《机器学习：从入门到入魔》

Machine Learning: From Zero to Hero

第五讲：概率类机器学习

Lecture 5: Probabilistic Machine Learning

薛延波 *CSL BOSS* 直聘

2019-07-25

目录

Probabilistic machine learning (PML)

Role of uncertainties in ML

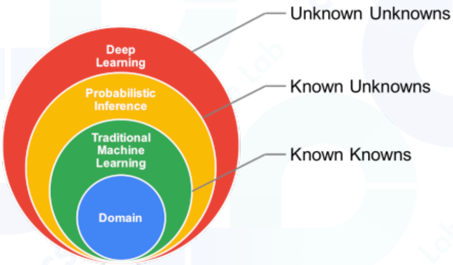


图: 工作原理和预测

- 不确定性的来源：数据 → 参数 → 模型结构
  - 数据的噪声：如图片的分辨度
  - 数据的量：如连续型的数据量为无限量
  - 数据的模糊性：如“Michael Jordan”既可以是体育明星，又可以是机器学习专家
  - 模型的复杂度：如参数太少 → 建模力太弱，参数太多 → 过拟合
- 概率与分布：一致的框架来“理解”和“操作”不确定性
- 模型的不确定性和数据的不确定性：都可以用概率与分布的概念来建模 → 确定性机器学习 vs 概率性机器学习
- 概率的两种解释：频率派（数据概率/极大似然估计）vs 贝叶斯派（主观概率/置信概率）

Math of PML in one page

- θ: param
- D: data
- m: model
- $P(D|\theta, m)$: LL of θ in m
- $P(\theta|m)$: prior of θ
- $P(\theta|D, m)$: posterior of θ
- $P(D|m)$: marginal LL/model evidence

Two rules [Mur12]

- Sum rule: $P(x) = \sum_{y \in \mathcal{Y}} P(x, y)$
- Product rule: $P(x, y) = P(x)P(y|x)$

Three task

- Learning: prior $\rightarrow$ posterior

$$P(\theta|D, m) = \frac{P(D|\theta, m)P(\theta|m)}{P(D|m)} \quad (1)$$

- Prediction: marginalization on θ

$$P(D_{test}|D, m) = \int P(D_{test}|\theta, D, m)P(\theta|D, m)\,d\theta \quad (2)$$

- Model comparison: which m is better

$$P(m|D) = \frac{P(D|m)P(m)}{P(D)} \quad (3)$$

$$P(D|m) = \int P(D|\theta, m)P(\theta|m)\,d\theta \quad (4)$$

Applications of PML

Areas of application [Gha15]

- probabilistic programming[vdMPYW18]: expressing probabilistic models as computer programs
- Bayesian optimization[SLA12]: (unknown) functions global optimization
- probabilistic data compression: compression based on Shannon's theory
- automatic statistician: discovering interpretable models from data
- hierarchical modelling: learning many related models, param of each model is sampled from the same distribution (prior), the final model is the integration of posterior distribution $P(Y|\theta) \leftarrow P(\theta|\mu) \leftarrow \cdots$, posterior $P(\theta, \mu, \cdots | Y)$

Energy-based models (EBMs)

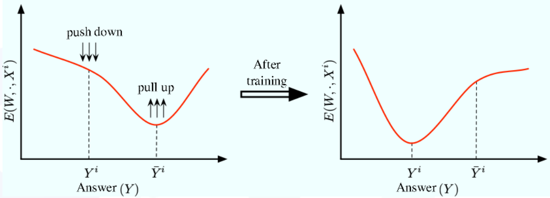
What are EBMs?

- Capture dependencies of variables by associating a scalar energy to each configuration [LeC06].

$$E(X, Y; \theta),$$

  where X: input, Y: label, and θ: parameters.
- **Overall goal**:

- *Desirable* configurations have low energies.
- *Undesirable* configurations have high energies.



- Normalization is **not** a must.
- Latent variables H can be introduced to compose much powerful model. Energy function $E(X, Y, H; \theta)$.
  - More expressive power
  - Both inference and learning are hard

What do EBMs do?

