Support Vector Inductive Logic Programming

Review of an article

Oksana Korol

Content

- Main points and motivation
- Background:
  - Chemistry
  - Support Vector Machines
  - Inductive Logic Programming
  - Propositionalization
- Support Vector Inductive Logic Programming
- Results
- Conclusion

Main points and motivation

Reviewed article:

"**Support Vector Inductive Logic Programming**"

By Stephen Muggleton, Huma Lodhi,
Ata Amini, and Michael J. E. Sternberg

Main points and motivation

- Goal: prediction of toxicity
- Intersection of SVM and ILP
- SVM provides for dimensionality independence
- ILP kernel captures relational information

Chemistry

- Molecular structure of toxic chemicals

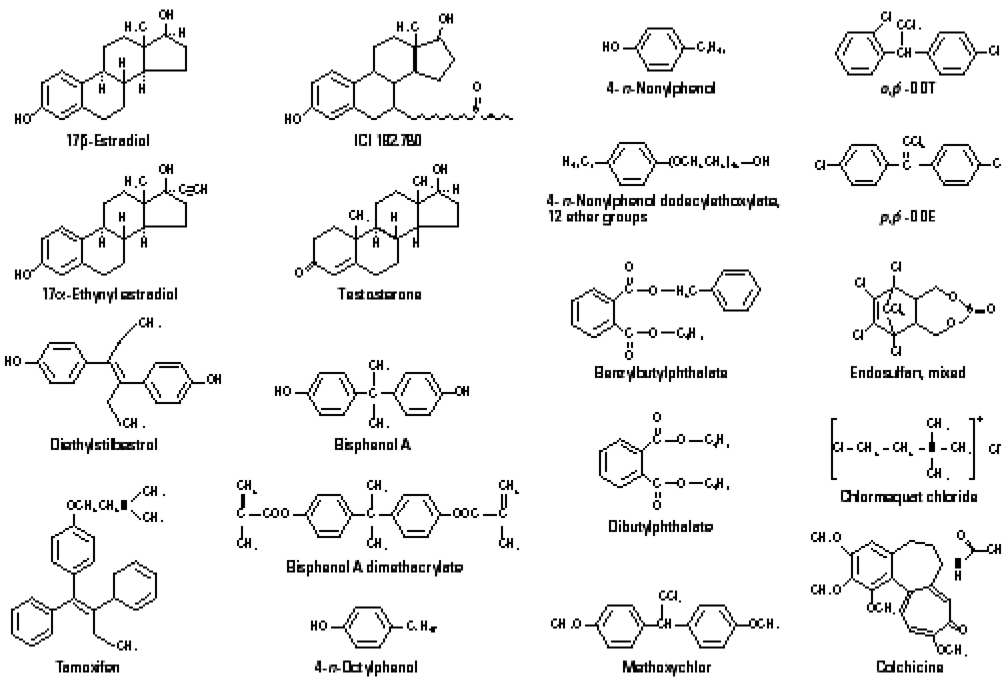


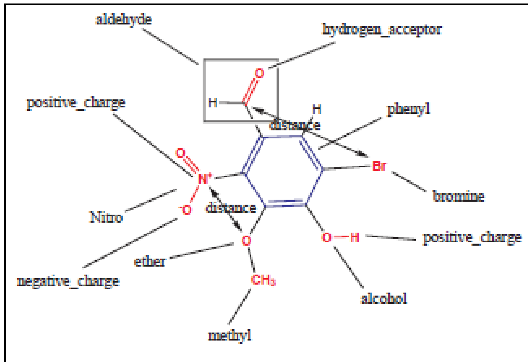
Table 1. COMPACT parameters for 30 chemicals.

