Journal of Pattern Recognition Research 1 (2009) 92-109

Received August 28, 2008. Accepted November 12, 2008.

Illumination Chromaticity Estimation Using Linear Learning Methods

Vivek Agarwal *agarwal1@purdue.edu*
School of Nuclear Engineering, Purdue University
400 Central Dr, West Lafayette, IN 47907, USA

Andrei V. Gribok *agribok@bioanalysis.org*
BHSAI/MRMC, Building 363 Miller Dr, Fort Detrick, MD 21792, USA

Andreas Koschan *akoschan@utk.edu*
Besma R. Abidi *besma@utk.edu*
Mongi A. Abidi *abidi@utk.edu*
Department of Electrical and Computer Engineering, The University of Tennessee
331 Ferris Hall, Knoxville, TN 37996, USA

WWW.JPRR.ORG

Abstract

In this paper, we present the application of two linear machine learning techniques; ridge regression and kernel regression for the estimation of illumination chromaticity. A number of machine learning techniques, neural networks and support vector machines in particular, are used to estimate the illumination chromaticity. These nonlinear approaches are shown to outperform many traditional algorithms. However, neither neural networks nor support vector machines were compared to linear regression tools in the past. We evaluate the performance of linear machine learning techniques and draw comparison with nonlinear machine learning techniques. Kernel regression achieves a mean root mean square chromaticity error of 0.052 while neural network results in 0.071. An improvement of 26% is achieved. Both quantitative and qualitative results show that the performances of the linear techniques are better when compared to nonlinear techniques on the same data set. Machine learning approaches are also compared with the gray-world and the scale by max algorithms. We perform uncertainty analysis of machine learning algorithms using a bootstrapped training data set to evaluate their consistency in the estimation of illumination chromaticity. Applications like video tracking and target detection, where illumination chromaticity estimation is important will be benefited by a better performance of linear machine learning algorithms.

Keywords: Illumination chromaticity, ridge regression, nonparametric kernel regression, neural networks, support vector machines, color constancy.

1. Introduction

Application of color as one of the major features in computer vision depends on the ability to approximately obtain a canonical color representation under various environmental conditions. There are many factors that can cause color variation with change in illumination being one of the important factors. It can be understood from the fact that a color image can be represented as a product of the surface reflectance $R(x, y, \lambda)$, the illumination property $L(\lambda)$, and the sensor characteristics $S(\lambda)$ which are functions of the wavelength λ, over a visible spectrum ω.

$$E_k(x, y, \lambda) = \int_{\omega} R(x, y, \lambda) L(\lambda) S_k(\lambda) d\lambda \tag{1}$$

© 2009 JPRR. All rights reserved. Permissions to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or to republish, requires a fee and/or special permission from JPRR.

where x and y are the image pixels, the subscript k represents the kth channel, and $E_k(x, y, \lambda)$ is the image corresponding to the kth channel ($k = R, G, B$). If constant surface reflectance and known sensor characteristics are assumed, then any variation in illumination may change the color appearance of the image.

In color constancy research, efforts are directed towards discounting the effect of illumination and obtaining a canonical color appearance. The human vision system exhibits an approximate color constancy processing. The same phenomena cannot be observed in machine vision systems. Color constancy process involves two steps. The first step is the estimation of the illumination chromaticity and the second step is the parametric computation of illuminant independent descriptor. However, algorithms like the gray-world (GW) and the scale by max (SBM) adjust the color of the image by scaling each channel of the image appropriately by a computed factor [1].

To achieve color constancy, many theories have been proposed by researchers in the field of color constancy [1]-[8]. Most of the theories identify color constancy as a very difficult and an under-constrained problem. According to Hadamard [9], a French mathematician, a problem is well-posed if the following three conditions are satisfied, (i) there exists a solution, (ii) this solution is unique, and (iii) this unique solution is stable. If any of these conditions is not satisfied, then the problem is ill-posed. In color constancy, the uniqueness and the stability of the solution cannot be guaranteed because of the high correlation between the color in the image and the color of the illuminant. This collinearity leads to imprecise estimation and a slight variation in collinearity may lead to a large variation in the estimation.

Our initial research on the application of ridge regression to color constancy was presented in [10]. However, in [10] many technical details on the performance of ridge regression were not explained. In addition, work on kernel regression based illumination chromaticity estimation and comprehensive comparison with other methods were not presented in [10]. Henceforth, in this paper, we present the application of two linear machine learning (ML) approaches; ridge regression (RR) and kernel regression (KR) to estimate the illumination chromaticity. Previously, neural networks (NN) and support vector machines (SVM) were applied to estimate the illumination chromaticity. However, neither neural networks nor support vector machines were compared to linear regression tools in the past. Therefore, we evaluate the performance of linear ML approaches in the estimation of illumination chromaticity and compare them with nonlinear ML algorithms. We perform the computation in the chromaticity space (r, g) in order to provide a fair comparison. An independent training data set [11] and test data set [12] of real images are used to train and evaluate the performance of both linear and nonlinear ML algorithms.

Based on our results, we show that the performance of linear ML algorithms in terms of mean root mean square (RMS) rg chromaticity error is better than the nonlinear ML algorithms on the same data set. Given the chromaticity estimate, the diagonal model [13] is used to obtain a color corrected image. The performances of both linear and nonlinear ML algorithms are also compared with basic color constancy algorithms like the gray-world and the scale by max because they are oldest and simplest algorithms.

We also discuss the issue of consistency of the learning algorithms in estimating the illumination chromaticity via uncertainty analysis using bootstrapping. Thus, the contributions of this paper are:

- Establishing that the linear regression tools could be as effective as other nonlinear ML algorithms in estimating the illumination chromaticity and in solving the color constancy problem.
- By comparing two linear regression tools we show that kernel regression can outperform ridge regression and nonlinear ML techniques.
- Uncertainty analysis of ML algorithms.

The paper is organized as follows: In Section 2, a brief summary of related works is presented. The representation of training and test data sets into design matrix and response vectors is discussed in Section 3. A discussion on each algorithm implemented in this paper is presented in Section 4. Algorithm evaluation and results are shown in Section 5. In Section 6, uncertainty analysis of ML algorithms and its significance is presented. Finally conclusions are drawn in Section 7.

2. Related Works

An under-constrained color constancy problem [1]-[8] requires algorithms to impose additional constraints to achieve a suitable solution. The constraints can be application specific or based on certain assumptions. Buchsbaum [1] assumes that the average of the reflected spectra corresponds to the actual illuminant. This is the main concept of the gray-world algorithm. Maloney et al. [2] assumes that surface reflectance falls in a two dimensional subspace as used in their algorithm for a trichormatic system. Brainard et al. [8] constructed a prior distribution to characterize illuminants and surfaces in the world. Then, using a given scene, they used the Bayesian rule for posterior estimation of illuminants and surfaces. Forsyth [3] estimated the color constant descriptors of the objects in a scene taken under standard canonical illuminants. The intersection of constraints given by the colors of the surfaces in the scene was used to estimate the color constant descriptor. Finlayson et al. [14] build a correlation matrix that correlates the chromaticity of the image with a predetermined scene illuminant based on maximum correlation. Agarwal et al. [15] categorized color constancy algorithms into *precalibrated approaches* and *data-driven approaches*. In precalibrated approaches, characteristics of sensor and canonical illumination are known. In data-driven approaches, sophisticated statistical and machine learning algorithms are discussed. Barnard et al. [16, 17] provide a comprehensive computational comparison between different color constancy algorithms both on real and synthetic data sets.

