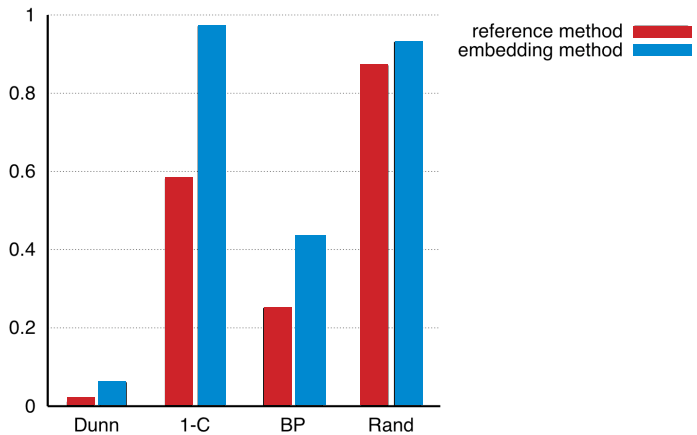


Letter medium

Clustering Results

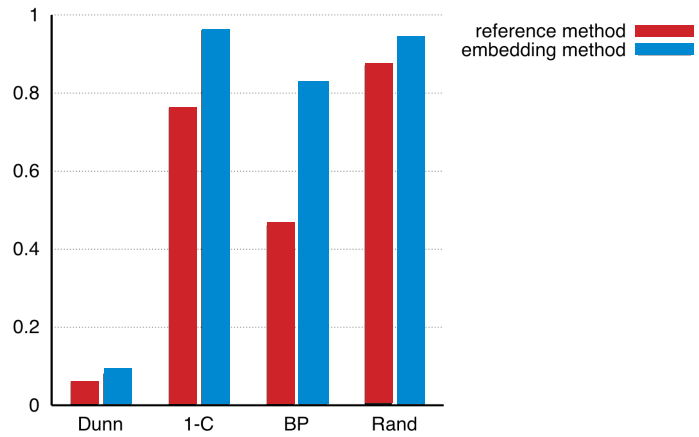


Letter high

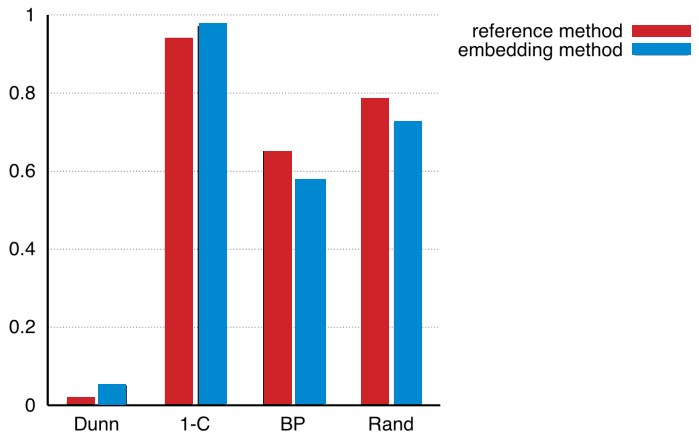


GREC

Clustering Results

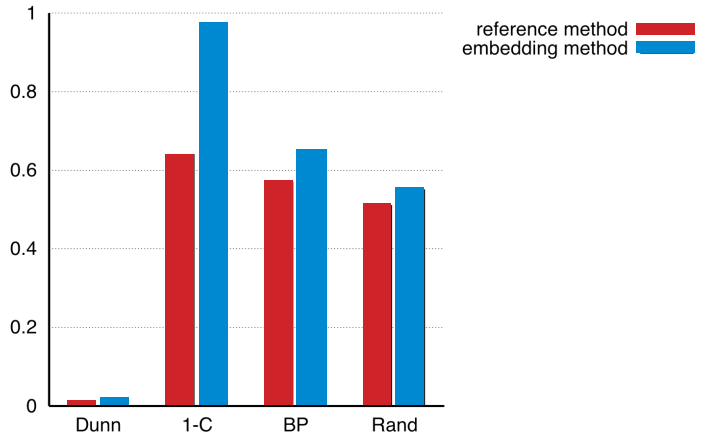


Digit

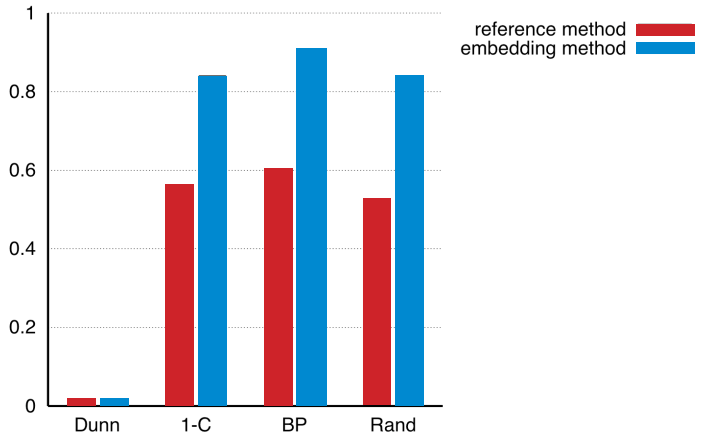


Fingerprint

Clustering Results

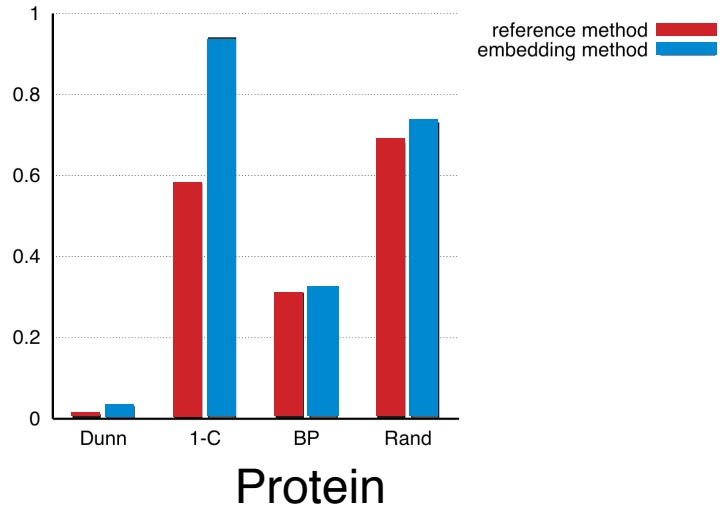


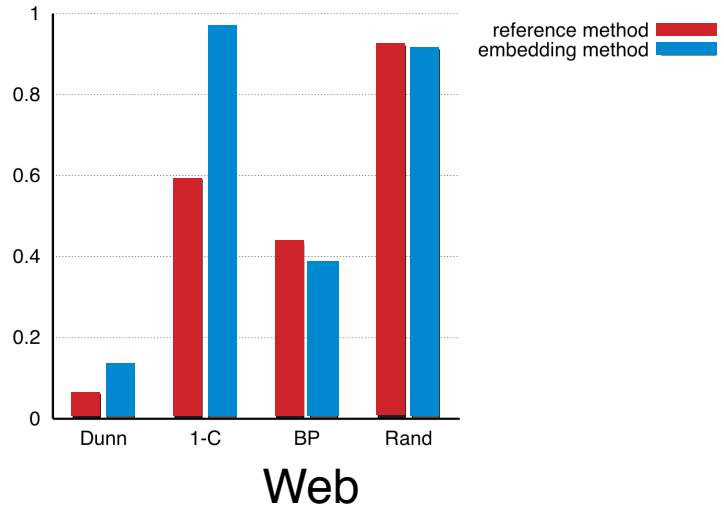
Mutagenicity



AIDS

Clustering Results



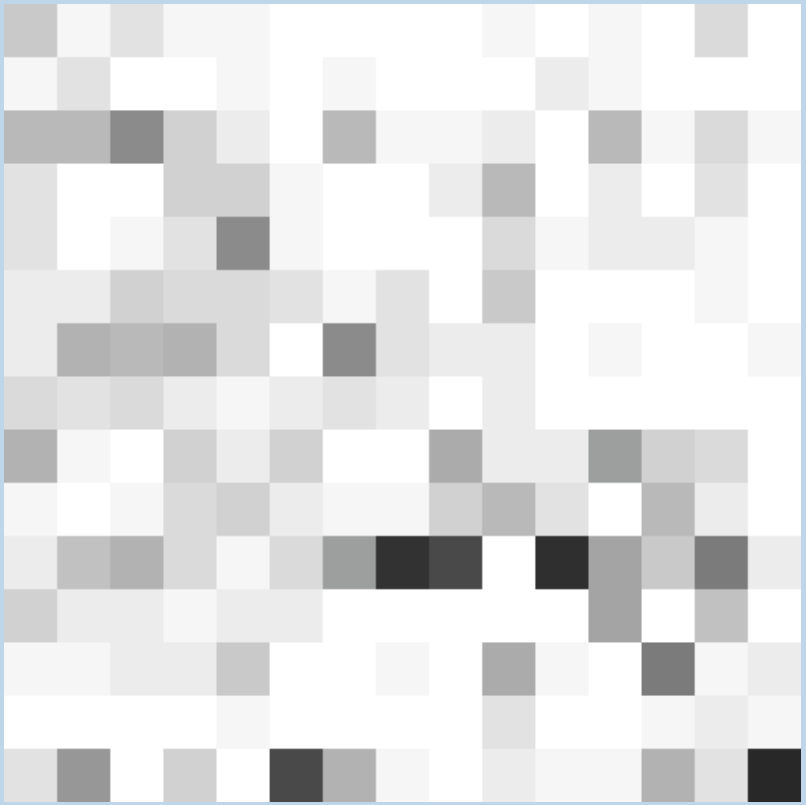


Clustering Results Overview

Validation index	wins	ties	losses
Dunn	9	1	0
C	10	0	0
BP	8	0	2
Rand	8	0	2

Confusion Matrix on Letter Data

