SPARSE CODED AUTOENCODER FEATURES FOR CHEMOMETRIC DATA ANALYSIS

Muhammad Bilal[1] *Mohib Ullah*[2]

[1]University of Trento, Italy.
[2]Norwegian University of Science and Technology, Norway.

ABSTRACT

We proposed a deep learning based chemometric data analysis technique. We trained L2 regularized sparse autoencoder end-to-end for reducing the size of the feature vector to handle the classic problem of curse of dimensionality in chemometric data analysis. We introduce a novel technique of automatic selection of nodes inside hidden layer of an autoencoder through pareto optimization. Moreover, linear regression, ε-SVR , and Gaussian process regressor are applied on the reduced size feature vector for the regression. We evaluated our technique on orange juice and wine dataset and results are compared against state-of-the-art methods. Quantitative results are shown on Normalized Mean Square Error (NMSE) and the results show considerable improvement in the state-of-the-art.

Index Terms— Chemometric data,sparse autoencoder, gaussian process regressor, pareto optimization.

1. INTRODUCTION

The term chemometric refers to the process of extracting meaningful information from data obtained through a chemical experiment [1]. It is a multidisciplinary field of study where techniques from statistical modeling, bayesian inference, deep learning, signal and image processing are applied for data analysis and achieve objective results. Usually, data come from fields like chemistry, Pharmaceutics, biochemistry, to name a few. In a nutshell, chemometric consist of three steps:

- Data acquisition
- Descriptive analysis
- Predictive analysis

Chemometric process begins with data acquisition phase. In our experiment, data is obtained through spectroscopy. In spectroscopy, the material under consideration (wine and orange juice in our case) is illuminated with a light source of specific wavelength and the reflection and transmission properties are recorded as a function of wavelength. The instrument that is used for measuring the intensity of light ray as a function of its wavelength is called photometer. The essential characteristics of a spectrophotometer is operational bandwidth, i.e. the range of wavelengths a spectrophotometer is capable of working on the test samples. The the transmission, reflectance and absorption capacities of a test sample are also important parameters. Modern spectrophotometer works in the following four steps:

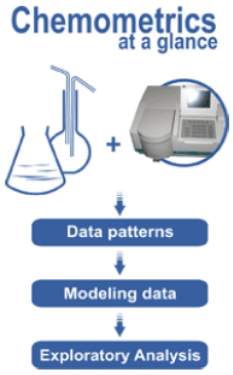


Fig. 1: Chemometric analysis in a nutshell.

- The light rays of the desired wavelength are passed through sample under test.
- The incident rays are either transmitted or reflected from the sample.
- The resultant light falls on photodetector device and produces current.
- The current produced is amplified and converted into absorption or transmission values that are later on used for analysis.

Based on the nature of application, spectroscopic analysis uses different light source like Ultraviolet-Visible (UV-Vis), Infrared (IR), Near Magnetic Resonance (NMR) and X-ray. We will here particularly focus on the IR spectroscopy due to the fact the two datasets for our analysis are collected through IR spectrophotometer.

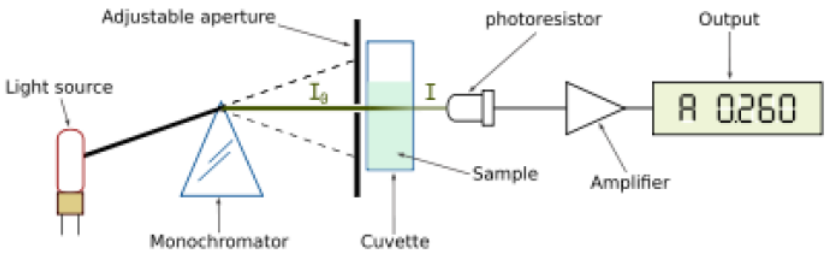


Fig. 2: Single beam spectrophotometer

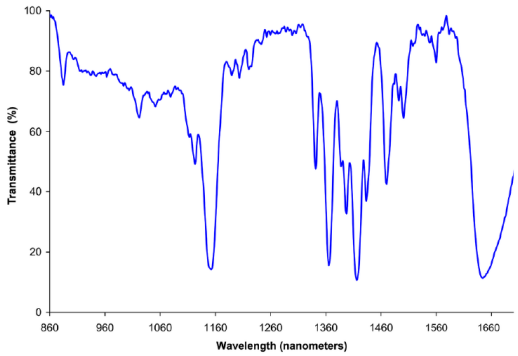


Fig. 3: Near-IR absorption spectrum of dichloromethane showing complicated overlapping overtones of mid IR absorption features

1.1. IR spectroscopy

It deals with the effects of interaction of infrared radiation with matter for experimental purposes. Infrared spectrometers are used to create infrared spectrum for experimental analysis on the samples, which is basically a graph of infrared light absorbance or transmittance on vertical axis and frequency or wavelength on horizontal axis, as given in fig. 3.

The infrared portion of electromagnetic spectrum is further divided into the following regions;

- Far infrared, with the range 40010 cm^1
- Mid infrared, with the range 4000400 cm^1
- Near infrared, with the range 140004000 cm^1

The mid infrared region is also called FTIR, where FT stands for Fourier Transform. It is called FTIR because to get the actual spectrum a Fourier transform is applied to original raw data that is obtained through FTIR spectrometer. Near infrared spectroscopy (NIR) has found wide use in applications that are best suited with rapid analyses that allow one to measure a sample in its natural state, as it does not require sample preparation.

Then second and third step of chemometric consist of description and predictive analysis. In descriptive analysis, properties of chemical systems are modeled for determining the fundamental relationships and structure of the system. While in predictive analysis, properties of chemical systems are modeled for predicting new properties or behavior of interest. In both cases, the datasets can be small but highly complex. Depending on the system requirements and complexity levels, the investigative parameters and variables can range from few hundreds to few thousands resulting into a large number of outcomes and case studies. This work is mainly focused on the predictive analysis.

Primarily, the focus is on estimating sugar and alcohol concentration in orange juice and wine. The data is acquired through near infrared (orange juice dataset) and mind infrared (wine dataset) reflectance spectroscopy. The acquired data suffers from the curse of dimensionality. An autoencoder network has been designed for reducing the data dimensionality and hence addressing this issues. The parameters selection of network is modeled as an optimization problem and pareto optimization is used to get the optimal set of parameters. Moreover, three regression models (linear, SVM, Gaussian process) have been used to get the estimation on sugar and alcohol concentration.

The paper is organized in the following order. In section 2, related work is elaborated. In the section 3, the proposed method is explained. In section 4, we present autoencoder training and parameter optimization. The loss function and the training strategies are explained in section 5 and 6, respectively. Section 7 illustrate our baseline regressor. In section 8, quantitative results are presented and section 9 concludes the paper.

2. RELATED WORK

Generally modern spectroscopic data is multivariate and collected data has components from the entire spectrum instead of a single wavelength over time (univariate date). It consists of more variables or features than observations or samples. This not only creates curse of dimensionality issue but also the consecutive variables in a spectrum are highly correlated in nature, that is, some spectral variables can be represented as linear combination of other independent variables. Pre-

vious work suggests that existence of such high collinearity between the spectral variables can result in inaccurate predictions [2, 3]. It is problematic to apply directly statistical methods due to high collinearity, like multiple linear regression (MLR) in [4–7].