A number of color constancy algorithms are developed to work entirely in some chromaticity space [6, 14, 18], and much progress has been made by taking advantage of the simplifications afforded by this strategy. Since these algorithms ignore the magnitude of the image pixels, they are potentially less powerful than algorithms which attempt to use information that may be implicit in those values. It is commonly recognized that specular highlights carry information about the illuminant chromaticity [4, 19], and the fact that they are relatively bright is of use to some algorithms. This suggests that it can be beneficial to use an algorithm which utilizes all the three parameters, chromaticity and image intensity, even if the goal is chromaticity correction.

Machine learning approaches to the color constancy problem were mainly based on neural networks. Cardei et al. [20, 21] and Funt et al. [22] proposed a multilayer perceptron neural network based approach in chromaticity space. They showed that neural networks achieved better color constancy than the color by correlation algorithm [14]. Moore et al. [23] developed a neural network to deal with multiple illuminations. Nayak et al. [24] proposed

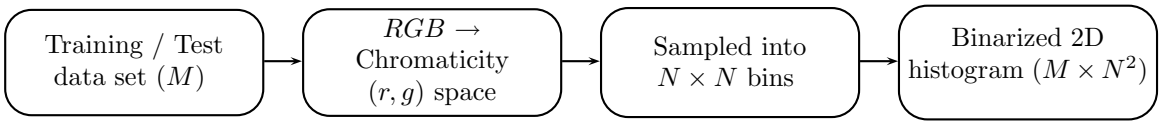

Fig. 1: A block diagram showing the steps to obtain the design matrix ($\mathbf{X}$).

an approach in RGB space to achieve color correction for skin tracking. Stanikunas et al. [25] performed investigation on color constancy using neural network and compared it to the human vision system. Funt et al. [26] proposed an algorithm based on support vector regression to achieve color constancy.

Recently, Schaefer et al. [27] proposed to combine the data obtained by a statistical color constancy algorithm, a version of the Color by Correlation algorithm [14], with that of a physics-based technique based on the dichromatic reflectance model. Manduchi [28] trained a Gaussian classifier with color samples from just one training image. Then, using a simple diagonal illumination model, the illuminants in a new scene that contains some of the surface classes seen in the training image, are estimated in a maximum likelihood framework. The assumption is made that one or more manually labeled images are available for training, where all surface classes are seen under the same "canonical" illuminant. Ebner [29] proposed a biologically inspired algorithm using genetic programming to evolve an algorithm for color constancy. The algorithm runs on a grid of processing elements and is based on the computation of local space average color that is used to estimate the illuminant locally for each image pixel. Given an estimate of the illuminant, the reflectances of the corresponding object points are computed.

3. Data Representation

We perform the computation in the chromaticity (r, g) space as this is a usual approach adopted by previous papers on color constancy. In this paper, we deal with only real images, so the preprocessing steps considered are similar to those taken by Cardei et al. [20]. The image is preprocessed before it is converted into chromaticity space. The clipped and dark pixels are removed. A pixel value below a threshold value of 7 (on a 0–255 scale) in any of the three RGB color channels are regarded as the dark pixel. The image is then averaged using a 5 by 5 local filter to reduce noise.

We use independent training data set [11] consisting of 2330 real images and the test data set [12] consisting of 320 real images. Since only the illumination chromaticity is of interest, not its intensity [20], each image in both data sets is converted into chromaticity space. Although there are many forms of chromaticity spaces, we use the one given by (2),

$$\begin{aligned} r &= R/(R+G+B) \\ g &= G/(R+G+B) \end{aligned} \tag{2}$$

The advantage of this chromaticity space is that it is bounded between 0 and 1, so it requires no additional preprocessing before inputting into ML algorithms. Using chromaticity space also reduces the dimensionality of data sets from 3D RGB space to 2D (r, g) space. The chromaticity space (r, g) is sampled into $N \times N$ bins which is later binarized to obtain a 2D binary chromaticity histogram of 1's and 0's where 1's represent the presence of illumination chromaticity in the bins and 0's represent the absence of illumination chromaticity in the

bins. The 2D $N \times N$ binary histogram matrix is represented as a vector of dimension $1 \times N^2$. Thus the design matrix ($\mathbf{X}$) of dimension $M \times N^2$ is obtained, where $M = 2330$ for training data set and $M = 320$ for test data set. The design matrix ($\mathbf{X}$) is expressed as $\mathbf{X} = (X_1, \ldots, X_M)$ where $X_i = (x_1, x_2, \ldots, x_{N^2})_i^T$, $i = 1, 2, , \ldots, M$. Fig. 3 illustrates the process to compute the design matrix ($\mathbf{X}$). The bin width is empirically selected. A bin width of $N = 32$ provides the best results in terms of illumination chromaticity estimation. Other larger bin width selections (48 and 64) did not improve the results much but slowed down the process drastically, while smaller bin width selections (8 and 16) performed poorly. This bin width selection is in accordance with [20]. In some practical situations, it is not possible to obtain large number of training images, so bootstrapping is used to generate a large number of training images [22].

A response vector $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_M)^T$ is required to obtain trained models, where $Y_i = (y_r, y_g)_i$, $i = 1, 2, \ldots, M$ is the chromaticity value corresponding to the chromaticity space (r, g) of each image in the data sets. The dimensionality of the response vector ($\mathbf{Y}$) is $M \times 2$. The response vector is known for the test data set [12] but unknown for the training data set [11]. The response vector ($\mathbf{Y}$) is obtained either by measuring the actual illumination chromaticity values while collecting the training images, or can be estimated using simple color constancy algorithms, such as, GW or SBM [22]. In this paper, the training data set consists of large number of real images taken under unknown illuminants with an uncalibrated camera (i.e., camera with unknown sensor sensitivities). So we make assumption (in accordance with [22]) that the training data set [11] contains relatively large number of colors. So the chromaticity space of the training data set is large. Under this assumption, the GW algorithm provides very accurate estimate of the illuminants in the training data set provided the data set is large. Hence, the GW estimate of the chromaticity of the training data set contain most (if not all) the chromaticity of the test data set. Funt et al. [22] showed that neural network trained using GW estimate of chromaticity, perform better than GW algorithm when tested on same unseen test images. We estimate the response vector for the training data set [11] using (3) as described in [22], i.e., the gray world assumption.

$$r_{GW} = 0.33\left(\frac{r_\mu}{r_{ill}}\right)$$
$$g_{GW} = 0.33\left(\frac{g_\mu}{g_{ill}}\right) \tag{3}$$

where $r_\mu = R_{av}/(R_{av} + G_{av} + B_{av})$, $g_\mu = G_{av}/(R_{av} + G_{av} + B_{av})$, and $\{r_{ill}, g_{ill}\}$ is the average chromaticity of the illuminant from all the images in the data set. The *RGB* triplet $[R_{av}, G_{av}, B_{av}]$ is computed by averaging the *RGB* values of all the pixels in the image. The actual response vector of the test data set is been made available by Barnard et al. [12]. The response vector of the test data set is simply the chromaticity response of the camera to a white patch as measured by Barnard et al. [12].

