
No Coding Farmer

Tao Hu

Department of Computer Science
Peking University
No.5 Yiheyuan Road Haidian District, Beijing, P.R.China
taohu@pku.edu.cn

Abstract

Some Miscellaneous Summary.

Contents

1	Expectation Maximization Introduction	3
1.1	EM Induction	3
1.2	EM convergence proof	3
1.3	Different Writing Style of EM Algorithm	3
2	EM applications	4
2.1	Gaussian Mix Model	4
2.2	Hidden Markov Model	4
2.3	Naive Bayesian	5
2.4	other papers	5
3	VAE	5
4	ADMM	5
5	Key steps you must know when building a DL Framework	8
5.1	Convolution	8
6	R-PCA	8
6.1	Solve RPCA by ADMM	8
6.2	Adaptive Penalty for ADMM	9
7	SFM	9
8	Reinforcement Learning	9

1 Expectation Maximization Introduction

1.1 EM Induction

$$L(\theta) = \sum_{i=1}^M \log p(X; \theta) = \sum_{i=1}^M \log \sum_z p(X, Z; \theta)$$

let θ_i be some distribution over z 's ($\sum_z \theta_i(z) = 1, \theta_i(z) \geq 0$)

$$\begin{aligned} & \sum_i \log p(X^{(i)}; \theta) \\ &= \sum_i \log \sum_{Z^{(i)}} \theta_i(Z^{(i)}) \frac{p(X^{(i)}, Z^{(i)}; \theta)}{\theta_i(Z^{(i)})} \\ &\geq \sum_i \sum_{Z^{(i)}} \theta_i(Z^{(i)}) \log \frac{p(X^{(i)}, Z^{(i)}; \theta)}{\theta_i(Z^{(i)})} \quad (f(x) = \log x \text{ is concave.}) \end{aligned}$$

$$\text{let } \frac{p(X^{(i)}, Z^{(i)}; \theta)}{\theta_i(Z^{(i)})} = C$$

the equality can be only reached when $\frac{p(X^{(i)}, Z^{(i)}; \theta)}{\theta_i(Z^{(i)})}$ is a constant.

we can get: $\sum_i \frac{p(X^{(i)}, Z^{(i)}; \theta)}{C} = 1$ namely: $\sum_i p(X^{(i)}, Z^{(i)}; \theta) = C$

further induction: $\theta_i(Z^{(i)}) = \frac{p(X^{(i)}, Z^{(i)}; \theta)}{\sum_i p(X^{(i)}, Z^{(i)}; \theta)} = p(Z^{(i)} | X^{(i)}; \theta)$

so the procedure of EM algorithm is:

Repeat Until Convergence:

- E-step: for each i , get $Q_i(Z^{(i)}) = p(Z^{(i)} | X^{(i)}; \theta)$
- M-step: $\theta := \argmax_{\theta} \sum_i \sum_{Z^{(i)}} Q_i(Z^{(i)}) \log \frac{p(X^{(i)}, Z^{(i)}; \theta)}{Q_i(Z^{(i)})}$

1.2 EM convergence proof

$$\text{let } l(\theta^{(t)}) = \sum_i \sum_{Z^{(i)}} Q_i^{(t)}(Z^{(i)}) \log \frac{p(X^{(i)}, Z^{(i)}; \theta)}{Q_i^{(t)}(Z^{(i)})}$$

then, we have the following inequality:

$$\begin{aligned} & l(\theta^{(t+1)}) \\ &\geq \sum_i \sum_{Z^{(i)}} Q_i^{(t)}(Z^{(i)}) \log \frac{p(X^{(i)}, Z^{(i)}; \theta^{(t+1)})}{Q_i^{(t)}(Z^{(i)})} \\ &\geq \sum_i \sum_{Z^{(i)}} Q_i^{(t)}(Z^{(i)}) \log \frac{p(X^{(i)}, Z^{(i)}; \theta^{(t)})}{Q_i^{(t)}(Z^{(i)})} \\ &\geq l(\theta^{(t)}) \end{aligned}$$

the first inequality is because: $l(\theta) \geq \sum_i \sum_{Z^{(i)}} Q_i^{(t)}(Z^{(i)}) \log \frac{p(X^{(i)}, Z^{(i)}; \theta)}{Q_i^{(t)}(Z^{(i)})} \forall \theta, Q_i$

the second inequality is because of the maximum of the M-step.

