Dipartimento di Informatica

LACAM Laboratory
Machine Learning

Fast and Accurate Denstiy Estimation with Extremely Randomized Cutset Networks

Nicola Di Mauro ☯ ***Antonio Vergari*** ☯ Teresa M.A. Basile Floriana Esposito

☯ = *both authors contributed equally*

*19th September - **ECML-PKDD 2017** - Skopje, Macedonia*

Outline

Density Estimation

Density estimation is the unsupervised task of learning an estimator for the joint probability distribution $p(\mathbf{X})$ from a set of i.i.d. samples $\mathcal{D} = \{\mathbf{x}^i\}_{i=1}^m$ over RVs $\mathbf{X} = \{X_1, \ldots, X_n\}$

Given such an estimator, one uses it to answers probabilistic queries about configurations on $\mathbf{X}$, i.e. to do ***inference***.

The main challenge in density estimation is balancing:

- the ***representation expressiveness*** of a model
- the ***cost of learning*** it
- and the ***cost of performing inference*** on it.

Tractable Probabilistic Models (TPMs)

Classical Probabilistic Graphical Models like ***Bayesian Networks*** (**BNs**) and ***Markov Networks*** (**MNs**) are highly expressive but exact **inference is generally *NP-hard*** Roth 1996.

Tractable Probabilistic Models (**TPMs**) on the other hand, are density estimators for which some kind of ***exact* inference is *tractable***, i.e. *polynomial* in the number of RVs or their domains.

$\rightarrow$ Learning them may still be hard to scale

Product of Bernoullis (PoBs)

The least expressive, assuming all RVs to be independent:

$$\mathsf{p}(\mathbf{x}) = \prod_{i=1}^{n} \mathsf{p}(x_i)$$

Learning a PoB has linear time complexity $O(nm)$.

Chow-Liu Trees (CLTrees)

A *directed tree-structured model* Meilă and Jordan 2000 over $\mathbf{X}$ is a BN in which each node $X_i \in \mathbf{X}$ has at most one parent, Pa_{X_i}.

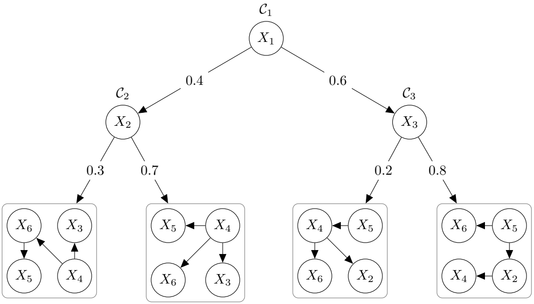
$$\mathsf{p}(\mathbf{x}) = \prod_{i=1}^{n} \mathsf{p}(x_i | \mathrm{Pa}_{x_i})$$

Learning a CLtree takes quadratic time $O(n^2(m + \log n))$.

Cutset Networks (CNets)

A Cutset Network (CNet) $\mathcal{C}$ is TPM represented via a weighted probabilistic model tree over $\mathbf{X}$ and recursively defined as:

1. a TPM $\mathcal{M}$, with scope$(\mathcal{M}) = \mathbf{X}$
2. a weighted disjunction (OR node) of two CNets $\mathcal{C}_0$ and $\mathcal{C}_1$ conditioned on RV $X_i \in \mathbf{X}$, with weights w_i^0 and w_i^1 s.t. $w_i^0 + w_i^1 = 1$, where scope$(\mathcal{C}_0) =$ scope$(\mathcal{C}_1) = \mathbf{X}_{\setminus i}$



Learning CNets I

All top-down greedy CNet learners can be unified in single template, LearnCNet:

LearnCNet($\mathcal{D}$, $\mathbf{X}$, α, δ, σ)

Input: a dataset $\mathcal{D}$ over RVs $\mathbf{X}$; α: δ min number samples; σ min number features
Output: a CNet $\mathcal{C}$ encoding $p_{\mathcal{C}}(\mathbf{X})$ learned from $\mathcal{D}$
if $|\mathcal{D}| > \delta$ **and** $|\mathbf{X}| > \sigma$ **then**
    $X_i \leftarrow \text{select}(\mathcal{D}, \mathbf{X}, \alpha)$ ▷ select the RV to condition on
    $\mathcal{D}_0 \leftarrow \{\xi \in \mathcal{D} : \xi[X_i] = 0\}, \mathcal{D}_1 \leftarrow \{\xi \in \mathcal{D} : \xi[X_i] = 1\}$
    $w_0 \leftarrow |\mathcal{D}_0|/|\mathcal{D}|, w_1 \leftarrow |\mathcal{D}_1|/|\mathcal{D}|$
    $\mathcal{C} \leftarrow w_0 \cdot \text{LearnCNet}(\mathcal{D}_0, \mathbf{X}_{\setminus i}, \alpha, \delta, \sigma) + w_1 \cdot \text{LearnCNet}(\mathcal{D}_1, \mathbf{X}_{\setminus i}, \alpha, \delta, \sigma)$
else
    $\mathcal{C} \leftarrow \text{learnLeafDistribution}(\mathcal{D}, \mathbf{X}, \alpha)$
return $\mathcal{C}$

