Artificial intelligence models for large scale buildings energy consumption analysis

Haixiang Zhao

ÉCOLE CENTRALE PARIS
ET MANUFACTURES
⟨⟨ ÉCOLE CENTRALE PARIS⟩⟩

THÈSE

présentée par

Hai-xiang ZHAO

pour l'obtention du

GRADE DE DOCTEUR

Spécialité : Informatique

Laboratoire d'accueil : Laboratoire mathématiques appliquées aux systèmes

Sujet : Artificial Intelligence Models for Building Energy Consumption Analysis

Soutenue le : 28 September 2011

devant un jury composé de :

Prof. Qingping Guo	Rapporteur
Prof. David Elizondo	Rapporteur
Prof. Choi-Hong Lai	Examinateur
Mr. Fred Lherminier	Co-encadrant
Prof. Frédéric Magoulès	Directeur de thèse

2011ECAP0036

Abstract

The energy performance in buildings is influenced by many factors, such as ambient weather conditions, building structure and characteristics, occupancy and their behaviors, the operation of sub-level components like Heating, Ventilation and Air-Conditioning (HVAC) system. This complex property makes the prediction, analysis, or fault detection/diagnosis of building energy consumption very difficult to accurately and quickly perform. This thesis mainly focuses on up-to-date artificial intelligence models to solve these problems.

In this thesis, recently developed models for solving these problems, including detailed and simplified engineering methods, statistical methods and artificial intelligence methods are reviewed. Then energy consumption profiles are simulated for single and multiple buildings, and based on these datasets, support vector machine models are trained and tested to do the prediction. The results from extensive experiments demonstrate high prediction accuracy and robustness of these models.

Then, Recursive Deterministic Perceptron (RDP) neural network model is used to detect and diagnose faulty building energy consumption. In the experiment, RDP model shows very high detection ability. A new approach is proposed to diagnose faults. It is based on the evaluation of RDP models, each of which is able to detect an equipment fault. How to select subsets of features influences the model performance. The optimal features are here selected based on the feasibility of obtaining them and on the scores they provide under the evaluation of two filter methods. Experimental results confirm the validity of the selected subset and show that the proposed feature selection method can guarantee the model accuracy and reduces the computational time.

Finally, one of the most difficult challenges of predicting building energy consumption is to accelerate model training when the dataset is very large. This thesis proposes an efficient parallel implementation of support vector machines based on decomposition method for solving such problems. The parallelization is performed on the most time-consuming work of training, i.e., to update the gradient vector f. The inner problems are dealt by sequential minimal optimization solver. The underlying parallelism is conducted by the shared memory version of Map-Reduce paradigm, making the

system particularly suitable to be applied to multi-core and multiprocessor systems. Experimental results show that our implementation offers a high speed increase compared to Libsvm, and it is superior to the state-of-the-art MPI implementation Pisvm in both speed and storage requirement.

Keywords: Energy efficiency, Building, Support Vector Machines, Recursive Deterministic Perceptron, Prediction, Fault detection and diagnosis, Feature selection, Parallel computing, Map-Reduce Paradigm

Acknowledgements

In the first place, I would like to thank my supervisor Prof. Frédéric Magoulès. During the past three years, he offered me a great help in my study, work, and even in everyday's life. He proposed this subject in the beginning, then he trusted on me, patiently supported and pushed me on the way to do the research. He trained me not only research ability and but also professionalism. It is fruitful and enjoyable to discuss with him every time. He always gave me valuable ideas and suggestions. When I was writing this thesis, he also gave me a lot of help. I gratefully thank to him for everything he did.

Many thanks to our partner, Terra Nova company, especially Mr. Fred Lherminier and Mrs. Florence Lai who are working there. They gave me very useful information and guidance in every meeting and discussion. They brought to me constructive ideas and suggestions, shared with me their professional experience, solid knowledge and keen insight for the industrial development. This thesis benefits a lot from them.

I can not wait to express my thanks to my colleges and friends, Lei Yu, Jie Pan, Fei Teng, Céderic Venet, Florent Pruvost, Thomas Cadeau, Thu Huyen Dao, Somanchi K Murthy, Sana Chaabane, Alain Rueyahana, Sidharth GS, Abel-Kassim Cheik Ahamed and many other students in Ecole Centrale Paris. They are so kind, helpful and very pleasant to be with. It is my great honor and pleasure to stay together with them for three years. I want to give them my great thanks for everything they have done, creating a lively, enjoyable and harmony working environment, sharing me knowledge and information, encouraging me when I was in the low point, offering me a hand when I need help, etc.

Many thanks to my thesis reviewers and the defense jury committee, Prof. Qingping Guo, Prof. David Elizendo and Prof. Choi-Hong Lai. They offered me a lot of valuable guidance and suggestions on my thesis.

Finally, I would like to give my deepest thanks to my parents, my wife, my sisters and all other members in my family, This work can not be finished without their continuous support and encouragement.

Contents

List of Figures

LIST OF FIGURES

List of Tables

LIST OF TABLES

1

Introduction

1.1 Overview and historical motivation

In Europe, buildings account for 40% of total energy use and 36% of total CO_2 emission [1]. We take France as an example, Figure 1.1 shows the annual energy consumption of each sector from 1990 to 2009. The part of industry decreased from 30% to 25%, that of transport was stable around 30%. However the usage of residential tertiary increased from 37% to 41%. We can see an increasing ratio of the building energy consumption during these years. Moreover, we can expect that the ratio will continue to increase in the future. The prediction of energy use in buildings is therefore significant for improving energy performance of buildings, leading to energy conservation and reducing environmental impact.

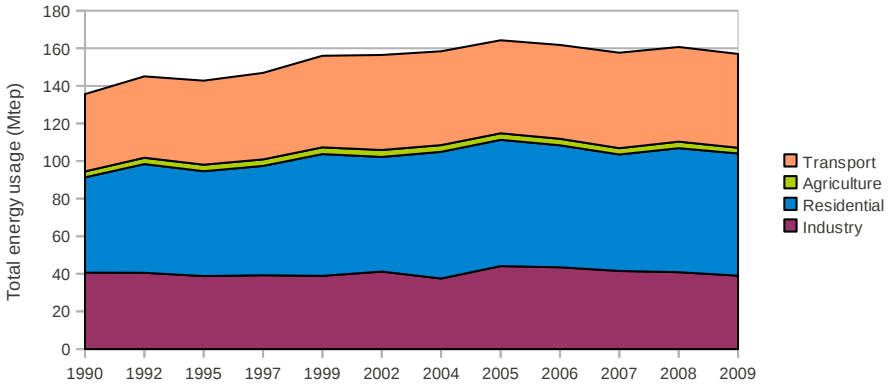

Figure 1.1: Annual energy consumption in each sector of France. (Source: [2])

However, the energy system in buildings is quite complex, as the energy types and building types vary greatly. In literatures, the main energy forms considered are heating/cooling loads, hot water and electricity consumption. The most frequently considered building types are office, residential and engineering buildings, varying from small rooms to big estates. The energy behavior of a building is influenced by many factors, such as weather conditions, especially the dry-bulb temperature, the building construction and thermal property of the physical materials used, the occupancy and their behavior, sub-level components such as Heating, Ventilating, and Air-Conditioning (HVAC), lighting systems, their performance and schedules.

Due to the complexity of the problem, precise consumption prediction is quite difficult. In recent years, a large number of approaches for this purpose, either elaborate or simplified, have been proposed and applied to a broad range of problems. This research work has been carried out in the process of designing new buildings, operation or retrofit of contemporary buildings, varying from building's sub-system analysis to regional or national level modeling. Predictions can be performed on the whole building or sub-level components by thoroughly analyzing each influencing factor or approximating the usage by considering several major factors. An effective and efficient model has always been the goal of the research community.

1.2 The prediction methods

This section reviews the recent work related to the modeling and prediction of building energy consumption. The methods used in this application include engineering, statistical and artificial intelligence methods. The most widely used artificial intelligence methods are Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs). In 2003 and 2010, Krarti and Dounis respectively provided two overviews of artificial intelligence methods in the application of building energy systems [3, 4]. Our work especially focuses on the prediction applications. To even further enrich the content and provide the readers with a complete view of various prediction approaches, this section also reviews engineering and statistical methods. Moreover, there are also some hybrid approaches which combine some of the above models to optimize predictive performance, such as [5–8]. In this section, we globally describe the problems, models, related problems such as data pre/post-processing, and the comparison of these models.

1.2.1 Engineering methods

The engineering methods use physical principles to calculate thermal dynamics and energy behavior on the whole building level or for sub-level components. They have been adequately developed over the past fifty years. These methods can be roughly classified into two categories, the detailed comprehensive method and the simplified method. The comprehensive methods use very elaborate physical functions or thermal dynamics to calculate precisely, step-by-step, the energy consumption for all components of the building with building's and environmental information, such as external climate conditions, building construction, operation, utility rate schedule and HVAC equipment, as the inputs. In this section, we concentrate on the global view of models and applications, while the details of these computational processes are far beyond the purpose of this review. Readers may refer to [9] for calculation details. For HVAC systems, in particular, the detailed energy calculation is introduced in [10]. The ISO has developed a standard for the calculation of energy use for space heating and cooling for a building and its components [11].

Hundreds of software tools have been developed for evaluating energy efficiency, renewable energy, and sustainability in buildings, such as DOE-2, EnergyPlus, BLAST, ESP-r [12]. Some of them have been widely used for developing building energy standards and analyzing energy consumption and conservation measures of buildings. Surveys of these tools are performed in [13, 14]. For readers' information, the U.S. Department of Energy (DOE) maintains a list of almost all the simulation tools [12], which is constantly updated.

Although, these elaborate simulation tools are effective and accurate, in practice, there are some difficulties. Since these tools are based on physical principles, to achieve an accurate simulation, they require details of building and environmental parameters as the input data. On one hand, these parameters are unavailable to many organizations, for instance, the information on each room in a large building is always difficult to obtain. This lack of precise inputs will lead to a low accurate simulation. On the other hand, operating these tools normally requires tedious expert work, making it difficult to perform and cost inefficient. For these reasons some researchers have proposed simpler models to offer alternatives to certain applications.

Al-Homoud [13] reviewed two simplified methods. One is degree day method in which only one index, degree day, is analyzed. This steady-state method is suitable for estimating small buildings' energy consumption where the envelope-based energy dominates. The other one is bin, also known as temperature frequency method, which can be used to model large buildings where internally generated loads dominate or loads are not linearly dependent on outdoor/indoor temperature difference.

Weather conditions are important factors to determine building energy usage. These take many forms, such as temperature, humidity, solar radiation, wind speed, and vary over time. Certain studies are conducted to simplify weather conditions in building energy calculations. White and Reichmuth [15] attempted to use average monthly temperatures to predict monthly building energy consumption. This prediction is more accurate than standard procedures which normally use heating and cooling degree days or temperature bins. Westphal and Lamberts [16] predicted the annual heating and cooling load of non-residential buildings simply based on some weather variables, including monthly average of maximum and minimum temperatures, atmospheric pressure, cloud cover and relative humidity. Their results showed good accuracy on low mass envelope buildings, compared to elaborate simulation tools such as ESP, BLAST, DOE2, etc.

As well as weather conditions, building characteristic is another important yet complex factor in determining energy performance.

Yao and Steemers [5] developed a simple method of predicting a daily energy consumption profile for the design of a renewable energy system for residential buildings. The total building energy consumption was defined as the summation of several components: appliances, hot water, and space heating. For each component, a specific modeling method was employed. For instance, to model electric appliances, they used the average end-use consumption from large amounts of statistical data. While modeling space heating demand, a simplified physical model was applied. Since the average value varies seasonally, this method predicts energy demand for one season at a time.

By adopting this divide-and-sum concept, Rice et al. [17] simplified each sub-level calculation to explain the system level building energy consumption. In the project "Updating the ASHRAE/ACCA Residential Heating and Cooling Load Calculation Procedures and Data" (RP-1199), Barnaby and Spitler [18] proposed a residential load

factor method, which is a simple method and is tractable by hand. The load contributions from various sources were separately evaluated and then added up. Wang and Xu [6] simplified the physical characteristics of buildings to implement the prediction. For building envelops, the model parameters were determined by using easily available physical details based on the frequency characteristic analysis. For various internal components, they used a thermal network of lumped thermal mass to represent the internal mass. Genetic algorithm was used to identify model parameters based on operation data. Yik et al. [19] used detailed simulation tools to obtain cooling load profiles for different types of buildings. A simple model, which is a combination of these detailed simulation results, was proposed to determine the simultaneous cooling load of a building.

Calibration is another important issue in building energy simulation. By tuning the inputs carefully, it can match the simulated energy behavior precisely with that of a specific building in reality. Pan et al. [20] summarized the calibrated simulation as one building energy analysis method and applied it to analyze the energy usage of a high-rise commercial building. After steps of repeated calibration, this energy model showed high accuracy in predicting the actual energy usage of the specified building. A detailed review of calibration simulation is provided in [21]. Since calibration is a tedious and time-consuming work, we can see that doing accurate simulation by a detailed engineering method is of high complexity.

We note that there is no apparent boundary between the simplified and elaborate models. It is also possible to do simplified simulation with some comprehensive tools, such as EnergyPlus [22]. Suggested by AI-Homoud, if the purpose is to study trends, compare systems or alternatives, then simplified analysis methods might be sufficient. In contrast, for a detailed energy analysis of buildings and sub-systems and life cycle cost analysis, more comprehensive tools will be more appropriate [13].

1.2.2 Statistical methods

Statistical regression models simply correlate the energy consumption or energy index with the influencing variables. These empirical models are developed from historical performance data, which means that before training the models, we need to collect enough historical data. Much research on regression models has been carried out on the following problems. The first is to predict the energy usage over simplified variables

such as one or several weather parameters. The second is to predict some useful energy index. The third one is to estimate important parameters of energy usage, such as total heat loss coefficient, total heat capacity and gain factor, which are useful in analyzing thermal behavior of building or sub-level systems.

In some simplified engineering models, the regression is used to correlate energy consumption with the climatic variables to obtain an energy signature [23–25]. Bauer and Scartezzini [23] proposed a regression method to handle both heating and cooling calculations simultaneously by dealing with internal as well as solar gains. Ansari et al. [26] calculated the cooling load of a building by adding up the cooling load of each component of the building envelope. Each sub-level cooling load is a simple regression function of temperature difference between inside and outside. Dhar et al. [27, 28] modeled heating and cooling load in commercial buildings with outdoor dry-bulb temperature as the only weather variable. A new temperature-based Fourier series model was proposed to represent nonlinear dependence of heating and cooling loads on time and temperature. If humidity and solar data is also available, they suggested using the generalized Fourier series model since it has more engineering relevance and higher prediction ability. Also taking dry-bulb temperature as the single variable for model developing, Lei and Hu [29] evaluated regression models for predicting energy savings from retrofit projects of office buildings in a hot summer and cold winter region. They showed that a single variable linear model is sufficient and practical to model the energy use in hot and cold weather conditions. Ma et al. [30] integrated multiple linear regression and self-regression methods to predict monthly power energy consumption for large scale public buildings. In the work of Cho et al. [31], the regression model was developed on 1-day, 1-week, 3-month measurements, leading to the prediction error in the annual energy consumption of 100%, 30%, 6% respectively. These results show that the length of the measurement period strongly influences the temperature dependent regression models.

Concerning the prediction of energy index, Lam et al. [32] used Principle Component Analysis (PCA) to develop a climatic index Z with regard to global solar radiation, dry- and wet-bulb temperature. They found that Z has the same trend as simulated cooling load, HVAC and building energy use. This trend was obtained from the analysis of correlation by a linear regression analysis. The model was developed based on

the data from 1979 to 2007. Ghiaus [33] developed a robust regression model to correlate the heating loss on the dry-bulb temperature by using the range between the 1^{st} and the 3^{rd} quartile of the quantile-quantile plot which gives the relation of these two variables.

Jiménez and Heras [34] used the Auto-Regressive model with eXtra inputs (ARX) to estimate the U and g values of building components. Kimbara et al. [35] developed an Auto-Regressive Integrated Moving Average (ARIMA) model to implement on-line prediction. The model was first derived on the past load data, and was then used to predict load profiles for the next day. ARIMA with eXternal inputs (ARIMAX) models have also been applied to some applications, predicting and controlling the peak electricity demand for commercial buildings [36] and predicting the power demand of the buildings [37]. In [37], Newsham and Birt put a special emphasis on the influence of occupancy, which can apparently increase the accuracy of the model.

Aydinalp-Koksal and Ugursal [38] suggested considering a regression-based method, called Conditional Demand Analysis (CDA), when we predict national level building energy consumption. In their experimental comparisons, CDA showed accurate predicting ability as good as neural networks and engineering methods, but was easier to develop and use. However, the drawback of the CDA model was lack of detail and flexibility, and it required a large amount of input information. CDA was also employed in the early work for analyzing residential energy consumption [39].

1.2.3 Neural networks

ANNs are the most widely used artificial intelligence models in the application of building energy prediction. This type of model is good at solving non-linear problems and is an effective approach to this complex application. In the past twenty years, researchers have applied ANNs to analyze various types of building energy consumption in a variety of conditions, such as heating/cooling load, electricity consumption, sub-level components operation and optimization, estimation of usage parameters. In this section, we review the previous studies and put them into groups according to the applications dealt with. Additionally, model optimization, such as pre-process of input data and comparisons between ANNs and other models, are highlighted at the end.

In 2006, Kalogirou [40] did a brief review of the ANNs in energy applications in buildings, including solar water heating systems, solar radiation, wind speed, air flow

distribution inside a room, prediction of energy consumption, indoor air temperature, and HVAC system analysis.

Kalogirou et al. [41] used back propagation neural networks to predict the required heating load of buildings. The model was trained on the consumption data of 225 buildings which vary largely from small spaces to big rooms. Ekici and Aksoy [42] used the same model to predict building heating loads in three buildings. The training and testing datasets were calculated by using the finite difference approach of transient state one-dimensional heat conduction. Olofsson et al. [43] predicted the annual heating demand of a number of small single family buildings in the north of Sweden. Later, Olofsson and Andersson [44] developed a neural network which makes long-term energy demand (the annual heating demand) predictions based on short-term (typically 2-5 weeks) measured data with a high prediction rate for single family buildings.

In [45], Yokoyama et al. used a back propagation neural network to predict cooling demand in a building. In their work, a global optimization method called modal trimming method was proposed for identifying model parameters. Kreider et al. [46] reported results of a recurrent neural network on hourly energy consumption data to predict building heating and cooling energy needs in the future, knowing only the weather and time stamp. Based on the same recurrent neural network, Ben-Nakhi and Mahmoud [47] predicted the cooling load of three office buildings. The cooling load data from 1997 to 2000 was used for model training and the data for 2001 was used for model testing. Kalogirou [48] used neural networks for the prediction of the energy consumption of a passive solar building where mechanical and electrical heating devices are not used. Considering the influence of weather on the energy consumption in different regions, Yan and Yao [49] used a back propagation neural network to predict building's heating and cooling load in different climate zones represented by heating degree day and cooling degree day. The neural network was trained with these two energy measurements as parts of input variables.

In the application of building electricity usage prediction, an early study [50] has successfully used neural networks for predicting hourly electricity consumption as well as chilled and hot water for an engineering center building. Nizami and Al-Garni [51] tried a simple feed-forward neural network to relate the electric energy consumption to the number of occupancy and weather data. González and Zamarreño [52] predicted short-term electricity load with a special neural network which feeds back part of its

outputs. In contrast, Azadeh et al. [53] predicted the long-term annual electricity consumption in energy intensive manufacturing industries, and showed that the neural network is very applicable to this problem when energy consumption shows high fluctuation. Wong et al. [54] used a neural network to predict energy consumption for office buildings with day-lighting controls in subtropical climates. The outputs of the model include daily electricity usage for cooling, heating, electric lighting and total building.

ANNs are also used to analyze and optimize sub-level components behavior, mostly for HVAC systems. Hou et al. [55] predicted air-conditioning load in a building, which is a key to the optimal control of the HVAC system. Lee et al. [56] used a general regression neural network to detect and diagnose faults in a building's air-handling unit. Aydinalp et al. [57] showed that the neural network can be used to estimate appliance, lighting and space cooling energy consumption and it is also a good model to estimate the effects of the socio-economic factors on this consumption in the Canadian residential sector. In their follow-up work, neural network models were developed to successfully estimate the space and domestic hot-water heating energy consumptions in the same sector [58].

In [47, 59], general regression neural networks were used for air conditioning set-back controlling, and for optimizing HVAC thermal energy storage in public and office buildings. Yalcintas et al. [60] used neural networks to predict chiller plant energy use of a building in a tropical climate. Later, they used a three-layer feed-forward neural network to predict energy savings in equipment retrofit [61]. Gouda et al. [62] used a multi-layered feed-forward neural network to predict internal temperature with easily measurable inputs which include outdoor temperature, solar irradiance, heating valve position and the building indoor temperature.

Building energy performance parameters can be estimated by neural networks. In [63–66], the authors estimated the total heat loss coefficient, the total heat capacity and the gain factor which are important for a reliable energy demand forecast. The method is based on an analysis of a neural network model that is trained on simple data, the indoor/outdoor temperature difference, the supplied heat and the available free heat. Kreider et al. [46] reported results of recurrent neural networks on hourly energy consumption data. They also reported results on finding the thermal resistance, R, and thermal capacitance, C, for buildings from networks trained on building data. Zmeureanu [67] proposed a method using the general regression neural networks to

evaluate the Coefficient of Performance (COP) of existing rooftop units. Yalcintas presented an ANN-based benchmarking technique for building energy in tropical climates, focused on predicting a weighted energy use index. The selected buildings are in large variety [68, 69].

The input data for the model training can be obtained from on-site measurement, survey, billing collection or simulation. The raw data may have noisy or useless variables, therefore it can be cleaned and reduced before model development. There is much research concerning the data pre-processing technologies. González and Zamarreño [52] predicted short-term electricity load by using two phases of neural networks. The first layer predicts climatic variables, while the second predicts energy usage, which takes the outputs of the first layer as inputs. The same two-phase technology was also used by Yokoyama et al. in predicting cooling load [45]. The trend and periodic change were first removed from data, and then the converted data was used as the main input for the model training. Additional inputs, including air temperature and relative humidity, were considered to use predicted values. Their effects on the prediction of energy demand were also investigated in this work.

