QSAR and Virtual Screening with Support Vector Machines

Jean-Philippe Vert
Jean-Philippe.Vert@ensmp.fr

Center for Computational Biology
Ecole des Mines de Paris

Chimiométrie 2006, Paris, France, December 1st, 2006

MINES PARIS ARMINES

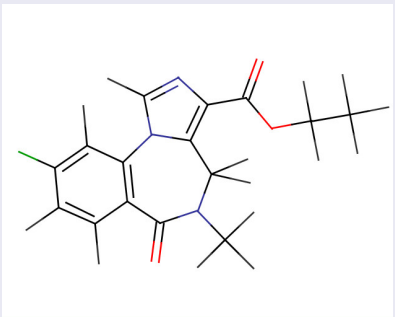
Ligand-Based Virtual Screening

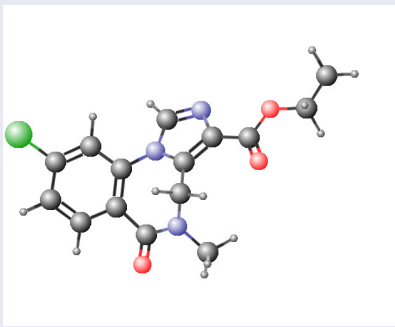
Objective

Build models to predict biochemical properties of small molecules from their structures.

Structures

$C_{15}H_{14}ClN_3O_3$





Properties

- binding to a therapeutic target,
- pharmacokinetics (ADME),
- toxicity...

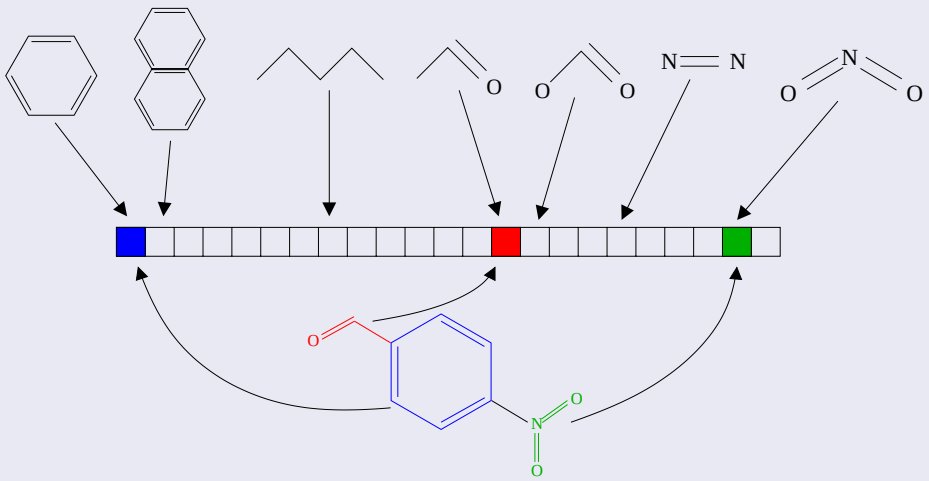
Issues and solution

Two important steps

1. Map each molecule to a vector of fixed dimension.
2. Apply an algorithm for regression or pattern recognition.

Example: 2D structural keys

A vector indexed by a limited set of informative stuctures



+ NN, PLS, decision tree, ...

Classical approaches

Difficulties

- Expressivity of the features (which features are relevant?)
- Dimension of the vector (memory storage, speed, statistical issues)

Our approach

Work implicitly in large (potentially infinite!) dimensions:

- Allows to consider a large number of potentially important features.
- No need to store explicitly the vectors (no problem of memory storage or hash clashes)
- Use of regularized statistical algorithm to handle the problem of large dimension

Classical approaches

Difficulties

- Expressivity of the features (which features are relevant?)
- Dimension of the vector (memory storage, speed, statistical issues)

Our approach

Work implicitly in large (potentially infinite!) dimensions:

- Allows to consider a large number of potentially important features.
- No need to store explicitly the vectors (no problem of memory storage or hash clashes)
- Use of regularized statistical algorithm to handle the problem of large dimension

Outline

Outline

The Machine Learning Paradigm

Objective

Predict a property y for objects x

- x = molecule, gene sequence, picture, ...
- y is continuous (regression) or discrete (pattern recognition)

A two-step approach

1. Training: observe a set

$$\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$

of labeled objects, and learn a function $f : \mathcal{X} \to \mathcal{Y}$

2. Test: Given a new object x, predict its label by $f(x)$.

MINES PARIS ARMINES

The Machine Learning Paradigm

Objective

Predict a property y for objects x

- x = molecule, gene sequence, picture, ...
- y is continuous (regression) or discrete (pattern recognition)

A two-step approach

1. Training: observe a set

$$\mathcal{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$

   of labeled objects, and learn a function $f : \mathcal{X} \to \mathcal{Y}$
2. Test: Given a new object x, predict its label by $f(x)$.

MINES PARIS ARMINES

Examples

In biomedical research..

- Virtual screening : x is the description of a molecule, y is the activity / toxicity / drugability ...
- Medical diagnosis and prognosis: x is a set of features (age, weight, transcriptome...), y is the risk / type of tumor / expected evolution of disease.
- Functional genomics : x is a set of gene features (sequence, expresssion...), y is the function of the gene
- ...

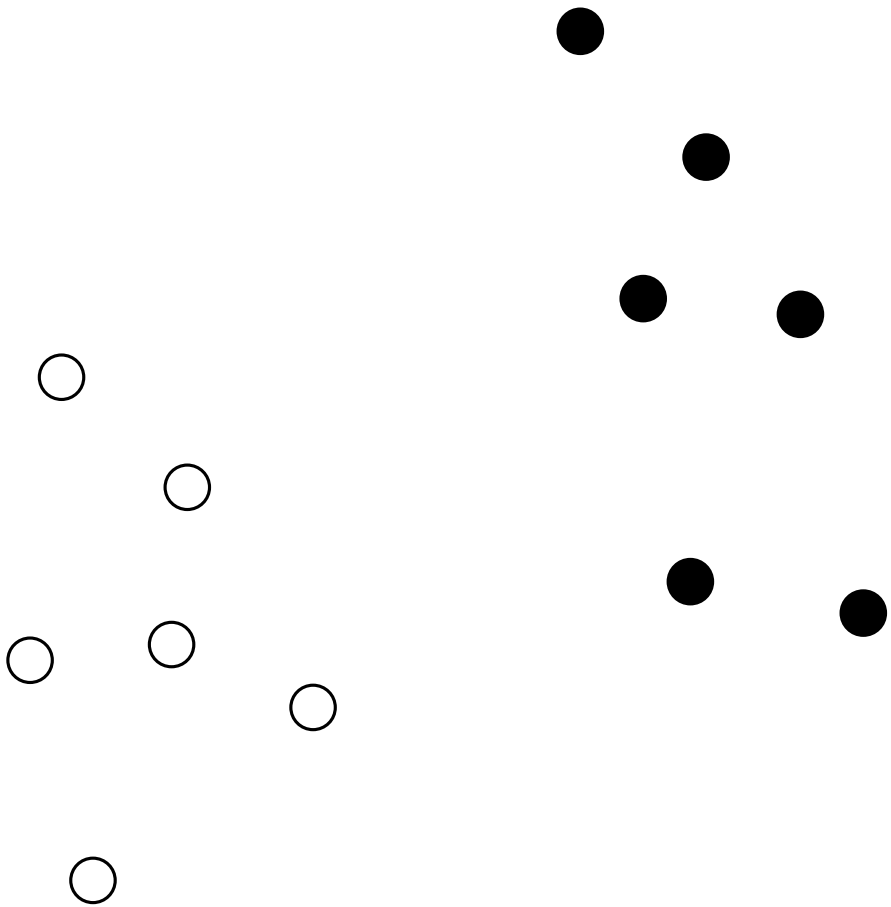
What is a SVM?

