Summary of Idea

Chen Qian

1 What's the problem?

What we want to do is to find out a model to construct the relationship between Hash and saliency map. In addition, we hope to have a fast and accurate algorithm to solve this problem. Regarding the construction of the hash model, it can be divided into two mainstream models. One is the local sensitive hash (LSH)(we call it one-step hash) method formed by the random projection matrix, and the other is the spectral hash (SH)(we call it two-step hash) method formed by the Laplacian matrix in graph theory. Given instances data $X = \{x_1, ..., x_n\} \in R^{p \times n}$, LSH model utilize the random projection matrix W to re-arrange the feature of data, then utilize the binary thresholding to get the hash codes. LSH model can be written as follows

$$\min_{B,W} ||B - sgn(WX^T)||_2^2 \tag{1}$$

In general, there is a constraint on W that W is a row of orthogonal matrix, i.e., $W^T W = I$. In actual processing, W is generally initialized as a random matrix generated by Gaussian distribution.

On the contrary, SH model utilize the feature representation model in Machine Learning. SH model can be written as follows:

$$\min_{B} tr B^T L B$$

$$s.t. \quad B^T 1 = 0, B^T B = I, B \in \{-1, 1\}^{p \times n}$$

where L is the Laplacian matrix of a graph composed of instance data. There, $B^T 1 = 0$ is called Balanced Constraint and $B^T B = I$ is called Uncorrelated Constraint. $B \in \{-1, 1\}^{p \times n}$ ensure the discreteness and validity of Hash codes.

In fact, LSH and SH have a certain connection, they actually use the same thinking method with different tools. LSH hopes to combine the features of the instances out of

order through an random arrangement, and then generate Hash codes. In statistics, the data generated after a certain random combination of different data is the same should be a small probability event. When this event occurs, it means that the assumption is not true, the original data is the same. I think SH achieves the purpose of this random arrangement through two processes, which is the processing from data to graph and from graph to Laplace matrix.

As for the nature of Hash that both hopes to have, they are actually the same. The first is about the binary constraint, one is to directly sgn the results of the arrangement, and the other is to pass a discrete constraint. Regarding the item of irrelevance, one is the uncorrelated in the process of changing by constraint, i.e., $W^T W = I$, and the other is the uncorrelated of the result of changing by constraint. As for the balance constraint, it is actually contained in the model in the LSH. The model generated by random projection, imposing a balance constraint will destroy the randomness of the projection.

Regarding saliency detection, in fact, superpixels that are difficult to be well interpreted by other locations in the image are detected as saliency content by a certain model. Given a image X, it can be divide into n super-pixel by simple linear iterative clustering (SLIC), then utilize the feature exaction method get the image feature matrix $F = \{f_1, ..., f_n\} \in R^{l \times n}$. The simplest model that uses interpretation is the model of rPCA for significance detection, which can be written as follow:

$$\min_{L,S} ||L||_* + ||S||_1$$
$$s.t. \quad F = L + S$$

where L is a low-rank matrix which can be seen as the background (non-saliency) and S is a sparse matrix which is the foreground (saliency). Regarding the problem of saliency detection, it is to detect a minimum error model under the mutual representation of superpixels in the optimization problem.

First of all, there is a certain similarity between Hash and saliency detection, both of which are about the mutual representation between instances. In Hash, if an image can be well represented by several other images, it means that there is a certain degree of similarity between these images. The difficulty of linking Hash and saliency detection lies in how to deal with problems in two dimensions. Hash is an established relationship between images and images, whose dimensions are equivalent to a "three-dimensional" problem. Saliency detection is a kind of connection established within an image, and its temperature is equivalent to a "two-dimensional" problem. **How to transform "two-dimensional" to "three-dimensional" problem, or "three-dimensional" to "two-dimensional" problem is the difficulty of establishing this model.**

2 Idea about the model

1. Ignore dimension: The easiest way to deal with two dimensions is to ignore the dimension problem. If all the data is in the same dimension, then the dimension problem will not exist. We can accomplish this with the help of the Laplacian matrix of feature representation in machine learning. Generally, the composition of a Laplacian matrix is obtained from a graph, which means that there is a step from data to graph when processing data. The most commonly used measure in this step is to use the two norm error between image pixels, which can be written as follow:

$$A = [w_{ij}] = \begin{cases} \exp -\dfrac{||x_i - x_j||^2}{\sigma^2} & if \quad x_i \in N_k(x_j) \quad or \quad x_j \in N_k(x_i) \\ o & otherwise \end{cases} \tag{2}$$

Then the degree matrix D can be get, so the Laplacian matrix is $L = D - A$.