Ben-Nakhi and Mahmoud [47] predicted the cooling load profile of the next day, and the model was trained on a single variable, outside dry-bulb temperature. Ekici and Aksoy [42] predicted building heating loads without considering climatic variables. The networks were trained by only three inputs, transparency ratio, building orientation and insulation thickness. Kreider and Haberl [70] predicted the nearest future with the input of nearest past data. For predicting far future, they used recurrent neural networks. Yang et al. [71] used accumulative and sliding window methods to train neural networks for the purpose of on-line building energy prediction. Sliding window constrained input samples in a small range.

Olofsson et al. [43] used PCA to reduce the variable dimension before predicting the annual heating demand. In their later work, they achieved long-term energy demand prediction based on short-term measured data [44]. Kubota et al. [72] used genetic algorithm for the variable extraction and selection on measured data, and then fuzzy neural networks were developed for the building energy load prediction. Here the variable extraction means translating original variables into meaningful information that is used as input in the fuzzy inference system. Hou et al. [55] integrated rough sets theory and a neural network to predict an air-conditioning load. Rough sets theory

was applied to find relevant factors influencing the load, which were used as inputs in a neural network to predict the cooling load. Kusiak et al. [73] predicted daily steam load of buildings by a neural network ensemble with five Multi-Layer Perceptrons (MLPs) methods since in several case studies, it outperforms 9 other data mining algorithms, including CART, CHAID, exhaustive CHAID, boosting tree, MARSplines, random forest, SVM, MLP, and k-NN. A correlation coefficient matrix and the boosting tree algorithm were used for variable selection. Karatasou et al. [7] studied how statistical procedures can improve neural network models in the prediction of hourly energy loads. The statistical methods, such as hypothesis testing, information criteria and cross validation, were applied in both inputs pre-processing and model selection. Experimental results demonstrated that the accuracy of the prediction is comparable to the best results reported in the literature.

The outputs of neural networks may not be exactly what we expected, Kajl et al. proposed a fuzzy logic to correct the outputs by post-processing the results of neural networks. The fuzzy assistant allows the user to determine the impact of several building parameters on the annual and monthly energy consumption [74, 75].

Some comparisons between neural network and other prediction models were performed in the research. Azadeh et al. [53] showed that the neural network was very applicable to the annual electricity consumption prediction in manufacturing industries where energy consumption has high fluctuation. It is superior to the conventional non-linear regression model through ANalysis of VAriance (ANOVA). Aydinalp et al. [57] showed that neural networks can achieve higher prediction performance than engineering models in estimating Appliance, Lighting and space Cooling (ALC) energy consumption and the effects of socio-economic factors on this consumption in the Canadian residential sector. Later, ANN was compared with CDA method in [38]. From this work we see that CDA has as high an ability to solve the same problem as ANN model, while the former is easier to develop and use. Neto [76] compared the elaborate engineering method with neural network model for predicting building energy consumption. Both models have shown high prediction accuracy, while ANN is slightly better than the engineering model in the short-term prediction.

1.2.4 Support vector machines

SVMs are increasingly used in research and industry. They are highly effective models in solving non-linear problems even with small quantities of training data. Many studies of these models were conducted on building energy analysis in the last five years.

Dong et al. [77] first applied SVMs to predict the monthly electricity consumption of four buildings in the tropical region. Three years' data was trained and the derived model was applied to one year's data to predict the landlord utility in that year. The results showed good performances of SVMs on this problem.

Lai et al. [78] applied this model on one year's electricity consumption of a building. The variables include climate variations. In their experiments, the model was derived from one year's performance and then tested on three months' behavior. They also tested the model on each daily basis dataset to verify the stability of this approach during short periods. In addition, they added perturbation manually to a certain part of the historical performance and used this model to detect the perturbation by examining the change of the contributing weights.

Li et al. [79] used SVMs to predict the hourly cooling load of an office building. The performance of the support vector regression is better than the conventional back propagation neural networks. Hou and Lian [80] also used SVMs for predicting cooling load of the HVAC system. The result shows that SVMs are better than the ARIMA model.

Li et al. [81] predicted the annual electricity consumption of buildings by back propagation neural networks, RBF neural networks, general regression neural networks and SVMs. They found that general regression neural networks and SVMs were more applicable to this problem compared to other models. Furthermore, SVM showed the best performance among all prediction models. The models were trained on the data of 59 buildings and tested on 9 buildings.

Liang and Du [8] presented a cost-effective fault detection and diagnosis method for HVAC systems by combining the physical model and a SVM. By using a four-layer SVM classifier, the normal condition and three possible faults can be recognized quickly and accurately with a small number of training samples. Three major faults are recirculation damper stuck, cooling coil fouling/block and supply fan speed decreasing.

The indicators are the supply and mixed air temperatures, the outlet water temperature and the valve control signal.

Certain research was performed for pre- or post-process model training. Lv et al. [82] used PCA to reduce variables before training SVMs for predicting building cooling load. Li et al. [83] used an improved PCA, called Kernel Principal Component Analysis (KPCA), before training SVMs to predict building cooling load. Li et al. [84] used fuzzy C-mean clustering algorithm to cluster the samples according to their degree of similarity. Then they applied a fuzzy membership to each sample to indicate its contribution to the model. In the post-processing, Zhang and Qi [85] applied Markov chains to do further interval forecasting after prediction of building heating load by SVMs.

1.2.5 Grey models

When the information of one system is partially known, we call this system a grey system. The grey model can be used to analyze building energy behavior when there is only incomplete or uncertain data. Very little work has been done regarding this model.

In 1999, Wang et al. [86] applied a grey model to predict building heat moisture system. The predicting accuracy is fairly high. Guo et al. [87] used an improved grey system to predict the energy consumption of heat pump water heaters in residential buildings. They evaluated the influence of data sample interval in the prediction accuracy and found that the best interval is four weeks. This model requires little input data and the prediction error is within a normal range. Zhou et al. [88] did on-line prediction of cooling load by integrating two weather prediction modules into a simplified building thermal load model which is developed in [6], one is the temperature/relative humidity prediction which is achieved by using a modified grey model, the other is solar radiation prediction using a regression model. Experimental results showed that the performance of the simplified thermal network model is improved as long as the predicted weather data from the first module is used in the training process.

1.2.6 Discussion and conclusion

From the above description and analysis, it is obvious that a large number of calculations are needed to evaluate the building energy system, from sub-systems to building

Table 1.1: Summary of the work.

Problems	Statistical	ANNs	SVMs
Heating/Cooling	[23][26][27][28]	[41][42][43][44] [49][45][46][47] [48]	[79][80][82][85]
Electricity	[30][36][37][53]	[50][52][53][54] [53]	[77][78][81]
Simplify	[27][28][29]	[47][42][43][72] [73]	
System level	[26][29][30][31]		
Sub-system		[55][56][57][58][59] [47][60][61] [62]	
Energy parameters	[34]	[63][64][65] [66] [46][67]	
Energy index	[32][33]	[68][69]	
Data pre/post-processing	[31][37]	[74][75][70][71] [7][73]	[83][82][84][85]

level and even regional or national level. The reviewed research work is briefly summarized in Table 1.1, distinguished by considered problems and models. We omit engineering methods since many of them can solve all of the problems. Each model has its own advantages in certain cases of applications. The engineering model shows large variations. Many considerations can be involved in developing this model. It can be a very elaborate, comprehensive model which is applicable for accurate calculations. In contrast, by adopting some simplifying strategies, it can become a light-weight model and is easy to develop while maintaining accuracy. A commonly accepted drawback of this detailed engineering model is that it is difficult to perform in practice due to its high complexity and the lack of input information. The statistical model is relatively easy to develop but its drawbacks are also apparent, that is inaccuracy and lack of flexibility. ANNs and SVMs are good at solving non-linear problems, making them very applicable to building energy prediction. They can give highly accurate prediction as long as model selection and parameters setting are well performed. SVMs show even more superior performance than ANNs in many cases [81]. The disadvantages of these two types of models are that they require sufficient historical performance data and are extremely complex. The comparative analysis of these commonly used models is

summarized in Table 1.2. We note that this is just a rough summary since each model has large uncertainty or variations and is still being developed.

Table 1.2: Comparative analysis of the commonly used methods for the prediction of building energy consumption

Methods	Model Complexity	Easy to use	Running speed	Inputs needed	Accuracy
Elaborate Eng.	Fairly high	No	Low	Detailed	Fairly High
Simplified Eng.	High	Yes	High	Simplified	High
Statistical	Fair	Yes	Fairly high	Historical data	Fair
ANNs	High	No	High	Historical data	High
SVMs	Fairly high	No	Low	Historical data	Fairly high

This section has reviewed the recent work on prediction of building energy consumption. Due to the complexity of building energy behavior and the uncertainty of the influencing factors, many models were proposed for this application aiming at accurate, robust and easy-to-use prediction. Elaborate and simplified engineering methods, statistical methods, artificial intelligence, especially neural networks and support vector machines, are widely used models. Research mainly concentrates on applying these models to new predicting problems, optimizing model parameters or input samples for better performance, simplifying the problems or model development, comparing different models under certain conditions. Each model is being developed and has its advantages and disadvantages, therefore it is difficult to say which one is better without complete comparison under the same circumstances. However, artificial intelligence is developing rapidly, many new and more powerful technologies developed in this field may bring alternatives or even breakthroughs in the prediction of building energy consumption.

1.3 Summary of the main contributions

The main contributions of this thesis are as follows.

1. Sufficiently reviews the previous work on building energy analysis, including elaborate and simple engineering methods, statistical methods and artificial intelligence methods.

2. Introduces SVM principles in detail, including certain extensions and applications.

3. Generates historical energy consumption datasets by simulating in EnergyPlus, develops an interface to simulate multiple buildings.

4. Applies SVMs in the building energy prediction. Extensive experiments are designed to test the accuracy and robustness of this model by training on different types of historical profiles. It also applies this model in predicting a completely new building by involving building structure characteristics.

5. Applies a neural network model RDP (Recursive Deterministic Perceptron) in building energy fault detection.

6. Proposes a new method for building energy fault diagnosis, based on RDP classifiers.

7. For reducing variable dimensions, a new feature selection approach is proposed.

8. A new parallel approach is proposed to optimize the SVM training. Abundant experimental tests on benchmark datasets are performed and the results show that our implementation is superior than the state-of-the-art implementation Pisvm and it is especially suitable to multi-core and multi-processor systems.

9. Parallel SVM for regression is implemented and applied to predict building energy consumption when the historical dataset is very large.

10. Based on the summary of the existing work, some open problems and important future prospects are proposed.

This thesis is organized as follows. Chapter 2 will introduce the principles of SVMs. An important issue of this model, quadratic problem solver, is discussed in depth.

Chapter 3 presents how to generate the historical consumption data for model training and testing. The detailed simulation process, including the building description, the output of simulation and the interface for simulating multiple buildings, is presented.

Chapter 4 applies SVMs in the prediction of the energy consumption for a single building as well as for multiple buildings, and applies RDP neural networks in faulty consumption detection and diagnosis. The model performance is sufficiently analyzed by abundant experiments.

Chapter 5 proposes a new feature selection approach for the energy prediction by SVM model. The previous work on feature selection, the complete procedure of our approach and experiments are presented.

Chapter 6 demonstrates a new parallel implementation of SVM on multi-core systems. The related work, the detail of this implementation, the comparison of this system with other existing ones are given in this chapter.

Finally, the whole work is summarized in Chapter 7. And some important open problems and future work are also proposed in this chapter.

2

Support Vector Machine

SVMs are a set of methods that extract models or patterns from data. They are usually thought to be the best supervised learning algorithms in solving problems such as classification, regression, transduction, novelty detection, and semi-supervised learning. A basic idea of these algorithms is the Structural Risk Minimization (SRM) inductive principle, which aims at minimizing the generalization error through minimizing a summation of empirical risk and a VC dimension term. In other words, it trades off the model performance in fitting the training data (minimize the empirical risk term) with the model complexity (minimize the VC dimension term) [89]. Therefore this principle is different from the commonly used Empirical Risk Minimization (ERM) principle which only minimizes the training error. Based on this principle, SVMs usually achieve higher generalization performance in solving non-linear problems than other supervised learning algorithms that only implement the ERM principle.

This chapter will introduce SVM algorithms. Firstly the principles of SVM for classification is introduced in Section 2.1. Then, another algorithm for regression purpose is introduced in Section 2.2. This algorithm will be used later in this thesis. Other extensions of SVM, such as one-class SVM, transductive SVM, are briefly introduced in Section 2.3. The crucial issue of SVM algorithm, quadratic problem solvers, is presented in Section 2.4. Finally, some applications of SVMs are described in Section 2.5 in order to show the high popularity of this model.

2.1 Principles of SVM

SVM for classification (SVC) purpose aims at finding a hyperplane to separate two classes with maximum margin. Given training data set X that includes l samples, let x_i denotes the i^{th} sample, y_i denotes the corresponding label with the value either -1 or 1, $i = 1, 2, ..., l$. Let us start from linearly separable classification problem. Figure 2.1 gives a simple example where each sample has only two dimensions. We use solid circles to represent the points whose label is 1 and use empty circles to denote the points with label -1. Our aim is to find the optimal hyperplane that can separate these two classes and then works well on prediction of labels for unknown new points. We formulate this hyperplane (classifier) as follows.

$$h_{w,b}(x) = g(w^T x + b) \tag{2.1}$$

where $g(z) = 1$ if $z \geq 0$, or $g(z) = -1$ if $z < 0$. So that in Figure 2.1 the best separating line is $w^T + b = 0$. Then the training problem becomes how to find the best separating line. That means how to find the optimal values for parameters w and b.

Intuitively, if one point is far from the decision boundary, we may have more confidence to label it with '1' or '-1'. So we know that the best separating line is the one which has the largest distance from the points in both sides. Thus the largest distance is called maximum margin. Based on this consideration, we can derive the following optimization problem for finding w and b.

$$\min_{w,b} \quad \frac{1}{2}||w||^2 \tag{2.2}$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1, \quad i = 1, ..., l \tag{2.3}$$

This is a convex quadratic optimization problem (QP) with linear constraints and it can be efficiently solved by off-the-shelf quadratic problem solvers.

However, in practice, most of the problems are not linearly separable, which means that the above ideal case does not always happen. To make the classifier suitable for

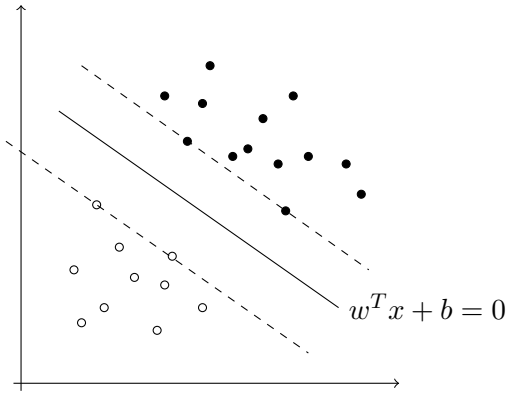

Figure 2.1: A linearly separable classification problem

non-linearly separable samples, we add the l_1-regularization to the function:

$$\min_{w,b} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}\xi_i \tag{2.4}$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1, ..., l \tag{2.5}$$

$$\xi_i \geq 0, \quad i = 1, ..., l \tag{2.6}$$

where ξ_i is an added slack variable corresponding to sample i. By doing this, the classifier will allow outliers which are misclassified (i.e. $\xi_i > 0$). In other words, the optimal hyperplane becomes less sensitive to the outliers. C is a user-defined constant value which is used to control the sensibility to outliers, i.e., if C is large, the classifier is more sensitive to the outliers. By adding the regularization term, the problem is becoming more complicated and can not be easily solved directly.

Let us use the method of Lagrange multipliers to derive the dual form of this optimization problem. The dual form is a finite optimization problem, one advantage of the dual form is that we can use kernel tricks to allow the model to solve non-linear problems. By introducing Lagrangian multipliers, we can form the lagrangian as:

$$L(w, b, \xi, \alpha, r) = \frac{1}{2}w^T w + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i[y_i(w^T x_i + b) - 1 + \xi_i] - \sum_{i=1}^{l} r_i\xi_i \tag{2.7}$$

where α and r are Lagrange multipliers with the constraints $\alpha_i, r_i \geq 0$. We let the

partial derivative of this Lagrangian with respect to w be equal to zero,

$$\Delta_w L = w - \sum_{i=1}^{l} \alpha_i y_i x_i = 0 \tag{2.8}$$

then we get:

$$w = \sum_{i=1}^{l} \alpha_i y_i x_i \tag{2.9}$$

Similarly, we set the partial derivatives with respect to b and ξ to be zero, then we get:

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2.10}$$

$$C - \alpha_i y_i - r_i = 0 \tag{2.11}$$

put equation (2.9) back to the Lagrangian, use (2.10), (2.11) to simplify the equation, then we can obtain the dual form of this problem:

$$\max_{\alpha} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j < x_i, x_j > \tag{2.12}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq C, \quad i = 1, ..., l \tag{2.13}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad i = 1, ..., l \tag{2.14}$$

where α is a vector of l variables which need to be optimized. So that this problem is a convex quadratic optimization problem with linear constraints. Since we have equation (2.9), if we find out α by solving this dual problem, we can write the decision function as:

$$\begin{aligned} w^T x + b &= (\sum_{i=1}^{l} \alpha_i y_i x_i)^T x + b \\ &= \sum_{i=1}^{l} \alpha_i y_i < x_i, x > + b \end{aligned} \tag{2.15}$$

this form is sparse since there are many α_i's equal to zero.

The dual form of the SVM can be written in the following convex quadratic form.

$$\min_{\alpha} \quad \frac{1}{2}\alpha^T Q\alpha - \sum_{i=1}^{l}\alpha_i \tag{2.16}$$

$$\text{subject to} \quad y^T\alpha = 0 \tag{2.17}$$

$$0 \leq \alpha_i \leq C \quad \forall i = 1, 2, ..., l \tag{2.18}$$

Where Q is a l by l positive semi-definite matrix. Each element of Q has the form $Q_{ij} = y_i y_j < x_i, x_j >$. We see that the entire algorithm is written in terms of the dot product $< x_i, x_j >$. If we use $\phi(x)$ to substitute x, then the dot product $< x_i, x_j >$ can be replaced as $< \phi(x_i), \phi(x_j) >$, we see there is no change to the algorithm. But the important thing is, $\phi(x)$ could be a mapping from a lower dimensional space to a higher feature space. So that for some problems of which points are non-linearly separable in the lower dimensional space, as long as we map them into higher dimensional space, it is possible to find a linear hyperplane for the newly generated points in the new feature space. This means we are able to solve non-linear problems by this mapping technique. Next, we will talk about kernel tricks which makes the algorithm more practical.

2.1.1 Kernel functions

Suppose that we have the feature mapping ϕ, we define the kernel function as $K(x, z) = \phi(x)^T\phi(z)$, then everywhere the inner product $< \phi(x_i), \phi(x_j) >$ can be further replaced by the kernel function $K(x_i, x_j)$. The elements of the matrix Q now become $Q_{ij} = y_i y_j K(x_i, x_j)$, hence, Q is also called kernel matrix. This replacement demonstrates that there is no need to explicitly express the form of the mapping ϕ. However, what kind of function can be used as the kernel function? Intuitively, we know a valid kernel should correspond to some feature mapping. In fact, the necessary and sufficient condition for a function to be a valid kernel is the following Mercer condition. Hence, the valid kernel is also called Mercer kernel.

Theorem 2.1.1 *Given $K : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$, for K to be a valid kernel, it is necessary and sufficient that for any given training set $\{x_1, x_2, ..., x_l\}, l < \infty$, the corresponding kernel matrix Q is symmetric positive semi-definite.*

Only Mercer kernel can be expressed in an equal form to $< \phi(x_i), \phi(x_j) >$. In practice, the commonly used kernel functions are:

- Linear function $K(x, z) = x^T z$
- Polynomial function $K(x, z) = (\gamma x^T z + coef)^d$
- Radial basis function (RBF) $K(x, z) = exp(-\gamma|x - z|^2)$
- Sigmoid function $K(x, z) = \tanh(\gamma x^T z + coef)$

Here we use x and z to substitute two samples x_i and x_j respectively.

The idea of kernels broadens the applicability of the SVMs, allowing the algorithm to work efficiently in the high dimensional feature space. Actually, the kernel tricks not only work well for SVMs, but also work for any learning algorithm which can be written in terms of inner product [90].