In order to deal with curse of dimensionality and data collinearity, different works have been proposed in literature [8]. Most of the previous work suggests to reduce the number of variables or features to cope with curse of dimensionality problem, thus allowing to get more accurate results through regression techniques. The feature reduction can be generally achieved in two different methods. The first one contains selecting the most relevant features based on a chosen criterion from the original set of features [9–12], while in second case the original features are transformed from one space to another in such a way to keep the reconstruction error as minimum as possible. This transformation of features set from one space to another can either be linear or non-linear depending upon the scope of application. The examples of later method are Principle Component Regression [13] (PCR) and Partial Least Square Regression (PLSR) [6]. PCR is composed of simple linear regression model based on few principle components of the original spectral data. While PLSR focuses on calculating the linear projections that shows maximum correlation with the output or target variable, thus estimating a linear regression model determined by the projected coordinates. Gujral et al. [14] used unlabeled data for reducing the the modeling error. Moreover, an analytic study on a sequential version of Optimal filtering (OF) based PLSR and PCA-based PLSR is conducted and it has been proved analytically that OFbased PLSR is equivalent to that of PCAbased PLSR.

Benoudjit et al. [10] proposed linear and nonlinear regression methodologies which are based upon an incremental routine for feature selection and using a validation set. In [11,12], different techniques have been introduced to improve the results of previous method by choosing the best feature set for initializing the routine and finding a feature selection strategy that depends entirely on the shared information between spectral data and target variable. An interesting approach to the chemometric problem has been discussed in [15], where instead of traditional feature reduction approach, the whole information in spectral data space is exploited by using Multiple Regression System (MRS). In MRS approach, the estimates obtained from a group of regression methods are fused together through a defined mechanism to get the final estimate. The results obtained from MRS can be more accurate than single regressor method provided the ensemble is properly designed. The results obtained from MRS technique outperform the ones obtained through traditional feature reduction techniques such as PCR and PLSR [6, 15].

Douak et al. [16] come up with two stage regression approach that is based on residual-based correction (RBC) concept. Their basic idea is to correct any adopted regressor, called functional estimator, by analyzing and modeling its residual errors directly in feature space. RBC is particularly a correction method and not a regressor which does not focuses on achieving the best possible accuracy on the given data set but rather improving the estimation model of the given regression error. In general this proposed technique is suitable for any regression method. The underlying motivation for RBC is that it is typically challenging to acquire a single feature set that is alone capable of giving highly accurate estimates over the total input space. This is due to the fact that the accuracy of the estimator depends on the region of the input space to which the analyzed pattern belongs to.

A semi-supervised approach has been presented in [17], where the unlabeled samples (whose spectral values are known, but the corresponding output concentration values are unknown) are used during the design and configuration of the regressor model in order to counterbalance the deficiency in labeled samples. The advantage in those samples come at no price from the data which is under analysis. Similarly, [18] et al. proposed a semi-supervised Fisher's linear discriminant analysis (LDA) for projecting the multivariate chemometric data into a lower dimensional space for better data separability. Consequently, these projections are used to classify test data into different groups.

Another solution to the problem of collecting training samples is proposed in [19], which is based on active learning approach for classification problems. In active learning the process starts from a small training set and the samples are added to it from large amount of unlabeled data until a stop criterion is reached. The added samples are labeled by human expert. In case of regression problems, the idea of active learning is used in [20] for estimation of chemical component of interest in spectroscopic data. In order to prepare the possible optimal training set, the process starts from small and sub optimal training set and iteratively chooses samples from the set of unlabeled data that gives smaller prediction errors. For this purpose, two active learning strategies are used in order to obtain an optimal training set which are tested on various linear and nonlinear regressors. The proposed method shows higher performance in terms of estimation accuracy in comparison to the methods where features are chosen randomly for the estimation of chemical components of interest for regression methods. However, the computational cost is higher due to the nature of learning approaches presented in [20]. In [21], the components of interest from the spectroscopic data are estimated from the ensemble of regressors using the induced ordered weighted averaging (IOWA) fusion operators. In the ensemble of regressors Gaussian process regression and extreme machine learning (EML) are used which are characterized with different mapping kernels. These two estimators have same prediction equation but problem formulation approach is different. In order to cope with model selection for EML, a novel automatic procedure is developed that is based on differential evolution. DE provides simple but powerful evolutionary optimization algorithm that uses

normalized mean square error as performance indicator for obtaining optimal ELM parameters.

The commonality among existing approaches is, they assumed that the training set is composed of sufficient number of training samples in order to obtain accurate estimates of the chemical components of interests through regression techniques. However, the process of obtaining training samples through spectroscopic technique is complex and costly. It can be subjected to errors because the concentration measurement is carried out manually by human experts. Consequently, the datasets collected for both training and testing purposes are limited and it possibly affects the performance of the overall estimation process.

The importance of feature reduction and salient features extraction have been highlighted in other computer vision applications as well like scene understanding [22–24], crowd analysis [25–27], illuminant estimation [28], segmentation [29, 30], and anomaly detection [31], to name a few. Similarly, In the last few years, deep learning has shown outstanding results on a image classification [32], segmentation [33], and tracking [34, 35] etc. Inspired by the success of deep learning in such applications, we proposed an automatic deep learning based feature extraction technique. In a nutshell, deep models learns hierarchical features [36,37] and have the capability to learn the structure of the underlying data. We designed an autoencoder neural network [38] for removing redundant and irrelevant features from the spectral data. It is an automatic feature extraction technique that is capable of linear and nonlinear feature extraction based upon the selection of parameters of the architecture. Pareto optimization technique is applied in order to choose the best architecture (in terms of model complexity) of autoencoder for the feature extraction. After extracting meaningful features from the original feature set, we exploited 3 repression techniques (Linear, ε-SVR, and Gaussian process regression) to solved our concentration estimation problem.

3. PROPOSED APPROACH

We focus on estimating sugar and alcohol concentration in orange juice and wine datasets. The whole pipeline can be seen in 3 discrete steps. In the first step, date is acquired from the liquids through near infrared (orange juice data set) and mid infrared (wine dataset) reflectance spectroscopy technique Fig. 4 (a,b). In the 2nd step, the acquired data is processed through auto-encoder neural network for feature reduction Fig. 4 (c) and then in the last step, three regression techniques (Gaussian process regressor, linear regressor, SVM regressor) has been used to estimate the concentration of the mentioned components Fig. 4 (d). The aim of autoencoder is to retrieve set of features which are the best representation of original data without redundancy. In a nutshell, the contribution of our work is three folds:

- We proposed a deep learning based framework for automatic chromatic data analysis.
- An autoencoder neural network is designed for addressing the problem of curse of dimensionality and data collinearity. Moreover, pareto optimization is used to struck the optimal number of parameters in the network.
- Three different regressor has been used for estimating the concentration of sugar and alcohol in the datasets and extensive experiments are conducted to validate the proposed scheme.