In equation (2), the graph node distance between two images is the l_2 norm of the image. So the easiest way I think of is to turn this metric into a measure of saliency. That is to say, the adjacency matrix here is no longer obtained by the l_2 norm of the image, but becomes the point-to-point difference of the image's saliency map. This method is the simplest. In fact, it is not the model of the hash problem but the metric that measures the similarity of the image. Therefore, any prior can be projected onto the graph in this way and then turned into a Laplacian matrix to be added to the hash coding process.

$$A' = [w_{ij}] = \begin{cases} \exp -\dfrac{||p_i - p_j||^2}{\sigma^2} & if \quad p_i \in N_k(p_j) \quad or \quad p_j \in N_k(p_i) \\ o & otherwise \end{cases}$$

where p_i is the prior information. In this form, the form of the hash problem has not changed, it is still the form of SH.

2. Increase the dimension of saliency detection: This method is similar to the method of using rPCA to detect the shot boundary in video. Given a data set $X = \{x_1, ..., x_n\}$, get the feature matrix $F = \{f_1, ..., f_n\}$, where f_i can be obtained using color distribution histogram, SIFT and other methods. What we want to get is the hash code $B = \{b_1, ..., b_n\}$. The saliency based hash problem can be written as follow:

$$\min_{B,L,S} ||B - sgn(WS)||_2^2 + ||L||_* + ||S||_1 \tag{3}$$

$$s.t. \quad F = L + S, W^T W = I$$

The meaning here is that in such a matrix F, the image can also be decomposed using rPCA. In this case, the low-rank matrix L represents the public information in the image,

that is, the background in a video environment. S represents the detailed information in the image, which is equivalent to the foreground in the video environment. If you use the saliency information for the hash process, you just want to give non-saliency a lower weight, so that it is limited reflected in the hash process. Because each image has information, it can be ignored. The reason for this phenomenon is that if the feature is given a lower weight, there will be a high probability of becoming 0 during the binarization process. In F, L is actually such a function. As the low-rank partial structure of all image information, if its structure is put into the Hash code, the matrix formed by rows equal to the first ten bits of code is a low-rank structure. This structure uses ten bits of information and may only be able to express 2 bits or 4 bits of information, which is obviously high consumption and low income. S is completely different. In S, it is actually equivalent to the special fingerprint possessed by each image. Each hash code formed by such a fingerprint has a distinctive label meaning.

Of course, there are also possible improvements here, especially the first one. The first term of equation (3) adopts the form in LSH, which can be replaced with the form in SH. But in the SH form, how to construct the problem between B and S is a key. One form that comes to mind is as follows:

$$\min_{B,L,S} trB^T MB + ||B^T B - S^T S||_2^2 + ||L||_* + ||S||_1$$

$$s.t. \quad F = L + S, B^T 1 = 0, B^T B = I, B \in \{-1, 1\}^{p \times n}$$

where M is the prior Laplacian matrix. The obvious shortcoming of this model is that the dimension of the Laplacian matrix is too large. And it is difficult to use the anchor graph to reduce the dimension of the Laplacian matrix.

3. Leveraging a hierarchical model: This model is based on a hierarchical model of graphs. Given a image dataset $X = \{x_1, ..., x_n\}$, utilize the SLIC method segmenting the every image into m super-pixel. Then use the superpixel where the center pixel is located as the root node of the subgraph to construct the graph in the image with the saliency as the metric. Then use the other metrics of image to construct the graph of the root node of the subgraph. In this way, we can get the single-connected adjacency matrix between sub-graphs, and then obtain a Laplacian matrix $L \in R^{mn \times nm}$. Then the model can be solved following the SH.

$$\min_{B} trB^T LB$$

$$s.t. \quad B^T 1 = 0, B^T B = I, B \in \{-1, 1\}^{p \times n}$$

Notice : For an instance x_i, the hash is not the b_i but the $[b_{k(i-1)+1}, ..., b_k i]$.

