Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Electrotécnica e de Computadores

Vasco da Silva Ferreira

Semi-Supervised Learning and Feature Ranking/Selection Techniques Applied to Soft Sensor Modeling

Dissertação de Mestrado Integrado em Engenharia Electrotécnica e de Computadores.

Coimbra

September, 2017

Universidade de Coimbra

Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Electrotécnica e de Computadores

Semi-Supervised Learning and Feature Ranking/Selection Techniques Applied to Soft Sensor Modeling

Vasco da Silva Ferreira

Dissertation submitted to the Electrical and Computer Engineering Department of the Faculty of Science and Technology of the University of Coimbra in partial fulfillment of the requirements for the Degree of Master of Science in Electrical and Computer Engineering.

Supervisor: Prof. Doutor Rui Alexandre de Matos Araújo
Co-Supervisor: Prof. Doutor Francisco Alexandre Andrade de Souza

Jury: Prof. Doutor Manuel Marques Crisóstomo
Prof. Doutor Lino José Forte Marques
Prof. Doutor Rui Alexandre de Matos Araújo

September, 2017

"Progress is made by trial and failure; the failures are generally a hundred times more numerous than the successes; yet they are usually left unchronicled."
Sir William Ramsay

Acknowledgements

I would first like to express my sincere gratitude to my advisors Prof. Doutor Rui Alexandre de Matos Araújo and Prof. Doutor Francisco Alexandre Andrade de Souza. Their valuable guidance, continuous support and scientific rigor helped me not only in achieving my goals but also in expanding my knowledge and critical thinking abilities.

Next, I would also like to acknowledge the Institute of Systems and Robotics for providing all resources needed during this final stage of my master's degree.

To all my friends that were present during these past years, the memories we shared in this beautiful city will accompany me for the rest of my life.

Special thanks to my girlfriend, Verónica, for all the support and belief even at the harder stages of writing this dissertation.

Last but certainly not least, I must express my very profound gratitude to my family, specially my parents Jaime and Clarinda, my sister Rita, and my grandfather Fernando. Their strong presence and great values helped immensely in my years of study.

Resumo

Sensores virtuais são modelos inferenciais baseados em software que usam variáveis de processo (disponíveis através de sensores), também conhecidas como variáveis "fáceis de medir", para estimar o valor de variáveis de qualidade (variáveis "difíceis de medir"), que não podem ser facilmente medidas, ou o seu processo de medição tem um alto custo associado (p.e. apenas pode ser feito esporadicamente, ou com atrasos temporais elevados). Os processos industriais estão geralmente equipados com um elevado número de sensores, medindo uma grande variedade de quantidades diferentes (p.e. temperatura, fluxo, abertura de válvulas, etc), disponíveis em tempo real a uma frequência constante. No entanto, em algumas circunstâncias, o valor das variáveis de qualidade só pode ser obtido através de análises laboratoriais, levando a uma frequência de medição não-fixa e atrasos temporais substanciais. Estes problemas podem levar a degradação de qualidade do produto final. Os sensores virtuais podem, nestes casos, possibilitar um maior grau de controlabilidade do processo através da disponibilização de estimações precisas dessas variáveis de qualidade.

A frequência de amostragem significativamente elevada das variáveis "fáceis de medir" quando comparada com a das variáveis "difíceis de medir" leva a que muitas amostras sejam descartadas na etapa de obtenção e filtragem de dados da modelação de sensores virtuais. Isto acontece já que a maior parte dos modelos usam abordagens de aprendizagem supervisionada, nas quais apenas as amostras com respetiva etiqueta (i.e. amostras para as quais as variáveis de processo têm correspondentes variáveis de qualidade) são usadas no processo de treino. Abordagens de aprendizagem semi-supervisionada, no entanto, usam tanto amostras com e sem etiqueta no processo de treino. Nesta dissertação, um método semi-supervisionado baseado em aprendizagem usando múltiplas vistas, regressão de mínimos quadrados co-regularizada (coRLSR), é implementado, usando também dados sem etiqueta para melhorar o desempenho de previsão.

Outra etapa muito importante na modelação de sensores virtuais é a seleção de características. O elevado número de sensores numa planta de produção leva a um elevado número de possíveis características de entrada, aumentando a complexidade global do problema de regressão. Na maior parte dos casos, muitas características apresentam correlações com outras, e o uso de todas para treino do modelo pode causar a deterioração do desempenho de previsão. De facto, a maior parte do trabalho de pesquisa científica neste tópico sugere que, em muitos casos, poucas características são precisas para estimação suficientemente precisa.

As abordagens descritas na literatura para classificação e seleção de características partilham um grau relativamente elevado de complexidade, que pode tornar o seu uso proibitivo em cenários em que o desempenho temporal é importante. Um método geral para classificação de características baseado em análise de sensibilidade é proposto nesta dissertação, de modo a que, de forma eficiente, a relevância de cada característica seja calculada sem a necessidade de retreinar o modelo.

Os testes foram efetuados num processo real de polimerização, de forma a avaliar ambas as técnicas de aprendizagem semi-supervisionada e de classificação/seleção de características. Os resultados mostraram que o modelo de regressão semi-supervisionada foi competitivo com os métodos de aprendizagem supervisionada mais populares de sensores virtuais quando nenhum procedimento de seleção de características foi efetuado. No entanto, usando o procedimento de seleção de características, o método de regressão semi-supervisionada implementado não ultrapassou as alternativas supervisionadas em desempenho de predição. Por outro lado, o procedimento de classificação e seleção de características proposto aumentou substancialmente o desempenho de predição de todos os modelos de regressão estudados. Para além disso, o conhecimento obtido pela classificação de características permite um subsequente aumento de desempenho de predição, quando usado explorando a natureza de múltiplas vistas do método coRLSR.

Palavras-chave: Sensores Virtuais, Inteligência Computacional, Aprendizagem Computacional, Aprendizagem Semi-Supervisionada, Estimação, Classificação de Características, Seleção de Características.

Abstract

Soft sensors are software-based inferential models that use process variables (available from online sensors), also known as easy-to-measure variables, to predict quality variables (hard-to-measure variables), which cannot be easily measured, or its measurement has high associated cost (e.g. can be only done sporadically, or with high delays). Industrial processes are generally equipped with a large number of sensors measuring a large variety of different quantities (e.g. temperature, flow rate, valve openings, etc), available in real-time at a constant frequency. However, in some settings, quality variables are only available by laboratory analysis (for example), leading to non-fixed measurement frequency and substantial delays. These issues can lead to quality degradation of the final product. Soft sensors can, in such cases, enable a higher degree of controllability of the process by providing accurate online estimations of those quality variables.

The significantly higher sampling rate of the easy-to-measure variables when comparing to that of the hard-to-measure variables leads to many samples being discarded at the data collection and filtering stage of soft sensor modeling. This happens since most models use supervised learning approaches, in which only the labeled samples (i.e. samples for which the process variables have corresponding quality values) are used in training. Semi-supervised learning approaches, however, use both labeled and unlabeled samples in training. In this dissertation, co-regularized least squares regression (coRLSR), a semi-supervised method based on multi-view learning, is implemented, using also unlabeled data to improve predictive performance.

Another very important stage in soft sensor modeling is feature selection. The large number of online sensors in a processing plant equates to a large number of possible input features, raising the overall complexity of the regression problem. In most cases, many features have correlations with one another, and the use of the entire available feature set for model training can deteriorate predictive performance. In fact, most of the research performed on this topic suggests that, in many cases, only few features are needed for sufficiently accurate predictions. The approaches described in the literature for feature ranking and selection share a relatively high degree of complexity, which can be prohibitive in time-sensitive scenarios. A general method for feature ranking based on sensitivity analysis is proposed in this dissertation, in order to efficiently compute each feature's relevance without retraining the model.

Testing was performed on a real-world polymerization batch process, in order to evaluate both techniques of semi-supervised learning and feature ranking/selection. Results showed that the semi-supervised regression model was competitive with the most popular soft sensor supervised model approaches when no feature selection procedure was performed. However, when performing feature selection, the implemented semi-supervised regression method did not surpass the supervised approaches in predictive performance. On the other hand, the proposed feature ranking and selection procedure substantially improved the predictive performance of all regression models considered. Furthermore, its capabilities were extended when using the feature ranking knowledge with the multi-view nature of coRLSR, enabling a subsequent improvement in predictive performance.

Keywords: Soft Sensors, Computational Intelligence, Computational Learning, Supervised Learning, Semi-Supervised Learning, Estimation, Feature Ranking, Feature Selection.

Symbols and Abbreviations

List of Acronyms

ANN	*Artificial Neural Network*
coRLSR	*co-Regularized Least Squares Regression*
EM	*Expectation Maximization*
FIR	*Finite Impulse Response*
kNN	*k Nearest Neighbors*
LS-SVM	*Least Squares Support Vector Machine*
MSE	*Mean Squared Error*
NN	*Neural Network*
NRMSE	*Normalized Root Mean Squared Error*
PLS	*Partial Least Squares*
RFE	*Recursive Feature Elimination*
RLSR	*Regularized Least Squares Regression*
RMSECV	*Root Mean Square Error of Cross Validation*
RMSE	*Root Mean Squared Error*
SSL	*Semi-Supervised Learning*
SS	*Soft Sensor*
SVM	*Support Vector Machine*

State of The Art and Background

n	*number of samples*
X	*set of samples*
x_i	*sample at instant i*
y_i	*label at instant i*
$\mathcal{X}$	*distribution of X*
$\mathcal{Y}$	*distribution of Y*

Co-Regularized Least Squares Regression

att	$\in \mathbb{R}^{M \times d}$ *logical matrix that contains the attribute/feature split for all views*
$\mathbf{C}$	*vector containing the solution coefficients for all views*
$\mathbf{c}_v$	*vector containing the solution coefficients for view* v
d	*number of input variables/features*
$\mathbf{f}_v$	$\in \mathbb{R}^{(N_v+Z)}$ *vector of predictions (by predictor* f_v*) for all labeled and unlabeled samples of view* v
f_v	*predictor of view* v
$\mathbf{G}_v$	*regularization matrix of view* v
$inst$	$\in \mathbb{R}^{M \times (N+Z)}$ *logical matrix that contains the instance/example split for all views*
$\mathbf{K}$	*kernel matrix of view* v
$k_v(\cdot,\cdot)$	*kernel function of view* v
$\mathbf{L}_v$	*labeled sub-matrix of the kernel matrix* $\mathbf{K}_v$
M	*total number of views*
N	*number of labeled samples*
N_v	*number of labeled samples in view* v
$\mathbf{U}_v$	*unlabeled sub-matrix of the kernel matrix* $\mathbf{K}_v$
X_j	j*-th feature/variable of* $\mathbf{X}$
$\mathbf{X_{est}}$	*matrix of samples used to compute the estimations* $\hat{\mathbf{y}}$
$\mathbf{X}$	$\in \mathbb{R}^{(N+Z) \times d}$ *matrix of training samples*
$\mathbf{x}_i$	$\in \mathbb{R}^d$ *input vector at instant* i
$\hat{\mathbf{y}}$	*estimates for the* $\mathbf{X_{est}}$ *matrix of samples*
$\mathbf{y}$	$\in \mathbb{R}^{(N+Z)}$ *vector of training labels*
$\mathbf{y}_v$	$\in \mathbb{R}^{N_v}$ *vector of labels in view* v
y_i	*label at instant* i
Z	*number of unlabeled samples*
Γ	*set of labeled samples*
$\mathcal{H}$	*Hilbert Space of functions*
$\mathcal{L}(\cdot,\cdot)$	*convex loss function*
λ	*tunable parameter*
ν	*regularization parameter*
ν_v	*regularization parameter of view* v
$\Omega[\cdot]$	*regularization term*
$\mathbf{\Phi}$	*set of training data*
$\mathbf{\Phi}_z^*$	*set of unlabeled samples*
$\mathbf{\Phi}_v$	*set of samples of view* v
σ	$\in \mathbb{R}^M$ *vector containing a kernel parameter for all views (standard deviation)*

Feature Ranking and Selection

d	*number of input variables*
E	*performance metric computed on the predictions of* f
$E_{(j)}$	*performance metric computed on the predictions of* $f_{(j)}$
f	*predictor trained with the* X *set*
$f_{(j)}$	*predictor trained with the permuted* $X_{(j)}$ *set*
$J(X_j)$	*rank of the* j*-th feature*
X	*input set of variables*
X_j	j*-th input variable*
$X_{(j)}$	*input set with the* j*-th variable permuted*
X_{-j}	*input set without the* j*-th variable*
$\mathbf{x}_i$	$\in \mathbb{R}^d$ *input vector at instant* i
$\hat{\mathbf{y}}$	*estimates computed using the intact* X *set*
$\hat{\mathbf{y}}_{(j)}$	*estimates computed using the perturbed* $X_{(j)}$ *set*
y	*output variable*
ϵ	*threshold*

Test and Results

f_v	*predictor of view* v
$k_v(\cdot,\cdot)$	*kernel function of view* v
M	*total number of views*
N_v	*number of labeled samples in view* v
n	*number of samples*
$\mathbf{x}_i$	*vector of input data at instant* i
$\overline{y}$	*average observed value of* y
y_i	*label at instant* i
$y_{\max}$	*maximum observed value of* y
$y_{\min}$	*minimum observed value of* y
$\mathcal{L}(\cdot,\cdot)$	*convex loss function*
λ	*tunable parameter*
ν_v	*regularization parameter of view* v
σ_v	*kernel parameter of view* v *(standard deviation)*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Context

Industrial processes are generally equipped with a large number of online sensors that measure a multitude of process variables (e.g. temperature, flow rate, etc) whose values can be obtained at a constant frequency with low cost. However, in many cases, quality variables are not available in real time, being measured sporadically, at a non-fixed frequency or with high delays (e.g. laboratory measurements). Soft sensors are software-based inferential models that use process variables to predict quality variables (for example), making them available at a higher frequency. This is particularly useful in settings that require laboratory analysis in order to measure product quality. The delay associated with these analysis can lead to quality degradation of the final product, proving the value that a soft sensor can have by providing accurate online estimates.

Most soft sensors use supervised methods to construct the above mentioned relationship between process (often easy-to-measure) and quality (often hard-to-measure) variables. These approaches only use labeled data (i.e. samples for which the process variables have corresponding quality value(s)) during the training stage. Semi-supervised methods, on the other hand, use both labeled and unlabeled data (i.e. samples for which the process variables do not have corresponding quality value(s)) in order to construct the model. Since the sampling rate of the process variables is much higher than the frequency of laboratory analysis, the number of unlabeled samples far exceeds the number of labeled ones, creating a setting that can favor semi-supervised methods.

The large number of different process variables in the industrial context can cause not only the discrepancy in the number of available labeled and unlabeled samples, but also the rising complexity of the regression problem. One of the simplest techniques for dimensionality reduction is to select a subset of the input variables/features, and discard the remaining ones. Frequently, there are features which carry little to none useful information for the solution of the problem. Also, some sets of features can show very strong correlations between each

other so that the same information is repeated in several features. For this reason, feature selection is a very important step in soft sensor design, as the presence of irrelevant features can decrease a model's predictive performance. The feature selection approaches in the literature frequently require a heavy computational effort, increasing the model development time. Focusing on lower computational cost, a feature ranking and selection method is devised. Based on sensitivity analysis, the rank of a feature can be obtained by perturbing a specific variable/feature and measuring the variation of the output, without the need of retraining the model. Then, a procedure of feature selection can be performed to select the subset that guarantees the best performance.

In this dissertation, co-Regularized Least Squares Regression (coRLSR), a semi-supervised approach based on Regularized Least Squares Regression (RLSR), is used to compare against some of the most popular supervised methods, such as Partial Least Squares (PLS), Support Vector Machines (SVM) and Neural Networks (NN). Furthermore, the devised algorithm of feature ranking and selection is used to improve not only coRLSR's performance but also the remaining mentioned approaches.

1.2 Goals

This dissertation aims to explore techniques of semi-supervised learning and feature ranking and selection to the soft sensor modeling context, inferring on the advantages brought by the studied approaches to predictive performance. The goals of the work are:

1. To implement an algorithm based on a semi-supervised multi-view approach capable of performing regression on a set of training data;

2. To develop and implement a feature ranking and selection method focusing on flexible adaptation and low computational cost;

3. To conduct tests on a real-world dataset in order to evaluate the effectiveness of the implemented algorithms.

1.3 Implementations and Key Contributions

All required practical work was done in the *MATLAB*® computation environment. Several scenarios were tested in order to analyze and validate the relevance of not only the methods for semi-supervised regression learning presented in Chapter 3, but also other supervised methods for regression in the soft sensor context. CoRLSR was implemented as a set of *MATLAB*® functions, in order to test and compare its predictive performance. Some coding optimizations were performed, leading to a decrease in runtime when compared to the original paper's results, further expanding coRLSR's applicability. Moreover, a feature ranking approach was devised, as presented in Chapter 4, constituting the most prominent

contribution of this dissertation. The proposed approach, based on sentitivity analysis, focuses on low complexity, leading to a computationally efficient procedure to rank features. The general nature of the procedure enables its use with any regression method, widening its applicability. The introduction of a parameter in the feature selection step lowers the computational demand when compared to the methods for feature ranking proposed in the literature. A real-world scenario was considered in order to infer the effectiveness of both the methods for semi-supervised regression learning and the feature ranking/selection techniques in the industrial context. The studied process belongs to the chemical industry field, where the advantages of soft sensors are well known. Furthermore, the nature of chemical industrial processes usually leads to the presence of large amounts of unlabeled data and many candidate input variables for regression. This setting allows for a deeper understanding on how the considered techniques improve the current state of soft sensor development.

From the work of this dissertation it also resulted the paper "Semi-Supervised Soft Sensor and Feature Ranking Based on Co-Regularised Least Squares Regression Applied to a Polymerization Batch Process" [Ferreira *et al.*, 2017], which is included in Appendix A, and that was published in the "IEEE 15th International Conference on Industrial Informatics (INDIN 2017)", that was held on July 24-26, 2017, in Emden, Germany.

1.4 Structure

This dissertation is divided into six chapters, the content of each being the following:

- In Chapter 1, the context and motivation about the studied topic is stated, as well as the main goals of the dissertation;
- In Chapter 2, a brief background on semi-supervised learning, co-learning and multi-view approaches is provided and a deeper insight into soft sensor modeling is given;
- In Chapter 3, the semi-supervised approach to soft sensor modeling is described, as well as its implementation;
- In Chapter 4, the devised method for feature ranking and selection is explained, along with its theoretical proof and practical implementation;
- In Chapter 5, several experiments for testing and validation of the methods stated in Chapters 3 and 4 are performed and their results are presented and discussed;
- In Chapter 6, the general conclusions are presented and the possible future work is discussed.

Chapter 2

State of the Art and Background

This chapter gives an overview of different types of paradigms in machine learning, as well as a brief history of semi-supervised learning, in order to clarify its key concepts. Then, a more in-depth look to co-training and multi-view methods is presented as it provides the theoretical basis to the semi-supervised method coRLSR. Finally, an insight on soft sensor development and state of the art model approaches is presented.

2.1 Machine Learning Paradigms

In machine learning there are two main entities: the teacher and the learner. The teacher has the knowledge required to perform a given task, and the learner's goal is to learn the knowledge to perform the task [Lampropoulos and Tsihrintzis, 2015]. The effort is then typically divided by these two entities: the more the inference capabilities of the learner the less the effort of the teacher and vice-versa. The most important type of learning is *Learning from Examples*, in which the teacher provides a set of observations to the learner so that he can infer on new samples. This type of learning can be discriminated in three categories: *unsupervised learning*, *supervised learning* and *reinforcement learning*.

Consider $X = (x_1, \ldots, x_n)$ a set of n samples (or examples), assuming that all samples are drawn i.i.d. (independently and identically distributed) from a common distribution on $\mathcal{X}$. The goal of the learner in *unsupervised learning* is to find interesting structure in the data X. This problem can be either that of discovering groups of similar examples within the data (clustering), determining the distribution of data that could have generated X (density estimation), among others [Bishop, 2006].

The goal of the learner in *supervised learning* is to find a mapping from x to y, given a training set of pairs (x_i, y_i) [Chapelle *et al.*, 2010]. The y_i are denominated labels of the samples x_i, with $Y = (y_1, ..., y_n) \in \mathcal{Y}$ being a set of labels. As before, it is assumed that the pairs (x_i, y_i) are sampled i.i.d. from some distribution which in this case ranges over $\mathcal{X} \times \mathcal{Y}$. This task is well defined, as a mapping can be evaluated in terms of its predictive performance on test samples. When the labels are continuous, the task is called regression. On the other

hand, when labels take values in a finite set (discrete), the task is called classification.

Reinforcement learning is fundamentally different from *supervised learning*. Here the learner does not have *a priori* knowledge of what to do. Instead, it must take actions to achieve a goal, and a reward is given by the teacher at each state according to the defined goal [Sutton and Barto, 1998]. Therefore, the learner must evolve through trial and error, balancing between exploration of possible new actions and exploitation of its current knowledge.

Semi-supervised learning (SSL) merges concepts from both *unsupervised learning* and *supervised learning*. The data used in training is made up of both labeled examples and unlabeled ones. In this sense, the most common approach is to see SSL as supervised learning with additional information on the distribution of the examples $x_i (i = 1, ..., n)$, as the goal is shared with supervised learning: to predict a label for a given x_i [Chapelle *et al.*, 2010].

2.2 Brief Background of Semi-Supervised Learning

One of the first ideas of using unlabeled data in the training stage dates as back as the late 1960s, with [Scudder, 1965], along with later developments by [Fralick, 1967] and [Agrawala, 1970], to give a few examples. This method/approach developed in these references, used for classification, is called self-learning, also known as self-training or self-labeling. It consists in the repeated use of a supervised method. After training on the labeled data only, in each step a part of the unlabeled examples is labeled according to the current decision function. These newly labeled examples are then used to retrain the supervised method and the process repeats. A big disadvantage of self-learning is that its effect is highly dependent on the supervised method chosen, as it can lead to the unlabeled data having no effect on the solution, being unclear what it is really being done. The SSL paradigm can be further sub-divided into *transductive learning* and *inductive learning*. In the *transductive learning*, or transduction, paradigm [Vapnik and Kotz, 1982], given a labeled training set, and an unlabeled test set, the goal is to perform predictions only for the unlabeled test points. Many methods developed later operate on this setting, while the rest rely on *inductive learning*, where the goal is to find a prediction function defined on the entire space of samples. The interest in SSL increased in the 1990s, in great part due to applications in natural language problems and text classification [Nigam *et al.*, 1998], [Blum and Mitchell, 1998], related to the expansion of the Internet. The majority of the developments were focused on classification problems, leaving regression as a fairly untouched area. From the late 2000s semi-supervised regression gained more popularity among researchers. These efforts were mostly based on co-training and multi-view learning (concepts explained later on), by [Zhou and Li, 2005], [Sindhwani *et al.*, 2005] and [Brefeld *et al.*, 2006], the later being the ones who proposed the coRLSR, the semi-supervised method used throughout this dissertation.

