Probabilistic graphical models: a tensorial perspective

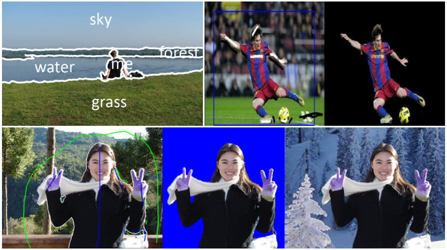
Dmitry Vetrov[1,2]
Alexander Novikov[1] Anton Rodomanov[2] Anton Osokin[3]

[1]Skolkovo Institute of Science and Technology, Russia [2]Higher School of Economics, Russia
[3]SIERRA, INRIA, France

Bayesian methods research group (http://bayesgroup.ru)

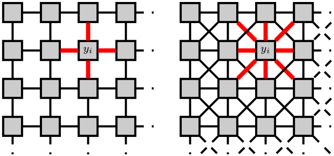
MMPR, September 2015

Motivational example: image segmentation



- **Task**: assign a label y_i to each pixel of an $M \times N$ image.
- Let $P(\mathbf{y})$ be the joint probability of labelling $\mathbf{y}$.
- Two extreme cases:
  - **No assumptions about independence**:
    - $O(K^{MN})$ parameters (K = total number of labels)
    - represents every distribution
    - intractable in general
  - **Everything is independent**: $P(\mathbf{y}) = p_1(y_1) \dots p_{MN}(y_{MN})$
    - $O(MNK)$ parameters
    - represents only a small class of distributions
    - tractable

- Provide a convenient way to define probabilistic models using graphs.
- Two types: directed graphical models and Markov random fields.
- We will consider only (discrete) Markov random fields.
- The edges represent dependencies between the variables.
- E.g., for image segmentation:

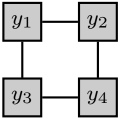


A variable y_i is independent of the rest given its immediate neighbours.

- The model:

$$P(\mathbf{y}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \Psi_c(\mathbf{y}_c),$$

  - Z: normalisation constant
  - $\mathcal{C}$: set of all (maximal) cliques in the graph
  - Ψ_c: non-negative functions which are called factors
- Example:



$$P(y_1, y_2, y_3, y_4) = \frac{1}{Z} \Psi_1(y_1)\Psi_2(y_2)\Psi_3(y_3)\Psi_4(y_4)$$
$$\times \Psi_{12}(y_1, y_2)\Psi_{24}(y_2, y_4)\Psi_{34}(y_3, y_4)\Psi_{13}(y_1, y_3)$$

The factors Ψ_{ij} measure 'compatibility' between variables y_i and y_j.

Main problems of interest

Probabilistic model:

$$P(\mathbf{y}) = \frac{1}{Z}\prod_{c\in\mathcal{C}}\Psi_c(\mathbf{y}_c) = \frac{1}{Z}\exp(-E(\mathbf{y})),$$

where E is the energy function:

$$E(\mathbf{y}) = \sum_{c\in\mathcal{C}}\Theta_c(\mathbf{y}_c), \qquad \Theta_c(\mathbf{y}_c) = -\ln\Psi_c(\mathbf{y}_c)$$

- Maximum a posteriori (MAP) inference:

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}} P(\mathbf{y}) = \operatorname*{argmin}_{\mathbf{y}} E(\mathbf{y})$$

- Estimation of the normalisation constant:

$$Z = \sum_{\mathbf{y}} P(\mathbf{y})$$

- Estimation of the marginal distributions:

$$P(y_i) = \sum_{\mathbf{y}\setminus y_i} P(\mathbf{y})$$

Tensorial perspective

- Energy and unnormalised probability are tensors:

$$\left.\begin{aligned} \mathbf{E}(y_1, \ldots, y_n) &= \sum_{c=1}^{m} \boldsymbol{\Theta}_c(\mathbf{y}_c), \\ \widehat{\mathbf{P}}(y_1, \ldots, y_n) &= \prod_{c=1}^{m} \boldsymbol{\Psi}_c(\mathbf{y}_c), \end{aligned}\right\} \quad \text{tensors (multidimensional arrays)}$$

  where $y_i \in \{1, \ldots, d\}$.
- In this language:
  - MAP-inference $\iff$ minimal element in $\boldsymbol{E}$
  - Normalisation constant $\iff$ sum of all the elements of $\widehat{\boldsymbol{P}}$

- TT-format for a tensor $\boldsymbol{A}$:

$$A(y_1, \ldots, y_n) = \underbrace{G_1[y_1]}_{1\times r_1}\underbrace{G_2[y_2]}_{r_1\times r_2}\ldots\underbrace{G_n[y_n]}_{r_{n-1}\times 1}$$

- Terminology:
  - G_i: TT-cores
  - r_i: TT-ranks
  - $r = \max r_i$: maximal TT-rank
- TT-format uses $O(ndr^2)$ memory to store $O(d^n)$ elements.
- Efficient only if the ranks are small.