Hence, EM causes the likelihood to converge monotonically.

1.3 Different Writing Style of EM Algorithm

There are many writing style of EM algorithm. here I just mention the book <Statistics Learning Method> by LiHang who is very famous in China.

EM algorithm from LiHang(Li-version):

Algorithm 1 EM from LIHang

Require: observation X , hidden variable Z , joint distribution $P(X, Z|\theta)$, conditional distribution $P(Z|Y, \theta)$
while Not convergence **do**
 E-Step: let $\theta^{(i)}$ is the i -th estimate of θ ,
 $Q(\theta, \theta^{(i)}) = E_z[\log P(X, Z|\theta)|X, \theta^{(i)}] = \sum_Z \log P(X, Z|\theta)P(Z|X, \theta^{(i)})$
 M-step: $\theta^{(i+1)} = \operatorname{argmax}_{\theta} Q(\theta, \theta^{(i)})$
end while
output model parameter θ

it seems that Li-version is different from the above version. however, they are the same. because:

- the above version just consider every data, so that it include subscript i . however Li-version only consider one data.
- the above version can be transformed to Li-version.

$$\begin{aligned} & \sum_Z Q(Z) \log \frac{P(X, Z; \theta)}{Q(Z)} \\ &= \sum_Z P(Z|X; \theta^{(t)}) \log \frac{P(X, Z; \theta)}{P(Z|X; \theta^{(t)})} \\ &= \sum_Z P(Z|X; \theta^{(t)}) \log P(X, Z; \theta) - \sum_Z P(Z|X; \theta^{(t)}) \log P(Z|X; \theta^{(t)}) \end{aligned}$$

as the variable is θ , so $\sum_Z P(Z|X; \theta^{(t)}) \log P(Z|X; \theta^{(t)})$ can be removed.

- $Q(\theta, \theta^{(i)}) = \sum_Z \log P(X, Z|\theta)P(Z|X, \theta^{(i)})$ can be also written as $Q(\theta, \theta^{(i)}) = \sum_Z \log P(X, Z|\theta)P(Z, X, \theta^{(i)})$, because X is a observation.

2 EM applications

2.1 Gaussian Mix Model

GMM can be solved by EM. notice here we use the expectation of EM:

$$\begin{aligned} & Q(\theta, \theta^{(i)}) \\ &= E_{\gamma}[\log P(y, \gamma|\theta)|y, \theta^{(i)}] \\ &= E[\sum_{k=1}^K [n_k \log \alpha_k + \sum_{j=1}^N \gamma_{jk} [\log \frac{1}{\sqrt{2\pi}} - \log \sigma_k - \frac{1}{2\sigma_k^2} (y_j - \mu_k)^2]]] \\ &= \sum_{k=1}^K [(E\gamma_{jk}) \log \alpha_k + \sum_{j=1}^N (E\gamma_{jk}) [\log \frac{1}{\sqrt{2\pi}} - \log \sigma_k - \frac{1}{2\sigma_k^2} (y_j - \mu_k)^2]] \end{aligned}$$

here $(E\gamma_{jk})$ can be easily calculated.

$\hat{\mu}_k, \hat{\sigma}_k^2$ can be acquired by derivation.

$\hat{\alpha}_k$ can be acquired by the derivation on the Lagrangian ($\sum_i^K \alpha_k = 1$).

2.2 Hidden Markov Model

HMM Learning Method is also called Baum-Welch algorithm. the target is learning $\lambda = (A, B, \pi)$.