4. Reduce the dimension of Hash: Transform the Hash code problem from matrix optimization to tensor optimization. Since I do not know much about the definition and optimization methods about tensor, a clear model has not been given yet. But what is certain is that if the Hash code of an example in the solution process is turned into a matrix, the problem of "cross dimensions" is perfectly solved.

Notice : All the above methods adopt the method of changing the original problem dimension. In fact, a cross-dimensional model is what we need actually. For example, let's take SH as an example, if we can obtain a model in which three items are determined, as shown below

$$\min_{B,L,S} tr(B^T M B) + \sum_{i=1}^{n} (||L^i||_* + ||S^i||_1)$$

where M is the Laplacian of original data. L^i and S^i are the low-rank and sparse matrix of super-pixel matrix F^i.

Obviously this model is incomplete. First of all, this model lacks a "bridge", which establish the relationship between each column of B and S. In addition, even if there is a "bridge", there are certain questions about whether this model can perform actual calculations. The problem is too complicated. Therefore, a better model remains to be explored.

3 Idea about the algorithm

In the improvement process, LSH only has one more W than SH, and its nature has not changed. It only needs to be solved by alternating iteration. So here mainly SH, to discuss some ideas about the simultaneous processing of three constraints. The problem can be written as follow:

$$\min_B \mathcal{L}(B) \tag{4}$$
$$s.t. \quad B^T B = I_p, B^T 1 = 0, B \in \{-1, 1\}^{n \times p}$$

The first is for orthogonal constraints. Here, the method of paper[1] is used to deal with orthogonal constraints. The orthogonal constraint is subjected to a convex relaxation, and the problem is optimized on the closed convex hull of the orthogonal constraint. In this way, the problem (4) can be reformulated as follow:

$$\min_B \mathcal{L}(B) - \frac{1}{2} < \Phi(\nabla f(B)^T B), B^T B - I_p > + \frac{\beta}{4} ||B^T B - I_p||_F^2 \tag{5}$$
$$s.t. \quad B \in \mathcal{M}, B^T 1 = 0, B \in \{-1, 1\}^{n \times p}$$

Observe (5), we will find that the form of (5) violates the paper[1] motivation. Since (5) here is the actual problem of Hash, there are two more constraints on B, which leads to the fact that the three constraints are actually a non-convex constraint. Especially the addition of discrete constraints makes the problem very NP-hard. So we need to improve (5). One way I thought of it was to add an auxiliary variable C. Then, (5) can be reformulated as follow:

$$\min_{B,C} \mathcal{L}(C) - \frac{1}{2} < \Phi(\nabla f(C)^T C), C^T C - I_p > + \frac{\beta}{4}||C^T C - I_p||_F^2 + \frac{\mu}{4}||B - C||_2^2 \qquad (6)$$

$$s.t. \quad C \in \mathcal{M}, B^T 1 = 0, B \in \{-1, 1\}^{n \times p}$$

Since the auxiliary variable term $\frac{\mu}{4}||B - C||_2^2$ is also quadratic differentiable, it does not change the assumption of the objective function in paper[1].

For problem (6), if there are no discrete constraints and equilibrium constraints, the optimal solution can be obtained directly by the first-order method or the second-order method. So the next step is to deal with the remaining two constraints. For convenience, we introduce the following notation.

$$h(B, C) = \mathcal{L}(C) - \frac{1}{2} < \Phi(\nabla f(C)^T C), C^T C - I_p > + \frac{\beta}{4}||C^T C - I_p||_F^2 + \frac{\mu}{4}||B - C||_2^2$$