- Two tasks:
  - **Inference**: Clamp some variables (X), find configuration of remaining variables (Y) to minimize the energy.

$$Y^* = \arg\min_{Y \in \mathcal{Y}} E(X, Y; \theta) \tag{5}$$

  - **Learning/Training**: *Tune* the (parameters of) energy function to make sure the energies for *known* visibles are lower than *unknown* visibles.

$$\theta^* = \min_{\theta \in \theta} \mathcal{L}(E(X, Y; \theta)), \tag{6}$$

where $\mathcal{L}(E(X, Y; \theta))$ is a loss function parameterized by θ that measures the quality of the energy function.

Applications of EBMs [LeC06]

- **Prediction/Classification/Decision-making**: find the Y that is most compatible with X, e.g., robot
- **Ranking**: rank all the Ys based on their compability with X, e.g. probabilistic search engine
- **Detection**: reduce the energy as Y becomes more and more compatible with X, e.g., face detection from video frames
- **Conditional density estimation**: generate conditional probability distribution over $\mathcal{Y}$ given X, e.g., subsystem fed to another system
- Can you think of other applications?

EBMs provide unified framework

- **Non-probabilistic models**: shaping the energy function so that the **overall goal** is satisfied
  - supported vector Markov model (SVMM) [ATH03]: energy function is a linear combination of feature function, i.e. $E(X, Y; \theta) = \theta f(X, Y)$
- **Probabilistic models**: regularizing energies to satisfy probability constraints (sum to one), i.e., satisfying the **overall goal** without violating the probability constraints.
  - Markov random fields (MRFs), conditional random fields (CRFs), Boltzmann machines (BMs), Energy-based generative adversarial networks (EBGANs) [JML16],

Components of EBMs

1. **Architecture**: the form of energy function $E(\cdot)$
2. **Inference algorithm**: brute force (low cardinality, easy), approximate inference (high cardinality, hard, estimating the partition function) $\rightarrow$ Eqn. (5)
3. **Loss function**: the measure to assess the quality of energy function $\rightarrow \mathcal{L}(X, Y; \boldsymbol{\theta})$ for X and Y in training data.
4. **Optimization method**: the ways to find the optimal solutions of Eqn. (6).
   - (sub)gradient-based methods: first-order derivative of objective function
   - Newton's method: second-order, mostly infeasible, Quasi-Newton method (BFGS)
   - will be covered in *ML Best Practices*

Loss functions

- Energy loss: $E(X^i, Y^i; \boldsymbol{\theta})$ or MSE
  - simplest and straightforward
  - collapsed solutions on desired energies, other energies will be neglected
- Generalized perception loss: $E(X^i, Y^i; \boldsymbol{\theta}) - \min_{Y \in \mathcal{Y}} E(X^i, Y; \boldsymbol{\theta})$
  - push down desired energies, pull up undesired energies
  - not efficient in creating energy gaps $\rightarrow$ flat energy landscape
  - lack of margin, may have stability problems
- Generalized margin loss: $Q(E(X^i, Y^i; \boldsymbol{\theta}), E(X^i, \bar{Y}^i; \boldsymbol{\theta}))$
  - hinge loss: $\max(.,.)$, log loss: $\log(1 + \exp(.,.))$
  - create energy gaps
  - tend to push $E(X^i, Y^i; \boldsymbol{\theta})$ and $E(X^i, \bar{Y}^i; \boldsymbol{\theta})$ as far as possible $\leftarrow$ regularization as a rescue

Y^i: correct label, $\bar{Y}^i$: incorrect label

Loss functions (cont'd)