4. Algorithms

In this section, we discuss ML algorithms for regression applications, the gray-world algorithm, and the scale by max algorithm. Machine learning algorithms are visualized as distribution free approaches that model the input-output association present in the data and provide a good generalization when presented with previously unseen data. This is achieved in two main stages, training and testing.

4.1 Ridge Regression

The application of machine learning algorithms to achieve color constancy [16, 17], [20]-[26] has been strictly restricted to nonlinear regression techniques. Existing literature does not provide comparison with linear regression techniques, except in [10]. A linear regression model is of the form,

$$\boldsymbol{\varepsilon} + \mathbf{Y} = \boldsymbol{f}(\mathbf{X}, \boldsymbol{\beta}) \tag{4}$$

where $\boldsymbol{\beta}$ is the regression model coefficient vector, $\boldsymbol{\varepsilon}$ is the noise assumed to be normally and independently distributed, and $f(\cdot)$ is the function relating the values of response vector $\mathbf{Y}$ to the predictors. Ordinary least squares coefficients are computed by minimizing the residual square error loss function and are analytically represented as,

$$\widehat{\boldsymbol{\beta}}_{ols} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{5}$$

Since the columns of the input matrix are highly correlated, the correlation matrix $(\mathbf{X}^T\mathbf{X})$ is rank deficient and its inverse does not exist. The situation can be corrected by dropping highly correlated inputs thus reducing the degree of collinearity. Heorl et al. [30] proposed an alternative approach to solve a rank deficient system, known as ridge regression. In ridge regression, the solution is obtained by minimizing the combination of both the sum of squared errors and the norm of the $\boldsymbol{\beta}$ vector:

$$C = \gamma\|\boldsymbol{\beta}\|_2^2 + \sum_{i=1}^{M} \|\mathbf{Y} - f(\mathbf{X}, \boldsymbol{\beta})\|_2^2 \rightarrow \min_{\boldsymbol{\beta}} \tag{6}$$

where γ is a non-negative regularization parameter that determines the trade-off between the error and the smoothness of the solution. The value of γ is computed using a k-fold cross validation method. The analytical solution of (6) gives the ridge regression coefficients,

$$\widehat{\boldsymbol{\beta}}_{ridge} = (\mathbf{X}^T\mathbf{X} + \gamma\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y} \tag{7}$$

where $\mathbf{I}$ is an identity matrix.

4.2 Kernel Regression

Kernel regression is a locally weighted polynomial and nonparametric regression technique [31, 32]. There are two forms of local regression models, univariate regression and multivariate regression. We use multivariate regression to estimate the illumination chromaticity. The multivariate kernel regression model is given by,

$$\varepsilon_i + Y_i = f(X_i, \beta(X_q)) \tag{8}$$

where Y_i are the output values corresponding to the input data X_i (image in this paper), $\beta(X_q)$ is the kernel coefficient for the nonlinear model $f(X_i, \beta(X_q))$ for each query, and ε_i is a noise assumed to be normally and independently distributed. In kernel regression, every computation is with respect to a query. Therefore, unless a query is provided, no further computation is performed. For this reason, it is also referred to as a lazy learning approach. The output response at every query X_q is obtained by minimizing the cost function,

$$C(X_q) = \sum_{i=1}^{N} \left[(f(X_i, \beta(X_q)) - Y_i)^2 K_H(d(X_i, X_q))/\mathbf{H}\right] \tag{9}$$

where $K_H(\cdot)$ is the kernel weight function, $d(X_i, X_q)$ is the distance between the input data vector X_i and the query data vector X_q, and $\mathbf{H}$ is a diagonal bandwidth matrix containing same bandwidth values h (such that $\mathbf{H}(i,i) = h$). A query is a single value variable in case of univariate regression model and a vector in case of multivariate regression model. The best estimate $\widehat{Y_i}$ is obtained by minimizing the cost function at every query X_q. Thus, from (9), the estimation of Y_i involves, (i) defining a distance function, (ii) selecting a kernel function, (iii) selecting bandwidth of the kernel function, and (iv) selecting the order of the polynomial fit.

There are different distance measures [31]. A scalar Euclidean distance measure is given as,

$$\begin{aligned} d(X_i, X_q) &= \sqrt{\sum_i (X_i - X_q)^2} \\ &= \sqrt{\sum_i (X_i - X_q)^T (X_i - X_q)} \end{aligned} \tag{10}$$

The computed distance is weighted using a kernel function,

$$\begin{aligned} w_{qi} &= K_H(d(X_i, X_q))/\mathbf{H}(i,i) \\ &= \exp(-d(X_i, X_q)/h^2), i = 1, 2, \ldots, M \end{aligned} \tag{11}$$

where w_{qi} is the weight computed at every query q. Different kernel functions exist [31, 32]. However, the choice of the kernel function becomes insignificant if the data set is large [31]. The kernel regression coefficients are obtained using,

$$\beta(\boldsymbol{q}) = \left(\mathbf{X}_q^T \mathbf{W}_q \mathbf{X}_q\right)^{-1} \mathbf{X}_q^T \mathbf{W}_q \mathbf{Y} \tag{12}$$

where

$$\mathbf{X}_q = \begin{bmatrix} 1 & X_i - X_q & \ldots & (X_i - X_q)^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_M - X_q & \ldots & (X_M - X_q)^p \end{bmatrix} \tag{13}$$

in which p is the order of the polynomial fit and $\mathbf{W}_q = diag(w_{q1}, w_{q2}, \ldots, w_{qM})$ is the weight matrix. The output estimate $\widehat{Y}$ is obtained using the expression,

$$\widehat{\mathbf{Y}}(\mathbf{X}_q, p, \mathbf{H}) = \mathbf{e}^T \beta(\boldsymbol{q}) \tag{14}$$

where $\mathbf{e}^T$ is the $(p+1) \times 1$ vector having 1 in the first entry and all other entries are zero.