2.3 Co-Training and Multi-View

The co-training paradigm, proposed by [Blum and Mitchell, 1998], contributed to the rising interest in SSL, being an important achievement in the field, as a departure from the highly appraised Expectation Maximization (EM) algorithm [Dempster *et al.*, 1977]. Co-training uses two classifiers, training them separately on two views (i.e. two attribute sets, each of which sufficient for learning and conditionally independent given the class). Then, after classifying the unlabeled examples, each classifier provides the other with the unlabeled examples (and respective labels) they feel most confident about. The process repeats, but now, each classifier's training set is augmented by the examples given by the other. The fundamental idea here is that *the two classifiers must agree on the labeled data as well as the unlabeled data*. Hypertext classification is a common application of this approach, in which setting it is considered that the links and text of each web page constitute two independent views of the same data. As can be noted, the requirements for co-training to operate are quite strict. Nevertheless, [Brefeld and Scheffer, 2004] found that in many domains, even when there is no natural split of the attributes, simply splitting them at random into different views can still be quite effective.

Multi-view learning is similar in concept to co-training, as the focus is the agreement among different (in this case not necessarily two, but multiple) learners. Most multi-view methods do not require the particular assumptions of co-training. Instead, multiple hypothesis are trained from the same labeled set of example, and at the same time aiming to minimize the disagreement on the unlabeled data. Zhou and Li [2005] applied these concepts to kNN regression, using two distinct measures for the two hypotheses (instead of two disjoint attribute sets). Sindhwani *et al.* [2005] proposed an approach similar to the non-parametric version of coRLSR for classification. The underlying framework is co-regularization, the goal of which is optimizing measures of agreement and smoothness over labeled and unlabeled data.

2.4 Soft Sensors

Soft sensors are software-based sensors that can perform a multitude of tasks in process control. They can be developed using many different techniques, and have been successfully applied to a number of different industrial settings, such as refineries [Yuan *et al.*, 2017b], semi-conductor manufacturing [Lee and Kim, 2015], chemical processes [Li *et al.*, 2016], batch fermentation processes [Jin *et al.*, 2014], to name a few examples.

Industries nowadays more than ever are faced with the challenging task of producing the highest quality products at the lowest cost possible in order to stay competitive. These two competing factors are relevant in determining the market success of an industry [Fortuna *et al.*, 2006]. Power and raw materials consumption are directly tied to the final product cost, while process variables can be used to determine product quality. Moreover, some other

factors have to be taken into consideration, such as pollution emissions, safety rules, device faults and malfunction, contributing to a higher complexity of the industrial control system.

All the above-mentioned factors constitute a big challenge to handle, as good solutions require a profound knowledge of the process and of its key parameters. Therefore, a large number of process variables must be monitored by installing measuring systems across all of the production facility. The environment on which these measuring devices operate is generally quite hostile, leading to unexpected faults to occur. Furthermore, a significant delay can be introduced by some measuring tools, possibly leading to an efficiency drop on the control system. It is obvious that the installment and maintenance of a large plant's measuring network is almost always costly and affects the productions' total running cost. For this reason, in some settings, the total number of monitored process variables and/or their measuring sample frequency is reduced. This, however, is not the norm. In many cases, the problem lies on the infrequent sampling of some process variables due to the lack of online sensors. When those process variables that directly affect product quality are only available by laboratory analysis (for example), high delays are introduced, reducing the overall controllability of the process [Facco *et al.*, 2009].

Well-designed soft sensors can solve the issues above, as they:

- have lower cost compared to hardware devices, and can easily be implemented on existing hardware (e.g. microcontrollers);
- work in conjunction with hardware sensors, providing information for fault detection tasks;
- overcome the time delays introduced by slow hardware sensors or offline analysis by estimating data in real-time.

2.4.1 Applications

Soft sensors have a wide range of applications, allowing their deployment in practically all industrial fields. As many industrial plants are equipped with real-time systems for the purpose of collecting online process information (e.g. obtained from sensors) or digital systems in charge of process monitoring/control, SSs have gained popularity among researchers and process control engineers in the last decades, nowadays being regarded as a valuable tool in the industrial context. The most common applications of SSs are online prediction, process monitoring and fault detection, and sensor fault detection and reconstruction [Kadlec *et al.*, 2009].

2.4.1.1 Online Prediction

In most industrial settings, some key process variables and quality variables are essential to ensure high production quality. In some cases, technical or economical limitations can

render infeasible the online measurement of these variables. Alternatively, they may only be available through laboratory analysis. The scarce availability of these measurements decreases the controllability of the associated process. Well designed soft sensors can provide real-time predictions of the desired variables, and can even be incorporated into the automated control loops of the process, if the predictions are accurate enough. Most chemical industrial processes share this issue, being the most common application for soft sensor online prediction, as their dynamics are intricate and there is often no way of obtaining the required information online. In [Jin *et al.*, 2014], a SS is developed to estimate substrate concentration in a chlortetracycline fermentation process, a kind of broad-spectrum antibiotics widely used in pharmaceutical, agriculture and animal husbandry. A general SS approach for estimating product composition from the temperature profile of a distillation processes is devised in [Rani *et al.*, 2013]. In polymerization processes, SSs are used to e.g. estimate resin quality (through acidity number and viscosity values) [Facco *et al.*, 2009] or melt index (MI) of a propylene production process [Shang *et al.*, 2014]. In [Yuan *et al.*, 2017b,a], SSs are developed for prediction of butane content in a debutanizer column (refinery process).

2.4.1.2 Process Monitoring and Fault Detection

At the base of an industrial control system is the control level, which implements the actual control loop. Above this is the supervision level, in charge of monitoring the process' operation almost without relying on the presence of human operators. Fault detection and diagnosis are part of the supervision tasks accomplished by modern industrial control systems. Supervision functions evolved from a simple check of important variables (raising a alarm upon a threshold value trespass) to the use of advanced techniques of computational intelligence, mathematical modeling, signal processing, etc. Modern fault detection and diagnosis systems aim to:

- detect faults in the components of the system in advance, while providing all the information available about the fault which is occurring (or has occurred);
- provide insight to maintenance and repair needs, enabling more efficient scheduling;
- provide a basis for the development of fault-tolerant systems.

Redundancy is a key aspect of fault detection and diagnosis strategies. By having two or more ways of determining the same quantity (e.g. variables, parameters), detection and diagnosis actions can be more effective. The underlying concept being redundancy in fault detection strategies is the comparison of information from the monitored system and a redundant source. A fault can then be detected when these two sources provide different sets of information. The redundant source can be of three main kinds: (*i*) physical, by physically replicating the component to be monitored; (*ii*) analytical, through the use of a mathematical model for the component; (*iii*) knowledge, consisting of heuristic information

about the process. In industrial applications, a fault detection and diagnosis algorithm should use more than one source of redundancy in order to effective. In [Huang, 2008] Bayesian methods are proposed for control loop monitoring and diagnosis. These methods are flexible in the presence of missing variables or missing data, and can be applied to problems with industrial application background, including soft sensors. A soft sensor capable of not only online prediction but also adaptive process monitoring is proposed in [Grbić *et al.*, 2013]. Based on the Gaussian mixture model and Gaussian process regression, the soft sensor uses a procedure for variable selection to reduce the dimensionality of the input space. In [Zhang *et al.*, 2016], a decentralized version of kernel partial least-squares is applied to nonlinear process monitoring, being applied to a continuous annealing process, strip steel producing.

2.4.1.3 Sensor Fault Detection and Reconstruction

Sensor validation is a specific case of fault detection in which the actual sensor (or set of sensors) is the system being monitored. While its basic purpose is to evaluate the reliability of a specific hardware sensor's measurement, it can also provide estimates in case of its malfunction. This way, soft sensors can be used as a source of analytical redundancy. Operating in parallel with hardware sensors, any difference between the outputs of the two can be used to detect any possible fault. Moreover, soft sensors can also act as a back-up measuring device once a fault has been detected without the need of ceasing a process' operation.

2.4.1.4 Other Applications

Soft sensors can have yet more possible applications than the ones mentioned so far. For example, the use of a soft sensor instead of a hardware sensor can be motivated by economical reasons. In some cases, a software tool is a more viable solution than an actual sensor. Still, without any form of redundancy, much care should be taken in assessing predictive performance, and periodic model validation should be performed by using measuring devices and retuning the soft sensor. The need for soft sensor maintenance is a well known issue, and can have different causes, such as a change to a new process working point, or a change in system parameters, in the case of slowly time-varying systems. This will be explored in greater detail later.

Another possible application is related to what-if analysis. As soft sensors are designed with either theoretical basis or historical process data, they may be able to predict the system output for any input trend. This can be used to simulate the system dynamics corresponding to input trends of interest, providing both a deeper understanding of the process and/or insight into adopting more suitable control policies. As expected, when using soft sensors purely based on previous data, the designer must ensure that the SS is trained on data representative of the whole system dynamics. Furthermore, the choice of the synthetic input must also be done according to the system dynamics itself.

2.4.2 Soft Sensor Design

A soft sensor developer faces many challenges during the different stages of SS design. There are critical choices to be made, influenced by the nature of most industrial settings, in order to obtain a reliable and useful source of additional information that, in many scenarios, only soft sensors can provide. Soft Sensor design is typically composed of the following main steps: *data collection and filtering*, *feature selection*, *model choice and training*, *model validation*, and *model maintenance* [Fortuna *et al.*, 2006; Kadlec *et al.*, 2009]. As the arrows in Figure 2.1 suggest, the process is not always linear. In fact, if issues arise at any stage of SS development, the designer must reconsider on the choices made so far in order to solve such issues.

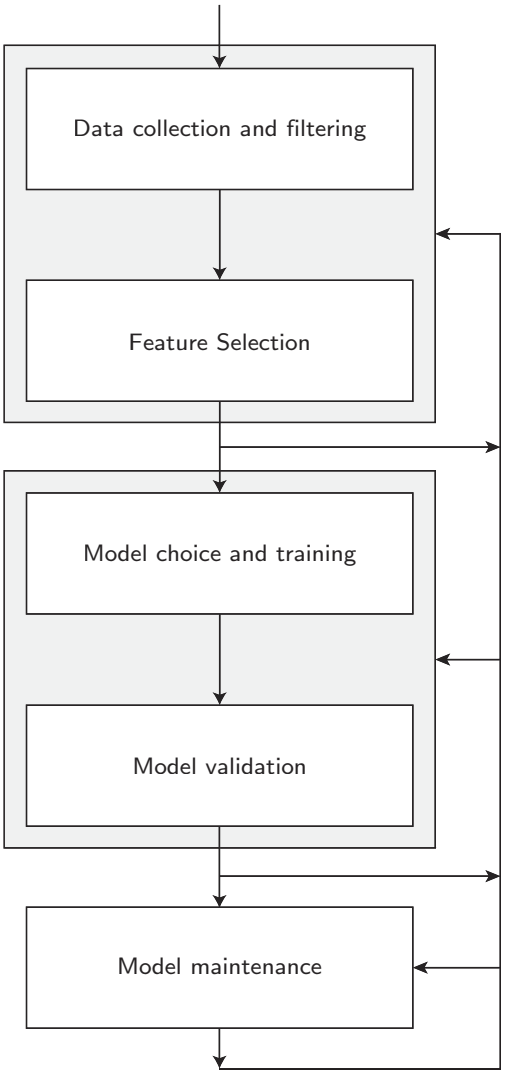


Figure 2.1: Typical stages of soft sensor design.

2.4.2.1 Data Collection and Filtering

Processing plants are generally heavily instrumented for the purpose of process control, resulting in a large set of operation data with many different sources (from all available online sensors). Real-world industrial data is not easy to work with, as some well-known problems arise with its collection and processing. In this first stage of development, plant experts and/or operators can provide a deeper insight into the process' nature, so as to mitigate the effect of issues such as the presence of outliers, sampling time disparity and missing data. These issues can be caused by malfunctioning measuring devices, for example, or by measuring limitations of the process itself. Moreover, the data used in training must be carefully chosen so that it can adequately represent the whole system dynamics.

2.4.2.1.1 Outliers Outliers are inconsistent observations in relation to the majority of stored data. Industrial data are usually corrupted by the presence of outliers, caused by, for example, sensor malfunction, communication errors or hardware degradation. After performing any procedure for outlier detection, the designer must evaluate the candidate outlier list in order to track possible false positive/negative detections) with the help of a system expert. This evaluation must be careful, as important information could be lost if observations labeled as outliers were to be removed from the data wrongly.

2.4.2.1.2 Missing Data In some cases, particular observations might have missing data for one or more features. As most models cannot handle such scenarios, this issue can be solved by one of two different approaches. One approach is to remove the observations that contain missing data, also known as listwise deletion. This approach is quick and simple, but should not be overused. When the number of samples with missing data is large relatively to the total number of samples, important information for the other features could be lost. In such cases, the recommended approach is to fill-in the missing values using some imputing method. In these kind of methods, the missing values are replaced by the mean value or a randomly selected value for the remaining observations of that feature. More complex methods involving feature modeling can also be applied.

2.4.2.1.3 Sampling Rates As mentioned before, in industrial applications, some variables have different sampling rate than others. This is particularly clear when considering the acquisition rates of easy-to-measure and hard-to-measure variables, as that of the easy-to-measure variables is typically much higher [Souza *et al.*, 2016]. Such phenomenon is illustrated in Figure 2.2. In order to use such data, it is required that the variables must be synchronized. This synchronization can be performed by:

- down-sampling the easy-to-measure data samples according to the slower sampling rate of the hard-to-measure variables, essentially excluding those that do not have a corresponding target value [Lu *et al.*, 2004; Kadlec and Gabrys, 2011]. This, however,

can lead to the development of inaccurate models, as some information is lost during the process, more if the sampling rate of the hard-to-measure variable is too low;

- estimating the hard-to-measure variables by using a finite impulse response (FIR) model. This approach tackles the information loss stated before, but has the drawback of requiring fine tuning of some parameters (namely the filter's length and weighting values) to be effective;
- using a method designed to operate on such a setting, based on semi-supervised learning, requiring no action on the data at this stage. These methods use both labeled and unlabeled data in the training stage, eliminating both the information loss problem of down-sampling and reliability issue related to the use of FIR models. Nonetheless, the choice of the specific model is a key factor on the overall effectiveness of the method.

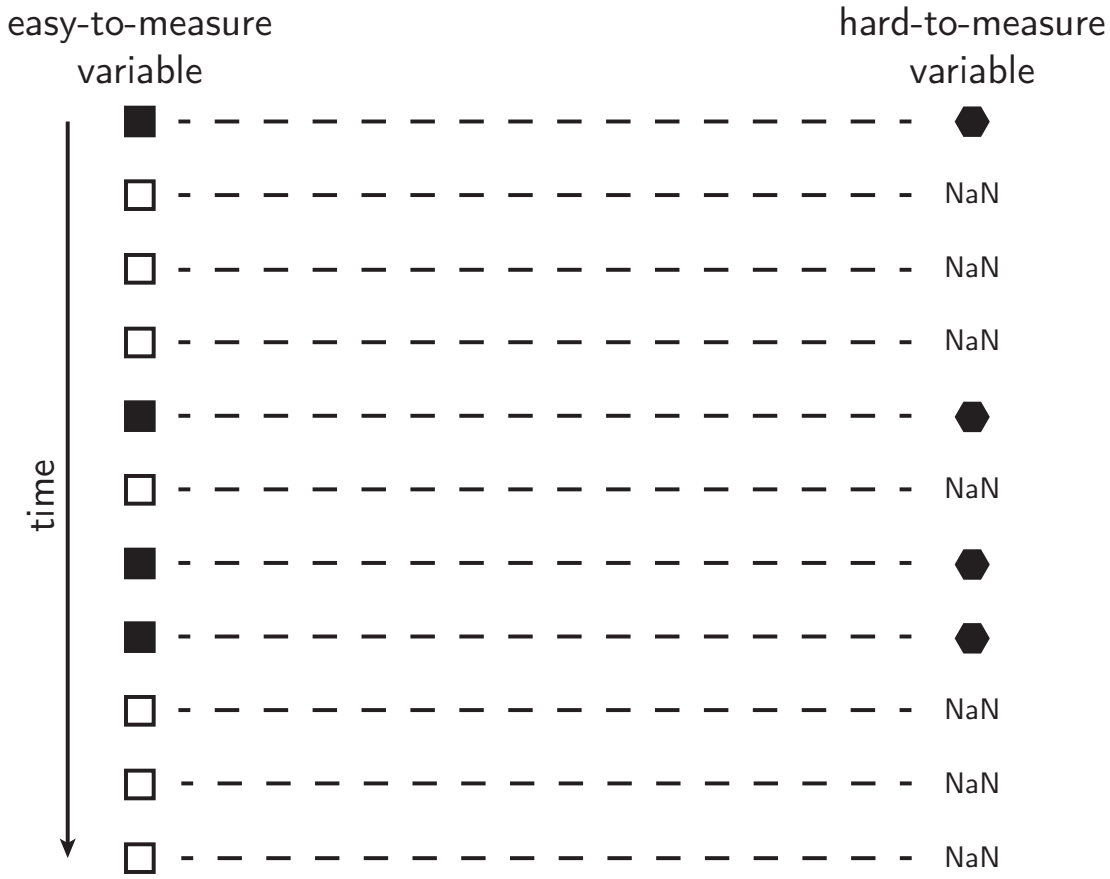


Figure 2.2: Different sampling rates between easy-to-measure and hard-to-measure variables.

2.4.2.2 Feature Selection

In most industrial settings, there are frequently many online sensors used for process control and monitoring. This equates to a large number of candidates for input vari-

ables/features, increasing the overall complexity of the system. System experts can, in most soft sensor applications, select a set of the most relevant features in order to efficiently train the model. Still, there are scenarios in which that selection is not straightforward, namely physically large and highly integrated processes, or can even be infeasible. In these settings, the soft sensor designer might have little knowledge about the process, resorting to black-box model approaches to map the input-output relationship. Most of the research work done suggests that usually only a small number of features is needed to efficiently compose the soft sensor model. Moreover, using only a few relevant features as several advantages, such as the reduction of the model development time, improvement of the model's predictive performance and maintainability, subsequently reducing the characteristic high dimensionality and complexity of real-world data. This is imperative, as in most industrial settings there are time constraints that can only be met if the number of features considered is kept low [Schmitt *et al.*, 2008].

On a basic level, four strategies can be distinguished [Nelles, 2013]:

- Use all inputs - this leads to high-dimensionality issues, such as longer development and training time. It can be practical if the number of potential inputs is small. However, it becomes cumbersome with the increase in the number of features;

- Try all input combinations - this approach would result in the best combination of features. Nevertheless, the high computational demand is obvious, since the number of combinations would increase rapidly with the number of considered features;

- Unsupervised feature selection - the typical tool for unsupervised input selection is based on principal component analysis (PCA) [Jolliffe, 2002]. With this approach non-relevant features can be discarded with low computational cost. The relevance criterion, however, can cause a highly relevant feature for the model performance to be discarded;

- Supervised feature selection - the goal here is to obtain the highest possible predictive accuracy, evaluated on each possible subset of features. Therefore, two components must be defined [Bishop, 1995]: (*i*) search procedure - a subset of features must be obtained using any type of procedure, such as random selection, since exhaustive search is infeasible for most scenarios; (*ii*) selection criterion - the quality of a subset of features must be evaluated in order to compare with the other subsets, usually by computing the estimation accuracy (sum of squares error in the case of regression tasks). Supervised feature selection methods represent the most powerful but also computationally expensive approaches.

Many strategies for feature selection have been proposed in conjunction with soft sensor design to various industrial settings, such as fabric textile [Schmitt *et al.*, 2008], refining process [Wang *et al.*, 2010], semiconductor manufacturing [Lee and Kim, 2015], chemical

polymerization [Lin *et al.*, 2015], and chemical process control [Li *et al.*, 2016]. In [Wang *et al.*, 2010], for example, a feature selection approach is devised to function along PLS regression. Here, a large number of feature subsets is randomly created, iteratively testing each feature subset through PLS and cross-validation in order to reach the one which guarantees the highest prediction power (measured by the root mean square error of cross validation, RMSECV). Similarly, [Li *et al.*, 2016] uses the regression vector of PLS to sort input features by their significance (given by their weights in the mentioned vector). A test of correlation is then performed. Starting with the less relevant features, if a certain feature has significant correlation with others, that feature is canceled, repeating the process until all features have been taken into consideration. A recursive feature elimination technique (RFE) is used with a SVM in [Lee and Kim, 2015]. A SVM is firstly used to rank all features. Then, the individual features' contribution is evaluated based on the support vectors of the trained SVM and weights are assigned to each one. The feature with the smallest weight is removed, and the process repeats iteratively. Fuzzy systems can also be involved into feature selection leading to more efficient and easy learning [Schmitt *et al.*, 2008; Lin *et al.*, 2015]. These approaches while effective can be computationally costly, as methods of cross-validation or techniques like SVM (that require parameter tuning) are somewhat complex. In high dimensionality scenarios, some of the most effective methods can even be prohibitive [Bravi *et al.*, 2017].

2.4.2.3 Model Choice and Training

There are, at a general level, two categories of soft-sensors: model-driven (also known as white-box) and data-driven (also known as black-box) [Kadlec *et al.*, 2009]. Model-driven soft sensors rely on mathematical models to describe the process in study. Thus, they require the knowledge of the physical and/or chemical properties in order to provide valuable information. The development of such models is not a straightforward task, and as they are usually designed with ideal conditions in mind, the sheer nature of the industrial noisy data can render them unusable. As a way to circumvent this disadvantage, data-driven soft sensors are constructed on the data measured within the processing plant, therefore staying more true to the process' real operating conditions. The large volume of process data available in industrial plants contributed to the rising popularity of data-driven soft sensor approaches, the most popular methods being based on partial least squares (PLS) [Zhang *et al.*, 2016], artificial neural network (ANN) [Rani *et al.*, 2013], and support vector machine (SVM) [Shang *et al.*, 2014]. The choice of the model to use is not a obvious one, since every method has its advantages and disadvantages, and possible scenarios in which to apply. Nevertheless, the existence of unlabeled data on some industrial settings might favor semi-supervised approaches, as they are able to incorporate such information to better map the relationship between inputs and outputs. Recently, semi-supervised versions of the above-mentioned methods were developed, showing the potential of the underlying concept in soft sensor modeling [Ge and Song, 2011; Ge *et al.*, 2014; Zhou *et al.*, 2014; Zhu *et al.*,

2015].