different select implementations have different complexities

Learning CNets: entCNet& dCSN

entCNet Rahman, Kothalkar, and Gogate 2014 choosing X_i to lower approximate average joint entropy $\rightarrow$ $O(mn^2)$

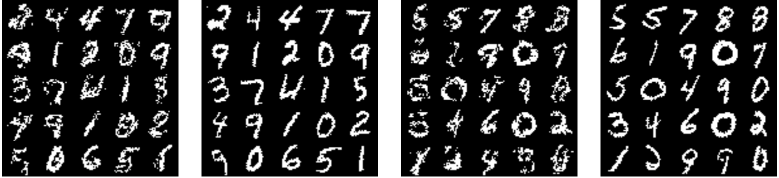
dCSN Di Mauro, Vergari, and Esposito 2015 choosing X_i in a principled way improving likelihood $\rightarrow$ $O(n^3(m + \log n))$

XCNets

XCNets (**Extremely Randomized CNets**) are CNets built by LearnCNet when select chooses one RV *completely at random*.

select time complexity $\rightarrow \boxed{O(1)}$!

Therefore a single XCNet is only only slightly less accurate than a CNet and as good at generating samples.



(a) (b) (c) (d)

Samples obtained from a CNet (a), resp. XCNet (c), learned on a binarized version of MNIST, and their nearest neighbor in training set (b), resp. (d).

Mixture of Experts

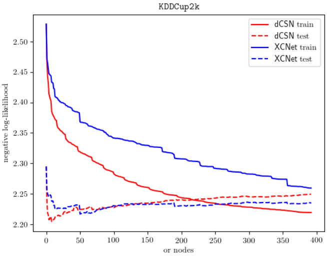
A single CNet can be seen as a peculiar ***mixture of experts***: the OR tree plays as a *deterministic gating function* and leaf distributions as *local experts*. A path $p = p_{(1)} p_{(2)} \cdots p_{(k)}$ connects the root to a single leaf $\mathcal{M}_l$ after observing $x_1 x_2 \cdots x_k$, Equation **??**.

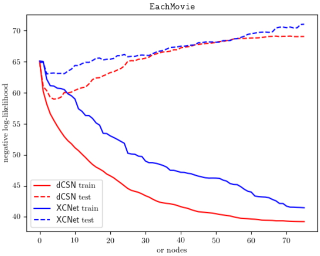
$\mathsf{p}_l(\mathbf{x}_{|\mathsf{sc}(\mathcal{C}) \setminus \mathsf{sc}(\mathcal{M}_l)})$ is decomposed by the chain rule across path p .

$\rightarrow$ randomly shuffling $X_{p_(0)}, \ldots, X_{p_k}$ does not influence inference!

Regularization

Single CNets learned with LearnCNet are prone to overfitting, randomization in XCNets alleviate this issue





Learning curves of CNets and XCNets (negative log-likelihoods) on KddCup2k and EachMovie The latter overfits much later that the former.

Ensembles of XCNets

Ensembles of XCNets do not require to additionally diversify components and learning up to 500 components is still faster than learning 40 of other variants:

$\textbf{CNet}_{\text{bag}}$	bagging entCSN Rahman and Gogate 2016
$\textbf{CNet}_{\text{boost}}$	boosting entCSN Rahman and Gogate 2016
$\textbf{dCSN}^{k}$	bagging dCSN Di Mauro, Vergari, and Basile 2015; Di Mauro, Vergari, and Esposito 2015

