Robust Classification using Mixtures of Dependency Networks

José A. Gámez Juan L. Mateo Thomas D. Nielsen José M. Puerta

University of Castilla-La Mancha and Aalborg University

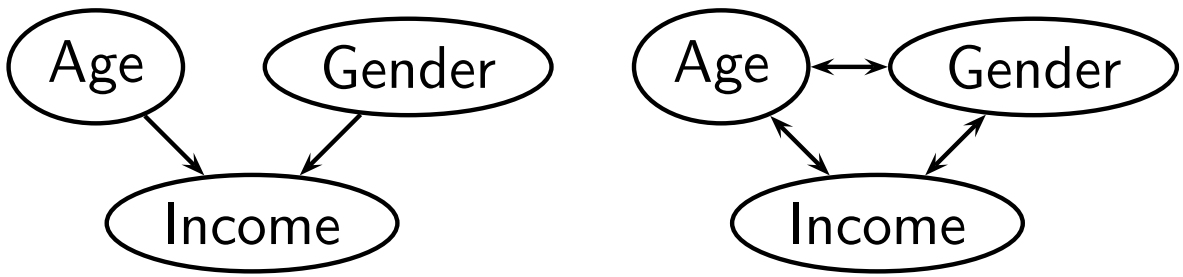
PGM - September 2008

Introduction

- Dependency networks (DNs) have been used as probabilistic models in several fields, mainly due to their ease of learning.
- In this work we consider the use of DNs for classification. In particular,
  - we focus on mixtures of DNs (multinets).
  - we investigate the possibility of reusing learning results across classes in order to reduce complexity and improve robustness.

Dependency networks

- A DN is a pair $(\mathbb{G}, \mathbf{P})$, where $\mathbb{G}$ is a directed graph, potentially cyclic, and $\mathbf{P}$ is a set of local probability distributions, one for each variable (Heckerman et al., 2000).
- The parent set for a variable X is the MB for X.



BN DN

- We focus on *general* DNs, which are suitable for automatic learning:

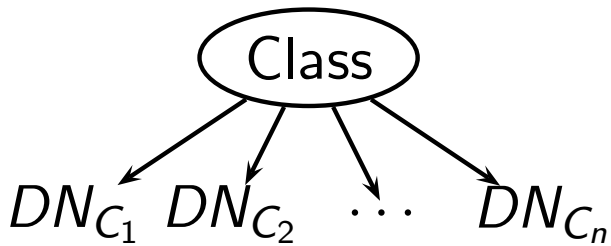
$$P(\mathbf{X}) \approx \prod_i P(X_i|Pa_i).$$

Note that in these networks we may have to deal with inconsistencies.

Mixtures of Dependency Networks

Multinets are widely used in classification, especially due to their ability to represent *contextual* relationships.

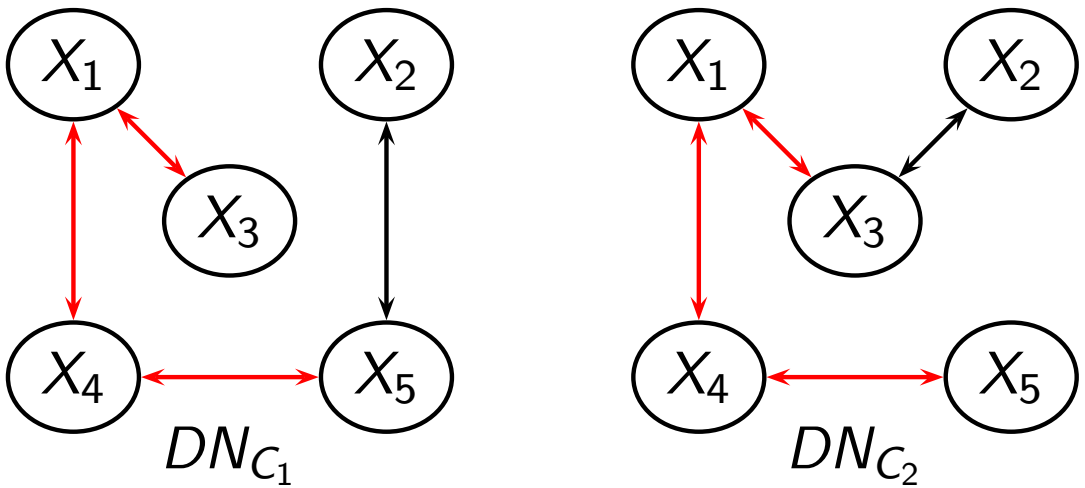
- We propose to use the same idea, but with DNs instead of BNs.



- We need to learn a DN for every class value (note that each model can be learned independently)
- Assuming that the data is faithful to a graph we can use a Markov blanket learner to establish the parent sets.

Motivation for re-usability

Assuming that the independence statements are not disjoint between the different class conditional DNs we may be able to **reuse** previous learned structures/MBs.



We hypothesize two advantages

- we may speed up learning.
- we may obtain a more robust classifier when data is scarce for some of the classes.