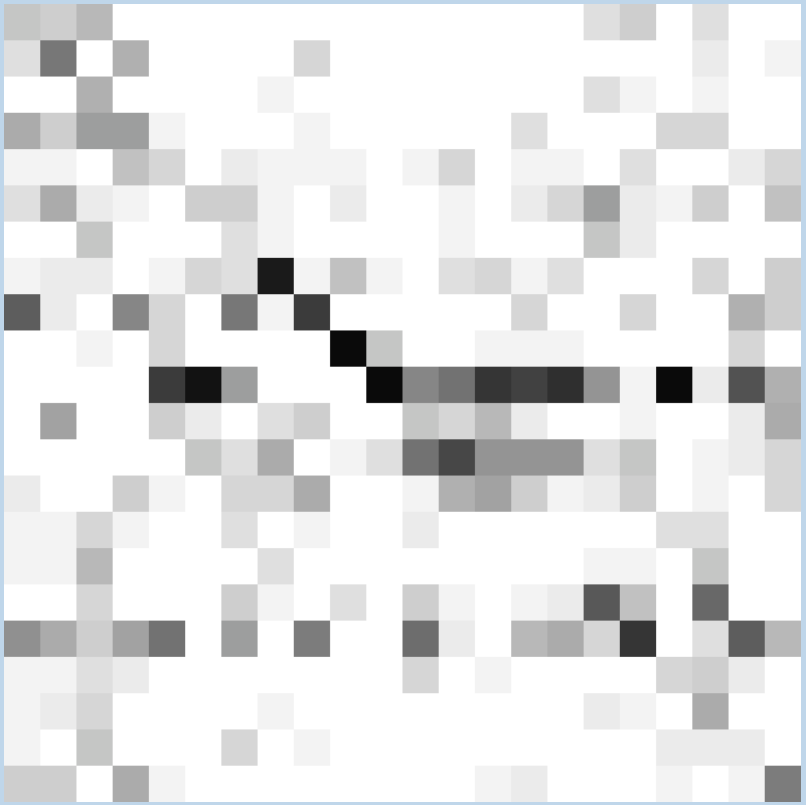
- Graph Domain



- Embedding Space

Confusion Matrix on GREC Data

- Graph Domain



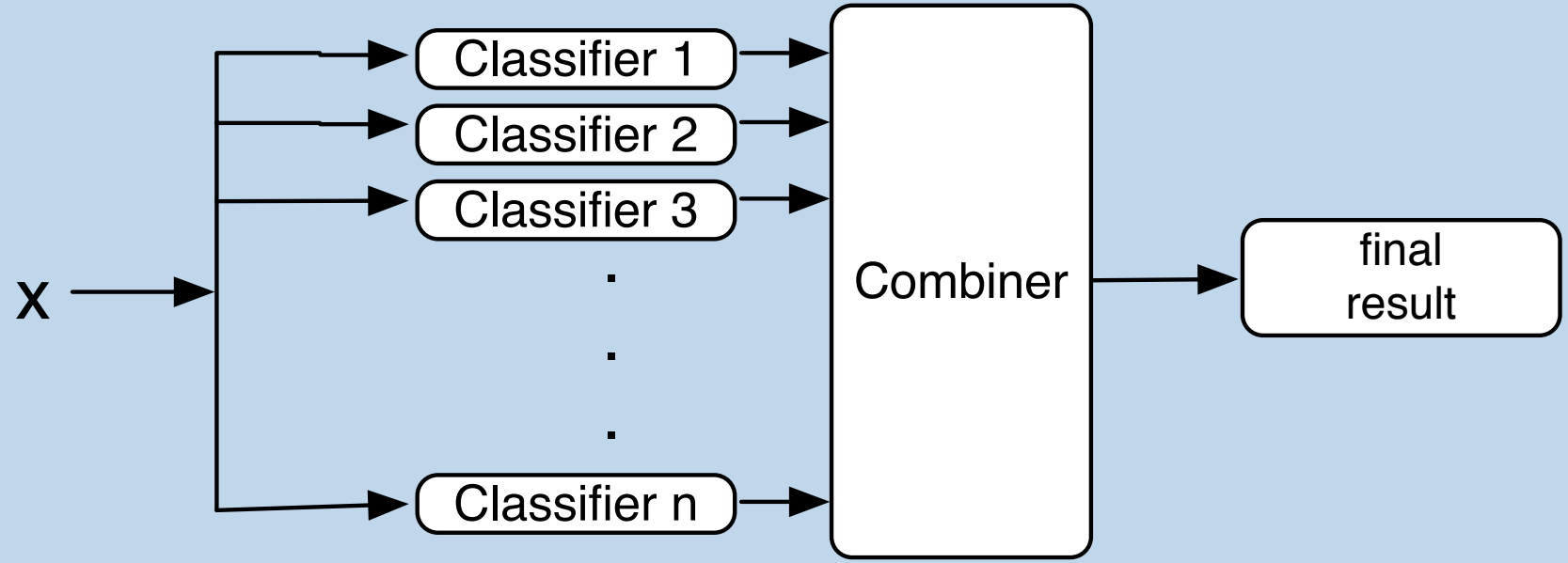
- Embedding Space

Contents

Multiple Classifier Systems

- Multiple Classifier and Multiple Clustering Systems (MCS), also called Ensemble Methods, have become a focus of intensive research
- They are based on the idea that the errors committed by one system can be corrected by the other ensemble members
- Many methods for building MCS have been proposed recently
- However, the majority of work is based on feature vector representations; there is almost no work on MCS for graph based representations