2.4.2.4 Model Validation

The model validation step aims to infer on the generalization capabilities of the trained model. This way, its ability of estimating accurately on previously unseen samples is evaluated. When training on large datasets, they are usually split into two sets: the training set, and the test set. The model is learned using the training subset, and its predictive performance is evaluated on the test set. Often, it is also used a validation set, which is used to evaluate the performance of the learning structure inside the learning procedure itself (e.g. as part of an intermediary assessment or a final termination condition). In contrast to the validation set, the test set is used to evaluate the performance of the learned structure, after and outside the learning algorithm (i.e. to evaluate the learning outcome). In the case of small datasets, cross-validation techniques like K-fold cross-validation and leave-one-out cross-validation are employed to evaluate the model's performance. In K-fold cross-validation, the training subset is split randomly into K folds, training the model using the samples from $(K-1)$ folds and evaluating its performance on the remaining fold. The process repeats for all K folds, and the performance metrics' values are averaged. The leave-one-out cross-validation technique is used more frequently on very small datasets, operating on the same principle as the K-fold one, with K being equal to the number of samples. The most used performance metrics are mean squared error (MSE) and its variants, root mean squared error (RMSE) and normalized root mean squared error (NRMSE). NRMSE can be expressed in percentage form, providing a more intuitive performance analysis. These performance values can be paired with the scores from another performance metric like the *R-squared*, also known as *coefficient of determination*, R^2, enabling a deeper insight on predictive performance. The R^2 metric represents the amount of variability of the data that is explained by the model. Further explanation on the metrics used will be provided in Section 5.2.

2.4.2.5 Model Maintenance

After the soft sensor is developed and deployed, it needs to be regularly maintained and tuned. As drifts in process data are likely to occur over time (due to process variations, for instance), the soft sensor must be retuned in order to keep its performance from degrading. Most soft sensor models do not include an automated process of maintenance. This requires the maintenance to be performed manually, increasing the overall cost for the SS application. Adaptive soft sensors can mitigate this issue. As they are designed to adapt to process variations, they may lead to far less need of maintenance and retuning. In [Ma *et al.*, 2009], an adaptive sensor is applied to a industrial o-xylene distillation column. The statistical identification of key variables allows for the soft sensor to choose the best variables needed at a given period. An automated mechanism for an adaptive soft sensor is proposed in [Grbić

et al., 2013], designed to operate in highly nonlinear time-varying processes. The devised approach acts on different levels of Gaussian models, retuning parameters as new samples are acquired. In [Xiong *et al.*, 2017], a moving-window approach is used in an adaptive SS, tested in both a sulfur recovery unit and an industrial debutanizer column. In this type of approach, the data used for model training changes as new samples are available, discarding the older ones.

Chapter 3

Co-Regularized Least Squares Regression

This chapter describes the theoretical background of the semi-supervised method used throughout this dissertation. In Section 3.1, the theoretical basis of the method is stated, and both its non-parametric exact version and semi-parametric approximation are derived, in Sections 3.2 and 3.3, respectively. CoRLSR's relation to RLSR is explained in Section 3.4. Finally, the practical implementation is presented in Section 3.5.

The methodologies developed in the work of this dissertation were presented in the paper "Semi-Supervised Soft Sensor and Feature Ranking Based on Co-Regularised Least Squares Regression Applied to a Polymerization Batch Process" [Ferreira *et al.*, 2017].

3.1 Semi-Supervised Regularized Regression

The coRLSR algorithm [Brefeld *et al.*, 2006] is based on casting co-training [Blum and Mitchell, 1998] as a regularized risk minimization problem in Hilbert spaces. As in other kernel approaches, the optimal solution can be described by a linear combination of kernel functions "centered" on the set of labeled and unlabeled examples. Given a set of training data $\mathbf{\Phi}$, the goal is to find the following solution

$$f(\cdot) = \underset{f^*(\cdot)\in\mathcal{H}}{\arg\min} \left\{ \sum_{i=1}^{N} \mathcal{L}(y_i, f^*(\mathbf{x}_i)) + \nu\Omega[f^*(\cdot)] \right\} \tag{3.1}$$

where $\mathcal{H}$ is a Hilbert space of functions, $\mathcal{L}(\cdot,\cdot)$ is a convex loss function, $\nu > 0$ is a parameter and $\Omega[f^*(\cdot)]$ is a regularization term.

Assuming that the examples can be represented in M different ways (e.g. defined by different features and/or different kernel functions), M predictors can be learned $f_1, \ldots, f_M$ that simultaneously depend on each other, satisfying an optimization criterion involving all views. The final prediction is then the average of all f_v predictors ($v = 1, \ldots, M$).

Given a set with M subsets containing labeled samples $\Gamma = \{\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_M\}$, where $\mathbf{\Phi}_v = \{\mathbf{x}_i, y_i\}_{i=1}^{N_v}$, $N_v \leq N$, and $v = 1, \ldots, M$, and a set of unlabeled samples $\mathbf{\Phi}_z^* = \{\mathbf{x}_z\}_{z=1}^{Z}$, the objective is to find functions $f_1 : \Gamma \to \mathbb{R}, \ldots, f_M : \Gamma \to \mathbb{R}$ from different Hilbert Spaces

$\mathcal{H}_1, \ldots, \mathcal{H}_M$ that minimize

$$Q(f_1, \ldots, f_M) = \sum_{v=1}^{M} \left[\sum_{i=1, \mathbf{x}_i \in \mathbf{\Phi}_v}^{i=N_v} \mathcal{L}(y_i, f_v(\mathbf{x}_i)) + \nu_v \|f_v(\cdot)\|^2 \right] + \lambda \sum_{u,v=1}^{u,v=M} \sum_{z=1, \mathbf{x}_z \in \mathbf{\Phi}_z^*}^{z=Z} \mathcal{L}(f_u(\mathbf{x}_z), f_v(\mathbf{x}_z)) \tag{3.2}$$

where, $\mathcal{L}(\cdot, \cdot)$ is the loss function, $\|f_v(\cdot)\|^2$ is the regularization term, ν_v is the regularization parameter and λ is a parameter that weighs the influence of pairwise disagreements. The optimization problem (3.2) focuses on minimizing both the *labeled error* (first part of the sum) and the *unlabeled error* (second part of the sum). Every view predictor is intended to have small training error on the labeled examples, i.e. small labeled error. Furthermore, the disagreement between views over unlabeled examples should also be minimized, i.e. small unlabeled error.

A function k is a kernel function if it is symmetric and positive semi-definite, generating a kernel matrix $\mathbf{K}$ with the same characteristics, through an inner product operation on $\mathbf{x}$. Choosing a k_v for each f_v ($v = 1, \ldots, M$) allows the use of the *representer theorem* [Wahba, 1990; Schölkopf *et al.*, 2001] to obtain the predictions for each v view (i.e. the solutions of Eq. (3.2)) as

$$f_v^*(\mathbf{x}_i) = \sum_{\mathbf{x} \in \bigcup_{\mathbf{\Phi}_v \in \Gamma} \mathbf{\Phi}_v \bigcup \mathbf{\Phi}_z^*} \alpha_v(\mathbf{x}) k_v(\mathbf{x}, \mathbf{x}_i), \tag{3.3}$$

where $k_v(\cdot, \cdot)$ is the reproducing kernel of the Hilbert Space $\mathcal{H}_v$, and $\alpha_v(\mathbf{x})$ is a coefficient associated with $\mathbf{x}$.

In this way, the vector $\mathbf{f}_v = \Big[\ldots, f_v(\mathbf{x}_i), \ldots \Big]^T_{\mathbf{x}_i \in \mathbf{\Phi}_v \bigcup \mathbf{\Phi}_z^*}$, for all $\mathbf{\Phi}_v \in \Gamma$, can be expressed as $\mathbf{K}_v \mathbf{c}_v$, and $\|f_v(\cdot)\|^2$ can be expressed as $\mathbf{c}_v^T \mathbf{K}_v \mathbf{c}_v$, where $[\mathbf{K}_v]_{ij} = k_v(\mathbf{x}_i, \mathbf{x}_j)$ and $[\mathbf{c}_v]_{i,1} = \alpha_v(\mathbf{x}_i)$. $\mathbf{K}_v$ is a strictly positive definite kernel matrix, i.e. it is symmetric and has no negative, and no zero eigenvalues. The notation used for each view's labels is $[\mathbf{y}_v]_i = y_i$ for $i = 1, \ldots, N_v$. For standard kernel methods like Eq. (3.1), defining the loss function as the squared difference between real and predicted output, $\mathcal{L}(y_i, f_v(\mathbf{x}_i)) = (y_i - f_v(\mathbf{x}_i))^2$ gives the ridge regression [Saunders *et al.*, 1998] or regularized least squares regression (RLSR) solutions.

3.2 Non-Parametric Least Squares Regression

With N_v training examples in view v and Z unlabeled examples, Eq. (3.2) can be rephrased as the following coRLSR optimization problem, in which the goal, for fixed $\lambda, \nu_v \geq 0$, is to minimize the following cost function

$$Q(\mathbf{C}) = \sum_{v=1}^{M} \left(\|\mathbf{y}_v - \mathbf{L}_v \mathbf{c}_v\|^2 + \nu_v \mathbf{c}_v^T \mathbf{K}_v \mathbf{c}_v \right) + \lambda \sum_{u,v=1}^{M} \|\mathbf{U}_u \mathbf{c}_u - \mathbf{U}_v \mathbf{c}_v\|^2 \tag{3.4}$$

over $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_M] \in \mathbb{R}^{N_1+Z} \times \ldots \times \mathbb{R}^{N_M+Z}$. $\mathbf{L}_v \in \mathbb{R}^{N_v \times (N_v+Z)}$ and $\mathbf{U}_v \in \mathbb{R}^{Z \times (N_v+Z)}$ are computed from a strictly positive definite kernel function and form a positive definite kernel matrix $\mathbf{K}_v \in \mathbb{R}^{(N_v+Z)\times(N_v+Z)}$ as

$$\mathbf{K}_v = \begin{pmatrix} \mathbf{L}_v \\ \mathbf{U}_v \end{pmatrix}. \tag{3.5}$$

Getting the derivative of (3.4) with respect to $\mathbf{c}_v$, yields

$$\nabla_{\mathbf{c}_v} Q(\mathbf{C}) = 2\mathbf{G}_v\mathbf{c}_v - 2\mathbf{L}_v^T y_v - 4\lambda \sum_{u:u\neq v} \mathbf{U}_v^T\mathbf{U}_u\mathbf{c}_u.$$

where

$$\mathbf{G}_v = \mathbf{L}_v^T\mathbf{L}_v + \nu_v\mathbf{K}_v + 2(M-1)\lambda\mathbf{U}_v^T\mathbf{U}_v.$$

At the optimum the following relation holds

$$[\nabla_{\mathbf{c}_1} Q(\mathbf{C}), \ldots, \nabla_{\mathbf{c}_M} Q(\mathbf{C})]^T = \mathbf{0},$$

and then the exact solution for the optimum of (3.4) can be found as

$$\begin{bmatrix} \mathbf{G}_1 & -2\lambda\mathbf{U}_1^T\mathbf{U}_2 & \cdots \\ -2\lambda\mathbf{U}_2^T\mathbf{U}_1 & \mathbf{G}_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1^T y_1 \\ \mathbf{L}_2^T y_2 \\ \vdots \end{bmatrix}. \tag{3.6}$$

The solutions can be found in time $O((M \cdot N + M \cdot Z)^3)$ (further details can be found in the original paper [Brefeld *et al.*, 2006]). This cubic time complexity in the number of labeled examples is shared with the most common supervised learning algorithms such as SVMs, RLSR, etc. However, cubic time complexity in the number of unlabeled examples renders coRLSR unusable, as in some industrial settings (as discussed in Section 2.4.2.1) the number of unlabeled examples far exceeds the number of labeled ones (i.e., $Z \gg N$). Then, a semi-parametric approximation was also devised in order to reduce the complexity in the number of unlabeled examples. This approximation was achieved by optimizing over functions that can be expanded in terms of training (labeled) instances only, and it is presented next.

3.3 Semi-Parametric Approximation

With N_v training examples in view v and Z unlabeled examples, Eq. (3.2) can be rephrased [Brefeld *et al.*, 2006] as the following coRLSR optimization problem, in which the goal, for fixed $\lambda, \nu_v \geq 0$, is to minimize the following cost function:

$$Q(\mathbf{C}) = \sum_{v=1}^{M} \left(\|\mathbf{y}_v - \mathbf{L}_v\mathbf{c}_v\|^2 + \nu_v\mathbf{c}_v^T\mathbf{L}_v\mathbf{c}_v \right) + \lambda \sum_{u,v=1}^{M} \|\mathbf{U}_u\mathbf{c}_u - \mathbf{U}_v\mathbf{c}_v\|^2 \tag{3.7}$$

over $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_M] \in \mathbb{R}^{N_1} \times \ldots \times \mathbb{R}^{N_M}$. $\mathbf{L}_v \in \mathbb{R}^{N_v \times N_v}$ and $\mathbf{U}_v \in \mathbb{R}^{Z \times N_v}$ are computed from a strictly positive definite kernel function and form a positive definite kernel matrix

$\mathbf{K}_v \in \mathbb{R}^{(N_v+Z)\times(N_v+Z)}$ as

$$\mathbf{K}_v = \begin{pmatrix} \mathbf{L}_v & \mathbf{U}_v^T \\ \mathbf{U}_v & * \end{pmatrix} \tag{3.8}$$

where the part marked by * is not needed.

Getting the derivative of (3.7) with respect to $\mathbf{c}_v$, yields

$$\nabla_{\mathbf{c}_v} Q(\mathbf{C}) = 2\mathbf{G}_v\mathbf{c}_v - 2\mathbf{L}_v y_v - 4\lambda \sum_{u:u\neq v} \mathbf{U}_v^T\mathbf{U}_u\mathbf{c}_u$$

where

$$\mathbf{G}_v = \mathbf{L}_v^2 + \nu_v\mathbf{L}_v + 2(M-1)\lambda\mathbf{U}_v^T\mathbf{U}_v.$$

At the optimum the following relation holds

$$[\nabla_{\mathbf{c}_1} Q(\mathbf{C}), \ldots, \nabla_{\mathbf{c}_M} Q(\mathbf{C})]^T = \mathbf{0},$$

and then the exact solution for the optimum of (3.7) can be found as

$$\begin{bmatrix} \mathbf{G}_1 & -2\lambda\mathbf{U}_1^T\mathbf{U}_2 & \cdots \\ -2\lambda\mathbf{U}_2^T\mathbf{U}_1 & \mathbf{G}_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 y_1 \\ \mathbf{L}_2 y_2 \\ \vdots \end{bmatrix}. \tag{3.9}$$

The solutions can be found in time $O((M \cdot N)^3 + M^2 \cdot Z)$ (further details can be found in the original paper [Brefeld *et al.*, 2006]). The reduction in complexity achieved by this approximation means that the proposed method is able to scale linearly with the number of unlabeled examples, being a lot more usable in real-world case scenarios.

3.4 Relation to Regularized Least Squares Regression

The solutions of Regularized Least Squares Regression can be found by using any of the above-mentioned methods, as both the non-parametric and semi-parametric versions of coRLSR are natural generalizations of regular RLSR. Considering the two optimization problems stated in Sections 3.2 and 3.3 one can obtain M independent RLSR solutions for $Z = 0$. Moreover, in the semi-parametric version, M independent RLSR solutions can also be obtained for $\lambda = 0$. Using only one view, $M = 1$, the second part of the Eq. (3.7) disappears and a single regularized least squares solution is obtained. This will be useful in testing in order to validate the predictive performance gains by using the semi-supervised methods stated in this chapter.

3.5 Practical Implementation

For the purpose of testing and validating the models, a set of *MATLAB*® functions was developed to implement both coRLSR's variants and regular RLSR as well. In designing

these functions, expandability and flexibility were regarded as key factors, as well as optimization, in order to reduce its runtime. The first function developed computes the optimal solutions of Eqs. (3.4) and (3.7), its required parameters being the following:

$$\text{function}[model] = \text{coRLSR}(\mathbf{X}, \mathbf{y}, M, \lambda, option, inst, att)$$

$\mathbf{X}$ is a matrix of training examples $\mathbf{X} \in \mathbb{R}^{(N+Z)\times d}$ and $\mathbf{y}$ is a vector of training labels $\mathbf{y} \in \mathbb{R}^{(N+Z)}$, with N labeled examples and Z unlabeled examples (the unlabeled examples in vector $\mathbf{y}$ typically consist of NaN values, as mentioned before and illustrated in Figure 2.2). M and λ are the parameters present in the definitions of both variants of coRLSR. Finally, *option* refers to the specific method used, as this function is able to obtain the solutions for non-parametric coRLSR, semi-parametric coRLSR and regular RLSR, and *inst* and *att* are optional parameters. The specific instance/example split between views can be configured by $inst \in \mathbb{R}^{M\times(N+Z)}$, a logical matrix that indexes/selects the examples that constitute each of the M views. Similarly, the attribute/feature split can be configured by $att \in \mathbb{R}^{M\times d}$, a logical matrix that indexes/selects the features that constitute each of the M views. A logical matrix contains zeros and ones, where the ones correspond to the elements that are extracted from some other matrix in order to form a sub-matrix. If undefined, all examples will be included in all views, and the features will be sequentially split amongst all views. The function outputs a struct *model*, containing the solution vector $\mathbf{C}$ (as it appears in Sections 3.2 and 3.3) as well as other variables and parameters (3.1).

Table 3.1: Fields description of the output struct.

struct *model*	
Field	**Description**
N	number of labeled examples
Z	number of unlabeled examples
$\mathbf{X}^*$	rearranged $\mathbf{X}$ so that the labeled examples come first
att	logical matrix containing the attribute/feature split
$inst$	logical matrix containing the instance/example split
σ	vector containing parameters used in the kernel function
$\mathbf{C}$	vector containing the solution coefficients for all views

In this work a Gaussian kernel was used (the definition is presented later in Section 5.3. For this reason, there is the need to save a parameter for each view. The vector $\sigma \in \mathbb{R}^M$ contains these parameters for each of the M views.

The use of a struct as the function output was chosen in order to simplify the process of obtaining the solution coefficients and the model's actual predictions. A second function was developed to provide estimates for any input matrix:

$$\text{function}[\hat{\mathbf{y}}] = \text{estimatecoRLSR}(\mathbf{X}_{\mathbf{est}}, model)$$

$\mathbf{X_{est}}$ is a matrix of examples for which to calculate estimations (Note that here $\mathbf{X_{est}}$ need not be the same $\mathbf{X}$ used in training), and *model* is the struct outputted by the previous function.

Figure 3.1 illustrates the general structure of both functions. The operation of the coRLSR() function can be explained by the following steps:

- $\mathbf{X}$ and $\mathbf{y}$ rearrangement - both input matrix and input vector must be rearranged so that all labeled examples come first and then the unlabeled ones. This is done in order to accommodate for the kernel matrix compositions for both methods, stated in (3.5) and (3.8);

- view creation - M views are created, by example split, by feature split, or a combination of both. Examples can be randomly chosen for each view, can be split sequentially, or all views can share all available examples. Features, in a similar manner can be randomly or sequentially split, creating either disjoint or joint sets. If *inst* and/or *att* are defined, the splits are performed accordingly. By default all instances are assigned to all M views, and the features are split sequentially;

- kernel computation - the kernel is computed using a positive definite function. Any type of positive definite kernel can be used. In this work, a Gaussian kernel is used, so its required parameters are also computed;

- coefficient solution - the required matrix is assembled and inverted in order to solve the linear equations stated in (3.6) and (3.9).

In a similar manner, the estimatecoRLSR() function performs the following steps:

- kernel computation - the kernel is computed using the examples in $\mathbf{X_{est}}$ and the ones in X from the *model* struct;

- estimation computation - the estimations are computed using the kernel and the coefficient solutions $\mathbf{C}$ from the *model* struct, using the formulation stated in (3.3).

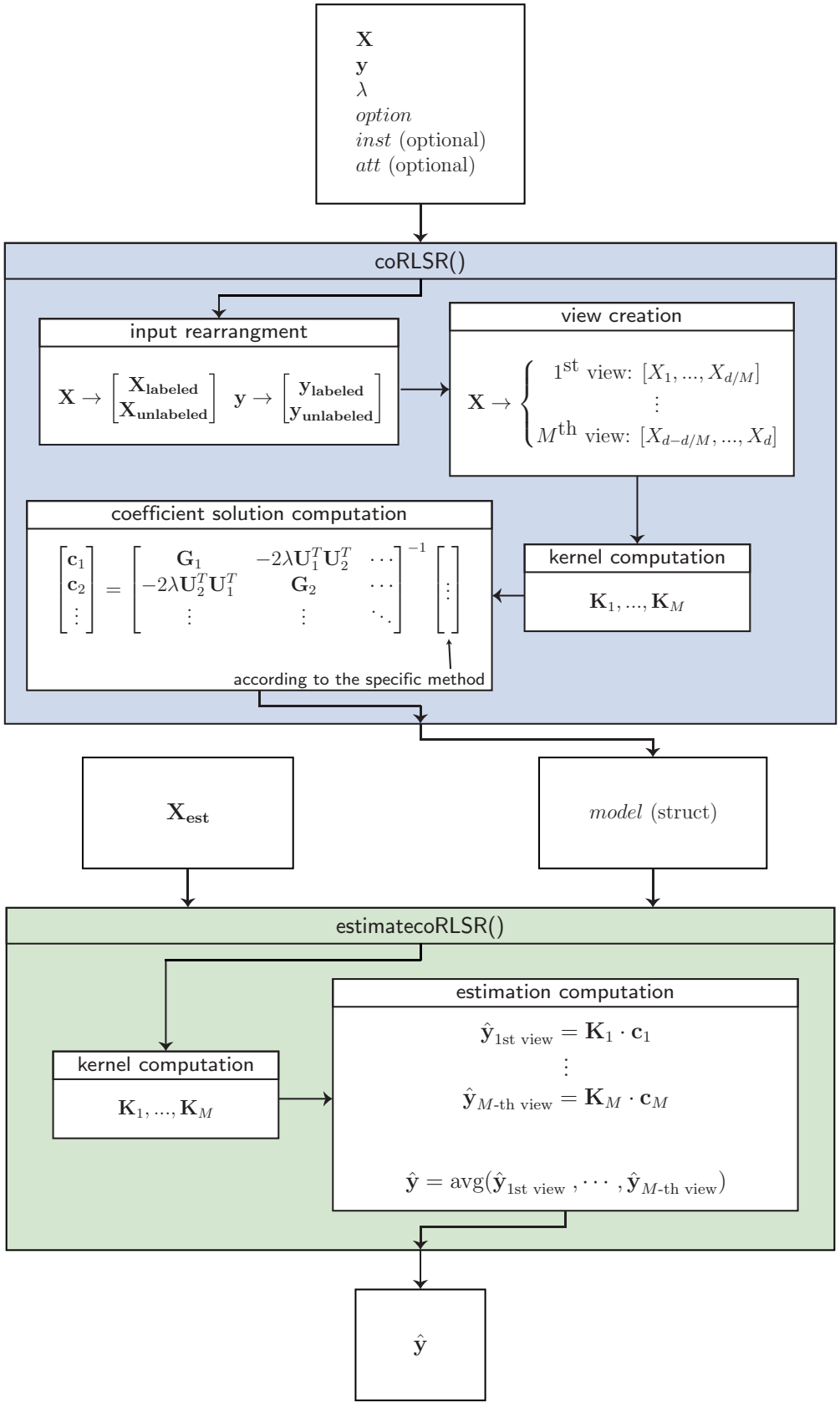


Figure 3.1: General structure of the coRLSR() and estimatecoRLSR() functions.