In the next section 4, autoencoder design, the parameter optimization and feature reduction strategy is explained.

4. AUTOENCODER

An autoencoder is a neural network that is trained to reconstruct the input data into the output with minimum amount of reconstruction error. They are designed in a way not to copy the input but to learn important and unique feature of the input data. Autoencoders are mainly used for pre-train deep networks dimensionality reduction, feature learning and generative modeling of data. It is composed of two main parts, input layer and output layer together with hidden layer connecting the two layers. The input layer has the same number of nodes as the output layer. To build an autoencoder we need encoding function at the input, decoding function at the output and loss function to calculate the amount of information loss between encoded representation and decoded representation of the input and output data respectively.

4.1. Encoder

It maps an input vector $x \in \mathbb{R}^n$, into encoded representation $h(x) \in \mathbb{R}^m$. The typical form is affine mapping followed by nonlinearity (eq. 1). The parameter set is $\Theta = (w, b)$ where w is weight matrix of size mxn and $b \in \mathbb{R}^m$ is bias vector, f is activation function.

$$h_\Theta = f(wx + b) \tag{1}$$

4.2. Decoder

It maps the resulting encoded representation $h(x)$ back into an estimate of reconstructed n-dimensional vector $r \in \mathbb{R}^n$, where

$$r_\phi = g(f(x)) \tag{2}$$

$$r_\phi = g(w'h + b') \tag{3}$$

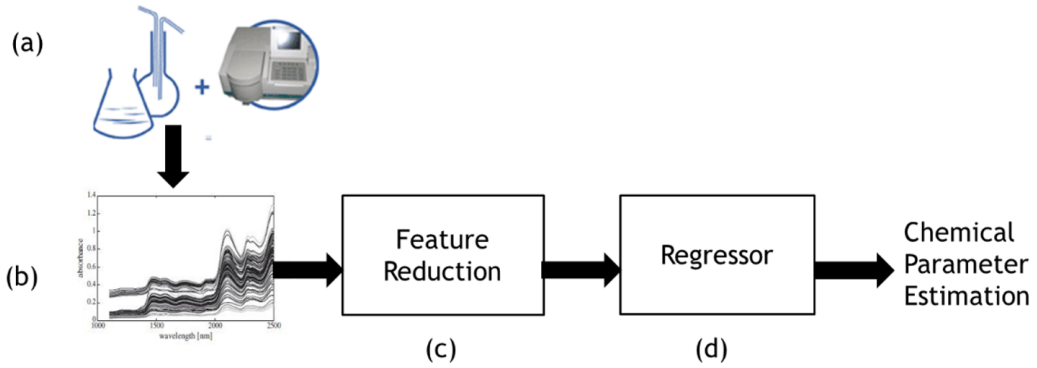


Fig. 4: (a) Data collection through spectrophotometry. (b) Spectral signature. (c) Feature reduction through autoencoder neural network. (d) Gaussian process regressor for estimating chemical parameter of interest.

The parameter $\phi = \{w', b'\}$, where w' is weight matrix of size is nxm, $b' \in \mathcal{R}^n$ is bias vector and g is activation function of the decoder. The autoencoder tries to learn a function $r_\phi \simeq x$, thus each training data $x(i)$ is mapped to corresponding reconstructed data $r(i)$. By forcing the number of hidden nodes lower than the dimension of the input data, the autoencoder tries to learn representative structure of the data or by imposing sparsity constraint on the hidden nodes the autoencoder learns useful features even for hidden nodes equal or higher than the input dimension. For a given N training samples the autoencoder learns to minimize the loss function eq. 5 such as mean squared reconstruction error by optimizing the model parameters Θ and Ω eq. 4

$$\Theta, \Omega = \arg\min_{\Theta,\Omega} \; L(x, r) \tag{4}$$

$$L = \frac{1}{N}\sum_{n=1}^{N}(x^i - r^i)^2 \tag{5}$$

4.3. Regularized autoencoder

In order to have a flexible model independent of the size of the hidden nodes and capability of the activation functions on the other hand that relies on the complexity of distribution of the data is an ideal condition. To achieve this, we introduce additional regularization parameters to the loss function, mainly sparsity of representation and smallness of the derivative of the representation (weight decay).

4.3.1. Regularizing by weight decay (L2 regularization)

To avoid overfitting - a problem where a model memorizes training data but not able to generalize when it is given unseen data leading to performance decline in the model, the rate in which the model reacts to changes in the training example distribution is penalized by forcing the autoencoder to learn most significant features.

$$\Omega_{weights} = \frac{1}{2}\sum_{l}^{L}\sum_{j}^{N}\sum_{i}^{K}(w_{ij}^{l}))^2 \tag{6}$$

where L the number of hidden layers is, N is the number of training examples and K is the number of features.

4.4. Sparse autoencoder

Typical use of sparse Autoencoders is to learn features for the purpose of classification or regression. Imposing sparsity constraint on the hidden nodes enables the model to learn unique statistical features of the data even when the number of hidden units are larger compared to the feature space of the input data. A neuron is considered active if its output value is close to maximum value of the activation function used (close to 1 for sigmoid activation function) and inactive if its output is close to minimum value (0 in case of sigmoid). The average activation of i^{th} hidden unit $\hat{\rho}_i$ eq. 7, where n is total number of inputs, x_j is the j^{th} training example and h_i activation of j^{th} hidden unit is given as:

$$\hat{\rho}_i = \frac{1}{n}\sum_{j=1}^{n} h_i(x_j) \tag{7}$$

$$\Omega_{sparsity} = \sum_{i=1}^{m} KL(\rho||\hat{\rho}_i) \tag{8}$$

Choosing sparsity parameter ρ small (ρ=0.01) and imposing $\hat{\rho}_i = \rho$ constraint, Kullback-Leibler divergence term is applied to penalize $\hat{\rho}_i$. Penalty value that diverges from ρ will give reasonable result.

$$KL(\rho||\hat{\rho}_i) = \rho\log(\frac{\rho}{\hat{\rho}_i}) + (1-\rho)\log(\frac{1-\rho}{1-\hat{\rho}_i}) \tag{9}$$

where $KL(\rho||\hat{\rho}_i)$ is the Kullback-Leibler divergence between Bernoulli random variable with ρ mean and Bernoulli

random variable with mean $\hat{\rho}_i$. If $\rho = \hat{\rho}_i$, $KL(\rho||\hat{\rho}_i) = 0$, else increases monotonically as $\hat{\rho}_i$ diverges from ρ. Minimizing the KL divergence penalty term leads to $\hat{\rho}_i$ to be close to ρ.