Experiments: Single Model Comparison

dataset	entCNet	dCSN	XCNet	$\mathrm{dCSN}_{\mathrm{PoB}}$	$\mathrm{XCNet}_{\mathrm{PoB}}$
NLTCS	-6.06	**-6.03**	6.06±0.01	-6.09	-6.17±0.05
MSNBC	**-6.05**	**-6.05**	-6.09±0.02	-6.05	-6.18±0.03
KDDCup2k	-	**-2.18**	-2.19±0.01	-2.19	-2.21±0.01
Plants	**-13.25**	**-13.25**	-13.43±0.07	-14.89	-15.66±0.22
Audio	**-42.05**	-42.10	-42.66±0.14	-42.95	-44.02±0.22
Jester	-55.56	**-55.40**	-56.10±0.19	-56.23	-57.39±0.15
Netflix	**-58.71**	**-58.71**	-59.21±0.06	-60.20	-61.40±0.25
Accidents	-30.69	**-29.84**	-31.58±0.24	-36.24	-40.22±0.46
Retail	**-10.94**	-11.24	-11.44±0.09	-11.06	-11.19±0.04
Pumsb-star	-24.42	**-23.91**	-25.55±0.34	-32.11	-39.91±2.48
DNA	-87.59	**-87.31**	-87.67±0.00	-98.83	-99.84±0.05
Kosarek	**-11.04**	-11.20	-11.70±0.13	-11.38	-11.80±0.07
MSWeb	**-10.07**	-10.10	-10.47±0.10	-10.19	-10.43±0.07
Book	**-37.35**	-38.93	-42.36±0.28	-38.21	-39.47±0.33
EachMovie	-58.37	**-58.06**	-60.71±0.89	-59.70	-62.58±0.38
WebKB	-162.17	**-161.92**	-167.45±1.59	-168.7	-174.78±0.81
Reuters-52	**-88.55**	-88.65	-99.52±1.93	-90.51	-100.25±0.57
20NewsG	-	**-161.72**	-172.6±1.40	-162.25	-167.39±0.74
BBC	-263.08	**-261.79**	**-261.79**±0.00	-264.56	-274.83±1.15
Ad	-16.92	**-16.34**	-18.70±1.44	-36.44	-36.94±1.41

Table 1. Average test log-likelihoods for entCNet, dCSN, XCNet and their PoB variants $\mathrm{dCSN}_{\mathrm{PoB}}$ and $\mathrm{XCNet}_{\mathrm{PoB}}$. For randomized models, mean and standard deviation over 10 runs are reported).

Experiments: Ensemble Model Comparison

dataset	$\text{CNet}^{40}_{\text{bag}}$	$\text{CNet}^{40}_{\text{boost}}$	dCSN^{40}	$\text{XCNet}^{40}_{\text{PoB}}$	XCNet^{40}	XCNet^{500}	ID-SPN	ACMN	WM
NLTCS	**-6.00**	-6.01	**-6.00**	-6.01	**-6.00**	**-5.99**	-6.02	-6.00	-6.02
MSNBC	-6.08	-6.15	**-6.05**	-6.11	-6.06	-6.06	**-6.04**	**-6.04**	**-6.04**
KDDCup2k	-2.14	-2.15	-2.15	**-2.13**	**-2.13**	**-2.13**	**-2.13**	-2.17	-2.16
Plants	-12.32	-12.67	-12.59	-13.09	**-11.99**	**-11.84**	-12.54	-12.80	-12.65
Audio	-40.09	-39.84	-40.19	-40.30	**-39.77**	**-39.39**	-39.79	-40.32	-40.50
Jester	-52.88	-52.82	-52.99	-53.64	**-52.65**	**-52.21**	-52.86	-53.31	-53.85
Netflix	-56.55	-56.44	-56.69	-57.64	**-56.38**	**-55.93**	-56.36	-57.22	-57.03
Accidents	-29.88	-29.45	**-29.27**	-36.92	-29.31	-29.10	-26.98	-27.11	**-26.32**
Retail	-10.84	**-10.81**	-11.17	-10.88	-10.93	-10.91	-10.85	-10.88	-10.87
Pumsb-star	-23.98	-23.46	-23.78	-32.91	**-23.44**	-23.31	-22.41	-23.55	**-21.72**
DNA	**-81.07**	-85.67	-85.95	-98.28	-84.96	-84.17	-81.21	**-80.03**	-80.65
Kosarek	-10.74	**-10.60**	-10.97	-10.91	-10.72	-10.66	**-10.60**	-10.84	-10.83
MSWeb	-9.77	-9.74	-9.93	-9.83	**-9.66**	**-9.62**	-9.73	-9.77	-9.70
Book	-35.55	**-34.46**	-37.38	-34.77	-36.35	-35.45	**-34.14**	-35.56	-36.41
EachMovie	-53.00	**-51.53**	-54.14	-51.66	-51.72	**-50.34**	-51.51	-55.80	-54.37
WebKB	-153.12	**-152.53**	-155.47	-155.83	-153.01	**-149.20**	-151.84	-159.13	-157.43
Reuters-52	-83.71	**-83.69**	-86.19	-85.16	-84.05	**-81.87**	-83.35	-90.23	-87.55
20NewsG	-156.09	-153.12	-156.46	**-152.21**	-153.89	**-151.02**	-151.47	-161.13	-158.95
BBC	**-237.42**	-247.01	-248.84	-251.31	-238.47	**-229.21**	-248.93	-257.10	-257.86
Ad	-15.28	-14.36	-15.55	-26.25	**-14.20**	**-14.00**	-19.05	-16.53	-18.35