Chapter 4

Feature Ranking and Selection

This chapter describes the devised method for feature ranking and selection. In Section 4.1, the theoretical background of the feature ranking method is presented. Then, in Section 4.2, the practical implementation of feature ranking and selection is explained, using coRLSR. Finally, in Section 4.3, a comparison to the feature selection methods stated in Subsubsection 2.4.2.2 is presented.

The methodologies developed in the work of this dissertation were presented in the paper "Semi-Supervised Soft Sensor and Feature Ranking Based on Co-Regularised Least Squares Regression Applied to a Polymerization Batch Process" [Ferreira *et al.*, 2017].

4.1 Feature Ranking Method Based on Sensitivity Analysis

As the most common model approaches are data-driven (coRLSR being one), the relevance of an individual feature on the output is not easily perceivable. Moreover, most industrial processing plants are heavily equipped with online sensors, further increasing this task's difficulty. Feature selection approaches existing in the literature have a high computational cost in that development stage. The method proposed in this chapter aims to rank features with low computational cost. It is based on sensitivity analysis, which is the study of how the uncertainty in the output of a mathematical model or system can be explained by different sources of uncertainty in its inputs. In this sense, the importance of an input can be measured by computing the variation of the output when the input is perturbed. This way, the influence of an individual feature can be measured without resorting to retraining the model. The proposed methodology is theoretically justified based on common assumptions, presented below.

Assume that the output of a regression model, defined as $f(\mathbf{x}_i)$, is achieved by the

following conditional mean [Bishop, 2006]:

$$f(X = \mathbf{x}_i) = \int_Y y\, p(y|X = \mathbf{x}_i)\, dy. \tag{4.1}$$

Assume that $X_{-j} = X \setminus X_j$. From Yang *et al.* [2009], the following theorem is stated:

Theorem 1:

$$P(y|X_{(j)}) = P(y|X_{-j}) \tag{4.2}$$

where $X_{(j)}$ is defined as

$$X_{(j)} = \{X_1, \ldots, X_j^*, \ldots, X_d\}$$

with X_j^* being a random permutation of X_j applied across all samples.

By applying Theorem 1 into equation (4.1), the following relation holds

$$\begin{aligned} f_{-j}(X) &= \int yP(y|X_{-j})\, dy \\ &= \int yP(y|X_{(j)})\, dy. \end{aligned} \tag{4.3}$$

Eq. (4.3) means that permuting a variable j in the trained model, is similar to having the output of the same model without the variable j. This brings several advantages, since instead of completely removing a feature and retraining the model, a random permutation is applied to mimic the output of the model without the respective variable, while the other features are kept unchanged.

The rank of a feature j is then defined by the absolute normalized difference between the prediction error of the model trained with the perturbed set $X_{(j)}$ and with the 'intact' set X:

$$J(X_j) = \frac{|E_{(j)} - E|}{E}, \tag{4.4}$$

where $E_{(j)}$ and E are performance metrics computed from the predictions using $X_{(j)}$ and X as input set, respectively.

4.2 Practical Implementation

The general structure of the feature ranking algorithm applied to coRLSR is illustrated in Figure 4.1. First, the model is trained with all features. Then, for each feature, a random permutation is applied, the estimates are computed and its rank is obtained from (4.4), using the estimates computed for the undisturbed set.

In order to select the most relevant features, a recursive feature elimination (RFE) approach is applied to delete the most irrelevant features. First, the model is trained using all features available and the rank of each individual feature is then obtained as stated above. Then, the least important feature is progressively eliminated and the performance of the estimation is computed. To speed up the process, a threshold ϵ is defined such that features j with $J(X_j) < \epsilon$ will be considered with no importance, being immediately discarded.

4.3 Method Comparison

The presented method can be classified as a supervised feature selection method, as stated Subsubsection 2.4.2.2. The search procedure consists in, after ranking all features, obtaining the first subset by applying the defined threshold. Subsequent subsets are created by removing the least relevant variable. The selection criterion used is a performance metric such as RMSE, in this case. When comparing with the other approaches mentioned, the proposed method's complexity appears to be equal or lower. For example, in [Wang *et al.*, 2010] a large number of feature subsets is randomly created, and its evaluation is done iteratively. This requires retraining the model for each of the feature subset. Considering there is a large number of feature subsets, this process can be very demanding, as there can be many possible subsets from the available set of features. In [Li *et al.*, 2016], a correlation test is performed for each feature in relation to the others, which can be a long process. In [Lee and Kim, 2015], a SVM is used to rank all features, which in of itself is a costly procedure. Then, a process of RFE (similarly to the proposed method) is applied. The devised feature ranking approach is inherently simple and cost effective (time-wise). The random permutation of an input feature and estimate computation is a relatively easy task since the model is already trained on data. However, the RFE procedure is dependent on the required time for the specific method. For this reason, the threshold is defined to speed up the process. In scenarios with a large number of input features the gap between the devised method and the others can be expanded since frequently few features are required for accurate prediction. The threshold then plays an important role in the proposed method, and when adequately defined can substantially decrease the possible feature subset size. The general nature of the devised feature selection method enables its use with any regression/prediction method. Furthermore, since the actual regression method is used in the feature ranking procedure a deeper evaluation is performed. Not only the relevance of a feature to the problem is evaluated, but also it is also evaluated its relevance within the context of its joint application with the specific employed regression approach, as it is expected that the ranking of features varies when different model approaches are used.

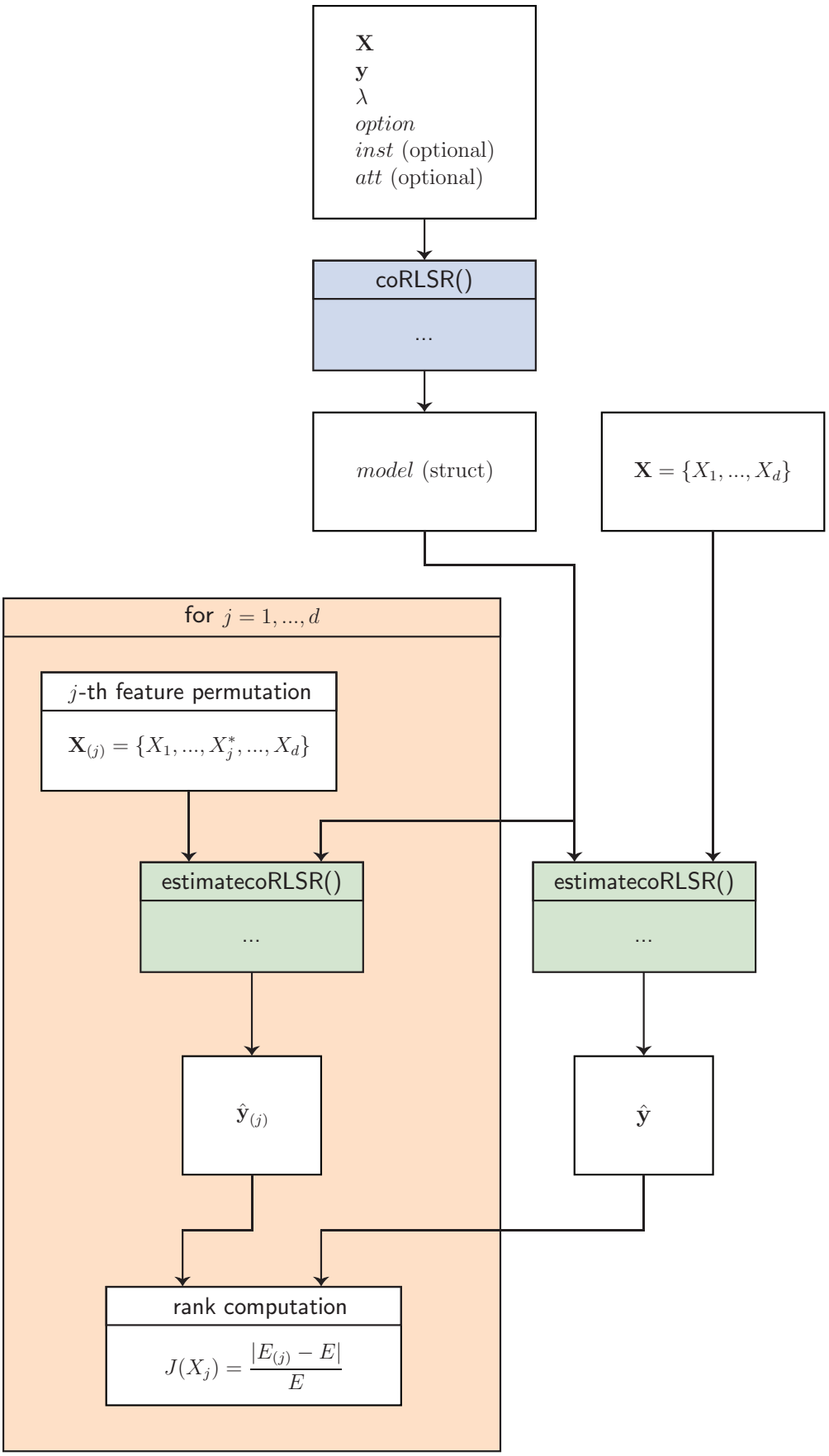


Figure 4.1: General structure of the feature ranking algorithm.

Chapter 5

Test and Results

This chapter presents all the tests performed in order to verify and validate the proposed methods. In Section 5.1, the real-world data set is introduced. Next, the performance metrics are presented in Section 5.2, and the experimental setup is presented in Section 5.3. The improvement to the RLSR method is shown in Section 5.4. In Section 5.5, the full tests and comparisons are presented. Firstly, the parameter tuning process and results of coRLSR are shown in Subsections 5.5.1 and 5.5.2, respectively. Then, these results are compared to the other methods in Subsection 5.5.3. The proposed feature ranking and selection method is applied to coRLSR in Subsection 5.5.4 and the to the remaining studied approaches in Subsection 5.5.5. Finally, view creation is explored in Subsection 5.5.6, and the value of soft sensors is stated in Subsection 5.5.7.

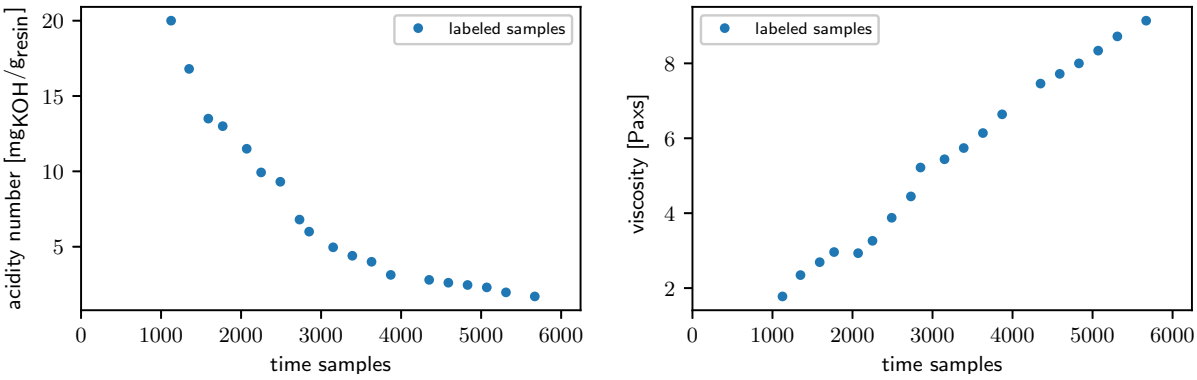
5.1 Real-World Industrial Dataset

The industrial setting explored in this dissertation is the production of a polyester resin, via a batch polymerization process. There are two quality variables measured along the process (hard-to-measure variables): N_A (acidity number) and μ (viscosity). The plant is equipped with several sensors for online measurement, connected to a process computer that records the values every 30 [s]. In total, 34 variables are collected and recorder, either by simply obtaining the measurements from sensors (temperatures, pressures, valve openings, etc.) or in controller setpoints, adjusted manually by the process operators. Each batch then contains anywhere between 4500 and 7500 recordings of each process variable (each batch takes 40-70 [h]). Product quality variables (N_A and μ) are not measured online. Instead, product samples are taken manually, quite infrequently and unevenly (once every 1.5 - 2[h], depending on the evolution of the batch), and are sent for laboratory analysis, which takes approximately 20 [min] to be completed, with the accuracy being $\sim 10\%$ of the reading. Moreover, the quality measurements are only available 8-10 [h] after the batch starts (i.e. after at least 1000 time instants have elapsed). This equates to only 15-20 quality measurements being available for each batch.

The assembled dataset (kindly provided by [Facco *et al.*, 2009]) contains data from 33 batches (16 months of operating effort), split into two subsets: 27 batches constitute the training set, and the remaining 6 represent the test set. From the over 200000 samples collected, only 663 are labeled, and the detailed distribution is illustrated in Table 5.1. This challenging real-world setting allows semi-supervised algorithms like coRLSR to possibly exploit the huge number of unlabeled examples available. From Figure 5.1, the scarcity of labeled samples is quite evident. Only 19 samples of the first training batch records are labeled, the remaining being unlabeled (6242 samples total).

Table 5.1: Distribution of labeled/unlabeled samples in the dataset.

Training subset *(163094 samples)*		**Test subset** *(37886 samples)*	
labeled	unlabeled	**labeled**	unlabeled
530	162564	**133**	37753



(a) Acidity number (b) Viscosity

Figure 5.1: First batch of the training subset.

5.2 Performance Metrics

The metrics used to measure the model predictive performance of all approaches tested were: *Root Mean Square Error* (RMSE), *Normalized Root Mean Square Error* (NRMSE) and *R-squared*, also known as *coefficient of determination* (R^2). All formulations are stated below.

5.2.1 Root Mean Square Error

Root Mean Square Error, also known as standard error of the regression, is one of the most widely used statistical measures to assess average model performance error. For a total

of n samples, the formulation is the following:

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2}{n}}. \tag{5.1}$$

The higher the difference in both sets of data (estimated and real output) the higher the RMSE. So, the fit is more useful for prediction as its RMSE is closer to 0.

5.2.2 Normalized Root Mean Square Error

Normalized Root Mean Square Error, as its name implies, is a normalized variant of RMSE. It is defined as:

$$\text{NRMSE} = \frac{\text{RMSE}}{(y_{\max} - y_{\min})}, \tag{5.2}$$

where $y_{\max}$ and $y_{\min}$ are the maximum and minimum observed values, respectively. Similarly to the RMSE, the closer to 0 the more useful the fit is. It ranges from 0 to 1, allowing an easier comparison between different datasets.

5.2.3 Coefficient of Determination

The coefficient of determination measures how successful the fit is in explaining the variation of the data. The formulation used is [Jin *et al.*, 2001]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}, \tag{5.3}$$

where $\overline{y}$ is the average observed value. Its value ranges from 0 to 1. Higher values of R^2 (closer to 1) translate to a better explanation of the variation in the data by the fit.

5.3 Experimental Setup

5.3.1 Co-Regularized Least Squares Regression

The co-Regularized Least Squares Regression formulation stated in Chapter 3 has some parameters/variables/functions that must be chosen in order for it to operate. These are:

- M - number of views used;
- $\mathcal{L}(\cdot,\cdot)$ - loss function;
- ν_v - regularization parameter for view v ($v = 1, \ldots, M$);
- λ - parameter that weighs the influence of pairwise disagreements;
- $k_v(\cdot,\cdot)$ - reproducing kernel of the Hilbert Space $\mathcal{H}_v$, computed from a kernel function.

Some of these changed values during testing, others were kept constant. For instance, the loss function used was the squared difference

$$\mathcal{L}(y_i, f_v(\mathbf{x}_i)) = (y_i - f_v(\mathbf{x}_i))^2. \tag{5.4}$$

A Gaussian kernel was used, computed from

$$k_v(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma_v}\right), \tag{5.5}$$

where for view v

$$\sigma_v = \frac{1}{{N_v}^2} \sum_{i,j=1}^{N_v} \|\mathbf{x}_i - \mathbf{x}_j\|^2. \tag{5.6}$$

Finally, the regularization term, ν_v (for view v) was computed as

$$\nu_v = \left(\sum_{i=1}^{N_v} \|\mathbf{x}_i\|/N_v\right)^{-1}. \tag{5.7}$$

Both ν_v and σ_v were computed from the labeled examples of each view. In the testing, 2 views were used, achieved with a sequential split of features. Since every view provides its estimation on the data, the final model's estimation is obtained by averaging the values for the 2 views. The remaining parameters/variables were changed according to the specific test conducted.

5.3.2 Other Methods

The other popular soft sensor model approaches used for comparison purposes were:

- Partial Least Squares (PLS) - model training was done using the function *plsregress()* (*MATLAB®*'s implementation of the *SIMPLS* algorithm), with 34 (total number of features) components;

- Support Vector Machines (SVM) - using the Least-Squares Support Vector Machines implementation [Suykens *et al.*, 2002] through the *LS-SVM Lab Toolbox* [Suykens *et al.*, 2011]. The tuning parameters were computed by the function *tunelssvm()* using leave-one-out cross-validation, and the training was performed using the function *trainlssvm()*;

- Neural Networks (NN) - using the function fitting neural network functionality provided by *MATLAB®*'s *Neural Network Toolbox*. A neural network was created with one hidden layer with 10 neurons by the *fitnet()* function, and trained by the function *train()*.

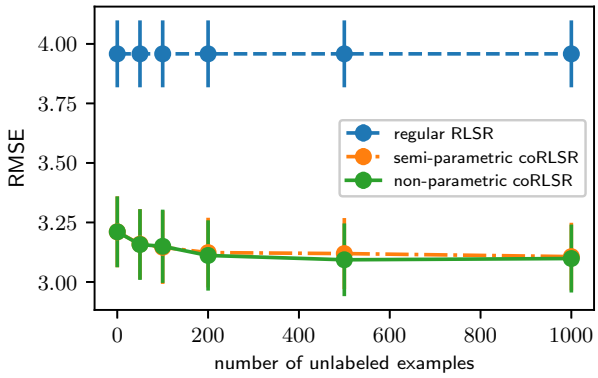


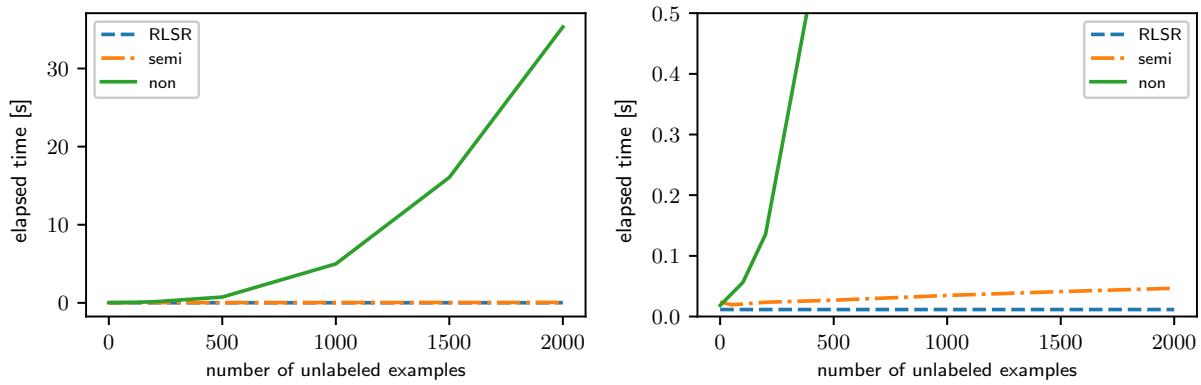
Figure 5.2: Acidity number predictive performance comparison between the RLSR and the semi-supervised and non-supervised variants of coRLSR.

5.4 Improvement to RLSR

The first set of tests was conducted in order to infer on the influence of unlabeled examples on the overall model predictive performance. Both forms of the coRLSR (non-parametric exact solution and semi-parametric approximation), both semi-supervised learning methods, were compared to RLSR, a supervised learning method, according to the RMSE metric. A fixed number of 100 labeled examples and a varying number of unlabeled examples (ranging from 0 to 1000) were taken at random from the training dataset. The RMSE was computed on the test dataset using the trained models, averaging its values from 25 runs. In both variants of coRLSR, 2 views were used ($M = 2$), with $\lambda = 0.1$.

The results for the acidity number prediction are shown in Figure 5.2, where the vertical lines on each data point represent the standard deviation of the RMSE metric. For 100 labeled examples and no unlabeled examples both variants of coRLSR outperform RLSR, confirming the advantage of the multi-view algorithms over the single-view approach. With the increase in the number of unlabeled examples, the distance between multi-view and single-view increases as well. As expected, the non-parametric (exact) solution of coRLSR had the lower RMSE of the methods compared. However, for higher number of unlabeled examples the semi-parametric approximation of coRLSR had RMSE of 3.1001 for 5000 and 3.0975 for 10000 unlabeled examples. The non-parametric variant of coRLSR, for these high numbers requires huge computational effort (as it will be explained bellow) and in some cases did not even run due to insufficient computer memory.

Regarding the computational time required to run each of the three methods, the results in Figure 5.3 confirm the theoretical findings. The regular RLSR, as expected, did not make use of unlabeled examples, so its computational cost was constant. The non-parametric coRLSR had the higher cost, as its complexity in the number of unlabeled examples is cubic. Finally, the semi-parametric coRLSR scaled linearly with the number of unlabeled examples, with its time being comparable to that of RLSR. These tests rendered the semi-parametric variant of coRLSR as the most favorable method in its ratio between predictive



(a) Normal view.