5. LOSS FUNCTION

Imposing sparsity constraint on the hidden nodes of an autoencoder enables the model to learn unique statistical features of the data even when the number of hidden units are larger compared to the feature space of the input data. A neuron is considered active if its output value is close to maximum value of the activation function used (close to 1 for sigmoid activation function) and inactive if its output is close to minimum value (0 in case of sigmoid). The average activation of i^{th} hidden unit $\hat{\rho}_i$ is given in eq. 7, where n is total number of inputs, x_j is the j^{th} training example and h_i activation of j^{th} hidden unit. The combining all the components, the synergetic loss function can be written as:

$$L(\Theta, \Omega) = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}(x_{kn} - r_{kn})^2 + \lambda * \Omega_{weights} + \beta * \Omega_{sparsity} \tag{10}$$

where $\Omega_{sparsity}$ is sparsity regularizer and calculated as eq. 8: Similarly, $\Omega_{weights}$ is $L2$ regularization term. It's task is to avoid overfitting by penalizing the rate in which the model reacts to changes in the training example distribution and forcing the model to learn most significant features. It is calculated as eq. 11

$$\Omega_{weights} = \frac{1}{2}\sum_{l}^{L}\sum_{j}^{N}\sum_{i}^{K}(w_{ij}^{l}))^2 \tag{11}$$

where L is number of hidden layers, N is the number of training examples and K is the number of features. In the lost function, λ is coefficient for the $L2$ regularization term, $\Omega_{weights}$ and β is the coefficient for sparsity regularization $\Omega_{sparsity}$ term.

6. TRAINING

Once the model is setup, our goal is to minimize the cost function $L(\Theta, \Omega)$ as a function of weights w and bias b. To train our autoencoder neural network, we initialized each parameter $w_{i,j}^{(l)}$,and $b_i^{(l)}$ to a small random value near zero ($\mathcal{N}(0, \varepsilon^2)$ distribution for a small ε)), and then apply stochastic conjugate gradient decent (SCG) algorithms to learn the network parameters of autoencoder. Random initialization is necessary, if all the parameters start off at identical values, then all the hidden layer units will end up learning the same function of the input.

SCG minimizes network parameters by taking steps in negative direction of the loss function. Starting with initial set of parameter values, the algorithm iteratively approaches towards a set of parameters values that minimize the loss function. Back-propagation is used to compute the derivative of the loss function with respect to network parameters. A loss function that penalizes for estimating r instead of x, gradient decent algorithm iteratively updates network parameter (weights and biases) in such a way that the reconstruction error at the output is minimized. A single iteration of SCG updates the parameters w, b as (eq. 12) and (eq. 13) respectively.

$$w_{ij}^{(l)} = w_{ij}^{(l)} - \zeta\frac{\partial}{w_{ij}^{(l)}}L \tag{12}$$

$$b_{i}^{(l)} = b_{i}^{(l)} - \zeta\frac{\partial}{b_{i}^{(l)}}L \tag{13}$$

Where ζ is learning rate and L is the loss function. The partial derivatives of the loss function $L(w, b; x, r)$ defined with respect to a single example (x, r) is given by (eq. 14) and (eq. 15) respectively.

$$\frac{\partial}{w_{ij}^{(l)}}L = \left[\frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{w_{ij}^{(l)}}L(w, b; x^{(i)}, r^{(i)})\right] + \lambda w_{ij}^{(l)} + \beta\left(-\frac{\rho}{\hat{\rho}_i} + \frac{1-\rho}{1+\hat{\rho}}\right) \tag{14}$$

$$\frac{\partial}{b_{i}^{(l)}}L = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{w_{ij}^{(l)}}L(w, b; x^{(i)}, r^{(i)}) \tag{15}$$

Back-propagation is used to efficiently compute these partial derivatives. Given a training example x, we will first run a forward pass to compute all the activations throughout the network, including the output value of the reconstruction r. Then, for each node i in layer l, we would like to compute an error term $\delta_i(l)$ that measures how much that node was responsible for any errors in our output. For an output node, we can directly measure the difference between the network's activation and the true target value, and use that to define $\delta_i(nl)$(where layer nl is the output layer). For hidden units we compute $\delta_i(l)$ based on a weighted average of the error terms of the nodes that uses $h_i(l)$ as an input. In Back-propagation algorithm, first perform a feed-forward pass computing the activation of all layers starting from the first hidden layer to the output layer nl, then for each output unit i in layer nl the error value is given by (eq. 16).

$$\delta^{(n_l)} = -(x - r) * g'(h^{(n_l)}) \tag{16}$$

$$\delta^{l} = (w^{l})^T\delta^{l+1} * g'(h^{(l)}) \qquad l = n_l - 1, n_l - 2, ..., 2 \tag{17}$$

For the hidden layers, the error is given by equation 16, and the desired partial derivatives are given by (eq. 18) for weight update and (eq. 19) for the bias term update.

$$\nabla_{w(l)}L = \delta^{l+1}(g^{l})^T \tag{18}$$

$$\nabla_{b(l)} L = \delta^{l+1} \tag{19}$$

where g is activation function at a given layer. By repeatedly taking conjugate gradient decent steps it minimizes the loss function L.

6.1. Optimization

The choice on model complexity and mean squared error (MSE) are two trade-offs to optimize the loss function. As we increase the number of hidden nodes MSE decreases. However, the increase in the number of hidden-neurons might lead to overfitting the data, thus decreasing the generalization performance of the model. The goal is to find an optimal solution for both model complexity and MSE that maximizes the model performance. We exploited Pareto Based Multi Objective Learning(PMOL) for estimating the optimal number of parameters for our autoencoder. It uses vector of objective functions and therefore number of optimal solutions are more than one. Pareto front of optimal solution is a set of non-dominated solutions, being chosen as optimal, if no objective can be improved without sacrificing at least one other objective. On the other hand a solution X is referred to as dominated by another solution Y if, and only if, Y is equally good or better than X with respect to all objectives. Pareto based multi object optimization can be formulated as eq. 20 with Q objective functions;

$$f(p) = [f_i(p), i = 1, ..., Q] \tag{20}$$

subjected to the equality constraints

$$g_j(p) = 0 \qquad j = 1, 2, ..., J \tag{21}$$

And the K inequality constraints

$$h_k(p) \leq 0 \qquad k = 1, 2, ..., K \tag{22}$$

The aim is to find vector p^* which minimizes $f(p)$, in our case since we have two objective function thus pareto based bi-objective learning problem can be formulated to minimize the two objectives, that is data fitting term and model complexity term given by eq. 23

$$f_1 = -\mathcal{L}(E|\Theta), \qquad f_2 = \gamma k \log(L) \tag{23}$$

where $f_1 = -\mathcal{L}(E|\Theta)$ is data fitting objective function and $f_2 = \gamma k \log(L)$ is model complexity objective function. f_1 is log-likelihood function that is found with a maximum likelihood estimation algorithm. $E = (\varepsilon_1, \varepsilon_2, ...\varepsilon_L)$ is a set of multi-dimensional reconstruction error. Assuming the error is multivariate normal distribution with Mean vector $M = 0$ and covariance Σ, then the distribution function is given by:

$$p(\varepsilon) = \mathcal{N}(M, \Sigma) = \frac{1}{2\pi\sqrt{|\Sigma n_f^2|}} \exp(-\frac{1}{2}\varepsilon^T \Sigma^{-1} \varepsilon) \tag{24}$$