Learning Markov blankets

We use the IAMB algorithm for learning the Markov blankets.

// Phase I (forward)

MB $= \emptyset$

while MB *has changed* **do**

  $Y = \arg\max_{X \in U \setminus (\mathbf{MB} \cup \{T\})} dep(X, T|\mathbf{MB})$

  if $Y \not\perp\!\!\!\perp T|\mathbf{MB}$ **then**

    MB $= \mathbf{MB} \cup \{Y\}$

// Phase II (backwards)

foreach $X \in \mathbf{MB}$ **do**

  if $X \perp\!\!\!\perp T|(\mathbf{MB} \setminus X)$ **then**

    MB $= \mathbf{MB} \setminus \{X\}$

return MB

IAMB was chosen over e.g. PCMB due to ease of implementation, tendency to generate smaller MBs, and immediate support for re-usability.

Learning Markov blankets

We use the IAMB algorithm for learning the Markov blankets.

```
// Phase I (forward)
MB = ∅
while MB has changed do
    Y = arg max_{X ∈ U\(MB∪{T})} dep(X, T|MB)
    if Y ⫫̸ T|MB then
        MB = MB ∪ {Y}
// Phase II (backwards)
foreach X ∈ MB do
    if X ⫫ T|(MB\X) then
        MB = MB\{X}
return MB
```

IAMB was chosen over e.g. PCMB due to ease of implementation, tendency to generate smaller MBs, and immediate support for re-usability.

SeededIAMB

```
// Phase I (forward)
MB = Seed // Seeded with previous learned structures
while MB has changed do
    Y = arg max_{X ∈ U\(MB∪{T})} dep(X, T|MB)
    if Y ⫫̸ T|MB then
        MB = MB ∪ {Y}
// Phase II (backwards)
foreach X ∈ MB do
    if X ⫫ T|(MB\X) then
        MB = MB\{X}
return MB
```

Theorem

If the independence tests are correct and that the database D is an iid. sample from a probability distribution P faithful to a DAG $\mathcal{G}$, then SeededIAMB identifies the true MB for the target variable T.

Re-usability: What to reuse?

More data yields more reliable MB estimates, so we first order the classes according to the number of instances.

Selecting what to reuse:

- We use the local log-likelihood

$$\frac{1}{N_j}\sum_{l=1}^{N_j}\log P(X|MB(X)_i, DN_i)(\mathbf{d}_l),$$

  as an indication of how well $MB(X)_i$ predicts X in $\mathcal{D}_{c_j}$.
- We use the empty network as threshold for the two strategies:
  - *BESTlogL* uses only the best structure above the threshold.
  - *THRESHOLDlogL* uses the union of all local structures above the threshold.
- Three other "blind" strategies:
  - *First* uses the first model as seed.
  - *Intersection* and *Union* uses the intersection and union, respectively, of the MBs for all the previously learned models.

Datasets

dataset	insts.	vars.	|class|	dataset	insts.	vars.	|class|
australian	690	14	2	labor	57	12	2
autos	205	23	7	mushroom	8124	23	2
balance	625	5	3	nursery	12960	9	5
breast-cancer	286	10	2	page-blocks	5473	11	5
breast-w	699	10	2	post-op	90	9	3
car	1728	7	4	segment	2310	20	7
cmc	1473	10	3	soybean	683	36	19
diabetes	768	7	2	spambase	4601	56	2
ecoli	336	7	8	vehicle	846	19	4
heart	270	14	2	vote	435	17	2
hepatitis	155	20	2	vowel	990	14	11
ionosphere	351	34	2	waveform	5000	20	3
iris	150	5	3	wine	178	14	3
kr-vs-kp	3196	37	2	zoo	101	17	7

Datasets from the UCI repository.

Algorithms

- Algorithms selected for comparison:
  - J48
  - Multilayer perceptron (NN)
  - k-nearest neigbours (kNN); $k = 1$ and $k = 3$, inverse distance weighted
  - Support vector machine (SVM)
  - Naive Bayes (NB)
  - k-dependence Bayesian classifier (kDB); $k = 1, 2, 3, 4$
  - Tree augmented naive Bayes (TAN)
  - Multinet with Bayesian networks (MultiBN); independence tests (PC) and score metric (BIC, K2)
  - Another DN-based classifier (ChiSqDN)
- In the proposed algorithm we performed independence tests based on standard statistical tests as well as by comparing BIC scores.