4.3 Neural Networks

Neural networks are a traditional learning method that learns complex input-output association by performing empirical risk minimization (ERM). There are many forms of neural network but we restrict our discussion to multilayer perceptron (MLP) feed-forward neural network using sigmoid activation function because most of the neural networks based color

constancy literature use MLP architecture. The sigmoid activation function $g(\cdot)$ is given by (15),

$$g(a) = \frac{1}{1 + \exp(-a)} \tag{15}$$

where a is the weighted sum of the inputs minus a threshold value.

The number of input nodes, the number of hidden layer, the number of hidden neurons, and number of output nodes defines the MLP architecture. The size of the MLP architecture defines the nature of the dimensionality of the function it is trying to model. Optimal selection of the architecture is one of the main steps towards optimizing the network behavior. A large architecture increases the training time and may also result in poor generalization due to over fitting. A small architecture may not be able to model the input-output association completely. This is commonly referred to as bias-variance dilemma [33]. Neural networks also have limitation to the dimensionality of the data, known as the curse of dimensionality. The output of the network can be defined as the weighted summation of all the inputs and a bias value transformed by nonlinear activation function. The backpropagation algorithm proposed by Rumelhart et al. [34] is in general used to train neural networks by minimizing the error function with respect to the weights. The basic backpropagation algorithm is based on the gradient descent algorithm. Many variations to gradient descent algorithm due to different forms of optimization routines are available [33]. Regularized training algorithms such as weight decay regularization and early stopping can also used to train neural networks [33].

4.4 Support Vector Machines

Support vector machines, a learning technique defined for both linear and nonlinear cases are based on the statistical learning theory of Vapnik et al. [35]. The support vector machines developed initially for classification purpose was extended to regression by Vapnik et al. [36]. SVM for regression seeks a continuous function $f(\cdot)$ that learns the input-output association by mapping them linearly on to a higher dimensional feature space. A simple expression of SVM for the nonlinear regression problem is,

$$f(\mathbf{X}, \mathbf{W}) = \sum_i w_i \phi_i(\mathbf{X}) + \boldsymbol{b} \tag{16}$$

where $\phi_i(\mathbf{X})$ is the mapping function of the design matrix $\mathbf{X}$ onto a higher dimensional feature space, w_i is the weight associated with every data i (image in this paper), $\mathbf{W}$ is the weight matrix, and $\boldsymbol{b}$ is the bias vector. The feature space of a nonlinear SVM is defined by kernel function.

The SVM extended to address the regression problem is known as support vector regression (SVR). Vapnik et al. [36] extended SVM to SVR by defining an appropriate linear loss function with ϵ insensitive zone known as ϵ insensitive loss function. The loss function is represented in (17) and is shown in Fig. 2. If the difference between the predicted $f(\mathbf{X}, \mathbf{W})$ and the measured $\mathbf{Y}$ value is within ϵ, then the error is zero, otherwise it is the magnitude of the difference between the predicted value and the radius of the tube. The parameter ϵ determines the range of an acceptable error and number of support vectors. The point that lies on the margin defined by ϵ are known as support vectors. When $\epsilon = 0$, the loss function is equivalentn to the least modulus function.

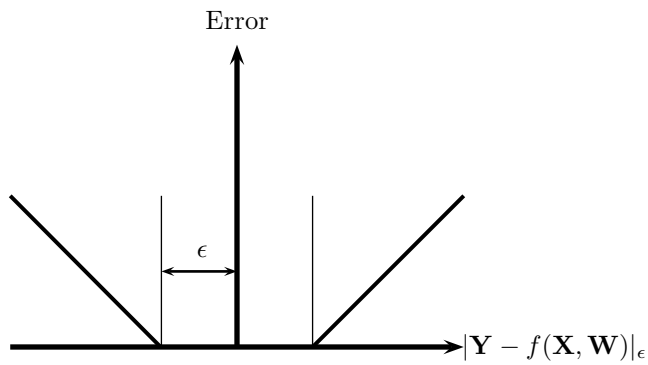

Fig. 2: The ε insensitive loss function.

$$|\mathbf{Y} - f(\mathbf{X}, \mathbf{W})|_\epsilon = \begin{cases} 0 & \text{if} |\mathbf{Y} - f(\mathbf{X}, \mathbf{W})| \leq \epsilon \\ |\mathbf{Y} - f(\mathbf{X}, \mathbf{W})| - \epsilon & \text{otherwise} \end{cases} \tag{17}$$

In SVR, the nonlinear regression function is obtained by minimizing both the empirical risk (R_{emp}) function and the weights simultaneously, i.e., regularized error function. This is known as structural risk minimization (SRM) and is represented in (18).

$$\begin{aligned} R &= \frac{1}{2}\|\mathbf{W}\|^2 + R_{emp} \\ &= \frac{1}{2}\|\mathbf{W}\|^2 + c \cdot \frac{1}{M}\sum_{i=1}^{M} |Y_i - K(X, X_i) - b|_\epsilon \end{aligned} \tag{18}$$

where $K(X, X_i)$ is a kernel function, M is the number of training images, and c is a non-negative regularization parameter that determines the tradeoff between the training error and the regularized weights. For example, a Gaussian Radial Basis Function (RBF) kernel is represented in (19).

$$K(X, X') = \exp\Big(-\frac{\|X - X'\|^2}{2\sigma^2}\Big) \tag{19}$$

where σ^2 defines the shape of the kernel function. There are many different admissible types of kernel functions [37]. The performance of the SVR depends upon the proper selection of parameters and kernel function. There is no defined theory on the selection of kernel functions. For details on optimization procedures to minimize the regression function ((18)) and kernel properties, we recommend the study of [37, 38].

Training SVR on a large data set is a cumbersome process as quadratic optimization becomes extremely difficult to solve using standard methods. However, there are many approaches to solve this issue [39, 40]. In comparison with neural networks, SVM obtains global minimization and achieves good generalization even on large data sets.

4.5 Gray World Algorithm

Gray world algorithm is one of the oldest and simplest color constancy algorithms [1]. It is still used as a benchmark for comparison for many color constancy algorithms. It assumes the illumination in each channel of the image averages to gray over the entire image. GW

based color constancy is achieved by computing the average of the pixel values in each channel of the image. This estimation is given by (20).

$$l_r^{GW} = mean(E_R), l_g^{GW} = mean(E_G), l_b^{GW} = mean(E_B) \tag{20}$$

where $l_r^{GW}, l_g^{GW}, l_b^{GW}$ values are simply average of pixels of each channel and E_R, E_G, E_B are the individual R, G, and B image channels.

4.6 Scale by Max Algorithm

Scale by max algorithm estimates the illuminant by measuring the maximum of the response in each channel.

$$l_r^{SB} = max(E_R), l_g^{SB} = max(E_G), l_b^{SB} = max(E_B) \tag{21}$$

The presence of specularities in images are often used to measure the illumination chromaticity.