1. Similar-Forward-Backward Method[2]: For the remaining two constraints, the most difficult thing to deal with is the binary constraint. The simplest method here refers to the paper[2], using similar-Forward-Backward method(sFB). For the problem without orthogonal constraints:

$$\min_{B,C} h(B, C)$$

$$s.t. \quad C \in \mathcal{M}, B^T 1 = 0, B \in \{-1, 1\}^{n \times p}$$

There, introduce the following indicator function

$$\delta_H(B) = \begin{cases} 0 & if \quad B \in H \\ +\infty & otherwise \end{cases}$$

where H is a nonempty and closed set. Let $\mathcal{H}$ denoted the binary codes space $\{-1, 1\}^{n \times p}$. The function $\delta_{\mathcal{H}}(B)$ yields infinity as long as one entry of B does not belong to the binary domain $\{-1, 1\}$. With the indicator function, the problem can be reformulated as follow:

$$\min_{B,C} h(B, C)$$

$$s.t. \quad C \in \mathcal{M}, B^T 1 = 0$$

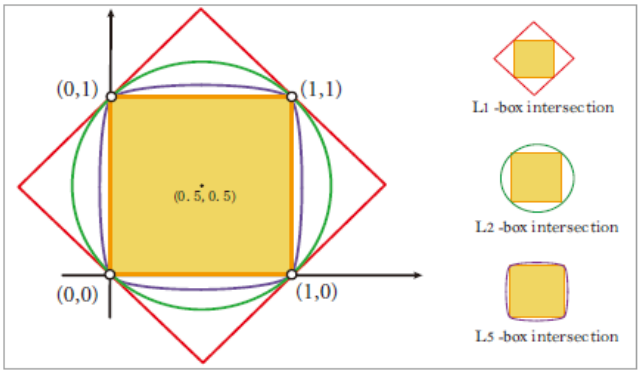

Fig. 1: Geometric illustration

Regarding the balance constraint, the previous article is a bit vague. There are two main opinions: complete relaxation and penalty function. Since the penalty term formed by the constraints is quadratic differentiable, adding or not adding will not affect the solution of the problem. Taking the penalty constraint to the objective function as an example, the problem becomes;

$$\min_{B,C} h(B,C) + \frac{\rho}{2}||B^T 1||^2$$

$$s.t. \quad B \in -1, 1^{n\times p}$$

Utilize the gradient method, when C is fixed

$$\nabla g(B) = \frac{\mu}{2}(B - C) + \rho B^1 1^T$$

With this, the binary optimization is conducted by updating variable B in each iteration with

$$B^{j+1} = sgn(B^j - \frac{1}{\lambda}\nabla g(B^j))$$

When B is fixed, it can iterate following the paper[1].

2. l_p-box & ADMM[3]: The idea handle the binary constraints by replacing them with an equivalent set of continuous constraints, namely the intersection between the box (n-convex constraints) and the shifted l_p-sphere (a non-convex constraints), of which a geometric interpretation is shown in Fig.1.

Proposition 1.

l_p-Box Intersection: The binary set $\{0,1\}^n$ can be equivalently replaced by the intersection between a box S_b and a $(n-1)$-dimensional sphere S_p as follows:

$$x \in \{0,1\}^n \Leftrightarrow x \in [0,1]^n \cap \{x : ||x - \frac{1}{2}1_n||_p^p = \frac{n}{2^p}\}$$

where $p \in (0, \infty)$*, and* $S_b = [0,1]^n = \{x | ||x||_\infty \leq 1\}, S_p = \{y : ||y - \frac{1}{2}1_n||_p^p = \frac{n}{2^p}\}$*. Note that* S_p *can be seen as a* $(n-1)$*-dimensional* l_p*-sphere centered as* $\frac{1}{2}1_n$ *with radius* $\frac{n^{\frac{1}{p}}}{2}$*.*

Rather than adding these equivalent continuous constraints into the objective as penalty methods do, the original problem can be solved by using the alternating direction method of multipliers (ADMM). First, introduce the auxiliary variables to separate the constraints, thus, simplifying the ADMM updates of all the primal variables without changing the form of the objective function, as formulated in problem (7).

$$\min_{B,C,y_1,y_2} h(B,C) \tag{7}$$
$$s.t. \quad C \in \mathcal{M}, B^T 1 = 0, B = y_1, B = y_2, y_1 \in S_b, y_2 \in S_p$$

where $S_b = \{y : 0 \leq y \leq 1\}, S_p = \{y : ||y - \frac{1}{2}1||_p^p = \frac{n}{2^p}\}$.