Main features

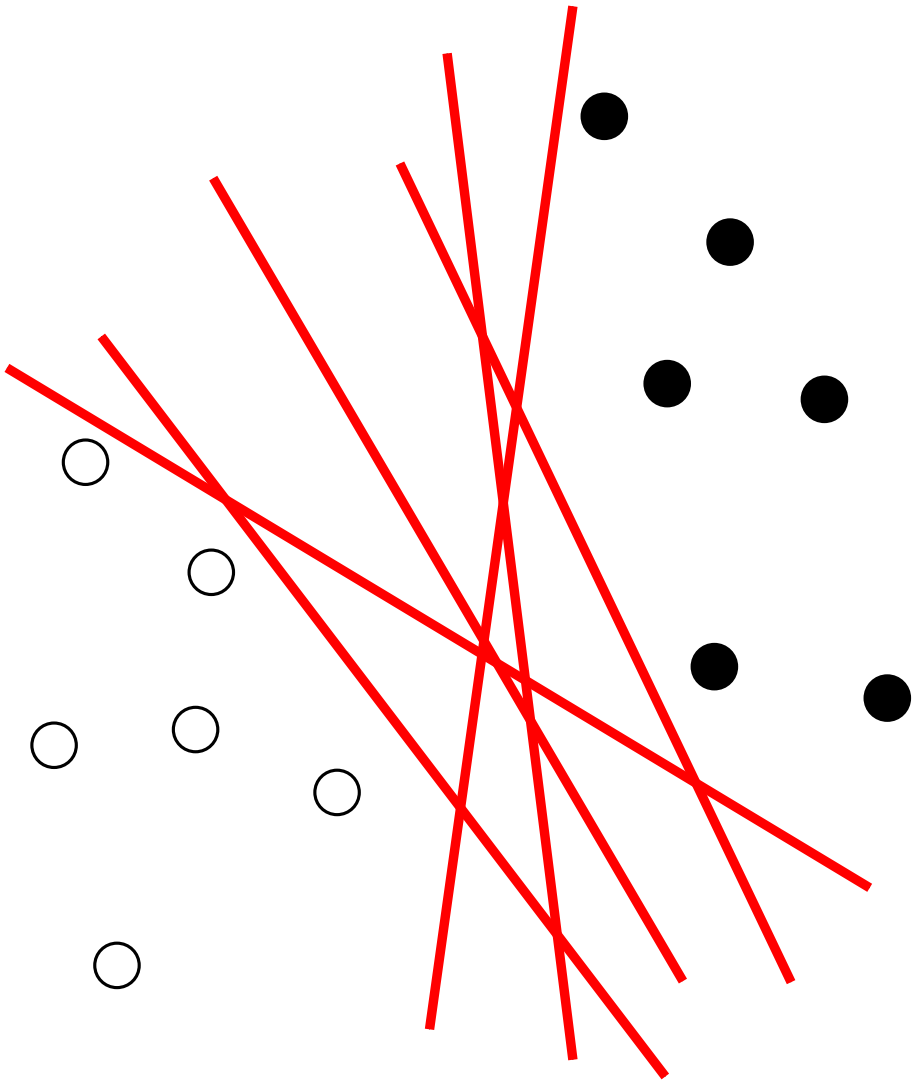
- an algorithm for pattern recognition and regression
- robust in high dimension (e.g., images, texts, microarrays, fingerprints)
- handles vectorial or structured data (e.g., sequences, graphs)
- allows easy integration of heterogeneous data (e.g., gene sequence and expression, docking score and molecule structure...)
- state-of-the-art performance on many real-world applications.