Q function is:

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_I \log P(O, I|\lambda) P(O, I|\bar{\lambda}) \\ P(O, I, \lambda) &= \pi_{i_1} b_{i_1}(o_1) a_{i_1 i_2} b_{i_2}(o_2) \dots a_{i_{T-1} i_T} b_{i_T}(o_T) \end{aligned}$$

so the Q function can also be written as:

$$Q(\lambda, \bar{\lambda}) = \sum_I \log \pi_{i1} P(O, I | \bar{\lambda}) + \sum_I (\sum_{t=1}^{T-1} \log a_{i,i+1}) P(O, I | \bar{\lambda}) + \sum_I (\sum_{t=1}^T \log b_{it}(o_t)) P(O, I | \bar{\lambda})$$

note here: I is not only one state. it includes state length from 1 to T, which all start from i_1

so we can solve the maximum of Q function by derivation on the Lagrangian polynomial (because exists these limitations: $\sum_{i=1}^N \pi_i = 1, \sum_{j=1}^N a_{ij} = 1, \sum_{i=1}^M b_i = 1$)

2.3 Naive Bayesian

2.4 other papers

We can use softmax to model transition probability, normal distribution to model emission probability.

it's a good example in Car that Knows Before You Do: Anticipating Maneuvers via Learning Temporal Driving Models, the AIO-HMM can be more complicated, which can be enriched by the graphic model by M.I Jordon.

3 VAE

here is a complete VAE tutorial [1]

$$\begin{aligned} \max \quad & \log P(x) \\ \text{lhs} = & \log \int P(x, z) dz \\ = & \log \int P(x/z) p(z) dz \\ = & \log \int \frac{P(x/z)}{q(z/x)} q(z/x) p(z) dz \\ = & \log E_{q(z/x)} \left[\frac{P(x/z)}{q(z/x)} p(z) \right] \\ \text{jensen's inequality, we can know: } & \geq E_{q(z/x)} [\log \frac{P(x/z)}{q(z/x)} p(z)] \\ = & E_{q(z/x)} [\log p(x/z)] + E_{q(z/x)} [\log \frac{p(z)}{q(z/x)}] \\ = & E_{q(z/x)} [\log p(x/z)] - E_{q(z/x)} [\log \frac{q(z/x)}{p(z)}] \\ = & E_{q(z/x)} [\log p(x/z)] - KL(q(z/x) || p(z)) \end{aligned}$$

4 ADMM

minimize $H(u) + G(v)$
subject to $Au + Bv = b$

$$\max_{\lambda} \min_{u,v} H(u) + G(v) + \langle \lambda, b - Au - Bv \rangle + \frac{\tau}{2} \|b - Au - Bv\|^2$$

Alternating Direction Method of Multipliers

$$\begin{aligned} u_{k+1} &= \arg \min_u H(u) + \langle \lambda_k, -Au \rangle + \frac{\tau}{2} \|b - Au - Bv_k\|^2 \\ v_{k+1} &= \arg \min_v G(v) + \langle \lambda_k, -Bv \rangle + \frac{\tau}{2} \|b - Au_{k+1} - Bv\|^2 \\ \lambda_{k+1} &= \lambda_k + \tau(b - Au_{k+1} - Bv_{k+1}) \end{aligned}$$

Distributed Problems

minimize $g(x) + \sum_i f_i(x)$
example: sparse least squares:
minimize $\mu \|x\|_1 + \frac{1}{2} \|Ax - b\|^2$

$$\text{minimize } \mu \|x\|_1 + \sum_i \frac{1}{2} \|A_i x - b_i\|^2$$

data stored on different servers

Transpose Reduction

$$\text{minimize } \frac{1}{2} \|Ax - b\|^2$$

$$x^* = (A^T A)^{-1} A^T b$$

distributed computation:

$$A^T b = \sum A_i^T b_i$$

$$A^T A = \sum A_i^T A_i$$

Unwrapped ADMM

$$\text{minimize } g(x) + f(Ax) = g(x) + \sum_i f_i(A_i x)$$

Example: SVM

$$\text{minimize } \frac{1}{2} \|x\|^2 + h(Ax)$$

A = data, h = hinge loss

Unwrapped form

$$\text{minimize } \frac{1}{2} \|x\|^2 + h(z)$$

subject to $z = Ax$

Transpose Reduction ADMM

scaled augmented Lagrangian:

$$\text{minimize } \frac{1}{2} \|x\|^2 + h(z) + \frac{\tau}{2} \|z - Ax - \lambda\|^2$$

