Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Supervised feature selection for linear and non-linear regression of L*a*b* color from multispectral images of meat

Sara Sharifzadeh [a,*], Line H. Clemmensen [a], Claus Borggaard [b], Susanne Støier [b], Bjarne K. Ersbøll [a]

[a] Department of Informatics and Mathematical Modeling, Technical University of Denmark, Lyngby, Denmark
[b] Danish Meat Research Institute, Roskilde, Denmark

ARTICLE INFO

Article history:
Received 17 December 2012
Received in revised form
23 July 2013
Accepted 9 September 2013
Available online 17 October 2013

Keywords:
L* a* b* color space
Multispectral imaging
Sparse regression
Artificial neural networks
Support vector machine
Supervised feature selection

ABSTRACT

In food quality monitoring, color is an important indicator factor of quality. The CIELab (L*a*b*) color space as a device independent color space is an appropriate means in this case. The commonly used colorimeter instruments can neither measure the L*a*b color in a wide area over the target surface nor in a contact-less mode. However, developing algorithms for conversion of food items images into L*a*b color space can solve both of these issues. This paper addresses the problem of L*a*b color prediction from multispectral images of different types of raw meat. The efficiency of using multispectral images instead of the standard RGB is investigated. In addition, it is demonstrated that due to the fiber structure and transparency of raw meat, the prediction models built on the standard color patches do not work for raw meat test samples. As a result, multispectral images of different types of meat samples (430–970 nm) were used for training and testing of the L*a*b prediction models. Finding a sparse solution or the use of a minimum number of bands is of particular interest to make an industrial vision set-up simpler and cost effective. In this paper, a wide range of linear, non-linear, kernel-based regression and sparse regression methods are compared. In order to improve the prediction results of these models, we propose a supervised feature selection strategy which is compared with the Principal component analysis (PCA) as a pre-processing step. The results showed that the proposed feature selection method outperforms the PCA for both linear and non-linear methods. The highest performance was obtained by linear ridge regression applied on the selected features from the proposed Elastic net (EN) -based feature selection strategy. All the best models use a reduced number of wavelengths for each of the L*a*b components.

1. Introduction

Monitoring the quality of meat products is a significant concern in the food industry. Supplying a consistent high quality product requires a continuous assessment in the meat industry. This requires a development of on-line inspection methods for automation of the inspection process (Sharifzadeh et al., 2012). Conventional assessment methods in this case are based on subjective visual judgment and laboratory tests which are time-consuming, destructive and inconsistent in terms of human accuracy.

The visual appearances such as the texture pattern and the color of the meat are the main criteria for both the manufacturer and customer. These parameters are linked to the chemical properties such as the water-holding capacity, intra-muscular (marbling) and protein content (Sun, 2010). As a result, surface color is an important parameter for quality measurement in the meat industry.

One efficient color space for quantification of food items is the CIELab or L*a*b* color space, due to its precise characteristics (Mendoza et al., 2006; Brewer et al., 2006). It is a device independent color space defined by the International Commission on Illumination – abbreviated as CIE in 1976. L*a*b* has a perceptually equal space. This means that the Euclidean distance between two colors in the CIELab color space is strongly correlated with the human visual perception (Tkalčič and Tasič, 2003). The L* is the luminance component and the a* and b* are chromatic components.

Colorimeters and spectrophotometers are traditional instruments for measurements of colors such as L*a*b* in the food industry. They provide a quantitative measurement in a similar way to the human eye (Wu and Sun, 2013; Balaban and Odabasi, 2006). Colorimeters, such as the Minolta chromameter or the Hunter Lab, are used to measure the color of primary radiation sources that emit light and secondary radiation sources that reflect or transmit external light (León et al., 2006). Therefore, color values are obtained optically but not mathematically. Before doing the measurements, the instrument is usually calibrated.

* Corresponding author. Tel.: +45 50361337; fax: +45 45882673.
E-mail addresses: sarash@dtu.dk, sarasharifzade@yahoo.com, sarash@dtu.dk (S. Sharifzadeh), lkhc@dtu.dk (L.H. Clemmensen), cbo@teknologisk.dk (C. Borggaard), sst@teknologisk.dk (S. Støier), bker@dtu.dk (B.K. Ersbøll).

0952-1976/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2013.09.004

Traditional instrumental measurements can only measure the surface of a sample that is uniform and rather small (Balaban and Odabasi, 2006). Hence, they cannot completely represent the surface characteristics especially when it is non-uniform and highly textured as is the case for meat. In order to have a global representation of the target surface, computer vision techniques can be used to quantify the color (Wu and Sun, 2013). This leads to the formation of a 3D map of L*a*b* color values. Such a map represents the spatial characteristics of the whole surface instead of a small area. Color space conversion techniques can be employed to transfer an image into the L*a*b* space with the desired numerical and visual specifications. Thereby, the images of the meat samples from other color spaces such as RGB or CMYK can be transferred into L*a*b* space. In this way, it is possible to convert each image pixel into L*a*b* and therefore, generalize the representation.

Reviewing the literature shows that, conversion to L*a*b* was mainly performed using RGB images. In Larrain et al. (2008) and Mendoza et al. (2006) standard sequential transformation into XYZ color space and then from XYZ to L*a*b* was used for RGB images of beef and vegetables respectively. In Fdhal et al. (2009), conversion for the RGB images of the standard color patches into L*a*b* was performed using BPANN.[1] In Cao and Jun (2011) and Cao and Jun (2008), RBFNN[2] and GRNN[3] were used for conversion from CMYK color space to CIELab respectively.

The use of RGB images has some drawbacks. An RGB image, captured by a digital camera, is formed by filtering the incoming photons into three broad primary channels representing the color variables; Red, Green and Blue (RGB). These three variables are enough to describe a color sensation. However, the intensity recorded in each channel is an integration over a large range of wavelengths and therefore, two objects with different spectral radiant power distribution may seem to have similar colors in an RGB image. This is called metameric failure, which means matching colorimetrically under one illumination, but differ under another. It occurs when the spectral radiant power distribution of two objects are different, but the rough splitting of photons fails to observe this Dissing et al. (2010). In addition, RGB is a device dependent color space and the color of an object may be slightly different in two different camera records.

Multispectral imaging is an alternative for solving these limitations. In a multispectral imaging system, the sampling frequency of the electromagnetic spectrum is high and images are formed in very narrow bands compared to the three broad intervals used in standard RGB imaging. Therefore, the distribution of incoming photons for each pixel is approximated correctly. Besides the visual bands that characterize the color information, the higher wavelengths such as NIR are related to the chemical characteristics. Therefore, spectral imaging has been widely used for food quality control applications (Gamal et al., 2009; Dissing et al., 2009; Sharifzadeh et al., 2013).

So far, multispectral imaging has never been used in color conversion of food items. Color conversion using the spectral images can be done based on statistical predictive models. The advantage of such methods over the standard matrix transformation was investigated in León et al. (2006). In that work, a sequential transformation was used for conversion of the RGB images of color samples into L*a*b*. In addition, OLS[4] linear regression and ANN[5] with early stopping generalization were employed and their results showed that the ANN model obtained the best performance. In Dissing et al. (2010), the multispectral images of the standard color patches were transformed into the CIE-XYZ using linear regression models.

[1] Back Propagation artificial neural network.
[2] Radial basis function neural network.
[3] Generalized regularized neural network.
[4] Ordinary least square.
[5] Artificial neural networks.

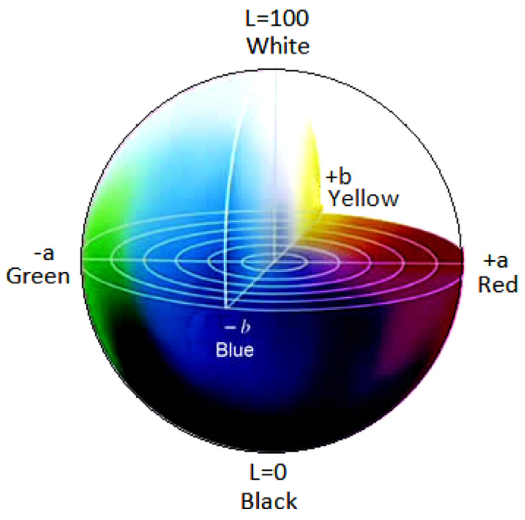

Fig. 1. L*a*b* 3D color space.

This paper focuses on conversion of multispectral images (430–970 nm) of different types of raw meat into L*a*b* units. In the following, we explain the main points investigated in this paper:

Since the food items can have variation, it is important to create and validate the prediction models on food products. Therefore, the use of real meat samples instead of the color patches for building the prediction models was investigated. Uncooked meat is translucent and transparent. Therefore the light reflected from it, not only comes from its surface but part of it comes from below the surface. Meat also has structure due to fibers with orientation. The color patches do not have structure and the light is reflected directly from the surface. Therefore, a model built on color patches do not work well on raw meat samples.

Due to the fact that the vision systems with their spectra are costly and not feasible to implement in the industry for online food productions, the sparsity is important and performing predictions using a minimum number of wavelengths would make the required vision system more cost efficient. Therefore, we propose a new supervised feature selection strategy based on EN and lasso[6] regression as a pre-processing step. The selected features were compared with PCA using three different regression strategies. A complete comparison between linear, non-linear and kernel-based regression methods was performed, which we did not see in the previous works. In order to have a general and fair judgment about the methods, the original data set was divided randomly into 25 training and test sets and the regression methods were tested on all of them and the average results were considered.

Finally, the results of the spectral images were compared with the RGB images.

The rest of the paper is organized as follows; Section 2 is about color description and Section 3 describes the data preparation. In Section 4, we describe linear, non-linear and kernel-based regression methods respectively. Section 5 is about the proposed supervised linear feature selection algorithm. Experimental results are presented in Section 6. Finally, there is a conclusion for this paper in Section 7.

[6] Least angle shrinkage and selection operator.

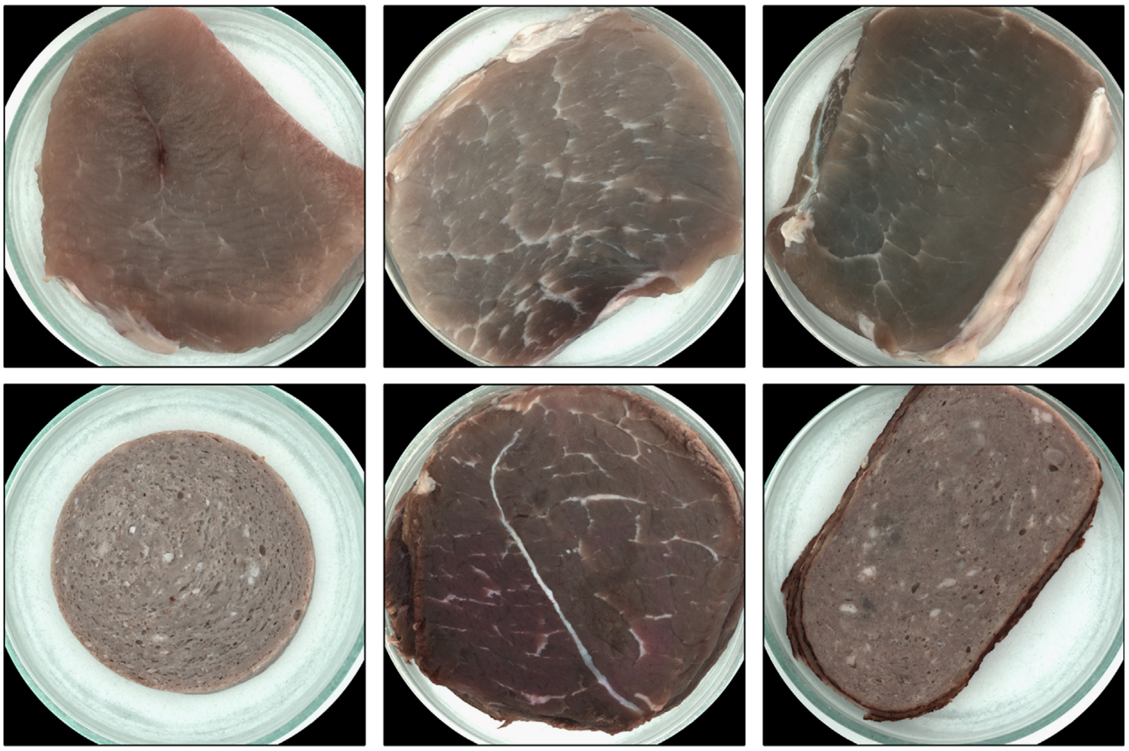
Fig. 2. Six different meat samples from the data set used in this paper.