Due to the secondary differentiability of balance constraints, there are still two treatment methods. For convenience, we will ignore this item for the time being.

Reformulate (7) into the following form equivalently.

$$\min_{B,C,y} h(B,C) + h(y)$$
$$s.t \quad AB = y$$

Here, $y = [y_1; y_2]$ and $A = [I_n, I_n] \in \{0,1\}^{2n \times n}$. $h(y) = \mathbb{I}_{\{y_1 \in S_b \cap C\}} + \mathbb{I}_{\{y_2 \in S_p\}} \cdot \mathbb{I}_{\{a\}}$ denotes the indications function: if a is true, then $\mathbb{I}_{\{a\}} = 0$, otherwise $\mathbb{I}_{\{a\}} = \infty$. Formulate the ADMM update steps for (7) based on the following augmented Lagrangian

$$h(B,C,y_1,y_2,z_1,z_2) = h(B,C) + h(y) + z_1^T(B - y_1) + z_2^T(B - y_2) + \frac{\rho}{2}||x - y_1||_2^2 + \frac{\rho}{2}||x - y_2||_2^2$$

Here, (z_1, z_2) indicated dual variables, while (ρ_1, ρ_2) are positive penalty parameters.

Then, the problem can be solved by the gradient decrease and projection method.

3. Another Relaxation About Two Term: This method relaxes balance constraints and discrete constraints into one constraint. Give the specific form first, and then analyze it.

$$\min_{B,C} h(B,C)$$
$$s.t. \quad C \in \mathcal{M}, ||b_i + 1||_0 \leq k/2$$

where k is the length of the hash code and b_i is the i-th column of B. Then binary the final B. Despite the lack of theoretical proof, we can explain the feasibility of this penalty from two different structures.

The reason why I would like to use a l_0 norm constraint to relax two constraints is mainly from the purpose of these two constraints to consider. First of all, the balance

constraint is to ensure that in a series of Hash codes, the number of 1 and -1 is equal, i.e., $k/2$. The paper[4] analyzes this constraint and believes that this constraint is too strong, which may lead to the phenomenon of overfitting of real data. Therefore, this constraint is completely relaxed directly. However, completely relaxing this constraint will also cause a problem, that is, the constraint of the balance of the code will be lost, and the entire code will cause incomplete expression of information. So how to deal with this constraint is a question worth thinking about. As for the discrete constraints, the way the teacher mentioned before that the l_0 norm is actually similar to discrete constraints, and they can be written in an equivalent form under certain circumstances. So I wondered if I could turn the two constraints into a problem. Returning to the form of l_0 norm constraint, the proximal operator of l_0 norm keeps the part that is greater than the threshold and punishes the part that is less than the threshold. This is similar to the structure of the discrete constraint. The discrete constraint of applying sgn is to transform the greater than 0 to 1, and the less than 0 to -1. If it is all that is greater than the threshold but is set to 1, the one that is less than the threshold is still all punished. In this way, in the Hash problem, the l_0 norm of a column of Hash codes ($\{0,1\}^k$) corresponds to the number of 1. Therefore, I think that such two constraints can be transformed into such a zero norm constraint.

Based on the analysis of the effects achieved by the two structures, I assume that the threshold of the proximal operator of l_0 is small or even close to 0. In this case, purely from the point of view of the penalty in the solution process, the effect of discrete constraints should be achieved. And this will not make the balance constraint too strong, as far as possible to prevent the occurrence of over-fitting of real data. Of course, if we feel too relaxed, we can add a lower bound to l_0, as follow:

$$\epsilon \leq ||b_i + 1||_0 \leq k/2$$

As for the choice of the lower bound, I think it can be considered from the perspective of combinatorial optimization, to see what range of data information can be expressed as much as possible.

Regarding the solution of this problem, I think it is quite simple, and it is even possible to use the quasi-Newton based hard threshold iterative algorithm proposed by the undergraduate graduation design.