(b) Close-up view.

Figure 5.3: Evolution of the required computational time for a fixed number of labeled examples and a varying number of unlabeled examples.

performance and computational cost, being the one used in the comparison against other popular methods.

5.5 Full Tests and Performance Comparison

5.5.1 Parameter Tuning

The λ parameter, as it appears in (3.7), weighs the disagreement between the predictions of different views. In order to minimize the mentioned cost function, it is expected that *the higher the value of λ the smaller the prediction disparity between views*. A quick test was conducted to verify this expectation, varying the values of λ from 0 to 10000 (an exaggerated value for illustrative purposes only). The RMSE between the predictions of both views was computed for each λ, and as expected, the disparity between the prediction of the 2 views decreases as λ increases, as illustrated in Figure 5.4. While a high λ guarantees that all views will have more similar predictions, those predictions might lose accuracy. Then, λ must be tuned so that different views may complement one another without decreasing each other's performance.

To find the optimal value of λ for each output variable, the following test was conducted for each output variable: for a fixed λ, considering only the training set, the model is trained by using 80% of its labeled examples (as well as 1000 unlabeled examples), and the RMSE on the remaining (labeled) examples is computed. This process was repeated for 25 runs (for each λ), each run with random unlabeled examples taken from the entire training set. As seen on Figure 5.5, the tests performed resulted in the optimal values of $\lambda = 0.9$ for the acidity number scenario and $\lambda = 0.3$ for the viscosity. Again, the vertical lines on each data point represent the standard deviation of the RMSE metric.

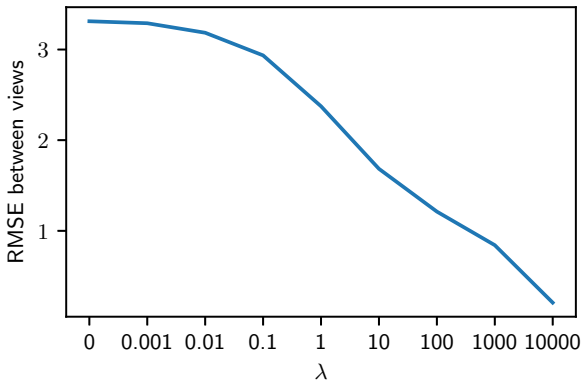
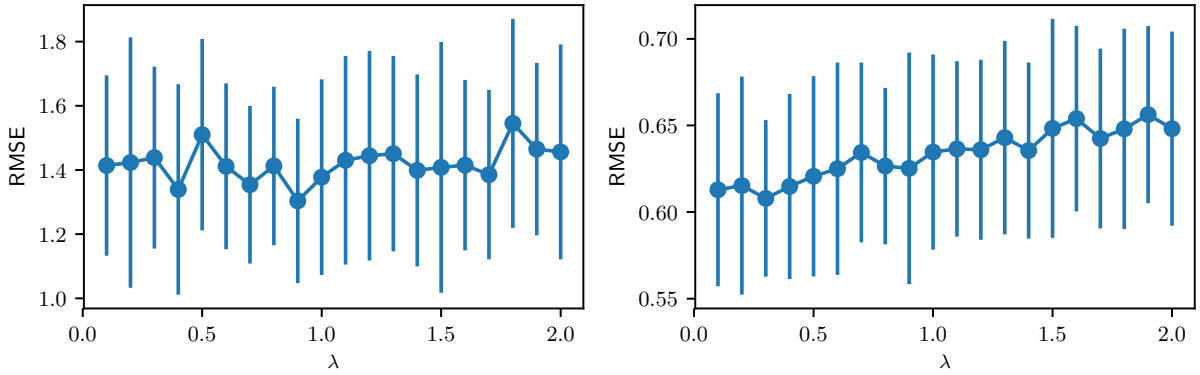


Figure 5.4: Effect of λ on the prediction disparity between views. The disparity is measured by the RMSE between the predictions of both views.



(a) Acidity number.

(b) Viscosity.

Figure 5.5: Influence of the λ parameter on the predictive performance.

5.5.2 Predictive Performance of CoRLSR

After the initial parameter tunning, the model was trained using all labeled examples from the training set, as well as 10000 unlabeled ones. The performance metrics were computed on the test set, and the values are shown in Table 5.2.

Table 5.2: Predictive performance of coRLSR (overall).

coRLSR					
Acidity number			**Viscosity**		
R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
0.86366	2.3394	0.082459	0.88295	0.9863	0.10526

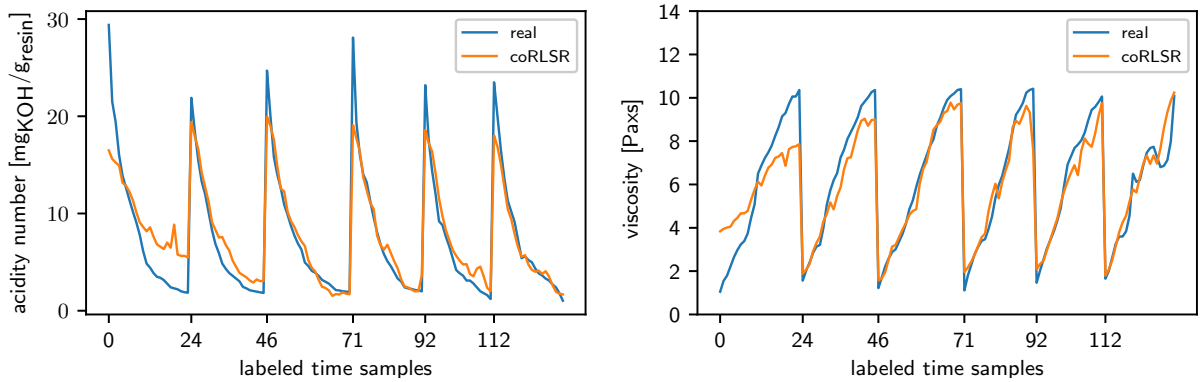
Even though the overall performance results were decent, as the R^2 score indicated that the trained model explains approximately 86% (acidity number) and 88% (viscosity) of the variability of the data, and the NRMSE was at most a little over 0.10, they were not ideal. As the studied scenario is a batch process, and the test subset has data from 6 different

batches, the performance analysis per batch seems relevant. The performance metrics were then computed for each individual batch that constitutes the test subset, as seen in Table. 5.3.

Table 5.3: Predictive performance of coRLSR (per batch).

	coRLSR					
	Acidity number			**Viscosity**		
	R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
#1	0.65162	4.2624	0.15472	0.69604	1.6923	0.18178
#2	0.93479	1.477	0.073557	0.91441	0.85487	0.097145
#3	0.95757	1.2356	0.054239	0.98199	0.40646	0.044252
#4	0.90056	2.1368	0.081837	0.92754	0.8243	0.088539
#5	0.90953	1.7501	0.079552	0.9297	0.73566	0.085572
#6	0.93725	1.52	0.067647	0.85226	0.85482	0.1014

Here it can be seen that coRLSR's estimates for both outputs on batch #1 were not sufficiently accurate, with R^2 scores as low as 0.65 and NRMSE as high as 0.18. This is backed up by Figure 5.6. Here, the coRLSR's estimation for all 6 batches is shown, and the ticks in the x axis represent the begining of a batch. Still, predictive performance on the rest of the batches was quite good, as the performance metrics were quite favorable, except for the viscosity prediction of batch #6.



(a) Acidity number.

(b) Viscosity.

Figure 5.6: Predictive performance of coRLSR.

5.5.3 Predictive Performance Comparison

Comparing with the other popular soft sensor model approaches, on an overall perspective, coRLSR had the most favorable scores for the acidity number prediction, and falling behind only LS-SVM on the viscosity prediction, as Tables 5.4 and 5.5 show.

Table 5.4: Predictive performance comparison (overall) - Acidity number.

Acidity number					
coRLSR			**PLS**		
R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
0.86366	2.3394	0.082459	0.84306	2.5099	0.08847
LS-SVM			**NN**		
R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
0.67251	3.6256	0.1278	0.57322	4.1056	0.14472

Table 5.5: Predictive performance comparison (overall) - Viscosity.

Viscosity					
coRLSR			**PLS**		
R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
0.88295	0.9863	0.10526	0.70608	1.563	0.16681
LS-SVM			**NN**		
R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
0.90939	0.86779	0.092614	0.69595	1.5384	0.16418

Similarly to Subsection 5.5.2, a performance analysis per batch was also performed. Considering first the acidity number prediction, all the approaches were able to produce good results for batches #2-6, with higher than 0.9 R^2 scores and lower than 0.1 NRMSE, as Table 5.6 shows. The struggling task of producing accurate predictions for batch #1 spans across all methods tested, quite evident from the R^2 scores in Table 5.6 and the general inaccuracy visible in Figure 5.7.

Regarding viscosity prediction, the values of which are present in Table 5.7, the performance values for batches #2-4 were generally quite good. PLS and NN achieved worse scores for batch #5, and for batch #6 all methods have decreased accuracy. Moreover, the reoccurring inaccuracy in batch #1 continued, highly evident in Figure 5.8. A possible cause for this could be the presence of irrelevant features, as their influence on the model training stage can decrease predictive performance. Therefore, the devised method for feature ranking and selection was used to hopefully improve the overall predictive performance of coRLSR, as will be described in below in Subsection 5.5.4.

5.5.4 Feature Ranking and Selection Applied to CoRLSR

Following the procedure stated in Section 4.1, after the previous training of the coRLSR model, a random permutation was applied to all time samples of a feature (for all available features), and its ranking was computed according to (4.4). The performance metric used was the RMSE on the training set, using a threshold of $\epsilon = 10^{-2}$ to discard non-relevant

Table 5.6: Predictive performance comparison (per batch) - Acidity number.

Acidity number						
	coRLSR			**PLS**		
	R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
#1	0.65162	4.2624	0.15472	0.55708	4.8061	0.17445
#2	0.93479	1.477	0.073557	0.96026	1.1531	0.057423
#3	0.95757	1.2356	0.054239	0.89347	1.9577	0.08594
#4	0.90056	2.1368	0.081837	0.90762	2.0595	0.078879
#5	0.90953	1.7501	0.079552	0.94476	1.3675	0.062159
#6	0.93725	1.52	0.067647	0.95872	1.233	0.054872
	LS-SVM			**NN**		
	R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
#1	0	8.1697	0.29654	0	10.037	0.36434
#2	0.98651	0.67181	0.033457	0.96101	1.1422	0.056881
#3	0.95782	1.2319	0.054079	0.96088	1.1863	0.052078
#4	0.96241	1.3137	0.050312	0.97392	1.0942	0.041907
#5	0.96338	1.1135	0.050612	0.93521	1.481	0.06732
#6	0.95143	1.3374	0.059518	0.96913	1.0662	0.04745

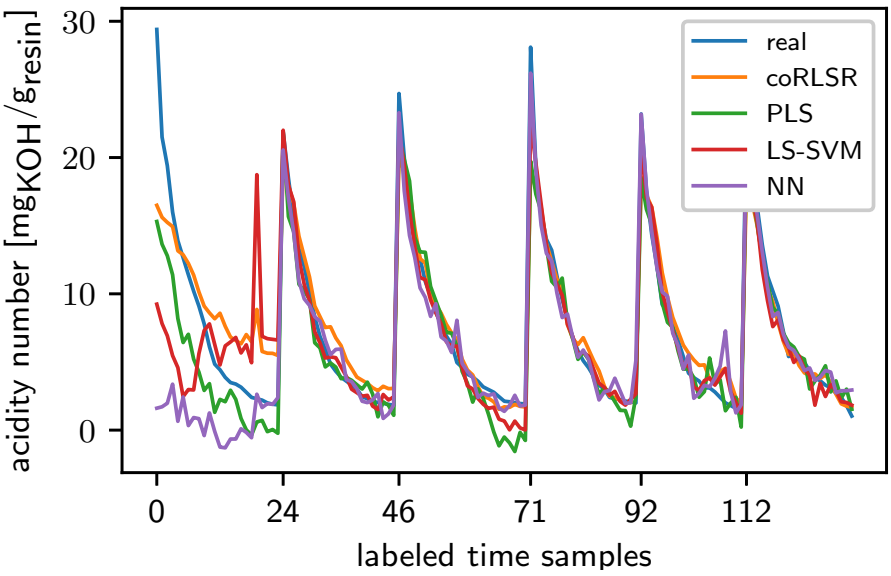


Figure 5.7: Predictive performance comparison - Acidity number.

features. This process was performed for both output variables, and the rank for each individual feature is shown in Figure 5.9, sorted by relevance (only a few labels are visible for the sake of clarity) for the features for which the rank is higher than the defined threshold. For the acidity number estimation, the defined threshold deemed irrelevant 17 features, confirming what was stated above. On the other hand, for the viscosity only 10 features

Table 5.7: Predictive performance comparison (per batch) - Viscosity.

Viscosity						
	coRLSR			**PLS**		
	R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
#1	0.69604	1.6923	0.18178	0	3.192	0.34286
#2	0.91441	0.85487	0.097145	0.93835	0.72551	0.082444
#3	0.98199	0.40646	0.044252	0.96443	0.57118	0.062186
#4	0.92754	0.8243	0.088539	0.92447	0.84158	0.090395
#5	0.9297	0.73566	0.085572	0.82975	1.1448	0.13317
#6	0.85226	0.85482	0.1014	0.81188	0.96458	0.11442
	LS-SVM			**NN**		
	R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
#1	0.78461	1.4246	0.15302	0	3.0718	0.32994
#2	0.92085	0.82206	0.093416	0.91203	0.86665	0.098483
#3	0.98297	0.39522	0.043028	0.97311	0.4966	0.054067
#4	0.9614	0.60165	0.064624	0.94074	0.74549	0.080074
#5	0.9465	0.64179	0.074653	0.80143	1.2364	0.14382
#6	0.83787	0.89547	0.10622	0.77468	1.0556	0.12522

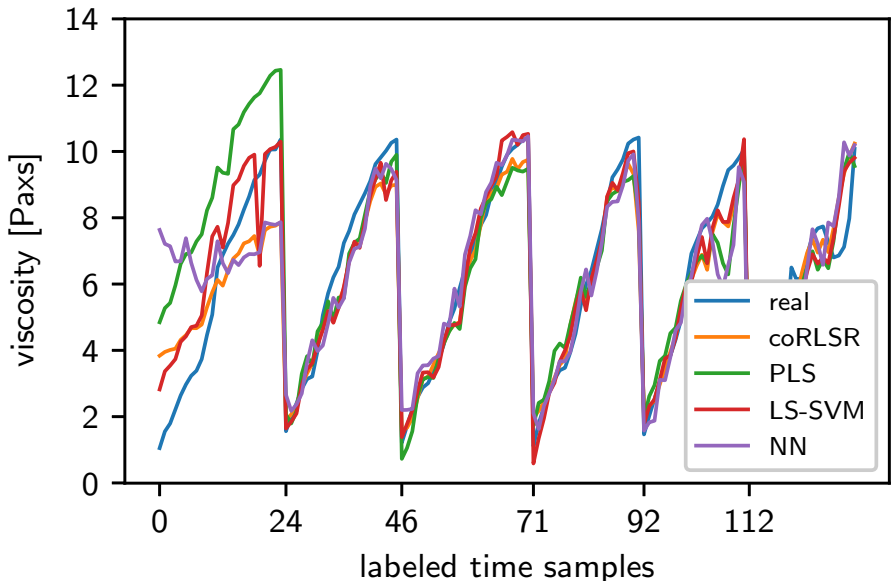
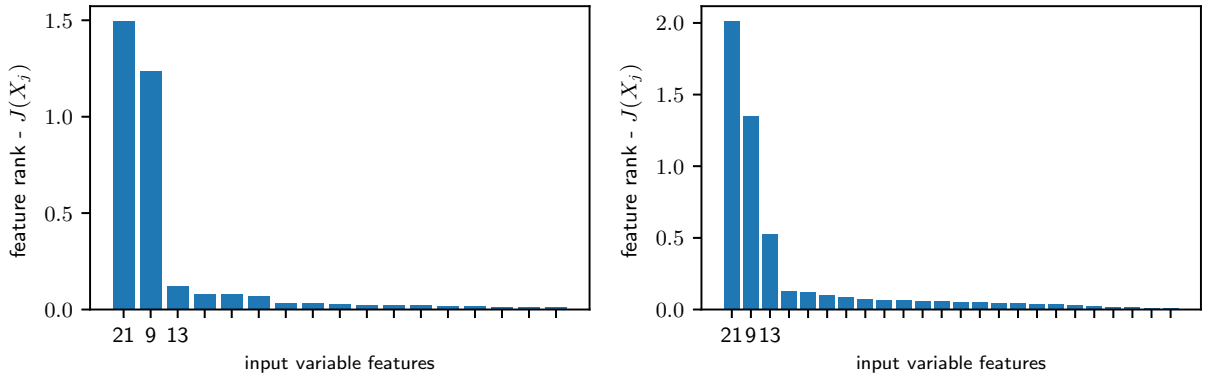


Figure 5.8: Predictive performance comparison - Viscosity.

were discarded, from which it was concluded that the features had generally more relevance on this setting. For both cases the most relevant features were by a fair margin the 21st, 9th and 13th, following the thought that only a few features might be needed for sufficiently accurate estimation.

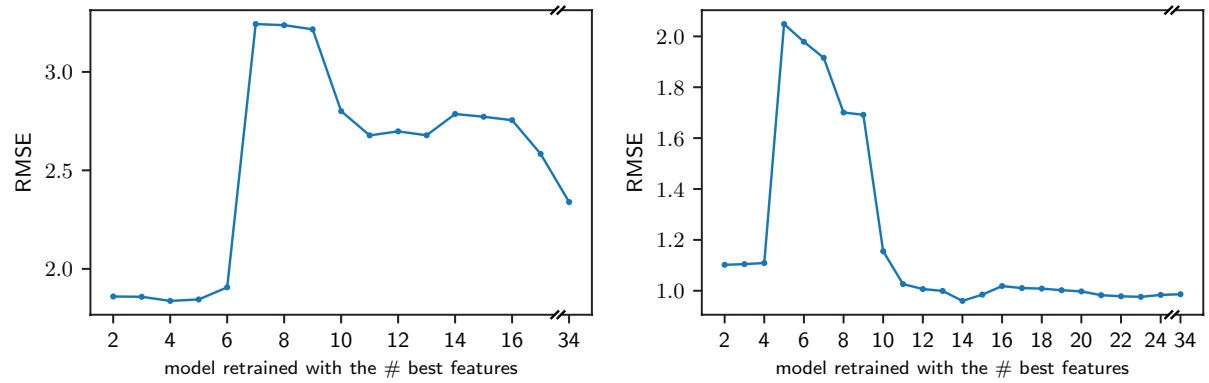
The next step of the procedure was the feature removal through a recursive feature



(a) Acidity number. (b) Viscosity.

Figure 5.9: Feature ranking computed on the training set.

elimination approach. For both scenarios the least relevant feature was progressively removed in order to reach the minimum RMSE. Figure 5.10 illustrates the RMSE obtained for each feature set size. It can be seen that, for the acidity number estimation, few features are needed in order to achieve higher predictive accuracy. The RMSE's rapid decrease seen in Figure 5.10(a) when less than 7 of the best features are used indicated that the optimal number of required features in this case is quite low. In fact, the minimum was reached when only the 4 best features were used. On the viscosity case, the overall higher relevance of features translated to a very different scenario in Figure 5.10(b). The decrease seen when using less than 5 of the best features was not enough to surpass the lower RMSE of the model trained with a higher number of features. The minimum was achieved with the best 14 features used to train the model.



(a) Acidity number. (b) Viscosity.

Figure 5.10: Effect of feature removal on the RMSE - coRLSR.

The effect of the proposed feature ranking and selection method on both the overall and per batch predictive performances can be seen in Tables 5.8 and 5.9 and visualized in Figure 5.11. On these tables, the "$\%^{(R^2)}$" column represents the performance improvement

(in the form of percentage of increase) of the R^2 metric, and the "$\%^{(NRMSE)}$" column represents the performance improvement (in the form of percentage of decrease) of both RMSE and NRMSE metrics (since the improvement values are the same), when going from the situation where all features are used to the situation where only the optimal subset of feartures is used. Positive percentage values correspond to a performance improvement of the considered metric (R^2 or RMSE/NRMSE). In the case of R^2, a performance improvement corresponds to higher scores for the retrained model (the model using feature selection), while for RMSE and NRMSE a performance improvement corresponds to lower values for the retrained model. The improvement in overall performance of the acidity number estimation is notorious, with an increase of 6% on the R^2 score and a reduction in NRMSE of 21%. The difficult task of estimating batch #1 was surpassed, as a NRMSE lower than 0.1 was achieved (equating to a reduction of 43%), while the other batches either saw a increase or a decrease in performance. They all still had good values of NRMSE and R^2, translating to a more balanced predictive performance when considering a per batch analysis. Results on the viscosity prediction indicated that the performance gains while present were small in this case. Here, the gain in overall predictive performance was explained by the increased estimation accuracy for batches #2-5. Batches #1 and #6 saw a decrease in performance, rendering coRLSR unusable for this scenario.

Table 5.8: Feature ranking and selection effect on predictive performance of coRLSR - Acidity number.