- **Negative log-likelihood (NLL) loss:** $-\log \prod_i p(Y^i|X^i;\boldsymbol{\theta})$
  - add *probabilistic* flavor to EBMs, apply maximum conditional probability principle
  - under Boltzmann distribution, NLL reduces to two parts: *clamped* model (push down energies) and *contrastive* model (pull up energies)
  - for zero temperature, NLL $\rightarrow$ generalized perception loss, for binary $\mathcal{Y}$, NLL $\rightarrow$ log loss.
  - known as *multi-class cross-entropy loss* in neural networks
  - very popular, used extensively
  - contrastive term not easy to calculate

Probabilistic models as EBMs

Probabilistic models as special EBMs

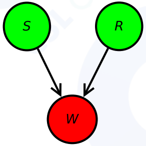
- EBMs:
  - Energies are unitless
  - Hard to combine different models
- Normalize energy function into probability (sum to 1) $\leftarrow$ a rich framework

$$
\begin{aligned}
\forall(X, Y) &\in \mathcal{X} \times \mathcal{Y} \\
E(X, Y; \theta) &\rightarrow p(X, Y; \theta), \\
\text{such that } p(X, Y; \theta) \geq 0 \quad &\text{and} \quad \sum_{X,Y} p(X, Y; \theta) = 1
\end{aligned}
$$

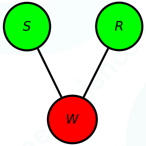
- Cons:
  - limited choices of energy functions
  - calculating contrastive term can be difficult

Probabilistic graphical models (PGMs)

- Bayesian (belief) network: directed PGM. $P(S,R,W) = P(S)P(R)P(W|S,R)$
  - explaining away (directed graph with V-shaped structure $S \to W$ and $R \to W$): S and R are independent; once W is given, S and R became dependent, knowing one can "explain away" the other
  - deep belief nets have many "V structures"
- Markov network: undirected PGM. $P(S,R,W) \propto \pi_1(S,W)\pi_2(R,W)$ with $\pi_i(\cdot)$ to be potential on cliques.
  - normalization is intractable



(a) Bayesian Network



(b) Markov Network

Exercise:

- S: sprinkler on
- R: raining
- W: grass wet
- $P(S) = 0.5$, $P(R) = 0.2$

- $P(W|S,R) = 0.99$
- $P(W|\bar{S},R) = 0.9$
- $P(W|S,\bar{R}) = 0.9$
- $P(W|\bar{S},\bar{R}) = 0$
- $P(S|W) =?$
- $P(S|W,R) =?$

Explaining away: solution

$$
\begin{aligned}
\color{red}{P(S|W)} &= \frac{P(W|S)P(S)}{P(W)} && (7)\\
&= \frac{(P(W|S,R)P(R) + P(W|S,\bar{R})P(\bar{R})) * P(S)}{P(W|S,R)P(S,R) + P(W|\bar{S},R)P(\bar{S},R) + P(W|S,\bar{R})P(S,\bar{R}) + P(W|\bar{S},\bar{R})P(\bar{S})P(\bar{R})} && (8)\\
&= \frac{(0.99 * 0.2 + 0.9 * 0.8) * 0.5}{0.99 * 0.5 * 0.2 + 0.9 * 0.5 * 0.2 + 0.9 * 0.5 * 0.8 + 0} = 0.8361 && (9)\\
\color{red}{P(S|W,R)} &= \frac{P(W,S,R)}{P(W,R)} = \frac{P(S)P(R)P(W|S,R)}{P(R)P(W|R)} && (10)\\
&= \frac{P(S)P(W|S,R)}{P(W|R)} = \frac{P(S)P(W|S,R)}{P(W|S,R)P(S) + P(W|\bar{S},R)P(\bar{S})} && (11)\\
&= \frac{0.5 * 0.99}{0.99 * 0.5 + 0.9 * (1 - 0.5)} = 0.5238 && (12)
\end{aligned}
$$

$$\color{red}{P(R|W)} = 0.3443, \color{red}{P(R|W,S)} = 0.2157$$

Boltzmann distribution

- Within exponential family, one common way:

$$p(X, Y, \theta) = \frac{e^{-\beta E(X, Y; \theta)}}{\sum_{x,y} e^{-\beta E(x,y;\theta)}}, \tag{13}$$

  where $Z(\theta) = \sum_{x,y} e^{-\beta E(x,y;\theta)}$ is the partition function, with $\beta = 1/T_0$ being the inverse temperature.
  The distribution is called *Boltzmann/Gibbs distribution.*
- Partition function can be intractable.