2. Color description

In principle, there are two methods for describing color; the spectral and the tristimulus data description (X-Rite, 2004). Spectral data, describes the surface properties of the colored object. It demonstrates how the surface affects (reflects, absorbs, transmits, or emits) light. Conditions such as lighting changes, the uniqueness of each human viewer, and different rendering methods have no effect on these surface properties. In this paper, the multispectral images of meat are the input images.

The tristimulus data which is a 3D color space, describes the color of an object, as it appears to human eye or sensor, and as it would be reproduced on a device such as a monitor or printer. A CIELab color could be considered as a point in a 3D coordinate color space as shown in Fig. 1. On the other hand, RGB and CMYK color representation describe a color as three values that can be mixed to generate the color. In contrast to these color spaces, CIELab is device-independent, meaning that the range of colors in this color space does not depend on the characteristics of a particular device, or the visual skills of a specific observer or the lightening condition. In addition, the RGB and CMYK color spaces are much smaller than the range of colors that is visible to the human eye.

In this paper, the output color is CIELab which is a uniform and widely used color scale. In this color space, L* defines the lightness and ranges from 0 to 100; a* denotes the red/green value; and b* the yellow/blue value. The range of both chromatic components is between -128 and 128. This Color space resembles a three-dimensional space and uses rectangular coordinates based on the perpendicular yellow-blue, green-red and illumination axes as shown in Fig. 1.

3. Data preparation

The meat data for this work was provided by the Danish Meat Research Institute. Fig. 2 shows six different samples of meat from the used data set. In this data set, there were images of different types of turkey, chicken, beef, veal and pork.

In order to prepare the reference L*a*b* measure, two Minolta Chroma Meters CR300 and CR400 were used. Each Minolta data was acquired at eight locations on each meat sample and the average and standard deviations of these readings were recorded. Then, the two Minolta results were averaged. The mean values were used as the reference L*, a* and b* for each sample. The average standard deviations will be used as a reference for evaluation of the accuracy of the prediction models.

Totally, we used 52 meat samples which were divided randomly into training and test sets 25 times. In each data set, the number of training samples were 38 which were used for building the models and the remaining 14 samples were kept as unseen data for the test step.

For each meat sample, multispectral images were acquired at 20 different wavelengths ranging from 430 to 970 nm using a VideometerLab. VideometerLab is a multispectral imaging device.[7] A sample is placed inside an integrating sphere. On top of the sphere, there is a camera which achieves a uniform and reproducible illumination. The illuminating diodes achieve the same level of intensity in all bands. They were calibrated radiometrically as well as geometrically to obtain the optimal dynamic range for each LED as well as to minimize distortions in the lens and thereby pixel-correspondence across the spectral bands. The optimal light condition avoids shadows and specular reflections (Dissing et al., 2009).

To form the feature vectors from the multispectral images, a Region of Interest (ROI) of size 200×200 pixel was selected from each sample image. In the next step, the pixel gray levels in each ROI were averaged at each wavelength. Therefore, we finally have 20 features per meat sample. The feature matrix is $X_{N \times P}$, where N denotes the number of samples and P is the number of wavelengths. The three output components are $L_{N \times 1}, A_{N \times 1}, B_{N \times 1}$.

[7] http://www.videometer.com

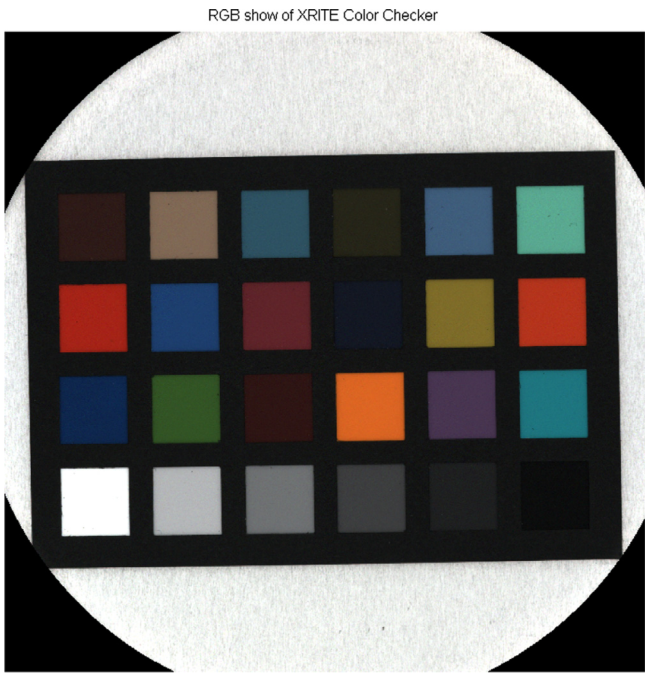

Fig. 3. The X-Rite color checker.

For ease of notation, we consider each of them as Y in the following sections.

One important point about the data set is that, we did not know the regions, where the measurements were performed. This means that, there is a deviation or mismatch between the regions from which, X is formed and the regions that Y values were measured.

In order to conduct the comparative experiment with the color checker, the standard X-Rite color checker was used. As shown in Fig. 3, it has 24 squares of colors in an 4×6 array. The multispectral images of this color checker were prepared in exactly the same wavelengths and light settings as the meat samples and the data set was formed in the same way. It has 24 samples in 20 wavelengths. The reference L*a*b* values of each color patch in the color checker is known.

4. Methods

In this section three regression strategies namely linear, non-linear and kernel-based methods that were used in this paper are explained. Due to the limited number of samples, a fivefold CV was applied on the training data for the optimal choice of model parameters in all the methods.

4.1. Linear regression

To convert the pixel intensities in the multispectral images into L*a*b* units, we can simply use the unbiased OLS model $\hat{Y}=X\hat{\beta}_{ols}+\varepsilon$, where ε is i.i.d. noise. However, since it is highly probable that some wavelengths have higher correlation to some of the output components (L*a*b*), selection and shrinking strategies can be useful.

One simple regularization method is ridge regression which uses the L_2 norm penalty to shrink some of the regression coefficients. This decreases the variance of the outputs. Another efficient regularization method is PLS which selects directions or components based on both the variance in the co-variates and their correlation with the response (Hastie et al., 2008). If there is a lot of variation in X that has no connection to the variation of outputs and instead, the response is highly sensitive to the low variations of input, PLS can be a good solution. Therefore, we apply the PLS regression to improve the result, in the case such scenario exists in our data.

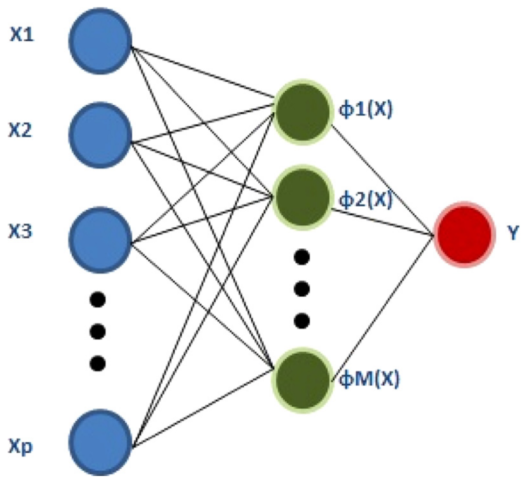

Fig. 4. The ANN diagram for regression with one hidden layer.

There are not necessarily prominent changes between images of all sequences of wavelengths and some of them are highly correlated. In this case a sparse solution such as lasso which uses the L_1 norm penalty can be employed:

$$\hat{\beta}_{lasso}=\operatorname{argmin}_{\beta}\left\{\frac{1}{2}\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{P}x_{ij}\beta_j\right)^2+\lambda\sum_{j=1}^{P}|\beta_j|\right\} \tag{1}$$

Here, β_j is the jth coefficient and λ controls the shrinkage rate. Another sparse regression method is EN. EN is in fact a compromise between lasso and ridge. Each regression coefficient is calculated as a weighted combination of ridge and lasso. EN selects variables like lasso, and shrinks together the coefficients of the correlated predictors like ridge (Hastie et al., 2008). The EN regression coefficients are computed by minimization of the following function:

$$\hat{\beta}_{EN}=\operatorname{argmin}_{\beta}\left\{\frac{1}{2}\sum_{i=1}^{N}\left(y_i-\beta_0-\sum_{j=1}^{P}x_{ij}\beta_j\right)^2+\lambda_1\sum_{j=1}^{P}|\beta_j|+\lambda_2\sum_{j=1}^{P}\|\beta_j\|,\right\} \tag{2}$$

where there are both L_1 and L_2 penalty terms. The sparse regression methods result in the use of less wavelengths. As mentioned before, this is important regarding the economical concerns.

4.2. Non-linear regression

ANN can be used as a non-linear regression solution. Fig. 4 shows the architecture of a simple ANN for regression with one hidden layer. First, M linear combinations of the input variables are built and then each combination is transformed using an activation function $h(.)$:

$$\phi_j(X)=h(\Sigma_{i=1}^{i=P}\alpha_{ij}x_k+\alpha_{0j}),\quad j=1,..,M \tag{3}$$

where α_{ij} is the weight parameter and α_{0j} is the bias. Then, the output $\hat{Y}$ is constructed as a linearly weighted combination of the non-linear basis functions $\phi_j(X)$:

$$\hat{Y}(X;\beta)=f\left(\sum_{j=1}^{M}\beta_j\phi_j(X)+\beta_0\right) \tag{4}$$

β_j and β_0 are the weight and bias parameters respectively, and $f(.)$ is an activation function which is usually, the identity function in the case of regression (Bishop, 2006).

Although this non-linear model is more complex and difficult to interpret, it may probably be more accurate for some types of data. Therefore, when there is no need for a detailed interpretation of the model, ANN may be a good solution which is the case for color conversion. The choice of basis function and the solution strategy for the weight parameters vary in different ANNs. In addition, the architecture of an ANN is also based on the number of hidden layers and neurons. As mentioned in Section 1, in many previous color conversion works, different types of ANN were used. Therefore, in this work, their application was investigated and compared with the linear methods.

4.2.1. *ANN modeling and parametrization*

One widely used ANN is the single hidden layer feed-forward ANN which uses a sigmoid basis function:

$$\phi_j(X) = \sigma_j(X) = \frac{1}{1+\exp(-S_j X)} \tag{5}$$

Here, S_j is the scale parameter which controls the activation rate. A large scale may cause hard activation around 0.

Another type is the RBFNN that uses a non-linear RBF[8] based on the Euclidean distance or Mahalanobis distance (like a Gaussian kernel function):

$$\phi_j(X) = \rho_j(\|X-\mu_j\|) \tag{6}$$

where μ_j is the center vector of the jth hidden node and $\rho$is the distance function. The RBFNN also has one hidden layer.

The parameters of the ANN models are commonly estimated by minimization of the sum of square function shown in Eq. (7), using the BP procedure (Hastie et al., 2008). This is a gradient descent process.

$$E(\beta) = \text{Min} \sum_{n=1}^{N} \|\hat{Y}(X_n;\beta) - Y\|^2 \tag{7}$$

BPANN is a well-known and widely used network and it has been used for color conversion problem as mentioned in Section 1. Although it is a powerful algorithm, it has some drawbacks. One important problem with the error function minimization for complex and flexible models is the over-fitting on training data and poor generalization. Because a complex model is more flexible in capturing the training data behavior. Other problems are slow convergence and the possibility that the network converges to a local minimum. The ANN algorithms are also sensitive to the initial points and it is recommended to restart the algorithm several times for this reason. We applied the simple BPANN as well as the generalized BPANN with early stopping on our data set. They were also used in León et al. (2006) and Fdhal et al. (2009), for conversion from RGB into L*a*b* units. Although they worked fine in some of the 25 random sets, the results were poor for most of them and the average results were not satisfactory. This is because of the above mentioned problems. Due to this oscillating and unstable behavior of BP, we employed other types of BPANN.