Multiple Classifier Systems



- Multiple Clustering Systems follow the same architecture

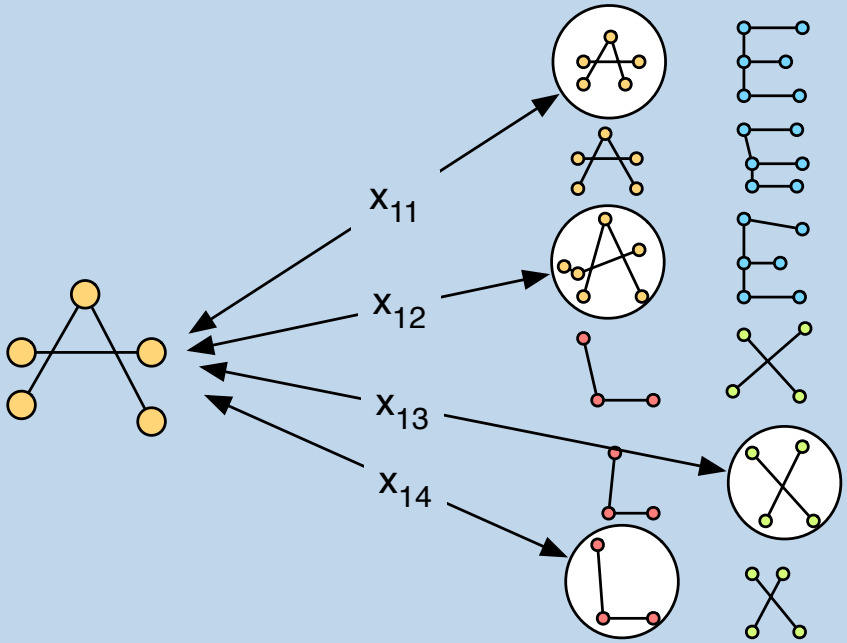
Standard Methods for MCS Creation

- Bagging (change the training set)
- Boosting (change the training set)
- Random feature subspace (change the features)
- Architecture variation (change the classifier)

Ensemble Generation for Graphs

- Use random prototype selection
- Train classifier using the whole training set
- Repeat random prototype selection until desired number of classifiers is obtained
- Apply ‘overproduce and select’ strategy, using a validation set