5. Results

Following the discussions on machine learning algorithms, we now explain how they are used to estimate the illumination chromaticity. The training data set [11] consists of 2330 real images and the testing data set [12] consists of 320 calibrated real images. Both data sets are independent. Fig. 3 shows a few example images used for training. Fig. 4 shows the chromaticity space (r, g) plot of 11 illuminants under which 320 test images in [12] were taken. The procedure explained in Section 3 is used to obtain the training and test design matrices. We obtain the response vector for the training data set [11] using the GW algorithm based on the procedure and assumptions described in Section 3. The response vector for the test data set is known [12].

The ridge regression coefficients ($\widehat{\boldsymbol{\beta}}_{ridge}$) are computed using the training design matrix and corresponding response vector for an optimized γ value. Using the same $\widehat{\boldsymbol{\beta}}_{ridge}$, we estimate the illumination chromaticity of previously unseen test image. The optimal value of γ is obtained using k-fold cross validation from a range of feasible γ values. In k-fold cross validation, the training data set is split into 'k' parts. For a given γ value, we use 'k-1' parts for training and the remaining 1 part for testing. The root mean square error (RMS) is computed and process is repeated for all the 'k' parts. The average RMS error is computed for a given γ value. The same procedure is repeated for the entire range of γ values. The minimum average RMS error gives the optimal γ value. In the case of ridge regression, the optimal γ value of 3.8362 is obtained by k-fold cross validation.

In kernel regression, each column vector of the design matrix is selected sequentially as a query vector $\boldsymbol{q}$ and the kernel coefficient $\beta(\boldsymbol{q})$ is computed for an optimal bandwidth matrix $\mathbf{H}$. The kernel $k_H(\cdot)$ used in this paper is the Gaussian kernel function ((19)). The value of the kernel bandwidth affects the output estimate, in the sense that it over-fits or under-fits the data. Therefore, the optimal selection of the bandwidth is essential to obtaining the right model. When a single bandwidth h is used for the entire data set, it is referred as global bandwidth. There are many techniques [31] to perform optimal bandwidth selection. We select the global bandwidth empirically. During testing, the optimal bandwidth obtained during training is used to estimate the illumination chromaticity of unseen test images data set.

We use a neural network of $1024 - 10 - 2$ architecture to compute the illumination chromaticity. In the given architecture, 1024 represents the number of bins in each image, 10

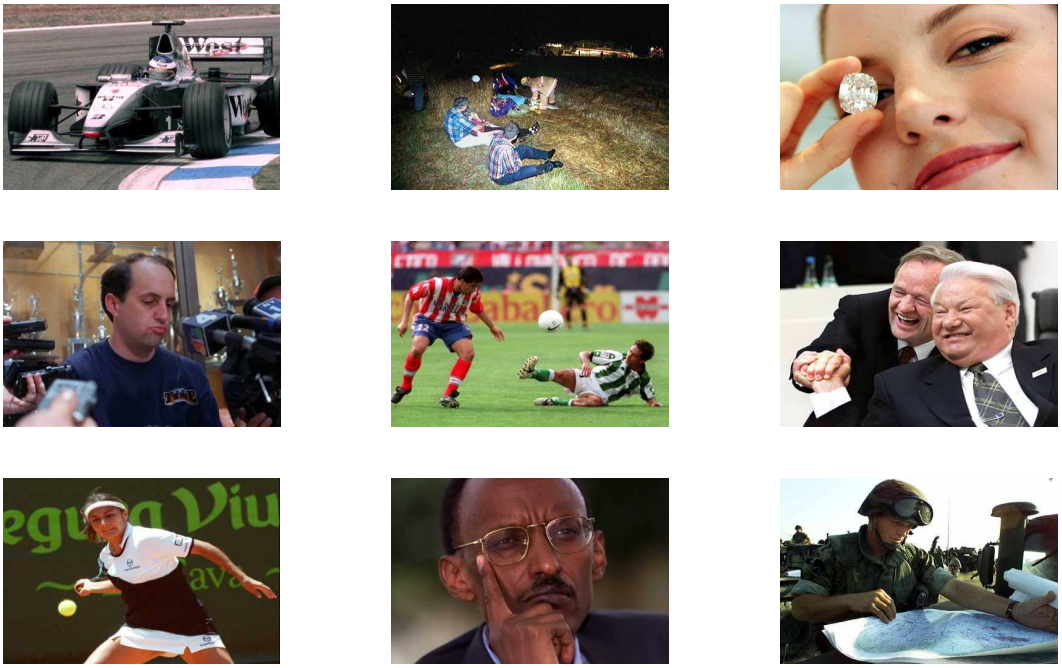

Fig. 3: A few example training images used for training purpose (from data set [11]).

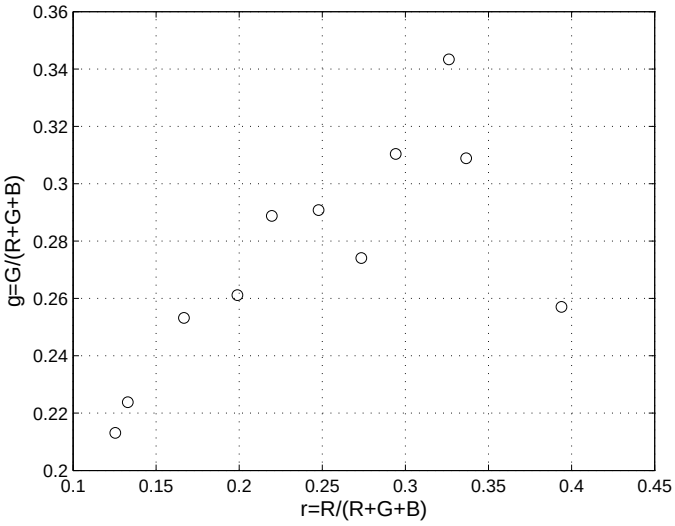

Fig. 4: A chromaticity space plot of 11 illuminants used to generate the test data set [12].

represent the number of hidden nodes in the single hidden layer, and 2 represents the output nodes corresponding to the chromaticity values r and g respectively. We experimented with different network architectures by varying the number of input nodes, i.e., sampling (100, 400, 1024, 1600, and 2500) and the number of nodes in the single hidden layer (10, 20, 30, 40, 50, 60, 70, 80, 90, 100). We observe that a network with a $1024-10-2$ architecture per-

Table 1: Optimal parameter values obtained by k-fold cross validation for each SVR.

SVR / Parameters	ε	σ^2	c
SVR_r	0.07	10	0.02
SVR_g	0.08	200	10

forms a better estimation of the illumination chromaticity when compared to other network architectures. The selection of the architecture is empirical and in accordance with the one used in [20]. We train the neural network with different training algorithms – conjugate gradient, resilient backpropagation, and other forms of conjugate gradient algorithms. Based on our experimental inference, we select the conjugate gradient algorithm for training as it performs better than other training algorithms on the same training data set.

