A Unified Active and Semi-Supervised Learning Framework for Image Compression

Xiaofei He * Ming Ji Hujun Bao
State Key Lab of CAD&CG, College of Computer Science, Zhejiang University
No. 388 Yu Hang Tang Rd., Hangzhou, Zhejiang, China
{xiaofeihe, jiming, bao}@cad.zju.edu.cn

Abstract

We consider the problem of lossy image compression from machine learning perspective. Typical image compression algorithms first transform the image from its spatial domain representation to frequency domain representation using some transform technique, such as Discrete Cosine Transform and Discrete Wavelet Transform, and then code the transformed values. Recently, instead of performing a frequency transformation, machine learning based approach has been proposed which uses the color information from a few representative pixels to learn a model which predicts color on the rest of the pixels. Selecting the most representative pixels is essentially an active learning problem, while colorization is a semi-supervised learning problem. In this paper, we propose a novel active learning algorithm, called Graph Regularized Experimental Design (GRED), which shares the same principle of the semi-supervised learning algorithm used for colorization. This way, active and semi-supervised learning is unified into a single framework for pixel selection and colorization. Our experimental results suggest that the proposed approach achieves higher compression ratio and image quality, while the compression time is significantly reduced.

1. Introduction

Due to the rapid growth of the number of images on the Web, there is an increasing demand for better image compression techniques [3]. A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information. The objective of image compression is to reduce redundancy of the image data in order to be able to store or transmit data in an efficient form. Typical image compression algorithms first transform the image from its spatial domain representation to frequency domain representation using some transform technique, such as Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT), and then code the transformed values (coefficients).

The mainstream signal-processing-based compression schemes share a common architecture, namely transform followed by entropy coding. Recently, technologies in machine learning have been applied to image compression and have shown remarkable progress [3]. Cheng et al. propose to treat image compression as a machine learning problem. Instead of performing a frequency transformation they store a grayscale version of the image and color labels of a few representative pixels. Using the stored information they apply Laplacian Regularized Least Squares (LapRLS, [2]) to learn a model which predicts the color for the rest of the pixels.

There are two fundamental steps in machine learning based image compression: selecting the most representative pixels as encoding and colorization [6] as decoding. The first step is essentially an active learning problem, while the second step is a semi-supervised learning problem. The previous approach develops a straightforward active learning scheme which does not directly optimize the objective function of colorization [3]. However, in order to archive the best compression performance, both encoding and decoding should be optimized simultaneously based on the same principle.

In this paper, we propose a novel active learning algorithm, called Graph Regularized Experimental Design (GRED), which is used to select the most representative samples. Our algorithm is fundamentally based on experimental design framework [1]. However, unlike traditional experimental design methods whose loss functions are only defined on the measured (labeled) points, the loss function of our proposed algorithm is defined on both measured and unmeasured points. Specifically, we use the same loss function as the LapRLS algorithm and select the samples which can lead to the minimum variances of the parameter estimates. This way, the performance of LapRLS can be improved. When applied to image compression, the color la-

*Corresponding author.

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

bels of the selected pixels together with the grayscale version of the image is stored as encoding. The LapRLS algorithm is applied to predict the color for the rest of the pixels as decoding.

The points below highlight the contributions of this paper:

1. A new approach for graph based active learning is developed, which is called GRED. For image compression, the GRED algorithm can be used to select the most representative pixels in a principled manner.
2. A unified active and semi-supervised learning framework for image compression is proposed. Both the encoding (active learning) and decoding (semi-supervised learning) share the same objective function so that the best compression ratio can be achieved.
3. Previous approach [3] applied Normalized Cut [10] to segment the image into small regions, based on which the learning model is built for compression. Our approach avoids using any segmentation technique, which significantly reduces the compression time.

The remainder of the paper is organized as follows: In Section 2, we provide a brief review of LapRLS and its application to image colorization and compression. Section 3 introduces our proposed GRED algorithm. Section 4 describes the nonlinear extension of our algorithm. We then present a unified active and semi-supervised learning framework for image compression in Section 5. A variety of experimental results are presented in Section 6. Finally, we provide some concluding remarks in Section 7.

2. Previous Work

In this section, we provide a brief description of the LapRLS algorithm [2] and its application to image colorization and compression [3]. We use $\mathbf{z}$ to denote the labeled point, and $\mathbf{x}$ to denote any point (either labeled or unlabeled).

Consider a linear regression model

$$y = \mathbf{w}^T\mathbf{x} + \epsilon \tag{1}$$

where y is the *observation*, $\mathbf{x} \in \mathbb{R}^n$ is the *independent variable*, $\mathbf{w}$ is the *weight vector* and ϵ is an unknown error with zero mean. Different observations have errors that are independent, but with equal variances σ^2. We define $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ to be the learner's output given input $\mathbf{x}$ and the weight vector $\mathbf{w}$.