ADMM:

$$x^{k+1} = \min_x \frac{1}{2} \|x\|^2 + \frac{\tau}{2} \|z^k - Ax + \lambda^k\|^2$$

$$z^{k+1} = \min_z h(z) + \frac{\tau}{2} \|z - Ax^{k+1} + \lambda^k\|^2$$

$$\lambda^{k+1} = \lambda^k + z^{k+1} - Ax^{k+1}$$

Minimization Steps

$$\text{minimize } l(a_3) + \frac{1}{2} \|z_2 - W_1 a_1\|^2 + \frac{1}{2} \|a_2 - \sigma(z_2)\|^2 + \frac{1}{2} \|z_3 - W_2 a_2\|^2 + \frac{1}{2} \|a_3 - \sigma(z_3)\|^2$$

Solve for weight: least squares(convex)

Solve for activations: least squares + ridge penalty(convex)

Solve for inputs: coordinate-minimization (non-convex but global)

Lagrange Multipliers

$$\text{minimize } l(a_3) + \frac{1}{2} \|z_2 - W_1 a_1\|^2 + \frac{1}{2} \|a_2 - \sigma(z_2)\|^2 + \langle \lambda_1, z_2 - W_1 a_1 \rangle + \langle \lambda_2, a_2 - \sigma(z_2) \rangle + \frac{1}{2} \|z_3 - W_2 a_2\|^2 + \frac{1}{2} \|a_3 - \sigma(z_3)\|^2 + \langle \lambda_3, z_3 - W_2 a_2 \rangle + \langle \lambda_4, a_3 - \sigma(z_3) \rangle$$

unstable because of non-linear constraints

Bregman Iteration

$$\text{minimize } l(a_3) + \langle \lambda, a_3 \rangle + \frac{1}{2} \|z_2 - W_1 a_1\|^2 + \frac{1}{2} \|a_2 - \sigma(z_2)\|^2 + \frac{1}{2} \|z_3 - W_2 a_2\|^2 + \frac{1}{2} \|a_3 - \sigma(z_3)\|^2$$

HOG feature dimension: 648

mid layer 1 num: 100

mid layer 2 num: 50

output layer: 1

Algorithm 2 ADMM_NN

Inputs:

data number: $n=10000$,
data dimension: $m=648$,
hidden layer 1 unit number: $a=100$
hidden layer 2 unit number: $b=50$
output layer unit number: 1
 a_0 m-n dimension,
 W_1 : a-m dimension
 z_1 : a-n dimension
 a_1 : a-n dimension
 W_2 : b-a dimension
 z_2 : b-n dimension
 a_2 : b-n dimension
 W_3 : 1-b dimension
 z_3 : 1-n dimension
labels: y 1-n dimension
 λ : 1-n dimension
activation function h is ReLu.

Initialize:

allocate $\{a_l\}_{l=1}^L, \{z_l\}_{l=1}^L$ with i.i.d Gaussian Distribution, and λ

Cache: a_0^\dagger

Warm Start:

for $i=1, \dots, 100$ **do**

for $l=1, 2, \dots, L-1$ **do**

$W_l \leftarrow z_l a_{l-1}^\dagger$

$a_l \leftarrow (\beta_{l+1} W_{l+1}^T W_{l+1} + \gamma_l I)^{-1} (\beta_{l+1} W_{l+1}^T z_{l+1} + \gamma_l h_l(z_l))$

$z_l \leftarrow \operatorname{argmin}_z \gamma_l \|a_l - h_l(z)\|^2 + \beta_l \|z - W_l a_{l-1}\|^2$

end for

$W_L \leftarrow z_L a_{L-1}^\dagger$

$z_L \leftarrow \operatorname{argmin}_z l(z, y) + \langle z, \lambda \rangle + \beta_L \|z - W_L a_{L-1}\|^2$

end for

Start ADMM:

while not converge **do**

for $l=1, 2, \dots, L-1$

do $W_l \leftarrow z_l a_{l-1}^\dagger$

$a_l \leftarrow (\beta_{l+1} W_{l+1}^T W_{l+1} + \gamma_l I)^{-1} (\beta_{l+1} W_{l+1}^T z_{l+1} + \gamma_l h_l(z_l))$