Since we have developed the kernel function K, if the problem is a classification problem of which the label of an instance is -1 or 1, we can use the following decision function to predict the labels of the new input x:

$$sgn\left(\sum_{i=1}^{l} \alpha_i y_i K(x_i, x) + b\right) \tag{2.19}$$

where b is a constant value which can be easily calculated in the training step, and

$$sgn(x) = \begin{cases} -1, & \text{for } x < 0 \\ 1, & \text{for } x > 0 \end{cases} \tag{2.20}$$

2.2 Support vector regression

Section 2.1 has discussed SVM for classification use, while in practice, there are many requirements for regression problems of which the target is a real continuous variable. For instance, the hourly energy consumption of a building is real continuous, it can not be solved by the classification model. Support vector regression (SVR) is designed for this purpose [91]. In this section, we introduce its principles. To make the estimation robust and sparse, we use a ε-insensitive loss function in constructing the model.

$$L(y - f(x)) = \begin{cases} 0 & \text{if } |y - f(x)| \leq \varepsilon \\ |y - f(x)| - \varepsilon & \text{otherwise} \end{cases} \tag{2.21}$$

This loss function indicates an ε-tube around the predicted target values as shown in Figure 2.2. This means that we assume there is no deviation of the predicted values

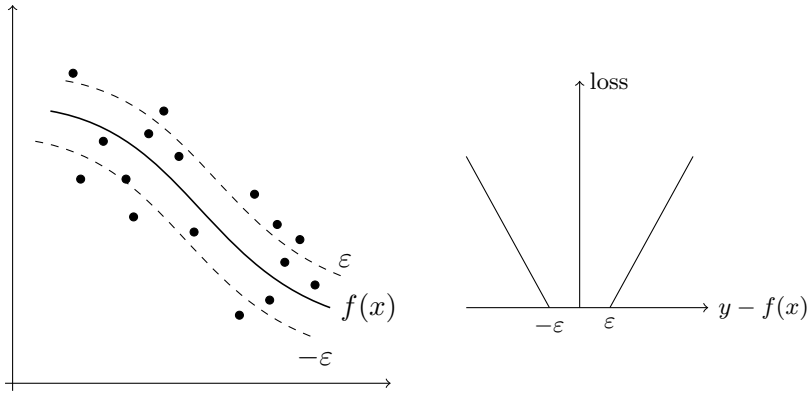

Figure 2.2: ε-tube for support vector regression

from the measured ones if they lie inside the tube (within the threshold ε). It is for the same reason that we introduce a slack variable into SVC as described in Section 2.1. Unlike SVC where only one slack variable is involved, for SVR, we introduce two slack variables ξ_i and ξ_i^*, $i = 1, 2, ..., l$. The objective function is as follows.

$$\min \quad \frac{1}{2}\|w\|^2 + C(\sum_{i=1}^{l} \xi_i^* + \sum_{i=1}^{l} \xi_i) \tag{2.22}$$

subject to the constraints

$$y_i - f(x_i) \leq \varepsilon + \xi_i^* \tag{2.23}$$

$$f(x_i) - y_i \leq \varepsilon + \xi_i \tag{2.24}$$

$$\xi_i^*, \xi_i \geq 0, \qquad i = 1, 2, ..., l \tag{2.25}$$

where C is a regularizing constant, which determines the trade off between the capacity of $f(x)$ and the number of points outside the ε-tube. To find the saddle point of the function (2.22) under the previous inequalities constraints, one can turn to the Lagrange function by introducing four Lagrange multipliers, α^*, α, γ^*, γ. The Lagrange function

becomes:

$$\begin{aligned} L(w,b,\xi^*,\xi,\alpha^*,\alpha,\gamma^*,\gamma) = \frac{1}{2}\|w\|^2 \\ + C(\sum_{i=1}^{l}\xi_i^* + \sum_{i=1}^{l}\xi_i) \\ - \sum_{i=1}^{l}\alpha_i[y_i - (wx_i) - b + \varepsilon + \xi_i] - \alpha_i^*[y_i - (wx_i) - b + \varepsilon + \xi_i^*] \\ - \sum_{i=1}^{l} l(\gamma_i^*\xi_i^* + \gamma\xi) \end{aligned} \tag{2.26}$$

The four Lagrange multipliers satisfied the constraints $\alpha^* \geq 0, \alpha \geq 0, \gamma^* \geq 0$ and $\gamma \geq 0, i = 1, 2, ..., l$. If the relations

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial b} = \frac{\partial L}{\partial \xi^*} = \frac{\partial L}{\partial \xi} = 0$$

occur, we are able to derive the following conditions,

$$w = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i)x_i \tag{2.27}$$

$$\sum_{i=1}^{l}\alpha_i^* = \sum_{i=1}^{l}\alpha_i \tag{2.28}$$

$$0 \leq \alpha_i^*, \alpha_i \leq C \tag{2.29}$$

$$C = \alpha_i^* + \gamma_i^* = \alpha_i + \gamma_i, \qquad i = 1, 2, ..., l \tag{2.30}$$

Putting them back into the above Lagrange function, we obtain the solution of the optimization problem which is equal to the maximum of the function (2.26) with respect to the Lagrange multipliers. The next step is to find α_i^* and α_i in order to maximize the following function:

$$W(\alpha_i^*, \alpha_i) = \sum_{i=1}^{l} y_i(\alpha_i^* - \alpha_i) - \varepsilon\sum_{i=1}^{l}(\alpha_i^* + \alpha_i) - \frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(x_i \cdot x_j) \tag{2.31}$$

under the constraints (2.28) and (2.29), where $x_i \cdot x_j$ stands for the dot product of two vectors x_i and x_j. Normally, only a certain part of the samples satisfy the property of $\alpha_i^* - \alpha_i \neq 0$, which are called support vectors (SVs). In fact, only these samples lying outside the ε-tube will contribute to determine the objective function. As in SVC, it is possible to use kernel function $K(x_i \cdot y_i)$ to replace the dot product. Since we have w as (2.27), the decision function can be developed to:

$$f(x) = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) K(x_i, x) + b \tag{2.32}$$

where α_i^* and α_i are determined by maximizing the quadratic function (2.31) under the constraints (2.28) and (2.29).

2.3 Other extensions

2.3.1 One-class SVM

In some classification problems, the classes are not very clear and we are not able to clearly label the samples as two classes or multiple classes. Sometimes, only the positive class is known and the negative samples can belong to any distributions. It can be regarded as $(1 + x)$-class problem. For instance, suppose we want to build a classifier which identifies researchers' web pages. While obtaining the training data, we can collect the positive samples by browsing some researchers' personal pages and form the negative samples which are not "a researcher's web page". Obviously, the negative samples vary in a large number of types and do not belong to any class. Therefore, it is hard to define the distribution of the negatives. In other words, "each negative sample is negative in its own way". This kind of problem can not be formulated as a binary class classification problem.

We are interested in the positive class and do not care too much about the negatives. One-class SVMs offer a solution to such classification problems [92]. The idea is to build a hyper-sphere which clusters the positive samples and separates them from the rest. Consider again the "maximum margin" and "soft margin" spirits, the hyper-sphere gives boundary to most of the positive points, not all, to avoid over-fitting. Suppose c is the center of the positive points, $\Phi(x)$ is the feature map, ξ is the slack variable, the

one-class SVMs is to solve the following problem:

$$\min_{R\in\mathbb{R},\xi\in\mathbb{R}^l} \quad R^2 + \frac{1}{\nu l}\sum_i \xi_i \tag{2.33}$$

$$\text{subject to} \quad ||\Phi(x_i) - c||^2 \leq R^2 + \xi_i \tag{2.34}$$

$$\xi_i \geq 0, i = 1, 2, ..., l \tag{2.35}$$

From this primal form, we can derive the dual form as:

$$\min_{\alpha} \quad \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) - \sum_i \alpha_i K(x_i, x_i) \tag{2.36}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq \frac{1}{\nu l} \tag{2.37}$$

$$\sum_i \alpha_i = 1 \tag{2.38}$$

where K is a kernel function and the ν is an upper bound on the fraction of outliers which are outside the estimated region and a lower bound on the fraction of support vectors. The decision function is formulated as:

$$f(x) = sgn\left(R^2 - \sum_{ij} \alpha_i \alpha_j K(x_i, x_j) + 2\sum_i \alpha_i K(x_i, x) - K(x, x)\right). \tag{2.39}$$

where $sgn(x)$ is defined as the same as Equation (2.20).

2.3.2 Multi-class SVM

When there are more than two classes in the data, and the aim is to build a classifier that can classify all of the classes, we call such problems multi-class classification problems. Multi-class SVMs are used to solve these problems. There are several approaches for implementing multi-class SVMs.

The first is one-against-all method. The idea is straightforward, suppose there are k classes in the training data, we train one SVM for one class, we suppose the samples in this class are positive and the rest from all other classes are negative. So that we totally train k SVMs. Given a new sample, we assign it to the class with the highest objective value [93].

The second one is pairwise method. We construct a normal two-class SVM for each

pair of classes. While training this SVM, we just use the training data within these two classes and ignore other training samples. So that if there are k classes, we will train $k(k-1)/2$ SVMs. In the estimation, we use a voting strategy, i.e., given an unknown objective sample, we use each SVM to evaluate which class it belongs to, and the final decision is the class which gets the maximum votes. This method is firstly used in [94] and [95].

The third one is called pairwise coupling. This method is designed for the case where the output of the two-class SVM can be written as the posterior probability of the positive class. For a given unknown example, the selected posterior probabilities $p_i = Prob(\omega_i|x)$ are calculated as a combination of the probabilistic outputs of all binary classifiers. Then the example is assigned to the class with the highest p_i. This method is proposed by Hastie and Tibshirani in [96]. However, the decision of a SVM is not a probabilistic value. Platt [97] proposed a sigmoid function to map the decision of a SVM classifier to the positive class posterior probability:

$$Prob(\omega|x) = \frac{1}{1+e^{Af+B}} \tag{2.40}$$

where A and B are parameters and f is the decision of SVM with regard to sample x. Another model Kernel Logistic Regression (KLR) [98] has the output form in terms of positive class posterior probability, so that it can be used directly as the binary classification in the pairwise coupling method [99].

2.3.3 v-SVM

When we stated one-class SVM in section 2.3.1, we introduced a new parameter ν. We stated that it controls training errors and the number of support vectors. By introducing this parameter, Schölkopf et al. [100] have derived a SVM model for classification problems names as ν-SVC where the original parameter C is replaced by ν. And also, for regression problems, the corresponding ν-SVR is developed.

The primal form of ν-SVC is:

$$\min_{w,b,\xi,\rho} \quad \frac{1}{2}||w||^2 - \nu\rho + \frac{1}{l}\sum_i \xi_i \tag{2.41}$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq \rho - \xi_i, \tag{2.42}$$

$$\xi_i \geq 0, i = 1, 2, ..., l, \rho \geq 0. \tag{2.43}$$

Then we can derive the dual form as:

$$\min_{\alpha} \quad \frac{1}{2}\alpha^T Q\alpha \tag{2.44}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq \frac{1}{l}, i = 1, ..., l \tag{2.45}$$

$$e^T\alpha \geq \nu, \quad y^T\alpha = 0. \tag{2.46}$$

where Q is the kernel matrix.

In ν-SVR, the new parameter is used to replace ε instead of C. The primal form is:

$$\min_{w,b,\xi,\xi^*,\varepsilon} \quad \frac{1}{2}||w||^2 + C(\nu\varepsilon + \frac{1}{l}\sum_i(\xi_i + \xi_i^*)) \tag{2.47}$$

$$\text{subject to} \quad (w^T x_i + b) - z_i \leq \varepsilon + \xi_i, \tag{2.48}$$

$$z_i - (w^T x_i + b) \leq \varepsilon + \xi_i^*, \tag{2.49}$$

$$\xi_i, \xi_i^* \geq 0, i = 1, ..., l, \varepsilon \geq 0. \tag{2.50}$$

The dual form is:

$$\min_{\alpha,\alpha^*} \quad \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + z^T(\alpha - \alpha^*) \tag{2.51}$$

$$\text{subject to} \quad \sum_i \alpha_i - \sum_i \alpha_i^* = 0, \tag{2.52}$$

$$\sum_i \alpha_i + \sum_i \alpha_i^* \leq C\nu, \tag{2.53}$$

$$0 \leq \alpha, \alpha^* \leq C/l, \quad i = 1, ..., l. \tag{2.54}$$

The decision functions of ν-SVC and ν-SVR for a given new unlabeled sample x are the same as that of SVC and ε-SVR, which are (2.19) and (2.32) respectively.

2.3.4 Transductive SVM

The above introduced SVMs are trained on labeled training data and then used to predict the labels on unlabeled testing data. We can call these models inductive SVMs. Contrary to this idea, transductive SVM is trained on the combination of the labeled training data and unlabeled testing data. Therefore, it is a semisupervised learning algorithm. The basic idea of transductive SVM is to build a hyperplane that maximizes the separation between labeled and unlabeled datasets [93].

Suppose there are l labeled data $\{(x_1, y_1), ..., (x_l, y_l)\}$ and u unlabeled data $\{x_1^*, ..., x_u^*\}$. To derive the hyperplane that separates these two datasets with maximum margin, firstly we label the unlabeled data by using an inductive SVM, suppose the resulting transductive data becomes $(x_1^*, y_1^*), ..., (x_u^*, y_u^*)$, then we solve the following problem:

$$\min_{w,b,\xi,\xi^*} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}\xi_i + C^*\sum_{j=1}^{d}\xi_j^* \tag{2.55}$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 - \xi_i \tag{2.56}$$

$$y_j^*(w^T x_j^* + b) \geq 1 - \xi_j^* \tag{2.57}$$

$$\xi_i \geq 0, i = 1, ..., l, \xi_j^* \geq 0, j = 1, ..., d. \tag{2.58}$$

where ξ and ξ^* are slack variables and C and C^* are two penalty constants. It is not necessary to use all of the unlabeled samples for learning, so that d ($d \leq u$) is introduced to control the number of transductive samples. The dual form is:

$$\min_{\alpha,\alpha^*} \quad \sum_{i=1}^{n}\alpha_i + \sum_{j=1}^{d}\alpha_j^* - \frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}G(\alpha_i, \alpha_j) + 2\sum_{i=1}^{n}\sum_{j=1}^{d}G(\alpha_i, \alpha_j^*) + \sum_{i=1}^{d}\sum_{j=1}^{d}G(\alpha_i^*, \alpha_j^*)\right) \tag{2.59}$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq C, 1 \leq i \leq n \tag{2.60}$$

$$0 \leq \alpha_j^* \leq C, 1 \leq j \leq d \tag{2.61}$$

$$\sum_{i=1}^{n} y_i\alpha_i + \sum_{j=1}^{d} y_j^*\alpha_j^* = 0 \tag{2.62}$$

where $G(\alpha_i, \alpha_j) = \alpha_i\alpha_j y_i y_j K(x_i, x_j)$, $G(\alpha_i, \alpha_j^*) = \alpha_i\alpha_j^* y_i y_j^* K(x_i, x_j^*)$, $G(\alpha_i^*, \alpha_j^*) = \alpha_i^*\alpha_j^* y_i^* y_j^* K(x_i^*, x_j^*)$. And the decision function is as follows:

$$f(x) = sgn\left(\sum_{i=1}^{n}\alpha_i y_i K(x_i, x) + \sum_{j=1}^{d}\alpha_j^* y_j^* K(x_j^*, x) + b\right). \tag{2.63}$$

The function $sgn(x)$ is defined as the same as Equation (2.20).

2.4 Quadratic problem solvers

Let us first introduce the Karush-Kuhn-Tucher (KKT) conditions which determine whether the optimized variables are solutions to the primal and dual problems.

Since SVMs have several variations and their forms are complex, to avoid tedious notations, we use a general and simplified problem to discuss the optimality conditions. Consider the following convex optimization problem with an equality and an inequality constraint:

$$\min_{x} \quad f(x) \tag{2.64}$$

$$\text{subject to} \quad g_i(x) \leq 0 \quad i = 1, ..., m \tag{2.65}$$

$$h_j(x) = 0 \quad j = 1, ..., p \tag{2.66}$$

where $f(x)$ and $g(x)$ are convex functions. The corresponding Lagrangian is $L(x, \alpha, \beta) = f(x) + \alpha^T g(x) + \beta^T h(x)$ where $\alpha \in \mathbb{R}^m$ ($\alpha_i \geq 0$) and $\beta \in \mathbb{R}^p$ are two vectors of lagrange multipliers. The variables of the vector $x \in \mathbb{R}^n$ are called primal variables and α_i's and β_i's are called dual variables. The optimization problem is equal to the following problems:

$$\text{Primal:} \quad \min_{x}[\max_{\alpha,\beta:\alpha_i \geq 0,\forall i} L(x, \alpha, \beta)] \tag{2.67}$$

$$\text{Dual:} \quad \max_{\alpha,\beta:\alpha_i \geq 0,\forall i}[\min_{x} L(x, \alpha, \beta)] \tag{2.68}$$

Suppose the solution of the optimization problem is x^*, α, β (local minimum point), they satisfy the constraints, then there is a vector of α and a vector of β which satisfy the KKT conditions which are as follows:

$$\Delta f(x^*) + \alpha^T \Delta g(x^*) + \beta^T \Delta h(x^*) = 0 \tag{2.69}$$

$$g_i(x^*) \leq 0, \quad i = 1, ..., m \tag{2.70}$$

$$h_j(x^*) = 0, \quad j = 1, ..., p \tag{2.71}$$

$$\alpha_i \geq 0, \quad i = 1, ..., m \tag{2.72}$$

$$\alpha_i g_i(x^*) = 0, \quad i = 1, ..., m \tag{2.73}$$

The first equation 2.69 is the gradient of the Lagrangian function, which indicates the

solution is the stationarity point of the Lagrangian function. The equation 2.70 and 2.71 guarantee the solution primal feasible while the equation 2.72 ensures dual feasible. The last one is called KKT dual complementarity condition which implies if $\alpha_i > 0$ then $g_i(x^*) = 0$, in this case, we call the equal constraints active constraints. In SVM, the active constraints are support vectors. When the KKT conditions are fulfilled to the dual form of SVM, the following KKT dual complementarity condition can be derived:

$$\alpha_i = 0 \Rightarrow y_i(w^T x_i + b) \geq 1 \tag{2.74}$$

$$\alpha_i = C \Rightarrow y_i(w^T x_i + b) \leq 1 \tag{2.75}$$

$$0 < \alpha_i < C \Rightarrow y_i(w^T x_i + b) = 1 \tag{2.76}$$

Many off-the-shelf quadratic problem solvers can be used to train SVMs. There are three important and widely considered methods, interior point, gradient descent and decomposition method. We will introduce the first two solvers in the following two sub-sections and leave the third one to be stated in Chapter 6.

2.4.1 Interior point method

The idea behind primal-dual interior point method is to solve the problem in terms of primal and dual forms simultaneously. The optimal solution point is found by searching and testing interior points in the feasible region iteratively. In the section, we present the fundamentals of this algorithm follows [101], [102] and [90]. Consider the following convex quadratic problem which is a general dual form of SVMs:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^T Q\alpha + c^T\alpha \\ \text{subject to} \quad & A\alpha = b \\ & l \leq \alpha \leq u \end{aligned} \tag{2.77}$$

where $\alpha \in \mathbb{R}^n$ is a vector of free variables, $c, l, u \in \mathbb{R}^n, b \in \mathbb{R}^m$ are vectors with constant value, $A \in \mathbb{R}^{m\times n}$ is a full rank coefficient matrix. To develop the dual form of this problem, we first change the inequalities constraint into positivity constraints by

introducing two slack variables s and t, then the constraints become:

$$A\alpha = b \tag{2.78}$$
$$\alpha - s = l \tag{2.79}$$
$$\alpha + t = u \tag{2.80}$$
$$s \geq 0 \tag{2.81}$$
$$t \geq 0 \tag{2.82}$$

Now, we will develop the dual form. By introducing five Lagrange multipliers $\lambda, k, -w, y, z$ to associate with the five constraints respectively, then formulating the Lagrangian function $L(\alpha, \lambda, k, w, y, z)$ and letting $\frac{\partial L}{\partial \alpha} = 0, \frac{\partial L}{\partial s} = 0, \frac{\partial L}{\partial t} = 0$, we get the following equations:

$$Q\alpha + c - A^T\lambda - k + w = 0 \tag{2.83}$$
$$y = k \tag{2.84}$$
$$z = w \tag{2.85}$$

Substitute these equations to the following problem which defines the dual form:

$$\max \quad L(\alpha,\lambda, k, w, y, z) \tag{2.86}$$
$$\text{subject to} \quad L'_\alpha = 0 \tag{2.87}$$
$$y \geq 0 \tag{2.88}$$
$$z \geq 0 \tag{2.89}$$

Then we can develop the dual form of our problem (2.77) as:

$$\max \quad -\frac{1}{2}\alpha^T Q\alpha + b^T\lambda + l^T k - u^T w \tag{2.90}$$
$$\text{subject to} \quad Q\alpha + c - A^T\lambda + w = k \tag{2.91}$$
$$k \geq 0 \tag{2.92}$$
$$w \geq 0 \tag{2.93}$$

where λ is the vector of dual variables. The KKT conditions are the constraints of the primal and dual forms (2.78-2.82 and 2.91-2.93) plus with the following complementar-

ity conditions:

$$s_i k_i = 0, \quad t_i w_i = 0, \quad \forall i = 1, 2, ..., n \tag{2.94}$$

These conditions mean the duality gap is zero, which is equivalent to the fact that the primal and dual objective functions reach the equal extreme value. It is known that the KKT conditions provide the necessary and sufficient conditions for a solution to be optimal.

In the interior point algorithm, if a solution satisfies the primal and dual constraints, we say this solution is primal and dual feasible. We approach the optimal solution by trying candidate points in the feasible region step by step. The complementarity conditions (2.94) are used to determine the quality of the current solution. For this purpose, we do not try to find the solution for (2.94) directly, instead, we set the duality gap as μ (> 0) and decrease μ iteratively until the duality gap is small enough. Correspondingly, the complementarity conditions are modified as follows:

$$s_i k_i = \mu, \quad t_i w_i = \mu, \quad \forall i = 1, 2, ..., n \tag{2.95}$$

In other words, we approximate the optimal solution (where duality gap is zero) by an iterative predictor-corrector approach. In each iteration, for a given μ, we find a more feasible solution, then decrease μ and repeat until μ falls below a pre-defined tolerance. So that the substantial problem of interior point algorithm is how we move from the current point $(\alpha, \lambda, k, w, y, z)$ to the next point $(\alpha', \lambda', k', w', y', z')$. Let $(\Delta\alpha, \Delta\lambda, \Delta k, \Delta w, \Delta y, \Delta z)$ be the search direction, then we have the updating as:

$$(\alpha', \lambda', k', w', y', z') = (\alpha, \lambda, k, w, y, z) + (\Delta\alpha, \Delta\lambda, \Delta k, \Delta w, \Delta y, \Delta z) \tag{2.96}$$

Put them in the KKT conditions, we require the next point satisfies:

$$A(\alpha + \Delta\alpha) = b \tag{2.97}$$

$$\alpha + \Delta\alpha - s - \Delta s = l \tag{2.98}$$

$$\alpha + \Delta\alpha + t + \Delta t = u \tag{2.99}$$

$$Q(\alpha + \Delta\alpha) + c - A^T(\lambda + \Delta\lambda) + w + \Delta w = k + \Delta k \tag{2.100}$$

$$(s + \Delta s)(k + \Delta k) = \mu \tag{2.101}$$

$$(t + \Delta t)(w + \Delta w) = \mu \tag{2.102}$$

Suppose μ is given, then the Newton method is used to solve these equations. The augmentation of the free variables in each iteration can be derived as:

$$\begin{bmatrix} -H & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta\alpha \\ \Delta\lambda \end{bmatrix} \begin{bmatrix} \bar{r} \\ \bar{v} \end{bmatrix} \tag{2.103}$$

where $H = (Q + s^{-1}k + t^{-1}w)$, $\bar{r}$ and $\bar{v}$ are appropriately defined residues, readers may consult [102] for more details. For solving $\Delta\lambda$, firstly we need to calculate the coefficient matrix $(AH^{-1}A^T)$. The calculation and factorization of this matrix is the most time-consuming process of interior point method, leading to a computational cost as high as $O(n^3)$ and the memory requirement is $O(n^2)$ for each iteration. If Q is easily invertible, the algorithm would be more efficient.