Algorithm 1 First-order method for solving PenC (PenCF)

Require: $f : \mathbb{R}^{n\times p} \mapsto \mathbb{R}$, $\beta > 0$;

Randomly choose X_0 on Stiefel manifold, set $k = 0$;
while not terminate **do**
  Compute inexact gradient: $D_k := \nabla_X \mathcal{L}(X_k, \Lambda)\big|_{\Lambda=\Lambda(X_k)}$;
  $\tilde{X}_{k+1} = X_k - \eta_k D_k$;
  if $\|X_{k+1}\|_F > K$ **then**
    $X_{k+1} = \frac{K}{\|\tilde{X}_{k+1}\|_F}\tilde{X}_{k+1}$;
  else
    $X_{k+1} = \tilde{X}_{k+1}$;
  end if
  $k++$;
end while
Return X_k.

Fig. 2: First-order method for solving PenC (PenCF)

4 Example

Given a SH problem, for feature of data set $X = [x_1, ..., x_n] \in R^{k\times n}$, find the hash code $B = [b_1, ..., b_n] \in R^{p\times n}$, where $p \leq k$

$$\min_B \mathcal{L}(B) = tr(BLB^T) \tag{4.1}$$

$$s.t. \quad BB^T = I_p, B^T 1 = 0, B \in \{-1, 1\}^{p\times n}$$

where L is the Laplacian matrix.

Firstly, to deal with the orthogonal constraint, (4.1) can be reformulated:

$$\min_{B,C} \mathcal{H}(B, C) = tr(CLC^T) - \frac{1}{2} < \Phi(\nabla\mathcal{L}(C)^T C), CC^T - I_p > + \frac{\beta}{4}||CC^T - I_p||_F^2 + \frac{\mu}{2}||B - C||_2^2 \tag{4.2}$$

$$s.t. \quad C \in \mathcal{M}, B^T 1 = 0, B \in \{-1, 1\}^{p\times n}$$

where $\Phi : R^{n\times n} \to R^{n\times n}, \Phi(M) = \frac{M+M^T}{2}$ denotes the linear operator that symmetrizes M. $\nabla\mathcal{L}(C) = 2LC^T$. Then the remaining two constraints of (4.2) can be relaxed to a l_0 norm.

$$\min_{B,C} \mathcal{H}(B, C) = tr(CLC^T) - \frac{1}{2} < \Phi(\nabla\mathcal{L}(C)^T C), CC^T - I_p > + \frac{\beta}{4}||CC^T - I_p||_F^2 + \frac{\mu}{2}||B - C||_2^2 \tag{4.3}$$

$$s.t. \quad C \in \mathcal{M}, ||b_i + 1||_0 \leq p/2$$

The solution of formula (4.3) adopts alternating iteration. When fixed B, the problem can be solved by the PenCF[1]. The original algorithm in shown in Fig.2. Then calculate the

inexact gradient:

$$\begin{aligned}
D :=& \nabla_C \mathcal{H}(B,C) \\
=& \nabla \mathcal{L}(C) - C\Phi(\nabla \mathcal{L}(C)^T C) - \frac{1}{2}\nabla \mathcal{L}(C)(CC^T - I_p) - \frac{1}{2}\nabla^2 \mathcal{L}(C)[C(CC^T - I_p)] \\
&+ \beta C(CC^T - I_p) - \mu(B - C) \\
=& 2LC^T - C\Phi(CL^T C) - \frac{1}{2}LC^T(CC^T - I_p) - \frac{1}{2}L[C(CC^T - I_p)] \\
&+ \beta C(CC^T - I_p) - \mu(B - C)
\end{aligned}$$

Then

$$D_k = 2LC_k^T - C\Phi(C_k L^T C_k) - \frac{1}{2}LC_k^T(C_k C_k^T - I_p) - \frac{1}{2}L[C_k(C_k C_k^T - I_p)] + \beta C_k(C_k C_k^T - I_p) - \mu(B_{k-1} - C_k) \tag{4.4}$$