The LapRLS algorithm makes use of both labeled and unlabeled points to learn a regression model f. It assumes that if two points $\mathbf{x}_i$, $\mathbf{x}_j$ are sufficiently close to each other, then their measurements $(f(\mathbf{x}_i),\ f(\mathbf{x}_j))$ are close as well. Suppose there are totally m points out of which k points are labeled. Let S be a similarity matrix. Thus, the LapRLS algorithm solves the following optimization problem:

$$\begin{aligned} J_{LapRLS}(\mathbf{w}) &= \textstyle\sum_{i=1}^{k} \big(f(\mathbf{z}_i) - y_i\big)^2 \\ &+ \tfrac{\lambda_1}{2} \textstyle\sum_{i,j=1}^{m} \big(f(\mathbf{x}_i) - f(\mathbf{x}_j)\big)^2 S_{ij} + \lambda_2\|\mathbf{w}\|^2 \end{aligned} \tag{2}$$

where y_i is the measurement (or, label) of $\mathbf{z}_i$. The loss function with our choice of symmetric weights S_{ij} ($S_{ij} = S_{ji}$) incurs a heavy penalty if neighboring points $\mathbf{x}_i$ and $\mathbf{x}_j$ are mapped far apart. Therefore, minimizing $J_{LapRLS}(\mathbf{w})$ is an attempt to ensure that if $\mathbf{x}_i$ and $\mathbf{x}_j$ are close then $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$ are close as well. There are many choices of the similarity matrix S. A simple definition is as follows:

$$S_{ij} = \begin{cases} 1, & \text{if } \mathbf{x}_i \text{ is among the } p \text{ nearest neighbors of } \mathbf{x}_j, \\ & \text{or } \mathbf{x}_j \text{ is among the } p \text{ nearest neighbors of } \mathbf{x}_i; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Let D be a diagonal matrix, $D_{ii} = \sum_j S_{ij}$, and $L = D - S$. The matrix L is called *graph Laplacian* in spectral graph theory [4]. Let $Z = (\mathbf{z}_1, \cdots, \mathbf{z}_k)$, $X = (\mathbf{x}_1, \cdots, \mathbf{x}_m)$, and $\mathbf{y} = (y_1, \cdots, y_k)^T$. The solution to the minimization problem (2) is given as follows:

$$\widehat{\mathbf{w}} = \big(ZZ^T + \lambda_1 XLX^T + \lambda_2 I\big)^{-1} Z\mathbf{y} \tag{4}$$

where I is an $n \times n$ identity matrix. LapRLS can also be performed in Reproducing Kernel Hilbert Space (RKHS, [8]) which gives rise to nonlinear solution. For more details about the algorithm, please see [2].

From machine learning perspective, image compression can be considered as a semi-supervised learning problem [3]. Given a set of color pixels (labeled examples) and a set of grayscale pixels (unlabeled examples) of an image, we want to learn a function which will predict color (labels) on the grayscale pixels. The assumption in LapRLS is also natural for image compression: if two pixels have similar intensity values and spatially close to each other then it is very likely that they have similar color values.

3. Graph Regularized Experimental Design

Another key step in learning based image compression is to select the most representative pixels. In this section, we introduce our active learning algorithm, called Graph Regularized Experimental Design (GRED), for this purpose based on LapRLS.

Often estimates of the parameters ($\widehat{\mathbf{w}}$) are of interest together with predictions of the responses ($\widehat{y}$) from the fitted model. The variances of the parameter estimates and predictions depend on the particular experimental designs used and should be as small as possible. We begin with the analysis of bias and variance.

3.1. Analysis of Bias and Variance

Since $y_i = \mathbf{w}^T\mathbf{z}_i + \epsilon$, we have $\mathbf{y} = Z^T\mathbf{w} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} = (\epsilon, \cdots, \epsilon)^T$. Clearly $E(\boldsymbol{\epsilon}) = \mathbf{0}$, where $\mathbf{0} = (0, \cdots, 0)$. We define

$$H = ZZ^T + \lambda_1 XLX^T + \lambda_2 I \tag{5}$$

and

$$\Lambda = \lambda_1 XLX^T + \lambda_2 I. \tag{6}$$

H is the hessian of $J_{LapRLS}(\mathbf{w})$. The bias of the estimator for the coefficient vector $\mathbf{w}$ can be computed as follows:

$$\begin{aligned} & E(\widehat{\mathbf{w}} - \mathbf{w}) \\ = \; & E(H^{-1}Z\mathbf{y}) - \mathbf{w} \\ = \; & H^{-1}ZE(\mathbf{y}) - \mathbf{w} \\ = \; & H^{-1}ZZ^T\mathbf{w} - \mathbf{w} \\ = \; & H^{-1}(ZZ^T + \Lambda - \Lambda)\mathbf{w} - \mathbf{w} \\ = \; & (I - H^{-1}\Lambda)\mathbf{w} - \mathbf{w} \\ = \; & -H^{-1}\Lambda\mathbf{w} \end{aligned} \tag{7}$$

By noticing that $Cov(\mathbf{y}) = \sigma^2 I$ and H is symmetric, the covariance matrix of $\widehat{\mathbf{w}}$ has the expression:

$$\begin{aligned} Cov(\widehat{\mathbf{w}}) &= Cov(H^{-1}Z\mathbf{y}) \\ &= H^{-1}ZCov(\mathbf{y})Z^TH^{-1} \\ &= \sigma^2 H^{-1}ZZ^TH^{-1} \\ &= \sigma^2 H^{-1}(H - \Lambda)H^{-1} \\ &= \sigma^2(H^{-1} - H^{-1}\Lambda H^{-1}) \end{aligned} \tag{8}$$

In order to make the estimator $\widehat{\mathbf{w}}$ as stable as possible, the size of covariance matrix $Cov(\widehat{\mathbf{w}})$ has to be as small as possible. Different measures of the size of the covariance matrix lead to different optimality criteria.