2.4.2 Stochastic gradient descent

Gradient descent methods approximate the minimum of the objective by iteratively updating the variables in the direction of gradient descent. Two main steps are repeated in the algorithm, one is calculating the descent direction $P(\Delta f(w))$ which is some function of the gradient and searching for step size η (> 0, also called learning rate), the other one is updating the variables in w.

Step 1. Calculating the direction $P(\Delta f(w^t))$ and step size η_t
Step 2. Updating variables $w^{t+1} = w^t - \eta_t P(\Delta f(w^t))$

These methods are common solvers for convex optimization problems. However, a well know disadvantage is the slow convergence rate, and even under some circumstances, the convergence may not be guaranteed. By considering this problem, Zhang [103] introduced stochastic gradient descent method in solving large scale linear prediction problems. This method can be easily applied for SVM QP optimization, performing directly on the primal form. Different from batch gradient descent methods, in which the whole data samples are examined in each iteration, Zhang's stochastic gradient descent method requires only one random sample from the training data in each iteration. The algorithm is guaranteed to be convergent in a number of iterations. As described in [103], the number of iterations T satisfies $1/T = O(\epsilon^2)$ where ϵ is a pre-defined accuracy.

When this algorithm is used for SVM QP solver, it works as follows. Suppose (x, y) is one sample of the training set A, the size of A is m, the loss function is $l(w, x, y)$, then the objective function can be written as

$$f(w) = \frac{\lambda}{2}||w||^2 + \frac{1}{m}\sum_{(x,y)\in A} l(w, x, y) \tag{2.104}$$

In iteration t, suppose the sample (x_t, y_t) is chosen, then we form the descent direction as $S_t^{-1}(\lambda w_{t-1} + l'_w(w_{t-1}, x_t, y_t))$ where S is a pre-conditioner used to accelerate the convergence rate. Therefore the variables are updated by the rule

$$w_t = w_{t-1} - \eta_t S_t^{-1}(\lambda w_{t-1} + l'_w(w_{t-1}, x_t, y_t)) \tag{2.105}$$

Then the stochastic gradient descent algorithm can be summarized as in Algorithm 2.1

Algorithm 2.1

Initialize w_0
for t = 1, 2, ..., T
  Choose one sample (x_t, y_t) from A randomly
  Calculate w_t as:
    $w_t = w_{t-1} - \eta_t S_t^{-1}(\lambda w_{t-1} + l'_w(w_{t-1}, x_t, y_t))$
Output w_T

We see that the number of iterations required to obtain a solution of accuracy ϵ is $O(1/\epsilon^2)$, in 2007, Shalev-Shwartz et al. [104] modified the stochastic gradient descent algorithm, further improved the required number of iterations to reach the scale of $O(1/\epsilon)$. Moreover, the total run-time of the proposed method can be quantitatively recorded as $O(d/(\lambda\epsilon))$ where d is relevant to the number of non-zero features in each sample. Since this rate does not depend on the number of samples, this method is especially suitable for solving SVM on large scale problems, even producing runtime decrease while the data size is increasing [105].

The new algorithm proposed by Shalev-Shwartz et al. is named Pegasos. It contains two main modifications on the previous stochastic gradient descent method. The first one is that, in each iteration, we choose k samples instead of only one sample for calculating sub-gradient. The other one is that, after updating w, we do one more step

to project w on the L_2 ball of radius $1/\sqrt{\lambda}$. In iteration t, after choosing a subset A_t which contains k samples from the training set A, the sub-level objective function can be written as

$$f(w) = \frac{\lambda}{2}||w||^2 + \frac{1}{k}\sum_{(x,y)\in A_t} l(w,x,y) \tag{2.106}$$

Now the variable updating rule includes two steps, one is normal gradient descent:

$$w_t = w_{t-1} - \eta_t f'_w(w_{t-1}) = (1-\eta_t\lambda)w_{t-1} - \frac{\eta_t}{k}\sum_{(x,y)\in A_t} l'_w(w_{t-1},x,y) \tag{2.107}$$

The other is scaling w_t by $\min\{1, \frac{1}{\sqrt{\lambda}||w_t||}\}$. The whole algorithm can be summerized in Algorithm 2.2.

Algorithm 2.2
Input: A, λ, T, k
Initialize: Choose w_0 satisfies $\|\|w_0\|\| \leq \frac{1}{\sqrt{\lambda}}$
for t = 1, 2, ..., T
Choose $A_t \subseteq A$, where $\|A_t\| = k$
Set learning rate $\eta_t = \frac{1}{\lambda t}$
Calculate $\hat{w}_t = (1-\eta_t\lambda)w_{t-1} - \frac{\eta_t}{k}\sum_{(x,y)\in A_t} l'_w(w_{t-1},x,y)$
Calculate $w_t = \min\{1, \frac{1}{\sqrt{\lambda}\|\|\hat{w}_t\|\|}\}\hat{w}_t$
Output w_T

In practice, if the loss function is defined as $l(w,x,y) = \max\{0, 1-y\langle w,x\rangle\}$, then it is only necessary to consider samples which have the attribute $y\langle w,x\rangle < 1$.

Each time, the k samples are chosen independent and identically distributed from the training set. Consider the extreme cases, if we choose $k = |A|$, then the algorithm becomes sub-gradient projection method. In contrast, if we select $k = 1$, then we obtain a variant of the previous stochastic gradient descent method. Experimental results show that the projection operation can largely improve the convergence speed [104].

2.5 Applications

Since SVMs produce strong generalization ability, they are becoming popular in a wide variety of application domains. This section briefly introduces some of them and shows

how SVMs are applied to solve these problems.

One important task of the Internet application is the automatical text categorization or what we call text classification. The aim is to automatically classify documents into pre-defined categories, such as classifying web pages into News, Sports, Science, Health, Technology, etc. Joachims [106] firstly introduced SVMs in the application of text categorization. In his work, each category, or we say each label, was treated as a binary classification problem. To collect the training data, each document was treated as a sample, and each distinct word was regarded as a feature. The value of the feature for each sample was the number of times the word occurs in this document. To reduce unnecessary features, the "stop words" such as "and", "or" and the words occurring less than three times were ignored. He also argued that SVMs were applicable in this application because they were essentially suitable to the properties of text which were high dimensional feature spaces, few irrelevant features and sparse instance vectors. Experimental results showed that SVMs outperformed other learning algorithms, such as Bayes, Rocchio, C4.5, k-NN, in both generalization ability and robustness. Since text can be classified automatically, we can use this technology in numerous applications. As concluded by Sebastiani and Ricerche [107], they could be automated indexing of scientific papers or patents, selective dissemination of information to customers, spam mail filtering, intrusion detection, authorship attribution, survey coding, and even automated essay grading .

Another application field that broadly uses SVMs is computational biology [108][109], including remote protein homology detection, classification of genes, tissue, proteins and other microarray gene expression analysis, recognition of translation start sites, prediction of protein-protein interactions, functional classification of promoter regions, and peptide identification from mass spectrometry data. Noble [108] explained why SVMs are successfully applied in these applications relevant to computational biology. First, these applications generally involve high dimensional data with noises, for which SVMs perform well compared to other intelligent methods. Second, the kernel methods can easily handle different sorts of data, such as vectors, strings, trees, graphs, etc. These non-vector data types are common in biology applications, leading to the requirement of specially designed kernels, such as Fisher kernel [110], composition-based kernel [111], Motif kernel [112], pairwise comparison kernel [113], etc.

SVMs have also been widely used in image processing. One of the important tasks is the content-based image retrieval (CBIR). Much research has been carried out on using SVMs together with relevance feedback method in CBIR. The usual steps to achieve this purpose are as follows. First, we retrieve images through a traditional method, then user feeds back the top images in the retrieval result with relevance or irrelevance information to form the training data. Next, SVMs are trained and used to calculate scores for all of the images. Finally, the images are sorted according to their scores which reflect the relevance of the images to user's requirements [114]. Besides image retrieval, SVMs are also applied in image classification, segmentation, compression, and other multimedia processing. These topics attract much more attention since multimedia content is blooming on the Internet in recent years.

Other applications also benefit from the high prediction performance of SVMs, such as E-learning, handwriting recognition, traffic prediction, 2-D or 3-D object detection, cancer diagnosis and prognosis, etc. A large amount of research is carried out on extensive SVMs in solving new problems — treating new applications, dealing with large scale datasets, and achieving active or online learning.

3

Data Generation

3.1 Common approaches

As described in Section 1.1, the energy system in buildings is complex with so many uncertainties, therefore, accurate consumption data, especially the time series for each variable are difficult to obtain. How to collect sufficient data is always an important concern in statistical analysis or model development. In the previous studies, there are three common approaches to solve this problem, survey or questionnaire from customers, measurement in real buildings and simulation by some well developed programs. Surveys or questionnaires are usually launched by utility companies, building management companies, energy analysis companies, or government organizations. For instance, the benchmarking model developed in [69] uses electricity consumption data which is collected from Commercial Buildings Energy Consumption Survey (CBECS) database. This energy database is developed by Energy Information Administration of the U.S. Department of Energy, containing energy-related information of commercial buildings in the United States, building characteristics, and their energy consumption and expenditures. Some other researchers use measurement data in their analysis. In the early work [48], the investigated building was a holiday home, the recorded variables were easily measurable, i.e., season, the insulation conditions in all four walls, actual thickness, whether the heat transfer coefficient was constant, and time of day. In [55], the load prediction model was tested on the measurements of an HVAC system in service. In [66], the authors investigated 87 single-family buildings in Sweden. Fifteen thermocouples were used for measuring the temperatures in each house. Nine of

them inside and six outside, providing the internal and external average temperatures respectively. The data was recorded every second and averaged for minute intervals.

The first two approaches, survey and measurement, aim at recording real consumption data. They usually take a very long time and run the risk of inaccuracy in practice. For the sake of simplicity, many researchers prefer to use simulation methods instead of them. For instance, in [76], the authors used EnergyPlus to simulate an administration building of the University of Sao Paulo. They kept the building description and its internal loads as simple as possible in order to avoid over-detailed modeling. They chose the climate data and a set of parameters that briefly described the building: geometry, wall and window materials, lighting, equipment and occupancy schedules. The daily total energy consumption was recorded and compared with the actual measured data. It turns out that they are highly consistent. In the simulation period, i.e., from 1^{st} January to 31^{st} March 2005, 80% of the simulated energy demands came quite close to the measured one. In fact, as long as calibration is well performed, simulation approach is possible to produce energy consumption data that approaches real profiles very closely [21].

There are a wild variety of programs that can be used to simulate building energy profiles, as listed in [12]. This thesis aims at developing effective and high performance artificial intelligence models. We are not overly concerned by the physical profiles of the building and its energy system, therefore, the data required in the experiments are obtained from simulation method. We choose EnergyPlus as our simulation tool. We will explain why EnergyPlus and how our simulation is performed in the next sections.

3.2 EnergyPlus

EnergyPlus is the succession of the well-known energy and load simulation tool BLAST and DOE-2.1E, maintained by U.S. department of Energy. Inheriting the capabilities and advantages of its two legacy programs, EnergyPlus is comprehensive in energy analysis and thermal simulation of complex building systems. It is an open-source, modular, structured program, and can be inspected and improved by any users or developers.

Figure 3.1 gives a brief view of the structure of this simulation program. Given a user's description of a building, EnergyPlus will calculate the heating and cooling loads

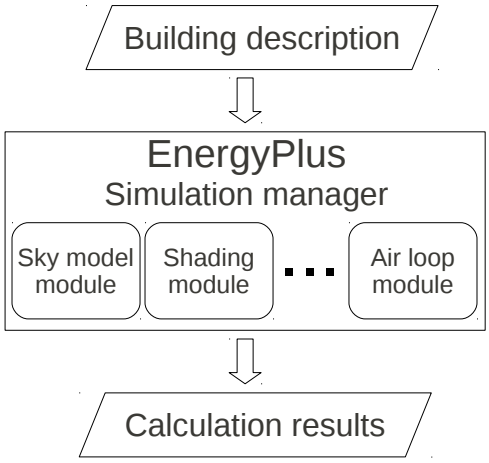

Figure 3.1: An overview of the simulation tool — EnergyPlus.

which are necessary to maintain thermal control setpoints, conditions throughout a secondary HVAC system and coil loads, and the energy consumption of primary plant equipment as well as many other simulation details that are necessary to verify that the simulation is performing in the way that the actual building would do. This integrated solution provides more accurate space temperature prediction which is crucial for system and plant sizing, occupant comfort and occupant health calculations.

The following list gives the most popular features of the EnergyPlus, which guarantee the high reliable outputs.

- Integrated, simultaneous solution of the building response and the primary and secondary system.
- Sub-hourly, user-definable time steps (15 minutes default).
- ASCII text based weather, input, and output files.
- Heat balance based solution for building thermal loads calculation.
- Transient heat conduction.
- Improved ground heat transfer modeling.
- Combined heat and mass transfer model.
- Thermal comfort models.

- Anisotropic sky model.
- Advanced fenestration calculations.
- Daylighting controls.
- Loop based configurable HVAC systems.
- Atmospheric pollution calculations.
- Links to other popular simulation environments/components.

More details on each feature can be found in the documentation library of EnergyPlus [115].

3.3 Simulation details

In order to test the model in this application sufficiently, we simulate energy consumption data for both single and multiple buildings. All of these buildings are for office use and located in France. For each dataset, we record the total heating demand or total electricity consumption as the target. We also record dozens of variables as the features into the datasets. Fortunately, these variables can be extracted easily from the output of the simulation program. Each sample of the dataset is hourly consumption.

Weather conditions are important factors that determine building's energy consumption. We suppose the simulated single building is located in an urban area, such as Paris-Orly, therefore the weather data in such a place is used as a part of inputs during the simulation. The inputs for EnergyPlus should be formatted in a file (with suffix as `.idf`). Other than the weather conditions, the inputs also contain the descriptions of buildings, occupant's behavior, etc. Since the inputs are in large quantity and there is no need to list them all, we put them into four categories as listed below, weather conditions, building structure characteristics, occupant behaviors, inner facilities and their schedules.

- **Weather conditions**, including dry bulb temperature, relative humidity, global horizontal radiation, wind speed, etc.
- **Building structure characteristics**, including shape, capacity, orientation, window/wall radio, fenestration/shading, building materials, thermal zones, etc.

- **Occupants' behaviors**, such as number of occupants, their leaving and entering time, thermal comfortable set points, ventilation, etc.

- **Inner facilities and their schedules**, such as lighting system, HVAC system, TV, PC, etc.

3.3.1 Weather conditions

Actually, the weather conditions are very complicated, they vary so much over the time and according to the location. Fortunately, the real recorded weather data for recent years is readily available. We choose them in our simulation. For example, when we want to simulate a building in Paris-Orly, we choose the weather data in this area as the inputs to EnergyPlus. The considered weather data includes dry bulb air temperature, relative humidity, wind speed, global horizontal radiation and ground temperature. To have an overview of their variations, the dry bulb air temperatures of the first 20 days in January and July are plotted in Figure 3.2, and the relative humidity on the same days are plotted in Figure 3.3. We can see many sudden changes in these curves.

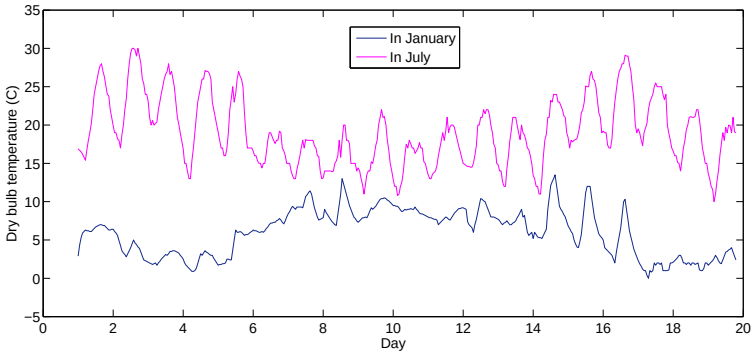

Figure 3.2: Dry bulb temperature in the first 20 days of January and July

In our work, we try to generate the consumption data of a single building in heating season initially. Then, by modifying some alterable input parameters, we generate the consumption profiles for multiple houses.

3.3.2 Simulating one single building

The first building is simulated in heating season, i.e., from November 1^{st} to March 31^{st}. The description of this building is shown in Table 3.1. In order to show more details,

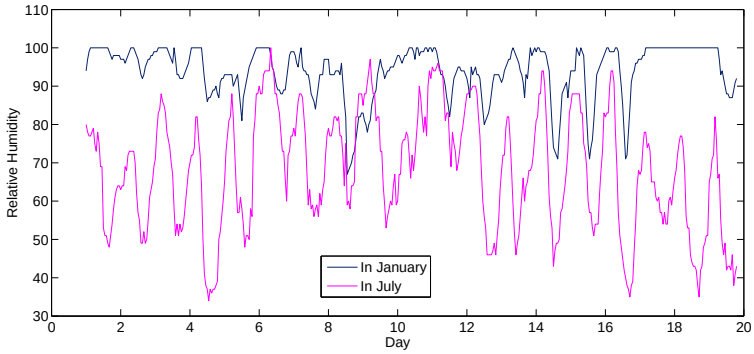

Figure 3.3: Relative humidity in the first 20 days of January and July

we extract the materials of surfaces and put them in a new Table 3.2. These materials determine the thermal behavior of the building envelope will significantly influence the total energy consumption. Furthermore descriptions of these materials can be found in the documents of EnergyPlus [115].

Table 3.1: Description of a single building (in metric units)

Parameters	Values
Location	Paris-orly, City
Duration	From Nov 1^{st} to Mar 31^{st}
Time Step	15min
Building Shape	Rectangle
Structure	Length:11 Width:10 Ceiling Height:4 North axis: 10^o
Fenestration Surface	$14m^2$ for each wall
Thermal Zones	1
People	14
Air Infiltration	0.0348 m^3/s
Heating Type	District Heating
Cooling Type	HVAC windowAirCondioner
Other Facilities	Light, Water heater

For the sake of simplicity, we suppose there is only one floor and one room in this building. Since it is in heating season, the energy consumed in this building mainly comes from three sources: the district heating which is used to keep the inside temperature at a constant level, electricity plants which are used mostly on working days, and hot water for office use. For the walls in each orientation, there are several construction layers due to thermal considerations. This explains why there are three

Table 3.2: Building materials in simulation

Structures	Material's name	Thickness(m)	Conductivity(W/mK)
Wall	1IN Stucco	0.0253	0.6918
	8IN Concrete HW	0.2033	1.7296
	Wall Insulation	0.0679	0.0432
Ground	MAT-CC05 8 HW CONCRETE	0.2032	1.311
Roof	Roof Membrane	0.0095	0.16
	Roof Insulation	0.1673	0.049
	Metal Decking	0.0015	45.006
Windows	Theoretical Glass [117]	0.003	0.0185

materials in the walls as described in Table 3.2. The roof is the same as the walls. The open/close time of the building and schedules of the inner equipments are carefully set as for normal office purposes in France.

During the simulation, the output is hourly damped. There are several output files in EnergyPlus, we extract the time series data mainly from the `.eso` file. Post-process of these data is required for further analysis, that is reformatting the data into the form required by the analyzing tools. The consumption target is district heating demand or total electricity consumption. Meanwhile, we take 25 variables as features. They are listed in the following categorizations.

- Day type, indicates if the current day is a holiday or normal working day
- Weather conditions
- Zone mean air temperatures
- Infiltration volume
- Heat gain through each window
- Heat gain through lights
- Heat gain from people
- Zone internal total heat gain

For readers to understand what the consumption profile looks like, we take the target, electricity consumption in the whole month of November, as an example, and draw it in Figure 3.4. Intuitively, the hourly consumption varies periodically. In the middle of one day, around 12 o'clock, the electricity requirement reaches maximum, while at night it is at the lowest level. We also see that, at the weekends and holidays (11^{st} Nov), there is a low energy demand.

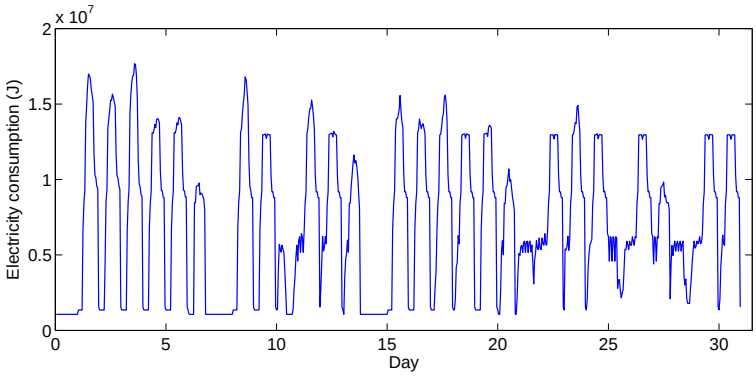

Figure 3.4: Hourly electricity consumptions of the single building in the first simulating month — November.

3.3.3 Simulating multiple buildings

In order to generate data for more buildings, we developed an interface to automatically control the simulation process. We suppose that all buildings are of the same type, specifically, for office use and have analogous characteristics. In our approach, the previously used input file is divided into two parts. The first part is called the alterable part, containing the parameters which are probably different for each building, such as structure characteristics, location, weather conditions, number of occupants, etc. Their values are obtained by stochastic methods, but should be in a reasonable domain. The second is the stable part where the parameters for each building are always the same, for instance, the schedules of inner electrical plants which each building shares since they are all for the same office use. When a new building is required, we first update the alterable part to make it specific to this building, then combine it with the stable part to create the final input file for EnergyPlus. In Chapter 4, we will analyze multiple buildings and require that all generated output data for each building must be put into

a single output file. We name it as `output.txt`. After successfully simulating one building, the program goes back to update the alterable part for simulating the next one. This process is repeated until the pre-defined number of buildings is finished. The whole process is shown in Figure 3.5.