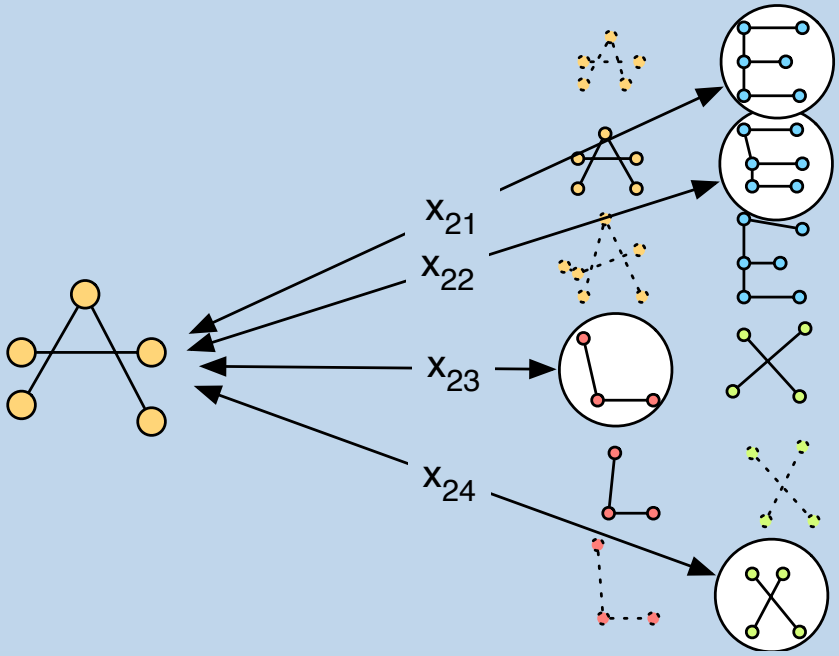
Randomized Graph Embedding



$v_1 = (x_{11}, x_{12}, x_{13}, x_{14})$

$$\varphi_n^{\mathcal{P}}(g) \mapsto (d(g, p_1), \ldots, d(g, p_n))$$

Randomized Graph Embedding

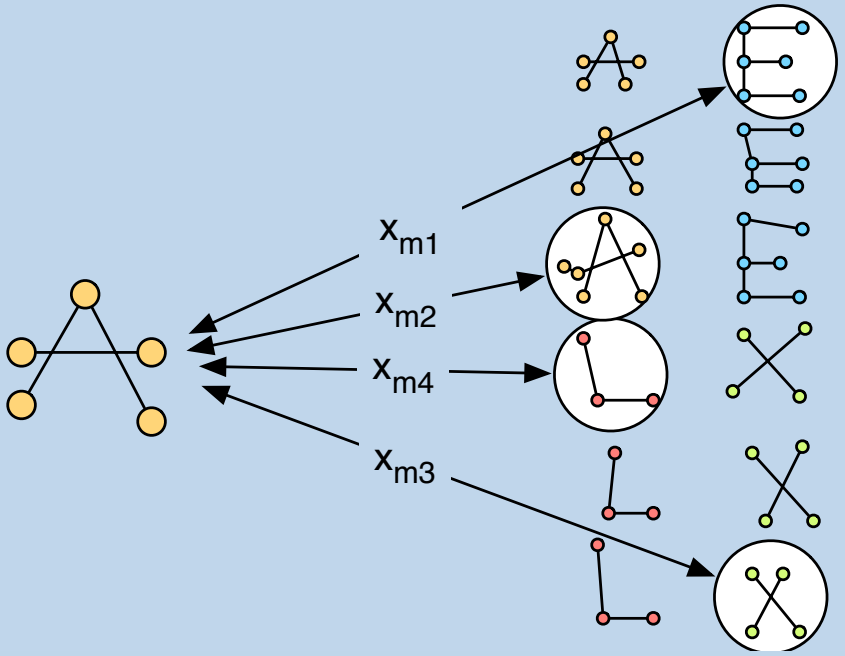


$v_1 = (x_{11}, x_{12}, x_{13}, x_{14})$

$v_2 = (x_{21}, x_{22}, x_{23}, x_{24})$

The selected graphs become temporarily unavailable for further selection

Randomized Graph Embedding



$v_1 = (x_{11}, x_{12}, x_{13}, x_{14})$

$v_2 = (x_{21}, x_{22}, x_{23}, x_{24})$

.
.
.

$v_m = (x_{m1}, x_{m2}, x_{m3}, x_{m4})$

Applying randomized graph embedding m times leads to m different vectorial descriptions of the same underlying graph

Experimental Evaluation: Classification

- Aim: compare classifiers in the embedding space with classifiers in the graph domain; compare an individual SVM with the ensemble
- Combination rules: Voting and Borda count
- Reference classifiers: same as in basic experiments

Ensemble Results

UNIVERSITÄT BERN

	ref. systems		ensemble methods	
Data Set	k-NN	sim	voting	Borda count
Letter low	99.3	99.6	99.2	99.1
Letter medium	94.4	94.9	95.2	95.2
Letter high	89.1	92.9	92.5 ①	92.8 ①
Digit	97.4	98.1	98.5 ①	98.4 ①
Fingerprint	79.1	82.0	83.0 ①	83.1 ①
GREC	82.2	71.6	92.3 ①	92.1 ①
AIDS	94.9	97.0	97.9 ①	97.8 ①
Mutagenicity	66.9	68.6	72.4 ①	71.1 ①
Protein	68.5	68.5	71.5	68.5
Webpage	80.6	82.9	83.7 ①	82.6 ①

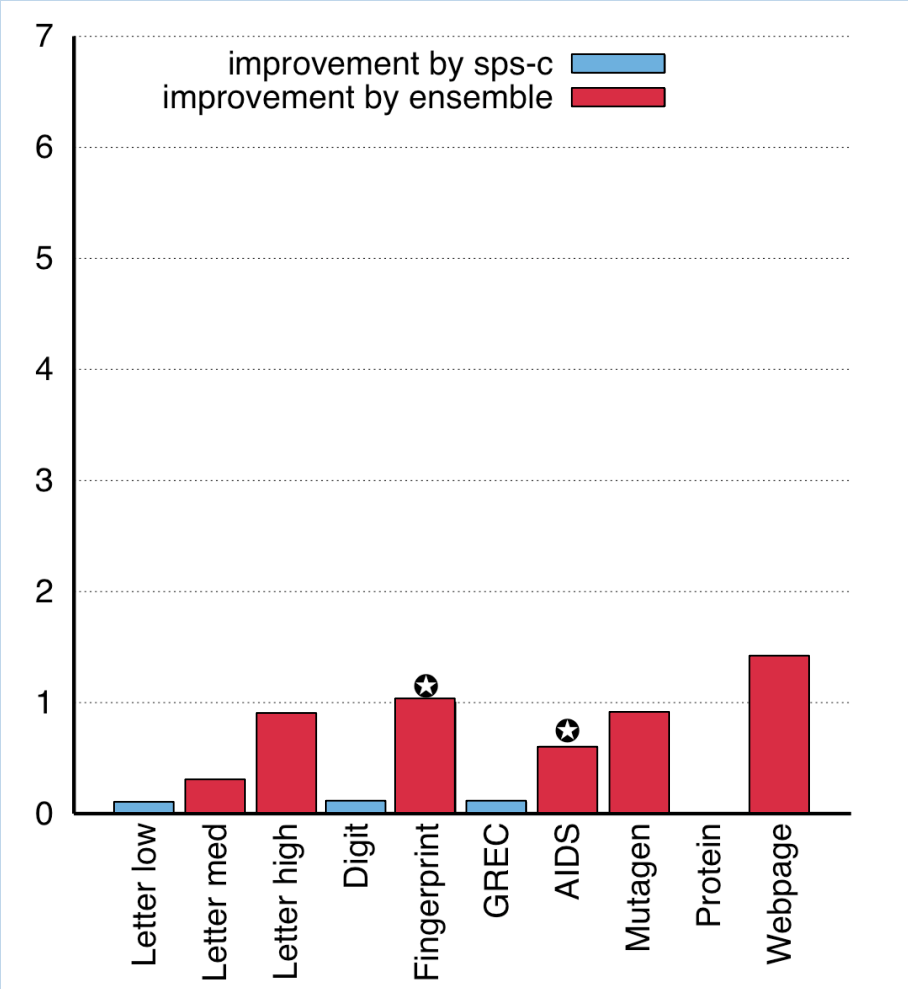
- Compared to k-NN: 9/8 improvements (7/7×①), 1/1 deterioration (0×❶).