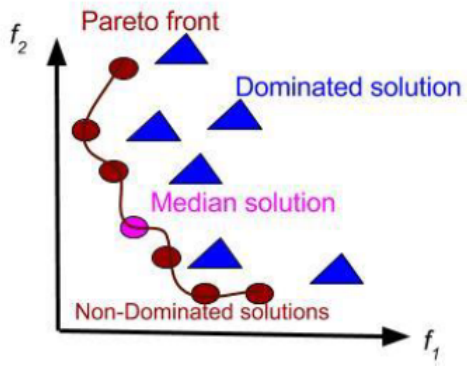


Fig. 5: Pareto front and the median solution

where the negative log-likelihood function will be represented by

$$-\mathcal{L}(E|\Theta) = \log \prod_{i=1}^{L} p(\varepsilon_i) \tag{25}$$

Assuming the features are identically independently distributed (iid), the covariance matrix Σ is given by;

$$\Sigma = \begin{bmatrix} \Omega_1^2 & 0 & 0 & \dots & 0 \\ 0 & \Omega_2^2 & 0 & \dots & 0 \\ 0 & 0 & \Omega_3^2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & \Omega_n^2 \end{bmatrix}$$

γ is a constant determined by a pareto optimizer, L is input training sample size, k is the number of parameters of the model to be estimated (weight and bias). Ω is the standard deviation.

Pareto-based multi objective learning algorithms are able to achieve a number of Pareto-optimal solutions, from which the user is able to extract knowledge about the problem and make a better decision when choosing the final solution. Once the pareto front optimal solutions are found, one optimal solution can be chosen using different methods for example the average, median etc. In this paper, we use median value as an optimal solution as shown in the fig. 5.

7. REGRESSION TECHNIQUES

Statistical regression is basically a way to predict unknown quantities from a batch of existing data. In all regression techniques the aim is to find 'the best' function that is estimated over training samples $(X_1, y_1), (X_n, y_n)$ in order to predict y given X, where $X \in \mathbb{R}^d$ and $y \in \mathbb{R}$. In our work, we have used the following three regression techniques for predicting the concentration of chemical component of interest in the given data sets.

- Linear regression (LR)

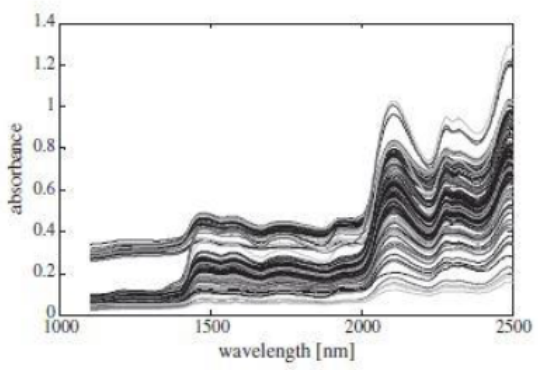


Fig. 6: Near-infrared spectra of orange juice training samples

- support vector regression (SVR)
- Gaussian process regression (GPR)

A detailed description of regression techniques is beyond the scope of this paper. Interested readers may refer to [39]. GPR worked well for our application because our data is non-linear and it fits very well for our problem.

8. EXPERIMENTS

8.1. Dataset description

In this work we have used two spectrophotometric datasets, coming from the food industry. The first dataset deals with determining sugar content in the orange juice sample by near infrared reflectance spectrometry [40]. The training and test samples in the orange juice data are 149 and 67 respectively; with 700 spectral variables that represents the absorbance ($\log 1/R$) between 1100nm and 2500nm. The value of R represents light reflected by the sample. The concentration of sugar ranges from 0% to 95.2% by weight in the sample. The spectra of orange juice obtained from the training set is shown in fig 6.

The second dataset deals with the determination of alcohol content by mid-infrared spectroscopy in wine samples [40]. The training and test data sets contain 91 and 30 spectra, respectively, with 256 spectral variables that are the absorbance ($\log 1/T$) at 256 wave numbers between 4000 and 400 cm^{-1} (where T is the light transmittance through the sample thickness). Alcohol content varies from 7.48% to 15.5% by volume. No preprocessing has been performed on the orange juice and wine datasets.

8.2. Estimation Error Assessment

For the quantitative results, the accuracy of the approach is represented in normalized mean square error (NMSE) metric,

Methods	**Orange juice**	**Wine**
Douak et al. [16]	01574	0.0070
Benoudjit et al. [15]	0.2435	0.0052
Alhichri et al. [21]	0.32076	0.0034
Ours	0.1711	0.0042

Table 1: Quantitative results of our method against 3 state-of-the-art methods. Numeric values shows Normalized Mean Square Error (NMSE), the lower is NMSE, the better is performance.

Dataset	**NMSE**
Orange Juice	5.3972
Wine	0.0517

Table 2: NMSE achieved on orange juice and wine dataset using all feature by linear regressor

which is define as:

$$NMSE = \frac{1}{V_{train+test}} \sum_{i=1}^{M} (y_{itest} - y'_{itest})^2 \qquad (26)$$

where M represents the number of testing samples, y_{itest} and y'_{itest} are the real and estimated outputs for the i^{th} test sample x_{itest} and $V_{train+test}$ is the combined variance of the training and test output samples y_{itrain} and y_{itest}.

8.3. Quantitative results

An extensive experiments are conducted to validate our approach. Initially, we used all the spectral features for both he datasets. Later, we used different number of features extracted through autoencoder neural networks for estimating the sugar and alcohol concentration in the samples.

8.3.1. All features

This section report the results obtained from different regressors by using ALL-features (original hyper dimensional input space), that is, 700 and 256 features for orange juice and wine data set, respectively.

For the ε-SVR, it was necessary to set three parameters, i.e., the width parameter of the kernel (γ), the regularization parameter (C), and the size of the insensitive loss tube (ε). In particular, C, γ and ε were varied from 10^4 to 10^3, from 10^4 to 10^3 and from 10^4 to 10^1, respectively. For the results given in Table 3, the choices of parameters were;

Dataset	**Kernel type**	**NMSE**
Orange juice	linear	0.1905
Orange juice	radial basis function	0.1820
wine	linear	0.0039
wine	radial basis function	0.0038

Table 3: NMSE achieved on orange juice and wine dataset using all feature by ε-SVR

Kernel function	**Basis function**	**Orange juice**	**Wine**
Squared exponential	constant	0.1316	0.0041
matern32	constant	0.1244	0.0038
matern32	constant	0.1313	0.0041
Squared exponential	linear	5.3972	0.0517
matern32	linear	5.3972	0.0517
matern32	linear	5.3972	0.0517

Table 4: NMSE achieved on orange juice and wine dataset using all feature by GPR

Parameter for orange juice:

- Linear: C = 183.29, $\varepsilon = 10^{-2}$
- Radial Basis Function: C= 103, $\varepsilon = 10^{3}$, γ = 0.3

Parameter for wine:

- Linear: C = 10^{3}, $\varepsilon = 10^{-2}$
- Radial Basis Function: C= 10^{3}, $\varepsilon = 10^{-2}$, γ = 0.15

8.3.2. Deep features through autoencoder

This section report the results obtained from three regressors using features that extracted through autoencoder neural network. The number of hidden units used in the autoencoder architecture are 2, 5, 10, 20,..., 200 with an increment value of 10 between 10 and 200 nodes. The encoder and decoder transfer functions used are 'satlin' [41] and 'logsig' [42].