To have an overview of the multiple buildings' consumption, we randomly choose one building and draw its consumption together with the consumption of the first building (simulated in 3.4) in Figure 3.6. We can see that the basic trends of these two waves are the same, but at some peak points, they are very different in value.

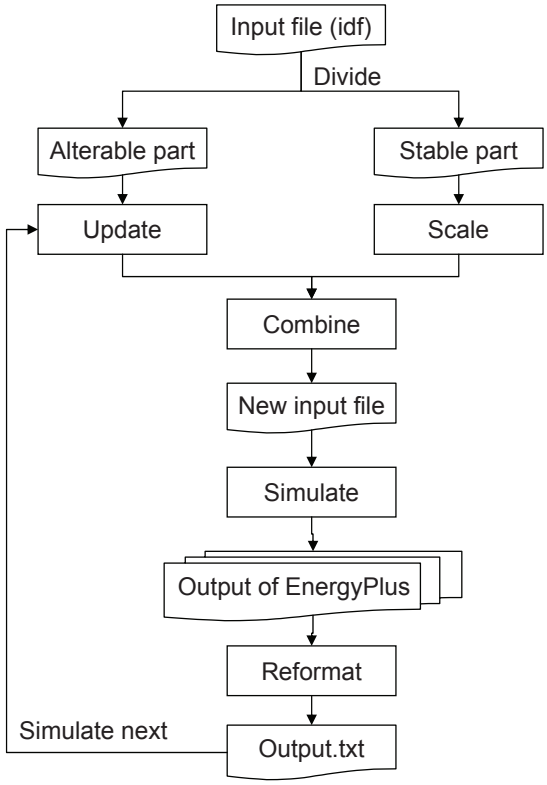

Figure 3.5: Flow chart of generating energy consumption data of multiple buildings

3.4 Discussion

Sufficient and precise consumption data is important for model evaluation. Unfortunately, it is difficult to obtain in practice. Simulation is a popular approach in the academic community. Since weather conditions are available and elaborate calibration

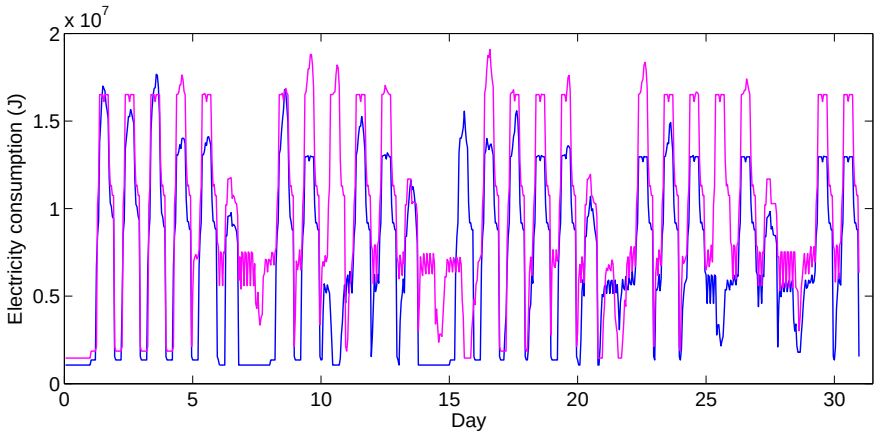

Figure 3.6: Hourly electricity consumptions of two buildings in November, distinguished by two colors.

can guarantee the accuracy, we also choose this approach in our work. When preparing the inputs of EnergyPlus, we carefully set the parameters according to real situations in France, in order to make the simulated building more like the real one. However, the real consumption data is still what we are looking for. Ideally, someone would continue taking measurements in some real buildings and maintain a database recording these data over a long period. This would be quite useful in the long run.

In the above multiple-building generation, all of the buildings are for office use, so their consumption might be similar. With our approach, it is quite easy to simulate totally different buildings, such as residential, commercial, or teaching buildings, which could vary from small spaces to large estates. The sub-level components can also be recorded in the simulation. However, since our aim is to develop high performance artificial intelligence models, there is no need to simulate all of these building types in this thesis.

4

Applications of Building Energy Analysis

In this chapter, we apply artificial intelligence models in building energy analysis. There are two main applications, one is predicting energy consumption, the other is to perform faulty consumption detection and diagnosis. Both of them are crucial for energy conservation in building design, retrofit and operation.

In the first application, we will investigate the performance of the SVR model in the prediction of energy consumption in the unknown future based on the historical behavior. Then we try to extract models from multiple buildings' performance and carry out the prediction of the consumption for a new building. Two types of energy, electricity consumption and heating demand, will be used as the targets in the experiments. Furthermore, we will profoundly test the robustness of this model by considering various situations and try to find out in which circumstance the better performance is achieved. For this purpose, we design three sets of experiments which are different in the dataset selection, and then analyze the trend of the model performance.

In the second application, we apply a classification model to detect faults in the building systems. We show the performance of the model in experiments and then propose a new approach to point out the source of the faults.

Section 4.1 introduces some practical issues while applying SVM model. Section 4.2 and Section 4.3 will present the details of the two applications, prediction and fault detection respectively. In the end, Section 4.4 gives conclusions and discussions.

4.1 Practical issues

This section will discuss some important practical issues of applying SVR in the building energy prediction. Firstly, we introduce the steps of the operation, then we show how to pre-process the datasets, and lastly we present the model selection.

4.1.1 Operation flow

In supervised learning theory, the experiments can be roughly divided into two steps, training and predicting. Accordingly, in order to evaluate the model, the dataset is divided into two sets, one is used for training, we call it training set, and the other is used for predicting, named as testing set. A decision model is obtained in the training step based on the training set to indicate the dependence of the target on the features. In the predicting step, the derived model is applied on the testing set to predict the target values with regard to new features. By comparing the predicted target with the real one in the testing set through some statistical methods, it is possible to evaluate the prediction performance of the model.

The above steps which are necessary for our experiments are shown in Figure 4.1. In Section 3.3, we have generated the historical consumption data in the output file `output.txt`. It then becomes the input of this analyzing process. After dividing this dataset, we pre-process the two pieces of data (will be introduced in Section 4.1.3) and format them as training and testing sets. Before training the model, we have to select the right model components and set the parameters to appropriate values. How to do these things will be introduced in Section 4.1.4.

4.1.2 Experimental environment

The hardware environment in the experiments is a workstation which has $8 * 2.5$GHz CPU, 1333MHz FSB and 4G memory. The operating system is Linux with kernel version 2.6.

There are many SVM implementations both academically and commercially available in present. The web site [116] maintains a list of these softwares. In the research community, SVM-Light [117] and Libsvm [118] are two widely used tools. They are well designed, efficient and have implemented full functions of SVM such as regression, classification, distribution estimation. They also provide elaborate description of the

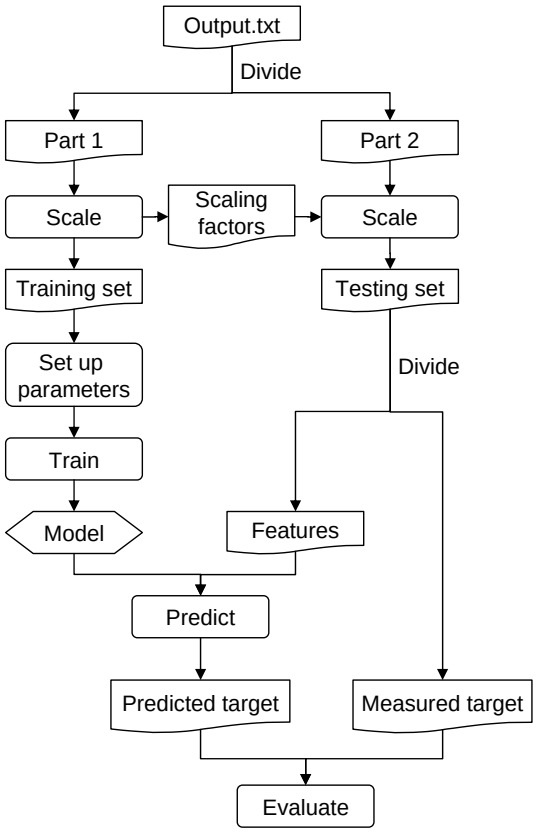

Figure 4.1: Flow chart of a learning process.

intrinsic mechanism and operating specifications. They are easy to learn and use. We choose Libsvm in our experiments since it was developed later than SVM-Light and it redesigned the superior features of this ancestor. It is intended to be a library and provides an interface to make it be easily integrated into other toolbox, such as Matlab, R, Weka, etc. Its implementation is based on the sequential minimal optimization quadratic problem solver.

4.1.3 Pre-processing data

Before training SVR, we need to scale the values linearly into a small range in order to avoid numerical problems in the training procedure. Here we chose the range $[0, 1]$. Suppose in the original training set, the value of i^{th} dimension of sample i is v_{ij}, then

the scaled value is:

$$\vec{v}_{ij} = \frac{v_{ij} - \min\{v_{kj}|k = 1, 2, ..., l\}}{\max\{v_{kj}|k = 1, 2, ..., l\} - \min\{v_{kj}|k = 1, 2, ..., l\}} \tag{4.1}$$

The target is also scaled into the same range. Moreover, the testing data should be scaled with the same scaling function as for the training set. Therefore, it is possible for the scaled testing values to be in a different range from that of the training values.

Since Libsvm is selected as the SVR implementation, the training data should be transformed into the format as required by this tool. Suppose there are 24 features, each sample should be in the following format:

T	$1 : v_1$	$2 : v_2$	$3 : v_3$	...	$24 : v_{24}$

where T is the target value and v_x is the value of xth feature.

4.1.4 Model selection

In our experiments, we select RBF as the kernel to train SVR models. Compared to other kernels, RBF is easier to use and very good in solving non-linear problems. The parameters needed are C, ε and γ. The first two are for SVR and last one is for RBF kernel. Choosing optimal values for these parameters is crucial in training a high performance model. The best parameters should have the ability to well predict on unknown data without causing over-fitting problem. In other words, the model should have high generalization ability as well as performing well on training set. For this purpose, the estimation of γ is solved by $\gamma = \sum_{i,j=1}^{l}(\left\|x_i - x_j\right\|^2)$ as proposed in [119], where x_i and x_j are the feature values of i^{th} and j^{th} samples. The SVR parameters C and ε are solved by stepwise 5-fold cross validation on randomly selected subset of the training data. The initial searching spaces are $\{2^{-3}, 2^{-2}, ..., 2^{8}\}$ and $\{2^{-10}, 2^{-9}, ..., 2^{-5}\}$ for C and ε respectively.

Let us use Table 4.1 to show how the 5-fold cross validation works. Firstly we split the samples into five pieces uniformly. Given a specific (C, ε) pair, we train and test the model five times on the pieces as the above table shows, and choose the best performance from {r_1, r_2, r_3, r_4, r_5} as the final result for this pair. Then we go to the next pair of (C, ε) and run the same process. We repeat this procedure until all of

Table 4.1: 5-fold cross validation

Dataset:	p_1	p_2	p_3	p_4	p_5

Steps	Train on	Test on	Result
1	p_2, p_3, p_4, p_5	p_1	r_1
2	p_1, p_3, p_4, p_5	p_2	r_2
3	p_1, p_2, p_4, p_5	p_3	r_3
4	p_1, p_2, p_3, p_5	p_4	r_4
5	p_1, p_2, p_3, p_4	p_5	r_5

the possible pairs are evaluated. Finally we are able to pick up the optimal pair which produces the best accuracy. This is exactly what we need in the later model training.

4.1.5 Model performance evaluation

Two methods are used to evaluate the model performance. One is the Mean Squared Error (MSE) which gives the average deviation of the predicted values to the real ones. The lower the MSE, the better the prediction performance. Suppose there are l testing samples, the decision function is $f(x)$ and the measured target is y, the MSE is defined as:

$$\text{MSE} = \frac{1}{l}\sum_{i=1}^{l}(f(x_i) - y_i)^2 \tag{4.2}$$

The other method is the Squared Correlation Coefficient (SCC) which lies in the range $[0, 1]$ and gives the ratio of successfully predicted number of target values on the total number of target values, i.e. how certain the predicted values are compared to the measured one. The higher the SCC, the stronger the prediction ability. It is defined as:

$$\text{SCC} = \frac{(l\sum_{i=1}^{l} f(x_i)y_i - \sum_{i=1}^{l} f(x_i)\sum_{i=1}^{l} y_i)^2}{(l\sum_{i=1}^{l} f(x_i)^2 - (\sum_{i=1}^{l} f(x_i))^2)(l\sum_{i=1}^{l} y_i^2 - (\sum_{i=1}^{l} y_i)^2)} \tag{4.3}$$

4.2 SVR in the energy prediction

In this section, three sets of experiments are going to be performed. The first one is aiming at testing how is the performance of SVR in the prediction of a single build-

ing's energy consumption, including two main types of energy: heating demand and electricity. In order to investigate the model robustness, we design the second set of experiments where the models are trained on different period of historical consumption, and the influence of training data size is fully studied. This set of experiments is still based on a single office building. Later, in the third experiment, we train the SVR model in multiple buildings and test how is the performance of this model in the prediction of a completely new building. The data used for model training and testing is simulated in EnergyPlus as described in Chapter 3. The models in the three sets of experiments are trained on different types of datasets.

4.2.1 Predict the energy consumption of a single building

In winter, the buildings in France consume large amounts of district heating. In the first model development, the district heating consumption data is gathered hourly from a single building in the heating season (from November 1^{st} to March 31^{st}). There are 3624 samples with 24 features in the final dataset. We take the samples of the last two days as testing set, and the rest of them for training use. Therefore the number of training samples is 3576 and that of testing samples is 48. The parameters of SVR model are set as $C = 16$, $\gamma = 0.7533$ and $\varepsilon = 0.01$ by the method described in 4.1.4. The final result of the prediction is plotted in Figure 4.2 where the predicted and actual targets are shown together. We can see that the two curves fit very well, which means that the model has a very good generalization performance. Actually, the number of support vectors (SVs) is 2229 and MSE and SCC are $2.3e-3$ and 0.927918 respectively. We also can see the good performance from these evaluation methods.

Another important type of energy consumed by buildings is electricity. We also train the SVR model to predict this type of energy. To be different from the previous experiment, this time we choose the consumption data through one whole year, and the testing dataset (contains 48 samples) is randomly selected from the global dataset. The features are the same as in the first experiment. The number of training samples is 8712. The parameters are set as $C = 16$, $\gamma = 0.3043$ and $\varepsilon = 0.01$. The result show that there are 2126 support vectors, MSE is $5.27e-4$, and SCC is 0.959905. The model performs even better in this case. The measured and predicted values are plotted in Figure 4.3. In most cases the two curves fit well except the hours 18 and 27.

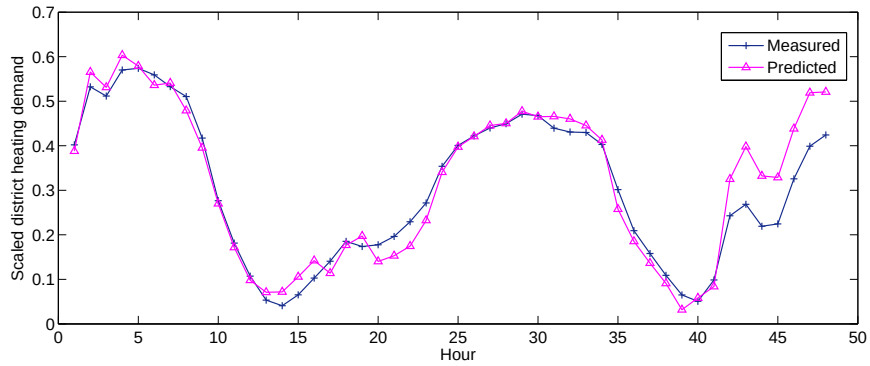

Figure 4.2: Measured and predicted district heating demand in heating season.

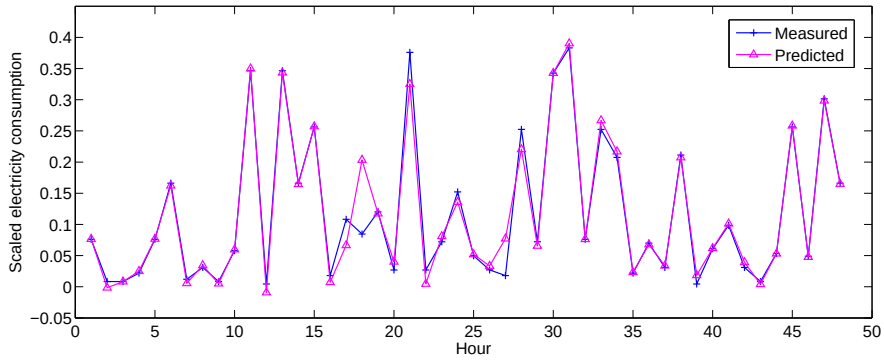

Figure 4.3: Measured and predicted electricity consumption in randomly selected 48 hours.

4.2.2 Further more performance test

The size of training dataset has certain influence on the model performance. In the previous sub-section, the first model is trained on five months' data while the second is trained on one year's data, and we see that the second prediction fits better than the first one. However, in the second training process, the testing data is drawn randomly from the whole dataset, this indicates that the characteristic of the testing samples is quite close to that of the training samples. In contrast, in the first training process, since the testing and the training datasets are for different days, they are more different in characteristics. In order to sufficiently study how is the model performance in this application, we design the following experiments which vary in training and testing datasets.

Table 4.2: The consumption period for the training and testing datasets.

Train on data	Test on data
Jan	Mar, Apr, ..., Dec
Jan - Apr	Jun, Jul, ..., Dec
Jan - Aug	Sep, Oct, Nov, Dec

Three models are trained. The training the testing datasets are described in Table 4.2. We choose electricity as the target since we have this data for a whole year.

The first model is trained on the consumption of only one month (January), then we use the derived model to predict the consumptions in other months (from February to December). To see the performance of the prediction on these months which are totally different from the training month, we select four months to draw the predicted errors which is defined as measured values minus predicted ones. They are March, May, July and September, as shown in Figure 4.4. The MSE and SCC of all of the months (from March to December) are drawn in Figure 4.7 and 4.8 respectively. We can see an obvious convex trend of the prediction accuracy, i.e., March, November and December hold the best performance while July is the worst. This is due to the reason that the previous three months are in winter as the same as January be. In other words, the characteristics of the first three months, such as weather conditions, HVAC service, are similar as that in January. In contrast, July is in summer, it is the most different month from January.

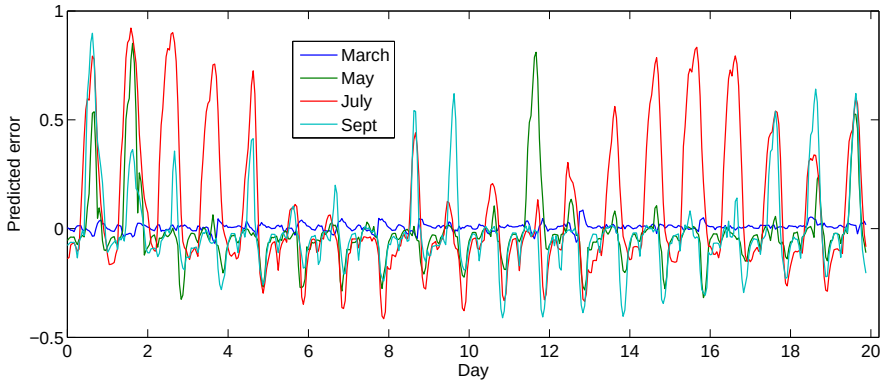

Figure 4.4: The prediction error of the model on the consumption of March, May, July and September. The model is trained on the data of January.

The second model is trained on the consumption from January to April which is three months more than the first training set. The prediction is tested on each month from May to December. We also draw four months' prediction errors in order for the readers to have a global view of the model performance, as shown in Figures 4.5. The MSE and SCC are also put in Figures 4.7 and 4.8. We still can see that in July the model has the worst performance while in December it has the best one. Another important thing is that from June to October, the present model performance is better than the previous one whose training data contains only January. This case indicates that sufficient data is important for training a good model.

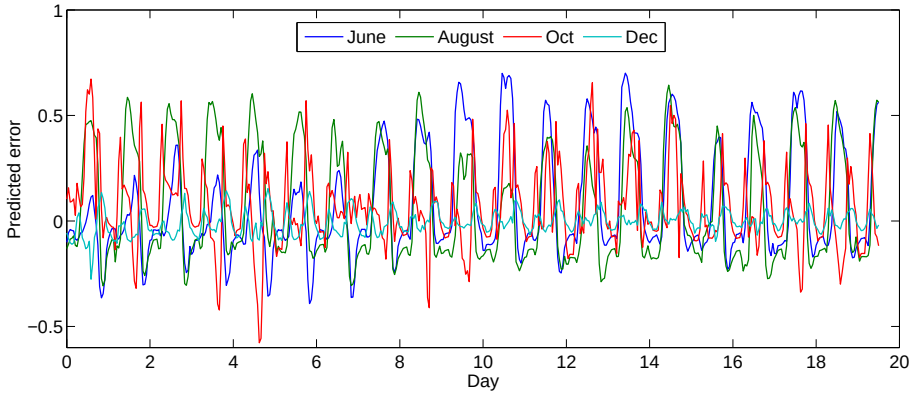

Figure 4.5: The prediction error of the model on the consumption of June, August, October and December. The model is trained on the data from January to April.

In the third model training, we even enlarge the training data period by using the consumption from January to August, even four more months than the previous one. Then the model is applied on each month between September and December. Again, we draw the four months' results in Figure 4.6, and put MSE and SCC in Figure 4.7 and 4.8 respectively. We can see that, in September and December, the present prediction accuracy is better than that of the previous two models. This indicates that the more training data the better the model performance. However, this is not always true. At least in November, the model is not as fine as the first one. Note that the first model has only one month data for training which is far less than the present one.

