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.

Author Info: Taohu ,Peking University

Contents

1 Expectation Maximization Introduction

1.1 EM Induction

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

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

$\sum_i logp(X^{(i)};\theta)$
$= \sum_i log \sum_{Z^{(i)}} \theta_i(Z^{(i)}) \frac{p(X^{(i)},Z^{(i)};\theta)}{\theta_i(Z^{(i)})}$
$\geq \sum_i \sum_{Z^{(i)}} \theta_i(Z^i) log \frac{P(X^{(i)}),Z^{(i)};\theta)}{\theta_i(Z^{(i)})} (f(x) = logx \quad is \quad concave.)$

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: $\theta := argmax_\theta \sum_i \sum_{Z^i} i(Z^{(i)}) log \frac{p(X^{(i)},Z^{(i)}),\theta)}{Q_i(Z^{(i)})}$

1.2 EM convergence proof

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:
$l(\theta^{(t+1)})$
$\geq \sum_i \sum_{Z^{(i)}} Q_i^{(t)}(Z^{(i)}) log \frac{p(X^{(i)},Z^{(i)},\theta^{(t+1)})}{Q_i^{(t)}(Z^{(i)})}$
$\geq \sum_i \sum_{Z^{(i)}} Q_i^{(t)}(Z^{(i)}) log \frac{p(X^{(i)},Z^{(i)},\theta^{(t)})}{Q_i^{(t)}(Z^{(i)})}$
$\geq l(\theta^{(t)})$

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,

$Q(\theta,\theta^{(i)}) = E_z[logP(X,Z|\theta)|X,\theta^{(i)}] = \sum_Z logP(X,Z|\theta)P(Z|X,\theta^{(i)})$

M-step: $\theta^{(i+1)} = argmax_\theta Q(\theta,\theta^{(i)})$

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.

  $\sum_Z Q(Z)log\frac{P(X,Z:\theta)}{Q(Z)}$
  $= \sum_Z P(Z|X;\theta^{(t)})log\frac{p(X,Z;\theta)}{p(Z|X;\theta^{(t)})}$
  $= \sum_Z P(Z|X;\theta^t)logP(X,Z;\theta) - \sum_Z P(Z|X;\theta^{(t)})logP(Z|X;\theta^{(t)})$

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

- $Q(\theta,\theta^{(i)}) = \sum_Z logP(X,Z|\theta)P(Z|X,\theta^{(i)})$ can be also written as $Q(\theta,\theta^{(i)}) = \sum_Z logP(X,Z|\theta)P(Z,X,\theta^{(i)})$,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:**

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

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:

$Q(\lambda,\bar{\lambda}) = \sum_I logP(O,I|\lambda)P(O,I|\bar{\lambda})$

$P(O,I,\lambda) = \pi_{i1}b_{i1}(o_1)a_{i1i2}b_{i2}(o_2)...a_{i_{T-1}i_T}b_{i_T}(o_T)$

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]

$$max \quad logP(x)$$
lhs = $log \int P(x, z) dz$
$= log \int P(x/z) p(z) dz$
$= \log \int \frac{P(x/z)}{q(z/x)} q(z/x) p(z) dz$
$= log E_{q(z/x)} [\frac{p(x/z)}{q(z/x)} p(z)]$
jenson's inequality,we can know: $\geq E_{q(z/x)} [log \frac{p(x/z)}{q(z/x)} p(z)]$
$= E_{q(z/x)} [log p(x/z)] + E_{q(z/x)} [log \frac{p(z)}{q(z/x)}]$
$= E_{q(z/x)} [log p(x/z)] - E_{q(z/x)} [log \frac{q(z/x)}{p(z)}]$
$= E_{q(z/x)} [log p(x/z)] - KL(q(z/x)||p(z))$