Ensemble Results

	ref. systems		ensemble methods	
Data Set	k-NN	sim	voting	Borda count
Letter low	99.3	99.6	99.2	99.1
Letter medium	94.4	94.9	95.2	95.2
Letter high	89.1	92.9	92.5	92.8
Digit	97.4	98.1	98.5	98.4
Fingerprint	79.1	82.0	83.0 ②	83.1 ②
GREC	82.2	71.6	92.3 ②	92.1 ②
AIDS	94.9	97.0	97.9 ②	97.8 ②
Mutagenicity	66.9	68.6	72.4 ②	71.1
Protein	68.5	68.5	71.5	68.5
Webpage	80.6	82.9	83.7	82.6

- Compared to sim: 8/6 improvements (4/3×①), 2/3 deterioration (0×❶).

Single Classifier vs. Classifier Ensemble



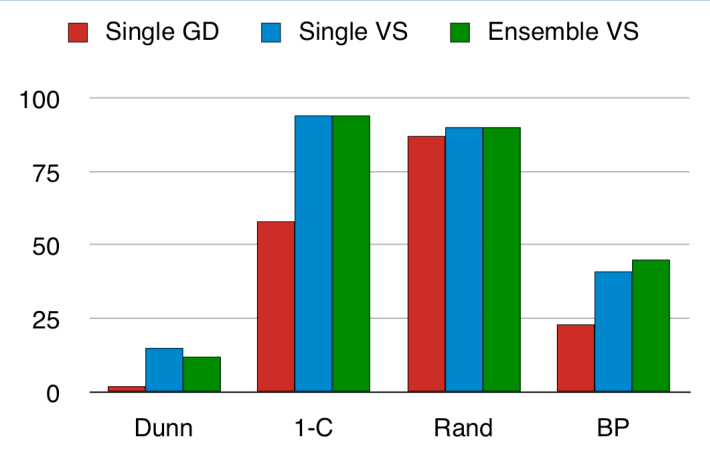
Experimental Evaluation: Clustering

- Aim: compare an ensemble of k-means clustering algorithms in the embedding space with clustering in the graph domain;
  compare the ensemble in the embedding space with an individual clustering algorithm in the embedding space

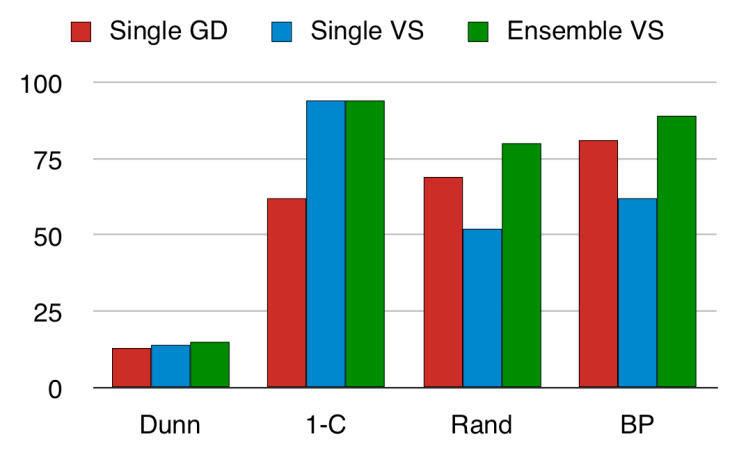
Combination of Clusterings

- Based on a co-association matrix $\mathbf{C} = (c_{ij})$
- Element c_{ij} indicates how often graph g_i and graph g_j appear in the same cluster
- The co-association matrix can be used as a similarity matrix
- In particular, the elements c_{ij} are used as kernel values $\kappa(g_i, g_j)$ in the kernelized k-means algorithm to produce the final clustering