MINES PARIS ARMINES

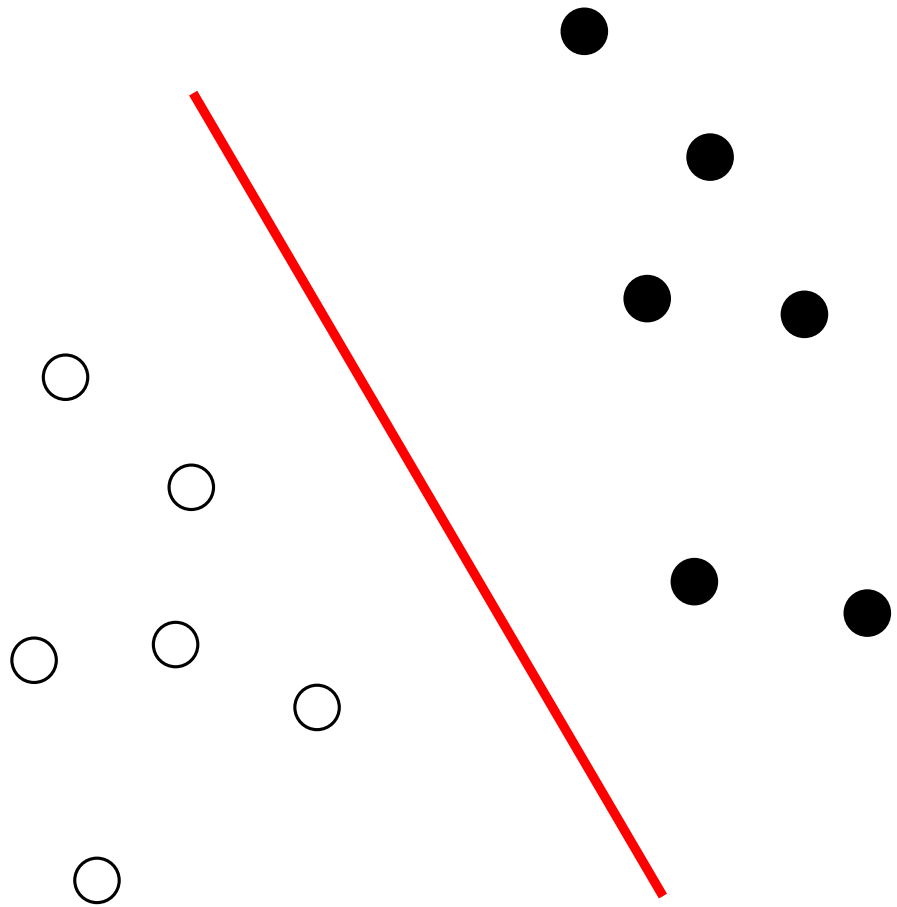
Simplest SVM



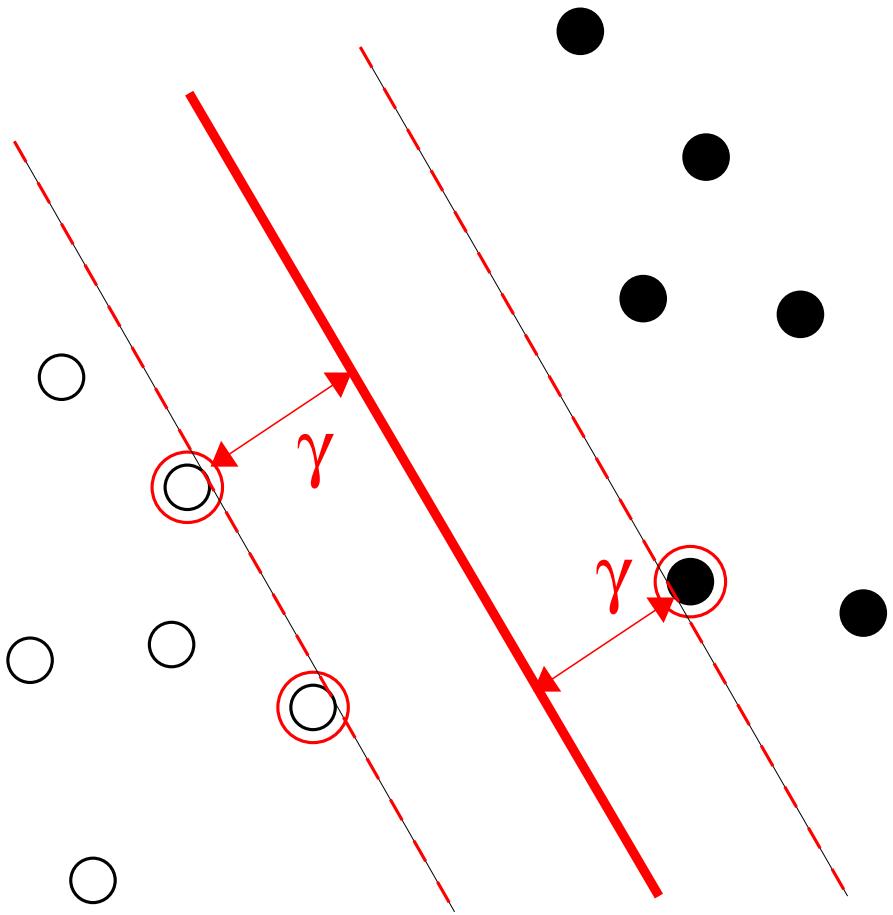
Simplest SVM



Simplest SVM



Simplest SVM



Linear SVM: implementation

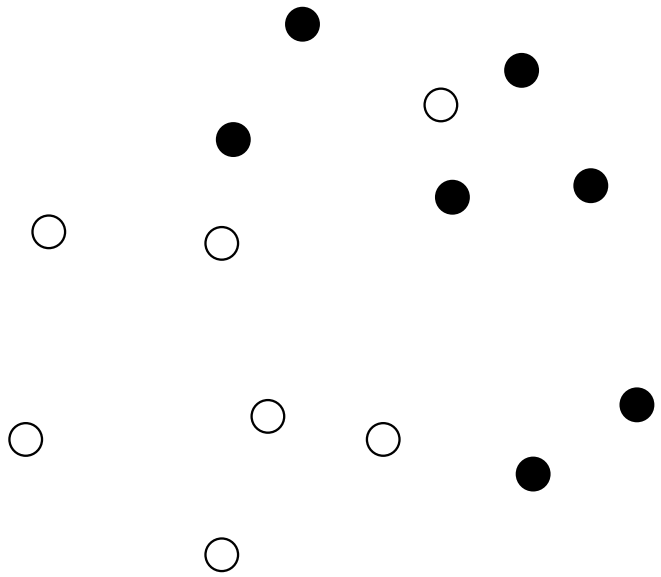
- After some algebra it is obtained by solving in $\alpha \in \mathbb{R}^n$ the following quadratic program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j x_i^\top x_j - \sum_{i=1}^{n} \alpha_i \\ \text{subject to} \quad & \alpha_i \geq 0, \quad i = 1, \ldots, n \,, \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \,. \end{aligned}$$

- Once α is found, the classification function is the sign of :

$$f(x) = \sum_{i=1}^{n} \alpha_i x_i^\top x + b \,.$$

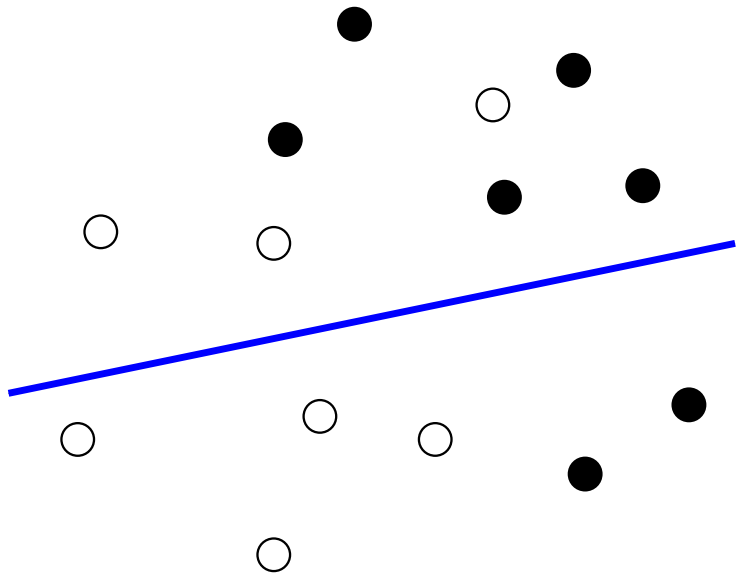
Linear SVM: non-separable case



Implementation

- Solution: find a trade-off between large margin and few misclassification
- Simple and elegant mathematical translation: replace $0 \leq \alpha_i$ by $0 \leq \alpha_i \leq C$, for some constant $C > 0$, in the optimization problem.

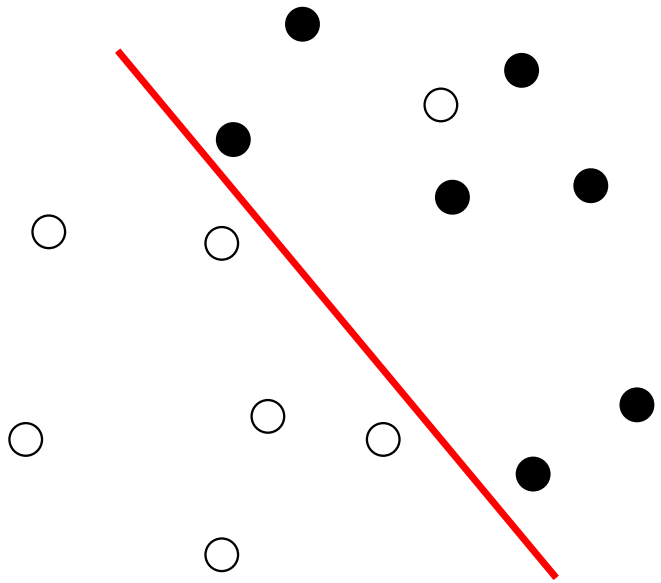
Linear SVM: non-separable case



Implementation

- Solution: find a trade-off between large margin and few misclassification
- Simple and elegant mathematical translation: replace $0 \leq \alpha_i$ by $0 \leq \alpha_i \leq C$, for some constant $C > 0$, in the optimization problem.

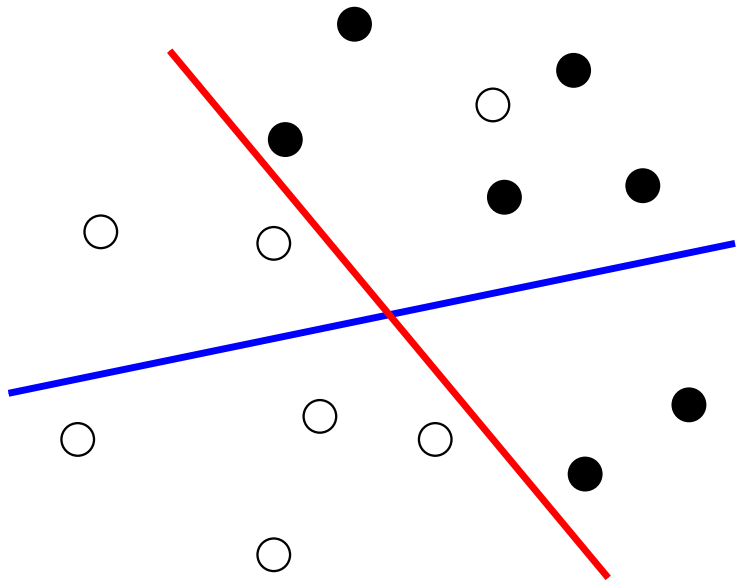
Linear SVM: non-separable case



Implementation

- Solution: find a trade-off between large margin and few misclassification
- Simple and elegant mathematical translation: replace $0 \leq \alpha_i$ by $0 \leq \alpha_i \leq C$, for some constant $C > 0$, in the optimization problem.

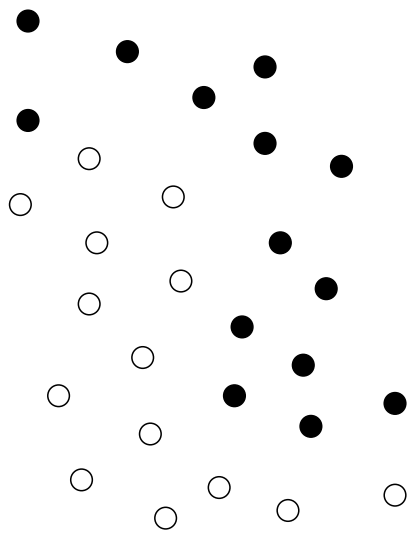
Linear SVM: non-separable case



Implementation

- Solution: find a trade-off between large margin and few misclassification
- Simple and elegant mathematical translation: replace $0 \leq \alpha_i$ by $0 \leq \alpha_i \leq C$, for some constant $C > 0$, in the optimization problem.