4 ADMM

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

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

Alternating Direction Method of Multipliers
$u_{k+1} = argmin_u H(u) + < \lambda_k, -Au > + \frac{\tau}{2} ||b - Au - Bv_k||^2$
$v_{k+1} = argmin_v G(v) + < \lambda_k, -Bv > + \frac{\tau}{2} ||b - Au_{k+1} - Bv||^2$
$\lambda_{k+1} = \lambda_k + \tau(b - Au_{k+1} - Bv_{k+1})$

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

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

data stored on different servers

Transpose Reduction
minimize $\frac{1}{2}||Ax-b||^2$
$x^* = (A^TA)^{-1}A^Tb$

distributed compution:
$A^Tb = \sum A_i^T b_i$
$A^TA = \sum A_i^T A_i$

Unwrapped ADMM
minimize $g(x) + f(Ax) = g(x) + \sum_i f_i(A_i x)$

Example: SVM
minimize $\frac{1}{2}||x||^2 + h(Ax)$
A = data,h = hinge loss

Unwrapped form
minimize $\frac{1}{2}||x||^2 + h(z)$
subject to z=Ax

Transpose Reduction ADMM
scaled augmented Lagrangian:
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
minimize $l(a_3) + \frac{1}{2}||z_2 - W_1a_1||^2 + \frac{1}{2}||a_2 - \sigma(z_2)||^2 + \frac{1}{2}||z_3 - W_2a_2||^2 + \frac{1}{2}||a_3 - \sigma(z_3)||^2$

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
minimize $l(a_3) + \frac{1}{2}||z_2 - W_1a_1||^2 + \frac{1}{2}||a_2 - \sigma(z_2)||^2 + < \lambda_1, z_2 - W_1a_1 > + < \lambda_2, a_2 - \sigma(z_2) > + \frac{1}{2}||z_3 - W_2a_2||^2 + \frac{1}{2}||a_3 - \sigma(z_3)||^2 + < \lambda_3, z_3 - W_2a_2 > + < \lambda_4, a_3 - \sigma(z_3) >$

unstable because of non-linear constraints

Bregman Iteration
minimize $l(a_3) + < \lambda, a_3 > + \frac{1}{2}||z_2 - W_1a_1||^2 + \frac{1}{2}||a_2 - \sigma(z_2)||^2 + \frac{1}{2}|||z_3 - W_2a_2||^2 + \frac{1}{2}||a_3 - \sigma(z_3)||^2$

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,..,100 **do**
for l=1,2,...,L-1 **do**
$W_l \leftarrow z_l a_{l-1}^{\dagger}$
$a_l \leftarrow (\beta_{l+1} W_{l+1}^T W_{l+1} + \gamma_l I)^{-1} (\beta_{l+1} W_{l+1}^T z_{l+1} + \gamma_l h_l(z_l))$
$z_l \leftarrow argmin_z \gamma_l ||a_l - h_l(z)||^2 + \beta_l ||z - W_l a_{l-1}||^2$

end for
$W_L \leftarrow z_L a_{L-1}^{\dagger}$
$z_L \leftarrow argmin_z l(z,y) + < z, \lambda > + \beta_L ||z - W_L a_{L-1}||^2$
end for
Start ADMM:
while not converge **do**
for l=1,2,...,L-1
do $W_l \leftarrow z_l a_{l-1}^{\dagger}$
$a_l \leftarrow (\beta_{l+1} W_{l+1}^T W_{l+1} + \gamma_l I)^{-1} (\beta_{l+1} W_{l+1}^T z_{l+1} + \gamma_l h_l(z_l))$
$z_l \leftarrow argmin_z \gamma_l ||a_l - h_l(z)||^2 + \beta_l ||z - W_l a_{l-1}||^2$
end for
$W_L \leftarrow z_L a_{L-1}^{\dagger}$
$z_L \leftarrow argmin_z l(z,y) + < z, \lambda > + \beta_L ||z - W_L a_{L-1}||^2$
$\lambda \leftarrow \lambda + \beta_L (z_L - W_L a_{L-1})$
end while

z_l argmin procedure:

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

choose one minimizer z from two choices.

z_L argmin procedure:

when $y_i = 0$:
$f(z) = \beta z^2 - (2\beta w\_a - \lambda) z + max(z, 0)$

$z^* = max(\frac{2\beta w\_a - \lambda - 1}{2\beta}, 0)$ or
$z^* = min(\frac{2\beta w\_a - \lambda}{2\beta}, 0)$
choose one which make f(z) smaller.

when $y_i = 1$:
$f(z) = \beta z^2 - (2\beta w\_a - \lambda)z + max(1 - z, 0)$
$z^* = max(\frac{2\beta w\_a - \lambda}{2\beta}, 1)$ 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):

when $y_i = -1$: $f(z) = max(1 + z) + \lambda z + \beta(z^2 - 2w\_az)$
$z^* = min(\frac{2\beta w\_a - \lambda}{2\beta}, -1)$ or
$z^* = max(\frac{2\beta w\_a - \lambda - 1}{2\beta}, -1)$ choose one which make f(z) smaller.

when $y_i = 1$:
$f(z) = max(1 - z, 0) + \lambda z + \beta(z^2 - 2w\_az)$
$z^* = min(\frac{2\beta w\_a - \lambda + 1}{2\beta}, 1)$ or
$z^* = max(\frac{2\beta w\_a - a - \lambda}{2\beta}, 1)$
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 :
$A_t(A, E; \Lambda) = min\_A, EL(A, E; \Lambda)$
$= min_{A,E}||A||_* + \Lambda||E||_1 + < \Lambda, D - A - E >$
$= min_A||A||_* - < \Lambda, A > + min_E \lambda||E||_1 - < \Lambda, E > + < \Lambda, D >$

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

$A_t(A, E; \Lambda) =< \Lambda, D >$
$S.t \quad ||\Lambda||_2 \leq 1, ||\Lambda||_\infty \leq \lambda$

so the dual problem is:
$max_\Lambda \quad < \Lambda, D >$
$S.t \quad ||\Lambda||_2 \leq 1, ||\Lambda||_\infty \leq \lambda$

6.1 Solve RPCA by ADMM

the ADMM sub-problem is:
A-sub-problem:
$A_{k+1} = argmin_A||A||_* + \frac{\beta}{2}||D - A - E_k + \Lambda_k/\beta||_F^2$

E-sub-problem:
$E_{k+1} = argmin_E \lambda||E||_1 + \frac{\beta}{2}||D - A_{k+1} - E + \Lambda_k||_F^2$

E-sub-problem has closed-form solution as follows:
$E_{k+1} = S_{\lambda\beta^{-1}}(D - A_k + \Lambda_k/\beta)$.

$S_\epsilon = sgn(x)max(|x| - \epsilon, 0)$,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