TT-format: efficient operations and advantages

Operation	Output rank
$\mathbf{C} = \mathbf{A} + \mathbf{B}$	$r(\mathbf{A}) + r(\mathbf{B})$
$\mathbf{C} = \mathbf{A} \odot \mathbf{B}$	$r(\mathbf{A}) r(\mathbf{B})$
$\text{sum}\, \mathbf{A}$	–
$\min \mathbf{A}$	–

MAP-inference	$\Longleftrightarrow$	minimal element in $\boldsymbol{E}$
Normalisation constant	$\Longleftrightarrow$	sum of all elements of $\widehat{\boldsymbol{P}}$

Both operations are provided by the TT-format.

Let's convert $\boldsymbol{E}$ and $\widehat{\boldsymbol{P}}$ to the TT-format.

Finding a TT-representation of an MRF

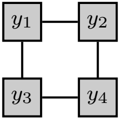
- TT-SVD (Oseledets, 2011): exact algorithm but only for small tensors
  No, MRF tensor is too big.

- AMEn-cross (Oseledets & Tyrtyshnikov, 2010): approximate algorithm; uses only a small fraction of the tensor's elements
  Possible, but there is a better way!

Converting the energy to the TT-format

$$\mathbf{E}(\mathbf{y}) = \sum_{c=1}^{m} \mathbf{\Theta}_c(\mathbf{y}_c)$$

- Each $\mathbf{\Theta}_c(\mathbf{y}_c)$ depends only on part of the all variables and is usually of low dimensionality $\Rightarrow$ can be converted to the TT-format using TT-SVD.
- Use the summation operation to build the TT-representation for $\mathbf{E}$.
- To do this, we need to add inessential variables $\mathbf{y} \setminus \mathbf{y}_c$ to every potential: $\mathbf{\Theta}_c(\mathbf{y}) \equiv \mathbf{\Theta}_c(\mathbf{y}_c)$.
- The same for the probability tensor, but use the Hadamard product.



- Let $\mathbf{y} = (y_1, y_2, y_3, y_4, y_5)$, $\mathbf{y}_c = (y_1, y_2, y_4)$.
- We already have the TT-format for $\boldsymbol{\Theta}_c(\mathbf{y}_c)$:

$$\boldsymbol{\Theta}_c(y_1, y_2, y_4) = G_1[y_1]G_2[y_2]G_4[y_4].$$

- To introduce y_3 and y_5, define the missing cores as identity matrices:

$$\boldsymbol{\Theta}_c(y_1, y_2, {\color{red}y_3}, y_4, {\color{red}y_5}) = G_1[y_1]G_2[y_2] \underbrace{{\color{red}I}}_{{\color{red}\equiv G_3[y_3]}} G_4[y_4] \underbrace{{\color{red}I}}_{{\color{red}\equiv G_5[y_5]}}.$$

- The maximal TT-rank does not increase!

The resulting algorithm

1. Compute the TT-decomposition for each individual potential $\boldsymbol{\Theta}_c(\mathbf{y}_c)$.
2. Add the inessential variables: $\boldsymbol{\Theta}_c(\mathbf{y}_c) \Rightarrow \boldsymbol{\Theta}_c(\mathbf{y})$.
3. Use the TT-summation to build $\mathbf{E}(\mathbf{y})$: $\mathbf{E}(\mathbf{y}) = \sum_{c=1}^{m} \boldsymbol{\Theta}_c(\mathbf{y})$.

Theorem

The maximal TT-rank of the tensor $\mathbf{E}$ *is polynomially bounded:*

$$r(\mathbf{E}) \leq d^{\frac{p}{2}} m,$$

where

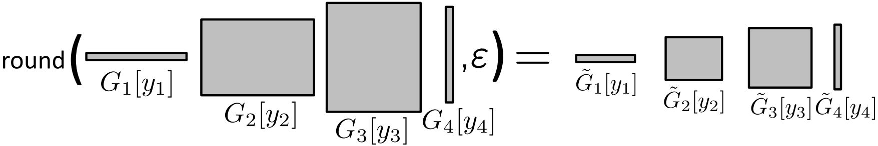
- d = *number of values that each variable can take;*
- m = *total number of potentials;*
- p = *maximal order of a potential (i.e. the maximal* $|\mathbf{y}_c|$*).*

Consider $p = 2$. Then $r(\mathbf{E}) \leq dm$ (linear dependence on m).