3.2. The Algorithm

The most important design criterion in application is that of D-optimality, in which the *determinant* of the covariance matrix is minimized. Two other popular criteria that have a statistical interpretation in terms of the covariance matrix are A- and E-optimality [1]. In A-optimality $\mathrm{Tr}(Cov(\mathbf{w}))$, the total variance of the parameter estimates, is minimized, equivalent to minimizing the *average* variance. In E-optimality the variance of the least well estimated linear combination $\mathbf{a}^T\widehat{\mathbf{w}}$ is minimized subject to the constraint that $\mathbf{a}^T\mathbf{a} = 1$. Thus the E in the name stands for *extreme*. For the detailed description about these optimality criteria, please see [1]. Let $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$ denote the set of all the data points and $\mathcal{Z}_k = \{\mathbf{z}_1, \cdots, \mathbf{z}_k\}$ denote the set of the first k selected points. Clearly, $\mathcal{Z}_k \subseteq \mathcal{X}$ and $k \leq m$.

In this work, we apply D-optimality to select the most informative samples due to its connection to the confidence region for the parameters [1]. Since the regularization parameters (λ_1 and λ_2) are usually set to be very small, we have:

$$|H^{-1} - H^{-1}\Lambda H^{-1}| \approx |H^{-1}|$$

Noticing that $|H|^{-1} = 1/|H|$, the problem is thus formally defined below:

Graph Regularized Experimental Design

$\max_{Z=(\mathbf{z}_1,\cdots,\mathbf{z}_k)} |H|$

where $\mathbf{z}_1, \cdots, \mathbf{z}_k$ are selected from$\{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$.

---

where $|\cdot|$ denote the determinant. In the following, we describe a sequential construction of graph regularized experimental designs. Let H_k be the hessian of J_{LapRLS} after k points are selected:

$$H_k = Z_k Z_k^T + \lambda_1 XLX^T + \lambda_2 I \tag{9}$$

where $Z_k = (\mathbf{z}_1, \cdots, \mathbf{z}_k)$. It is easy to show that the graph Laplacian L is positive semi-definite. Therefore, H_k is positive definite and invertible. The first data point $\mathbf{z}_1$ is selected such that $|\mathbf{z}_1\mathbf{z}_1^T + \lambda_1 XLX^T + \lambda_2 I|$ is maximized. Suppose k points have been selected. In other words, Z_k is given. Thus, the $(k+1)$-th point is selected such that $|H_{k+1}|$ is maximized:

$$\mathbf{z}_{k+1} = \arg\max_{\mathbf{z}\in\mathcal{X}-\mathcal{Z}_k} |H_k + \mathbf{z}\mathbf{z}^T| \tag{10}$$

By using matrix determinant lemma [5], the determinant of $H_k + \mathbf{z}\mathbf{z}^T$ can be written as a multiplicative updated of the determinant of H_k:

$$|H_k + \mathbf{z}\mathbf{z}^T| = |H_k| \cdot \left(1 + \mathbf{z}^T H_k^{-1}\mathbf{z}\right) \tag{11}$$

Since $|H_k|$ is a constant while selecting the $(k+1)$-th point, Eq. (10) can be rewritten as follows:

$$\mathbf{z}_{k+1} = \arg\max_{\mathbf{z}\in\mathcal{X}-\mathcal{Z}_k} \mathbf{z}^T H_k^{-1}\mathbf{z} \tag{12}$$

Once the $(k+1)$-th point is selected, the inverse of H_{k+1} can be updated based on the inverse of H_k, by using the Sherman-Morrison formula [9]:

$$\begin{aligned} H_{k+1}^{-1} &= (H_k + \mathbf{z}_{k+1}\mathbf{z}_{k+1})^{-1} \\ &= H_k^{-1} - \frac{H_k^{-1}\mathbf{z}_{k+1}\mathbf{z}_{k+1}^T H_k^{-1}}{1 + \mathbf{z}_{k+1}^T H_k^{-1}\mathbf{z}_{k+1}} \end{aligned} \tag{13}$$

Our analysis shows that we do not need to compute the matrix determinant and inverse. Instead, at each iteration we select a new point $\mathbf{z}$ such that $\mathbf{z}^T H_k^{-1}\mathbf{z}$ is maximized and the inverse of H_k can be efficiently updated in terms of Eq. (13).

4. Nonlinear Generalization

In this section, we describe how to generalize our algorithm to nonlinear problem by using kernel tricks [8].

We consider the problem in a feature space $\mathcal{F}$ induced by some nonlinear mapping: $\phi : \mathbb{R}^n \to \mathcal{F}$. For a properly chosen ϕ, an inner product $\langle , \rangle$ can be defined on $\mathcal{F}$ which makes for a so-called Reproducing Kernel Hilbert Space (RKHS). More specifically, $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ holds where $\mathcal{K}(\cdot, \cdot)$ is a positive semi-definite kernel function. Several popular kernel functions are: Gaussian kernel $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2)$; polynomial kernel $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = (1 + \langle \mathbf{x}_i, \mathbf{x}_j \rangle)^d$; Sigmoid kernel $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = tanh(\langle \mathbf{x}_i, \mathbf{x}_j \rangle + \alpha)$.