Nonlinear SVM



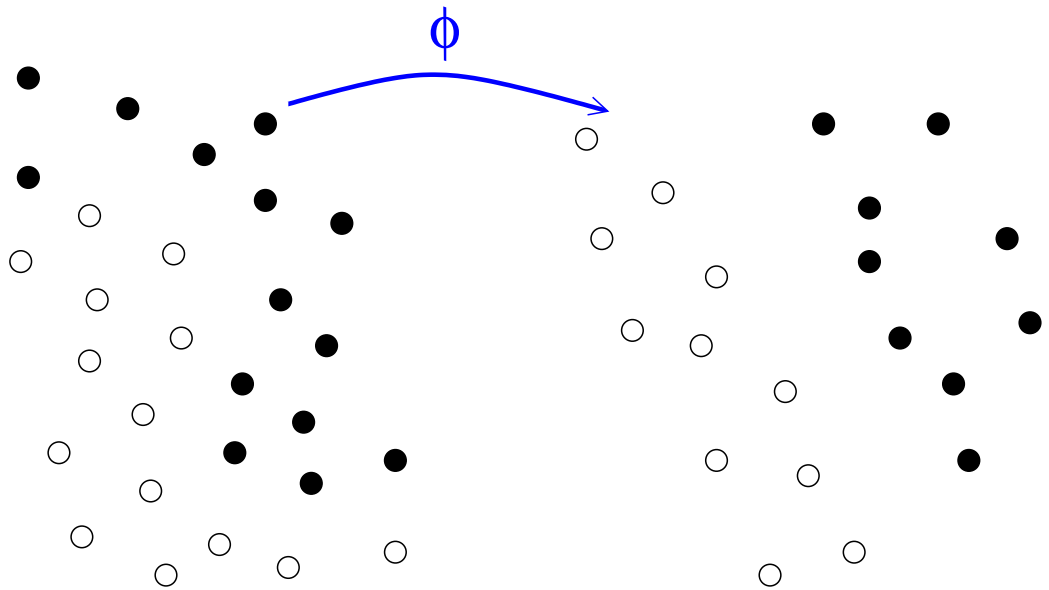
The idea

- Define a (nonlinear) mapping

$$\phi : \mathcal{X} \to \mathcal{F} \subset \mathbb{R}^p \ .$$

- Run a linear SVM in the feature space.

Nonlinear SVM



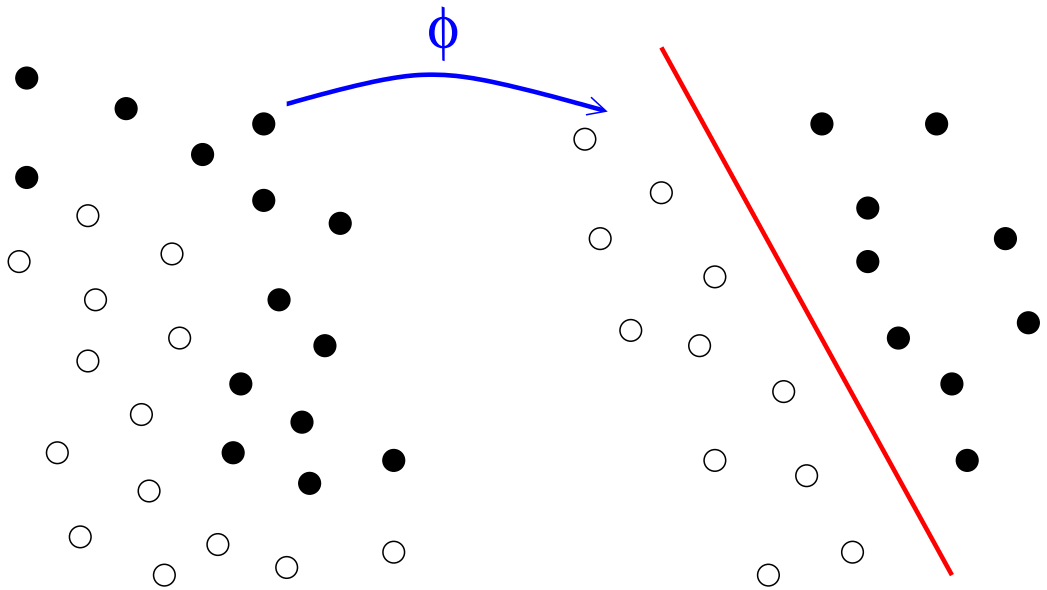
The idea

- Define a (nonlinear) mapping

$$\phi : \mathcal{X} \to \mathcal{F} \subset \mathbb{R}^p \ .$$

- Run a linear SVM in the feature space.

Nonlinear SVM



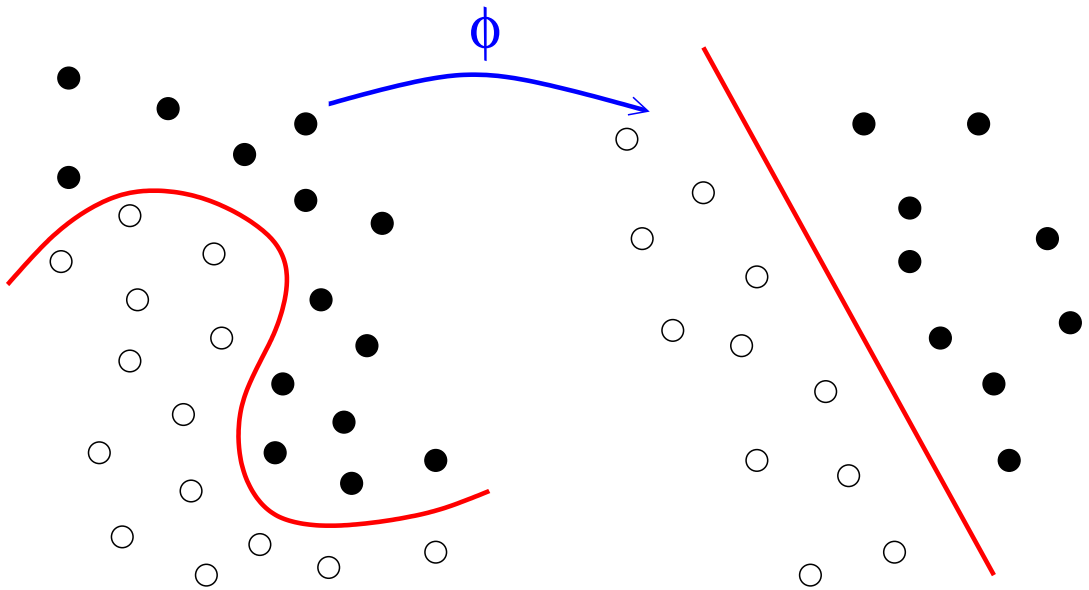
The idea

- Define a (nonlinear) mapping

$$\phi : \mathcal{X} \to \mathcal{F} \subset \mathbb{R}^p \ .$$

- Run a linear SVM in the feature space.

Nonlinear SVM



The idea

- Define a (nonlinear) mapping

$$\phi : \mathcal{X} \to \mathcal{F} \subset \mathbb{R}^p \ .$$

- Run a linear SVM in the feature space.

Nonlinear SVM: implementation

- Solve in $\alpha \in \mathbb{R}^n$:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j \Phi(x_i)^\top \Phi(x_j) - \sum_{i=1}^{n} \alpha_i \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n\,, \\ & \sum_{i=1}^{n} \alpha_i y_i = 0\,. \end{aligned}$$

- Once α is found, the classification function is the sign of :

$$f(x) = \sum_{i=1}^{n} \alpha_i \Phi(x_i)^\top \Phi(x) + b$$

The kernel tricks

Important idea!