Experiments: Learning Times

dataset	dCSN	XCNet	$\mathrm{dCSN}_{\mathrm{PoB}}$	$\mathrm{XCNet}_{\mathrm{PoB}}$	dCSN^{40}	$\mathrm{XCNet}^{40}_{\mathrm{PoB}}$	XCNet^{40}	XCNet^{500}	ID-SPN
NLTCS	0	0.2	0.1	0.01	10	0.2	0.01	3	310
MSNBC	12	0.3	0.7	0.01	499	13.1	13	155	46266
KDDCup2k	112	0.5	12.0	0.32	4126	21.2	16	247	32067
Plants	15	0.3	45.5	0.22	325	1.0	6	77	18833
Audio	58	0.3	74.8	0.48	980	0.8	6	136	21009
Jester	50	0.2	95.6	0.26	989	0.3	4	83	10412
Netflix	75	0.2	2.8	0.02	1546	0.4	9	118	30294
Accidents	54	0.2	153.7	0.04	996	0.7	11	138	15472
Retail	263	0.8	5.8	0.01	3780	3.2	13	164	4041
Pumsb-star	118	0.6	26.2	0.02	2260	0.8	23	290	20952
DNA	30	0.1	4.4	0.01	224	0.06	3	40	3040
Kosarek	588	2.4	41.2	0.01	10033	10.8	43	524	17799
MSWeb	1215	7.2	7.4	0.01	17123	13.2	129	1592	19682
Book	9235	9.7	113.0	0.04	155634	1.9	316	3476	61248
EachMovie	1297	7.1	4.7	0.01	16962	1.1	127	2601	118782
WebKB	4997	11.0	238.0	0.03	18875	0.9	190	2237	45451
Reuters-52	9947	39.3	24.3	0.05	65498	2.7	414	8423	70863
20NewsG	16866	51.3	40.7	0.01	153908	4.4	506	9883	163256
BBC	21381	8.4	7.3	0.02	69572	0.4	256	4251	61471
Ad	5212	116.5	134.0	0.08	75694	4.2	2403	30538	87522

Table 3. Times (in seconds) taken to learn the best models on each dataset for dCSN, XCNet, $\mathrm{dCSN}_{\mathrm{PoB}}$, $\mathrm{XCNet}_{\mathrm{PoB}}$, their ensembles and ID-SPN

Conclusions

References

- Di Mauro, Nicola, Antonio Vergari, and Teresa M.A. Basile (2015). “Learning Bayesian Random Cutset Forests”. In: *Proceedings of ISMIS*. Springer, pp. 122–132.
- Di Mauro, Nicola, Antonio Vergari, and Floriana Esposito (2015). “Learning Accurate Cutset Networks by Exploiting Decomposability”. In: *Proceedings of AIXIA*. Springer, pp. 221–232.
- Meilă, Marina and Michael I. Jordan (2000). “Learning with mixtures of trees”. In: *Journal of Machine Learning Research* 1, pp. 1–48.
- Rahman, Tahrima and Vibhav Gogate (2016). “Learning Ensembles of Cutset Networks”. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI’16. Phoenix, Arizona: AAAI Press, pp. 3301–3307. URL: `http://dl.acm.org/citation.cfm?id=3016100.3016365`.
- Rahman, Tahrima, Prasanna Kothalkar, and Vibhav Gogate (2014). “Cutset Networks: A Simple, Tractable, and Scalable Approach for Improving the Accuracy of Chow-Liu Trees”. In: *Machine Learning and Knowledge Discovery in Databases*. Vol. 8725. LNCS. Springer, pp. 630–645.
- Roth, Dan (1996). “On the hardness of approximate reasoning”. In: *Artificial Intelligence* 82.1–2, pp. 273–302.

Discuss