Boltzmann Machines

Boltzmann machines

- **Hopfield network** [Hop82] :
  - recurrent neural network (undirected) with threshold units (deterministic):

$$s_i = \begin{cases} +1 & \text{if } b_i + \sum_j s_j w_{ij} \geq \theta_i, \\ -1 & \text{otherwise.} \end{cases}$$

- **Boltzmann machines (BMs)** [DHS85]:
  - recurrent neural network with stochastic binary units:

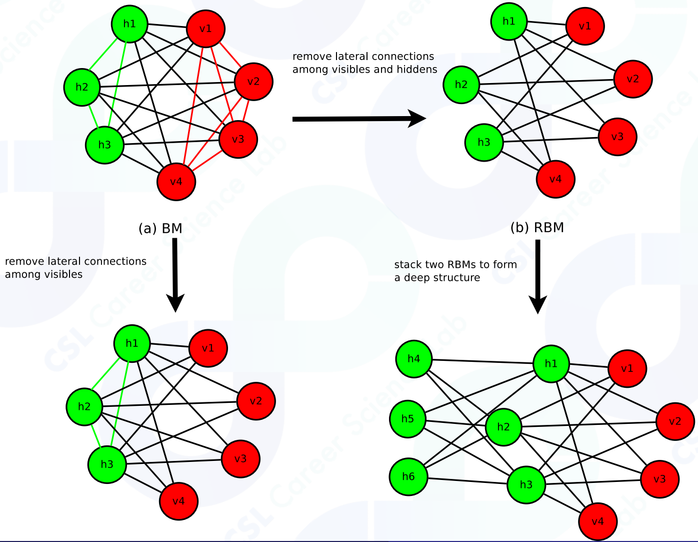
$$p(s_i = 1) = \mathrm{sigm}(b_i + \sum_j s_j w_{ij}),$$

    where $\mathrm{sigm}(\cdot)$ is the sigmoid function.
- Both have the same energy function:

$$E(\mathbf{s}) = \mathbf{b}^T\mathbf{s} + \mathbf{s}^T\mathbf{W}\mathbf{s},$$

  where $\mathbf{b}$ is the bias vector and $\mathbf{W}$ is the connection weight matrix (upper triangular).
- When temperature $T_0 = 0$, BMs become Hopfield networks.

A Family of Boltzmann Machines



Other distant relatives of BMs that we rarely visit:

- high-order BMs [Sej86]
- non-binary BMs (sigmoid $\rightarrow$ softmax) [WRZH05].

What's an RBM?

- Machine learning
  - find *patterns* or *statistical regularities* present in data
  - generative learning (vs. discriminative learning)
    - generative learning: learn joint distribution $p(\text{data}, \text{label})$
    - discriminative learning: learn conditional distribution $p(\text{label}|\text{data})$
- Energy function:

$$E(\mathbf{v}, \mathbf{h}) = \mathbf{b}^T\mathbf{v} + \mathbf{c}^T\mathbf{h} + \mathbf{h}^T\mathbf{W}\mathbf{v} \tag{14}$$

  - $\mathbf{v}$: visibles, $\mathbf{h}$: hiddens, $\boldsymbol{\theta} = \{\mathbf{b}, \mathbf{c}, \mathbf{W}\}$: model parameters
  - $\mathbf{v}$ and $\mathbf{h}$ follow a Boltzmann distribution:

$$p(\mathbf{v}, \mathbf{h}) = \underbrace{\frac{1}{Z}}_{\text{partition function}} \exp\{-E(\mathbf{v}, \mathbf{h})\} \tag{15}$$

- Learning RBM: *modifying* energy function so that training data *matches* the probability density defined by the energy function.