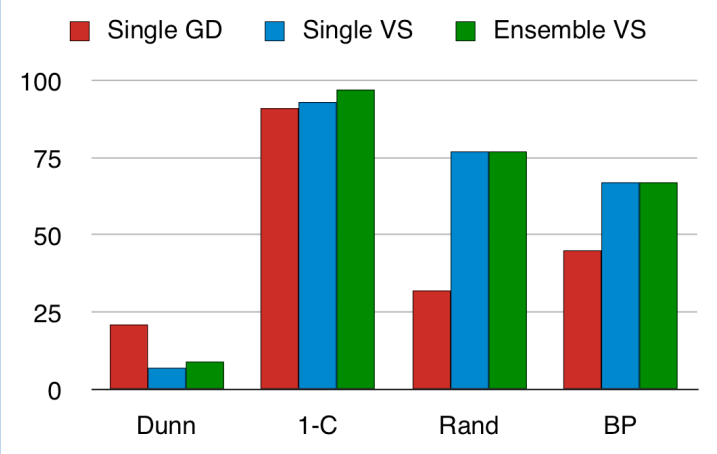
Results



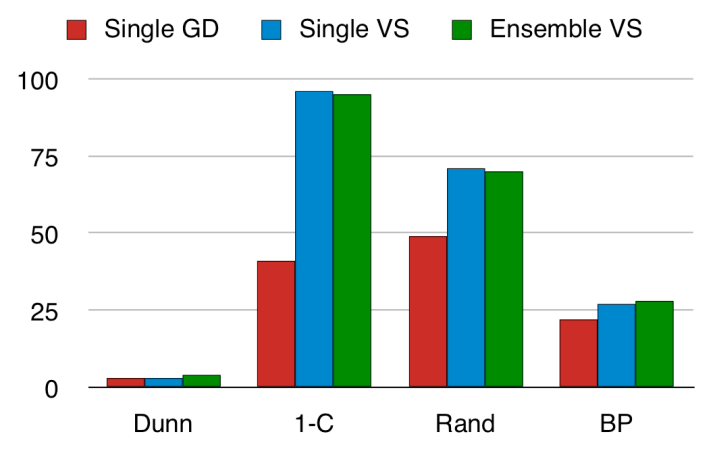
Letter



COIL

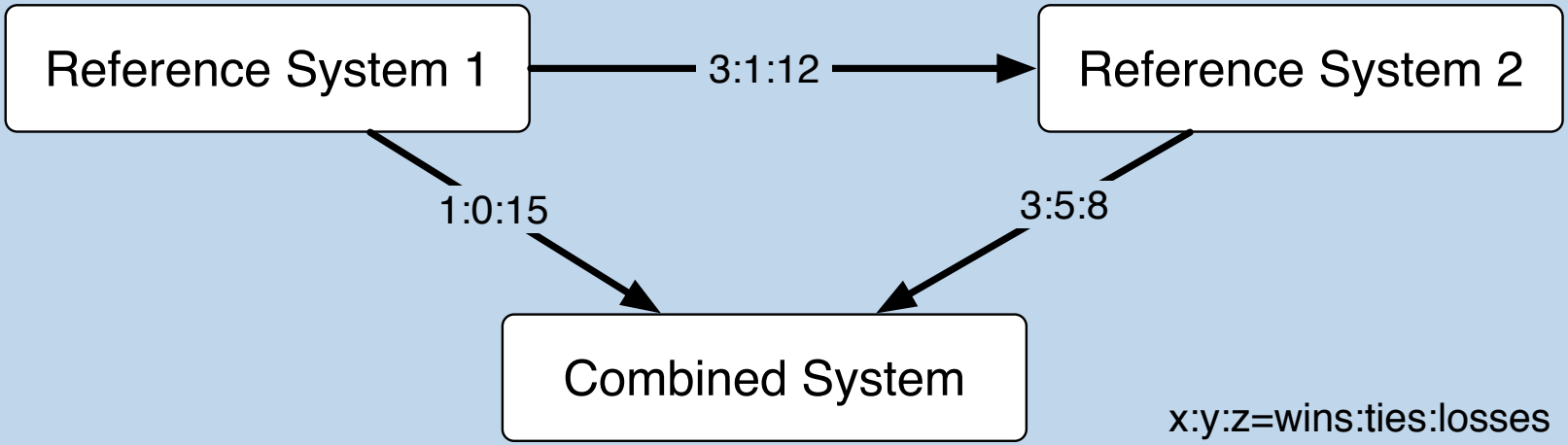


Fingerprint



Protein

Summary of Results



Contents

Further Refinements

- Lipschitz embedding:
  Instead of individual prototypes $p_1, \ldots, p_n$, use sets $P_1, \ldots, P_n$ of prototypes
- PCA and kernel PCA:
  Use all available graphs for the embedding and apply some dimensionality reduction method; alternatively, the dimensionality reduction problem can be viewed as a feature selection problem
- Prototype reduction methods for nearest neighbor classification:
  Use editing, condensing, and similar procedures
- These methods have the potential of further improvements over the basic methods discussed so far

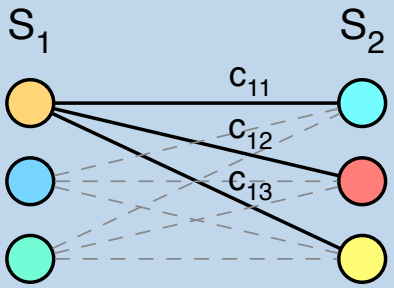
Contents

Complexity of GED

- The novel graph embedding methods depend on the graph edit distance
- However, the complexity of graph edit distance computation is exponential
- Possible solutions:
  - Restriction to special classes of graphs, e.g., graphs with unique node labels; complexity becomes linear; applications in web text mining and computer network analysis
  - **Use of approximate (suboptimal) methods**
- In our case:
  - Suboptimal versions of the A* algorithm
  - **Bipartite graph matching**

Bipartite Graph Matching

- Bipartite graph matching, also known as the Hungarian method or Munkres' algorithm, provides us with a method to solve the assignment problem



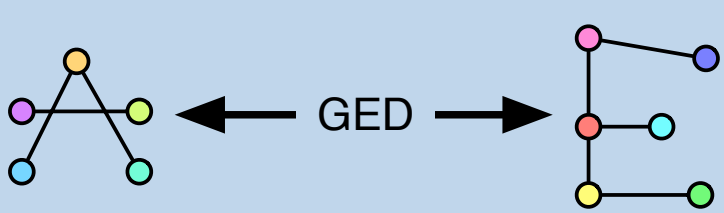
optimization problem:

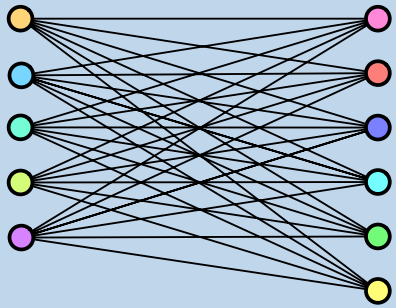
$$\sum_{i=1}^{n} c_{i\pi_i} \stackrel{!}{=} \min_{\pi}$$

- Typical applications in operations research (assignment of tasks to machines)
- The method is optimal for the assignment problem
- It has a time complexity of $O(n^3)$

Bipartite Graph Matching for GED

- The assignment problem has nothing to do with GED computation
- However GED can be reformulated (simplified) such that Munkres' algorithm becomes applicable



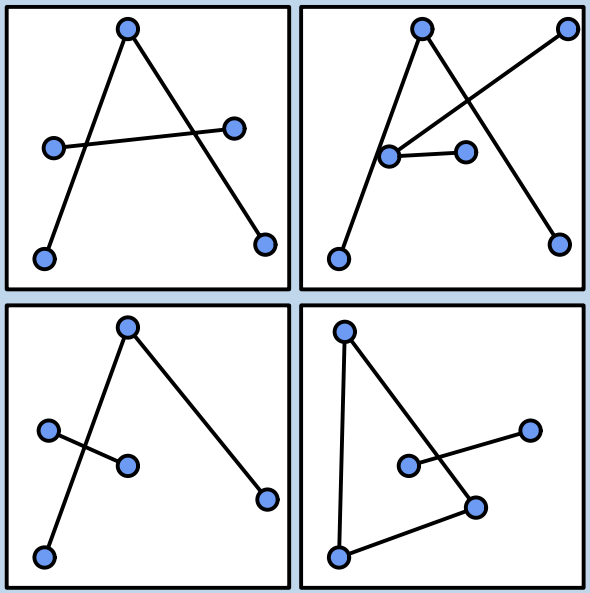


- Different reformulations are possible (considering only nodes, or nodes plus their local edge neighborhood)
- Munkres' algorithm returns an optimal solution for the assignment problem
- However, we get only a suboptimal solution for GED