Chemical	Area/ Depth^2	ΔE (eV)	COMPACT radius[a]	CYP1 interaction	Molecular collision diameter	CYP2E[b] activation	Overall[c] COMPACT prediction
1. Scopolamine	1.6	15.4	8.6	–	7.9	–	–
2. Codeine	1.4	14.1	7.9	–	8.1	–	–
3. 1,2-Dihydro-2,2,4-trimethylquinoline	1.8	13.1	7.0	–	6.9	–	–
4. Nitromethane	1.4	14.8	8.3	–	4.7	+	+
5. Tetrahydrofuran	1.6	20.9	13.0	–	5.1	–	–
6. t-Butylhydroquinone	1.8	14.3	7.7	–	6.7	–	–
7. Ethylbenzene	2.1	16.7	9.2	–	6.0	–	–
8. Chloroprene	3.7	13.5	5.7	±	5.3	+	+
9. Cobalt sulfate	NP	NP	NP	NP	NP	NP	NP
10. D & C Yellow No. 11	2.0	13.0	6.8	–	7.8	–	–
11. Isobutyraldehyde	1.5	17.5	10.2	–	5.3	–	–
12. Molybdenum trioxide	NP	NP	NP	NP	NP	NP	NP
13. 1-Chloro-2-propanol	1.5	16.4	9.3	–	5.3	–	–
14. Diethanolamine	2.9	20.3	11.9	–	5.8	–	–
15. Phenolphthalein	2.0	13.9	7.3	–	8.1	–	–
16. Pyridine	4.3	16.2	7.6	–	5.2	–	–
17. Xylenesulfonic acid	3.7	13.9	6.0	±	6.7	–	±
18. Furfuryl alcohol	2.6	15.3	7.8	–	5.4	±	±
19. Primaclone	1.7	15.9	8.8	–	7.3	–	–
20. Ethylene glycol monobutyl ether	3.3	20.5	11.9	–	6.4	–	–
21. Gallium arsenide	NP	NP	NP	NP	NP	NP	NP
22. Isobutene	2.3	17.9	10.0	–	5.1	–	–
23. Methyleugenol	1.8	14.8	8.0	–	7.1	–	–
24. Oxymetholone	2.7	13.9	6.7	–	8.5	–	–
25. Anthraquinone	9.4	13.5	4.3	+	7.0	–	+
26. Emodin	6.6	12.3	3.1	+	7.5	–	+
27. Citral	1.5	14.2	7.9	–	6.8	–	–
28. Sodium nitrite	NP	NP	NP	NP	NP	NP	NP
29. Cinnamaldehyde	6.9	13.3	3.9	+	6.2	+	+
30. Vanadium pentoxide	NP	NP	NP	NP	NP	NP	NP

Abbreviations: NP, not predicted; +, positive; –, negative. [a]COMPACT radius, $\sqrt{(\Delta E - 9.5)^2 + (a/d^2 - 7.8)^2}$. Positive CYP1 if COMPACT radius <5.5; ± if radius is between 5.5 and 6.5. [b]CYP2E activation, molecular collision diameter <6.5 and ΔE < 15.0. [c]Overall COMPACT prediction is the summation of CYP1 interaction and CYP2E activation.

Images from Environmental Health Perspectives

Chemistry



Br: 1
C: 8
H: 6
N: 1
O: 5
Single Bonds: 16
Double and Aromatic Bonds: 5

a) SVILP representation	b) Gärtner/Chevaleyre representation

Fig. 1. Molecule represented using a) SVILP representation which employs a kernel based on domain-expert informed chemical background knowledge indicated by the annotations on the figure and b) Gärtner/Chevaleyre bag-of-atoms uses Multi-Instance (MI) kernel based on frequency of occurrences of atoms and atom pairs.