TT-rounding

TT-rounding procedure: $\tilde{\mathbf{A}} = \mathrm{round}(\mathbf{A}, \varepsilon)$:

1. reduces TT-ranks
2. tensors are close (ε = accuracy)

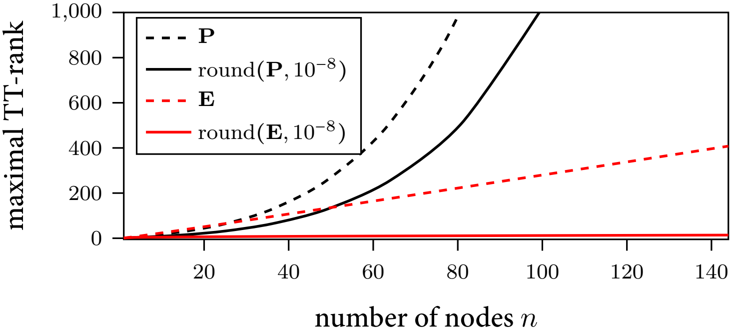


The TT-format for the probability

- We could find the TT-representation of $\widehat{\mathbf{P}}$ analogously:

$$\widehat{\mathbf{P}} = \bigodot_{c=1}^{m} \mathbf{\Psi}_c.$$

- However, the TT-ranks of $\widehat{\mathbf{P}}$ are exponential:



- We need to compute Z without explicitly building the TT for $\widehat{\mathbf{P}}$.

Normalisation constant estimation

- Kronecker product property: $ab = a \otimes b, \quad a, b \in \mathbb{R}$.
- Mixed product property: $AC \otimes BD = (A \otimes B)(C \otimes D)$.
- Then

$$\begin{aligned}
\widehat{\mathbf{P}}(\mathbf{y}) &= \prod_{c=1}^{m} \boldsymbol{\Psi}_c(\mathbf{y}) \\
&= \bigotimes_{c=1}^{m} \boldsymbol{\Psi}_c(\mathbf{y}) = \bigotimes_{c=1}^{m} \left(G_1^c[y_1] \cdots G_n^c[y_n]\right) \\
&= \left(G_1^1[y_1] \otimes \cdots \otimes G_1^m[y_1]\right) \cdots \left(G_n^1[y_n] \otimes \cdots \otimes G_n^m[y_n]\right).
\end{aligned}$$

- Denote $A_i[y_i] = G_i^1[y_i] \otimes \cdots \otimes G_i^m[y_i]$ (this is a huge matrix).
- Then

$$\begin{aligned}
{\color{red}Z} &= \sum_{\mathbf{y}} \widehat{\mathbf{P}}(\mathbf{y}) = \sum_{y_1, \ldots, y_n} A_1[y_1] \ldots A_n[y_n] \\
&= \underbrace{\left(\sum_{y_1} A_1[y_1]\right)}_{B_1} \ldots \underbrace{\left(\sum_{y_n} A_n[y_n]\right)}_{B_n} {\color{red}= B_1 \cdots B_n}.
\end{aligned}$$

The algorithm

- We have obtained the following expression:

$$Z = B_1 \dots B_n,$$

- Each matrix B_i is huge but can be exactly represented in the TT-format.
- The algorithm:
  1. $\mathbf{f}_1 := B_1$
  2. $\mathbf{f}_2 := \mathrm{round}(\mathbf{f}_1 B_2, \varepsilon)$
  3. $\mathbf{f}_3 := \mathrm{round}(\mathbf{f}_2 B_3, \varepsilon)$
  4. $\dots$
  5. $\mathbf{f}_n := \mathrm{round}(\mathbf{f}_{n-1} B_n, \varepsilon)$
  6. $\widetilde{Z} := \mathbf{f}_n;$
- This approach can be generalized to marginal distributions as well:

$$\widehat{\mathbf{P}}_i(y_i) = B_1 \dots B_{i-1} \, A_i[y_i] \, B_{i+1} \dots B_n,$$

The TT-method for the MAP-inference:

1. Convert the energy to the TT-format;
2. Find the minimal element in this tensor.

We compare this method with the popular TRW-S algorithm on several real-world image segmentation problems from the OpenGM database.