Let ϕ_X and ϕ_Z denote the corresponding data matrices in the Hilbert space:

$$\phi_X = \Big(\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_m)\Big)$$

$$\phi_Z = \Big(\phi(\mathbf{z}_1), \cdots, \phi(\mathbf{z}_k)\Big)$$

Thus, the hessian in the Hilbert space can be written as follows:

$$\phi_H = \phi_Z \phi_Z^T + \lambda_1 \phi_X L \phi_X^T + \lambda_2 I \tag{14}$$

Since ϕ is unknown, there is no way to compute the determinant of ϕ_H. Instead, we consider the regression model in RKHS:

$$y = \boldsymbol{\nu}^T \phi(\mathbf{x}) + \epsilon, \quad \boldsymbol{\nu} \in \mathcal{F}. \tag{15}$$

Thus, the objective function (2) in RKHS can be written as follows:

$$\begin{aligned} J_{LapRLS}(\boldsymbol{\nu}) &= \sum_{i=1}^{k} \left(\boldsymbol{\nu}^T \phi(\mathbf{z}_i) - y_i\right)^2 \\ &\quad + \frac{\lambda_1}{2} \sum_{i,j=1}^{m} \left(\boldsymbol{\nu}^T \phi(\mathbf{x}_i) - \boldsymbol{\nu}^T \phi(\mathbf{x}_j)\right)^2 S_{ij} \\ &\quad + \lambda_2 \|\boldsymbol{\nu}\|_{\mathcal{F}}^2 \end{aligned} \tag{16}$$

We have the following proposition.

Proposition 4.1. *Let $\mathcal{F}_X = \{\sum_{i=1}^m \alpha_i \phi(\boldsymbol{x}_i) | \alpha_i \in \mathbb{R}\}$ be a subspace of $\mathcal{F}$, the minimum solution to the problem (16) is in $\mathcal{F}_X$.*

Proof. Let $\mathcal{F}_X^\perp$ be the orthogonal complement of $\mathcal{F}_X$, i.e. $\mathcal{F} = \mathcal{F}_X \oplus \mathcal{F}_X^\perp$. Thus, for any point $\boldsymbol{\nu} \in \mathcal{F}$, it has orthogonal decomposition as follows:

$$\boldsymbol{\nu} = \boldsymbol{\nu}_{\mathcal{F}_X} + \boldsymbol{\nu}_{\mathcal{F}_X^\perp}, \quad \boldsymbol{\nu}_{\mathcal{F}_X} \in \mathcal{F}_X, \ \boldsymbol{\nu}_{\mathcal{F}_X^\perp} \in \mathcal{F}_X^\perp.$$

Since $\langle \boldsymbol{\nu}_{\mathcal{F}_X^\perp}, \phi(\mathbf{x}_i) \rangle = 0$, $i = 1, \cdots, m$, and $\|\boldsymbol{\nu}\|_{\mathcal{F}}^2 = \|\boldsymbol{\nu}_{\mathcal{F}_X}\|_{\mathcal{F}}^2 + \|\boldsymbol{\nu}_{\mathcal{F}_X^\perp}\|_{\mathcal{F}}^2 \geq \|\boldsymbol{\nu}_{\mathcal{F}_X}\|_{\mathcal{F}}^2$, it is easy to see that:

$$J_{LapRLS}(\boldsymbol{\nu}) \geq J_{LapRLS}(\boldsymbol{\nu}_{\mathcal{F}_X})$$

This completes the proof. □

From Proposition 4.1, we see that $\boldsymbol{\nu}$ can be represented as a linear combination of $\phi(\mathbf{x}_i)$, $i = 1, \cdots, m$:

$$\boldsymbol{\nu} = \sum_i \alpha_i \phi(\mathbf{x}_i) = \phi_X \boldsymbol{\alpha}, \tag{17}$$

where $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_m)$. Since ϕ_X is a constant matrix in RKHS, Eq. (17) tells us that the only parameter we need to estimate is $\boldsymbol{\alpha}$. Similar to the linear algorithm described in Section 3.2, here we select the points such that the size of $Cov(\boldsymbol{\alpha})$ is minimized.

From Eq. (17), we have:

$$Cov(\boldsymbol{\nu}) = Cov(\phi_X \boldsymbol{\alpha}) = \phi_X Cov(\boldsymbol{\alpha}) \phi_X^T \tag{18}$$

Let ϕ_X^{-1} be the *Moore-Penrose* inverse of ϕ_X. By noticing that $Cov(\boldsymbol{\nu}) \approx \sigma^2 \phi_H^{-1}$, we have:

$$\begin{aligned} & Cov(\boldsymbol{\alpha}) \\ \approx\ & \sigma^2 \phi_X^{-1} \phi_H^{-1} (\phi_X^T)^{-1} \\ =\ & \sigma^2 \left(\phi_X^T \left(\phi_Z \phi_Z^T + \lambda_1 \phi_X L \phi_X^T + \lambda_2 I\right) \phi_X\right)^{-1} \end{aligned}$$

Let K_{XZ} be a $m \times k$ matrix ($K_{XZ,ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{z}_j)$), K_{ZX} be a $k \times m$ matrix ($K_{ZX,ij} = \mathcal{K}(\mathbf{z}_i, \mathbf{x}_j)$), and K_{XX} be a $m \times m$ matrix ($K_{XX,ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$). Thus, we have:

$$Cov(\boldsymbol{\alpha}) \approx \sigma^2 \left(K_{XZ} K_{ZX} + \lambda_1 K_{XX} L K_{XX} + \lambda_2 K_{XX}\right)^{-1}. \tag{19}$$

The nonlinear problem is defined as follows:

Kernel Graph Regularized Experimental Design

$$\max_{Z = (\mathbf{z}_1, \cdots, \mathbf{z}_k)} |K_{XZ} K_{ZX} + \lambda_1 K_{XX} L K_{XX} + \lambda_2 K_{XX}|$$

where $\mathbf{z}_1, \cdots, \mathbf{z}_k$ are selected from $\{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$.