Inductive Logic Programming

- Introduced by Stephen Muggleton in 1992

Inductive Logic Programming (ILP)

=

Machine Learning Λ Logic Programming

=

Learning with Logic

Inductive Logic Programming

- Induction – reasoning from specific to general
- Logic programs are the set of Horn clauses that follow the rules of the first order logic:

$$mother(X, Y) \longleftarrow parent(X, Y) \wedge female(X) .$$

$female(Jane).$	$female(Ann).$
$male(Jack).$	$male(John).$
$parent(Jane, Ann).$	$parent(Jack, Ann).$

- Questions: Is Jane a mother of Ann?
  Who is a mother of Ann?

Inductive Logic Programming

ILP is represented in logic programs which are used to derive a solution to a problem by inducing a hypothesis based on a set of positive and negative examples.

Inductive Logic Programming

- **Concept learning**: given a background knowledge B and experimental observations E (consisting of positive $E+$ and negative $E-$ examples) find a hypothesis H such that:

$$B \land H \models E$$

- B, E and H are each logic programs

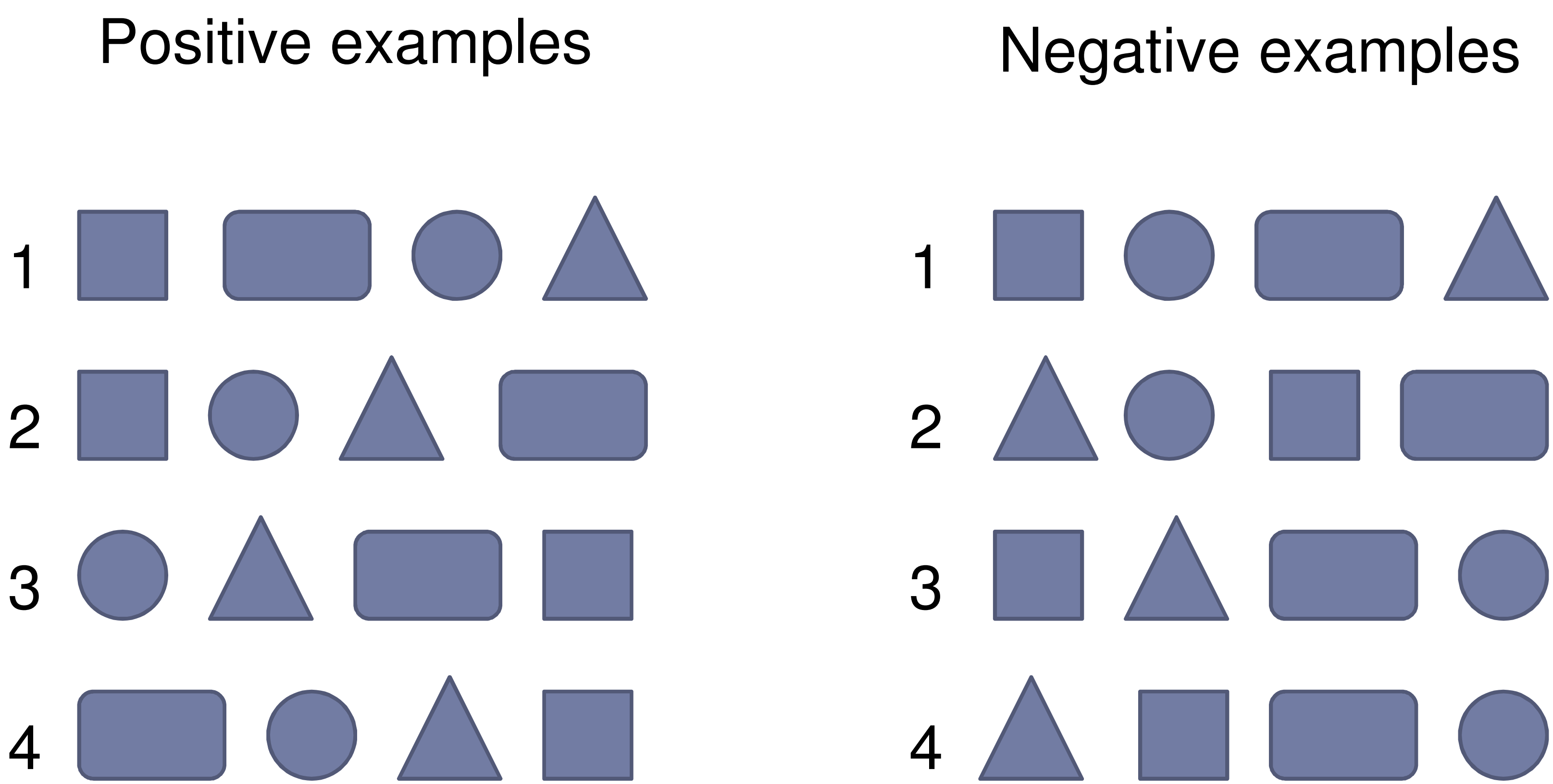
Inductive Logic Programming

- *B*, *H*, and *E* should satisfy the following conditions:

Prior Satisfiability. $B \wedge E^- \not\models$
Posterior Satisfiability. $B \wedge H \wedge E^- \not\models$
Prior Necessity. $B \not\models E^+$
Posterior Sufficiency. $B \wedge H \models E^+$

Inductive Logic Programming

- Classify the following:

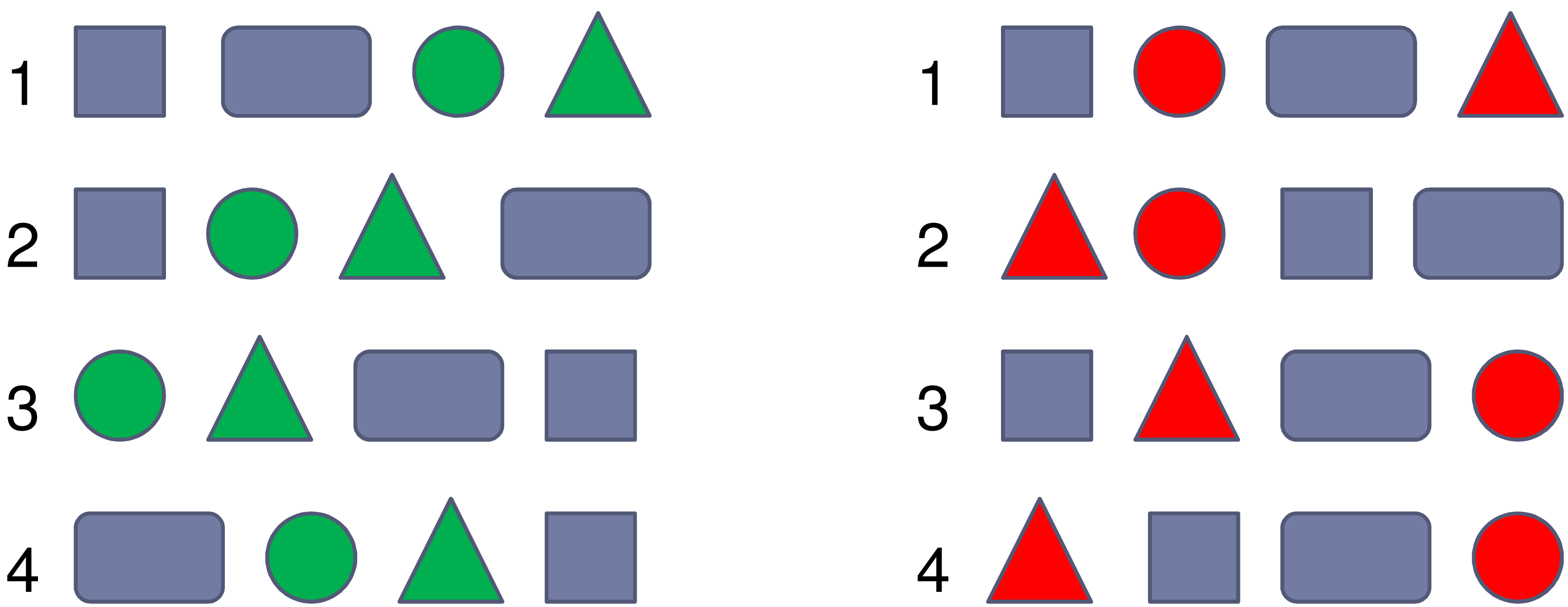


Inductive Logic Programming

- Classify the following:

Positive examples

Negative examples



Inductive Logic Programming

- *B* would specify following rules:
  - before(X, Y) :- <*when position of X is less than position of Y*>
  - adjacent(X, Y) :- <*when there is no other object between X and Y*>
- Then the resulting theory *H* will be:
  - positive :- before(circle, triangle), adjacent(triangle, circle).

Support Vector Machines

- Take any problem and transform it into a high dimensional space, so that it becomes linearly separable, but

- Calculations to obtain the separability plane can be done in the original input space *(kernel trick)*

Info from CSI5387 lecture notes by Dr. Stan Matwin

Support Vector Machines

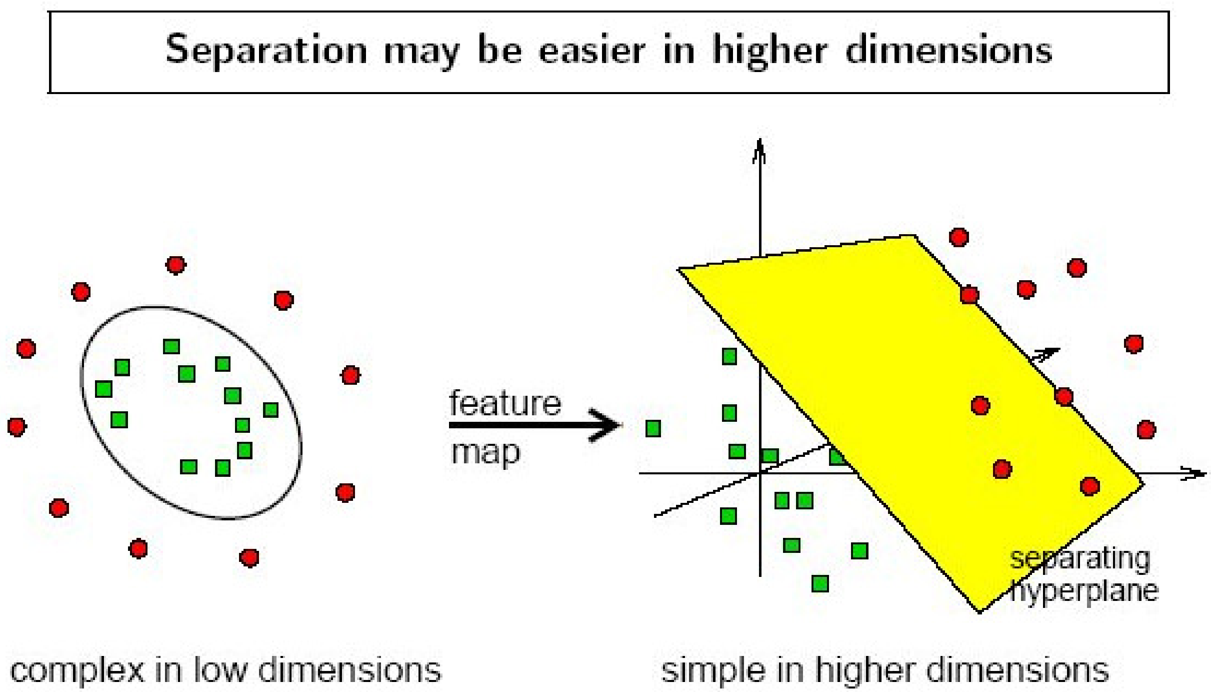
- *I.e. SVM learning process consists of 2 stages:*

1. Map the input data, $d_1, \ldots, d_n \in D$, into some higher dimensional space H through a non-linear mapping φ that is given by $\varphi : D \rightarrow H$.