So for iteration $k+1$, $\tilde{C}_{k+1} = C_k - \eta_k D_k$ where $\eta_k \leq \eta$ and $\eta = min\{\frac{\delta}{8KM_4}, \frac{\beta\delta^2}{9K^2M_4^2}\}$. In addition, there is $\delta \in (0, \frac{1}{3})$, $M_0 := \sup_{x\in\mathcal{M}} max\{1, ||\nabla \mathcal{L}(X)||_F\}$, $M_1 = \sup_{x\in\mathcal{M}} max\{1, ||\Phi(\nabla \mathcal{L}(X)^T X)||_F\}$, $M_2 = \sup_{x\in\mathcal{M}} max\{1, \nabla_C h(B,C)\}$, $M_2 = \sup_{x\in\mathcal{M}} max\{1, ||\nabla^2 \mathcal{L}(C)||_F\}$, $K > \sqrt{p}$, $M_4 = M_0 + M_1 K + \beta\delta K$, $L_1 := \sup_{X,Y\in\mathcal{M}} max\{1, \frac{||\Phi(\nabla \mathcal{L}(X)^T X) - \Phi(\nabla \mathcal{L}(Y)^T Y)||_2}{||X-Y||_2}\}$ and $\beta \geq max\{\frac{2}{3}n(M_0 + M_1), 3M_1 + 6L_1\}$

Then if $||\tilde{C}_{k+1}||_F > K$ then $C_{k+1} = \frac{K}{||\tilde{C}_{k+1}||_F}\tilde{C}_{k+1}$, else $C_{k+1} = \tilde{C}_{k+1}$.

Then when fixed C, the problem can be solved by Iterate Hard Thresholding algorithm (IHT). The problem can be rewritten as follow:

$$\min_B \mathcal{Q}(B) = \frac{\mu}{2}||B - C_k||_2^2$$

$$s.t. \quad ||b_i + 1||_0 \leq p/2$$

Then calculate the gradient about B

$$G := \mu(B - C_k)$$

So the gradient descent can be written as follow:

$$\tilde{B}_k = B_{k-1} - \gamma_k \mu(B_{k-1} - C_k)$$

Then using the proximal operator $\psi_\lambda(x)$

$$B_k = \psi_\lambda(\tilde{B}_k + 1)$$

Although there are explicit iteration, there are three problems should be thought. The first is the step size of the IHT iteration. In the classic IHT algorithm, only one method is the adaptive step size, called the NIHT method. But its step size is related to the perception

matrix in the compressed sensing problem, and it seems to be unsuitable here. I think the simplest IHTC method can be used to try:

$$\gamma_k = max\{0.9\gamma^k, 0.0001\}$$

The second question is about the multiplier μ selection of the fidelity term of the auxiliary variable C. The simplest form is the constant 1, a little more complicated is that it keeps increasing to 1. The most difficult problem is the choice of hard threshold function. In the iterative hard threshold algorithm, there are two forms of threshold. One is the most traditional hard threshold function, and the other is to retain the first k threshold forms in HTP. The simple form of HTP is not applicable. The most traditional form is still applicable here, but the choice of threshold feels a very troublesome problem, the main reason is that the inaccuracy of one-step update will lead to the accumulation of errors. I think it can be modified. The original balanced constraint is actually equivalent to choosing the threshold of the hard threshold function as the median of all values. So my idea is to randomly select a number between the median and the maximum value as the threshold when updating.

$$\psi_\lambda(x) = \begin{cases} x, & if \quad x \geq \lambda \\ 0, & otherwise \end{cases}$$

where λ is selected randomly in $[a, b]$, $a = max\{b_i\}$ and $b = median\{b_i\}$. But I don't know if this randomness will affect the solution of the problem. It is best if we can change a theoretical threshold selection method.

References

[1] Xiao Nachuan , Liu Xin , Yuan Ya-xiang. A Class of Smooth Exact Penalty Function Methods for Optimization Problems with Orthogonality Constraints[J].

[2] Shen F , Zhou X , Yang Y , et al. A Fast Optimization Method for General Binary Code Learning[J]. IEEE Transactions on Image Processing, 2016:1-1.

[3] B. Wu and B. Ghanem, "ℓ_p-Box ADMM: A Versatile Framework for Integer Programming," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 7, pp. 1695-1708, 1 July 2019, doi: 10.1109/TPAMI.2018.2845842.

[4] Zhao K, Lu H, Mei J. Locality preserving hashing. Proceedings of the National Conference on Articial Intelligence. 4. 2874-2880, 2014.