In the literature, there are ANNs that employ different strategies to overcome these problems (Bishop, 2006; Bishop and Tipping, 2003; Hagan et al., 1996). In this paper we applied some of these strategies and compared their results; The ANN with Adaptive learning rate and momentum term was tested to accelerate the convergence. In addition, different regularized ANNs were used to constrain the parameters. In the following, the tested ANNs will be explained in detail.

[8] Radial basis function.

4.2.2. *ANN with adaptive learning rate and momentum term*

Considering the error minimization in Eq. (7), the gradient $\nabla E(\beta)$ can be obtained by means of back-propagation of errors through the layers. This gradient is used in the family of gradient training algorithms which iteratively form:

$$\beta_{k+1} = \beta_k - \eta_k \nabla E(\beta^k), \quad k = 0, 1, 2, \ldots \tag{8}$$

where β_k is the current weight, $-\eta_k$ is the learning rate and k is the step number and $-\eta_k \nabla E(\beta^k)$ shows the search direction. The BP gradient-based training algorithms minimize the error function using the above gradient decent or steepest descent method with constant, heuristically chosen, learning rate.

The learning rate determines how fast a network will learn the relationships between input and output patterns. A smaller value of the learning rate means a slower learning process. In fact, the optimal learning rate changes during the training process, as the algorithm moves across the performance surface. Therefore, the performance of the steepest descent algorithm would improve, if the learning rate changes during the training process. An adaptive learning rate attempts to keep the learning step size as large as possible while keeping learning stable (Hagan et al., 1996).

The idea about using a momentum BP is to stabilize the weight change and smooth the osculation in the trajectory. Therefore, a fraction of the previous weight change $\Delta\beta^k$ is considered in updating of the current weights β^{k+1}. Acting like a low-pass filter, momentum allows the network to ignore small local minima in the error surface and slide through them. It also speeds the convergence because, when all weight changes are in the same direction, the momentum amplifies the learning rate.

$$\Delta\beta^{k+1} = \gamma\Delta\beta^k - (1-\gamma)\eta_k \nabla E(\beta^k), \quad k = 0, 1, 2, \ldots \tag{9}$$

where γ is the momentum coefficient and should be between 0 and 1. This gives the system a certain amount of inertia since the weight vector will tend to continue moving in the same direction unless opposed by the gradient term.

Both the BP with adaptive learning rate and BP with momentum term were applied on the 25 data sets.

4.2.3. *Regularization of ANN*

Feed-Forward ANN regularization: The simplest regularizer is the quadratic in which, a penalty term is added to the error function and penalizes the sum of weights toward zero similar to the regularization of the linear methods. The results of this method were acceptable on the validation sets and some of the test sets. However, the average test results were not satisfactory, showing very unstable and oscillating response on the different sets. This may happen due to the convergence in a local minimum.

These poor results will not be presented in this paper. Instead, the Bayesian regularization was used. It is an interesting approach which estimates the ANN parameters by a probabilistic approach (Bishop, 2006). Both the model output targets Y and parameters β are characterized as random variables with normal distributions. Then, the Bayesian rule is applied, to calculate their prior and posterior probabilities. Consequently, the predictive distribution of the output is obtained, using the sum and product rules for probabilities as shown in Eq. (10). For more details we refer to (Bishop, 2006; Bishop and Tipping, 2003).

$$P(\hat{Y}|X, Y_{tr}) = \int P(\hat{Y}|X,\beta).P(\beta|Y_{tr})\, d\beta \tag{10}$$

where, Y_{tr} denotes the data used for training the model. The averaging nature of the Bayesian method over many different possible solutions solves the over-fitting problem.

Another regularized ANN that was tested is the Nr_quadratic neural regressor with a quadratic cost function from DTU:toolbox (Kolenda et al., 2002). This is a two layer feed-forward ANN with a hyperbolic tangent non-linear functions for the hidden layer and linear output layer. The weights of the ANN are optimized with a MAP[9] approach and the quadratic error function is augmented with a Gaussian prior over the weights. An adaptive regularization is used to prevent the over fitting. For more information, we refer to the documents provided in Kolenda et al. (2002).

BPANN are sensitive to the number of neurons in their hidden layers. Too few neurons can lead to under fitting and too many neurons can cause over fitting. For this reason, for training of all the ANN algorithms described in Sections 4.2.2 and 4.2.3, loops are used for the best choice of the number of hidden nodes. Algorithm 1 shows the procedures used to train the ANN model. In each CV iteration, there is a loop on hidden nodes size. There is also another loop which repeats the training for each fold and each hidden node size several times. This will restart the network, training from different initial points and also helps to avoid falling in a local minimum. The output network from this algorithm will be used for the test data.

Algorithm 1. Training algorithm for ANNs described in Sections 4.2.2 and 4.2.3.

```
Inputs: Training data $(X_{tr}, Y_{tr})$
Initialization:
    • $HD$=vector of hidden neuron size
    • $Rep$=number of repetition times
    • Initialize the 5 fold indices
Algorithm:
1. For $cv$=1,…,5 repeat:
        Divide the inputs into training and validation sets
2. For $nhd$=1,…,$HD$ repeat:
3. For $rp$=1,…,$Rep$ repeat:
        Train the ANN with nhd number of hidden nodes
        Calculate the training error matrix $(HD \times Rep)$
End loops 2 and 3
        Find the vector of minimum training error $(1 \times HD)$
        Find their corresponding trained ANN
        Use these ANN to calculate the validation error $(cv \times HD)$
End loop 1
    • Find the minimum validation error
    • Find the corresponding ANN with best nhd
Output: Best trained ANN and validation error
```

RBFNN Regularization: For generalization of the RBFNN, the GRNN is used (Specht, 1991). In GRNN, the best prediction with minimum variance is obtained as the conditional mean value of Y_{tr} given X.

$$\hat{Y}(X) = E\langle Y_{tr}|X\rangle = \int_{-\infty}^{+\infty} Y_{tr} P(Y_{tr}|X)\, dY_{tr} \tag{11}$$

This could be calculated using the joint probability. GRNN uses a nonparametric approach to calculate the joint probability $P(X, Y_{tr})$ by a Gaussian isotropic kernel (Parzen window). The resulting probabilistic output is shown in Eq. (13). The numerator is the sum of the weighted training targets which contribute according to their joint probabilities with the input test sample, to form the output target. The denominator normalizes the solution.

$$\hat{Y}(X) = \frac{\int_{-\infty}^{+\infty} Y_{tr} P(X, Y_{tr})\, dY_{tr}}{\int_{-\infty}^{+\infty} P(X, Y_{tr})\, dY_{tr}} \tag{12}$$

$$\hat{Y}(X) = \frac{\sum_{i=1}^{N} Y_{tr}^{i} \exp\left(-\frac{D_i^2}{2\sigma^2}\right)}{\sum_{i=1}^{N} \exp\left(-\frac{D_i^2}{2\sigma^2}\right)} \tag{13}$$

where $D_i = (X - X_{tr}^i)^T (X - X_{tr}^i)$ and Y_{tr}^i, X_{tr}^i are the ith training sample values. σ is the standard deviation of the Gaussian kernel and is called the smoothing parameter. As can be seen from this equation, the contribution weights are in fact the Mahalanobis distance of the test input from the training samples. This means that the closer training samples will contribute more in the prediction of the output target. The smoothing parameter has great effect on the output prediction. With larger σ, more training data will contribute in the target output than with a small σ. In each CV iteration, we loop over different σ values and repeated the training like in Algorithm 1, for the proper choice of σ.

4.3. Kernel-based regression

SVM[10] was used as a kernel-based method for regression. SVM is characterized based on a maximum margin algorithm. Given the set of training data $\{(x_1, y_1), \ldots, (x_N, y_N)\}$, SVM finds a $f(x)$ function that has at most εdeviation from the actual target y. For this aim, the features are mapped to an M-dimensional feature space using non-linear basis functions $(h(x))$. Then, a linear model is constructed in this feature space:

$$f(x, \beta) = \sum_{m=1}^{M} \beta_m h_m(x) + \beta_0 \tag{14}$$

To estimate β_m and β_0, a new type of loss function called ε*−sensitive* loss function is used:

$$V_\varepsilon(r) = \begin{cases} 0 & \text{if } |r| < \varepsilon \\ |r| - \varepsilon & \text{otherwise} \end{cases} \tag{15}$$

The objective function to be minimized is as follows:

$$\min_{\beta, \beta_0} L(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2} \sum \beta_m^2 \tag{16}$$

The second term in Eq. (16) controls the complexity level of the model. This optimization leads to a kernel based solution:

$$\hat{f}(x) = h(x)^T \hat{\beta} = \sum_{i=1}^{N} \alpha_i K(x, x_i), \quad \hat{\alpha} = (HH^T + \lambda I)^{-1} Y \tag{17}$$

where $K(x, x_i) = \sum_{m=1}^{M} h_m(x) h_m(x_i)$. For more information, we refer to Hastie et al. (2008).

5. The proposed supervised linear feature selection

Feature selection can be used as a pre-processing step before all the explained methods. It helps to avoid over fitting by reducing the number of trainable parameters as much as possible.

Since the sparse linear regression methods perform both feature selection and regression together, it is not expected that a feature selection step improve their results. But, for non-sparse regression methods, it can be effective.

In the case of a feed-forward ANN, with a flexible number of hidden nodes, it is well known that the hidden layer can be regarded as taking the role of feature selection and dimension reduction. In in each CV iteration of Algorithm 1, the loop over the number of hidden nodes performs this selection properly. It has been demonstrated that CV is a successful model selection method (Shi and Xu, 2006). In addition, for ANN models, feature selection

[9] Maximum a posteriori.

[10] Support vector machine.

can be applied on the input variables $X_{N\times P}$ as a pre-processing step before the regression. It can be combined with any type of neural network.

One common dimension reduction method is PCA. It projects the variables orthogonally into a new space in which, they are sorted according to their variances. Therefore, it is possible to exclude features with low variance from the model. But, PCA is an unsupervised feature selection algorithm. This means that it does not consider the important information in the target values Y_{tr} and their dependencies to the training spectra X_{tr}. In addition, PCA is not a sparse feature reduction method. Because each principal component is a linear combination of all the variables.

However, according to the reasons described in Section 1, we are interested in using a minimum number of wavelengths. Although the sparse linear regression methods such as EN and lasso perform this, to improve the prediction results, we propose to use them for supervised linear feature selection. As described in Section 4.1, these methods will remove the redundant and irrelevant variables from the model, even with low or high variance. Algorithm 2 shows the different steps of our proposed supervised feature selection algorithm to form the reduced feature sets from EN.

In Algorithm 2, the vector *Freq* was used to record the number of times each wavelength had non-zero regression coefficients and *w* was the vector of wavelengths. EN regression was repeated 4 times on each of the 25 input training sets. The 4 repetitions were done to cancel the effect of randomness in CV loops. At the final iteration, the frequency of being non-zero for each of the 20 coefficients were obtained. The sorted *Freq* vector shows the top frequent non-zero coefficients. Their corresponding wavelengths could be found in the re-ordered version of the *w* vector according to the sorted *Freq*. At this step, the number of wavelengths, to be used as the final selected features were determined. For this aim, another iteration over all possible candidate numbers (1–20) were tested. In the case of higher number of wavelengths (when $N \ll P$) this can be reduced to a limited candidate list. The average RMSE[11] was considered as a criterion for the final decision. The best number of features among the 20 candidates corresponds to the one with the minimum RMSE (n). Finally, the selected wavelengths were used to form the new training and test feature matrices. The same algorithm was used for feature selection by lasso. These two method were compared with PCA.