The feature space of nonlinear SVR is defined by kernel functions as described in Section 4. Due to its performance in terms of algorithm convergence and robustness, we select the Radial Basis Function (RBF) kernel ((19)). Support vector machines are inherently defined for a single output; so two separate SVMs namely; SVR_r and SVR_g, corresponding to chromaticity r and g respectively are used to model the input-output functionality. Optimal selection of the free parameters (ε, σ^2, and c) is necessary for good minimization of the actual risk (R) in (18) and is performed using k-fold cross validation. The parameter value at which the RMS error is least is selected as the optimal value. Table 1 shows the optimized parameter values for each SVR. In our research, the C++ implementation of SVR suggested by Collobert et al. [41] is used as it can handle large data set.

Besides ML algorithms, we also apply GW and SBM algorithms to color correct test images and evaluate algorithms performance. The estimate of illumination chromaticity using each ML algorithm is compared with the actual chromaticity value using the mean RMS rg error,

$$\left(\frac{1}{M}\sum_{i=1}^{M}\frac{1}{K}\sum_{k=1}^{K}\left(e_i^{(k)}-t_i^{(k)}\right)^{1/2}\right) \tag{22}$$

where M is the number of image pixels, K is the number of channels ($K = 2$ for the chromaticity space and $K = 3$ for the RGB space), $e_i^{(k)}$ is the estimated chromaticity of image i corresponding to the image channel k and $t_i^{(k)}$ is the analagous quantity of the actual image's chromaticity. Note, during training the actual chromaticity t is not available and is estimated using the gray world assumption and that during testing we used the actual chromaticity as measured by Barnard et al. [12]. The error measure ((22)) considers the error in the final color constancy results, which is the difference between the corrected image, and the exact target image taken under the canonical illuminant. In the case of ML algorithms, $K = 2$. For fair comparison with GW and SBM algorithms, the estimated illumination by these methods is converted into chromaticity space, i.e., $K = 2$. Given the chromaticity estimation by ML algorithms, the diagonal model [13] is used to color correct the images. The intensity of the image pixels is adjusted such that the average intensity of the image remains constant. Fig. 5 is an example showing the performance of the various algorithms, where Fig. 5(a) is the target image and Fig. 5(b) is the image to be color correct. Fig. 5(c)-Fig. 5(h) illustrates the performance of each algorithm on Fig. 5(b). Table 2 shows the mean RMS rg chromaticity error computed using (22). Table 2 contains also the error value computed when no correction (NC) is performed, i.e., the output image is the same

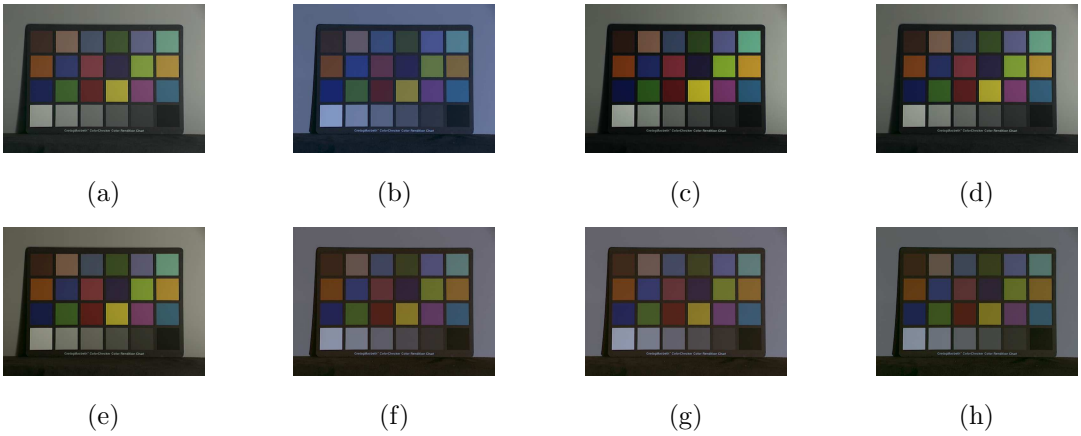

Fig. 5: An example showing the color correction using each algorithm on the "Macbeth" real image. (a) Target image, (b) input image, (c) ridge regression, (d) kernel regression, (e) support vector machine, (f) neural network, (g) gray-world, and (h) scale by max.

Table 2: Mean RMS rg error of all the calibrated images using each algorithm.

Algorithms	Mean RMS rg error
No correction (NC)	0.124
Kernel regression	0.052
Ridge regression	0.060
Support vector regression	0.066
Neural networks	0.071
Gray world	0.072
Scale by Max	0.077

as the input image. The mean RMS rg error value of 0.052 obtained by kernel regression in Table 2 signifies that the difference between the estimated and the target chromaticity values is 5.2%. During the diagonal transformation, we assume that the average intensity of the target and the corrected image is the same. This implies that the final color constancy result, i.e., the difference between the corrected image, and the exact target image taken under the canonical illuminant is 5.2%. From the example in Fig. 5 and mean RMS rg error in Table 2, it is observed that linear ML algorithms perform better than the other algorithms.

Both SVR and ridge regression are based on structural risk minimization and achieve a global optimization. Opposite to this, the neural network solution is based on ERM and can be trapped in local minima. Ridge regression can be considered as a special case of SVR, if the RBF kernel is replaced with the linear kernel in (18). Funt et al. [26] studied RBF and polynomial kernels, but they did not include linear kernels into their investigation.

Kernel regression is a linear technique that can work with nonlinear data. The ability of the kernel regression to handle nonlinearity is possibly the reason that it performs better than the ridge regression which cannot handle nonlinearity in the data. Both SVM and kernel regression are kernel based approaches. The choice of kernel determines the output. However, if the data set is large, kernel regression becomes independent of the choice of the kernel function [31]. To present a fair comparison, Gaussian kernel is used in both

Table 3: Variance obtained during uncertainty analysis.

Chromaticity Coordinates / Algorithms	r	g
Neural Networks	$2.5e-03$	$3.5e-03$
Support Vector Machines	$6.3e-06$	$4.6e-06$
Ridge Regression	$4.2e-04$	$3.9e-04$
Kernel Regression	$5.1e-07$	$7.5e-06$

kernel regression and SVM. Kernel regression requires only optimal selection of the shape of the kernel function when compared to SVM which have more parameters to be selected optimally in order to obtain good generalization accuracy.

6. Uncertainty Analysis

Uncertainty analysis is an important measure to evaluate the performance of data-driven algorithms. We perform this analysis to evaluate the consistency of the ML algorithms in estimating the illumination chromaticity. In the past, publications using ML approaches to color constancy problem [20]-[22], [26] did not include similar analysis, except in [10].

In the case of neural networks, the random initial start, training goal selection, architecture selection, and selection of training algorithms are the factors that contribute to uncertainty. Because of the random initial start in neural networks, the uniqueness of solution is not guaranteed even if other factors are selected carefully and fixed. Such uncertainty due to random initial start does not exist in the case of SVR, RR, and KR. In this analysis, we bootstrap the training data set. The bootstrapping is performed by randomly selecting the training images from the sample space of 2330 images to generate new training data set of same size. Since the process is based on random selection, some of the images in the new data set may be repeated and some may be missing. This process is repeated to generate 100 new data sets of images. These new data sets are subsets of the original training data set.