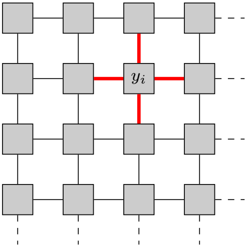
Problem	**Variables**	**Labels**	**TRW-S**	**TT**	**Time** (sec)
gm6	320	3	45.03	43.11	637
gm29	212	3	56.81	56.21	224
gm66	198	3	75.19	74.92	172
gm105	237	3	67.81	67.71	230
gm32	100	7	150.50	289.29	257
gm70	122	7	121.78	163.60	399
gm85	143	7	168.30	228.40	1 912
gm192	99	7	114.51	174.78	180

- Spin glass model:

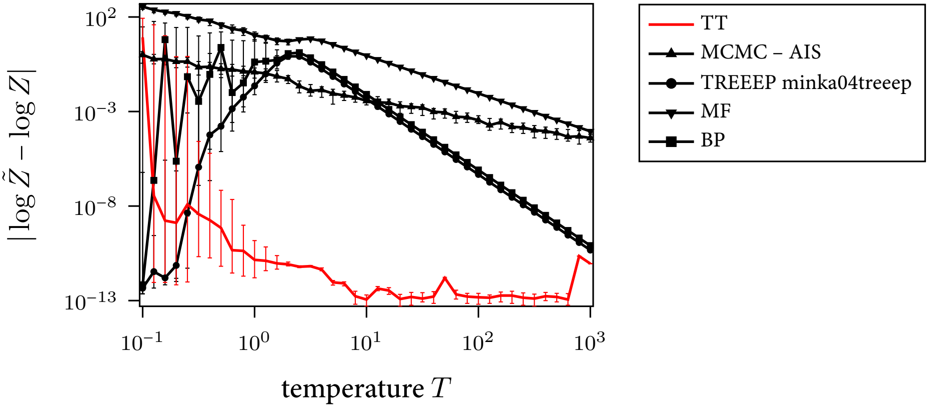
$$\widehat{\mathbf{P}}(\mathbf{y}) = \prod_{i=1}^{n} \exp\left(-\frac{1}{T} h_i y_i\right) \prod_{(i,j)\in\mathcal{E}} \exp\left(-\frac{1}{T} c_{ij} y_i y_j\right),$$

where $y_i \in \{-1, 1\}$.

- Terminology:
  - T: temperature
  - h_i: unary coefficients
  - c_{ij}: pairwise coefficients

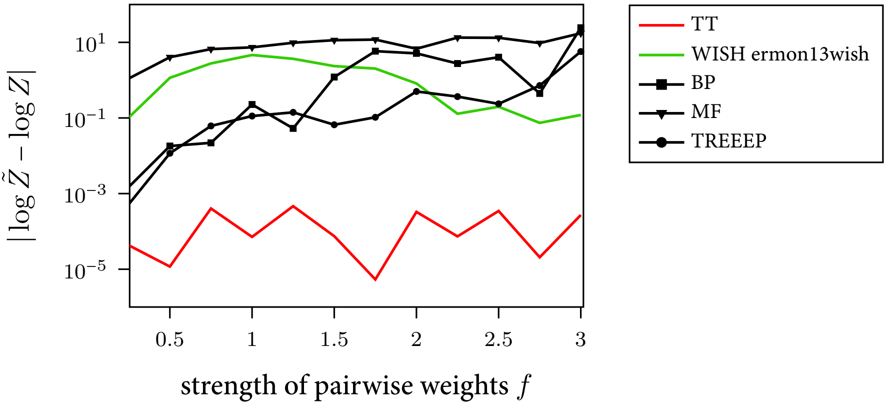


- Compare against methods from the LibDAI library ([?]).

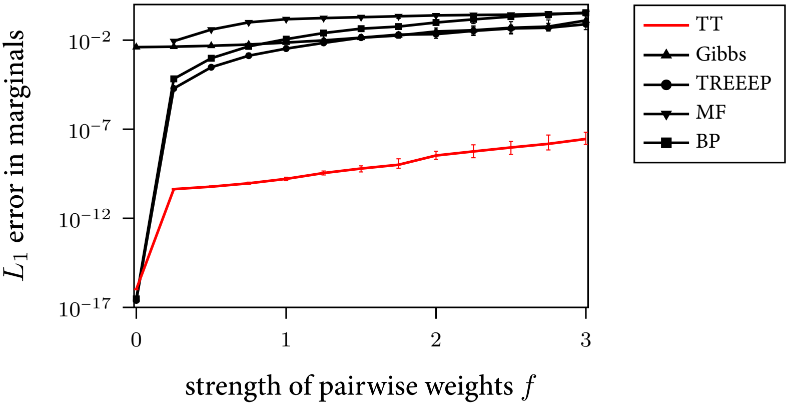


Comparison on the Ising model (all pairwise weights are equal $c_{ij} = 1$).

Experiments: WISH



Comparison on the data from the WISH paper, $T = 1$, $c_{ij} \sim U[-f, f]$.



Spin glass models, $T = 1$, $c_{ij} \sim U[-f, f]$.

Conclusions

- TT-format is very effective for the energy tensor. We have a good method for finding its TT-representation.
- However, TT-format is not suitable for the probability tensor.
- We have proposed an algorithm which estimates the normalisation constant without building the probability tensor.
- This algorithm is much more accurate than other state-of-the-art methods.