Orange juice dataset

For the orange juice dataset, 700 spectral features are reduced to 2, 5, 10, 20,..., 200 features, with an increment value of 10. The extracted features (2, 5, 10, 20,..., 200 with an increment value of 10) of 149 training samples are used one by one to train the regression model of Linear, Gaussian process and support vector regressors. For testing, 67 samples from test data set is used and the performance of each regressor is shown in terms of normalized means square error NMSE value. In table 5, the best NMSE values achieved on three regressors are reported.

Regressor	**Number of features**	**NMSE**
Linear	140	0.1368
Gaussian process	50	0.1711
ε-SVR	160	0.3038

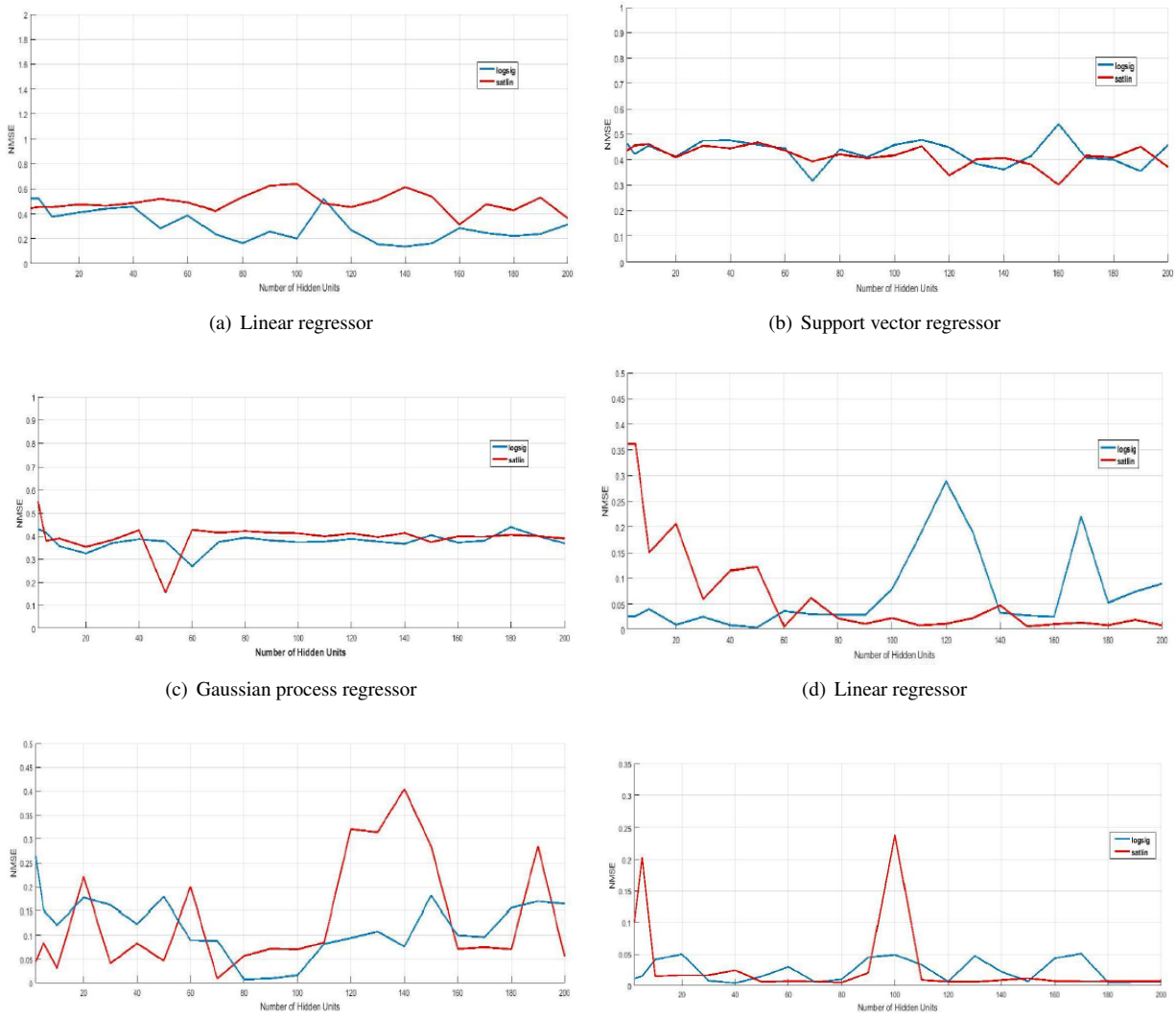
Table 5: NMSE achieved on orange juice dataset using deep feature

In case of GPR, the square exponential kernel function is adopted with 'constant' basis function. For SVR, the result reported in table 5 is achieved using 'radial basis function'. kernel function with regularization parameter C = 103, $\varepsilon = 10^{-2}$ and kernel width γ = 0.1. In particular, C, γ and ε were varied from 10^{-4} to 10^{3}, from 10^{-4} to 10^{3} and from 10^{-4} to 10^{-1}, respectively. The result obtained using linear kernel function is quiet close to the one reported in table 5, which is NMSE = 0.3084. In fig. 7 (a-c), we demonstrate the varying behavior of NMSE over different hidden units used in autoencoder during feature extraction process for all the three regressors.

In case of linear regressor, the performance achieved using 'logsig' activation function for hidden units or nodes in encoder and decoder of autoencoder is more consistent and better than the one achieved used 'satlin', as number of hidden nodes increases. While for GPR and SVR, the behavior of NMSE achieved using 'logsig' and 'satlin' activation functions is consistent and more stable over the set of hidden units chosen for the feature extraction through autoencoder.

8.3.3. Autoencoder parameter optimization for orange juice

To determine the optimal number of hidden nodes in the network, we use pareto dominance optimization technique to optimizes the trade-off between number of hidden nodes and model complexity. Figure 8 shows a plot of dominated and non-dominated solution using pareto dominance technique for orange juice data set. Based on pareto optimal solution, the number of hidden nodes are 40. Once the optimal number of parameters for the network is found, we use them to form a deep neural network. The models are built by cascading SAE. We experimented with 2 hidden layers (stacked autoencoder(AE)). From table 6, the overall NMSE decrease comparing to the values obtained using single layer,



(a) Linear regressor

(b) Support vector regressor

(c) Gaussian process regressor

(d) Linear regressor

(e) Support vector regressor

(f) Gaussian process regressor

Fig. 7: NMSEs achieved for (a) Linear, (b) Support vector and (c) Gaussian process regressor for orange juice dataset. NMSEs achieved for (d) Linear, (e) Support vector and (f) Gaussian process regressor for wine dataset.