The images in these bootstrapped data sets are converted into 2D binary chromaticity histograms and design matrices are obtained as described in Section 3. Thus, we have 100 design matrices. All the four ML algorithms are trained on each design matrix and parameters are selected as discussed earlier. As a result, we obtain 100 trained models per algorithm. The illumination chromaticity of test data set is estimated using all the 100 trained models per algorithm. The probability density function (PDF) of the estimated chromaticity values and the variance of distribution of each ML algorithm is computed. For illustration purpose, we present the PDF and variance values obtained in the chromaticity space for a single test image in Fig. 6 and Table 3 respectively. From the plots in Fig. 6 and variance values presented in Table 3, we observe that SVR, RR, and KR are more consistent and have smaller variances in their estimation of illumination chromaticity when compared to neural network estimation.

An important aspect of this analysis is that we are not concerned with the amount of variability the bootstrapping produces in the training data set. Rather, if any variations occur in the training data set, all the data driven algorithms are to be retrained to account for those variations. This analysis helps us understand how well the trained model accommodates the variations, and the variance gives the amount of deviation in the estimation. The large variance in the case of a neural network may lead to an unacceptable estimation

of chromaticity. This also supports the fact that NN performs local minimization and the uniqueness of the solution cannot always be guaranteed in real-time applications.

7. Conclusions

In this paper, we evaluated the performance of linear regression tools, ridge regression and kernel regression in estimating the illumination chromaticity. The lowest mean RMS rg error of 0.052 is achieved by kernel regression where nonlinear learning technique achieves an error of 0.071 on the same data set which states an improvement of 26% over nonlinear techniques. The accuracy of estimation of linear techniques provides better color constancy when compared to nonlinear ML algorithms, gray-world, and scale by max. In our evaluation of linear regression tools, we have shown that they have the ability to learn the association between the image color and illumination color.

The results showed that by comparing two linear regression tools, kernel regression performed better than ridge regression and other nonlinear ML techniques on the same data sets. These algorithms performed also better than GW and SBM. The uncertainty analysis of ML algorithms is performed for the first time in this paper and we showed that RR, KR, and SVR are more stable than neural networks in estimating the illumination chromaticity. The ability of RR and KR to be solved analytically and the linear nature makes them computationally simpler compared to SVR. These features provide linear ML approaches with a significant advantage over SVR in real time color constancy.

Acknowledgments

The authors like to thank the computer vision laboratory at Simon Fraser University for making the color image dataset publicly available. Parts of this work were supported by the University Research Program in Robotics under grant DE-FG52-2004NA25589 by DOE.

References

[1] G. Buchsbaum, A spatial processor model for object color perception. *Journal of Franklin Institute*, 310, 1-26, 1980.

[2] L. T. Maloney and B. A. Wandell, Color constancy: A method for recovering surface reflectance. *Journal of Optical Society of America A*, 3(1), 29-33, 1986.

[3] G. D. Forsyth, A novel algorithm for color constancy. *International Journal of Computer Vision*, 5(1), 5-36, 1990.

[4] M. D'Zmura and G. Iverson, Color constancy I. Basic theory of two stage linear recovery of spectral descriptions for lights and surfaces. *Journal of Optical Society of America A*, 10(10), 2148-2165, 1993.

[5] B. V. Funt and M. S. Drew, Color space analysis of mutual illuminationspectral descriptions for lights and surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2), 319-1326, 1993.

[6] G. H. Finlayson, Color in perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10), 1034-1038, 1996.

[7] S. Tominaga, Surface reflectance estimation by the dichromatic mode. *Color Research and Application*, 21(2), 104-114, 1996.

[8] D. H. Brainard and W. T. Freeman, Bayesian color constancy. *Journal of Optical Society of America A*, 14(7), 1393-1411, 1997.

[9] J. Hadamard, Lectures on Cauchy's Problem in Linear Partial Differential Equations, Yale University Press, New Haven, 1953.

[10] V. Agarwal, A. V. Gribok, and M. A. Abidi, Machine learning approach to color constancy. *Neural Networks*, 20(5), 559-563, 2007.

[11] C. Rosenberg, M. Herbert, and A. Ladsariya, Bayesian color constancy with non-Gaussian models. In *Proceedings of NIPS*, 2003.

[12] K. Barnard, L. Martin, B. Funt, and A. Coath, A data set for color research. *Color Research and Application*, 27(3), 147-151, 2002.

[13] G. H. Finlayson, M. S. Drew, and B. V. Funt, Color constancy: Generalized diagonal transforms suffice. *Journal of Optical Society of America A*, 11(11), 3011-3020, 1994.

[14] G. Finlayson, S. Hordley, and P. Hubel, Color by correlation: A simple unifying framework for color constancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11), 1209-1221, 2001.

[15] V. Agarwal, B. R. Abidi, A. Koschan, and M. A. Abidi, An overview of color constancy algorithms. *Journal of Pattern Recognition Research*, 1(1), 42-54, 2006.

[16] K. Barnard, V. C. Cardei, and B. V. Funt, A comparison of computational color constancy algorithms - Part I: Methodology and experiments with synthesized data. *IEEE Transactions on Image Processing*, 11(9), 972-983, 2002.

[17] K. Barnard, L. Martin, A. Coath, and B. V. Funt, A comparison of computational color constancy algorithms - Part II: Experiments with image data. *IEEE Transactions on Image Processing*, 11(9), 985-996, 2002.

[18] B. Funt, V. Cardei, and K. Barnard, Learning color constancy. In *Proceedings of IS&T/SID 4th Color Imaging Conference: Color Science, Systems, and Applications*, Scottsdale, AZ, 58-60, 1996.

[19] S. Tominaga and B. A. Wandell, Standard surface-reflectance model and illuminant estimation. *Journal of Optical Society of America A*, 6(4), 576-584, 1989

[20] V. Cardei, B. V. Funt, and K. Barnard, Estimating the scene illumination chromaticity using a neural network. *Journal of Optical Society of America A*, 19(12), 2374-2386, 2002.

[21] V. Cardei, B. V. Funt, and K. Barnard, Modeling color constancy with neural networks. In *Proceedings of International Conference on Vision, Recognition, and Action: Neural Models of Mind and Machine*, 29-31, 1997.

[22] B. V. Funt and V. Cardei, Bootstrapping color constancy. In *Proceedings of SPIE, Electronic Imaging IV*, 3644, 1999.

[23] M. J. Allman, and R. M. Goodman, A real time neural system for color constancy. *IEEE Transactions on Neural Networks*, 2(2), 237-247, 1991.

[24] Nayak and S. Chaudhuri, Self-induced color correction for skin tracking under varying illumination. In *Proceedings of International Conference on Image Processing*, 1009-1012, 2003.