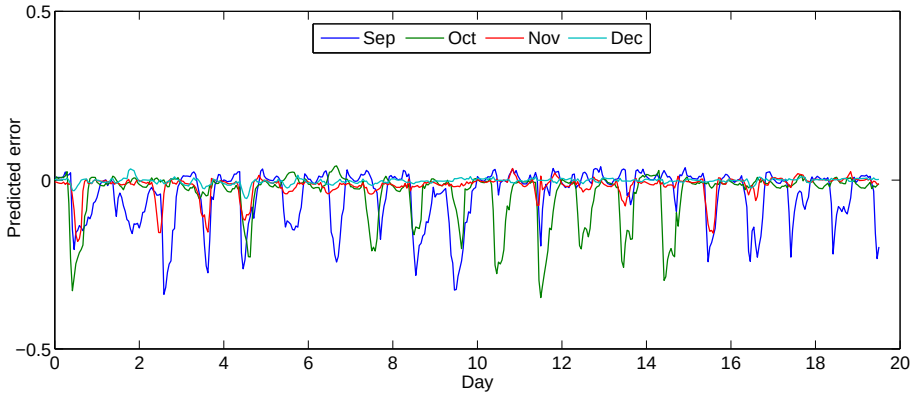

Figure 4.6: The prediction error of the model on the consumption of September, October, November and December. The model is trained on the data from January to August.

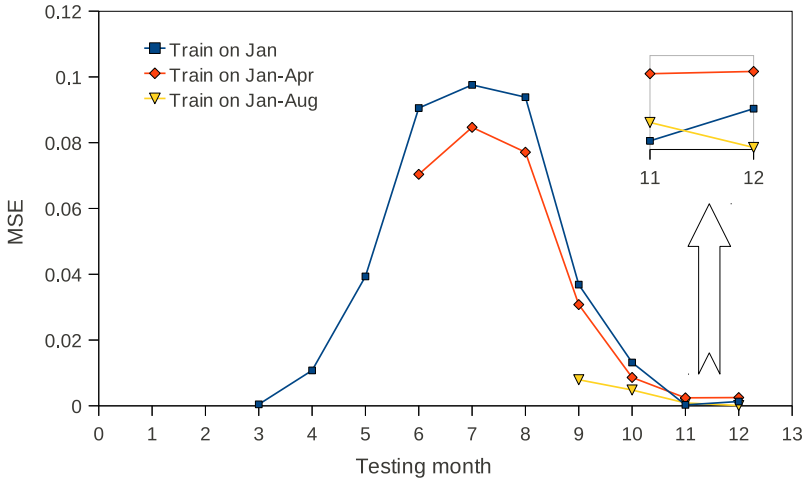

Figure 4.7: MSE of the three models on the designed testing months.

4.2.3 Multiple buildings

The above learning processes are based on the energy consumption of a single building and the evaluation of the model is to predict the unknown future in the same building. In practice, it is quite useful to predict how much energy is required in a completely new building. Therefore, in the second experiment, we tried to learn a model based on the energy data where the building structures are involved. That is to say, we trained a model from the consumption behaviors of several buildings, then applied the model to predict the behavior of a totally different building. In this experiment, one hundred

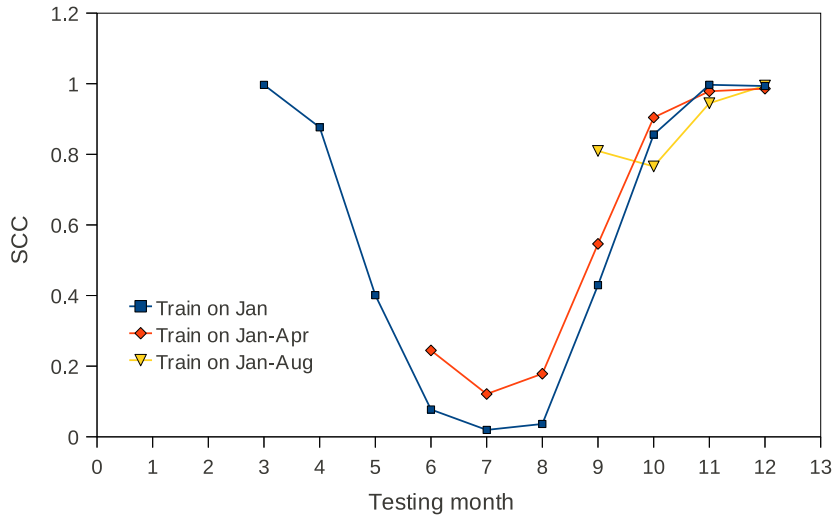

Figure 4.8: SCC of the three models on the designed testing months.

buildings are simulated in the heating season. They are in the same weather conditions but have different properties, such as different orientations, volumes, people densities and fenestration. We chose the data of the first 99 buildings as the training set and data of the last building as the testing set. The number of features is 28. The number of training samples is 358776, and the number of testing samples is 3624. The model parameters are set as $C = 4$, $\gamma = 0.3179$ and $\varepsilon = 0.01$. In the training step, the number of SVs is 27501, while in the predicting step, MSE is $5.01e-5$ and SCC is 0.997639. The predicted and measured values on the first 100 samples in test dataset are plotted in Figure 4.9. The results show us that SVR has a very good prediction performance in building energy consumption when building diversity is taken into account. It provides us the possibility to predict the energy performance of a building for designing as well as for retrofitting.

4.3 Fault detection and diagnosis

The early knowing of the abnormal performance of interior electric equipments is important for building operation and energy conservation. This work aims at fault detection and diagnosis (FDD) of building energy consumption which is mainly due to performance degradation, poorly maintenance or improper operation of the installed electric

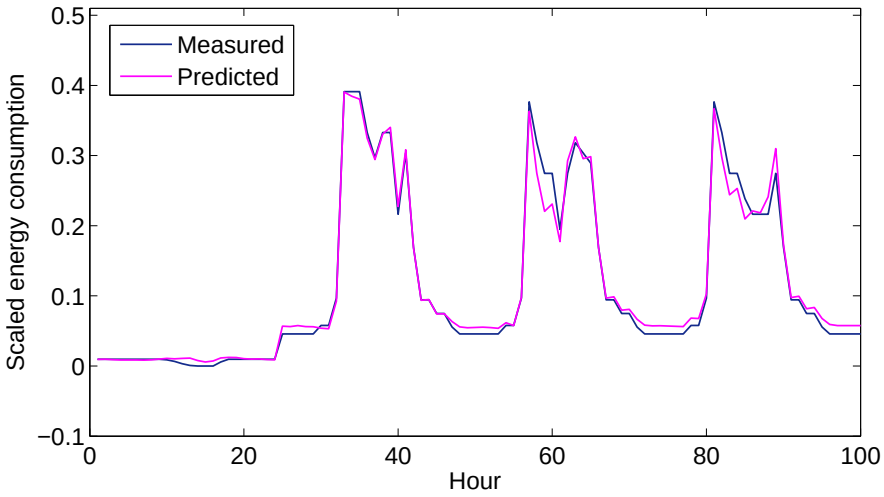

Figure 4.9: Measured and predicted electricity consumption for a totally new building, the model is trained on 99 buildings.

systems.

These kind of non-abrupt faults are difficult to monitor and even more difficult to isolate. FDD thus attracts lots of attention in recent decades. Previous work mainly focuses on particular electric systems, such as Air-Handling Units (AHU) [56, 120, 120–124], air-conditioning systems [125–128], vapor compression system [129–131], HVAC or the whole building level analysis [8, 132]. Many advanced methods or techniques are utilized, such as ANNs [123, 130], multi-step fuzzy model [122], transient pattern analysis [133], SVM [8], residual analysis [56], statistic method [124, 126, 127], rule-based method [129, 134], hybrid method [125, 128], etc.

For example, Lee et al. [120] introduced an ANN for FDD in an AHU. They discussed 11 faults from fan failure to sensor failure in the system, and demonstrated that the recovered estimate of the supply air temperature can be used as a feedback to the control loop and can bring the supply air temperature back to the setpoint. Lee et al. [56] proposed general regression neural networks to generate estimates of sensor values and control signals for AHUs, then faults were determined when residuals exceeded predefined thresholds. House et al. [121] compared five models in detecting and diagnosing seven faults of a AHU system, i.e., ANN, nearest neighbor, nearest prototype classifier, rule-based classifier and Bayes classifier. Qin and Wang [125] surveyed 10 main faults in the variable air volume air-conditioning systems. A hybrid method, which includes

expert rules, performance indexes and statistical process control models, was proposed to implement FDD for these faults. Liang and Du [8] used residual analysis method to detect three faults in a HVAC system, i.e., recirculation damper stuck, cooling coil fouling/block and supply fan speed decreasing. A multi-layer SVM classifier was developed to diagnose these three kind of faults. Liu et al. [132] investigated seven faults in the whole building level and detected them by engineering method.

In this work, we introduce the Recursive Deterministic Perceptron (RDP) neural network, which is an effective classification model, to solve FDD in the whole building level. The model is derived from historical profiles and contains the knowledge of normal and faulty behavior. The used data includes certain real-time physical variables that are measured by sensors and meters installed throughout the building. This model shows very high detection ability in the experiments.

We also propose a new method to diagnose the detected faults. Given a sample of faulty consumption, this method can point out which electric equipment causes this fault. Furthermore, it can list all of the possible causes in the probability descending order, making us easy to deal with more broaden problems. Our method is based on the evaluation of several RDP models, each of which is designed to be able to detect unique device fault. The experimental result shows that this method can diagnose faults correctly.

Next, RDP neural network is briefly introduced in Section 4.3.1. Then it is trained and applied to detect faults in building energy consumption in Section 4.3.2 and further used to diagnose faults in Section 4.3.3.

4.3.1 RDP model

RDP feed-forward multi-layer neural network can solve any two-class classification problems and the convergence is always guaranteed [135]. It is a multi-layer generalization of the single layer perceptron topology and essentially retains the ability to deal with non-linearly separable sets.

The construction of RDP does not require pre-defined parameters since they are automatically generated. The basic idea is to augment the dimension of the input vector by addition of intermediate neurons (INs). These INs are added progressively at each time step, obtained by selecting a subset of points from the augmented input vectors. Selection of the subset is done so that it is linearly separable from the subset

containing the rest of the augmented input points. Therefore, each IN is obtained using linear separation methods. Different choice of the linear separability testing method would influence the model convergence time and topology size [136, 137]. The algorithm stops when the two classes become linearly separable at a higher dimension. Hence, RDP Neural Network reports completely correct decision boundary on the training dataset, and the knowledge extracted can be expressed as a finite union of open poly topes [138].

Definition 1 *A recursive deterministic perceptron (RDP) P on $\mathbb{R}^d$ is defined as a sequence $[(\boldsymbol{w}_0, t_0, a_0, b_0), ..., (\boldsymbol{w}_n, t_n, a_n, b_n)]$ such that $\boldsymbol{w}_i \in \mathbb{R}^{d+i}$ and $t_i, a_i, b_i \in \mathbb{R}$, and $a_i < b_i$ for $0 \leq i \leq n$*

- *$(\boldsymbol{w}_0, t_0, a_0, b_0)$ for $0 \leq i \leq n$ is termed as Intermediate Neuron(IN) of the RDP P*
- *Height(P) corresponds to the number of INs in P*
- *$P(i, j)$ is the RDP $[(\boldsymbol{w}_i, t_i, a_i, b_i), ..., (\boldsymbol{w}_j, t_j, a_j, b_j)]$ for $0 \leq i \leq j \leq height(P) - 1$*

Definition 2 *(Semantic of RDP). Let P be a RDP on $\mathbb{R}^d$. The function $\mathcal{F}(P)$ is defined in $\mathbb{R}^d$, such that:*

if height(P) = 1 then :

$$\mathcal{F}(P)(\boldsymbol{y}) = \begin{cases} a, & \text{if } \boldsymbol{w}^T \boldsymbol{y} + t < 0 \\ b, & \text{if } \boldsymbol{w}^T \boldsymbol{y} + t > 0 \end{cases}$$

and if height(P) > 1 then :

$$\mathcal{F}(P)(\boldsymbol{y}) = \begin{cases} \mathcal{F}(P(1,n))(Adj(\boldsymbol{y}, a_0)), & \text{if } \boldsymbol{w}_0^T \boldsymbol{y} + t_0 < 0 \\ \mathcal{F}(P(1,n))(Adj(\boldsymbol{y}, b_0)), & \text{if } \boldsymbol{w}_0^T \boldsymbol{y} + t_0 > 0 \end{cases}$$

Definition 3 *Let X, Y be two subsets of $\mathbb{R}^d$ and let P be a RDP on $\mathbb{R}^d$. Then, X and Y are linearly separable by P if $(\forall \boldsymbol{x} \in X, \mathcal{F}(P)(\boldsymbol{y}) = c_1)$ and $(\forall \boldsymbol{y} \in Y, \mathcal{F}(P)(\boldsymbol{y}) = c_2)$ where $\{c_1, c_2\} = \{a_n, b_n\}$, denoted by $X \|_P Y$. The set $\mathcal{S}(P) = \{\boldsymbol{x} \in \mathbb{R}^d | \mathcal{F}(P)(\boldsymbol{x}) = b_n\}$ is termed as the decision region of P and is the knowledge embedded in P.*

There are three methods for constructing RDP neural networks: batch, incremental and modular [138]. The one adopted for the FDD in our study is the incremental RDP. In this progressive learning, the network trains a single data point at a time. This is

highly parallel to the physical reality since as the operating conditions vary with time, the model adapts itself for fault identification. As learning proceeds, the network first tries to classify the data point within the existing framework. If the classification is not possible, the knowledge is interpolated by adding a new IN without disturbing the previously obtained knowledge. The training continues further until all the remaining points are classified by the network model. The procedure of this algorithm is shown in Figure 4.10.

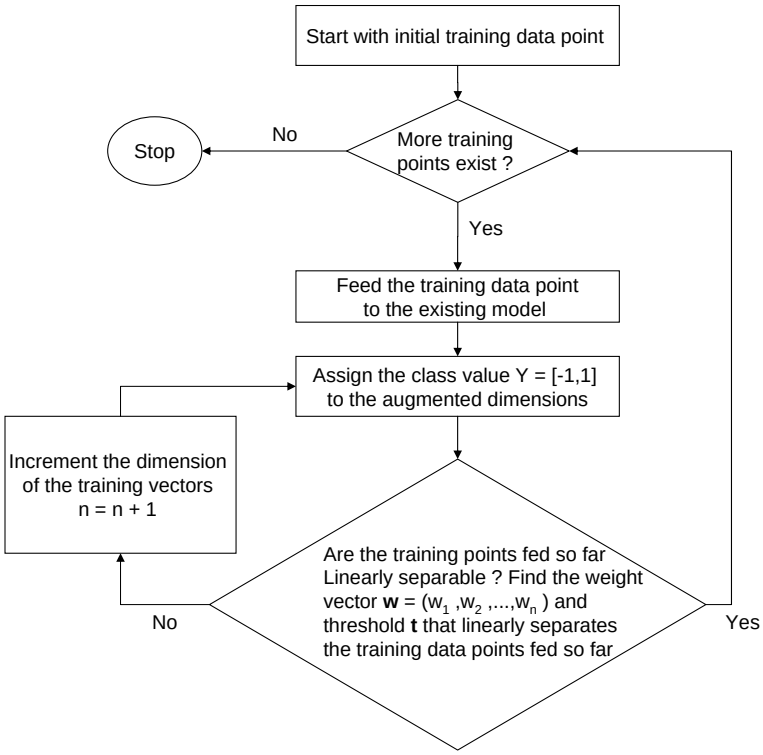

Figure 4.10: Flowchart of the incremental RDP Model

4.3.2 Building energy fault detection

This section deals with detection of building energy consumption faults using selected variables. RDP classification models have been developed and tested for buildings with various electric systems and distinct fault regimes. This has been done to analyze the applicability and robustness of the RDP models to detect faults in diverse system domains.

4.3.2.1 Introduce faults to the simulated building

In our experiments, the daily electricity consumption is considered as the energy form. An office building is simulated under EnergyPlus. The characteristics of this building are similar to the one discussed in Section 3.3.2, but equipped with a HVAC system. The designed heating type is Heat Pump while cooling type is Central Chiller. We introduce these electric devices into the building in order to verify whether or not the faults caused by them can be detected. The sub-systems which might cause detectable consumption errors are Lights, Fan, Coil, Pump and Chiller.

The run period for the simulation is one year. The recording is dumped daily. The output variables which are used for detection of abnormality in the building energy consumption are listed below. Their units are Joules.

1. Facility Electric Consumption, is the accumulated electric consumption of the building zones, the plant loops, the air loops, the interior as well as exterior electric equipment.

2. InteriorEquipment Electric Consumption, includes all of the interior equipments in all zones.

3. Cooling Electric Consumption, stands for the electric consumption for cooling purpose.

4. Fans Electricity Consumption, is the electric power input to the Fans.

5. Pumps Electricity Consumption, is the electric power input to the Heat Pumps.

6. Chiller Electric Consumption, is the electric power input to the Chiller.

The parameters of these electric equipments are set to default values in EnergyPlus, indicating normal usage. Then, one-year long simulation which represents normal condition is performed. After that, faults are introduced to this building, and once again, one-year long faulty consumption is simulated. The faults are introduced by changing performance parameters of equipments in order to reflect their performance degradation. We select six parameters of four equipments and decrease 35% of their performance. They are listed in Table 4.3. The first column lists equipments which might

cause detectable consumption errors. The second column gives the performance parameters of the corresponding equipment. The third column is the performance changes from normal usage to fault.

Table 4.3: The faults introduced to the building.

Equipment	Parameter	Change
Fan	Fan efficiency	$0.6 \rightarrow 0.39$
	Motor efficiency	$0.9 \rightarrow 0.59$
Coil	Design water flow rate	$0.0022 \rightarrow 0.003\ m^3/s$
Pump	Motor efficiency	$0.9 \rightarrow 0.59$
Chiller	Reference COP	$3 \rightarrow 1.95$

Although performance degradation is set to equipments, the building is not always in faulty usage. In reality, there are quite number of days that the equipments are not used frequently. In these cases, the consumption changes are not obvious and undetectable. Therefore, the whole year profiles with faults introduced can not be simply considered as faulty usage (or we say errors). Based on the generated normal and faulty datasets, we set a threshold to determine whether or not a sample is a real error. The threshold will determine the number of errors in the final datasets. Here we simply choose 1000J. If the consumption change in current day is less than 1000J, then this day is considered as in normal condition. In contrast, if higher than 1000J as the threshold, then the current day is considered as fault. In the final datasets, each normal sample is assigned with a label 1 while faulty samples are labeled 0.

To have an overview of consumption changes after introducing faults, Facility Electric Consumption is chosen to depict, as shown in Figure 4.11. We can see obvious consumption increasing in most of the days when faults are introduced.

4.3.2.2 Experiments and results

Two experiments are performed to test the ability of RDP model in fault detection. In the first, almost ten months' normal and ten months' faulty data are combined to train the model and the remaining two months combination data is used for testing. In the second, only six months' data is used for training and the remaining six months' data is used for testing. The number of samples of these datasets are shown in Table 4.4. The number of features is six for all of the datasets.

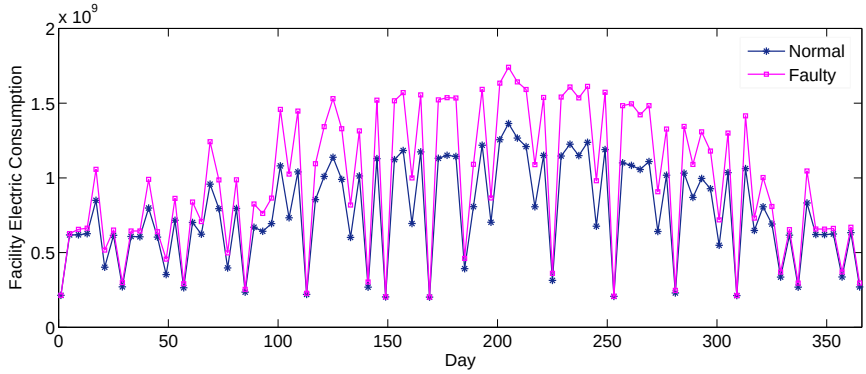

Figure 4.11: Normal and faulty Facility Electric Consumption in one year. The unit is J.

Table 4.4: Number of samples in the datasets.

Experiment 1		Experiment 2	
Training set	Testing set	Training set	Testing set
610	120	365	365

The experiments are performed on a dual-core computer with 2.4G*2 CPU, 2G memory and windows XP installed. The testing results are shown in Table 4.5. We can see that the prediction accuracy is very high, achieving 100% on both training datasets and 100%, 97.81% on two testing datasets respectively.

Table 4.5: Results of RDP model in two experiments.

	Experiment 1	Experiment 2
Time (s)	13.78	8
Model size	9	9
Accuracy on training set (%)	100	100
Accuracy on testing set (%)	100	97.81

4.3.3 Fault diagnosis

We propose a new method to identify the faulty equipment which causes the detected abnormal consumption. Considering the above building, suppose the possible sources of faults are Chiller, Coil, Fan and Pump, we denote them as E_1, E_2, E_3 and E_4 respectively. What we want is a procedure that can tell us which equipment works abnormally when given faulty consumption. A even better result is a list of equipments in the order of fault risk, since more than one equipment might be wrong simultaneously.

If such kind of list is provided, then we can easily know that which one is the most possible source of the fault and which one is the second, third, and so on.

Let us solve this problem. Four models are trained in order to predict errors caused by each single source, denoted by M_1, M_2, M_3 and M_4. One requirement is that each model can predict errors only caused by the corresponding source and it reports errors caused by other sources as normal situation. For instance, M_3 can only detect Fan (E_3) errors and would report errors caused by Chiller, Coil or Pump as normal consumption. If this requirement is satisfied, then we can use the procedure depicted in Figure 4.12 to isolate the faulty equipment. It works as follows. Given some faulty samples, which can be detected by RDP models as discussed in the above section, we input them to the four models and record their prediction accuracy, then the order of possible faulty equipment is exactly the descending order of these four accuracy. The reason is that, when the prediction accuracy of M_i is high, the input faulty samples are more like the training profiles of M_i, thus these faulty samples are more likely to be caused by the corresponding equipment E_i, rather than caused by other equipments.