- To any mapping $\Phi : \mathcal{X} \to \mathcal{F}$ corresponds a kernel function K:

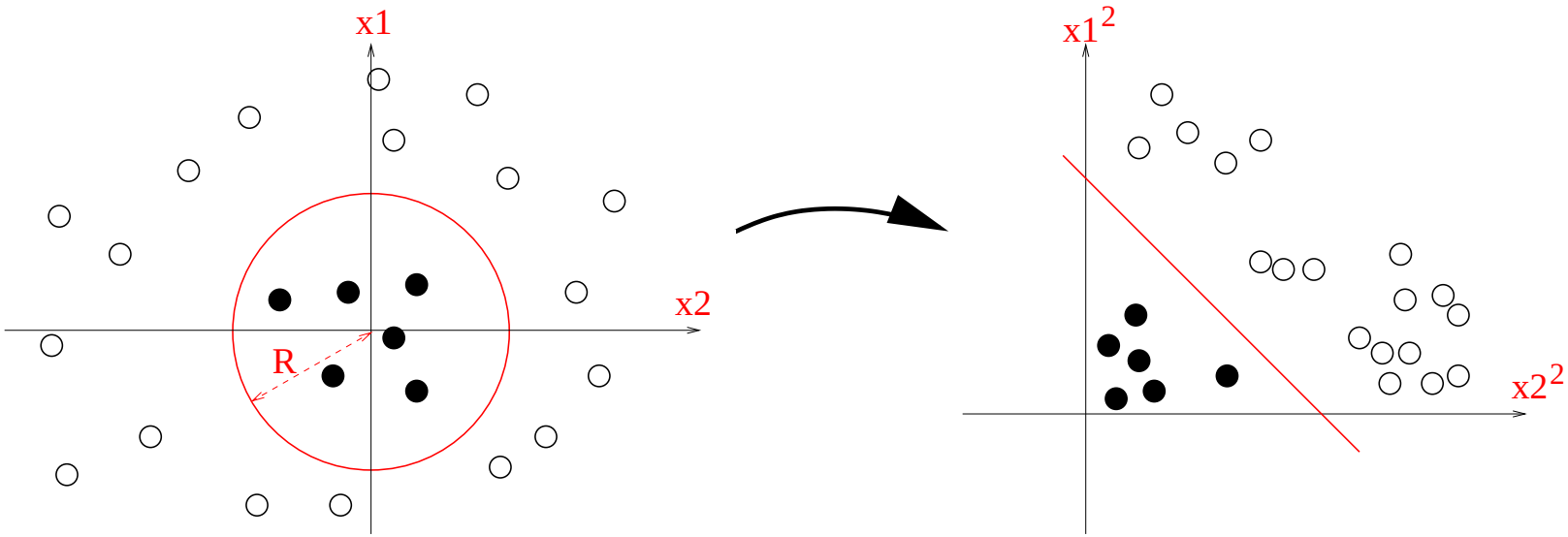
$$K(x, x') = \Phi(x)^\top \Phi(x') \, .$$

- SVM only need K, rather than Φ:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^n \alpha_i \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n \, , \\ & \sum_{i=1}^n \alpha_i y_i = 0 \, . \end{aligned}$$

MINES PARIS ARMINES

Example: polynomial kernel



For $x = (x_1, x_2)^\top \in \mathbb{R}^2$, let $\Phi(x) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \in \mathbb{R}^3$:

$$\begin{aligned} K(x, x') &= x_1^2 x_1'^2 + 2x_1x_2x_1'x_2' + x_2^2 x_2'^2 \\ &= \left(x_1x_1' + x_2x_2'\right)^2 \\ &= \left(x^\top x'\right)^2 . \end{aligned}$$

Kernel examples

For vectors

- The linear kernel

$$K_{lin}\left(\mathbf{x}, \mathbf{x}'\right) = \mathbf{x}^\top \mathbf{x}' \ .$$

- The polynomial kernel

$$K_{poly}\left(\mathbf{x}, \mathbf{x}'\right) = \left(\mathbf{x}^\top \mathbf{x}' + a\right)^d \ .$$

- The Gaussian RBF kernel:

$$K_{Gaussian}\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}' \|^2}{2\sigma^2}\right) \ .$$

MINES PARIS ARMINES

Working with kernels

Main features

1. There exist conditions to ensure that a function $K(x, x')$ is a valid kernel (symmetry, positive definiteness).
2. No need to compute the corresponding Φ.
3. A kernel K can be thought of as a measure of similarity (inner products) between the data points.
4. The kernel trick allows to work implicitly in a (possibly large-dimensional) feature space, in particular:
   - to obtain non-linear versions of linear methods (nonlinear kernels)
   - to extend these methods to non-vector data (kernels for general objects)
5. SVM are designed not to overfit the training data even in infinite dimension.
6. Kernel engineering for complex objects is a hot topic!

Kernel and kernel methods summary

Performance

- State-of-the-art in many real-world applications
- Resistant to large dimensions

Data representation

- Data do not need to be explicitly vectors
- A similarity function $K(x, x')$ between data is enough
- K must be symmetric and positive definite

Kernels in chemoinformatics

- We need kernels for molecules!
- Inner products of classical vector / fingerprint representations will work, but we can do better.

Kernel and kernel methods summary

Performance

- State-of-the-art in many real-world applications
- Resistant to large dimensions

Data representation

- Data do not need to be explicitly vectors
- A similarity function $K(x, x')$ between data is enough
- K must be symmetric and positive definite

Kernels in chemoinformatics

- We need kernels for molecules!
- Inner products of classical vector / fingerprint representations will work, but we can do better.

Kernel and kernel methods summary

Performance

- State-of-the-art in many real-world applications
- Resistant to large dimensions

Data representation

- Data do not need to be explicitly vectors
- A similarity function $K(x, x')$ between data is enough
- K must be symmetric and positive definite

Kernels in chemoinformatics

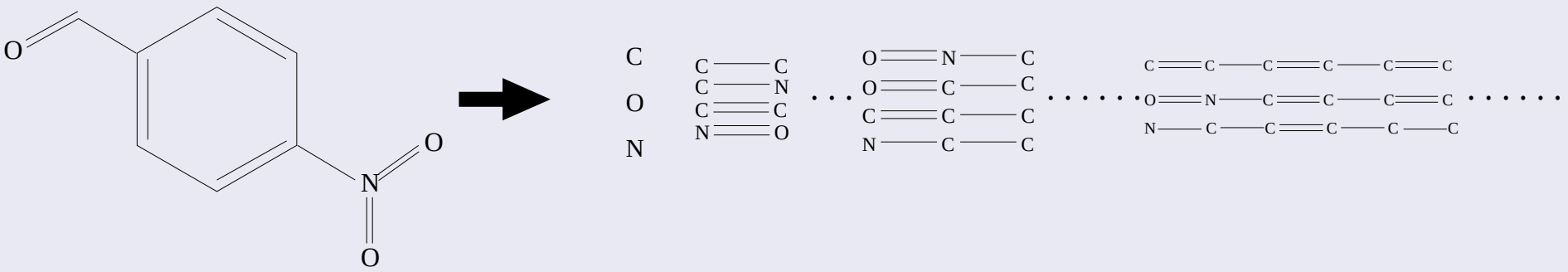
- We need kernels for molecules!
- Inner products of classical vector / fingerprint representations will work, but we can do better.