Let $\mathbf{u}_i$ be the i-th column vector of K_{XX} and $\mathcal{U}$ be the set of $\{\mathbf{u}_1, \cdots, \mathbf{u}_m\}$. $\mathbf{u}_i$ corresponds to $\mathbf{x}_i$, and $\mathbf{u}_i = (\mathcal{K}(\mathbf{x}_i, \mathbf{x}_1), \cdots, \mathcal{K}(\mathbf{x}_i, \mathbf{x}_m))^T$. Thus, the first data point $\mathbf{v}_1 \in \mathcal{U}$ is selected such that $|\mathbf{v}_1 \mathbf{v}_1^T + \lambda_1 K_{XX} L K_{XX} + \lambda_2 K_{XX}|$ is maximized. Suppose k points have been selected, and we define:

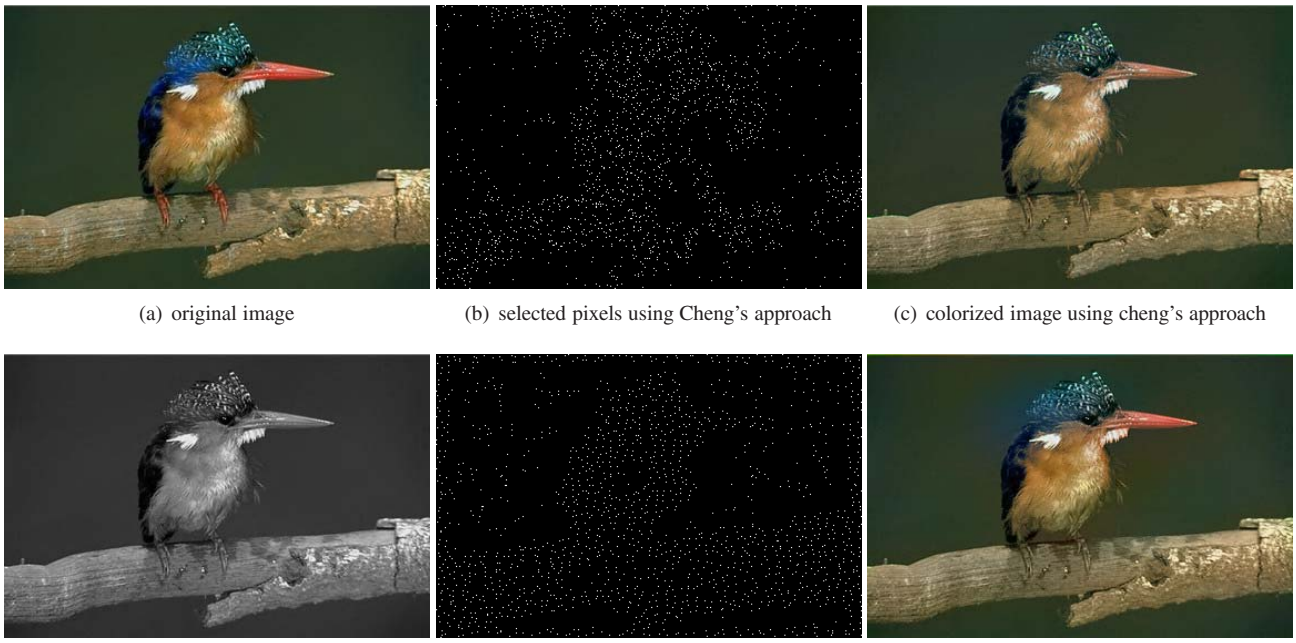
$$M_k = K_{XZ_k} K_{Z_k X} + \lambda_1 K_{XX} L K_{XX} + \lambda_2 K_{XX} \tag{20}$$

Let $\mathcal{V}_k = \{\mathbf{v}_1, \cdots, \mathbf{v}_k\}$. The $(k+1)$-th point is selected by solving the following optimization problem:

$$\mathbf{v}_{k+1} = \arg\max_{\mathbf{v} \in \mathcal{U} - \mathcal{V}_k} |\mathbf{v}\mathbf{v}^T + M_k|, \tag{21}$$

which is equivalent to the following problem by matrix determinant lemma [5]:

$$\mathbf{v}_{k+1} = \arg\max_{\mathbf{v} \in \mathcal{U} - \mathcal{V}_k} \mathbf{v}^T M_k^{-1} \mathbf{v}, \tag{22}$$



(a) original image (b) selected pixels using Cheng's approach (c) colorized image using cheng's approach

(d) grayscale image (e) selected pixels using GRED (f) colorized image using GRED

Figure 2. Image compression example. As can be seen, our approach obtains visually more pleasing results than Cheng's approach.

(a) NCut (b) rectangular segmentation

Figure 1. (a) Cheng's approach uses NCut to segment the image. (b) Our approach simply segments the image into small rectangles.