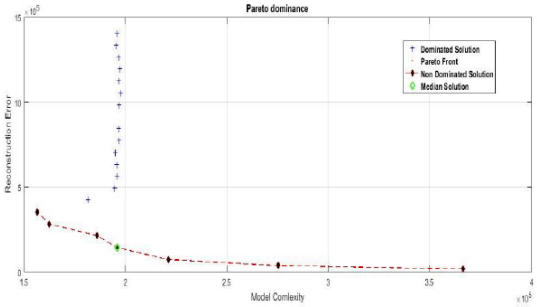


Fig. 23. Pareto Dominance for orange juice dataset

Fig. 8: Pareto dominance for orange juice dataset

Regressor	**Number of hidden units in first layer**	**NMSE using 1 hidden layer**	**Number of hidden units in second layer**	**NMSE using 2 hidden layers**
Linear	140	0.1368	100	0.3086
Gaussian process	50	0.1711	150	0.4098
ε-SVR	160	0.3038	100	0.4059

Table 6: Comparison of best NMSEs achieved using one and two hidden layers for orange juice dataset

indicating increasing the hidden layer have negative effect in the model. This is due to the fact that we already selected an optimal number of hidden units in the first layer that best represents the network architecture.

8.3.4. Wine dataset

In wine dataset, the number of spectral features are 256 which are less as compare to the ones in orange juice dataset. Like orange juice dataset experiment, the extracted features (2, 5, 10, 20,..., 200 with an increment value of 10) of 91 training samples are used one by one to train the regression model of Linear, Gaussian process and support vector regressors. For testing, 30 samples from test data set is used and the performance of each regressor is shown in terms of normalized means square error NMSE value. In table 7, the best NMSE values achieved on three regressors are reported.

In fig. 7(d-f), we demonstrate the varying behavior of NMSE over different hidden units used in autoencoder during feature extraction process for all the three regressors.

Regressor	**Number of features**	**NMSE**
Linear	50	0.0039
Gaussian process	40	0.0042
ε-SVR	80	0.0075

Table 7: NMSE achieved on wine dataset using deep feature

Regressor	**Number of hidden units in first layer**	**NMSE using 1 hidden layer**	**Number of hidden units in second layer**	**NMSE using 2 hidden layers**
Linear	50	0.0039	30	0.0035
Gaussian process	40	0.0043	30	0.0137
ε-SVR	80	0.0075	100	0.0241

Table 8: Comparison of best NMSEs achieved using one and two hidden layers for wine dataset

8.3.5. Autoencoder parameter optimization for wine dataset

Just like orange dataset, we did quantitative analysis through deep learning by using 2 hidden layers (stacked AE). From table 11, the overall NMSE decrease comparing to the values obtained using single layer for GPR and SVR, indicating increasing the hidden layer have negative effect on the model. This is due to the fact that we already selected an optimal number of hidden units in the first layer which represents the network architecture optimally.

To determine the optimal number of hidden nodes in the network, we use pareto dominance optimization technique to optimizes the trade-off between number of hidden nodes and model complexity as we did for the orange juice dataset. The fig. 9 shows a plot of dominated and non-dominated solution using pareto dominance technique for wine data.

It can be seen from Table 5 and 7 that in contrast to the concentration of alcohol in the wine samples, the estimation of sugar concentration appears more difficult despite a larger number of spectral data and training samples. This is due to high nonlinearity of the spectral signature of this chemical component.

8.4. Comparsion with state-of-the-art

In this section we summarize the results of the two datasets and comparison can be made with the given state of art results.

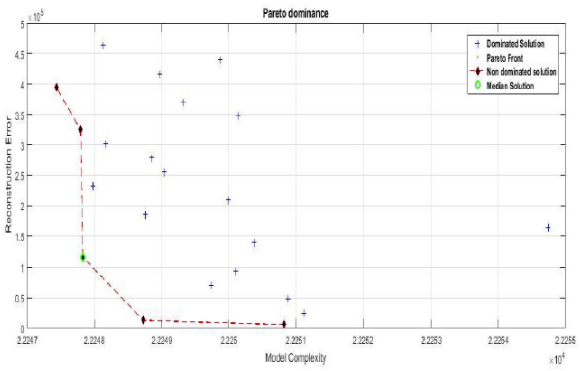


Fig. 9: Pareto dominance for wine dataset

Regressor methods	**NMSE values for orange juice dataset**	**NMSE values for wine dataset**
PLSR	0.161	0.0053
RBFN	0.1574	0.007
SVM	0.2497	0.0081

Table 9: NMSE values achieved on 3 regression methods implemented with residual based correction (RBC) [16]

9. CONCLUSION

We propose a novel deep learning based chemometric data analysis technique for estimating sugar and alcohol concentration in orange juice and wine samples. L2 regularized sparse autoencoder is trained end-to-end for reducing the size of the feature vector to handle the classic problem of curse of dimensionality in chemometric data analysis. The optimal set of parameters of the autoencoder are selected through pareto optimization. Three regressor namely, Gaussian process, support vector and linear regressor are applied on the reduced size feature vector for estimating the concentration of sugar and alcohol in the corresponding samples. Extensive quantitative analysis are conducted with different configuration and the results are compared with state-of-the-art methods. Quantitative results are shown on Normalized Mean Square Error (NMSE) and our approach shows better results on both datasets.

Regressor methods	**NMSE values for orange juice dataset**	**NMSE values for wine dataset**
Linear	0.1368	0.0039
GPR	0.1711	0.0042
ε-SVR	0.3038	0.0075

Table 10: NMSE values for both datasets using number of extracted features suggested by pareto optimization.

10. REFERENCES

[1] Svante Wold, "Chemometrics; what do we mean with it, and what do we want from it?," *Chemometrics and Intelligent Laboratory Systems*, vol. 30, no. 1, pp. 109–115, 1995.

[2] Jon M Sutter and John H Kalivas, "Comparison of forward selection, backward elimination, and generalized simulated annealing for variable selection," *Microchemical journal*, vol. 47, no. 1-2, pp. 60–66, 1993.

[3] Dominique Bertrand, "La spectroscopie proche infrarouge et ses applications dans les industries de lalimentation animale," *Productions Animales 3 (15), 209-219.(2002)*, 2002.

[4] David A Belsley, Edwin Kuh, and Roy E Welsch, *Regression diagnostics: Identifying influential data and sources of collinearity*, vol. 571, John Wiley & Sons, 2005.

[5] Tomas Eklöv, Per Mårtensson, and Ingemar Lundström, "Selection of variables for interpreting multivariate gas sensor data," *Analytica Chimica Acta*, vol. 381, no. 2-3, pp. 221–232, 1999.

[6] Paul Geladi, "Some recent trends in the calibration literature," *Chemometrics and Intelligent Laboratory Systems*, vol. 60, no. 1-2, pp. 211–224, 2002.

[7] Harald Martens and Paul Geladi, *Multivariate calibration*, Wiley Online Library, 1989.

[8] Michel Verleysen et al., "Learning high-dimensional data," 2003.

[9] John A Cornell, "Classical and modern regression with applications," 1987.

[10] Nabil Benoudjit, E Cools, Marc Meurens, and Michel Verleysen, "Chemometric calibration of infrared spectrometers: selection and validation of variables by non-linear models," *Chemometrics and Intelligent Laboratory Systems*, vol. 70, no. 1, pp. 47–53, 2004.

[11] Nabil Benoudjit, Damien François, Marc Meurens, and Michel Verleysen, "Spectrophotometric variable selection by mutual information," *Chemometrics and Intelligent Laboratory Systems*, vol. 74, no. 2, pp. 243–251, 2004.