Experimental Evaluation

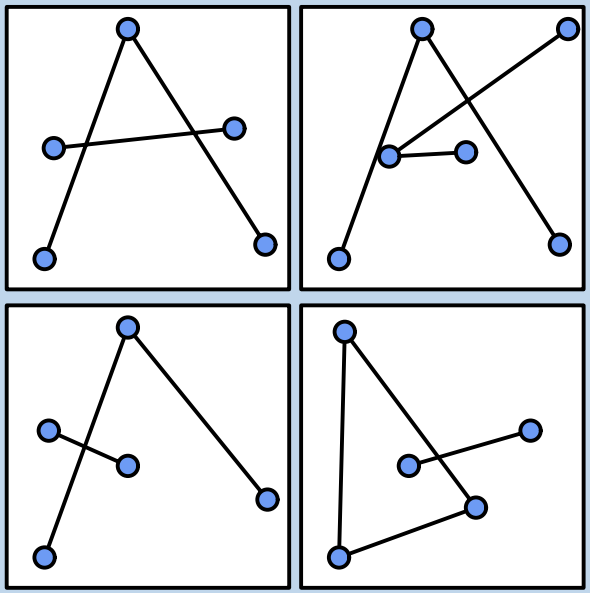
- There are two questions to be answered:
  - How much computation time can be saved by the suboptimal algorithms?
  - How much classification accuracy do we loose by applying suboptimal rather than optimal algorithms?

Letter Data Set



Method	Time [ms]
Tree Search	468.0
Plain	0.2
Adjacency	2.8

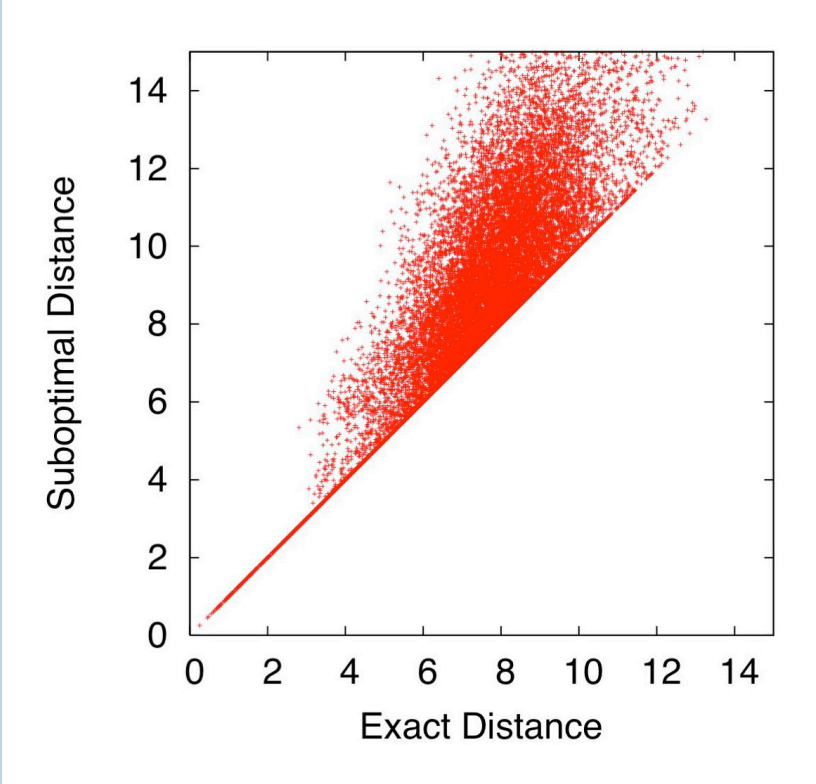
Letter Data Set



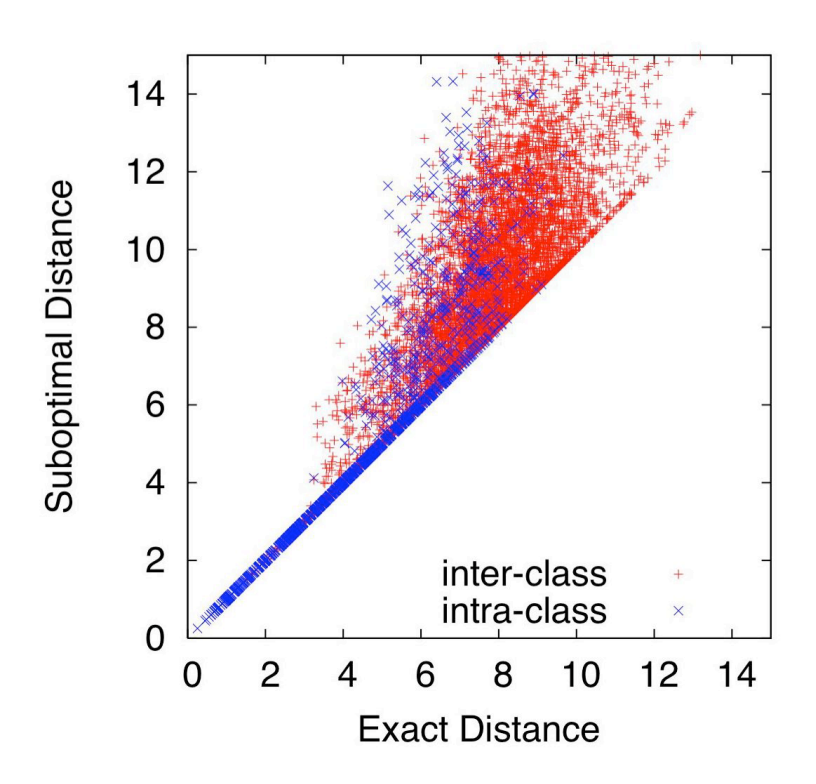
Method	Time [ms]	Accuracy
Tree Search	468.0	80.7
Plain	0.2	84.0 ○
Adjacency	2.8	84.0 ○

○ Statistically significantly better than the reference system

Suboptimality



Suboptimality



- Suboptimality mainly leads to an increase of inter-class distances, while most of the intra-class distances are not strongly affected