Algorithm 2. The proposed algorithm for feature selection using EN.

```
Inputs: 25 sets of $(X_{tr}, X_{ts}, Y_{tr})$
Initialization:
  • $Freq$=vector of zeros $(1 \times 20)$
  • $W$=vector of the 20 wavelengths
Algorithm:
1. For all the 25 sets and for $rep=1,\ldots,4$ repeat:
     Compute $\beta_{EN}$ by training an EN regression model with 5 fold cv
     Add one to the $Freq$ elements with non-zero $\beta_{EN}$ coefficients
End loop 1
     Sort $Freq$ in descending order
     Re-order the corresponding elements in $W$ with respect to $Freq$
2. For $i=1,\ldots, 20$ repeat:
3. For all the 25 sets and for $rep=1,\ldots,4$ repeat:
     Compute the RMSE of regression on the training data using the corresponding first $i$ wavelengths of $W$
End loops 2 and 3
     Average the RMSEs over the 25 sets and 4 iterations
     Find the index of the minimum average RMSE among the 20. $(n)$
     Select the first $n$ top wavelengths from $W$, $(sel_{EN})$ and form the 25 sets of $Xtr_{EN}, Xts_{EN}$
Output: 25 sets of $(Xtr_{EN}, Xts_{EN})$
```

6. Experimental results

In this section, first the evaluation criteria for prediction models will be introduced. Then, we will show the results from the experiments on the X-Rite color checker. In the next step, the results of applying linear, non-linear and kernel-based models on all the spectral data will be presented. Then, we will show the results of the same models on the selected features from both our proposed method and PCA. Since there were many tables of results, only the box plots are illustrated here and the complete tables are presented in the appendix. The RGB images experimental results will be shown next and also an L*a*b* image will be formed. Finally, there will be a discussion.

6.1. Evaluation measures for prediction models

R-square (R^2), RMSE and ΔE measures are used for evaluation of the models.

R^2 is a statistical measure that shows the amount of data variation explained by a regression model. In order to calculate the R^2, RSS,[12] TSS[13] and ESS[14] are defined as follows:

$$RSS = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \quad TSS = \sum_{i=1}^{N} (y_i - \overline{Y})^2, \quad ESS = \sum_{i=1}^{N} (\hat{y}_i - \overline{Y})^2 \tag{18}$$

The most general definition of the (R^2) or coefficient of determination is

$$R^2 = \left(1 - \frac{RSS}{TSS}\right) \times 100 \tag{19}$$

In this definition, (R^2) is calculated based on the unexplained variance by the model or in other words the variance of the model's error.

RMSE shows the estimated standard deviation of the error and is calculated as follows:

$$RMSE = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2} \tag{20}$$

As mentioned in Section 3, the average standard deviation of the Minolta measurements can be used as a reference for evaluation of the prediction models. Table 1 shows the overall average of standard deviations for all the 14 samples in the 25 test sets. The estimated RMSE as the standard deviation of the prediction model, can be compared with these measured values.

[11] Root mean square error.

[12] Residual sum of squares.

[13] Total sum of squares.

[14] Explained sum of squares.

The delta error ΔE shows the color difference. A ΔE of 1 or less is not perceptible by human eye. A ΔE between 3 and 6 is typically considered as an acceptable match in commercial applications.

Table 1
The average of the standard deviations over the 25 test sets for L*a*b* components.

Component	The average standard deviation
L*	2.237
a*	1.115
b*	0.879

Table 2
The training and test results of the prediction model built on the color checker and meat data.

LOOCV-EN	Color checker model			Meat model		
	L*	a*	b*	L*	a*	b*
R^2_{tr}%	93.25	95.55	95.71	90.73	94.85	83.92
$RMSE_{tr}$	4.75	5.08	6.89	2.50	1.25	1.02
R^2_{ts}%	84.06	−482.42	−521.64	87.63	87.28	68.07
$RMSE_{ts}$	3.18	12.20	6.26	2.78	1.76	1.41
ΔE_{ts}		12.35			3.22	

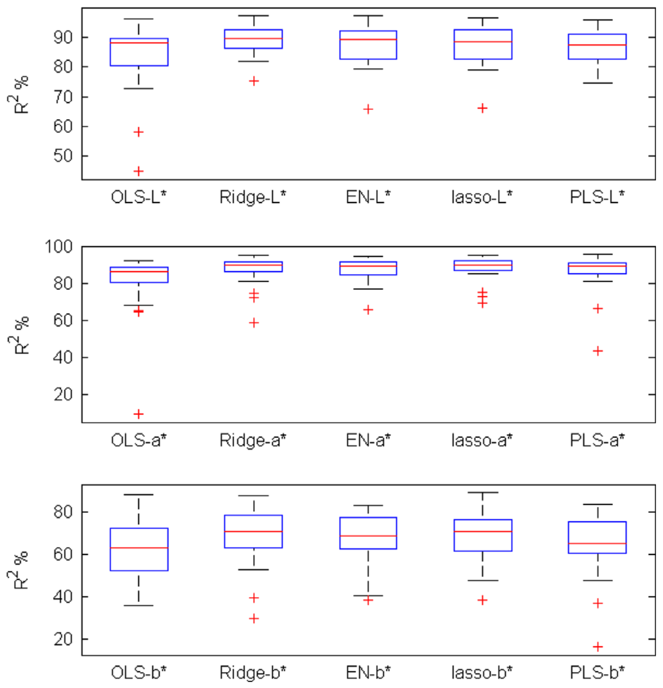

Fig. 6. R^2 box plots of the L*a*b* prediction for linear models on the 25 random test sets.

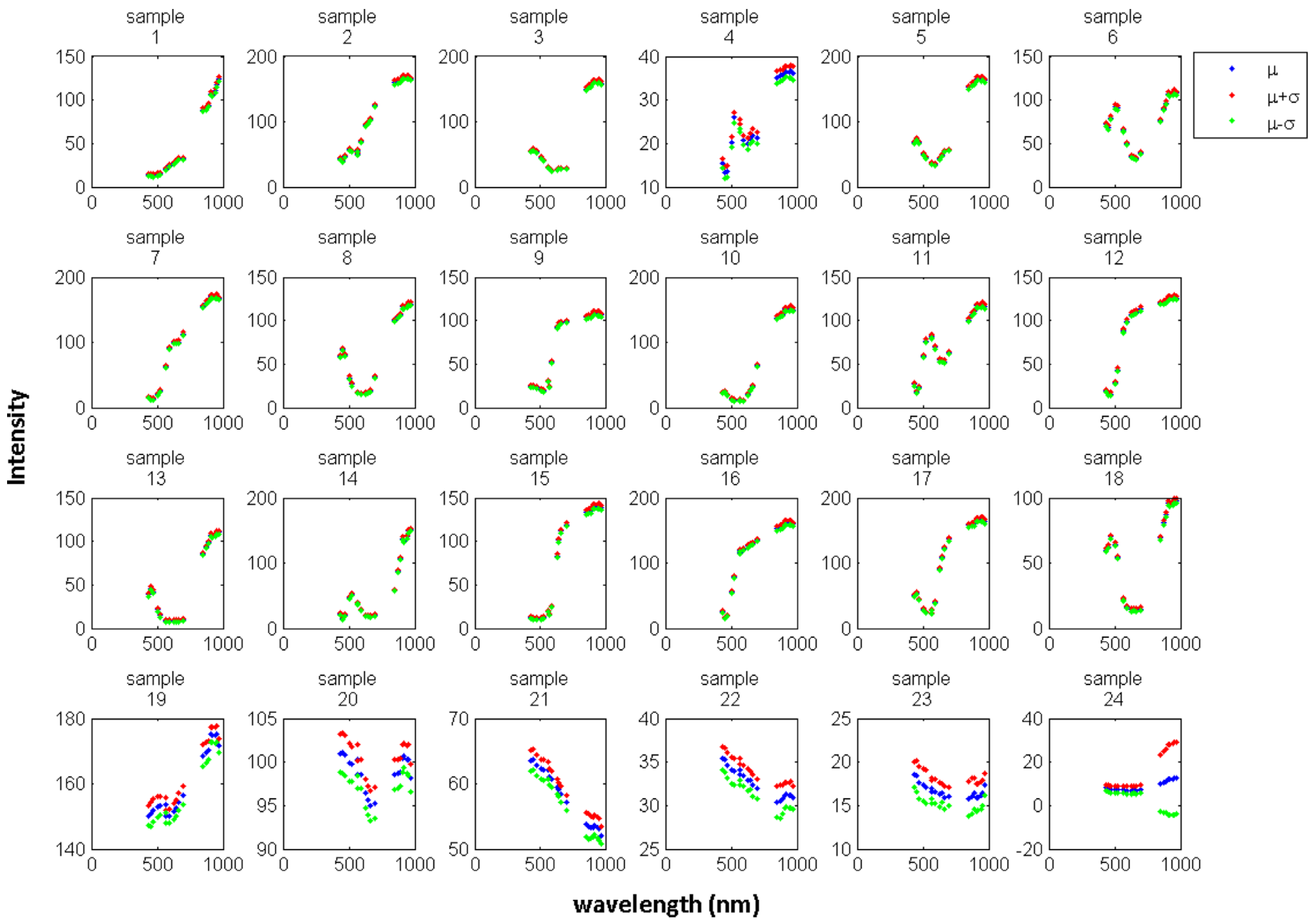

Fig. 5. The plot of the $(\mu \pm \sigma)$ in 20 wavelengths for the 24 ROI of the color patches. The horizontal axis shows the wavelength.

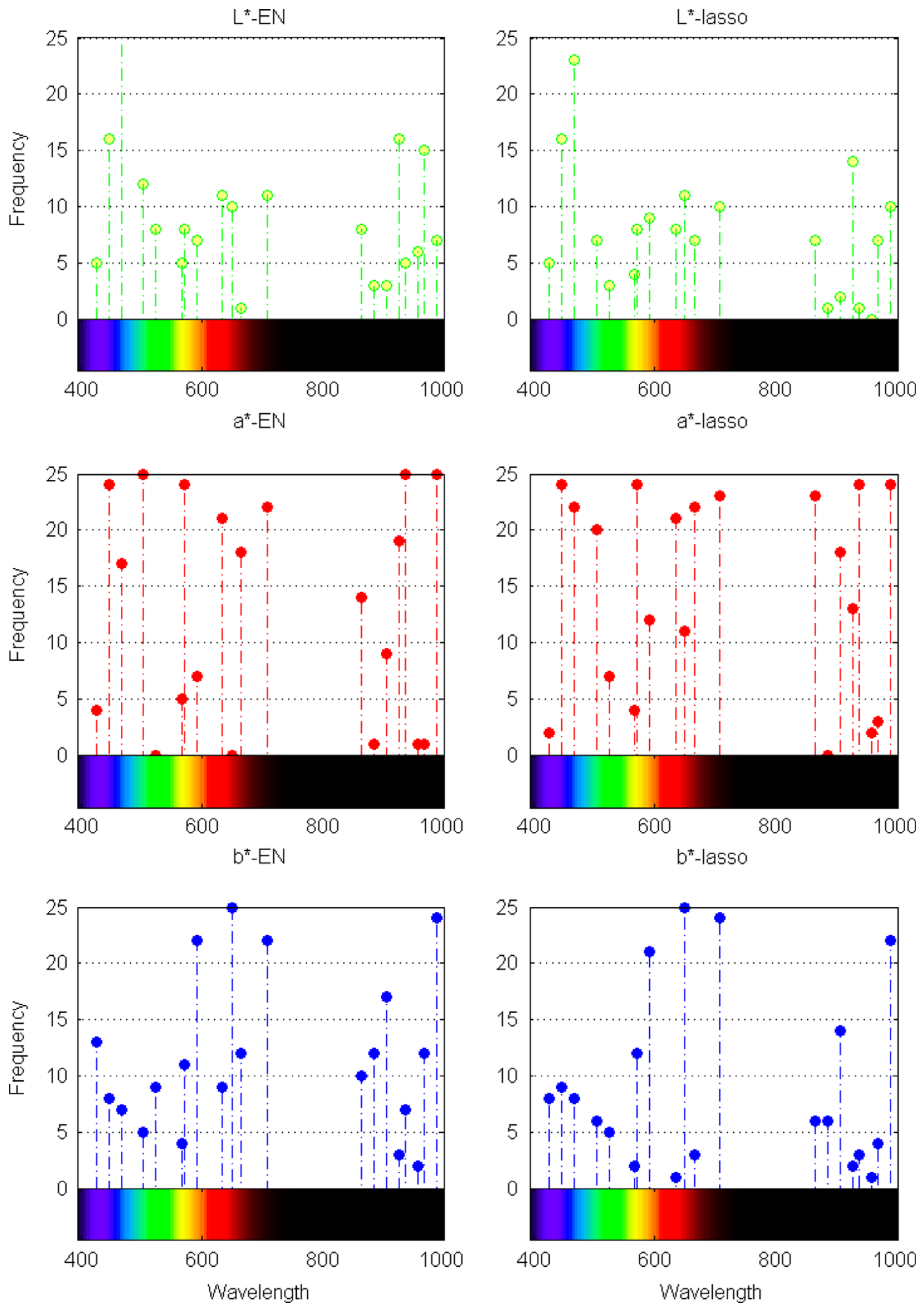

Fig. 7. The frequency map of the selected wavelengths by EN (left) and lasso (right).