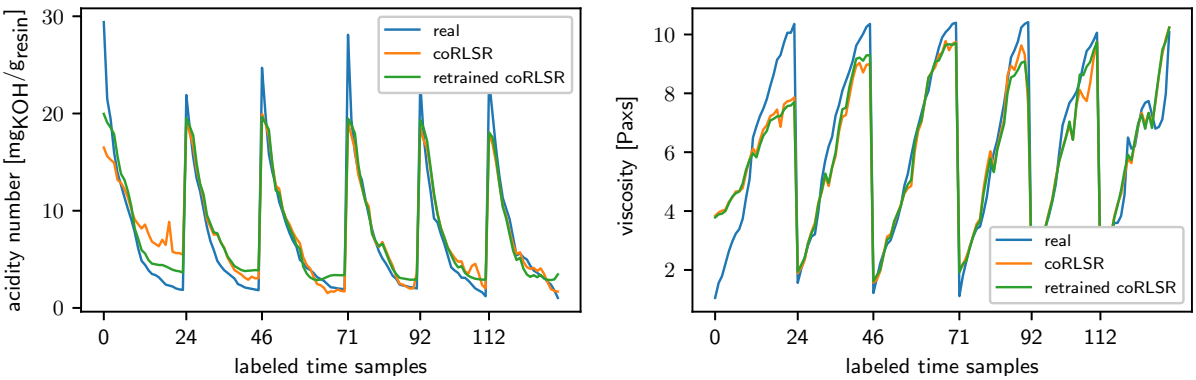
Acidity number								
	coRLSR			**retrained coRLSR**				
	R^2	RMSE	NRMSE	R^2	$\%^{(R^2)}$	RMSE	NRMSE	$\%^{(NRMSE)}$
overall	0.86366	2.3394	0.082459	0.91384	*6*	1.8597	0.065552	*21*
#1	0.65162	4.2624	0.15472	0.88574	*36*	2.4411	0.088605	*43*
#2	0.93479	1.477	0.073557	0.90078	*-4*	1.8219	0.090734	*-23*
#3	0.95757	1.2356	0.054239	0.94664	*-1*	1.3856	0.060824	*-12*
#4	0.90056	2.1368	0.081837	0.9076	*1*	2.0597	0.078885	*4*
#5	0.90953	1.7501	0.079552	0.92611	*2*	1.5816	0.071893	*10*
#6	0.93725	1.52	0.067647	0.9265	*-1*	1.6451	0.073212	*-8*

5.5.5 Feature Ranking and Selection Applied to All Methods

The general nature of the underlying concept used in feature ranking makes it so that it can be applied not only to coRLSR, but to any regression method. Therefore, a similar feature ranking procedure was performed for all methods, and the least relevant features were removed in order for the minimum RMSE to be reached for both output variables. The improvement in the curve-fitting is quite substantial and it can be seen in Figure 5.12 for all

Table 5.9: Feature ranking and selection effect on predictive performance of coRLSR - Viscosity.

Viscosity								
	coRLSR			**retrained coRLSR**				
	R^2	RMSE	NRMSE	R^2	$\%^{(R^2)}$	RMSE	NRMSE	$\%^{(NRMSE)}$
overall	0.88295	0.9863	0.10526	0.88911	*1*	0.96003	0.10246	*3*
#1	0.69604	1.6923	0.18178	0.67953	*-2*	1.7377	0.18665	*-3*
#2	0.91441	0.85487	0.097145	0.94404	*3*	0.69121	0.078546	*19*
#3	0.98199	0.40646	0.044252	0.985	*0*	0.37097	0.040389	*9*
#4	0.92754	0.8243	0.088539	0.92809	*0*	0.82116	0.088202	*0*
#5	0.9297	0.73566	0.085572	0.96554	*4*	0.51508	0.059914	*30*
#6	0.85226	0.85482	0.1014	0.83928	*-2*	0.89157	0.10576	*-4*



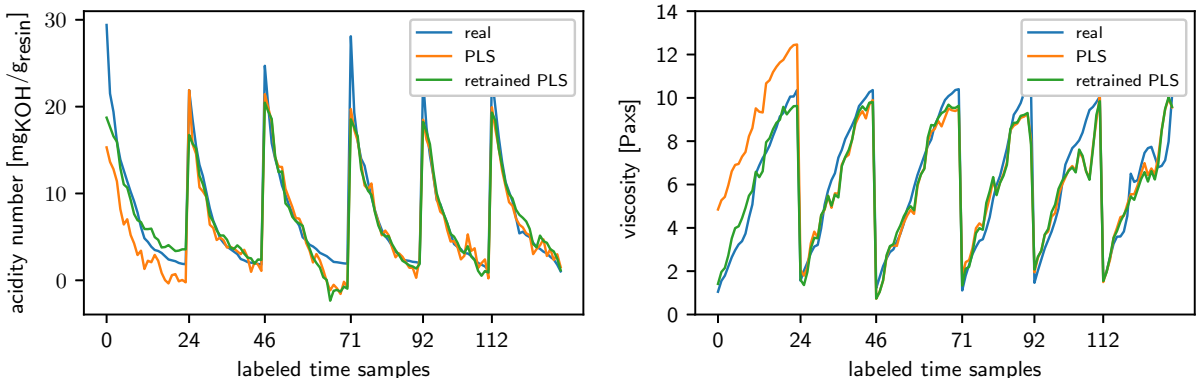
(a) Acidity number.

(b) Viscosity.

Figure 5.11: Feature ranking and selection effect on predictive performance - coRLSR.

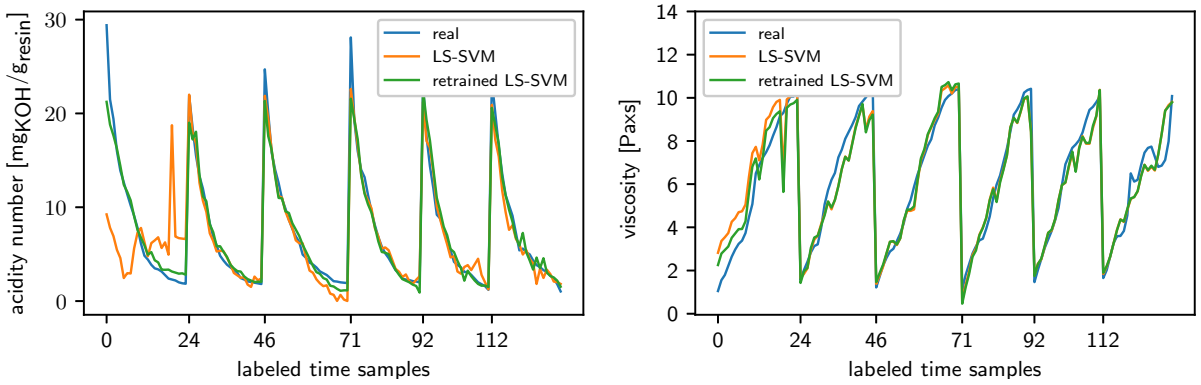
methods and output variables.

Considering a quantitative look, an analysis similar to the one performed for coRLSR in Subsection 5.5.4 was done. Looking at the results of the acidity number prediction, the values on Table 5.10 showed that all methods were able to greatly benefit from the feature ranking and selection approach, with overall R^2 scores greater than 0.9 and NRMSE lower than 0.1. In Table 5.10 (and also in Table 5.11, below), the "%" columns represent the performance improvement of the performance index shown in the column to the left of the respective "%" sign. The performance improvement takes the form of a percentage of increase for the R^2 index, and a percentage of decrease for the cases of the RMSE and NRMSE and indexes. The overall performance improvement was explained mostly by the very large decrease in prediction error in batch #1. The NN model saw the greatest reduction in NRMSE of about 89%, followed by LS-SVM, with 77%, and PLS, with 44%. On the R^2 metric, the ∞ symbol for both NN and LS-SVM is justified by their scores of 0 for this batch in the case without



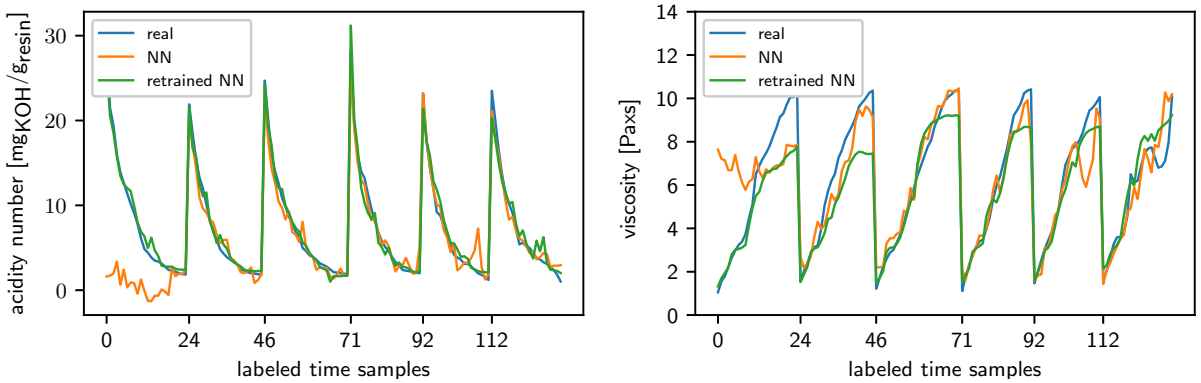
(a) PLS - Acidity number.

(b) PLS - Viscosity.



(c) LS-SVM - Acidity number.

(d) LS-SVM - Viscosity.



(e) NN - Acidity number.

(f) NN - Viscosity.

Figure 5.12: Feature ranking and selection effect on the predictive performance of all methods.

feature selection. This constitutes a very important increase in performance, since their improved R^2 scores were greater than 92% on NN and LS-SVM. On the remaining batches a more balanced performance variation was seen, with some scores improving and others decreasing by not so great amounts, yielding an overall performance improvement tendency. The methods that took most advantage of the devised feature selection method were clearly LS-SVM and NN. NN was able to achieve R^2 greater than 0.95 across all batches, constituting

a nearly-perfect estimation, as it can be seen in Figure 5.12(e). LS-SVM followed with a still impressive predictive performance, as observed in Figure 5.12(c). Following the thought that in most cases few features are in fact relevant, these retrained models used at most 10 features (in the case of NN), reinforcing the importance of feature selection in soft sensor modeling.

Table 5.10: Feature ranking and selection effect on predictive performance of all methods - Acidity number.

Acidity number										
	retrained coRLSR					**retrained PLS**				
	R^2	%	RMSE	NRMSE	%	R^2	%	RMSE	NRMSE	%
overall	0.91384	*6*	1.8597	0.065552	*21*	0.90355	*7*	1.9676	0.069354	*22*
#1	0.88574	*36*	2.4411	0.088605	*43*	0.86321	*55*	2.6709	0.096948	*44*
#2	0.90078	*-4*	1.8219	0.090734	*-23*	0.94069	*-2*	1.4086	0.07015	*-22*
#3	0.94664	*-1*	1.3856	0.060824	*-12*	0.86186	*-4*	2.2293	0.097863	*-14*
#4	0.9076	*1*	2.0597	0.078885	*4*	0.89337	*-2*	2.2127	0.084745	*-7*
#5	0.92611	*2*	1.5816	0.071893	*10*	0.95284	*1*	1.2635	0.057433	*8*
#6	0.9265	*-1*	1.6451	0.073212	*-8*	0.94694	*-1*	1.3978	0.062206	*-13*
	retrained LS-SVM					**retrained NN**				
	R^2	%	RMSE	NRMSE	%	R^2	%	RMSE	NRMSE	%
overall	0.95711	*42*	1.3121	0.046249	64	0.97317	*88*	1.0378	0.036581	*76*
#1	0.92938	∞	1.9192	0.069661	77	0.97799	∞	1.0715	0.038891	*89*
#2	0.97127	*-2*	0.98045	0.048827	-46	0.98319	*2*	0.74987	0.037344	*34*
#3	0.9684	*1*	1.0662	0.046804	13	0.98975	*3*	0.60733	0.026661	*49*
#4	0.94706	*-2*	1.5591	0.059713	-19	0.96482	*-1*	1.2709	0.048675	*-16*
#5	0.96849	*1*	1.0328	0.046946	7	0.96758	*3*	1.0476	0.047618	*29*
#6	0.97591	*3*	0.94181	0.041914	30	0.95056	*-2*	1.3492	0.060046	*-27*

Considering the viscosity prediction, the statement in the previous section on the general greater relevance of features remains true in this scenario. Performance index results for the viscosity case are shown in Table 5.11. The retrained models used most of the features from the total feature set, possibly leading to a smaller performance gain by using feature selection. Still, PLS saw a decrease of 46% on the NRMSE value and LS-SVM a decrease of 9%, on the overall perspective. PLS and LS-SVM represent the best candidates for accurate estimation, with favorable scores of R^2 and NRMSE, the LS-SVM being the most balanced across all different batches.

CoRLSR fell behind the other methods in terms of predictive performance. While on the acidity number the prediction performance scores were good, on the viscosity case the estimation was not sufficiently accurate. This means that, for this dataset, coRLSR was not able to exploit the large number of unlabeled examples available, not reaching to the levels of predictive performance of SVM or NN approaches. On the other hand, the proposed method

Table 5.11: Feature ranking and selection effect on predictive performance of all approaches - Viscosity.

Viscosity										
	retrained coRLSR					**retrained PLS**				
	R^2	%	RMSE	NRMSE	%	R^2	%	RMSE	NRMSE	%
overall	0.88911	*1*	0.96003	0.10246	*3*	0.91558	*30*	0.83765	0.089397	*46*
#1	0.67953	*-2*	1.7377	0.18665	*-3*	0.95685	∞	0.63766	0.068492	*80*
#2	0.94404	*3*	0.69121	0.078546	*19*	0.94377	*1*	0.69288	0.078736	*4*
#3	0.985	*0*	0.37097	0.040389	*9*	0.9718	*1*	0.50859	0.055372	*11*
#4	0.92809	*0*	0.82116	0.088202	*0*	0.91902	*-1*	0.87142	0.0936	*-4*
#5	0.96554	*4*	0.51508	0.059914	*30*	0.81488	*-2*	1.1938	0.13886	*-4*
#6	0.83928	*-2*	0.89157	0.10576	*-4*	0.78736	*-3*	1.0255	0.12165	*-6*
	retrained LS-SVM					**retrained NN**				
	R^2	%	RMSE	NRMSE	%	R^2	%	RMSE	NRMSE	%
overall	0.92539	*2*	0.78744	0.084039	*9*	0.85932	*20*	1.0813	0.1154	*30*
#1	0.86988	*11*	1.1073	0.11893	*22*	0.7931	∞	1.3962	0.14997	*55*
#2	0.9143	*-1*	0.85541	0.097205	*-4*	0.6889	*-24*	1.6298	0.1852	*-88*
#3	0.97971	*-0*	0.43136	0.046963	*-9*	0.959	*-1*	0.61324	0.066766	*-23*
#4	0.95887	*-0*	0.62107	0.06671	*-3*	0.92638	*-2*	0.83089	0.089247	*-11*
#5	0.94913	*0*	0.62582	0.072795	*2*	0.93861	*17*	0.68746	0.079966	*44*
#6	0.84248	*1*	0.88265	0.1047	*1*	0.837	*8*	0.89786	0.10651	*15*

for feature ranking and selection showed great potential as the improvement in predictive performance is substantial across all methods tested. As it relies on random permutation of a feature after the model is trained, the relevance of the feature for both the problem in question and the specific prediction method used is taken into consideration. Following this, the multi-view nature of coRLSR was combined with the existence of feature ranking information in order to infer on possible further improvements to its predictive performance.

5.5.6 Strategy of View Creation for Multi-View Methods

So far, in all testing, the 2 views used in coRLSR were created by sequentially splitting the available feature set, forming two disjoint sets of features. In order to briefly infer on the influence of certain types of view splits, a quick test was devised using the acidity number scenario, as only 4 features were needed to achieve the minimum RMSE. The feature indexes of the these 4 features are, in decreasing order of importance, $\{21, 9, 13, 33\}$, and they were split in the following manner: 1^{st} view: $\{9, 13\}$, 2^{nd} view: $\{21, 33\}$. After some empirical evaluation and rearranging of the constitution of the views using the same features as before, the following split was considered: 1^{st} view: $\{21, 13\}$, 2^{nd} view: $\{9, 33, 13\}$. The split was achieved not sequentially, but by assigning the features according to their relevance

alternating between the two views (neither the view's external order nor the internal order matter to the final result). Additionally, the 13$^{\text{th}}$ feature was also assigned to the 2$^{\text{nd}}$ view, creating two sets that are not disjoint. Retraining was then performed with the mentioned view split, and the model's prediction yet again obtained by averaging both views' estimations.

Table 5.12: Influence of a different view creation strategy on predictive performance.

	Acidity number					
	retrained coRLSR			**newly retrained coRLSR**		
	R^2	RMSE	NRMSE	R^2	RMSE	NRMSE
overall	0.91384	1.8597	0.065552	0.93034	1.6721	0.058941
#1	0.88574	2.4411	0.088605	0.89139	2.38	0.086388
#2	0.90078	1.8219	0.090734	0.92233	1.6119	0.080276
#3	0.94664	1.3856	0.060824	0.96432	1.133	0.049739
#4	0.9076	2.0597	0.078885	0.92429	1.8645	0.071409
#5	0.92611	1.5816	0.071893	0.9416	1.4062	0.063916
#6	0.9265	1.6451	0.073212	0.95587	1.2748	0.056733

The results are presented in Table 5.12 and show that there was a substantial improvement in predictive performance using the newly retrained version of coRLSR. For all batches, the R^2 scores increased, and the RMSE/NRMSE values were reduced, leading to a better overall predictive performance, as measured by the R^2 and RMSE/NRMSE. This superficial testing was performed to further extend the applicability of the proposed algorithm as multi-view methods can take advantage of the additional feature information. Results also showed that future developments in the algorithm could be done in order to accommodate this deeper connection between multi-view methods and feature ranking/selection.

5.5.7 On the Value of Soft Sensors

One of the advantages of soft sensors, as mentioned in Section 2.4, is that they can provide real-time estimates of the desired/target variable(s) (when their accuracy is good). This is particularly useful in scenarios such as the one described in Section 5.1, since quality variables are only available through laboratory analysis, with delays of up to 20 [min]. In contrast, after the training stage, the soft sensor can provide estimates for the target variables as quickly as process/input variables are made available (in this case, every 30 [s]). The difference in the number/frequency of values of quality variables available is noticeable. Considering, for example, batch #3, Figure 5.13 illustrates this discrepancy in number the number of samples available of the target variable. For this batch only, the soft sensor lead to an increase from 25 measurements to approximately 5000 measurements, which could greatly benefit the process control, as immediate (while accurate) estimations provide more controllability capabilities

than 20 [min] delayed measurements.

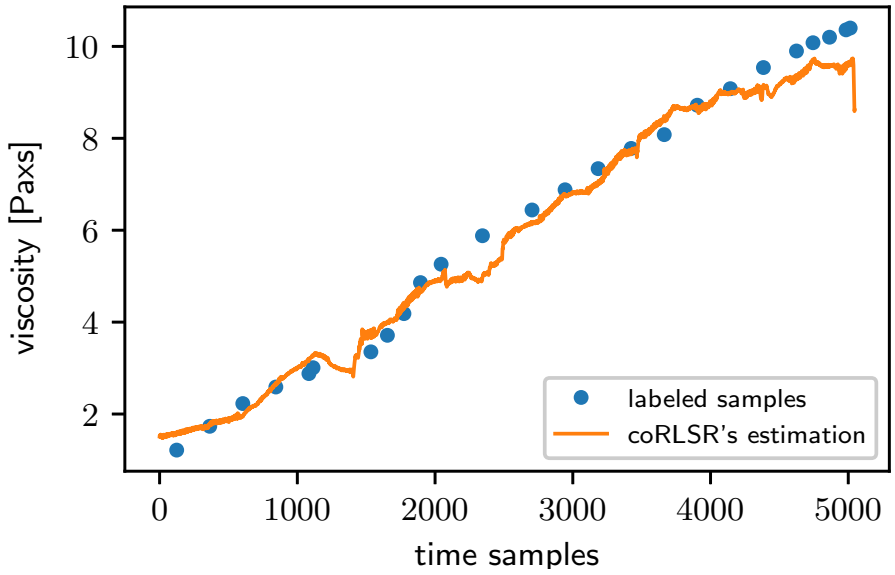


Figure 5.13: Soft sensor output vs. laboratory analysis for batch #3 - Viscosity.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This dissertation presented a semi-supervised regression method that is able to benefit from the existence of unlabeled data, as its improvement in predictive performance with the increase in number of unlabeled examples was notorious. Moreover, a new approach to feature ranking and selection based on sensitivity analysis was proposed, focused on flexibility and low computational cost. A real-world dataset was used in order to more accurately test the proposed method and compare it with the most popular approaches used in soft sensing. With no feature ranking and selection procedure performed, coRLSR's predictive performance was competitive with the other approaches. Using the proposed method for feature ranking and selection produced substantial improvements in performance for all methods. Here, coRLSR's results were not as good as LS-SVM and NN, for example, being in some cases an inadequate method for use in soft sensor modeling. Still, a look in the multi-view nature of coRLSR allowed for a deeper connection with the feature ranking method, showing that it is possible to further improve its performance by adequately defining the views. The relevance of soft sensors was yet again stated, as the capability of providing real-time estimations is of the utmost importance in process control, specially in scenarios where quality information is not readily available (such as in the presented real-world industrial setting).

6.2 Future Work

Further experimentation on coRLSR's could improve its performance. In this work only Gaussian kernels were used, and so other types of kernels could be investigated and the corresponding influence measured. The quick testing done on view creation hinted that the influence of the design of the views on multi-view methods like coRLSR could be explored, since feature ranking information is available. The method for feature ranking and selection presented can also be improved, as a way to automatically select the direct feature rejection

threshold value for each different scenario could improve the performance of the method. Semi-supervised learning is an exciting field of machine learning, and as research interest grows, more methods will be developed with stronger predictive capabilities.

Appendix A

Published Paper

This appendix includes a paper that resulted from the work of this dissertation. The paper is entitled "Semi-Supervised Soft Sensor and Feature Ranking Based on Co-Regularised Least Squares Regression Applied to a Polymerization Batch Process" [Ferreira *et al.*, 2017], and was published in the "IEEE 15th International Conference on Industrial Informatics (INDIN 2017)", that was held on July 24-26, 2017, in Emden, Germany.

Semi-supervised Soft Sensor and Feature Ranking Based on Co-Regularised Least Squares Regression Applied to a Polymerization Batch Process

Vasco Ferreira*, Francisco A. A. Souza*, Rui Araújo*
*Institute of Systems and Robotics (ISR-UC), and
Department of Electrical and Computer Engineering (DEEC-UC),
University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal
Email: {vascoferreira,fasouza,rui}@isr.uc.pt

***Abstract*—In this paper a semi-supervised regression model based on co-training is applied on the soft sensor context, together with a feature ranking approach which has the purpose of removing irrelevant features. The description of both the methods of semi-supervised regression and feature ranking, as well as the theorethical foundation of the proposed feature ranking approach are also given. To evaluate the proposed methodology, a real-world polymerization industrial process was used as example. The results demonstrate that the devised feature ranking and selection improves the semi-supervised regression model.**

I. Introduction

Soft sensors are inferential/predictive models that use on-line available sensors, also called as easy-to-measure variables (e.g. temperature, flow rate, etc) to predict quality variables which cannot be automatically measured at all, or can only be measured at high cost, sporadically, or with high delays (these are also refereed as hard-to-measure variables; e.g. laboratory measurements) [1]. They are important tools for application in many industrial processes, such as pulp and paper mills, cement kiln, just to give a few examples.