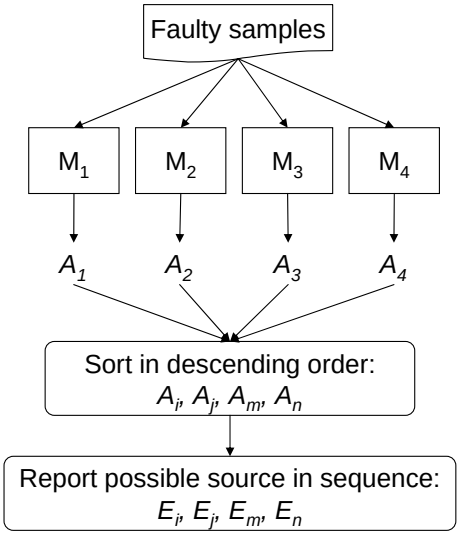

Figure 4.12: The flow chart of fault diagnosis. M_i is the i^{th} model, A_i is the prediction accuracy through the i^{th} model, E_i stands for the i^{th} equipment.

Now the problem is how to train these models to make them only sensitive to their corresponding equipments' degradation. The solution is simple. The only thing we need to do is to make the models predict errors caused by other sources as normal situation. To implement this idea, suppose we have four faulty datasets, each of which contains

one faulty equipment, then we create training dataset for each model as follows. We use an example to explain this procedure. Suppose we want to create training data for model M_1, we pick up the faulty samples from the faulty dataset which contains E_1 degradation, associate them with label 0 and put in the training dataset. Then we extract faulty samples from other three datasets, associate them with label 1 and put them together into the training set. This means in generating model M_1, we consider faults caused by E_2, E_3 and E_4 as normal samples. Finally, the model is trained on this dataset and is expected to be only sensitive to E_1 degradation.

One experiment is designed to test our method. Firstly we generate four faulty datasets under EnergyPlus to indicate four equipments' degradation, with the parameters setting as described in Table 4.3. Then we use the above method to prepare four RDP models. The data for model training is the first ten months' profiles. Then we implement the procedure as described in Figure 4.12 and verify that whether or not this method can report the right faulty equipment. Chiller is chosen as the example. The input faulty samples are the remaining two months' faulty consumption data when Chiller fault is introduced. The accuracy of the four models on this testing dataset is shown in Table 4.6. It is easy to know the sequence of the possible faulty equipments is $E_1 > E_3 > E_4 > E_2$, and the accuracy of E_1 is obviously higher than that of others. Therefore, we can easily judge that Chiller is the most likely source of the fault. This method reports exactly the right answer.

Table 4.6: RDP model accuracy in the diagnostic procedure with the example inputs.

A_1	A_2	A_3	A_4
94.12	1.96	11.76	5.88

4.4 Discussion

This chapter applies SVR in the prediction of building energy consumption. We demonstrate the performance of this model in extensive experiments, representing various situations, such as single/multiple buildings, electricity/district heating consumptions, small/large training sets, close/far away of the training and testing profiles, etc. SVR shows very high generalization ability and robustness in these tests. However, these tests still have shortages. Our data is obtained from simulation method, it is relatively

clean and regular, furthermore, our simulated building is relatively simple, thus the training data can not adequately reflect reality.

Later, RDP neural network model is used to detect and diagnose faults in the whole building level systems. We manually introduce faults into equipments and obtain one-year long abnormal consumption data. We train a RDP model on a dataset which includes 10 months normal and 10 months abnormal consumption. Then we successfully use this model to detect errors in the last two months' data. RDP model showed very high performance in the detection. The proposed diagnostic method can list the equipments in order, according to their possibilities of occurring faults. This ability may lead us to diagnose faults caused by more than one source. Even, we would be able to deal with the problem in which several equipments cause similar faults. In the future work, we may test different classification models in this application.

5

Model Optimization — Feature Selection

5.1 Introduction

In Chapter 1, we have stated that there are a great number of factors which probably impact on energy dynamics of a building. And in Chapter 4, when predicting one single buildings' energy profiles, we selected 24 features to train the model, including variables from weather conditions, energy profiles of each sub-level component, ventilation, water temperature, etc. While predicting multiple buildings' consumption, four more features which represent building structure characteristics were used. However, we do not guarantee that these features are the right choices, nor can we even say they are all useful. How to reasonably choose a subset of appropriate features to be used in model learning is one of the key issues in machine learning. On one hand, using different sets of features would probably change the performance of the models in accuracy and learning speed. On the other hand, the optimal set of features would make the predictive models more practical.

For the first time, we discuss how to select subsets of features for SVR applied to the prediction of building energy consumption. We present a heuristic approach for selecting subsets of features in this chapter, and systematically analyze how it will influence the model performance. The motivation is to develop a feature set that is simple enough and can be recorded easily in practice. The models are trained by SVR with different kernel methods based on three data sets. The feature selection (FS)

method is evaluated by comparing the models' performances before and after FS is performed.

The plan of this chapter is organized as follows. Section 5.2 presents related work. Section 5.3 discusses general FS methods and in particular the ones introduced in this work. Section 5.4 illustrates, with several numerical experiments, the robustness and efficiency of the proposed method. Finally, Section 5.5 gives the conclusion and discussion.

5.2 Related work

FS is a challenging subject and is widely studied in the machine learning community. PCA and KPCA are two broadly used methods in exploratory data analysis and training predictive models [139]. In a raw dataset, there would be correlations between variables. PCA aims at reducing these correlations and making variance of data as high as possible. It converts a set of possibly correlated features by using an orthogonal transformation into a set of uncorrelated features, which are called principal components. After the PCA processing, some new features are created and the total number of features will be reduced. KPCA is developed as an extension of PCA by involving kernel methods for extracting nonlinear principal components [140]. This allows us to obtain new features with higher-order correlations between original variables.

Factor analysis is similar to PCA which can be used to reduce dimensionality. It investigates whether a number of variables are linearly related to a lower number of unobservable variables (factors). The obtained interdependencies between original variables can be used to reduce the set of variables. Independent component analysis (ICA) is another feature extraction method. It is a powerful solution to the problem of blind signal separation. In this model, the original variables are linearly transformed into mixtures of some unknown latent variables which are statistically independent, so that these latent variables are called independent components of the original data. Different from PCA, which attempts to uncorrelate data, ICA aims at composing statistically independent features [141].

The above four methods have been used as data pre-processing methods for SVMs in various applications [142][143][144][145]. However, they are not the right choices for us. Our aim is to find the set of features which are not only optimal for learning

algorithm, but also reachable in practice. It means that we need to select features from the original set without developing new features.

Some FS methods specially designed for SVMs have been proposed. Weston et al. [146] reduced features by minimizing the radius-margin bound on the leave-one-out error via a gradient method. Fröhlic and Zell [147] incrementally chose features based on the regularized risk and a combination of backward elimination and exchange algorithm. Gold et al. [148] used a Bayesian approach, Automatic Relevance Determination (ARD), to select relevant features. In [149], Mangasarian and Wild proposed a mixed-integer algorithm which was considered to be straightforward and easily implementable. All of these methods focus on eliminating irrelevant features or improving generalization ability. However, they do not consider the feasibility of selected features in a specific application domain, such as predicting energy consumption. Furthermore, they were implemented only for classification problems.

To the best of our knowledge, there is little work concerning FS of building energy consumption with regard to machine learning methods. Most of the existing work derives models based on previously established sets of features. Madsen et al. [150] derived their continuous-time models on five variables, which are room air temperature, surface temperature, ambient dry bulb temperature, energy input from the electrical heaters and solar radiation on southern surface. Neto et al. [76] built their neural network based on the input of daily average values of dry bulb temperature, relative humidity, global solar radiation and diffuse solar radiation. Azadeh et al. [53] and Maia et al. [151] forecast electrical energy consumption through analyzing the varying inner targets without any contributory variable involved. Yokoyama et al. [45] considered only two features, air temperature and relative humidity in their neural network model. Tso et al. [152] used more than 15 features in their assessment of traditional regression analysis, decision tree and neural networks. Similar approaches can be found in [47, 77, 78, 153].

5.3 The algorithm

FS aims at selecting the most useful feature set to establish a good predictor for the concerned learning algorithm. The irrelevant and unimportant features are discarded in order to reduce the dimensionality. Several advantages will be achieved if we wisely

select the best subset of features. The first is the simplification of the calculation while keeping the dimensionality minimized, which could contribute to avoiding the curse problem of dimensionality. The second is the possible improvement of accuracy of the developed model. The third is the improved interpretability of the models. The last is the feasibility of obtaining accurate feature samples, especially for some time series problems in practice.

The methods for FS can be classified into three categories: filter, wrapper and imbedded method. The filter method aims at ranking features with some correlation or mutual information criteria, and selecting the features with the highest ranks. It can be regarded as a pre-processing step without the model training algorithm involved, which means it is independent of the predictor designed in the learning step. The wrapper method assesses the subsets of features according to the degree of accuracy they contribute to a given predictor. The embedded method evaluates the usefulness of feature sets in the same way as the wrapper method does, while the selection proceeds directly in the training process and can thus avoid multiple training for each candidate subset. More detailed information can be found in the work [154].

Two FS methods will be used in our approach to pre-process raw data before model training. The first one ranks the features individually by correlation coefficient between each feature and the target. We use CC to stand for this method. The correlation coefficient between two vectors is defined as:

$$\text{Correlation}(f) = \frac{N\sum XY - (\sum X)(\sum Y)}{\sqrt{[N\sum X^2 - (\sum X)^2][N\sum Y^2 - (\sum Y)^2]}} \tag{5.1}$$

The other method is called Regression, Gradient guided feature Selection (RGS), which is developed by A. Navot et al. in the application of brain neural activities [155]. We choose this method since it is designed specially for regression and has shown competitive ability to handle a complicated dependency of the target function on groups of features. The basic idea is to assign a weight to each feature and evaluate the weight vectors of all the features simultaneously by gradient ascent. The non-linear function K-Nearest-Neighbor (KNN) is applied as the predictor to evaluate the dependency of the target on the features. The estimated target of sample x under KNN is defined as:

$$\hat{f_w}(x) = \frac{1}{Z}\sum_{x' \in N(x)} f(x')e^{-d(x,x')/\beta} \tag{5.2}$$

where $N(x)$ is the set of K nearest neighbors of sample x. $d(x, x') = \sum_{i=1}^{n}(x'_i - x_i)^2 w_i^2$ is the distance between sample x and one of its nearest neighbors x', n is the number of features, w is the weight vector and w_i is the specific weight assigned to i^{th} feature. $Z = \sum_{x' \in N(x)} e^{-d(x',x)/\beta}$ is a normalization factor and β is a Gaussian decay factor. Then the optimal w can be found by maximize the following evaluation function:

$$e(w) = -\frac{1}{2}\sum_{x \in S}(f(x) - \hat{f}_w(x))^2 \tag{5.3}$$

where S is the samples for model training. Since $e(w)$ is smooth almost everywhere in a continuous domain, one can solve the extremum seeking problem by gradient ascent method. More details can be found in [155].

5.4 Experiments and results

In this section, we will present how our method is implemented and test it with extensive experiments.

5.4.1 Description of the raw feature set

We use the data generated in Chapter 3. For a single building, the hourly electric demands, together with hourly profiles of 23 features, are recorded through one year. Table 5.1 shows the recorded features and their units. It is known that building structures like room space, wall thickness and windows area, play important roles in the total energy consumption of a building. However, for one particular building, these variables have constant values through the simulation period, which means they do not contribute to SVR model learning. Therefore, it is practical to discard these variables in this set without losing accuracy in the model. Later, in the data of multiple buildings, we will take these factors into consideration. Concerning the data analyzing process, the data for model training is the first 10 months of one year's consumption (from January 1^{st} to October 31^{st}) and for model testing is the remaining two months (from November 1^{st} to December 31^{st}).

Since people do not usually work at weekends and holidays, the energy requirement on these days is quite low, compared to normal working days. This means weekends and holidays have totally different energy behaviors from working days. Take the 56th

Table 5.1: The 23 features for the model training and testing on one building's consumption.

Features	Unit
Outdoor Dry Bulb	C
Outdoor Relative Humidity	%
Wind Speed	m/s
Direct Solar	W/m^2
Ground Temperature	C
Outdoor Air Density	kg/m^3
Water Mains Temperature	C
Zone Total Internal Total Heat Gain	J
People Number Of Occupants	-
People Total Heat Gain	J
Lights Total Heat Gain	J
Electric Equipment Total Heat Gain	J
Window Heat Gain for each wall	W
Window Heat Loss for each wall	W
Zone Mean Air Temperature	C
Zone Infiltration Volume	m^3
District Heating Outlet Temp	C

day as an example, it is a Saturday, the energy consumption for that day is 0.18, compared to other working days which have a normal consumption of more than 4, it can thus be safely ignored. It is proved that if we distinguish these two types of days, when training the model by predictive models like neural networks, considerable performance improvements could be achieved [7, 71]. Therefore, to simplify the model in our practice, we only select the consumption data of working days for use in model training and testing. Consequently, the number of samples for training is 5064 and for testing is 1008.

5.4.2 Implementation

In this part, we experimentally analyze our approach to select the best subset of features for training statistical models on building energy consumption data. Since the number of features does not have significant effect on the computational cost of SVM training, we put our focus primarily on the following two aspects, which are also two evaluation criteria of our method. The first is that the selected features should be potentially

the most important ones to the predictor. In other words, the model generalization error should still be acceptable after FS. For this purpose, we have to involve some FS algorithms in the object and choose the features with highest rankings or scores. The second is to make sure that the selected features can be easily obtained in practice. Concerning the energy data, the values of the chosen features for each observation can normally be collected from measurements, surveys, related documents like building plans and so on. However in practice, the efficient and accurate data is difficult to obtain, therefore, reducing necessary features is always welcome.

The two methods RGS and CC described in Section 5.3 are applied to evaluate the usefulness of features. The object data set is the previous consumptions of working days. The scores for each feature are listed in Table 5.2 in columns 2 and 3. We can see that even the same feature could probably have totally different scores under the evaluation of two FS algorithms. For example, the outdoor dry bulb temperature is the most important feature under the judgment of RGS, while on the contrary, it is almost useless according to CC ranking method. As experimental results have shown, the features with the highest scores under RGS are generally more useful than those with the highest ranks according to CC. This indicates that RGS method is more applicable to SVR than CC method. However, since the feature subsets with low scores are possibly still useful for the learning algorithms [154], we take both RGS and CC into consideration while choosing the features.

The weather data can be recorded on site or gathered from meteorological department. We keep two weather features that have the highest scores under RGS, which are Dry Bulb Temperature and Outdoor Air Density. And at the same time, we discard Relative Humidity, Wind Speed, Direct Solar and Ground Temperature, no matter how their variations could contribute to energy requirement, as we naturally thought. The Water Mains Temperature, which gives water temperatures delivered by underground water main pipes and Electrical Equipment Heat Gain, which is the heat gain of the room from electrical equipment such as lights or TVs, are determined by their power and occupants' schedule. They could probably be measured or assessed in actual buildings. We divide the room into several zones according to their thermal dynamics. The two features, Zone Mean Air Temperature, which is the effective bulk air temperature of the zone, and Zone Infiltration Volume which denotes hourly air infiltration of the zone, could also be measured or estimated in a normally operated building. All of the

Table 5.2: The scores of features evaluated by RGS and CC selection methods. The stars indicate selected features in that case.

Features	RGS	CC	Case1	Case2	Case3	Case4	Case5
Outdoor Dry Bulb	1.61	0.29	*		*		*
Outdoor Relative Humidity	0.62	0.26			*	*	
Wind Speed	0.52	0.01			*	*	
Direct Solar	0.54	0.47				*	
Ground Temperature	0.99	0.07				*	
Outdoor Air Density	1.26	0.20	*				*
Water Mains Temperature	1.30	0.07	*				*
Zone Total Internal Total Heat Gain	1.01	0.67					
People Number Of Occupants	0.93	0.68	*	*	*		
People Total Heat Gain	0.93	0.68		*		*	
Lights Total Heat Gain	1.13	0.05	*		*		*
Electric Equipment Total Heat Gain	1.06	0.69	*	*	*		*
Window Heat Gain for each wall	1.03	0.62		*		*	
Window Heat Loss for each wall	0.93	0.50		*		*	
Window Heat Gain for each wall	0.82	0.35				*	
Window Heat Loss for each wall	0.82	0.49				*	
Window Heat Gain for each wall	0.73	0.56		*		*	
Window Heat Loss for each wall	0.82	0.48				*	
Window Heat Gain for each wall	0.89	0.56		*		*	
Window Heat Loss for each wall	0.95	0.50		*		*	
Zone Mean Air Temperature	1.14	0.22	*				*
Zone Infiltration Volume	1.00	0.34	*		*		
District Heating Outlet Temp	0.95	7.35e-4			*	*	

above selected features have scores not less than 1. A special case we have to consider is the People Number Of Occupants. This feature takes a middle place under RGS, but since it can be easily counted in real life and has a very high score under the evaluation of CC, we choose to keep it in the final subset. All other features will be discarded since they get low scores or are hard to collect in actual buildings. For example, Zone Total Internal Total Heat Gain is difficult to obtain directly and District Heating Outlet Temp is useless according to CC. The selected features are indicated with stars in column Case1 in Table 5.2.

New data sets for both training and testing are generated by eliminating useless features from the data sets used in the previous experiment. Then, the model is retrained from the new training data and after applying the model to predict on the testing data, our results are as follows: MSE is $6.19e-4$ and SCC is 0.97. To obtain a clear view

of how the model performance changes before and after FS, we plot the measured and predicted daily consumptions in Figure 5.1. The relative errors are within $(-16\%, 12\%)$ as show in Figure 5.2. We note that after FS, the number of features is 8, which is only one third of the original set which has 23 features. However, compared to the results before FS, the model's prediction ability is still very high and the selected subset is therefore regarded as acceptable.

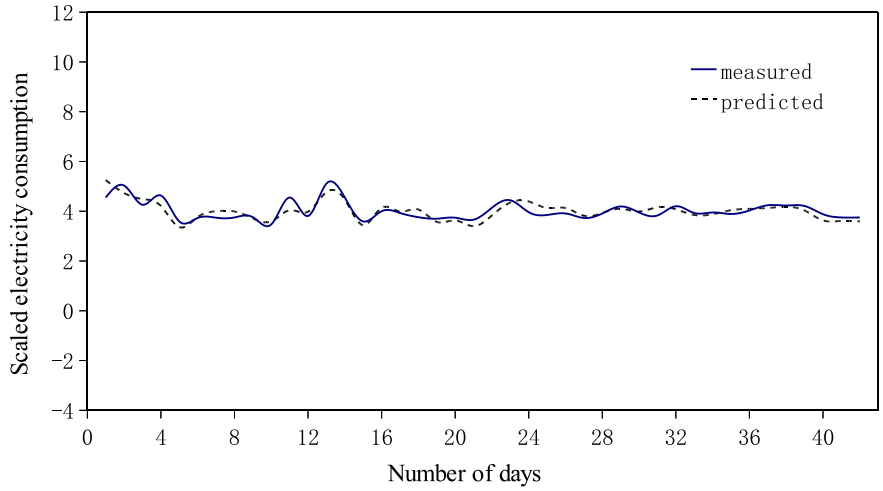

Figure 5.1: The comparison of measured and predicted daily electricity consumption for a particular building on working days, with FS performed.

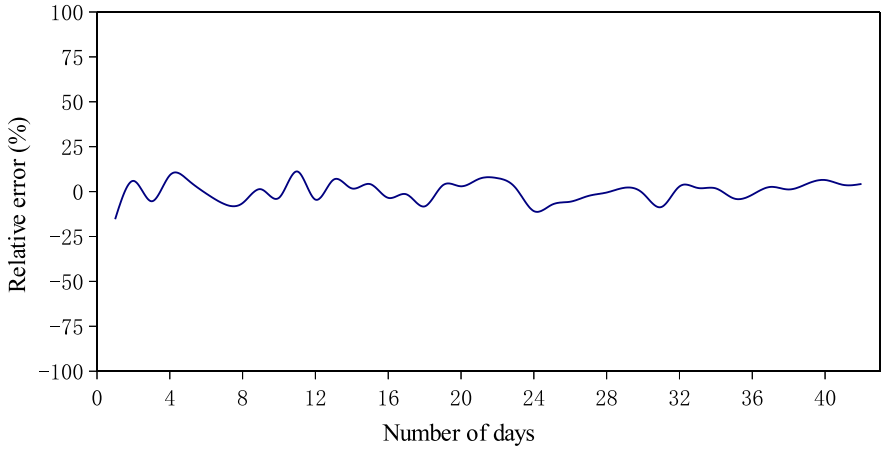

Figure 5.2: The relative error for the prediction.

Four other subsets are formed in order to further evaluate if the selected feature set

is optimal. They are indicated by columns Case2, Case3, Case4 and Case5 in Table 5.2. In case 2, we select the top 8 features under the evaluation of CC alone. By doing this, we are aiming at demonstrating whether the single CC is sufficient to select the best feature set. The Zone Total Internal Total Heat Gain feature is also ignored in this case just as we do in case 1. In case 3, we change three of the selected features to three other unselected ones. Outdoor Air Density, Water Mains Temperature and Zone Mean Air Temperature, which are selected in case 1, are substituted with Outdoor Relative Humidity, Wind Speed and District Heating Outlet Temp. In case 4, all of the selected features are substituted with other unselected ones except Zone Total Internal Total Heat Gain, which is not regarded as being directly obtainable in practice. In the last case, two features which gain the lowest scores are removed from the selected subset. They are People Number Of Occupants and Zone Infiltration Volume.

Based on these considerations, four new data sets are generated both for training and testing, and a model is retrained for each case. We show the results of all five cases in Table 5.3. Two conclusions can be reached according to the results, the first one is that the designed FS method is valid due to model performance in case 1 outperforming the other three cases. The other conclusion is that SVR model with RBF kernel has stable performance since high prediction accuracy is always achieved on all of the four subsets.

Table 5.3: Comparison of model performance on different feature sets. NF: Number of features, MSE: Mean squared error, SCC: Squared correlation coefficient.