Since the ΔE calculations are illuminant-dependent, calculations from colors viewed or measured under different illuminants are not comparable (Upton, 2006).

$$\Delta E = \sqrt{(L-\hat{L})^2 + (a-\hat{a})^2 + (b-\hat{b})^2} \tag{21}$$

6.2. Color checker test results

As described in Section 1, due to the transparency and texture structure of the raw meat, the use of multispectral images of meat may probably work better than the standard color checkers for color prediction. This was investigated by performing two experiments on the color checker data and meat samples.

In the first experiment, the color checker data was used for training a prediction model and in the second one, the 25 meat training sets were used. Then, they were applied for prediction on the respecting training sets as well as the 25 test sets. The average results were considered. The linear sparse EN regression was used to form the prediction model for L*a*b* color components. Since the color checker data had limited number of samples ($X_{24\times20}$), LOOCV[15] was used for both experiments on the color checker and meat data. This helps to have good generalization while finding the optimal model parameters. The results are presented in Table 2.

As can be seen, both models were capable of predicting on their own training data. But, the color checker failed to predict the color components for the meat data as expected regarding the physical characteristics of the raw meat. The negative R^2 shows the high

[15] Leave one out cross validation.

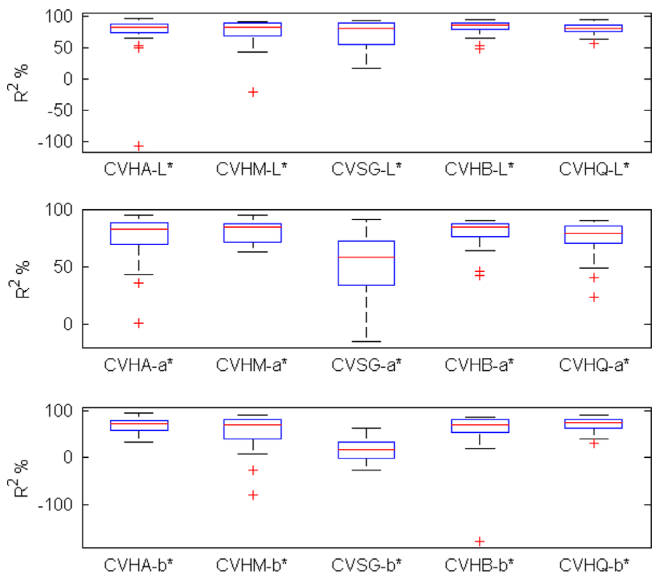

Fig. 8. R^2 box plots of the L*a ∗b ∗ prediction for non-linear models on the 25 random test sets.

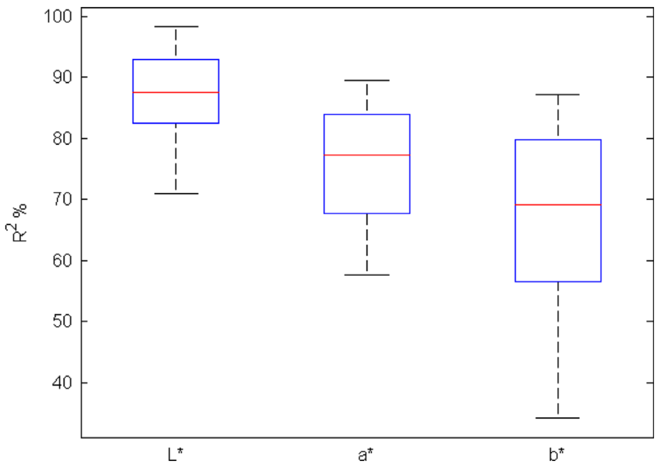

Fig. 9. R^2 box plots of the L*a ∗b ∗ components from SVM prediction results on the 25 random test sets.

RSS in Eq. (19). These results motivate us to use the multispectral images of meat to build the prediction models.

The errors in the case of the color checker training data was higher than expected. The reason for this was investigated by calculation of 0.95% confidence interval of the mean values of the color patches. First, the standard error of the regions of interests in the 24 patches of color was calculated from which, we computed $\Delta x_{24\times 20}$ for the 95% confidence interval of the mean values. Then, it was used for calculation of the confidence intervals for the three components and the averaged results were considered.

$$\Delta L_{24\times 20} = \Delta x_{24\times 20}\beta_{L(20\times 1)} \rightarrow \overline{\Delta L} = 2.67 \tag{22}$$

$$\Delta a_{24\times 20} = \Delta x_{24\times 20}\beta_{a(20\times 1)} \rightarrow \overline{\Delta a} = 7.34 \tag{23}$$

$$\Delta b_{24\times 20} = \Delta x_{24\times 20}\beta_{b(20\times 1)} \rightarrow \overline{\Delta b} = 7.90 \tag{24}$$

These results explain the reason for high $RMSE_{tr}$ for the color checker. In addition, The average values of the ROIs for the 24 color patches plus/minus the standard deviation within each region, along different wavelengths ($\mu \pm \sigma$), are plotted in Fig. 5. It shows

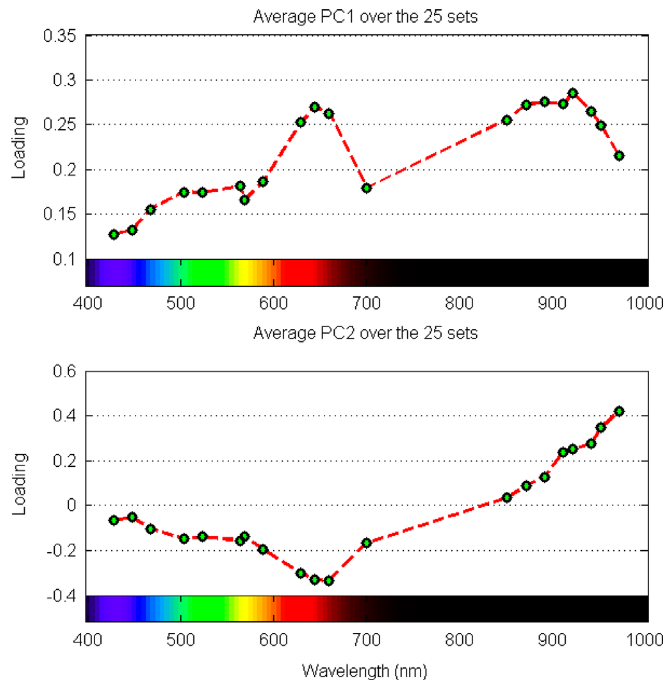

Fig. 10. The average of the first 2 PCs from the 25 random data sets versus the wavelengths.

that, although the color patches seem to be uniform, there is still variation in the spectral images of each color patch.

6.3. *Linear model results*

In this section the results of applying the linear regression methods described in Section 4.1 are presented. As stated before, the tables of average results on the 25 training and test sets are shown in the appendix. In Fig. 6, the box plots of the R^2 of the test results over the 25 different sets are shown. The R^2 results for the L* and a* components were better than b* component. The test RMSEs (see the appendix), show higher prediction error compared to the measurements errors shown in Table 1. The training set results was better than the test set. The best ΔE_{ts} was 3.12 obtained from the ridge regression. Since the 25 sets were generated randomly, possibly some of the training sets did not include the existing variation inside the original data set. Considering the fact that the original data set consists of a few samples of different types of meat, the above mentioned issue, may explain some far data points from the median in the box plots.

Since we are interested in sparse solutions, the number of times that the EN and lasso regression coefficients were non-zero in the 25 sets are illustrated for the three components in Fig. 7. We call this a frequency map because, it shows the frequency of having non-zero coefficients for each wavelength. Comparing the wavelengths with the spectrum of colors shown in the bottom of the plots, helps to find which wavelengths are mostly selected by EN and lasso. As can be seen, some near infra-red wavelengths in all cases were among the top most frequent bands.

6.4. *ANN results*

In this section, the results of applying the non-linear regression methods described in Section 4.2 are presented. Since in this paper different ANNs are compared, their names are contracted for the ease of notation. For feed-forward ANN, a simple one hidden layer architecture similar to the Fig. 4 was considered. The algorithm shown in Algorithm 1 was used for training the generalized feed-

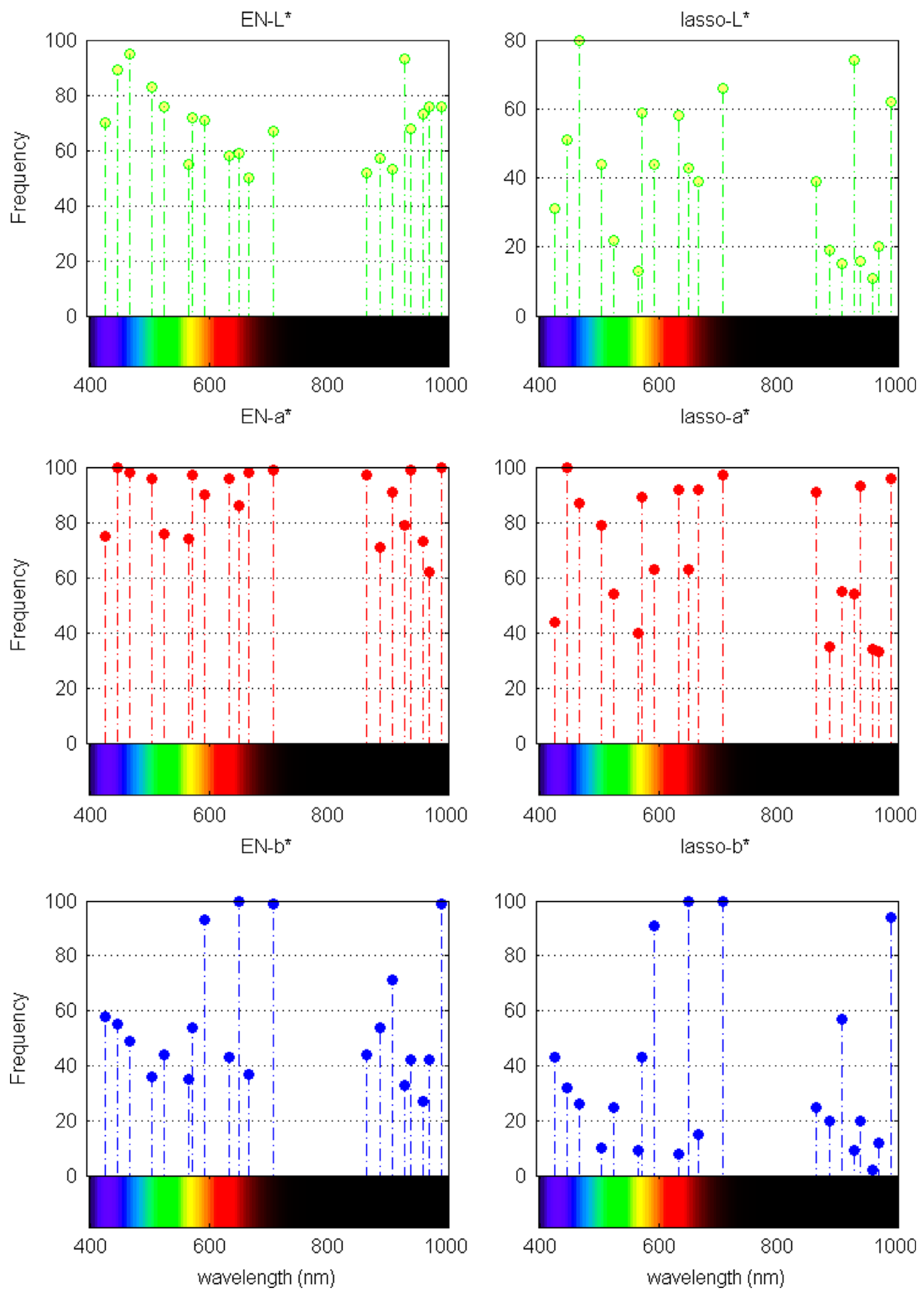

Fig. 11. The frequency map of the selected wavelengths by EN and lasso in four iterations for each of the 25 sets.

forward ANN with adaptive learning rate (CVHA), momentum BP (CVHM), Bayesian regularization (CVHB) and Neural regressor with quadratic cost function (CVHQ). The range of hidden neurons sizes were $\{5, 10, 20, 40, 60, 80, 100\}$. Similar algorithm was used for training the GRNN (CVSG). However, a loop for the best choice of the smoothing value σ was used instead of the hidden neurons loop. The regularized RBFNN model is a 2 layer network. For the smoothing value σ, 100 different values were generated logarithmically between 0.01 to 10.