[25] R. Stanikunas, H. Vaitkevicius, and J. J. Kulikowski, Investigation of color constancy with a neural network. *Neural Networks*, 17(3), 327-337, 2004.

[26] B. V. Funt and W. Xiong, Estimating illumination chromaticity via support vector regression. In *Proceedings of 12th Color Imaging Conference: Color Science and Engineering Systems and Applications*, 47-52, 2004.

[27] G. Schaefer, S. Hordley, and G. Finlayson, A combined physical and statistical approach to color constancy. In *Proceedings of IEEE Computer Science Conference on Computer Vision and Pattern Recognition*, 148-153, 2005.

[28] R. Manduchi, Learning outdoor color classification. *IEEE Transactions on Pattern Recognition and Machine Learning*, 28(11), 1713-1723, 2006.

[29] M. Eber, Evolving color constancy. *Pattern Recognition Letter*, 2(11), 1220-1229, 2006.

[30] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55-67, 1979.

[31] C. G. Atkeson, A. W. Moore, and S. Schaal, Locally weighted learning. *Artificial Intelligence Review*, 11, 11-73, 1997.

[32] M. P. Wand and M. C. Jones, *Kernel Smoothing*, CRC Press LLC, Florida, 2000.

[33] C. M. Bishop. *Neural Network for Pattern Recognition*. Oxford University Press, Oxford, 1996

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, in Parallel Data Processing, D. Rumelhart and J. McClelland, (Ed.), 1, chapter 8, Cambridge, MA: MIT Press 1986 pp. 318-362.

[35] V. N. Vapnik, *Statistical Learning Theory*, John Wiley and Sons, Inc., NJ, 1998.

[36] V. N. Vapnik, S. Golowich, and A. Smola, Support vector method for function approximation, regression estimation, and signal processing, in: Advances in neural information processing systems, 9, Cambridge, MA: MIT Press, 1997.

[37] N. Cristianini and J. S. Taylor. *An Introduction to Support Vector Machines and Other Kernel Based Learning Methods*. Cambridge University Press, 2000.

[38] A.J. Smola and B. Scholkopf. A Tutorial on Support Vector Regression. Neuro COLT2 Technical Report Series, 1998.

[39] E. Osuna, R. Freund, and F. Girosi, An improved training algorithm for support vector machines, in color constancy. In *Proceedings of IEEE Workshop on Neural Networks for Signal Processing VII*, J. Principe, L. Giles, N. Morgan, and E. Wilson (Ed.), 276-285, 1997.

[40] J. C. Platt, Fast training of support vector machines using sequential minimal optimization, in Advances in Kernel Methods, B. Scholkopf, C. Burges, and A. Smola, (Ed.), Cambridge, MA: MIT Press, 1998.

[41] R. Collobert and S. Bengio, Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1, 143-160, 2001.

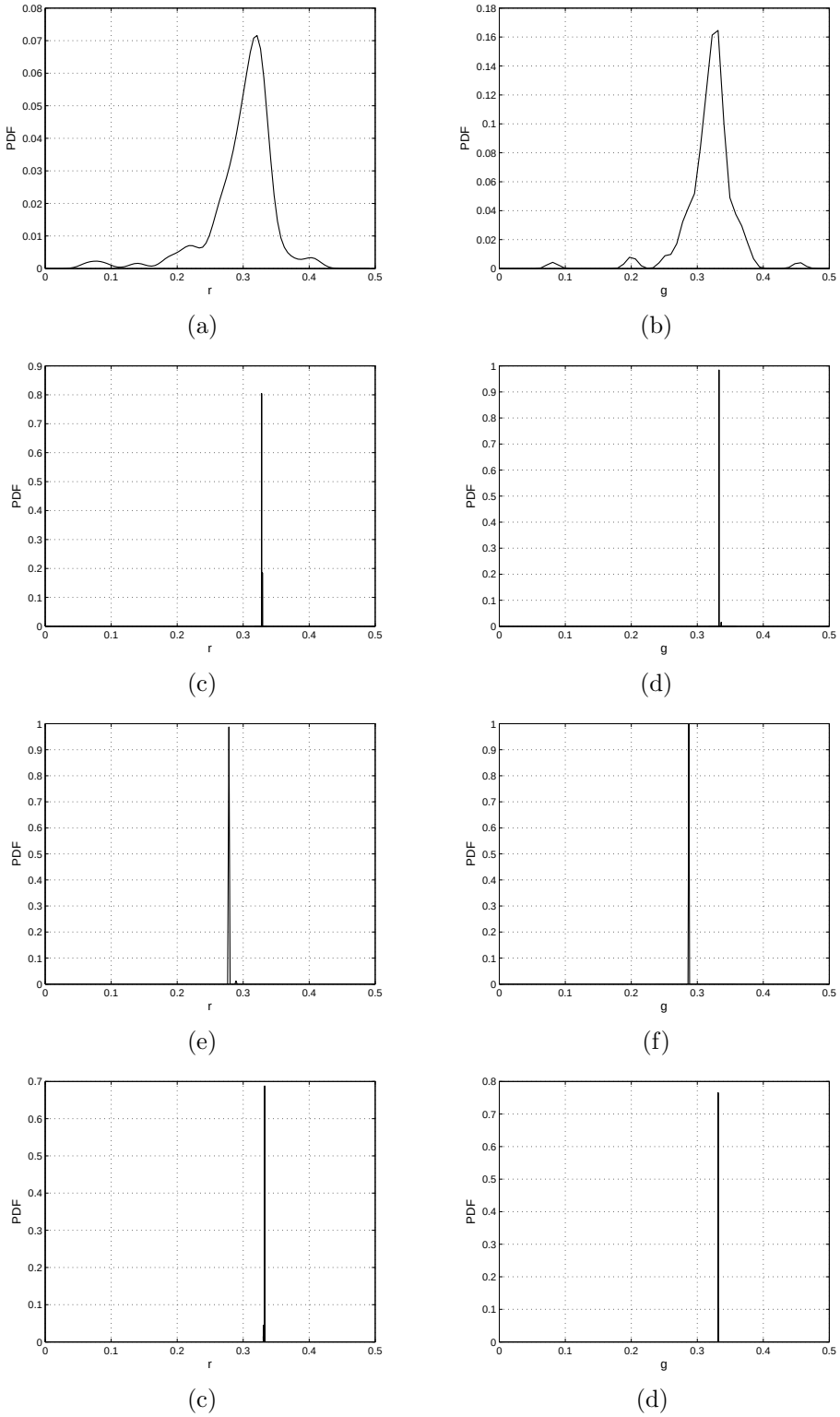

Fig. 6: Probability density function (PDF) of the estimated illumination chromaticity of a single test image using bootstrapped training data set. (a) - (b) Neural networks, (c) -(d) Support vector regression, (e) - (f) Ridge regression, and (e) - (f) Kernel regression.