Outline

Motivation: 2D Fingerprints

Features

A vector indexed by a large set of molecular fragments



Pros

- Many features
- Easy to detect

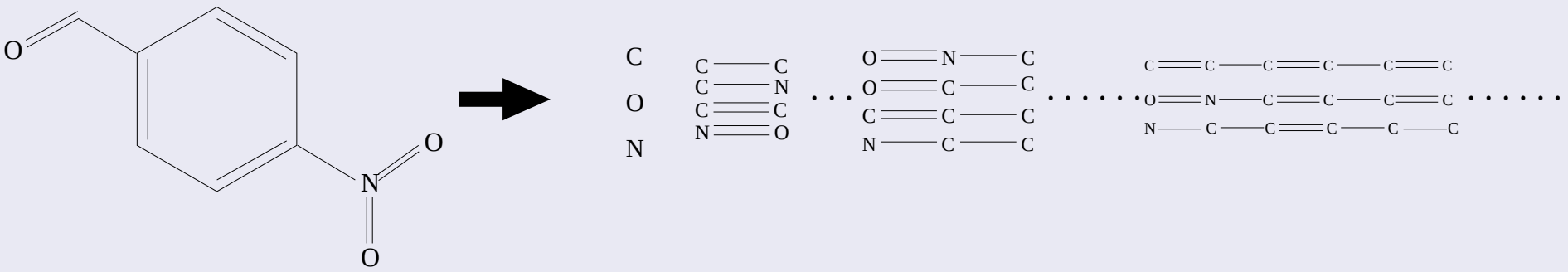
Cons

- Too many features?
- Hashing $\Longrightarrow$ clashes

Motivation: 2D Fingerprints

Features

A vector indexed by a large set of molecular fragments



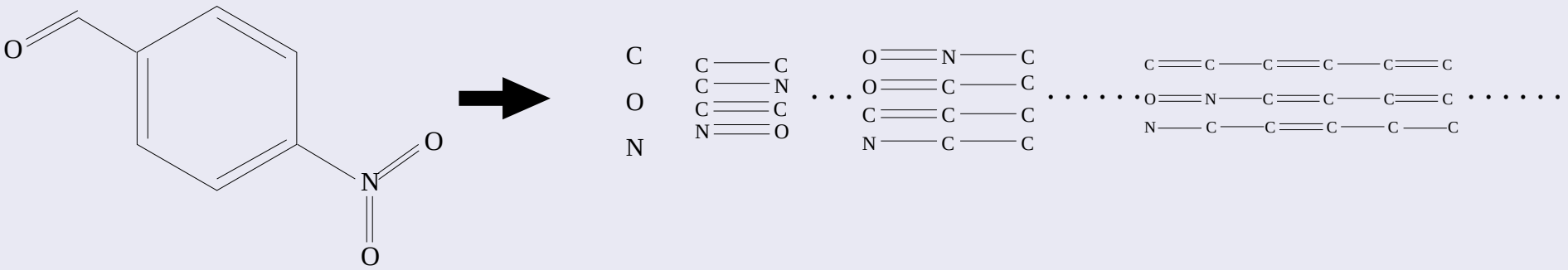
Pros

- Many features
- Easy to detect

Cons

- Too many features?
- Hashing $\Longrightarrow$ clashes

SVM approach



Let $\Phi(x)$ the vector of fragment counts:

- Long fragments lead to large dimensions :
  SVM can learn in high dimension
- $\Phi(x)$ is too long to be stored, and hashes induce clashes:
  SVM do not need $\Phi(x)$, they just need the kernel

$$K(x, x') = \phi(x)^\top \phi(x') \; .$$

2D fingerprint kernel

Definition

- For any $d > 0$ let $\phi_d(x)$ be the vector of counts of all fragments of length d:

$$\phi_1(x) = (\quad \#(\mathtt{C}), \#(\mathtt{O}), \#(\mathtt{N}), \quad \ldots)^\top$$
$$\phi_2(x) = (\quad \#(\mathtt{C\text{-}C}), \#(\mathtt{C{=}O}), \#(\mathtt{C\text{-}N}), \quad \ldots)^\top \quad \text{etc...}$$

- The 2D fingerprint kernel is defined, for $\lambda < 1$, by

$$K_{2D}(x, x') = \sum_{d=1}^{\infty} \lambda^d \phi_d(x)^\top \phi_d(x') \; .$$

- This is an inner product in the space of 2D fingerprints of infinite length.

2D kernel computation

Theorem

The 2D fingerprint kernel between two molecules x and x' can be computed with a worst-case complexity $O\left((|x| \times |x'|)^3\right)$ (much faster in practice).

Remarks

- The complexity is not related to the length of the fragments considered (although faster computations are possible if the length is limited).
- Solves the problem of clashes and memory storage.
- Allows to work with infinite-length fingerprints without computing them!

MINES PARIS ARMINES

2D kernel computation

Theorem

The 2D fingerprint kernel between two molecules x and x' can be computed with a worst-case complexity $O\left((|x| \times |x'|)^3\right)$ (much faster in practice).

Remarks

- The complexity is not related to the length of the fragments considered (although faster computations are possible if the length is limited).
- Solves the problem of clashes and memory storage.
- Allows to work with infinite-length fingerprints without computing them!

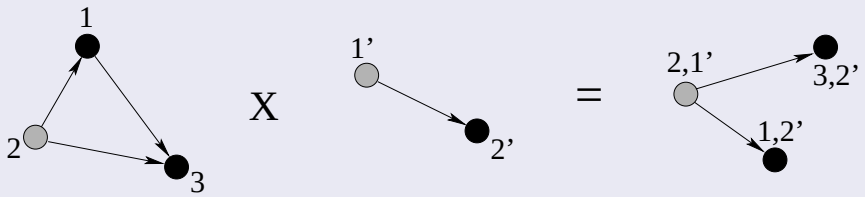
MINES PARIS ARMINES

2D kernel computation: Sketch (1/2)

- Let $\mathcal{F}(x)$ be the set of fragments of the molecule x (with repeats). Let $l(f)$ be the label of fragment f (e.g., $c-c$), and $|f|$ its length. Then the kernel can be rewritten:

$$K_{2D}(x,x') = \sum_{f\in\mathcal{F}(x)} \sum_{f\in\mathcal{F}(x)} \mathbf{1}\left(\mathbf{l(f)} = \mathbf{l(f')}\right) \lambda^{|\mathbf{f}|}.$$

- For any two molecules (graphs) G_1 and G_2, compute the product graph $G = G_1 \times G_2$:



- There is a bijection between:
  - each fragments of G,
  - each pair of fragments in G_1 and G_2 with same label.

2D kernel computation: Sketch (2/2)

- Therefore the kernel can be rewritten:

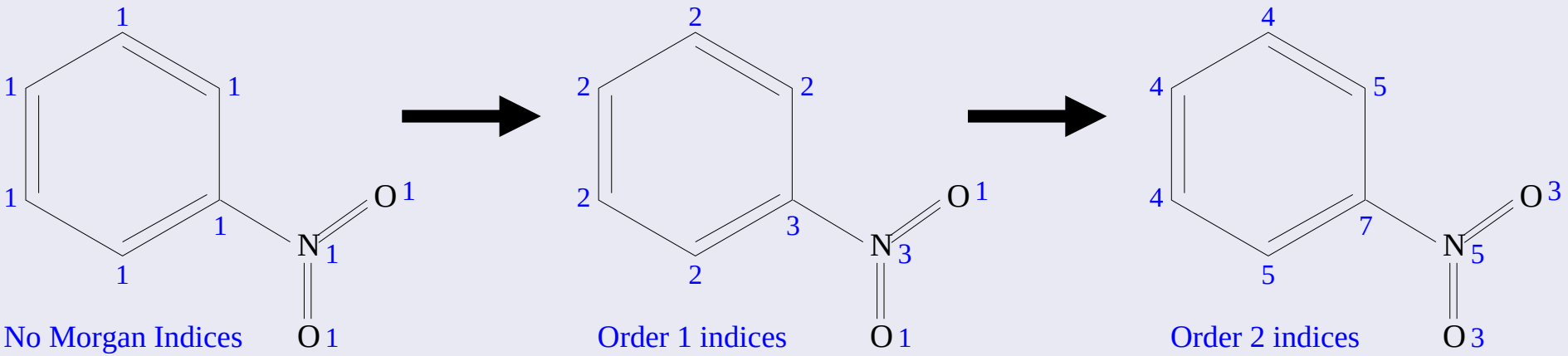
$$K_{2D}(x, x') = \sum_{f \in \mathcal{F}(G)} \lambda^{|f|} .$$

- Let A be the adjacency matrix of G. For any $d \geq 1$, $\left[A^d\right]_{i,j}$ is the number of fragments of length d starting in i and ending in j.
- Therefore the kernel is the sum of the elements of the matrices:

$$\lambda A + \lambda^2 A^2 + \lambda^3 A^3 + \ldots = (I - \lambda A)^{-1} - I .$$