2. Construct a linear function f in the space

Support Vector Machines

- The kernel function may transform the data into a higher dimensional space to make it possible to perform the separation.



Info from http://www.dtreg.com/svm.htm

Propositionalization

- Propositionalization – techniques to transform relational (first order logic) representation to propositional (fixed sized feature vectors)
- Involves construction of structural features from relational background knowledge
- => Any propositional learner can be applied after propositionalization
- SVILP is similar in its use of support-vector technology to the domain-dependent bottom-up propositionalisation approach

Bottom-Up Propositionalization

- Discover fragments that occur frequently in the dataset (ex. circle followed by triangle)
- Bottom-up approach to fragment generation: generate only those fragments that really occur in the examples
- Algo: depth-first search for fragments for each data point (ex. sequence)

Support Vector ILP

- In short: SVM with ILP as a kernel function
- Like in ILP, assume background knowledge B, examples E and a hypothesis H
- SVILP bases a kernel on the predictions of the clauses h in H

Support Vector ILP

- Kernel is built by forming a binary hypothesis-instance association matrix $M: h_i \times d_j$, where $h_i \in H$ and $d_j \in D$.
- For each hypothesis clause h in H:

$$h : D \rightarrow \{True, False\}.$$

- Conversely the τ function gives the hypothesised clauses covering any particular instance:

$$\tau\,(d_i) = \{h : \exists h \in H, (B, h \models d_i)\}$$

- So the kernel function is as follows:

$$K(d_i, d_j) = f(\tau\,(d_i) \cap \tau\,(d_j))$$

Results

- Tested on the new DSSTox dataset (as opposed to Mutagens)
- Used 5-fold cross validation with mean squared error (MSE) and R-squared evaluation
- Compared results with well known QSAR software TOPKAT(Toxicity Prediction by Komputer Assisted Technology)
- Also compared to following techniques: partial least squares (PLS), multi instance kernels (MIK) , an RBF kernel

Results

	MSE	R-squared
CHEM	0.811	0.519
PLS	0.671	0.593
MIK	0.838	0.503
SVILP	**0.574**	**0.655**

Fig. 8. MSE and R-squared for CHEM, PLS, MIK and SVILP.

	Accuracy
ILP (CProgol5.0)	55
CHEM	58
PLS	71
MIK	60
SVILP	**73**

Fig. 9. Accuracy for ILP, CHEM, PLS, MIK and SVILP.

Conclusion

- Accuracy is good, but perhaps not the best way to evaluate the approach
- No mention of the performance time

- The kernel works within the standard ILP setting of generalisation with respect to background knowledge (not just atomic generalization)
- SVILP method shows significant improvement with respect to the other methods
- Follow up work confirms this (see references)

References / Further reading

- Muggleton, S., Lodhi, H., Amini, A., Sternberg, M.J.E. "**Support vector inductive logic programming**". In Proceedings of the Eighth International Conference on Discovery Science, volume 3735 of LNAI, 2005.
- Cannon, E.O., Amini, A., Bender, A., Sternberg, M.J.E, Muggleton, S.H., Glen. R.C., Mitchell, J.B.O. "**Support vector inductive logic programming outperforms the naive Bayes classifier and inductive logic programming for the classification of bioactive chemical compounds**". In Journal of Computer-aided Molecular Design, Vol. 21, No. 5, MAY 2007.
- Muggleton, S.H. "**Inverse Entailment and Progol**". In New Generation Computing, 13 (1995) 245-286.
- Kramer, S., E, F. "**Bottom-up propositionalisation**". In: Proceedings of the ILP-2000 Work-In-Progress Track. Imperial College, London (2000) 156–162

Questions