Fig. 8 shows the box plots of R^2 test results. We can see that, there are some very far outliers from the median which may affect the overall average results significantly. Such a case can be seen for example, for the CVHB prediction for b* component. This may happen in ANN due to the inappropriate initial point or a convergence to a local minimum. Among the tested ANNs, the GRNN (CVSG) shows the lowest performance. Like linear models, the non-linear models work fine on the training data. The best training results are obtained from the CVHQ, CVHB and CVHA and for the test data, the best two models are the CVHQ and CVHB. The best ΔE_{ts} was 3.85 obtained by CVHQ. The average training results are satisfactory however, the test results are not better than the linear models using all the 20 wavelengths (see the appendix). One reason can be the high number of input variables. Regarding the higher complexity of the ANNs than the linear models, reducing their input variables may improve the results.

6.5. SVM results

Fig. 9 shows the box plots of R^2 test results for the three components using SVM. The results of the SVM regression model does not show a significant improvement compared to the previous methods. During training the model, a linear kernel obtained the best result and was used in the final model. In

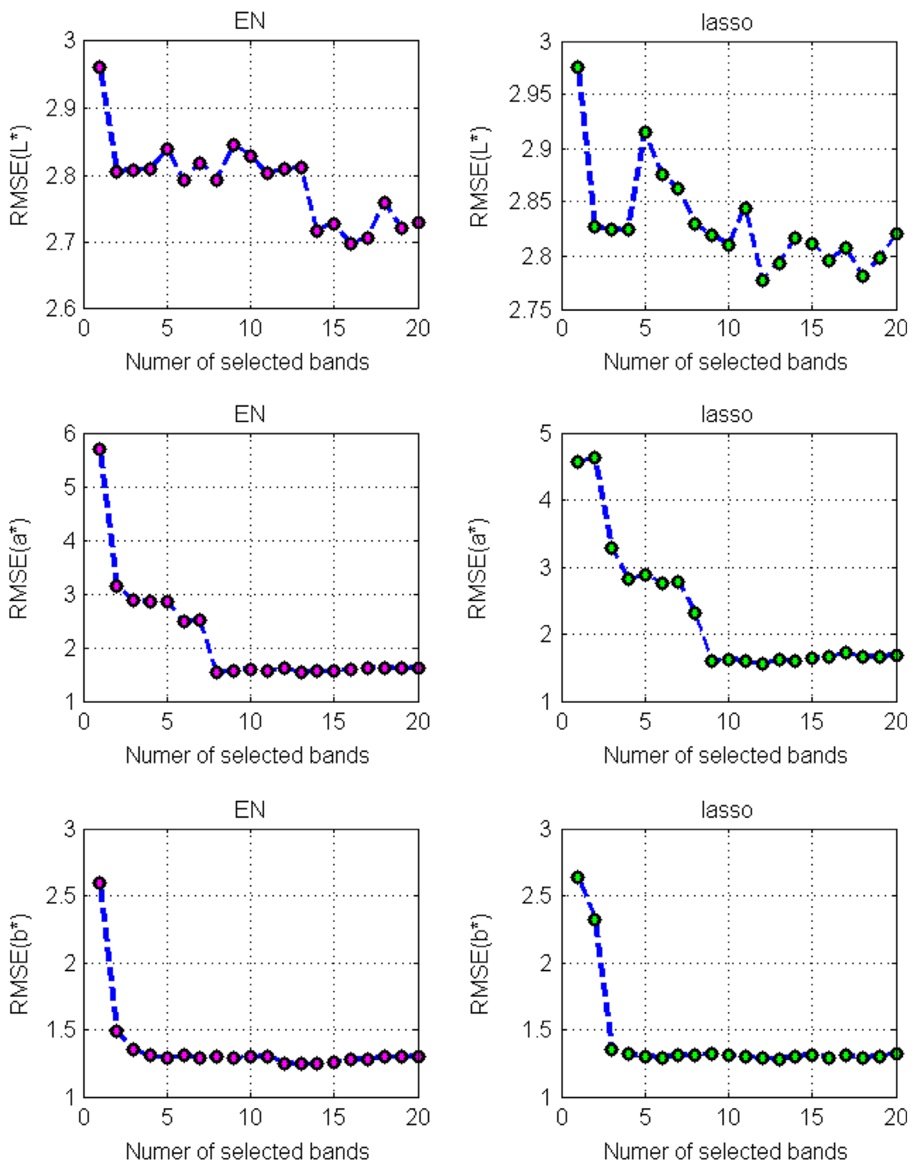

Fig. 12. The average RMSE results of EN and lasso regression for 20 candidate subsets of the sorted wavelengths.

Table 3
The number of top wavelengths selected by EN and lasso.

Component	EN	lasso
L*	16	12
a*	8	12
b*	13	13

contrast to the previous models, there are no outliers in the output results.

6.6. *Feature selection results*

The proposed supervised feature selection strategy based on EN and lasso in Section 5 as well as PCA were used to reduce the number of wavelengths. Then the resulting reduced spectral data was employed in training the models.

First, a PCA analysis was performed on each of the 25 data sets. The 97% of the variation was explained just by the first two PC components in all cases, which was a very significant reduction in data dimension. Fig. 10 shows the average of the selected PCs in the 25 data sets with respect to the wavelengths. As can be seen, both PCs enhance the higher part of the wavelengths corresponding to the NIR wavelengths. The first PC which describes more than 90% of the variations has another peak around the red color area, that corresponds to the different color ranges of the meat samples and can explain the correlation with the a* component. However, the second component shows a negative correlation peak in the red color area. It also has two small peaks in blue and yellow ranges which explains the b* color component.

The second and third sets of reduced features were formed using the Algorithm 2 in a supervised approach. Fig. 11 shows the frequency map of the 20 wavelengths by EN and lasso. Similar to the PC components, the near infra-red wavelengths have high frequencies in all cases specially, for the a* component. In addition, some visible bands were among the high frequent wavelengths. These frequencies were sorted in a descending order and their corresponding 20 wavelengths were also re-ordered. Then, for each of the 25 training sets, a candidate subset of the top wavelengths were considered and an EN regression was applied for 4 iterations. The candidate subset length was varied from 1 to 20. The average RMSE results of these 20 candidate subsets are

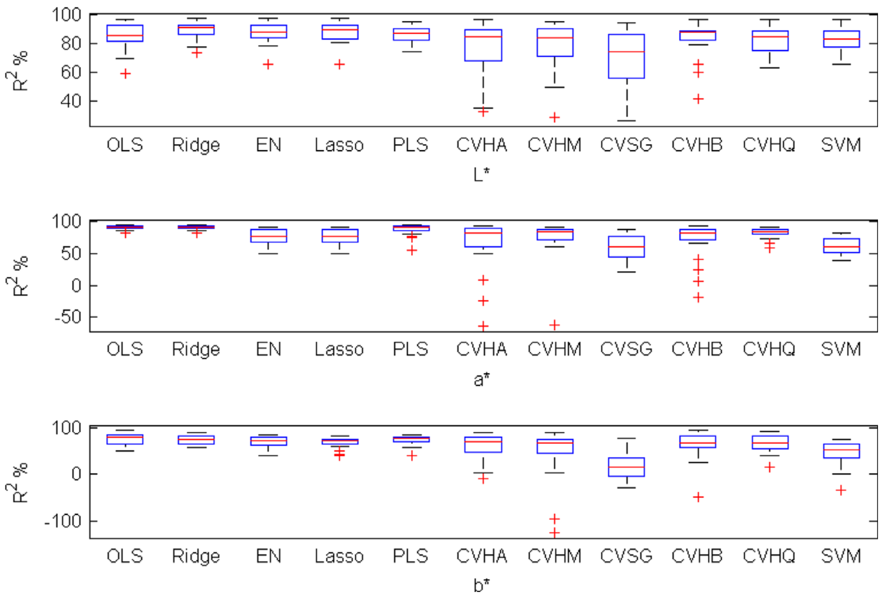

Fig. 13. R^2 box plots of the test data using EN-based Feature selection for linear, non-linear and SVM methods.

Table 4
Average R^2 and ΔE_{ts} of the test data on pseudo RGB features.

R^2	Ridge	EN	Lasso	CVHB	CVHQ	SVM
L*	87.53	87.34	87.60	88.13	86.48	87.10
a*	47.35	35.03	33.96	49.06	62.00	31.87
b*	14.02	9.11	10.33	28.87	20.84	7.95
ΔE_{ts}	4.68	4.95	4.92	4.38	4.30	4.99

illustrated in Fig. 12. The minimum RMSE corresponds to the best number of top wavelengths. Table 3, shows the final number of selected bands for each component.

The reduced sets of features obtained from the PCA and the proposed method were used to build the prediction models. Fig. 13 shows the box plots of the R^2 test results for linear, non-linear and SVM regression methods. This figure just shows the results of the EN-based feature selection. In the case of linear models, we can see that by using less bands, the results are better than Fig. 6, except for the two sparse methods, EN and lasso, as we expected. Comparison of Fig. 8 with this figure shows that, the use of less wavelengths did not made considerable changes in the median for non-linear models. Many outliers can be seen in the both box plots of the ANN methods. The lowest median among all methods was for CVSG in all the three components. Comparing Fig. 9 for SVM with this figure does not show important differences.

The complete results are presented in the appendix. The PCA did not improve the results in almost all cases. Comparing the results for the ANN models show some improvements in the maximum averages obtained on the test sets. This does not mean that all the non-linear models results were improved by the proposed features. In the case of SVM results, the most prominent improvement obtained for the a* component. Comparing all the results, the best ΔE_{ts} was 2.87 obtained from the ridge regression using the EN-based feature selection.

6.7. Comparison with RGB images

In order to investigate the effect of the number of wavelengths in the accuracy of the regression models, we have extracted the RGB components from the 20 original bands. Then, these pseudo RGB features were used to perform L*a*b* prediction using the best linear and non-linear models from the previous experiments as well as the SVM method. The average results over the 25 data sets are presented in Table 4. The prediction result in the case of L* component, is good, showing that for brightness component, the use of three RGB bands may be enough. The results for the chromatic components are worse than the multispectral bands specially in the case of the b* component. We can see that the complex non-linear methods can do significantly better predictions on the features from the limited RGB bands for the chromatic components, compared to the linear and kernel-based models. All the ΔE_{ts} values are above 4.

Although a real RGB image captured by a CCD camera may not be exactly the same as the images we formed by band extraction over the multispectral images, the poor prediction results for the color components compared to the results using multispectral bands, can demonstrate the superiority of the multispectral imaging.

6.8. Displaying L*a*b* components

In order to visualize the results of the L*a*b* color predictions, we made a prediction for all the pixels of a meat sample. To form these images, one of the trained ridge models on the EN-based feature selection method was used for each of the three components. Fig. 14 illustrates the pseudo RGB image and the corresponding images of the L*a*b* components. In the L* image, the main structure of the marbled meat is distinguishable. In the a* and b* image, we can observe the color variation in different parts of the meat.