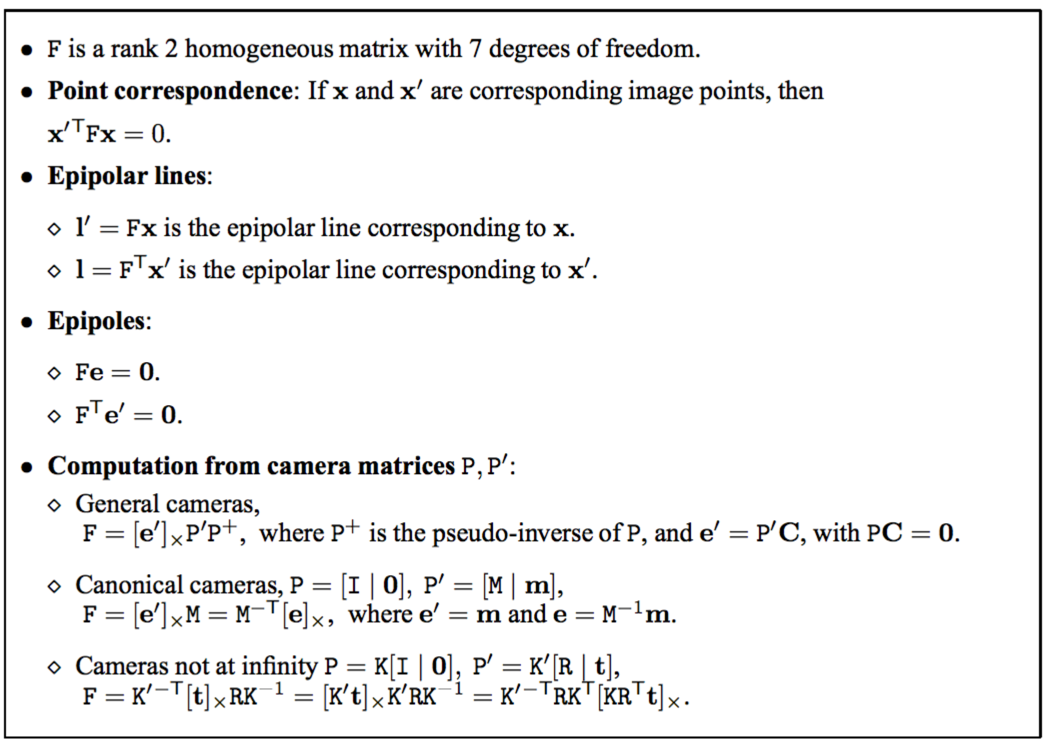
Lin et al.[2] suggest updating the penalty parameter β as follows:
$\beta_{k+1} = min(\beta_{max}, \rho\beta_k)$
where ρ_{max} is an upper bound of $\{\beta_k\}$. the value of ρ is defined as:

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

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

7 SFM

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



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

Figure 1: Summary of Fundamental matrix properties

8 Reinforcement Learning

Markov Property:
$P(s_{t+1}|s_t) = P(s_{t+1}|s_1, ..., s_t)$

Stochastic Policy:
$\pi(a|s) = P(a_s = a|s_t = s)$

Value function:
$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

State-value function:
$V_\pi(s) = E_\pi(G_t|s_t = s)$

Action-value function:
$Q_\pi(s,a) = E_\pi(G_t|s_t = s, a_t = a)$

Bellman Equation:
$V_\pi(s) = E_\pi(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...|s_t = s)$
$= E_\pi(r_{t+1} + \gamma(r_{t+2} + \gamma r_{t+3} + ...)|s_t = s)$
$= E_\pi(r_{t+1} + \gamma G_{t+1}|s_t = s)$
$= E_\pi(r_{t+1} + \gamma V_\pi(s_{t+1})|s_t = s)$

For state-value function, Bellman Equation can be written as:
$V_\pi(s) = E_\pi(r_{t+1} + \gamma V_\pi(s_{t+1})|s_t = s)$
$= \sum_{\alpha \in A} \pi(a|s) \sum_{s' \in S} P(s^{'}|s,a)[r(s,a,s^{'}) + \gamma V_\pi(s^{'})]$
$r(s,a,s^{'})$ is same with r_{t+1} to some extent.

For action-value function, Bellman Equation can be written as:
$Q_\pi(s,a) = E_\pi(r(s,a,s^{'}) + \gamma Q_\pi(s^{'},a^{'})|s,a)$
$= \sum_{s' \in S} P(s^{'}|s,a)[r(s,a,s^{'}) + \gamma \sum_{a' \in A} \pi(a^{'},s^{'}) Q_\pi(s^{'},a^{'}))]$

Normally, we just assume the $\pi(a|s), p(s^{'}|s,a), r(s,a,s^{'})$ are known. so we can solve the linear equation above. however we data become huge, it is not accessible to solve the Bellman Equation directly.

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.