Learn a regularizer and get insight of traditional optimization algorithms

Yu Liu[1]

[1]MSOR Student in SEAS, yl4342

October 2020

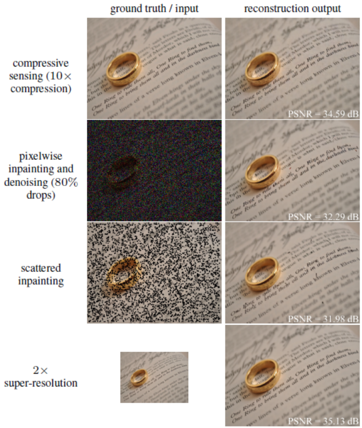
Content

- Inverse problem in image processing
- ADMM
- Unrolled Optimization and Neumann Network

Inverse problem in image processing

the heart problem

Reconstruct image $x \in R^d$ from observation $y \in R^m$ of the form $y = Ax + n$, n is the noise.



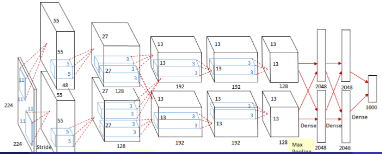
Inverse problem in image processing

iterative reconstruction

- Regularize the inverse problem with prior knowledge, and use optimization algorithm to solve it.
- Example: solve this problem, $\min_x \frac{1}{2}||y - Ax||_2^2 + \lambda\phi(x)$
- $\phi(x)$ can be sparse constraint, low-rank constraint or total variation penalty.

learning

- Use a deep neural network to learn the relation between x and y.
- Example: fully connected neural network, convolutional neural network.



Inverse problem in image processing

- Iterative method can solve any inverse problem, but have poor performance relatively. And it takes a long time to deal with one image.
- Learning method have good performance but is designed for one specific problem and data set. And we do not use the information of transform matrix A.
- We want to combine these two method.

ADMM[2]

ADMM

We start with: $\min_x \frac{1}{2}||y - Ax||_2^2 + \lambda\phi(x)$.

- Transform this problem: $\min_{x,z} \frac{1}{2}||y - Az||_2^2 + \lambda\phi(x), s.t. x = z$
- Write the augmented Lagrangian function:
  $L(x, z, u) = \frac{1}{2}||y - Az||_2^2 + \lambda\phi(x) + \frac{\rho}{2}||x - z + u||_2^2$
- Apply ADMM:
  $x^{k+1} = arg\min_x \frac{\rho}{2}||x - z^k + u^k||_2^2 + \lambda\phi(x)$
  $z^{k+1} = arg\min_z \frac{1}{2}||y - Az||_2^2 + \frac{\rho}{2}||x^{k+1} - z + u^k||_2^2$
  $u^{k+1} = u^k + x^{k+1} - z^{k+1}$
- The update of u, z is simple and the hardest part is the update of x.

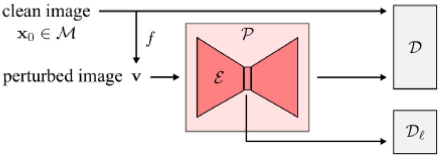
ADMM

Proximal Operator

- $prox_f(x) = arg\min_u(f(u) + \frac{1}{2}||u - x||_2^2)$
- special case: $prox_{I_C}(x) = Pc(x)$
- Notice that $x^{k+1} = prox_{\frac{\lambda}{\rho}\phi}(z^k - u^k)$, which means this step is a proximal operator of the regularizer ϕ.
- The best regularizer is the indicator of all natural image I_M, M is the set of all natural image. And if we choose this indicator as the regularizer, the step above become the projection on M.

ADMM

- So we want to train a classifier D to approximate I_M, and based on it to train a projection function P to approximate $prox_{I_M}$. We use neural net work to learn them.
- The detailed structure is as follows.
- By adding noise to clean image x we get perturbed image v, and use the pairs (x_i, v_i) to train D, P.
- Also, the out come of P can also be used to train D. And we add a additional classifier D_l in the latent space of the encoder part of P, this can help to avoid overfitting(add a regularization term in object function).