Similar to the linear algorithm described in Section 3.2, the inverse of M_k can be updated as follows:

$$M_{k+1}^{-1} = M_k^{-1} - \frac{M_k^{-1}\mathbf{v}_{k+1}\mathbf{v}_{k+1}^T M_k^{-1}}{1 + \mathbf{v}_{k+1}^T M_k^{-1}\mathbf{v}_{k+1}} \qquad (23)$$

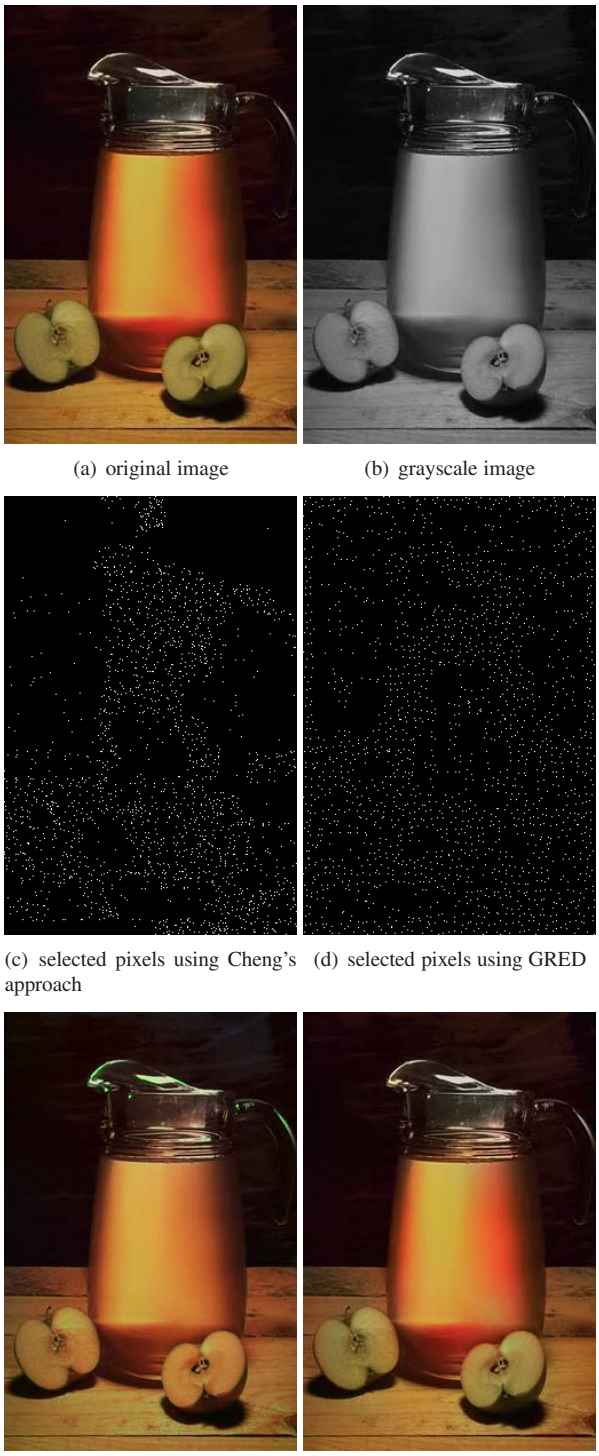
As can be seen, the computation of the nonlinear algorithm is essentially the same as that of the linear algorithm, except that the data vectors $\mathbf{x}_i (i = 1, \cdots, m)$ are replaced by $\mathbf{u}_i (i = 1, \cdots, m)$.

5. A Unified Learning Framework for Image Compression

In this section we describe how to apply our GRED algorithm, together with the LapRLS algorithm, to image compression. In order to measure the quality of image compression, we employ Peak Signal to Noise Ratio (PSNR) score.

Following Cheng et al. [3], we work in YUV space where Y is the intensity (luminance) channel, U and V are the chrominance channels that encode the color. We predict U, V values independently. In [3], the features used to represent a pixel include the spatial location in the image grid and the local texture information. However, our empirical investigation has shown that texture information is not useful for predicting color information. Therefore, in our work we only use spatial location and grayscale value to characterize each pixel.

Given an image with n pixels, the LapRLS algorithm involves inverting a dense $n \times n$ matrix, which is computationally intractable. To reduce the computational burden, Cheng et al. [3] propose to use *super-pixel* representation [7] obtained by segmenting the image into regions using Normalized Cut (NCut, [10]), and pick pixels randomly from these regions. However, NCut itself is a time consuming algorithm. In our work we avoid using any sophisticated segmentation algorithms, but simply segment the image into rectangular regions. Fig. 1 shows an example of NCut segmentation and rectangular segmentation. The number of regions is set to 2800 in our experiments according to the image size. We randomly pick one pixel from each region, so



(a) original image (b) grayscale image

(c) selected pixels using Cheng's approach (d) selected pixels using GRED

(e) colorized image using Cheng's approach (f) colorized image using GRED

Figure 3. Image compression example.

there are totally 2800 pixels over which we then construct a 4-nearest neighbor graph.

Once the graph is constructed, we can apply our active learning algorithm to select the most representative pixels. The quality of the compressed image usually varies with the number of selected color pixels (labels). The more selected color pixels, the higher image quality. On the other hand, selecting more color pixels reduces the compression ratio. Since the graph is constructed on the 2800 pixels, rather than all the pixels in the image, the maximum number of selected color pixels (labels) is 2800. For decoding, the LapRLS algorithm is applied to learn a function to predict the color on the rest of the grayscale pixels.