Extensions 1: label enrichment

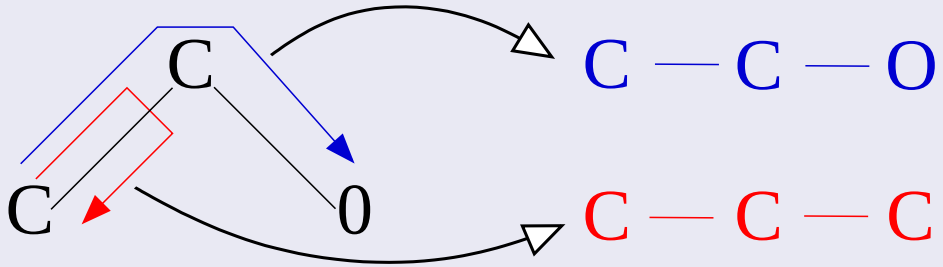
Atom relabebling with the Morgan index



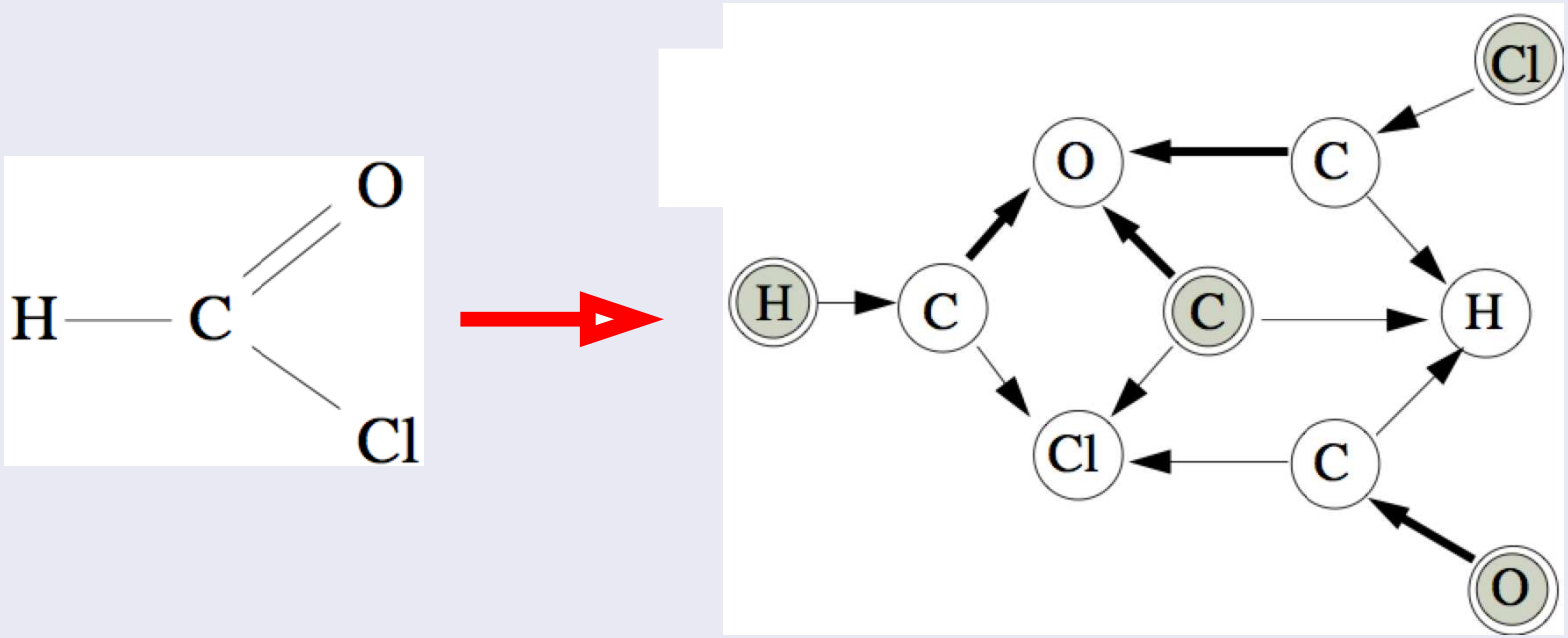
- Compromise between fingerprints and structural keys features.
- Other relabeling schemes are possible.
- Faster computation with more labels (less matches implies a smaller product graph).

Extensions 2: filter out tottering fragments

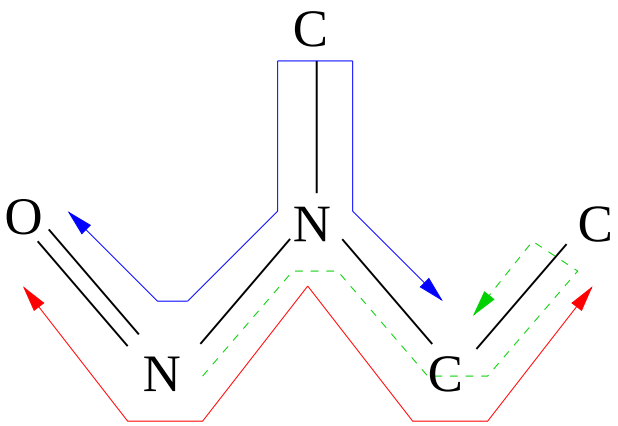
Tottering fragments

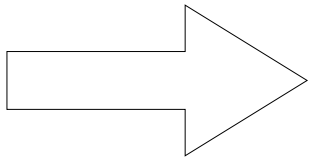


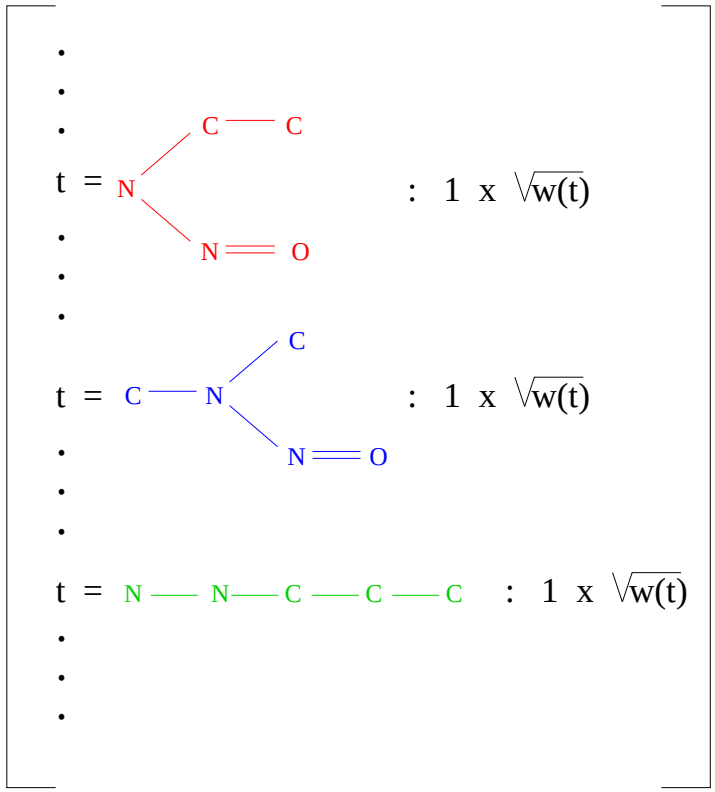
Solution: graph transform



Extensions 3: tree-like fragments







Experiments

MUTAG dataset

- aromatic/hetero-aromatic compounds
- high mutagenic activity /no mutagenic activity
- 188 compouunds: 125 + / 63 -