Accuracy

	kDB	MultiBN	MultiDN	SVM
australian	84.64	85.42	**86.35**	84.93
autos	79.02	79.81	73.27	**82.14**
balance	74.05	73.82	74.08	74.11
breast-cancer	70.28	69.79	70.49	71.12
breast-w	96.22	96.57	97.34	96.68
car	93.26	91.68	91.01	92.45
cmc	**53.96**	52.49	53.29	53.96
diabetes	77.47	77.68	**79.27**	76.77
ecoli	84.82	85.12	82.26	83.87
heart	81.63	80.3	82.37	83.7
hepatitis	**87.88**	86.45	85.81	85.55
ionosphere	91.97	**92.65**	92.19	90.48
iris	95.07	94.53	96.27	**96.40**
kr-vs-kp	94.22	96.49	95.27	95.24

	kDB	MultiBN	MultiDN	SVM
labor	89.8	92.22	**94.72**	92.29
mushroom	99.87	**100.00**	99.95	99.99
nursery	93.26	95.57	93.73	93.06
page-block	95.75	96.42	96.24	**96.81**
post-op	66.22	66.89	66.89	69.56
segment	94.17	94.94	91.36	95.56
soybeam	91.6	**94.00**	93.35	92.5
spam-base	92.73	93.65	92.58	**93.81**
vehicle	71.23	71.28	70.33	**73.14**
vote	93.75	93.89	94.16	**95.22**
vowel	73.17	69.82	64.99	79.07
waveform	82.25	81.79	81.26	**84.86**
wine	97.08	97.98	98.31	97.87
zoo	94.65	94.26	94.06	94.26
aver.	85.72	85.91	85.4	**86.62**

The best classifier on average is SVM. Only kDB (with $k = 1$), MultiBN, and MultiDN (with BIC) are comparable.

Learning Time

	kDB	MultiBN	MultiDN	SVM
australian	543	1036	249	**174**
autos	711	3226	951	**268**
balance	**30**	46	37	85
breast-cancer	89	372	94	**72**
breast-w	196	228	130	**68**
car	173	166	**114**	352
cmc	396	428	**281**	653
diabetes	83	96	**67**	81
ecoli	**42**	59	63	325
heart	213	256	129	**40**
hepatitis	348	433	254	**33**
ionosphere	3995	5716	1849	**105**
iris	**14**	22	17	60
kr-vs-kp	47422	63283	18906	**1545**

	kDB	MultiBN	MultiDN	SVM
labor	41	75	62	**29**
mushroom	25730	93243	13727	**3792**
nursery	2481	2087	**1285**	17293
page-block	1960	4760	**1443**	2763
post-op	**29**	58	37	61
segment	4875	14380	**2343**	2863
soybeam	8425	4374	4383	**2651**
spam-base	217476	79629796	39136	**8311**
vehicle	1557	15215	1428	**653**
vote	573	847	389	**49**
vowel	**792**	8088	994	1582
waveform	10810	6131	**3815**	10478
wine	144	148	122	**66**
zoo	**157**	166	217	303

SVM and MultiDN are significantly faster than the other two methods and there is no significant difference between them.

Computational savings

	Complexity[1]	Time
NOREUSE	2045	3303
BESTlogL	1920	3671
TRHESHOLDlogL	1925	3861
First	1915	3201
Intersection	**1899**	**3136**
Union	1945	3400

[1]Complexity is equal to the number of calls to the score function (or statistical tests made) times the number of variables involved in those calls.

- In all cases reusability reduce the complexity.
- Only *First* and *Intersection* employ less time than the baseline.
- The overhead for *BESTlogL* and *THRESHOLDlogL* is due to the computation of likelihood.

Robustness on scarce data

Difference in accuracy relative to learning without reusability.

	BESTlogL	*THRESHOLDlogL*	*First*	*Intersection*	*Union*
autos	1.17	1.07	1.07	-0.20	0.87
balance	0.00	0.00	-0.03	-0.03	0.00
breast-cancer	0.28	0.28	0.28	0.28	0.28
breast-w	0.00	0.00	0.00	0.00	0.00
car	0.00	0.00	0.00	0.00	0.00
cmc	0.19	0.20	0.20	0.11	0.29
diabetes	0.03	0.03	0.08	0.08	0.08
ecoli	0.00	0.00	0.00	0.00	0.00
hepatitis	0.26	0.26	0.65	0.65	0.65
ionosphere	-0.11	-0.11	-0.17	-0.17	-0.17
nursery	0.00	0.00	0.00	0.00	-0.58
page-block	0.04	0.04	0.05	0.04	0.05
post-op	0.00	0.00	0.00	0.00	0.00
soybeam	-0.09	0.03	-0.03	0.06	-0.70
spam-base	0.36	0.36	0.13	0.13	0.13
vote	0.09	0.09	0.23	0.23	0.23
zoo	0.00	0.00	0.00	0.00	0.20
average	0.13	0.13	0.14	0.07	0.08

BESTlogL, *THRESHOLDlogL*, and *First* improve significantly the accuracy.

Where does the improvement happen?

	0	1
0	**21.8**	11.8
1	10.2	111.2

Without

	0	1
0	**22.6**	11.6
1	9.4	111.4

First

- Instance distribution in `hepatitis` dataset: **32**,123
- With reusability the overall accuracy is increased but specially in the class with less data

Conclusions

- A mixture of DNs based classifier seems to hold some potential wrt. accuracy and computational cost.
- As expected, reusability can lead to a reduction in the number of computations required.
  - This saving can result in a reduced learning time; mainly for the uninformed selection strategies.
- The idea of reusability appears interesting, but further investigation is needed!