$z_l \leftarrow \operatorname{argmin}_z \gamma_l \|a_l - h_l(z)\|^2 + \beta_l \|z - W_l a_{l-1}\|^2$

end for

$W_L \leftarrow z_L a_{L-1}^\dagger$

$z_L \leftarrow \operatorname{argmin}_z l(z, y) + \langle z, \lambda \rangle + \beta_L \|z - W_L a_{L-1}\|^2$

$\lambda \leftarrow \lambda + \beta_L (z_L - W_L a_{L-1})$

end while

z_l argmin procedure:

$$z_l = \begin{cases} \max(\frac{a_l \gamma_l + W_l a_{l-1} \beta_l}{\gamma_l + \beta_l}, 0) & z \geq 0 \\ \min(W_l a_{l-1}, 0) & z \leq 0 \end{cases}$$

choose one minimizer z from two choices.

z_L argmin procedure:

when $y_i = 0$:

$$f(z) = \beta z^2 - (2\beta w_a - \lambda)z + \max(z, 0)$$

$$z^* = \max(\frac{2\beta w_a - \lambda - 1}{2\beta}, 0) \text{ or}$$

$$z^* = \min(\frac{2\beta w_a - \lambda}{2\beta}, 0)$$

choose one which make $f(z)$ smaller.

when $y_i = 1$:

$$f(z) = \beta z^2 - (2\beta w_a - \lambda)z + \max(1 - z, 0)$$

$$z^* = \max(\frac{2\beta w_a - \lambda}{2\beta}, 1) \text{ or}$$

$$z^* = \min(\frac{2\beta w_a - \lambda + 1}{2\beta}, 1)$$

choose one which make $f(z)$ smaller.

z_L argmin procedure(when l is a standard hinge loss):

$$\text{when } y_i = -1: f(z) = \max(1 + z) + \lambda z + \beta(z^2 - 2w_a z)$$

$$z^* = \min(\frac{2\beta w_a - \lambda}{2\beta}, -1) \text{ or}$$

$$z^* = \max(\frac{2\beta w_a - \lambda - 1}{2\beta}, -1) \text{ choose one which make } f(z) \text{ smaller.}$$

when $y_i = 1$:

$$f(z) = \max(1 - z, 0) + \lambda z + \beta(z^2 - 2w_a z)$$

$$z^* = \min(\frac{2\beta w_a - \lambda + 1}{2\beta}, 1) \text{ or}$$

$$z^* = \max(\frac{2\beta w_a - \lambda - 1}{2\beta}, 1)$$

choose one which make $f(z)$ smaller.

5 Key steps you must know when building a DL Framework

5.1 Convolution

6 R-PCA

RPCA problem:

$$\min_{A,E} \|A\|_* + \lambda \|E\|_1$$

S.t $D=A+E$

RPCA dual problem:

Augmented Lagrangian is :

$$A_t(A, E; \Lambda) = \min_{A,E} \Lambda(A, E; \Lambda)$$

$$= \min_{A,E} \|A\|_* + \Lambda \|E\|_1 + \langle \Lambda, D - A - E \rangle$$

$$= \min_A \|A\|_* - \langle \Lambda, A \rangle + \min_E \lambda \|E\|_1 - \langle \Lambda, E \rangle + \langle \Lambda, D \rangle$$

both of the sub-problem is conjugate function, according to the property of conjugate function :

$$A_t(A, E; \Lambda) = \langle \Lambda, D \rangle$$

$$\text{S.t } \|\Lambda\|_2 \leq 1, \|\Lambda\|_\infty \leq \lambda$$

so the dual problem is:

$$\max_{\Lambda} \langle \Lambda, D \rangle$$

$$\text{S.t } \|\Lambda\|_2 \leq 1, \|\Lambda\|_\infty \leq \lambda$$

6.1 Solve RPCA by ADMM

the ADMM sub-problem is:

A-sub-problem:

$$A_{k+1} = \operatorname{argmin}_A \|A\|_* + \frac{\beta}{2} \|D - A - E_k + \Lambda_k / \beta\|_F^2$$

E-sub-problem:

$$E_{k+1} = \operatorname{argmin}_E \lambda \|E\|_1 + \frac{\beta}{2} \|D - A_{k+1} - E + \Lambda_k / \beta\|_F^2$$

E-sub-problem has closed-form solution as follows:

$$E_{k+1} = S_{\lambda\beta^{-1}}(D - A_k + \Lambda_k/\beta).$$

$S_\epsilon = \text{sgn}(x)\max(|x| - \epsilon, 0)$, which is the same form as shrinkage.