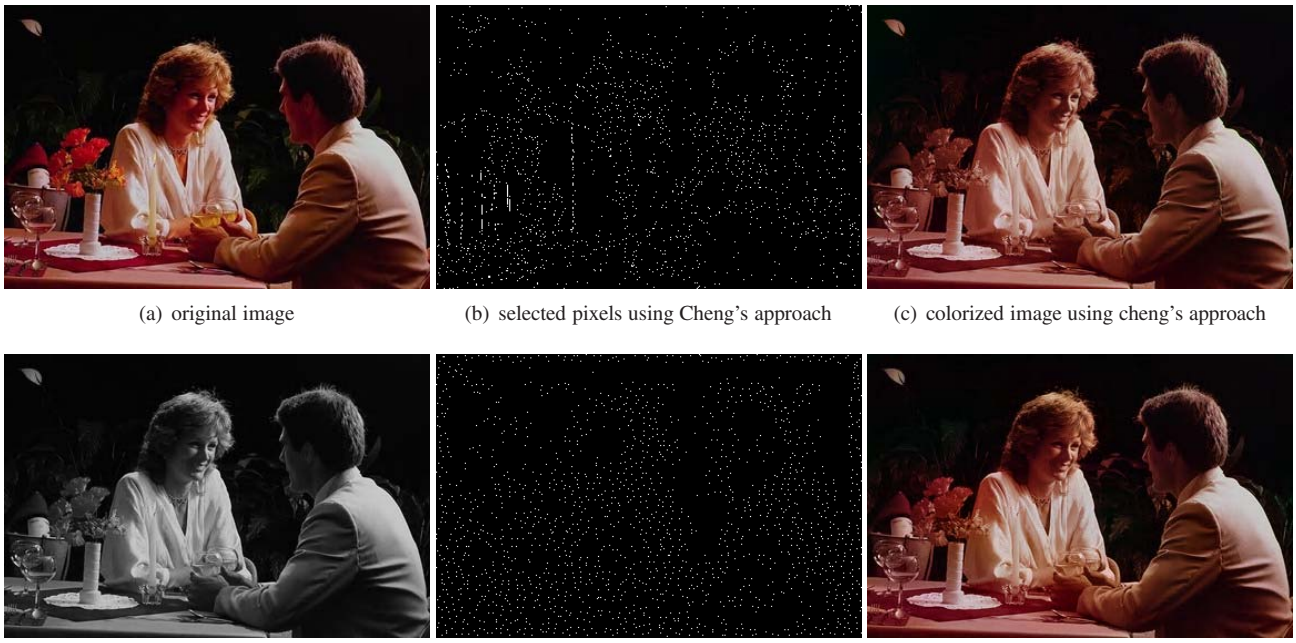
6. Experimental Results

In this section, we investigate the performance of our proposed approach for image compression. Several illustrative examples and experimental evaluations are provided.

6.1. Evaluation on Image Quality

In this subsection, we fix the number of selected color pixels (labels) and compare our approach to Cheng's approach [3] in terms of image quality.

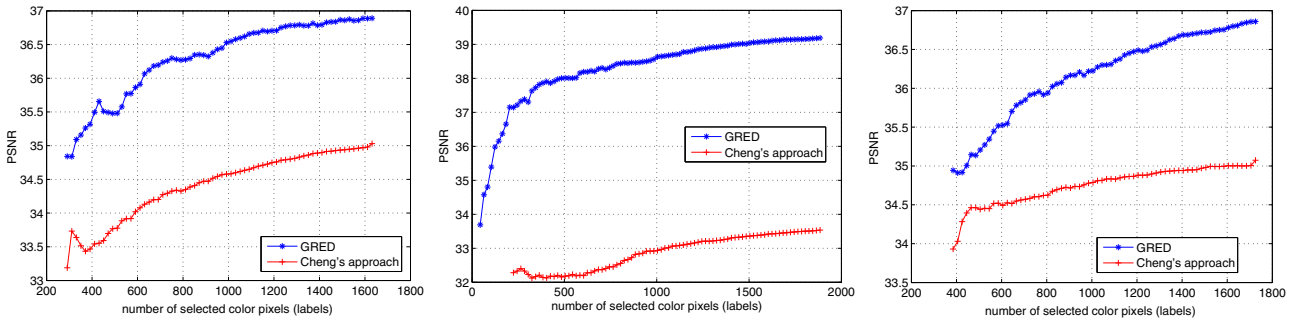
Our first test example is an image of bird, as shown in Fig. 2a. The image is of size 256×384. The corresponding grayscale image is depicted in Fig. 2d. We first use all the 2800 color pixels randomly picked from image regions to learn a compression model by using Cheng's approach. Let p be the maximum PSNR that can be achieved. As more color pixels selected from the 2800 pixels, PSNR tends to converge to p very quickly. In other words, there is little improvement on PSNR after selecting a certain number of color pixels. Therefore, in our experiments, we select color pixels until the PSNR reaches $p - 0.1$. This way, 1631 pixels are chosen. Note that, decreasing the PSNR by 0.1 almost has no effect on the image quality, but significantly improves the compression ratio. That is why it is not necessary to use to all the 2800 color pixels. In our algorithm, we choose the same number of color pixels, leading to the same compression ratio as Cheng's approach. Fig. 2b and 2e depict the pixels chosen by Cheng's active learning approach and our GRED algorithm, respectively. Comparing to Cheng's approach, our algorithm chooses more pixels on the bird and bough and less pixels on other regions. From the pixels chosen by our algorithm, we can see a rough contour of the bird and bough. Fig. 2c and 2f show the corresponding colorized (or uncompressed) images obtained by Cheng's approach and our approach, respectively. As can be seen, our approach produces visually more pleasing results. Using the same number of color pixels as labels, Cheng's approach achieve a PSNR of 35.03, and our approach achieve a PSNR of 36.89. Fig. 5 shows the plot of PSNR versus the number of selected color pixels. We have also compared with the JPEG2000 method by using Adobe Photoshop CS4. To achieve the same compression ratio, the