In traditional approaches, a plenty of available unlabeled data (i.e. samples where the easy-to-measure variables do not have a corresponding hard-to-measure value) is not used during the model learning. The presence of unlabeled data comes from the fact that the frequency at which the easy-to-measure variables are made available is usually much higher than the measurement frequency of the hard-to-measure variables such as laboratory measurements, and the synchronization between the easy-to-measure and hard-to-measure variables is done in accordance to the slow sampling rate of the hard-to-measure variables, usually by excluding the unlabeled data [2], [3]. Other approaches usually make use of FIR (finite impulse response) filters to sincronize the unlabeled and labeled data [4], [5]. However, the use of both the labeled and unlabeled data (partially labeled data) for soft sensors applications has shown to be an effective approach when compared to the model trained with only the labeled data [6], [7]. Despite this fact, there are currently few methodologies to deal with irrelevant features in such kind of settings. Then, following the idea that a variable ranking approach might improve the performance accuracy in the semi-supervised context, as it does in supervised learning, this paper proposes a new general algorithm for variable ranking and its application in the semi-supervised learning area with application to the soft sensors context.

There are two types of soft-sensors: model-driven and data-driven, the key difference being that on data-driven (also known as black-box) the relationship between the input and output is constructed based on empirical data, while on model-driven (also known as white-box) such relationship is based on mathematical/physical models [1]. The large volume of process data available in industrial plants contributed to the rising popularity of data-driven soft sensor approaches, the most popular methods being based on partial least squares (PLS) [8], principal component regression (PCR) [9], artificial neural network (ANN) [10], and support vector machine (SVM) [11]. The choice of the model to use is not straightforward, since every method has its advantages and disadvantages, and possible scenarios in which to apply. The existence of unlabeled data on some industrial settings might favor semi-supervised methods, as they are able to incorporate such information to better map the input-output relationship. Although all above-mentioned methods are supervised, semi-supervised versions have been developed, showing the potential of the concept in soft sensor modeling [6], [7], [12], [13], [9]. Feature selection is a fundamental stage in soft sensor development. As the number of features/variables to be considered in the modeling stage is rather high, selecting only the most relevant ones can lower the complexity of the problem at hand. This is imperative, as in most industrial settings there are time constraints that can only be met if the number of features considered is kept low [14]. Many strategies for feature selection have been proposed in conjunction with soft sensor design to various industrial settings, such as fabric textile [14], refining process [15], semiconductor manufacturing [16], chemical polymerization [17], and chemical process control [18]. Most methods involve an intricate number of steps in order to evaluate all features and select the most relevant. Some use cross-validation to evaluate subsets of features [15], obtaining those subsets randomly. Others are based on methods

978-1-5386-0836-4/17/$31.00 ©2017 IEEE

like SVM [16] and PLS [18]. These approaches can be quite expensive on a computational perspective, and can even be prohibitive in a high-dimensionality scenario [19].

To further improve the predictive performance of semi-supervised learning, this paper resorts to a strategy of feature ranking and selection based on sensitivity analysis. In sensitivity analysis, the importance of an input is measured by computing the variation of the output when the input is perturbed. The proposed methodology is theoretically justified based on common assumptions. This method allows the measurement of the influence of an individual feature without resorting to retraining the model, lowering the potential computational complexity. The experiments, performed on a real-world industrial batch polymerization process, show that ranking the feature set and selecting only some of the most relevant features from it, leads to a substantial improvement in predictive performance, allowing for a more accurate representation of the problem in question while keeping the complexity and computational cost low.

Section II defines notation. In section III the coRLSR and its semi-parametric approximation is detailed. The method of feature selection and ranking is discussed in section IV. A brief explanation on the real-world application used in experiments is provided in Section IV. Finally, Section VI contains the experimental results and Section VII the final remarks.

II. Notation

The notation used here is defined as follows. Assume that finite random variables are represented by capital letters and their values for the corresponding lowercase letters, e.g. random variable A, and corresponding value a. Matrices and vectors are represented by boldface capital letters, e.g. $\mathbf{A}$ and boldface lowercase letters, e.g. $\mathbf{a}$, respectively.

The input and output/target variables are defined as $X = \{X_1, \ldots, X_d\}$ and Y, respectively. The variables $X_1, \ldots, X_d$ can take N different values as $\{x_{ij} \in X_j | j = 1, \ldots, d \wedge i = 1, \ldots, N\}$, and similarly for Y as $\{y_i \in Y | i = 1, \ldots, N\}$. The input and output vector at instant i are defined as $\mathbf{x}_i = [x_{i1}, \ldots, x_{id}]^T$ and y_i, where $\mathbf{X}$, with elements $X_{ij} = x_{ij}$, and $\mathbf{y}$, with elements $y_{i,1} = y_i$ are the input matrix and output vector containing all the N examples, respectively. The set of examples is defined as $\mathbf{\Phi} = (\mathbf{x}_i, y_i)_{i=1}^N$. The probability that a random variable takes the value a, i.e. $\{A = a\}$, is defined as $p(a)$.

III. Efficient Co-Regression

The Co-Regression algorithm, proposed by Brefeld *et al.* [20], is based on a kernel approach. In those methods, given set of training data $\mathbf{\Phi}$, the goal is to find the following solution

$$f(\cdot) = \underset{f^*(\cdot) \in \mathcal{H}}{\arg\min} \left\{ \sum_{i=1}^{N} \mathcal{L}(y_i, f^*(\mathbf{x}_i)) + \nu \Omega[f^*(\cdot)] \right\} \quad (1)$$

where $\mathcal{H}$ is a Hilbert space of functions, $\mathcal{L}(y, \cdot)$ is a convex loss function, $\nu > 0$ is a parameter and $\Omega[f(\cdot)]$ is a regularization term.

For M-view learning ($M \geq 1$), the objective is to find M new functions from different Hilbert Spaces $\mathcal{H}_v$, indexed by v, where each $\mathcal{H}_v$ is defined by different instances, descriptions, features, and/or different kernel functions, such that the error of each function on the training data and the disagreement between the functions on the unlabeled data is as small as possible. Hence, given a set with M sets containing labeled samples $\Gamma = \{\mathbf{\Phi}_1, \ldots, \mathbf{\Phi}_M\} \subseteq \mathcal{M}$, where $\mathbf{\Phi}_v = \{\mathbf{x}_i, y_i\}_{i=1}^{N_v}$ and $v = 1, \ldots, M$, and a set of unlabeled samples $\mathbf{\Phi}_z^* = \{\mathbf{x}_z\}_{z=1}^{Z}$, the objective is to find functions $f_1 : \Gamma \to \mathbb{R}, ..., f_M : \Gamma \to \mathbb{R}$ from different Hilbert Spaces $\mathcal{H}_1, ..., \mathcal{H}_M$ that minimize

$$Q(f_1, ..., f_M) = \sum_{v=1}^{M} \left[\sum_{i=1, \mathbf{x_i} \in \mathbf{\Phi}_v}^{N_v} \mathcal{L}(y_i, f_v(\mathbf{x}_i)) + \nu_v \|f_v(\cdot)\|^2 \right] + \lambda \sum_{u,v=1}^{M} \sum_{z=1, \mathbf{x}_z \in \mathbf{\Phi}_z^*}^{Z} \mathcal{L}(f_u(\mathbf{x}_z), f_v(\mathbf{x}_z)) \quad (2)$$

where, $\mathcal{L}(\cdot)$ is the loss function, $\|f_v(\cdot)\|^2$ is the regularization term, ν_v is the regularization parameter and λ is a parameter that weights the influence of pairwise disagreements. According to the *representer theorem* (Wahba [21]; Schölkopf *et al.* [22]), the solutions of Eq. (2) have always the following form

$$f_v^*(\mathbf{x}_i) = \sum_{\mathbf{x} \in \bigcup_{\mathbf{\Phi}_v \in \Gamma} \mathbf{\Phi}_v \bigcup \mathbf{\Phi}_z^*} \alpha_v(\mathbf{x}) k_v(\mathbf{x}, \mathbf{x_i}), \quad (3)$$

where $k_v(\cdot, \cdot)$ is the reproducing kernel of the Hilbert Space $\mathcal{H}_v$, and $\alpha_v(\mathbf{x})$ is a coefficient associated with $\mathbf{x}$.

In this way, the vector $[\mathbf{f}_v]_{i,1} = f_v(\mathbf{x}_i)$, composed of the elements $\mathbf{x}_i \in \bigcup_{\mathbf{\Phi}_v \in \Gamma} \mathbf{\Phi}_v \bigcup \mathbf{\Phi}_z^*$, can be expressed as $\mathbf{K}_v \mathbf{c}_v$ and $\|f_v(\cdot)\|^2$ as $\mathbf{c}_v^T \mathbf{K}_v \mathbf{c}_v$, where $[\mathbf{K}_v]_{ij} = k_v(\mathbf{x}_i, \mathbf{x}_j)$ and $[\mathbf{c}_v]_{i,1} = \alpha_v(\mathbf{x}_i)$. $\mathbf{K}_v$ is a strictly positive definite kernel matrix, i.e., it is symmetric and has no negative, and no zero eigenvalues. The notation used for each view's labels is $[\mathbf{y}_v]_i = y_i$ for $i = 1, \ldots, N_v$. For standard kernel methods like Eq. (1), defining the loss function as the squared difference between real and predicted output, $\mathcal{L}(y_i, f_v(\mathbf{x}_i)) = (y_i - f_v(\mathbf{x}_i))^2$ gives the ridge regression (Sauders *et al.* [23]) or regularised least squares regression (RLSR) solutions.

A. Semi-Parametric Approximation of coRLSR

This paper focuses on the semi-parametric approximation of coRLSR proposed in [20], where the authors also proposed a exact solution of the (non-parametric) coRLSR, but such solution has cubic time complexity in the number of unlabeled examples, which is unfeasible in semi-supervised soft sensors applications, since the number of unlabeled samples is large. On the other hand, the semi-parametric solution is able to scale linearly with the number of unlabeled examples, and provides the same performance solution as the exact solution.

With N_v training examples in view v and Z unlabeled examples, Eq. (2) can be rephrased as the following coRLSR

optimization problem, in which the goal, for fixed $\lambda, \nu \geq 0$, is to minimize the following cost function

$$Q(\mathbf{C}) = \sum_{v=1}^{M} \left(\|\mathbf{y}_v - \mathbf{L}_v \mathbf{c}_v\|^2 + \nu \mathbf{c}_v^T \mathbf{L}_v \mathbf{c}_v \right) + \lambda \sum_{u,v=1}^{M} \|\mathbf{U}_u \mathbf{c}_u - \mathbf{U}_v \mathbf{c}_v\|^2 \tag{4}$$

over $\mathbf{C} = [\mathbf{c}_1, ..., \mathbf{c}_M] \in \mathbb{R}^{N_1} \times \ldots \times \mathbb{R}^{N_M}$. $\mathbf{L}_v \in \mathbb{R}^{N_v \times N_v}$ and $\mathbf{U}_v \in \mathbb{R}^{Z \times N_v}$ are computed from a strictly positive definite kernel function and from a positive definite kernel matrix $\mathbf{K}_v \in \mathbb{R}^{(N_v+Z)\times(N_v+Z)}$ as

$$\mathbf{K}_v = \begin{pmatrix} \mathbf{L}_v & \mathbf{U}_v^T \\ \mathbf{U}_v & * \end{pmatrix}$$

where the part marked by * is not needed.

Getting the derivative of (4) with respect $\mathbf{c}_v$, yields

$$\nabla_{\mathbf{c}_v} Q(\mathbf{C}) = 2\mathbf{G}_v \mathbf{c}_v - 2\mathbf{L}_v \mathbf{y}_v - 4\lambda \sum_{u:u\neq v} \mathbf{U}_v^T \mathbf{U}_u \mathbf{c}_u.$$

where

$$\mathbf{G}_v = \mathbf{L}_v^2 + \nu \mathbf{L}_v + 2(M-1)\lambda \mathbf{U}_v^T \mathbf{U}_v.$$

At the optimum the following relation holds

$$[\nabla_{\mathbf{c}_1} Q(\mathbf{C}), \ldots, \nabla_{\mathbf{c}_2} Q(\mathbf{C})]^T = \mathbf{0},$$

and then the exact solution for the optimum of (4) can be found as

$$\begin{bmatrix} \mathbf{G}_1 & -2\lambda \mathbf{U}_1^T \mathbf{U}_2 & \cdots \\ -2\lambda \mathbf{U}_2^T \mathbf{U}_1 & \mathbf{G}_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 y_1 \\ \mathbf{L}_2 y_2 \\ \vdots \end{bmatrix}.$$

The solutions can be found in time $O(M^3 N^2 Z)$ (assuming $Z \geq N = \max_v N_v$). Further details on coRLSR can be found in the original paper [20].

IV. Feature Ranking and Selection

The coRLSR method, similarly to other kernel methods, aims to find the optimal solution in the Hilbert space. It is given by a linear combination of kernel functions "centered" on the set of training instances (labeled and unlabeled). In this sense, the direct relationship between input and output is not straightforward. This means that there is no direct way to assess the influence that a variable has in the output.

Assume that the output of a regression model, defined as $f(\mathbf{x_i})$, is achieved by the following conditional mean [24]:

$$f(X = \mathbf{x}_i) = \int_Y y\, p(y|X = \mathbf{x}_i)\, dy. \tag{5}$$

Assume that $X_{-j} = X \setminus X_j$. From (Yang *et. al* [25]), the following theorem is stated

Theorem 1:

$$P(y|X_{(j)}) = P(y|X_{-j}) \tag{6}$$

where $X_{(j)}$ is defined as

$$X_{(j)} = \{X_1, ..., X_j^*, ..., X_d\}$$

with X_j^* being a random permutation of X_j applied across all samples.

By applying Theorem 1 into equation (5), the following relation holds

$$\begin{aligned} f_{-j}(X) &= \int y P(y|X_{-j})\, dy \\ &= \int y P(y|X_{(j)})\, dy. \end{aligned} \tag{7}$$

Eq. (7) means that permuting a variable j in the trained model, is similar to having the output of the same model without the variable j. This brings several advantages, since instead of completely removing a feature and retraining the model, a random permutation is applied to mimic the output of the model without the respective variable, while the other features are kept unchanged.

The rank of a feature j is then defined by the absolute normalized difference between the prediction error of the model trained with the perturbed set $X_{(j)}$ and with the 'intact' set X:

$$J(X_j) = \frac{|E_{(j)} - E|}{E}, \tag{8}$$

where $E_{(j)}$ and E are performance metrics computed from the predictions using $X_{(j)}$ and X as input set, respectively. In this work the mean square error (RMSE) is used as the performance metric.

In order to select the most relevant features, a recursive feature elimination (RFE) approach was applied to delete the most irrelevant features. First, the model is trained using all features available and the rank of each individual feature is then obtained as stated above. Then, the least important feature is progressively eliminated and the performance of the estimation is computed. To speed up the process, a threshold ϵ is defined such that features j with $J(X_j) < \epsilon$ will be considered with no importance, being immediately discarded.

V. Polymerization Process

This section describes the industrial process used in the experiments. This is a batch process from the polymerization industry. The process under study is the production of a polyester resin, there are two quality variables measured along the process (hard-to-measure variables): N_A (acidity number) and μ (viscosity). The plant is equipped with several online sensors, connected to a process computer that records their measurements every 30 [s]. In total, there are 34 variables (features) being recorded at a fairly high frequency (when compared to the total batch duration, which ranges between 40 and 70 h). Each batch then contains 4500-7500 recordings of each process measurement. Quality variables (N_A and μ) are not measured online. Instead, product samples are taken manually, quite infrequently and unevenly (one sample per $1.2 - 2$ [h]), and are only collected about $8 - 10$ [h] after the batch starts. In this way, only a few quality indicators are available for each batch ($15 - 20$ measurements).

The assembled dataset contains process and quality measurements from 33 batches, equating 16 months of operation.

Two subsets were created: 27 batches constitute the training dataset, while the other 6 batches represent the testing dataset. In total, 200980 samples were collected, where only 663 of the samples have labels (quality indicators). This challenging real-world setting allows semi-supervised algorithms to exploit the huge number of unlabeled examples available. More details about the process can be found at (Facco *et al.* [26]).

VI. Experimental Results

The training dataset (containing data from 27 batches) was used in the experiments to train the model, while the testing dataset was used to compute estimation accuracy. The training dataset has 530 labeled examples and 162564 unlabeled examples. In our testing, all the labeled examples were used as well as 10000 unlabeled ones. Three performance metrics where used: root-mean-square error, RMSE $= \sqrt{\text{MSE}} = \sqrt{\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2}$, its normalized variant, NRMSE $=$ RMSE$/(y_{\max} - y_{\min})$, and the R-square (R^2) coefficient of determination, where $y_{\max}$ and $y_{\min}$ are the maximum and minimum observed output values, respectively. The RMSE is also known as the standard error of the regression, allowing the quantification of the predictive error. The closer the value is to 0, the more useful is the fit for prediction. The NRMSE is similar to the RMSE, facilitating the comparison between different types of variables (e.g. viscosity and acidity). Finally, R^2 measures how successful the fit is in explaining the variation of the data. The formulation used is [27]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2},$$

where $\overline{y}$ is the average observed value. Higher values of R^2 (closer to 1) translate to a better explanation of the variation in the data by the fit.

A Gaussian kernel was used (for each view),

$$k_v(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma_v)$$

with

$$\sigma_v = \frac{1}{n^2}\sum_{i,j=1}^{n}\|\mathbf{x}_i - \mathbf{x}_j\|^2,$$

and

$$\nu_v = \left(\sum_{i=1}^{n}\|\mathbf{x}_i\|/n\right)^{-1}$$

as a regularization parameter (σ_v and ν_v were computed from the labeled examples of each view). Two views were used in all experiments, in these the features were split sequentially among them, creating two disjoint sets. The optimal λ was found by considering only the training set, using cross-validation across 25 trials, being the one which guaranteed the smallest RMSE.

To evaluate the proposed approach, the following methodology was considered. For all datasets, the coRLSR model was trained with all variables, then the variables were ranked according to Eq. (8), on the training set using RMSE as the performance metric, and using a threshold of $\epsilon = 10^{-2}$ to discard non-relevant features.

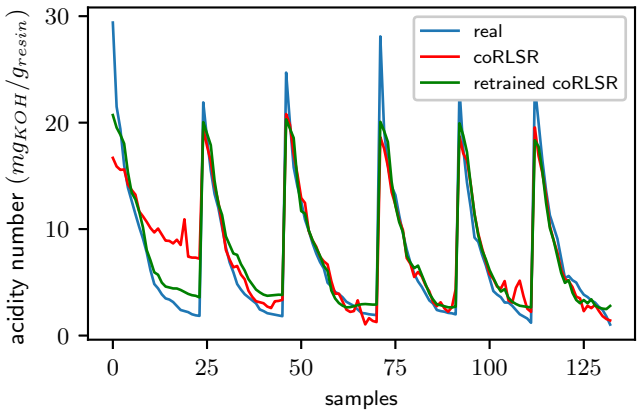


Figure 1: Estimation accuracy of coRLSR and re-trained coRLSR - acidity number.

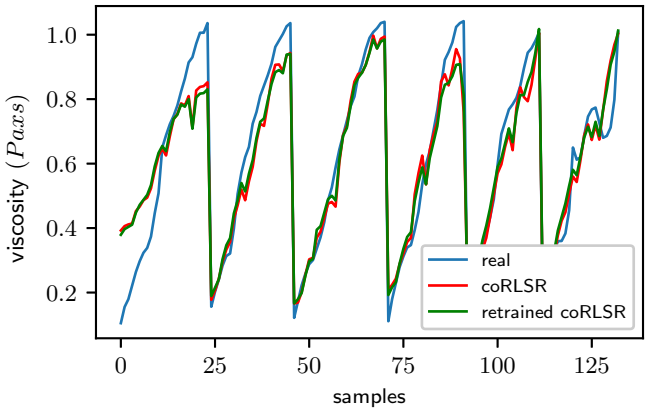


Figure 2: Estimation accuracy of coRLSR and re-trained coRLSR - viscosity.

Table I: Performance metrics - without feature ranking vs. with feature ranking

Variable	Metric	Without	With
acidity	RMSE	2.675	**1.727**
	NRMSE	0.137	**0.095**
	R^2	0.735	**0.903**
viscosity	RMSE	0.940	**0.915**
	NRMSE	0.112	**0.107**
	R^2	0.849	**0.851**

The first stage of estimation, without feature ranking, provided the estimations that can be seen in Fig. 1 for acidity number (N_A) and in Fig. 2 for viscosity (μ). The results have shown a good overall performance, except in the first batch, represented by the first section of the curve in both figures. The actual performance metrics values are shown in Table I.

In both cases (acidity and viscosity estimation) the most relevant features were the 21$^{\text{st}}$, the 9$^{\text{th}}$ and the 13$^{\text{th}}$ as indicated in Fig. 3, and Fig. 4 which shows the ranking given by Eq. (8), with respect to the feature; the labels for the other features were removed from the chart for the sake of clarity. The random permutation of the other features did not impact the model's predictive performance nearly as much. The defined threshold deemed irrelevant 14 features in the

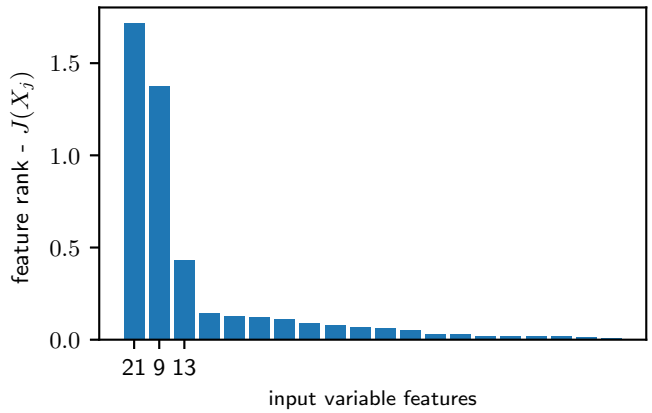


Figure 3: Feature ranking computed on the training set - acidity number.

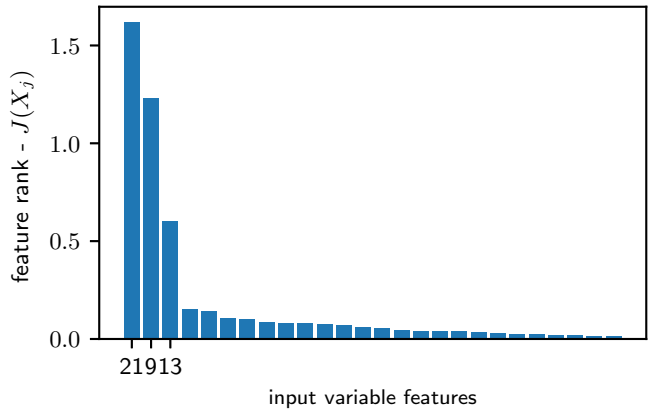


Figure 4: Feature ranking computed on the training set - viscosity.