Results

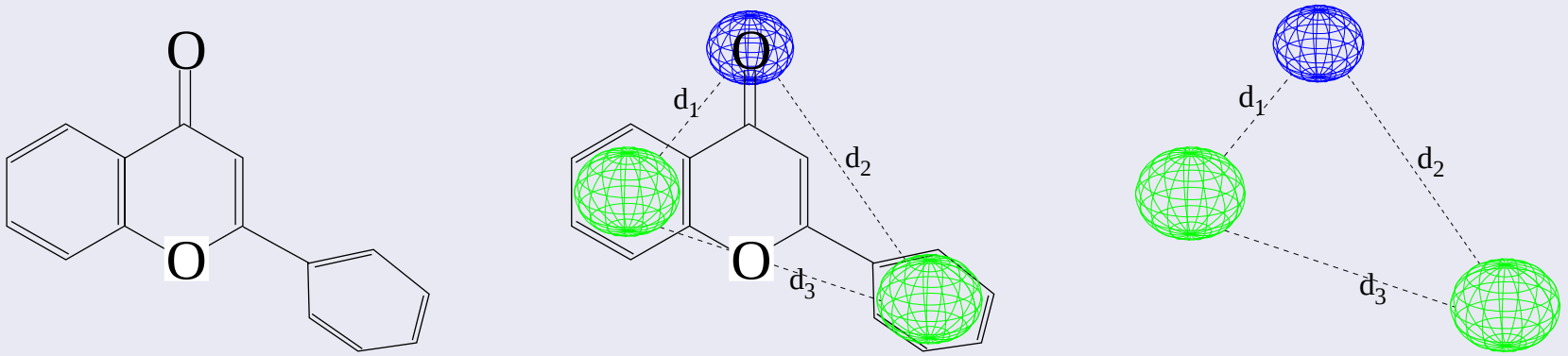
10-fold cross-validation accuracy

Method	Accuracy
Progol1	81.4%
2D kernel	91.2%

Outline

Space of pharmacophore

3-points pharmacophores



A set of 3 atoms, and 3 inter-atom distances:

$$\mathcal{T} = \{((x_1, x_2, x_3), (d_1, d_2, d_3)), x_i \in \{\text{atom types}\}; d_i \in \mathbb{R}\}$$

3D fingerprint kernel

Pharmacophore fingerprint

1. Discretize the space of pharmacophores $\mathcal{T}$ (e.g., 6 atoms or groups of atoms, 6-7 distance bins) into a finite set $\mathcal{T}_d$
2. Count the number of occurrences $\phi_t(x)$ of each pharmacophore bin t in a given molecule x, to form a pharmacophore fingerprint.

3D kernel

A simple 3D kernel is the inner product of pharmacophore fingerprints:

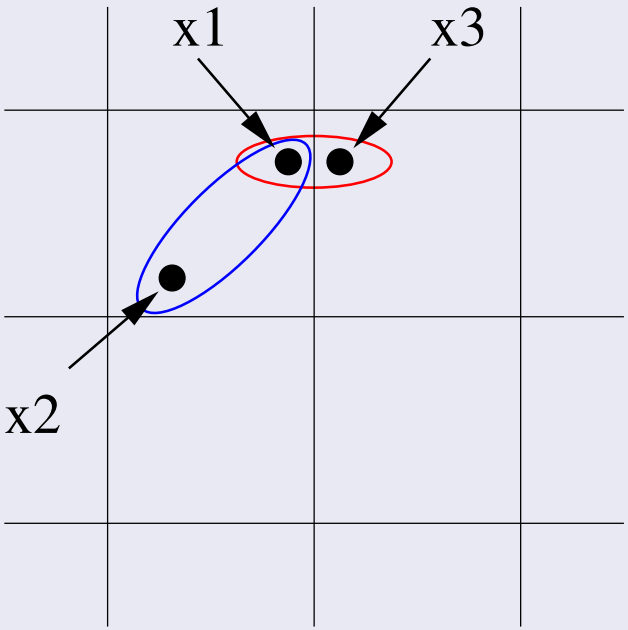
$$K(x, x') = \sum_{t \in \mathcal{T}_d} \phi_t(x) \phi_t(x') \ .$$

Discretization of the pharmacophore space

Common issues

1. If the bins are too large, then they are not specific enough
2. If the bins are too large, then they are too specific

In all cases, the arbitrary position of boundaries between bins affects the comparison:



$$\rightarrow d(x_1, x_3) < d(x_1, x_2)$$
$$\text{BUT} \quad \text{bin}(x_1) = \text{bin}(x_2) \neq \text{bin}(x_3)$$

Kernels between pharmacophores

A small trick

$$
\begin{aligned}
K(x,y) &= \sum_{t \in \mathcal{T}_d} \phi_t(x)\phi_t(y) \\
&= \sum_{t \in \mathcal{T}_d} \Big(\sum_{p_x \in \mathcal{P}(x)} \mathbf{1}(\text{bin}(\mathbf{p_x}) = \mathbf{t}) \Big) \Big(\sum_{p_y \in \mathcal{P}(y)} \mathbf{1}(\text{bin}(\mathbf{p_y}) = \mathbf{t}) \Big) \\
&= \sum_{p_x \in \mathcal{P}(x)} \sum_{p_y \in \mathcal{P}(y)} \mathbf{1}(\text{bin}(\mathbf{p_x}) = \text{bin}(\mathbf{p_y}))
\end{aligned}
$$

General pharmacophore kernel

$$
K(x,y) = \sum_{p_x \in \mathcal{P}(x)} \sum_{p_y \in \mathcal{P}(y)} K_P(p_x, p_y)
$$

New pharmacophore kernels

- Discretizing the pharmacophore space is equivalent to taking the following kernel between individual pharmacophores:

$$K_P(p_1, p_2) = \mathbf{1}\left(\text{bin}(\mathbf{p_x}) = \text{bin}(\mathbf{p_y})\right)$$

- For general kernels, there is no need for discretization!
- For example, is $d(p_1, p_2)$ is a Euclidean distance between pharmacophores, take:

$$K_P\left(p_1, p_2\right) = \exp\left(-\gamma d\left(p_1, p_2\right)\right) \ .$$

MINES PARIS ARMINES

Experiments

4 public datasets

- BZR: ligands for the benzodiazepine receptor
- COX: cyclooxygenase-2 inhibitors
- DHFR: dihydrofolate reductase inhibitors
- ER: estrogen receptor ligands

	TRAIN		TEST	
	Pos	Neg	Pos	Neg
BZR	94	87	63	62
COX	87	91	61	64
DHFR	84	149	42	118
ER	110	156	70	110

Experiments

Results (accuracy)

Kernel	BZR	COX	DHFR	ER
2D (Tanimoto)	71.2	63.0	76.9	77.1
3D fingerprint	75.4	67.0	76.9	78.6
3D not discretized	**76.4**	**69.8**	**81.9**	**79.8**

MINES PARIS ARMINES

Outline

Summary

- SVM is a powerful and flexible machine learning algorithm. The kernel trick allows the manipulation of non-vectorial objects at the cost of defining a kernel function.
- The 2D kernel for molecule extends classical fingerprint-based approches. It solves the problem of bit clashes, allows infinite fingerprints and various extensions.
- The 3D kernel for molecule extends classical pharmacophore fingerprint-based approaches. It solves the problems of bit clashes and of discretization.
- Both kernels improve upon their classical counterparts, and provide competitive results on benchmark datasets.

MINES PARIS ARMINES

Acknowledgements

- Pierre Mahé (CBIO)
- Tatsuya Akutsu, Nobuhisa Ueda, Jean-Luc Perret (Kyoto University)
- Liva Ralaivola (U Marseille)

References

- Kashima, H., Tsuda, K., and Inokuchi, A. *Marginalized kernels between labeled graphs*. Proceedings of the 20th ICML, 2003, pp. 321-328.
- P. Mahé, N. Ueda, T. Akutsu, J.-L. Perret, and J.-P. Vert. *Graph kernels for molecular structure-activity relationship analysis with SVM*. J. Chem. Inf. Model., 45(4):939-951, 2005.
- P. Mahé, L. Ralaivola, V. Stoven, and J-P Vert. *The pharmacophore kernel for virtual screening with SVM*. J. Chem. Inf. Model., 46(5):2003-2014, 2006.
- P. Mahé and J.-P. Vert. *Graph kernels based on tree patterns for molecules*. Technical report HAL:ccsd-00095488, 2006.
- P. Mahé. *Kernel design for virtual screening of small molecules with support vector machines*. PhD thesis, Ecole des Mines de Paris, 2006.
- Open-source kernels for chemoinformatics: `http://chemcpp.sourceforge.net/`

MINES PARIS ARMINES