A-sub-problem has a closed-form solution offered by Singular Value Thresholding(SVT): suppose that the SVD of $W = D - E_k + \Lambda_k/\beta_k$ is $W = U\Sigma V^T$, then the optimal solution is $A = US_{\beta^{-1}}(\Sigma)V^T$.

6.2 Adaptive Penalty for ADMM

Lin et al.[2] suggest updating the penalty parameter β as follows:

$$\beta_{k+1} = \min(\beta_{max}, \rho\beta_k)$$

where ρ_{max} is an upper bound of $\{\beta_k\}$. the value of ρ is defined as:

$$\rho = \begin{cases} \rho_0 & \text{if } \frac{\beta_k \max(\sqrt{\eta_A} \|x_{k+1} - x_k\|_2, \sqrt{\eta_B} \|y_{k+1} - y_k\|_2)}{\|c\|_2} < \epsilon_2 \\ 1 & \text{otherwise} \end{cases}$$

where η_A, η_B is linearized Taylor second-order factor.

7 SFM

<https://www.robots.ox.ac.uk/~vgg/hzbook/hzbook2/HZepipolar.pdf>

- F is a rank 2 homogeneous matrix with 7 degrees of freedom.
- **Point correspondence:** If \mathbf{x} and \mathbf{x}' are corresponding image points, then $\mathbf{x}'^T \mathbf{F} \mathbf{x} = 0$.
- **Epipolar lines:**
 - ◊ $\mathbf{l}' = \mathbf{F} \mathbf{x}$ is the epipolar line corresponding to \mathbf{x} .
 - ◊ $\mathbf{l} = \mathbf{F}^T \mathbf{x}'$ is the epipolar line corresponding to \mathbf{x}' .
- **Epipoles:**
 - ◊ $\mathbf{F} \mathbf{e} = \mathbf{0}$.
 - ◊ $\mathbf{F}^T \mathbf{e}' = \mathbf{0}$.
- **Computation from camera matrices \mathbf{P}, \mathbf{P}' :**
 - ◊ General cameras,
 $\mathbf{F} = [\mathbf{e}']_{\times} \mathbf{P}' \mathbf{P}^+$, where \mathbf{P}^+ is the pseudo-inverse of \mathbf{P} , and $\mathbf{e}' = \mathbf{P}' \mathbf{C}$, with $\mathbf{P} \mathbf{C} = \mathbf{0}$.
 - ◊ Canonical cameras, $\mathbf{P} = [\mathbf{I} \mid \mathbf{0}]$, $\mathbf{P}' = [\mathbf{M} \mid \mathbf{m}]$,
 $\mathbf{F} = [\mathbf{e}']_{\times} \mathbf{M} = \mathbf{M}^{-T} [\mathbf{e}]_{\times}$, where $\mathbf{e}' = \mathbf{m}$ and $\mathbf{e} = \mathbf{M}^{-1} \mathbf{m}$.
 - ◊ Cameras not at infinity $\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]$, $\mathbf{P}' = \mathbf{K}'[\mathbf{R} \mid \mathbf{t}]$,
 $\mathbf{F} = \mathbf{K}'^{-T} [\mathbf{t}]_{\times} \mathbf{R} \mathbf{K}^{-1} = [\mathbf{K}' \mathbf{t}]_{\times} \mathbf{K}' \mathbf{R} \mathbf{K}^{-1} = \mathbf{K}'^{-T} \mathbf{R} \mathbf{K}^T [\mathbf{K} \mathbf{R}^T \mathbf{t}]_{\times}$.

Figure 1: Summary of Fundamental matrix properties

8 Reinforcement Learning

Acknowledgments

References

- [1] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.
- [2] Zhouchen Lin, Risheng Liu, and Zhixun Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems* 24, pages 612–620. Curran Associates, Inc., 2011.