What's an RBM (cont'd)?

- Objective - minimize negative log-likelihood (NLL):

$$-\log p(\mathbf{v}) = \underbrace{-\log \sum_{\mathbf{h}} \exp\{-E(\mathbf{v},\mathbf{h})\}}_{\text{Free Energy}} + \log Z \tag{16}$$

- Cost function: average NLL of the training data $\mathcal{D}$ (L elements)

$$\mathcal{L}(\boldsymbol{\theta},\mathcal{D}) = -\frac{1}{L}\sum_{\mathbf{v}_i \in \mathcal{D}} \log p(\mathbf{v}_i;\boldsymbol{\theta}), \tag{17}$$

- Optimal parameter of model:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta},\mathcal{D}) \tag{18}$$

How to train an RBM?

overview

Two phases:

- positive phase: given the training data, sample the hiddens $\longrightarrow$ increase probability of training data.
- negative phase: both visibles and hiddens are sampled from the model $\longrightarrow$ decrease probability of samples generated by the model.

What makes training (and evaluating) RBM hard?

- sampling in negative phase is hard.
- Given the the trained parameter $\boldsymbol{\theta}$ [LS10],
  - evaluating the probability of $p(\mathbf{v})$ is hard
  - generating an evaluatable representation of the true distribution $p(\mathbf{v})$ is also hard
- better sampler $\longrightarrow$ better training $\longrightarrow$ better model
- QPU comes to the rescue

Conclusions

Conclusions

- EBMs use a scalar energy to represent variable dependencies
- Probabilistic models are special categories of EBMs, along with non-probabilistic models
- The components of EBMs include: architecture, inference algo., loss function, and optimization method
- Boltzmann machines are very popular probabilistic models
- Training of Boltzmann machines are hard due to contrastive term

课程大纲

- ~~机器学习简介~~
- ~~机器学习的数学基础~~
- ~~线性模型（线性回归、感知机、支持向量机）~~
- ~~神经网络模型~~
- ~~前期总结和回顾~~
- **概率类机器学习（本节）**
- 高级模型
- 工业实践

References I

- Yasemin Altun, Ioannis Tsochantaridis, and Thomas Hofmann, *Hidden markov support vector machines*, Proceedings of the 20th International Conference on Machine Learning (ICML-03) (2003).
- Ackley David, Geoffery Hinton, and Terrence Sejnowski, *A learning algorithm for boltzmann machines*, Cognitive science Elsevier **9** (1985), no. 1, 147–169.
- Zoubin Ghahramani, *Probabilistic machine learning and artificial intelligence*, Nature **521** (2015), no. 7553, 452.
- John Hopfield, *Neural networks and physical systems with emergent collective computational abilities*, Proceedings of the national academy of sciences (1982), 2554–2558.
- Zhao Junbo, Michael Mathieu, and Yann LeCun, *Energy-based generative adversarial network*, arXiv preprint arXiv 1609.03126 (2016).

References II

- Yann LeCun, *Predicting structured outputs*, ch. A Tutorial on Energy-Based Learning, MIT Press, 2006.
- Philip Long and Rocco Servedio, *Restricted boltzmann machines are hard to approximately evaluate or simulate*, Proceedings of the 27th International Conference on Machine Learning (ICML-10) (2010).
- K. Murphy, *Machine learning: a probabilistic perspective*, MIT Press, 2012.
- Terrence Sejnowski, *Higher order boltzmann machines*, AIP Conference Proceedings **151** (1986), no. 1.
- Jasper Snoek, Hugo Larochelle, and Ryan P. Adams, *Practical bayesian optimization of machine learning algorithms*, Advances in neural information processing systems] (2012), 2951–2959.

References III

Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood, *An introduction to probabilistic programming*, arXiv preprint arXiv:1809.10756 (2018).

M. Welling, M. Rosen-Zvi, and G. Hinton, *Exponential family harmoniums with an application to information retrieval*, Advances in Neural Information Processing Systems **17** (2005), no. 1481-1488.