[12] Fabrice Rossi, Amaury Lendasse, Damien François, Vincent Wertz, and Michel Verleysen, "Mutual information for the selection of relevant variables in spectrometric nonlinear modelling," *Chemometrics and intelligent laboratory systems*, vol. 80, no. 2, pp. 215–226, 2006.

[13] Svante Wold, Kim Esbensen, and Paul Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[14] Paman Gujral, Michael Amrhein, Rolf Ergon, Barry M Wise, and Dominique Bonvin, "On multivariate calibration with unlabeled data," *Journal of Chemometrics*, vol. 25, no. 8, pp. 456–465, 2011.

Regressor methods	**NMSE values for orange juice dataset**	**NMSE values for wine dataset**
PCR	0.2596	0.003
PLSR	0.2435	0.0052
VW-PLS	0.2315	0.0031
MRS with linear regressors (AFS)	0.2334	0.0038
MRS with linear regressors (WFS)	0.2354	0.0033
MRS with linear regressors (NLFS)	0.1461	0.0030
MRS with RBFN regressors (AFS)	0.3123	0.0038
MRS with linear regressors (WFS)	0.1806	0.0028
MRS with linear regressors (NLFS)	0.1742	0.0026

Table 11: Results achieved on both the wine and orange juice datasets by three reference regression methods and proposed MRS approach implemented with UPS technique [15]

Regressor methods	**NMSE values for orange juice dataset**	**NMSE values for wine dataset**
Fusion average	0.31978	0.00329
WCS	0.31844	0.00277
IOWA	0.1355	0.00265

Table 12: Results for orange juice and wine datasets [21]

[15] N Benoudjit, F Melgani, and H Bouzgou, "Multiple regression systems for spectrophotometric data analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 95, no. 2, pp. 144–149, 2009.

[16] F Douak, N Benoudjit, and F Melgani, "A two-stage regression approach for spectroscopic quantitative analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 109, no. 1, pp. 34–41, 2011.

[17] Xiaojin Zhu, "Semi-supervised learning literature survey," 2005.

[18] Deirdre Toher, Gerard Downey, and Thomas Brendan Murphy, "Semi-supervised linear discriminant analysis," *Journal of Chemometrics*, vol. 25, no. 12, pp. 621–630, 2011.

[19] Burr Settles, "Active learning literature survey. 2010," *Computer Sciences Technical Report*, vol. 1648.

[20] Fouzi Douak, Farid Melgani, Naif Alajlan, Edoardo Pasolli, Yakoub Bazi, and Nabil Benoudjit, "Active learning for spectroscopic data regression," *Journal of Chemometrics*, vol. 26, no. 7, pp. 374–383, 2012.

[21] Haikel AlHichri, Yakoub Bazi, Naif Alajlan, Farid Melgani, Salim Malek, and Ronald R Yager, "A novel fusion approach based on induced ordered weighted averaging operators for chemometric data analysis," *Journal of Chemometrics*, vol. 27, no. 12, pp. 447–456, 2013.

[22] Habib Ullah and Nicola Conci, "Crowd motion segmentation and anomaly detection via multi-label optimization," in *ICPR workshop on pattern recognition and crowd analysis*, 2012.

[23] Habib Ullah and Nicola Conci, "Structured learning for crowd motion segmentation," in *2013 IEEE International Conference on Image Processing*. IEEE, 2013, pp. 824–828.

[24] Mohib Ullah, Habib Ullah, Nicola Conci, and Francesco GB De Natale, "Crowd behavior identification," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 1195–1199.

[25] Habib Ullah, Muhammad Uzair, Mohib Ullah, Asif Khan, Ayaz Ahmad, and Wilayat Khan, "Density independent hydrodynamics model for crowd coherency detection," *Neurocomputing*, vol. 242, pp. 28–39, 2017.

[26] Paolo Rota, Habib Ullah, Nicola Conci, Nicu Sebe, and Francesco GB De Natale, "Particles cross-influence for entity grouping," in *21st European Signal Processing Conference (EUSIPCO 2013)*. IEEE, 2013, pp. 1–5.

[27] Habib Ullah, *Crowd Motion Analysis: Segmentation, Anomaly Detection, and Behavior Classification*, Ph.D. thesis, University of Trento, 2015.

[28] Fawad Ahmad, Asif Khan, Ihtesham Ul Islam, Muhammad Uzair, and Habib Ullah, "Illumination normalization using independent component analysis and filtering," *The Imaging Science Journal*, vol. 65, no. 5, pp. 308–313, 2017.

[29] Habib Ullah, Mohib Ullah, and Muhammad Uzair, "A hybrid social influence model for pedestrian motion segmentation," *Neural Computing and Applications*, pp. 1–17, 2018.

[30] Habib Ullah, Mohib Ullah, and Nicola Conci, "Dominant motion analysis in regular and irregular crowd scenes," in *International Workshop on Human Behavior Understanding*. Springer, 2014, pp. 62–72.

[31] Habib Ullah, Ahmed B Altamimi, Muhammad Uzair, and Mohib Ullah, "Anomalous entities detection and localization in pedestrian flows," *Neurocomputing*, vol. 290, pp. 74–86, 2018.

[32] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi, "Inception-v4, inception-resnet and the

impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[33] Mohib Ullah, Ahmed Mohammed, and Faouzi Alaya Cheikh, "Pednet: A spatio-temporal deep convolutional neural network for pedestrian segmentation," *Journal of Imaging*, vol. 4, no. 9, pp. 107, 2018.

[34] Mohib Ullah and Faouzi Alaya Cheikh, "Deep feature based end-to-end transportation network for multi-target tracking," in *IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 3738–3742.

[35] Mohib Ullah and Faouzi Alaya Cheikh, "A directed sparse graphical model for multi-target tracking," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1816–1823.

[36] Mohib Ullah, Ahmed Kedir Mohammed, Faouzi Alaya Cheikh, and Zhaohui Wang, "A hierarchical feature model for multi-target tracking," in *IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 2612–2616.

[37] Habib Ullah, Muhammad Uzair, Arif Mahmood, Mohib Ullah, Sultan Daud Khan, and Faouzi Alaya Cheikh, "Internal emotion classification using eeg signal with sparse discriminative ensemble," *IEEE Access*, vol. 7, pp. 40144–40153, 2019.

[38] Pierre Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 2012, pp. 37–49.

[39] Carl Edward Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*, pp. 63–71. Springer, 2004.

[40] Marc Meurens, "Spectrophotometric samples of orange juice and wine for chemometric analysis," .

[41] HK Chang and WA Chien, "Neural network with multi-trend simulating transfer function for forecasting typhoon wave," *Advances in Engineering Software*, vol. 37, no. 3, pp. 184–194, 2006.

[42] Mohammad Dorofki, Ahmed H Elshafie, Othman Jaafar, Othman A Karim, and Sharifah Mastura, "Comparison of artificial neural network transfer functions abilities to simulate extreme runoff data," *International Proceedings of Chemical, Biological and Environmental Engineering*, vol. 33, pp. 39–44, 2012.