6.9. Discussion

We investigated the use of multispectral images of raw meat for L*a*b* color prediction. Considering the variation in the results of the same methods on the 25 random sets, the important role of an appropriate training set, covering the existing variation of the population, in success of the prediction model becomes clear. Another point is that, comparison of the best results of different models show that, the use of a sub-set of features can improve the results. In our work, the proposed supervised linear feature selection algorithm outperformed the PCA for all tested methods. However, the best results were obtained by applying a non-sparse linear regression method like ridge on these features. SVM was the next best method for the selected features. Although the non-linear methods are more complex and more time-consuming in training,

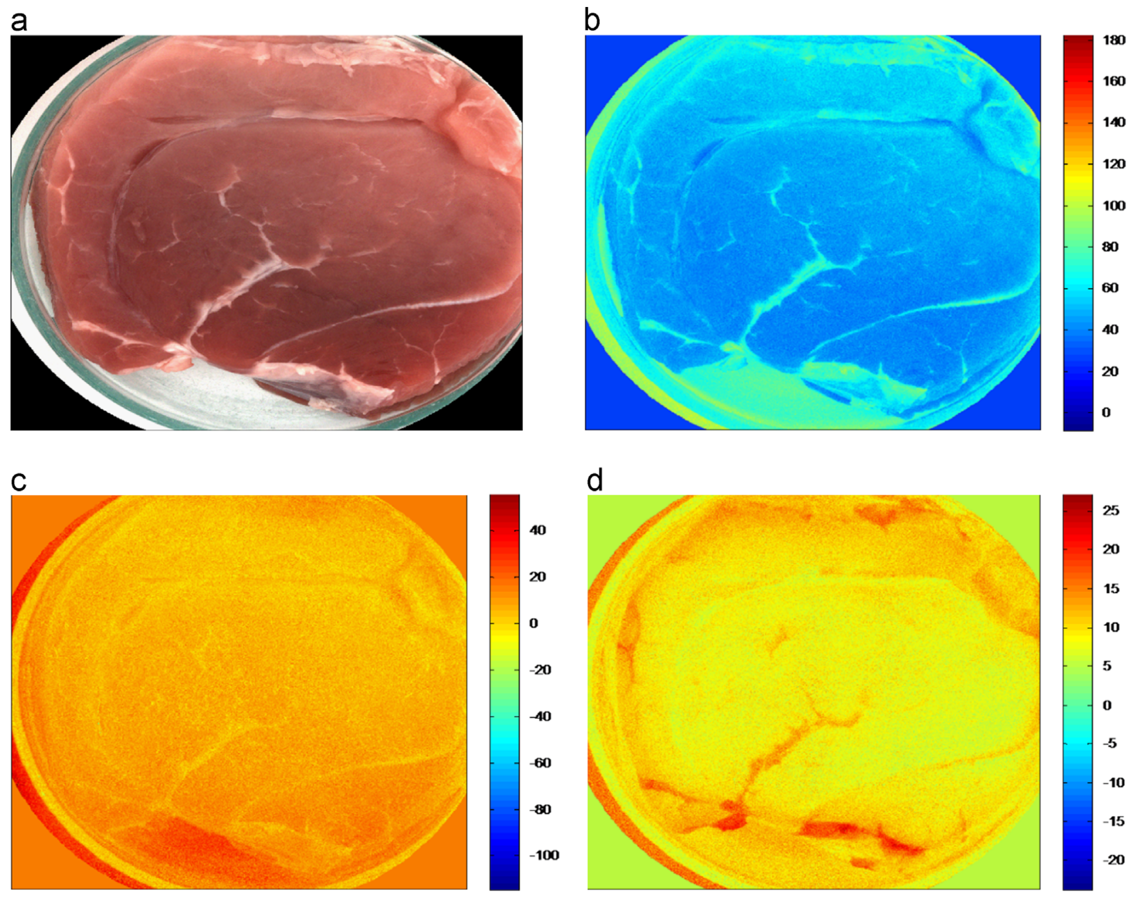

Fig. 14. (a) The RGB image of a meat sample and (b–d) its corresponding predicted L*a*b* components, respectively.

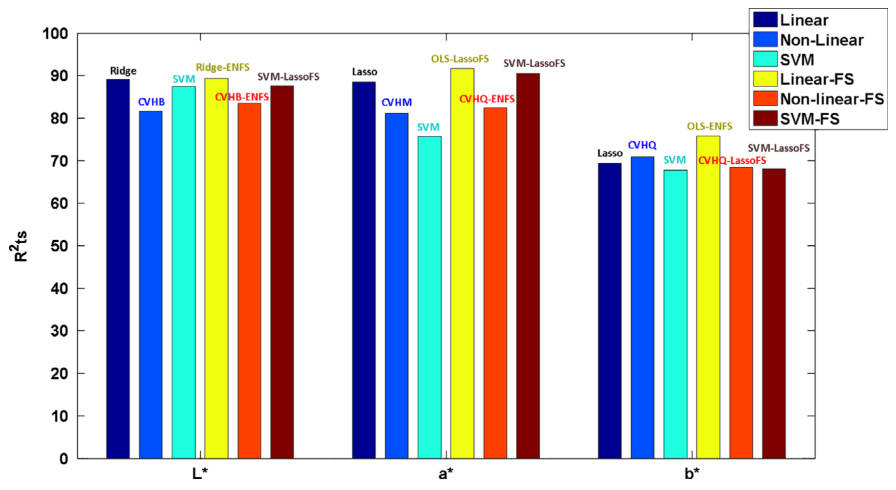

Fig. 15. Comparison of the best average R^2 test results for the linear, non-linear and SVM methods and their combinations with the feature selection (FS) methods.

they did not obtain higher results in average, compared to the two other methods. Their box plots show that, an inappropriate initial point or a convergence to a local minimum may affect the final model dramatically and their average results may not improve due to these few poor outliers. On the other hand, the results show that more complex models work better on limited number of features. The L*a*b* predictions from pseudo RGB features support this.

In addition, we found that for prediction of the L* component, simple RGB bands give good average result. But, they fails to gain acceptable results for the chromatic components.

Another important point in terms of the reduction in wavelengths is that, for each of the three components, the reduced number of wavelengths by the proposed method can perform an acceptable prediction. The best average test results of the all three strategies and their combination with the pre-processing methods are compared in Fig. 15. In addition, the comparison of the best ΔE_{ts} of these four approaches are presented in Fig. 16.

The selected features in Fig. 11, showed high frequencies in selection of the NIR wavelengths together with some visible bands in all cases. This shows the importance of the spectral imaging.

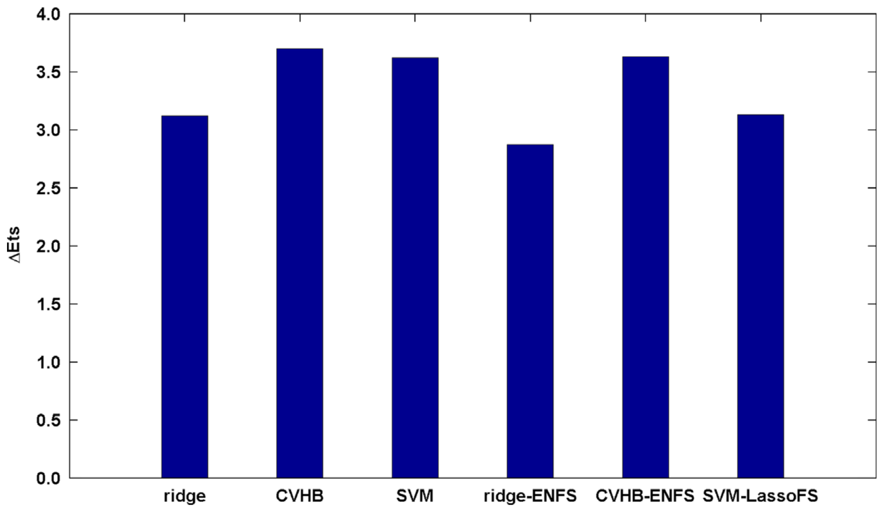

Fig. 16. Comparison of the best average ΔE_{ts} for the linear, non-linear and kernel-based methods and their combinations with the feature selection (FS) methods.

In Cao and Jun (2008), the GRNN (CVSG) was suggested for CMYK color conversion into the L*a*b*. Considering the tested non-linear models, we can see that in the case of multispectral images of meat, this model shows the lowest performance compared to the other models. However, there was no comparison in Cao and Jun (2008) between different ANN models. In Larrain et al. (2008), the regression of colorimeter measurements on RGB images of only beef samples gained the highest R^2 for a* component (96%), while for the two other components it was less than 60%. In our work, the best R^2 was also obtained for a* component and the R^2 of the two other components was higher. However, the best ΔE_{ts} in our work was less than that work (2.87 and 1.57 respectively). The main reasons are the random division and averaging over 25 test sets and also the use of different meat types (veal, beef, chicken, pork, etc.) than one item, makes the fitting task with the prediction models more difficult.

We believe that, the mismatch between the regions where measurements were performed and the ROI regions are likely one main source of error in our models. In addition, as stated before, the random division of the original data set,with limited samples of many varieties, into training and test sets can be another source of error. Because it raises the possibility that some of the training sets do not cover the existing variability inside the original data set and therefore, the average results be decreased.

7. Conclusion

In this paper, multispectral images of different kinds of raw meat were used for prediction of the L*a*b* color components, which is useful for food quality inspection. The use of meat images was preferred over the use of standard color checkers due to the special characteristics of raw meat such as transparency and fiber structure. Results from the experiments supports this. Three regression strategies, linear, non-linear and kernel-based (SVM) were compared for color conversion. In addition, finding a sparse solution with a minimum number of wavelengths is important, since they are economically more effective for industrial vision systems. Therefore, a supervised linear feature selection algorithm was proposed. This method was compared with PCA using all three strategies. In order to generalize the results and make a reliable comparison between different methods, the original data set was randomly divided 25 times into training and test sets. Comparison of the results showed that the proposed feature selection strategy with non-sparse linear regression gained the best average results for all the color components. Finally, comparison with the pseudo RGB data showed the superiority of the multispectral data for prediction of the chromatic components.

Acknowledgments

This work was (in part) financed by the Center for Imaging Food Quality project which is funded by the Danish Council for Strategic Research (contract no 09-067039) within the Program Commission on Health, Food and Welfare.

Appendix A

See Tables A1–A10.

Table A1
Average R^2 of the training and test data for linear models.

R^2%	OLS	Ridge	EN	Lasso	PLS
Training					
L*	96.64	94.19	91.00	91.67	92.90
a*	97.87	96.84	94.82	96.13	96.46
b*	92.98	86.66	82.73	83.47	84.12
Test					
L*	83.40	89.04	87.40	87.27	86.52
a*	80.45	87.26	87.13	88.56	86.11
b*	62.61	68.61	67.16	69.33	64.39

Table A2
Average RMSE and ΔE of the training and test data for linear models.

RMSE	OLS	Ridge	EN	Lasso	PLS
Training					
L*	1.51	1.98	2.45	2.37	2.17
a*	0.80	0.97	1.25	1.08	1.03
b*	0.67	0.93	1.05	1.03	1.02
ΔE_{tr}	1.59	2.09	2.61	2.47	2.30
Test					
L*	3.23	2.65	2.80	2.83	2.97
a*	2.11	1.75	1.77	1.67	1.80
b*	1.51	1.39	1.44	1.38	1.49
ΔE_{tr}	3.61	3.12	3.26	3.18	3.38

Table A3
Average R^2 of the training and test data for ANN models.

R^2%	CVHA	CVHM	CVSG	CVHB	CVHQ
Training					
L*	95.87	93.93	91.27	94.53	99.82
a*	96.04	94.77	92.69	96.03	99.70
b*	91.25	91.40	75.45	92.00	98.15
Test					
L*	73.18	73.88	68.65	81.68	79.96
a*	73.94	81.13	50.86	79.65	74.59
b*	69.72	53.29	18.00	56.60	70.86

Table A4
Average RMSE and ΔE of the training and test data for ANN models.

RMSE	CVHA	CVHM	CVSG	CVHB	CVHQ
Training					
L*	1.63	1.98	2.21	1.89	0.23
a*	1.07	1.25	1.42	1.08	0.19
b*	0.70	0.74	1.15	0.72	0.26
ΔE_{tr}	1.84	2.18	2.49	2.04	0.26
Test					
L*	3.82	3.89	4.34	3.37	3.59
a*	2.38	2.13	3.39	2.21	2.43
b*	1.37	1.60	2.30	1.55	1.33
ΔE_{tr}	3.90	3.99	5.19	3.70	3.85

Table A5
Average results for the training and test data by SVM regression.