Contents

Summary and Conclusions

- Graphs are embedded in vector spaces by means of prototype selection and graph edit distance
- In this way we can utilize the high representational power of graphs and make available all pattern recognition methods that have been developed for feature based object representations
- The proposed method can be applied to classification and clustering tasks
- It is applicable to any type of graph
- Furthermore, it is applicable not only in conjunction with kernel methods, but to any other type of algorithm that operates in a feature space

Summary and Conclusions

- The proposed approach to embedding allows one to build multiple classifier and clustering systems in a straightforward fashion
- To avoid computational complexity problems, approximate methods for edit distance computation can be used; they lead to a substantial gain in computational speed, but don't compromise classification performance
- The experimental results show superior performance over reference systems over a spectrum of different applications and graphs with different characteristics

Related Publications 1/7

- Efficient Graph Edit Distance
  - Riesen, K., Jiang, X. and Bunke, H.: Exact and Inexact Graph Gatching – Methodology and Applications. To appear in book *Managing and Mining Graph Data*
  - Bunke, H. and Riesen, K.: Graph Edit Distance – Optimal and Suboptimal Algorithms with Applications. In book *Analysis of Complex Networks*, 2009.
  - Riesen, K., Fankhauser, S. and Bunke, H.: Efficient Suboptimal Graph Isomorphism. In Proc. 7th. Int. Workshop on GbR, 2009
  - Riesen, K. and Bunke, H.: Approximate Graph Edit Distance Computation by means of Bipartite Graph Matching. In IMAVIS, 2008.
  - Riesen, K., Neuhaus, M. and Bunke, H.: Bipartite Graph Matching for Computing the Edit Distance of Graphs. In Proc. 6th Int. Workshop on GbR, 2007.

Related Publications 2/7

- Efficient Graph Edit Distance (cont'd)
  - Riesen, K., Fankhauser, S. and Bunke, H.: Speeding up Graph Edit Distance Computation with a Bipartite Heuristic. In Proc. 5th. Int. Workshop on MLG, 2007
  - Neuhaus, M., Riesen, K. and Bunke, H.: Fast Suboptimal Algorithms for the Computation of Graph Edit Distance. In Proc. 11th Int. Workshop on SSPR, 2006.
- Dissimilarity Space Embedding
  - Bunke, H. and Riesen, K.: On Dissimilarity Embedding of Graphs in Vector Spaces. To appear in book *Handbook of Pattern Recognition and Computer Vision*.
  - Riesen, K. and Bunke, H.: Graph Classification by means of Lipschitz embedding. In SMC, 2009.

Related Publications 3/7

- Dissimilarity Space Embedding (cont'd)
  - Riesen, K. and Bunke, H.: Graph Classification Based on Vector Space Embedding. In IJPRAI, 2009.
  - Riesen, K. and Bunke, H.: Reducing the Dimensionality of Dissimilarity Space Embedding Graph Kernels. In AAI, 2009.
  - Riesen, K. and Bunke, H.: Feature Ranking Algorithms for Improving Classification of Vector Space Embedded Graphs. In Proc. 13th Int. Conf. on CAIP, 2009.
  - Riesen, K., Frinken, V. and Bunke, H.: Improving Graph Classification by Isomap. In Proc. 7th. Int. Workshop on GbR, 2009
  - Riesen, K. and Bunke, H.: Dissimilarity Based Vector Space Embedding of Graphs using Prototype Reduction Schemes. In Proc. 6th Int. Conf. on MLDM, 2009.

Related Publications 4/7

UNIVERSITÄT BERN

- Dissimilarity Space Embedding (cont'd)
  - Riesen, K. and Bunke, H.: Cluster Ensembles based on Vector Space Embeddings of Graphs. In Proc. 8th Int. Workshop on MCS, 2009.
  - Riesen, K. and Bunke, H.: Non-linear Transformations of Vector Space Embedded Graphs. In Proc. 8th Int. Workshop on PRIS, 2008.
  - Bunke, H. and Riesen, K.: Recent Developments in Graph Classification and Clustering using Graph Embedding Kernels. In Proc. 8th Int. Workshop on PRIS, 2008.
  - Bunke, H. and Riesen, K.: Graph Classification based on Dissimilarity Space Embedding. In Proc. 12th int. Workshop on SSPR, 2008.

Related Publications 5/7

- Dissimilarity Space Embedding (cont'd)
  - Riesen, K. and Bunke, H.: On Lipschitz Embeddings of Graphs. In Proc. 12th Int. Conf. on KES, 2008.
  - Riesen, K. and Bunke, H.: Kernel k-Means Clustering Applied to Vector Space Embeddings of Graphs. In Proc. 3rd Workshop on ANNPR, 2008.
  - Fischer, A., Riesen, K. and Bunke, H.: An Experimental Study of Graph Classification using Prototype Selection. In Proc. 19th Int. Conf. on PR, 2008.
  - Ferrer, M. et al.: An Approximate Algorithm for Median Graph Computation using Graph Embedding. In Proc. 19th Int. Conf. on PR, 2008.
  - Bunke, H. and Riesen, K.: A Familiy of Novel Graph Kernels for Structural Pattern Recognition. In Proc. 12th CIARP, 2007.

Related Publications 6/7

UNIVERSITÄT BERN

- Dissimilarity Space Embedding (cont'd)
  - Riesen, K., Neuhaus, M. and Bunke, H.: Graph Embedding in Vector Spaces by Means of Prototype Selection. In Proc. 6th Int. Workshop on GbR, 2007.
  - Riesen, K., Kilcherr, V. and Bunke, H.: Reducing the Dimensionality of Vector Space Embeddings of Graphs. In Proc. 5th Int. Conf. on MLDM, 2007.
  - Riesen, K. and Bunke, H.: Classifier Ensembles for Vector Space Embedding of Graphs. In Proc. 7th Int. Workshop on MCS, 2007.
  - Riesen, K. and Bunke, H.: Structural Classifier Ensembles for Vector Space Embedded Graphs. In Proc. 20th Int. JCNN, 2007

Related Publications 7/7

- Databases
  - Riesen, K. and Bunke, H.: IAM Graph Database Repository for Graph Based Pattern Recognition and Machine Learning. In Proc. 12th Int. Workshop on SSPR, 2008.
  - `www.iam.unibe.ch/fki/databases/iam-graph-database`
- Miscellaneous
  - Neuhaus, M., Riesen, K. and Bunke, H.: Novel Kernels for Error-tolerant Graph Classification. In SV, 2009.
  - Brügger, A. et al.: Generalized Graph Matching for Data Mining and Information Retrieval. In Proc. ICDM, 2008.