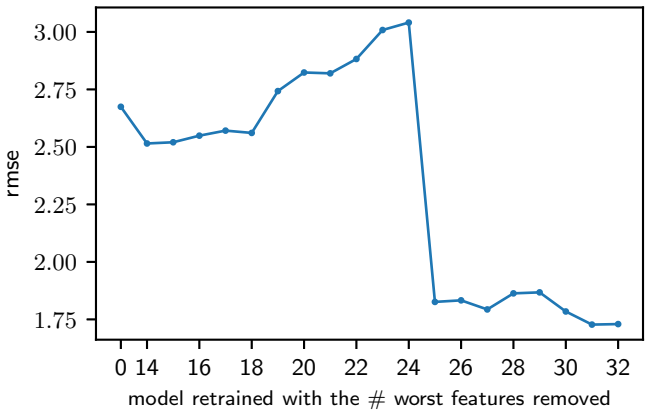


Figure 5: Effect of feature removal on the RMSE - acidity number.

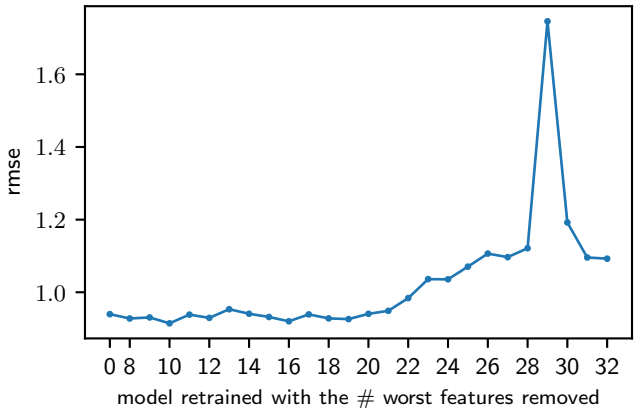


Figure 6: Effect of feature removal on the RMSE - viscosity.

acidity estimation and 8 features on the viscosity side, from which it was concluded that the features had generally more relevance on the viscosity estimation. The feature removal process was performed in order to reach the minimum RMSE. In the acidity number, the minimum was achieved when the model was retrained with the 31 worst (least relevant) features removed (being used only the 3 most relevant). Regarding viscosity, the minimum was reached with the removal of the 10 worst features (being used the 24 most relevant), confirming the higher relevance of the individual features in this case. The model performance while removing the features are demonstrated in Figs. 5 and 6. The overall improvement can be seen by analyzing the values on Table I. The use of feature ranking lowered the RMSE and normalized RMSE (NRMSE) in both the acidity and viscosity cases, providing a higher R^2, improving the curve fitting. This improvement was more noticeable on the acidity number prediction (Fig. 1) versus the viscosity prediction (Fig. 2).

VII. Conclusion

In this paper, the CoRLSR was used as a semi-supervised soft sensor model, and was applied in a real case polymerization process. Moreover, a simple feature ranking method was proposed to deal with the presence of irrelevant features during the modeling. Empirical results on the batch process under study showed that there is an improvement in prediction accuracy, as the method is able to not only use the extra data points from the unlabeled examples to produce good results but also to benefit from a computationally inexpensive strategy to rank and select features to further improve the results. The proposed method for feature ranking helped in discarding irrelevant features, therefore reducing the high-dimensionality of the data commonly found in industrial settings. Following on that, the capabilities of the feature ranking method can be applied to several other methods, as the general underlying concept has low associated computational cost when compared to most of the alternatives. Furthermore, it can provide crucial knowledge to construct views in multiple-view methods like coRLSR.

Acknowledgment

The authors acknowledge the support of FCT project UID-EEA-00048-2013. Francisco A. A. Souza has been supported by Fundação para a Ciência e a Tecnologia (FCT) under grant SFRH/BPD/112774/2015.

REFERENCES

[1] F. A. A. Souza, R. Araújo, and J. Mendes, "Review of soft sensors methods for regression applications," *Chemometrics and Intelligent Laboratory Systems*, vol. 152, pp. 69–79, March 2016.

[2] P. Kadlec and B. Gabrys, "Local learning-based adaptive soft sensor for catalyst activation prediction," *AIChE Journal*, vol. 57, no. 5, pp. 1288–1301, May 2011.

[3] N. Lu, Y. Yang, F. Gao, and F. Wang, "Multirate dynamic inferential modeling for multivariable processes," *Chemical Engineering Science*, vol. 59, no. 4, pp. 855–864, February 2004.

[4] Y. Wu and X. Luo, "A novel calibration approach of soft sensor based on multirate data fusion technology," *Journal of Process Control*, vol. 20, no. 10, pp. 1252–1260, December 2010.

[5] L. Xie, H. Yang, and B. Huang, "Fir model identification of multirate processes with random delays using em algorithm," *AIChE Journal*, vol. 59, no. 11, pp. 4124–4132, November 2013.

[6] Z. Ge and Z. Song, "Semisupervised bayesian method for soft sensor modeling with unlabeled data samples," *AIChE Journal*, vol. 57, no. 8, pp. 2109–2119, August 2011.

[7] Z. Ge, B. Huang, and Z. Song, "Mixture semisupervised principal component regression model and soft sensor application," *AIChE Journal*, vol. 60, no. 2, pp. 533–545, February 2014.

[8] Y. Zhang, Y. Fan, and W. Du, "Nonlinear process monitoring using regression and reconstruction method," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1343–1354, July 2016.

[9] X. Yuan, Z. Ge, Z. Song, Y. Wang, C. Yang, and H. Zhang, "Soft sensor modeling of nonlinear industrial processes based on weighted probabilistic projection regression," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 4, pp. 837–845, April 2017.

[10] A. Rani, V. Singh, and J. Gupta, "Development of soft sensor for neural network based control of distillation column," *ISA Transactions*, vol. 52, no. 3, pp. 438–449, May 2013.

[11] C. Shang, X. Gao, F. Yang, and D. Huang, "Novel bayesian framework for dynamic soft sensor based on support vector machine with finite impulse response," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1550–1557, July 2014.

[12] L. Zhou, Z. Song, J. Chen, Z. Ge, and Z. Li, "Process-quality monitoring using semi-supervised probability latent variable regression models," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8272–8277, 2014.

[13] J. Zhu, Z. Ge, and Z. Song, "Robust semi-supervised mixture probabilistic principal component regression model development and application to soft sensors," *Journal of Process Control*, vol. 32, pp. 25–37, August 2015.

[14] E. Schmitt, V. Bombardier, and L. Wendling, "Improving fuzzy rule classifier by extracting suitable features from capacities with respect to the choquet integral," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 5, pp. 1195–1206, October 2008.

[15] D. Wang, J. Liu, and R. Srinivasan, "Data-driven soft sensor approach for quality prediction in a refining process," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 1, pp. 11–17, February 2010.

[16] T. Lee and C. O. Kim, "Statistical comparison of fault detection models for semiconductor manufacturing processes," *IEEE Transactions on Semiconductor Manufacturing*, vol. 28, no. 1, pp. 80–91, February 2015.

[17] C.-T. Lin, N. R. Pal, S.-L. Wu, Y.-T. Liu, and Y.-Y. Lin, "An interval type-2 neural fuzzy system for online system identification and feature elimination," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 7, pp. 1442–1455, July 2015.

[18] J. Li, C. Duan, and Z. Fei, "A novel variable selection approach for redundant information elimination purpose of process control," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 3, pp. 1737–1744, March 2016.

[19] L. Bravi, V. Piccialli, and M. Sciandrone, "An optimization-based method for feature ranking in nonlinear regression problems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 4, pp. 1005–1010, April 2017.

[20] U. Brefeld, T. Gärtner, T. Scheffer, and S. Wrobel, "Efficient co-regularised least squares regression," in *Proceedings of the 23rd International Conference on Machine Learning*. ACM, 2006, pp. 137–144.

[21] G. Wahba, *Spline Models for Observational Data*. SIAM, 1990.

[22] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *International Conference on Computational Learning Theory*. Springer, 2001, pp. 416–426.

[23] C. Saunders, A. Gammerman, and V. Vovk, "Ridge regression learning algorithm in dual variables." in *ICML*, vol. 98, 1998, pp. 515–521.

[24] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006, ch. 1.5.5.

[25] J.-B. Yang, K.-Q. Shen, C.-J. Ong, and X.-P. Li, "Feature selection for mlp neural network: The use of random permutation of probabilistic outputs," *IEEE Transactions on Neural Networks*, vol. 20, no. 12, pp. 1911–1922, October 2009.

[26] P. Facco, F. Doplicher, F. Bezzo, and M. Barolo, "Moving average pls soft sensor for online product quality estimation in an industrial batch polymerization process," *Journal of Process Control*, vol. 19, no. 3, pp. 520–529, March 2009.

[27] R. Jin, W. Chen, and T. W. Simpson, "Comparative studies of metamodelling techniques under multiple modelling criteria," *Structural and Multidisciplinary Optimization*, vol. 23, no. 1, pp. 1–13, December 2001.

Bibliography

[Agrawala, 1970] A. Agrawala. Learning with a Probabilistic teacher. *IEEE Transactions on Information Theory*, vol. 16, no. 4, pp. 373–379, 1970. (Cited on page 6).

[Bishop, 1995] Christopher M Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995. (Cited on page 14).

[Bishop, 2006] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*, chap. 1.5.5. Springer, 2006. (2 citations in pages 5, and 28).

[Blum and Mitchell, 1998] Avrim Blum and Tom Mitchell. Combining Labeled and Unlabeled Data with Co-training. In: *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pp. 92–100. ACM, 1998. (3 citations in pages 6, 7, and 19).

[Bravi *et al.*, 2017] Luca Bravi, Veronica Piccialli, and Marco Sciandrone. An Optimization-Based Method for Feature Ranking in Nonlinear Regression Problems. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 4, pp. 1005–1010, April 2017. (Cited on page 15).

[Brefeld and Scheffer, 2004] Ulf Brefeld and Tobias Scheffer. Co-EM Support Vector Learning. In: *Proceedings of the 21st International Conference on Machine Learning*, p. 16. ACM, 2004. (Cited on page 7).

[Brefeld *et al.*, 2006] Ulf Brefeld, Thomas Gärtner, Tobias Scheffer, and Stefan Wrobel. Efficient Co-Regularised Least Squares Regression. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 137–144. ACM, 2006. (4 citations in pages 6, 19, 21, and 22).

[Chapelle *et al.*, 2010] Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 1st ed., 2010. ISBN 0262514125, 9780262514125. (2 citations in pages 5, and 6).

[Dempster *et al.*, 1977] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (methodological)*, pp. 1–38, 1977. (Cited on page 7).

[Facco *et al.*, 2009] Pierantonio Facco, Franco Doplicher, Fabrizio Bezzo, and Massimiliano Barolo. Moving Average PLS Soft Sensor for Online Product Quality Estimation in an Industrial Batch Polymerization Process. *Journal of Process Control*, vol. 19, no. 3, pp. 520–529, March 2009. (3 citations in pages 8, 9, and 32).

[Ferreira *et al.*, 2017] Vasco Ferreira, Francisco A. A. Souza, and Rui Araújo. Semi-Supervised Soft Sensor and Feature Ranking Based on Co-Regularised Least Squares Regression Applied to a Polymerization Batch Process. In: *Proc. IEEE 15th International Conference on Industrial Informatics (INDIN 2017)*, pp. 257–262. IEEE, Emden, Germany, July 24-26 2017. (4 citations in pages 3, 19, 27, and 53).

[Fortuna *et al.*, 2006] Luigi Fortuna, Salvatore Graziani, Alessandro Rizzo, and Maria Gabriella Xibilia. *Soft Sensors for Monitoring and Control of Industrial Processes (Advances in Industrial Control)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 1846284791. (2 citations in pages 7, and 11).

[Fralick, 1967] S. Fralick. Learning to Recognize Patterns without a Teacher. *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 57–64, 1967. (Cited on page 6).

[Ge and Song, 2011] Zhiqiang Ge and Zhihuan Song. Semisupervised Bayesian Method for Soft Sensor Modeling with Unlabeled Data Samples. *AIChE Journal*, vol. 57, no. 8, pp. 2109–2119, August 2011. (Cited on page 15).

[Ge *et al.*, 2014] Zhiqiang Ge, Biao Huang, and Zhihuan Song. Mixture Semisupervised Principal Component Regression Model and Soft Sensor Application. *AIChE Journal*, vol. 60, no. 2, pp. 533–545, February 2014. (Cited on page 15).

[Grbić *et al.*, 2013] Ratko Grbić, Dražen Slišković, and Petr Kadlec. Adaptive Soft Sensor for Online Prediction and Process Monitoring Based on a Mixture of Gaussian Process Models. *Computers & chemical engineering*, vol. 58, pp. 84–97, November 2013. (2 citations in pages 10, and 16).

[Huang, 2008] Biao Huang. Bayesian Methods for Control Loop Monitoring and Diagnosis. *Journal of process control*, vol. 18, no. 9, pp. 829–838, October 2008. (Cited on page 10).

[Jin *et al.*, 2014] Huaiping Jin, Xiangguang Chen, Jianwen Yang, and Lei Wu. Adaptive Soft Sensor Modeling Framework Based on Just-in-time Learning and Kernel Partial Least squares Regression for Nonlinear Multiphase Batch Processes. *Computers & Chemical Engineering*, vol. 71, pp. 77–93, December 2014. (2 citations in pages 7, and 9).

[Jin *et al.*, 2001] Ruichen Jin, Wei Chen, and Timothy W Simpson. Comparative Studies of Metamodelling Techniques Under Multiple Modelling Criteria. *Structural and Multidisciplinary Optimization*, vol. 23, no. 1, pp. 1–13, December 2001. (Cited on page 33).

[Jolliffe, 2002] Ian T Jolliffe. *Principal Component Analysis*. Springer, 2 ed., 2002. (Cited on page 14).

[Kadlec and Gabrys, 2011] Petr Kadlec and Bogdan Gabrys. Local Learning-Based Adaptive Soft Sensor for Catalyst Activation Prediction. *AIChE Journal*, vol. 57, no. 5, pp. 1288–1301, May 2011. (Cited on page 12).

[Kadlec *et al.*, 2009] Petr Kadlec, Bogdan Gabrys, and Sibylle Strandt. Data-driven Soft Sensors in the Process Industry. *Computers & Chemical Engineering*, vol. 33, no. 4, pp. 795–814, April 2009. (3 citations in pages 8, 11, and 15).

[Lampropoulos and Tsihrintzis, 2015] Aristomenis S. Lampropoulos and George A. Tsihrintzis. *Machine Learning Paradigms: Applications in Recommender Systems*. Springer Publishing Company, Incorporated, 2015. ISBN 3319191349, 9783319191348. (Cited on page 5).

[Lee and Kim, 2015] Taehyung Lee and Chang Ouk Kim. Statistical Comparison of Fault Detection Models for Semiconductor Manufacturing Processes. *IEEE Transactions on Semiconductor Manufacturing*, vol. 28, no. 1, pp. 80–91, February 2015. (4 citations in pages 7, 14, 15, and 29).

[Li *et al.*, 2016] Jiachen Li, Chaojing Duan, and Zhongyang Fei. A Novel Variable Selection Approach for Redundant Information Elimination Purpose of Process Control. *IEEE Transactions on Industrial Electronics*, vol. 63, no. 3, pp. 1737–1744, March 2016. (3 citations in pages 7, 15, and 29).

[Lin *et al.*, 2015] Chin-Teng Lin, Nikhil R Pal, Shang-Lin Wu, Yu-Ting Liu, and Yang-Yin Lin. An Interval Type-2 Neural Fuzzy System for Online System Identification and Feature Elimination. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 7, pp. 1442–1455, July 2015. (Cited on page 15).

[Lu *et al.*, 2004] Ningyun Lu, Yi Yang, Furong Gao, and Fuli Wang. Multirate Dynamic Inferential Modeling for Multivariable Processes. *Chemical Engineering Science*, vol. 59, no. 4, pp. 855–864, February 2004. (Cited on page 12).

[Ma *et al.*, 2009] Ming-Da Ma, Jing-Wei Ko, San-Jang Wang, Ming-Feng Wu, Shi-Shang Jang, Shyan-Shu Shieh, and David Shan-Hill Wong. Development of Adaptive Soft Sensor Based on Statistical Identification of Key Variables. *Control Engineering Practice*, vol. 17, no. 9, pp. 1026–1034, September 2009. (Cited on page 16).

[Nelles, 2013] Oliver Nelles. *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Springer Science & Business Media, 2013. (Cited on page 14).

[Nigam *et al.*, 1998] Kamal Nigam, Andrew McCallum, Sebastian Thrun, Tom Mitchell, *et al.* Learning to Classify Text from Labeled and Unlabeled Documents. *AAAI/IAAI*, vol. 792, 1998. (Cited on page 6).

[Rani *et al.*, 2013] Asha Rani, Vijander Singh, and JRP Gupta. Development of Soft Sensor for Neural Network Based Control of Distillation Column. *ISA Transactions*, vol. 52, no. 3, pp. 438–449, May 2013. (2 citations in pages 9, and 15).

[Saunders *et al.*, 1998] Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge Regression Learning Algorithm in Dual Variables. In: *ICML*, vol. 98, pp. 515–521. 1998. (Cited on page 20).

[Schmitt *et al.*, 2008] Emmanuel Schmitt, Vincent Bombardier, and Laurent Wendling. Improving Fuzzy Rule Classifier by Extracting Suitable Features From Capacities With Respect to the Choquet Integral. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 5, pp. 1195–1206, October 2008. (2 citations in pages 14, and 15).

[Schölkopf *et al.*, 2001] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A Generalized Representer Theorem. In: *International Conference on Computational Learning Theory*, pp. 416–426. Springer, 2001. (Cited on page 20).

[Scudder, 1965] H. Scudder. Probability of Error of Some Adaptive Pattern-Recognition Machines. *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965. (Cited on page 6).

[Shang *et al.*, 2014] Chao Shang, Xinqing Gao, Fan Yang, and Dexian Huang. Novel Bayesian Framework for Dynamic Soft Sensor Based on Support Vector Machine with Finite Impulse Response. *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1550–1557, July 2014. (2 citations in pages 9, and 15).

[Sindhwani *et al.*, 2005] Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. A Co-Regularization Approach to Semi-Supervised Learning with Multiple Views. In: *Proceedings of ICML Workshop on Learning with Multiple Views*, pp. 74–79. 2005. (2 citations in pages 6, and 7).

[Souza *et al.*, 2016] Francisco A. A. Souza, Rui Araújo, and Jérôme Mendes. Review of Soft Sensors Methods for Regression Applications. *Chemometrics and Intelligent Laboratory Systems*, vol. 152, pp. 69–79, March 2016. (Cited on page 12).

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press Cambridge, 1998. (Cited on page 6).

[Suykens *et al.*, 2002] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002. (Cited on page 34).

[Suykens *et al.*, 2011] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. Least Squares - Support Vector Machines - Matlab/C Toolbox. http://www.esat.kuleuven.be/sista/lssvmlab/, 2011. (Cited on page 34).

[Vapnik and Kotz, 1982] Vladimir Naumovich Vapnik and Samuel Kotz. *Estimation of Dependences Based on Empirical Data*, vol. 40. Springer-Verlag New York, 1982. (Cited on page 6).

[Wahba, 1990] Grace Wahba. *Spline Models for Observational Data*. SIAM, 1990. (Cited on page 20).

[Wang *et al.*, 2010] David Wang, Jun Liu, and Rajagopalan Srinivasan. Data-Driven Soft Sensor Approach for Quality Prediction in a Refining Process. *IEEE Transactions on Industrial Informatics*, vol. 6, no. 1, pp. 11–17, February 2010. (3 citations in pages 14, 15, and 29).

[Xiong *et al.*, 2017] Weili Xiong, Yanjun Li, Yujia Zhao, and Biao Huang. Adaptive Soft Sensor Based on Time Difference Gaussian Process Regression with Local Time-Delay Reconstruction. *Chemical Engineering Research and Design*, vol. 117, pp. 670–680, January 2017. (Cited on page 17).

[Yang *et al.*, 2009] Jian-Bo Yang, Kai-Quan Shen, Chong-Jin Ong, and Xiao-Ping Li. Feature Selection for MLP Neural Network: The Use of Random Permutation of Probabilistic Outputs. *IEEE Transactions on Neural Networks*, vol. 20, no. 12, pp. 1911–1922, October 2009. (Cited on page 28).

[Yuan *et al.*, 2017a] Xiaofeng Yuan, Zhiqiang Ge, Biao Huang, Zhihuan Song, and Yalin Wang. Semisupervised JITL Framework for Nonlinear Industrial Soft Sensing Based on Locally Semisupervised Weighted PCR. *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 532–541, April 2017. (Cited on page 9).

[Yuan *et al.*, 2017b] Xiaofeng Yuan, Zhiqiang Ge, Zhihuan Song, Yalin Wang, Chunhua Yang, and Hongwei Zhang. Soft Sensor Modeling of Nonlinear Industrial Processes Based on Weighted Probabilistic Projection Regression. *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 4, pp. 837–845, April 2017. (2 citations in pages 7, and 9).

[Zhang *et al.*, 2016] Yingwei Zhang, Yunpeng Fan, and Wenyou Du. Nonlinear Process Monitoring Using Regression and Reconstruction Method. *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1343–1354, July 2016. (2 citations in pages 10, and 15).

[Zhou *et al.*, 2014] Le Zhou, Zhihuan Song, Junghui Chen, Zhiqiang Ge, and Zhao Li. Process-Quality Monitoring using Semi-Supervised Probability Latent Variable Regression Models. *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8272–8277, 2014. (Cited on page 15).

[Zhou and Li, 2005] Zhi-Hua Zhou and Ming Li. Semi-Supervised Regression with Co-Training. In: *IJCAI*, vol. 5, pp. 908–913. 2005. (2 citations in pages 6, and 7).

[Zhu *et al.*, 2015] Jinlin Zhu, Zhiqiang Ge, and Zhihuan Song. Robust Semi-Supervised Mixture Probabilistic Principal Component Regression Model Development and Application to Soft Sensors. *Journal of Process Control*, vol. 32, pp. 25–37, August 2015. (Cited on page 15).