Unrolled Optimization and Neumann Network[1]

Gradient Descent

We start with: $\min_x \frac{1}{2}||y - Ax||_2^2 + r(x)$.

- Let the gradient of r be R, and the gradient descent is:
  $x_{k+1} = x_k - \eta(A^T(Ax_k - y) + R(x_k))$
  rewrite it as:
  $x_{k+1} = (I - \eta A^T A)x_k - \eta R(x_k) + \eta A^T y$

Another perspective

- If $R(x)$ is linear, for example, $r(x) = \frac{1}{2}x^T Rx$, so $R(x) = Rx$.
  And we can compute the optimizer: $x^* = (A^T A + R)^{-1} A^T y$
- We treat it as a Nuemann series:$(I - A)^{-1} = \Sigma_{k=0}^{\infty} A^k$
  And we get: $A^{-1} = \eta \Sigma_{k=0}^{\infty}(I - \eta A)^k$
- So we have: $x^* = \Sigma_{k=0}^{\infty}(I - \eta A^T A - \eta R)^k(\eta A^T y)$
  And we truncate it into:$\hat{x} = \Sigma_{k=0}^{N}(I - \eta A^T A - \eta R)^k(\eta A^T y)$

Unrolled Optimization and Neumann Network

Two ways to compute $\Sigma_{k=0}^{N} A^k$

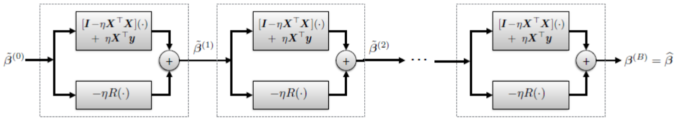
- Set $X_0 = I$, and let $X_{k+1} = AX_k + I$, then $X_N = \Sigma_{k=0}^{N} A^k$.
- Set $X_0 = I$, and let $X_{k+1} = AX_k$, then $\Sigma_{k=0}^{N} X_k = \Sigma_{k=0}^{N} A^k$.

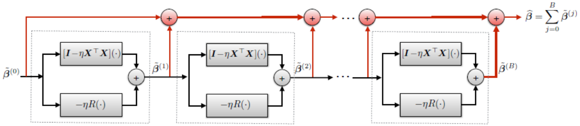
Two ways to compute $\hat{x}$

- Set $X_0 = \eta A^T y$, and let $X_{k+1} = (I - \eta A^T A - \eta R)X_k + \eta A^T y$, then $X_N = \Sigma_{k=0}^{N} \hat{A}^k X_0$.
- Set $X_0 = \eta A^T y$, and let $X_{k+1} = (I - \eta A^T A - \eta R)X_k$, then $\Sigma_{k=0}^{N} X_k = \Sigma_{k=0}^{N} \hat{A}^k X_0$.

Unrolled Optimization and Neumann Network

- Since gradient descent method is of sense, we can treat the two method as heuristic methods($R(x)$ may not be linear). So we have the following two structures. In both of them, $R(x)$ is learned by a network.
- The first method is the Unrolled Optimization method and the second is the Neumann Network method.





Summary

- To combine iteration and learning is to first using network to learn a classifier, a proximal operator, that's all to learn the structure of the natural image set, which is a better prior knowledge. With this prior knowledge, we can have better iteration method.
- Also, to combine this two can solve on a specific kind of data set, but we can deal with all kinds of problems, make full use of the transform matrix A and the data set. In some situation it is a better idea.

Reference

Davis Gilton, Greg Ongie, and Rebecca Willett.
Neumann networks for linear inverse problems in imaging.
IEEE Transactions on Computational Imaging, 6:328–343, 2019.

JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan.
One network to solve them all–solving linear inverse problems using deep projection models.
In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.