(a) original image (b) selected pixels using Cheng's approach (c) colorized image using cheng's approach

(d) grayscale image (e) selected pixels using GRED (f) colorized image using GRED

Figure 4. Image compression example.



(a) bird (b) drink cup (c) couples

Figure 5. PSNR versus the number of selected color pixels.

PSNR obtained by JPEG2000 is 32.63.

Fig. 3-4 show two other examples on images of drink cup and couples. The number of selected color pixels are 1884 and 1725, respectively. The PSNR obtained by our approach for these two images are 39.18 and 36.86. The PSNR obtained by Cheng's approach for these two images are 33.53 and 35.07. The PSNR obtained by JPEG2000 for these two images are 33.60 and 33.15. Our approach significantly outperforms both Cheng's approach and JPEG2000 in terms of image quality.

6.2. Evaluation on Compression Ratio

In this subsection, we fix the PSNR and compare our approach to Cheng's approach in terms of compression ratio.

In the case of the image of bird, the original JPEG image occupies 57686 bytes on disk, while the grayscale version occupies 18871 bytes. To achieve the PSNR of 35.03 (see subsection 6.1), Cheng's approach needs to choose 1631 color pixels, whereas our approach only needs to choose 331 color pixels. Each pixel for which we store color information requires 4 bytes of additional storage: 2 bytes for the color information (the luminance channel is already present in the grayscale image), and 2 bytes to encode its location [3]. Thus, Cheng's approach needs 6524 bytes for extra storage, leading to a compression ratio of $(18871 + 6524)/57686 = 44.0\%$. Similarly, the compression ratio of our approach on this image is $(18871 + 331 \times 4)/57686 = 35.0\%$. We have also compared with the JPEG2000 method by using Adobe Photoshop CS4. To achieve the same PSNR value, the compression ratio obtained by JPEG2000 is 49.6%. Table 1 shows the compression ratio for the other two images. As can be seen, our

Table 1. Comparison on compression ratio.

image	JPEG size (color)	JPEG size (grayscale)	# of selected pixels (Cheng)	# of selected pixels (GRED)	compression ratio (Cheng)	compression ratio (GRED)	comp. ratio (JPEG2000)
bird	57686	18871	1631	331	44.0%	35.0%	49.6%
couples	71209	23459	1725	465	42.6%	35.6%	46.5%
drink cup	57084	16692	1884	44	42.4%	29.6%	42.7%

Table 2. Comparison on compression time (in seconds).

image	Cheng's approach	GRED
bird	9986	928
couples	9339	1303
drink cup	8854	168

approach significantly outperforms both Cheng's approach and JPEG2000 in terms of compression ratio.

We have also compared our approach to Cheng's approach in terms of compression time, as shown in Table 2. Our approach is much faster than Cheng's approach. This is mainly because: 1) Cheng's approach uses Normalized Cut [10] to segment the image into regions, which is very time consuming. Our approach avoids using any sophisticated segmentation algorithms, but simply divides the image into rectangular regions; 2) Cheng's approach needs to select much more color pixels than our approach. Every time the new color pixels are selected, the learning model has to be reconstructed, which is time consuming.

7. Conclusion

We have introduced a novel active learning algorithm called Graph Regularized Experimental Design. The proposed algorithm is motivated from recent progress on graph based semi-supervised learning. Using techniques from optimum experimental design, we select those points such that the size of the covariance matrix of the estimated coefficients is minimized. We further propose a unified active and semi-supervised learning framework for image compression. The proposed compression approach significantly outperforms Cheng's approach [3] in terms of image quality, compression ratio and compression time.

Acknowledgements

The work was supported by Program for Changjiang Scholars and Innovative Research Team in University (IRT0652,PCSIRT), National Science Foundation of China under Grant 60875044, and National Key Basic Research Foundation of China under Grant 2009CB320801.

References

[1] A. C. Atkinson and A. N. Donev. *Optimum Experimental Designs*. Oxford University Press, 2007.

[2] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

[3] L. Cheng and S. Vishwanathan. Learning to compress images and videos. In *Proceedings of the* 24^{th} *International Conference on Machine Learning*, Corvallis, OR, 2007.

[4] F. R. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. AMS, 1997.

[5] D. A. Harville. *Matrix Algebra from a Statistician's Perspective*. Springer-Verlag, 1997.

[6] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *Proceedings of the 2004 ACM SIGGRAPH Conference*, Los Angeles, CA, 2004.

[7] X. Ren and J. Malik. Learning a classification model for segmentation. In *Proc. 9th IEEE International Conference on Computer Vision*, Nice, France, 2003.

[8] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

[9] J. Sherman and W. J. Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Annals of Mathematical Statistics*, 21:124–127, 1950.

[10] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.