Component	R^2_{tr}%	$RMSE_{tr}$	R_{ts}%	$RMSE_{ts}$
L*	89.74	2.62	87.49	2.79
a*	88.11	1.90	75.71	2.44
b*	82.55	1.07	67.78	1.41
ΔE_{tr}		3.073		
ΔE_{ts}		3.62		

Table A6
Average R^2 of the training and test data for linear models using the selected features.

R^2%		OLS	Ridge	EN	Lasso	PLS
Training						
L*	EN	95.25	93.92	90.48	90.88	92.97
	Lasso	94.68	93.67	90.48	90.92	93.20
	PCA	85.45	85.45	85.15	85.13	85.31
a*	EN	95.47	95.46	89.53	89.39	95.21
	Lasso	96.95	96.65	90.32	89.79	96.61
	PCA	75.19	75.19	15.46	14.10	38.62
b*	EN	90.46	87.57	84.45	83.60	85.34
	Lasso	90.45	86.72	83.32	83.83	85.90
	PCA	57.28	57.28	55.15	55.06	28.98
Test						
L*	EN	85.10	89.30	87.69	88.03	86.57
	Lasso	87.50	88.91	87.63	88.03	87.33
	PCA	83.47	83.48	84.38	84.38	84.51
a*	EN	91.42	91.46	76.53	76.43	88.18
	Lasso	91.70	91.19	79.88	79.01	90.58
	PCA	62.71	62.75	−1.68	−3.27	19.43
b*	EN	75.73	73.57	69.26	68.36	73.54
	Lasso	75.71	71.69	68.78	68.16	71.20
	PCA	44.25	44.32	46.98	47.86	−0.77

Table A7
Average RMSE of the training and test data for linear models using the selected features.

RMSE%		OLS	Ridge	EN	Lasso	PLS
Training						
L*	EN	1.79	2.02	2.53	2.48	2.16
	Lasso	1.90	2.06	2.54	2.48	2.14
	PCA	3.13	3.13	3.16	3.17	3.15
a*	EN	1.17	1.18	1.78	1.79	1.21
	Lasso	0.96	1.01	1.71	1.75	1.01
	PCA	2.74	2.74	5.08	5.12	4.33
b*	EN	0.79	0.90	1.01	1.03	0.97
	Lasso	0.79	0.93	1.05	1.03	0.96
	PCA	1.68	1.68	1.72	1.73	2.16
ΔE_{tr}	EN	1.99	2.18	2.95	2.91	2.36
	Lasso	1.99	2.16	2.93	2.90	2.24
	PCA	3.98	3.99	5.76	5.79	5.17
Test						
L*	EN	3.02	2.60	2.76	2.73	2.96
	Lasso	2.82	2.66	2.77	2.73	2.84
	PCA	3.22	3.22	3.15	3.15	3.12
a*	EN	1.47	1.47	2.38	2.39	1.67
	Lasso	1.45	1.48	2.23	2.28	1.54
	PCA	3.03	3.03	5.15	5.19	4.48
b*	EN	1.21	1.29	1.39	1.42	1.29
	Lasso	1.21	1.33	1.39	1.41	1.34
	PCA	1.85	1.85	1.83	1.82	2.52
ΔE_{ts}	EN	3.11	2.87	3.54	3.54	3.22
	Lasso	3.03	2.99	3.48	3.48	3.17
	PCA	4.26	4.26	5.80	5.82	5.37

Table A8
Average R^2 of the training and test data for ANN models using the selected features.

R^2%		CVHA	CVHM	CVSG	CVHB	CVHQ
Training						
L*	EN	95.39	94.34	93.19	93.12	99.84
	Lasso	95.55	92.59	92.80	94.20	99.73
	PCA	85.57	87.54	86.54	86.87	91.70
a*	EN	95.33	93.87	94.42	95.58	99.16
	Lasso	95.84	94.73	90.58	96.71	99.45
	PCA	85.46	83.83	85.56	83.64	86.09
b*	EN	91.65	90.80	70.88	91.67	98.64
	Lasso	92.43	76.92	77.66	91.41	98.39
	PCA	78.32	74.77	66.30	75.95	80.44
Test						
L*	EN	75.63	78.24	68.70	83.45	81.97
	Lasso	77.19	72.48	66.17	80.95	79.50
	PCA	60.60	72.81	73.65	82.26	79.27
a*	EN	67.07	74.54	60.20	72.96	82.46
	Lasso	78.49	67.59	57.37	76.65	77.17
	PCA	35.04	42.97	54.16	61.84	61.77
b*	EN	58.56	47.30	16.64	61.50	66.95
	Lasso	57.90	37.95	15.37	66.10	68.42
	PCA	25.73	41.49	36.18	33.72	48.39

Table A9
Average RMSE of the training and test data for ANN models using the selected features.

RMSE%		CVHA	CVHM	CVSG	CVHB	CVHQ
Training						
L*	EN	1.74	1.95	1.94	2.14	0.23
	Lasso	1.71	2.06	2.00	1.94	0.31
	PCA	3.03	2.88	2.96	2.96	2.34
a*	EN	1.18	1.35	1.19	1.14	0.42
	Lasso	1.12	1.25	1.55	0.99	0.29
	PCA	2.08	2.20	2.06	2.21	2.04
b*	EN	0.73	0.77	1.24	0.73	0.23
	Lasso	0.70	1.04	1.05	0.74	0.26
	PCA	1.18	1.28	1.42	1.25	1.13
ΔE_{tr}	EN	2.01	2.23	2.23	2.22	0.49
	Lasso	1.93	2.33	2.33	2.04	0.48
	PCA	3.37	3.41	3.40	3.42	2.92
Test						
L*	EN	3.79	3.60	4.37	3.23	3.37
	Lasso	3.77	3.84	4.54	3.34	3.50
	PCA	4.81	4.16	4.07	3.32	3.54
a*	EN	2.56	2.38	3.11	2.40	2.07
	Lasso	2.24	2.68	3.14	2.33	2.34
	PCA	3.79	3.56	3.32	3.03	3.01
b*	EN	1.55	1.69	2.33	1.49	1.42
	Lasso	1.59	1.84	2.35	1.42	1.38
	PCA	2.10	1.88	2.01	2.02	1.80
ΔE_{ts}	EN	4.10	3.94	5.02	3.64	3.66
	Lasso	3.94	4.21	5.28	3.68	3.85
	PCA	5.49	4.97	4.85	4.28	4.40

Table A10
Average results of the training and test data for SVM model using the selected features.

Component	Method	$R^2_{tr}\%$	$RMSE_{tr}$	$R_{ts}\%$	$RMSE_{ts}$
L*	EN	89.83	2.62	87.32	2.79
	Lasso	89.49	2.66	87.62	2.77
	PCA	85.08	3.17	83.05	3.26
a*	EN	89.67	1.77	78.05	2.31
	Lasso	95.64	1.15	90.46	1.55
	PCA	73.70	2.82	62.34	3.08
b*	EN	81.16	1.11	70.00	1.40
	Lasso	80.91	1.12	68.12	1.41
	PCA	56.06	1.70	44.68	1.84
ΔE_{tr}	EN		3.02		
	Lasso		2.71		
	PCA		4.03		
ΔE_{ts}	EN		3.55		
	Lasso		3.13		
	PCA		4.31		

References

Balaban, M.O., Odabasi, A.Z., 2006. Measuring color with machine vision. Food Technology 60, 32–36.

Bishop, M.C., 2006. Pattern Recognition and Machine Learning. Springer, Springer Science + Business Media, LLC.

Bishop, M.C., Tipping, E. Michael, 2003. Bayesian regression and classification. Advances in Learning Theory: Methods, Models and Applications, IOS Press-NATO Science Series III: Computer and Systems Sciences 52.

Brewer, M.S., Novakofski, J., Freise, K., 2006. Instrumental evaluation of pH effects on ability of pork chops to bloom. Meat Science 72 (4), 596–602.

Cao, C., Jun, Q., 2008. Study on color space conversion between CMYK and CIE L*a*b* based on generalized regression neural network. Proceedings of the IEEE International Conference on Computer Science and Software Engineering 2, 275–277.

Cao, C., Jun, Q., 2011. Study on color space conversion based on RBF neural network. Advanced Materials Research 174 (28), 28–31.

Dissing, B.S., Carstensen, J., Larsen, R., 2010. Multispectral colormapping using penalized least square. Journal of Imaging Science and Technology 54 (3) 0304011–16.

Dissing, B.S., Clemmensen, H.L., Løje, H., Ersbøll, K.B., Nissen, J.A., 2009. Temporal reflectance changes in vegetables. In: IEEE International Conference on Computer Vision Workshops (ICCV), pp. 1917–1922.

Fdhal, N., Kyan, M., Androutsos, D., Sharma, A., 2009. Color space transformation from RGB to CIELAB using neural networks. In: Proceedings of the 10th Pacific Rim Conference on Multimedia: Advances in Multimedia Information Processing Springer, vol. 9, pp. 1011–1017.

Gamal, E., Ning, W., Clément, V., 2009. Detecting chilling injury in red delicious apple using hyperspectral imaging and neural networks. Postharvest Biology and Technology 52 (1), 1–8.

Hagan, M.T., Demuth, H.B., Beale, M.H., 1996. Neural Network Design. PWS, Boston, USA.

Hastie, T., Tibshirani, R., Jerome, F., 2008. The Elements of Statistical Learning. Springer, Springer Science + Business Media, NY.

Kolenda, T., Sigurdsson, S., Winther, O., Hansen, L.K., Larsen, J., 2002. DTU:Toolbox. ISP Group, Informatics and Mathematical Modeling, Technical University of Denmark.

Larrain, R.E., Schaefer, D.M., Reed, J.D., 2008. Use of digital images to estimate CIE color coordinates of beef. Food Research International 41, 380–385.

León, K., Mery, D., Pedreschi, F., Jorge, L., 2006. Color measurement in L*a*b* units from RGB digital images. Elsevier-Food research International 39 (10), 1084–1091.

Mendoza, F., Dejmekb, P., Aguilera, J.M., 2006. Calibrated color measurements of agricultural foods using image analysis. Postharvest Biology-Elsevier Science B. V. 4, 285–295.

Sharifzadeh, S., Clemmensen, L.H., Løje, H., Ersbøll, K.B., 2013. Statistical quality assessment of pre-fried carrots using multispectral imaging. In: Proceedings of the 18th Scandinavian Conference on Image Processing Springer Lecture Notes in Computer Science, vol. 7944, pp. 620–629.

Sharifzadeh, S., Skytte, J.L., Nielsen, O.H.A., Ersbøll, K.B., Clemmensen, L.H., 2012. Regression and sparse regression methods for viscosity estimation of acid milk from it's SLS features. In: Proceedings of the IEEE International Conference on Systems, Signals and Image Processing, pp. 52–55.

Shi, L., Xu, L., 2006. Comparative investigation on dimension reduction and regression in three layer feed-forward neural network. International Conference on Artificial Neural Networks ICANN 4131/2006, 51–60.

Specht, F.D., 1991. A general regression neural network. IEEE Transaction on Neural Networks 2 (6), 568–576.

Sun, D.W. (Ed.), 2010. Hyperspectral Imaging for Food Quality Analysis and Control. Elsevier, London, UK.

Tkalčič, M., Tasič, J.F., 2003. Color spaces - perceptual, historical and applicational background. EUROCON- Computer as a Tool, pp. 304–308.

Upton, S., 2006. Delta E: the color difference. 〈http://www.colorwiki.com/wiki/Delta_E:_The_Color_Difference〉, Online; accessed 17-July-2012.

Wu, D., Sun, D.W., 2013. Colour measurements by computer vision for food quality control – a review. Trends in Food Science and Technology 29 (1), 5–20.

X-Rite, 2004. The color guide and glossary. 〈http://graphics.tech.uh.edu/courses/4373/materials/X-Rite_Color_Guide_2004.pdf〉, Online; accessed 17-July-2012.