	Case1	Case2	Case3	Case4	Case5
NF	8	8	8	14	6
MSE	6.2e-4	1.9e-3	7.5e-4	2.1e-3	9.2e-4
SCC	0.97	0.93	0.96	0.90	0.96

5.4.3 Model evaluation on multiple buildings data

Previously, we tested the FS method on one particular building's consumption over a year. In this section, we investigate how the subset of features influences the model performance on multiple buildings' consumptions.

We choose the consumption data in the winter season for 50 buildings. The differences among these buildings mainly come from the weather conditions, building struc-

tures and the number of occupants. We suppose these buildings randomly distributed in five cities in France, which are Paris-Orly, Marseilles, Strasbourg, Bordeaux and Lyon. As outlined in Figure 5.3, the five cities vary remarkably in ambient dry bulb temperatures, making the datasets represent energy requirements under five typical weather conditions. The buildings have diverse characteristics with randomly generated length, width, height and window/wall area ratio. The number of occupants is determined by the ground area and people density of the buildings. The time series data of those buildings is combined together to form the training sets. One more building is simulated for model evaluation purpose.

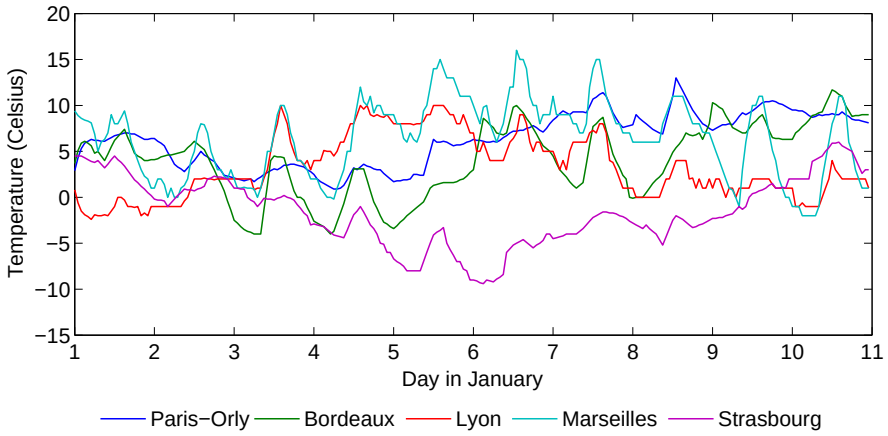

Figure 5.3: Dry bulb temperature in the first 11 days of January.

Two sets of consumption data are designed. The first set has 20 buildings and the second includes all 50 buildings. To fully investigate how FS on these two data sets influences SVR models, two kernels are involved. Besides RBF kernel, we also test the performance of FS on SVR with a polynomial kernel, which is also applicable on non-linear problems. The kernel parameters r is set to zero and d is estimated by 5-fold cross validation in a searching space $\{2, 3, ..., 7\}$. Selected features for representing multiple buildings are the feature sets for a single building plus building structures. Therefore, FS for multiple buildings has reduced the number of features from 28 to 12. The changes of MSE and SCC on these data sets are shown in Table 5.4. For information, the results of one single building is indicated in the same table.

After FS, the accuracy of the prediction on 50 buildings' consumptions improves significantly. With regard to 20 buildings' consumptions, MSE increases to a certain

Table 5.4: Prediction results of SVR with two kernel methods on three data sets. (BF: Before feature selection, AF: After feature selection, MSE: Mean squared error, SCC: Squared correlation coefficient).

			One building	20 buildings	50 buildings
RBF kernel	BF	MSE	4.8e-4	4.3e-4	4.4e-4
		SCC	0.97	0.97	0.97
	AF	MSE	6.2e-4	2.1e-3	3.7e-4
		SCC	0.97	0.96	0.97
Polynomial kernel	BF	MSE	8.0e-4	5.8e-4	5.9e-4
		SCC	0.96	0.96	0.96
	AF	MSE	2.1e-3	0.19	4.7e-4
		SCC	0.91	0.85	0.98

extent, indicating a decrease in prediction accuracy. However, from the standpoint of SCC, the performance of the model with RBF kernel involved is quite close to the situation without FS performed, as shown in Figure 5.4(a). With regard to the polynomial kernel, when training on the original data sets, the prediction ability of the model is just as good as RBF kernel, indicating that the polynomial kernel is also applicable on such a problem. After adopting FS, the performance of the model improves in the case of 50 buildings. Unfortunately, it decreases largely in the case of 20 buildings. It seems that the polynomial kernel is not as stable as RBF kernel when applied to such problems. However, we can see that it performs better for the case in 50 buildings than for that of 20 buildings. The same trend is also found for RBF kernel. These phenomena indicate that the proposed FS approach could give better performance to the models when more training samples are involved.

Another advantage of FS for statistical models is the reduction of training time. We show the time consumed for training SVR models with RBF kernel in Figure 5.4(b) where the time is in the logarithm form. The training time after FS is obviously less than that before FS, but the reduction is not too much. This phenomenon can be explained by the different parameter values we assigned for the learning algorithm, which always have a great influence on the training speed. We note that the time for choosing parameters for the predictor via cross validation is too long to be ignored when evaluating a learning algorithm. While in this chapter we primarily focus on the influences of FS on predictors, the labor and time for choosing model parameters are not considered here since they are quite approximate before and after FS.

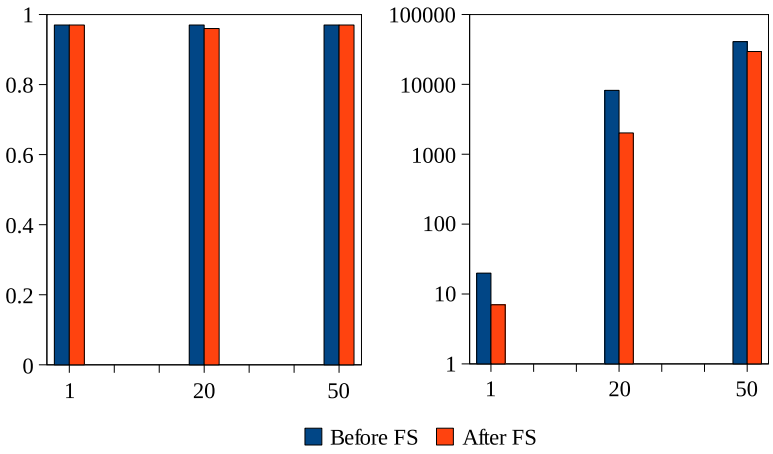

Figure 5.4: (a)The comparison of model performance from the standpoint of SCC before and after FS for RBF kernel. X-axis represents the three datasets, Y-axis is the SCC. (b)The comparison of training time before and after FS for RBF kernel. Y-axis stands for training time (in seconds).

5.5 Discussion

This chapter introduces a new feature selection method for applying support vector regression to predict the energy consumptions of office buildings.

To evaluate the proposed feature selection method, three data sets are first generated by EnergyPlus. They are time series consumptions for one, twenty and fifty buildings respectively. We assume that the developed models are applied to predict the energy requirements of actual buildings, therefore, the features are selected according to their feasibility in practice. To support the selection, we adopt two filter methods: the gradient guided feature selection and the correlation coefficients, which can give each feature a score according to its usefulness to the predictor. Extensive experiments show that the selected subset is valid and can provide acceptable predictors. Performance improvement is achieved in some cases, e.g., accuracy remarkably enhanced for the models with either radial basis function or a polynomial kernel on fifty buildings' data, and the time for model learning decreases to a certain extent. We also identify that the performance improves when more training samples get involved. Besides radial basis function kernel, we proved that a polynomial kernel is also applicable to our application. However, it does not seem as stable as radial basis function kernel. Furthermore, it requires more complicated pre-processing work since more kernel parameters need to

be estimated.

This preliminary work on feature selection for building energy consumptions has paved a way for its further progress. It serves as the first guide for selecting an optimal subset of features when applying machine learning methods to the prediction of building energy consumption.

6

Model Optimization — Parallelize SVM

6.1 Introduction

As introduced in Chapter 2, the essential computation of SVMs is to solve a quadratic problem, which is both time and memory costly. This presents a challenge when solving large scale problems. Despite several optimizing or heuristic methods such as shrinking, chunking [156], kernel caching [117], approximation of kernel matrix [157], Sequential Minimal Optimization (SMO) [158] and primal estimated sub-gradient solver [104], a more sophisticated and satisfactory resolution is always expected for this challenging problem. As stated in Chapter 1, the building's energy system is extremely complex involving large number of influence factors, making tackling large scale datasets common.

With the development of chip technologies, computers with multi-core or multi-processor are becoming more available and affordable in the modern market. This chapter, therefore, attempts to investigate and demonstrate how SVMs can benefit from this modern platform when solving the problem of predicting building energy consumption. A new parallel SVMs that is particularly suitable to this platform is proposed. Decomposition method and inner SMO solver compose the main procedure of training. A shared cache is designed to store the kernel columns. For the purpose of achieving easy implementation without sacrificing performance, the new parallel programming framework Map-Reduce is chosen to perform the underlying parallelism.

The proposed system is, therefore, named as MRPsvm (abbreviation of "Map-Reduce parallel SVM"). We parallelize both classification and regression algorithms. Comparative experiments are conducted on five benchmarking datasets and three energy consumption datasets for SVC and SVR respectively, showing significant performance improvement in our system compared to the sequential implementation Libsvm [118] and the state-of-the-art parallel implementation Pisvm [159].

This chapter is organized as follows. Section 6.2 states the related work. Section 6.3 introduces the decomposition QP solver method for SVC. Section 6.4 explains some important issues with the MRPsvm's implementation. Section 6.5 presents numerical experiments on the performance test and comparative analysis based on benchmarking datasets. Section 6.6 uses the same mechanism to parallelize SVR and shows its application in predicting building energy consumption. Conclusions are drawn in section 6.7.

6.2 Related work

Several approaches have been proposed to parallelize SVM, mostly for solving classification problems. They can be classified into several categories according to the type of QP solver. Based on stochastic gradient descent method, P-packSVM optimizes SVM training directly on the primal form of SVM for arbitrary kernels [160]. Very high efficiency and competitive accuracy have been achieved. Psvm proposed in [157] is based on interior point QP solver. It approximates the kernel matrix by incomplete Cholesky Factorization. Memory requirement is reduced and scalable performance has been achieved. Bickson et al. [161] solve the problem by Gaussian belief propagation which is a method from complex system domain. The parallel solver brings competitive speedup on large scale problems. The decomposition method attracts more attention than the above solvers. Graf et al. [162] train several SVMs on small data partitions, then they aggregate support vectors from two-pair SVMs to form new training samples on which another training is performed. The aggregation is repeated until only one SVM remains. A similar idea is adopted by Dong et al. [163], in their work, sub-SVMs are performed on block diagonal matrices which are regarded as the approximation of the original kernel matrix. Consequently, nonsupport vectors are removed when dealing with these sub-problems. Zanni et al. [164] parallelize SVM-light with improved

working set selection and inner QP solver. Hazan et al. [165] propose a parallel decomposition solver using Fenchel Duality. Lu et al. [166] parallelize randomized sampling algorithms for SVM and SVR.

Cao et al. [167] and Catanzaro et al. [168] parallelize SMO solver for training SVM for classification. Both works mainly focus on updating gradient for KKT condition evaluation and the working set selection. The difference between them lies in the implementation details and the programming models. Specifically, the first work is conducted by using MPI on clusters while the second by Map-Reduce threads on modern GPU platform. In our work, we also adopt SMO algorithm. But we use it as the inner QP solver without any parallel computation, in fact, we perform the parallelization on external decomposition procedure. The main advantage of our coarse-grained parallelism is that it can significantly reduce the burden of overheads since the number of iterations in global decomposition procedure (where $n \gg 2$) is much smaller than that of pure SMO algorithm (where $n = 2$). Although both GPU SVM [168] and our system are implemented in threads, we will not compare them in experiments since they are designed for different platforms and GPU SVM is specially used to solve classification problems.

6.3 Parallel QP solver

In Chapter 2, we have introduced interior point and gradient descent methods for QP solver. As we have stated, the third one is decomposition method. We present this approach in the following, and then based on it, we develop our parallel implementation.

6.3.1 Decomposition method

Decomposition method reduces the problem into smaller tasks, then solves these small tasks and finally achieves global convergence. This method has attracted a lot of attention as the QP solver for SVMs in recent years, since it is quite efficient for large scale problems and its memory requirement is also much less. It was firstly proposed by Osuna et al. to decompose the dual problem of SVMs [156]. In each small task, a working set which contains certain parts of α is selected to be optimized, while the rest of α remains at a constant value. The program repeats the select-optimize process

iteratively until global optimality conditions are satisfied. In each iteration, only the involved partition of kernel matrix needs to stay in the memory.

Similar to the dual form of SVC (Equation (2.16)), we can write the general dual form of SVMs as:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^T Q\alpha - p\alpha \\ \text{subject to} \quad & y^T\alpha = 0, \quad 0 \leq \alpha \leq C \end{aligned} \tag{6.1}$$

Let B denote the working set which has n variables and N denote the non-working set which has $(l-n)$ variables. Then, α, y, Q and p can be correspondingly written as:

$$\alpha = \begin{vmatrix} \alpha_B \\ \alpha_N \end{vmatrix}, \quad y = \begin{vmatrix} y_B \\ y_N \end{vmatrix}, \quad Q = \begin{vmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{vmatrix}, \quad p = \begin{vmatrix} p_B \\ p_N \end{vmatrix} \tag{6.2}$$

Accordingly, the small task of the dual form in this case can be written as:

$$\begin{aligned} \min \quad & \frac{1}{2}\alpha_B^T Q_{BB}\alpha_B - \alpha_B^T(p_B - Q_{BN}\alpha_N) \\ & + \frac{1}{2}\alpha_N^T Q_{NN}\alpha_N - \alpha_N^T p_N \end{aligned} \tag{6.3}$$

subject to

$$\alpha_B^T y_B + \alpha_N^T y_N = 0 \tag{6.4}$$

$$0 \leq \alpha_B \leq C \tag{6.5}$$

Since the last term $(\frac{1}{2}\alpha_N^T Q_{NN}\alpha_N - \alpha_N^T p_N)$ of (6.3) remains constant in each iteration, it can be omitted while calculating, so that function (6.3) basically holds the same form as the original objective function (6.1). One of the advantages of this decomposition method is that the newly generated task is small enough to be solved by most off-the-shelf methods, requiring less storage space, which is probably affordable for modern computers. To the extreme, if the working set contains only two variables each time, the derived algorithm is SMO. Actually, this is an efficient inner small task solver due to its relative simplicity, yet high performance characteristics. This kind of binary sub-problem can be easily solved analytically [158] [118]. As stated in [117], the solution of (6.3) is strictly feasible towards the optimum solution of the global

problem (6.1). This feature guarantees the global convergence of the decomposition method.

The KKT optimality conditions have been discussed in Section 2.4. In practice, they are verified through evaluating the gradient of (6.1), i.e., $f_i = \sum_{j=1}^{l} \alpha_j Q_{ij} + p_i$ for all $i = 1, ..., l$. This procedure can be summarized as follows. First, we classify the training samples into two categories

$$I_{up}(\alpha) = \{i | \alpha_i < C, y_i = 1 \text{ or } \alpha_i > 0, y_i = -1\} \tag{6.6}$$

$$I_{low}(\alpha) = \{i | \alpha_i < C, y_i = -1 \text{ or } \alpha_i > 0, y_i = 1\} \tag{6.7}$$

Then we search two extreme values $m(\alpha)$ and $M(\alpha)$:

$$m(\alpha) = \max_{i \in I_{up}(\alpha)} - y_i f_i \tag{6.8}$$

$$M(\alpha) = \min_{i \in I_{low}(\alpha)} - y_i f_i \tag{6.9}$$

And then, we define the stopping criterion as:

$$m(\alpha) - M(\alpha) \leq \epsilon \tag{6.10}$$

The selection of working set directly influences the speed of convergence. For inner SMO solver, a maximal violating pair is selected to be the binary working set according to the first or second order information [169]. We do not state here how inner SMO solver works since it has been discussed in detail in [158] and [118]. For the selection of working set B, we simply consider the first order information and select the maximal violating pairs, as proposed by [164]. Suppose the required size of B is n, we choose q ($q < n$) variables from α by sequentially selecting pairs of variables which satisfy (6.8) and (6.9). The remaining $(n-q)$ variables are chosen as those who entered B in the last iteration but have not yet been selected in current B. The selection of these $(n - q)$ variables follows the sequence: free variables ($0 < \alpha_i < C$), lower bound variables ($\alpha_i = 0$), upper bound variables ($\alpha_i = C$). The reason for putting restraint on the number of new variables entering the working set is to avoid frequent entering-leaving

of certain variables. Otherwise, the speed of convergence would considerably slow down [164].

After the working set is optimized, f is updated by the newly optimized $\alpha_j, \forall j \in B$. This procedure is crucial as it prepares f to do optimality condition evaluation and working set selection for the next iteration. In fact, this is the most computational expensive step in SVM training due to the heavy work of computing Q_{ij}. The updating procedure can be written as follows:

$$f_i^* = f_i + \sum_{j \in B} \Delta\alpha_j Q_{ij} \qquad i = 1, ..., l \tag{6.11}$$

where $\Delta\alpha_j$ is the newly optimized α_j minus the old α_j. The whole decomposition method is summarized in Algorithm 6.1.

Algorithm 6.1 Decomposition solver of SVM

Input: data set $(x_i, z_i), \forall i \in 1, ..., l$
Initialize: $\alpha_i = 0$, y_i, f_i, $\forall i \in 1, ..., l$
Calculate: I_{up}, I_{low}, $m(\alpha)$, $M(\alpha)$
Repeat
  select working set B until $\mid B \mid = n$
  update α_i by SMO solver, $\forall i \in B$
  update f_i, $\forall i \in 1, ..., l$
  calculate I_{up}, I_{low}, $m(\alpha)$, $M(\alpha)$
Until $m(\alpha) - M(\alpha) \leq \epsilon$

6.4 Implementation

This section discusses some important issues with MRPsvm implementation. The underlying parallelism is based on Map-Reduce framework. The communication and data decomposition is especially designed for multi-core and multiprocessor systems.

6.4.1 Map-Reduce for solving underlying parallelism

Map-Reduce is a new parallel programming framework originally proposed in [170]. It allows users to write code in a functional style: map computations on separated data, generate intermediate key-value pairs and then reduce the summation of intermediate values assigned to the same key. A runtime system is designed to automatically handle

low-level mapping, scheduling, parallel processing and fault tolerance. It is a simple, yet very useful framework. It can help people extract parallelism of computations on large datasets by taking advantage of distributed systems.

Problem (6.11) can be regarded as a summation of several computational expensive terms, as shown in the top right corner in Figure 6.1. Therefore, Map-Reduce is naturally suitable to deal with this problem. The working set B is uniformly decomposed into several small pieces, the calculation of f^* is also divided into several parts in the same manner as for B. Each part is then assigned to a mapper. After the parallel calculations of these mappers, final f^* is added up by the reducer. Here, $(j_k, k = 1, ..., n)$ is the variable index of working set in kernel matrix, which gives the kth variable in B with its index in Q as j_k. In practice, since some of $\Delta\alpha_i$ are so marginal that they can be omitted, it is not necessary to update f on all of the n variables.

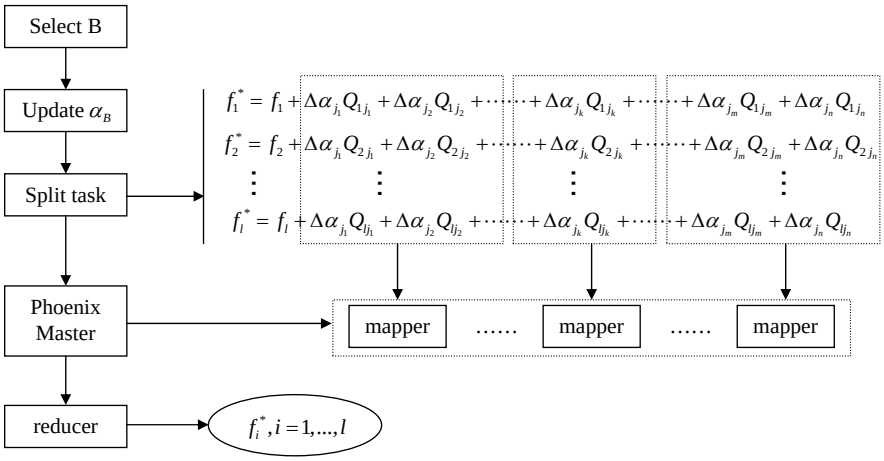

Figure 6.1: Architecture of the parallelization in one iteration.

In some recent work, Map-Reduce has proved to be an effective parallel computing framework on multi-core systems. Chu et al. [171] have developed Map-Reduce as a general programming framework on multi-core systems for machine learning applications. Phoenix designed in [172] implements a common API for Map-Reduce. It allows users to easily parallelize their applications without conducting concurrency management. The Map-Reduce tasks are performed in threads on multi-core systems. An efficient integrated runtime system is supposed to handle the parallelization, resource management and fault recovery by itself. This system is adopted as the underlying Map-Reduce handler in our MRPsvm. We show the parallel architecture of one itera-

tion in Figure 6.1. The small tasks are uniformly distributed to mappers which have the same number of processors. Phoenix serves as the role of creating and managing mappers and reducers, making the system is easy to implement.

6.4.2 Caching technique

As stated in the previous sections, for large scale problems, kernel matrix Q is too large to be stored in memory, and the calculation of kernel elements Q_{ij} is the dominant work that slows down the training. It is an effective technique to cache the kernel elements in memory as much as possible. MRPsvm maintains a fix-sized cache which stores recently accessed or generated kernel columns. The cache replacement policy is a simple least-recent-use strategy, the same as that of Libsvm. Only the column currently needed but not hit in the cache will be calculated. All parallel maps share unique copy of cache in the shared memory. In consequence, the operation of inserting a new column into the cache performed by whichever map should be synchronized.

For inner SMO solver, the kernel matrix size is dependent on the size of working set B which is normally set as 1024 according to the knowledge gained by experience, it is practical to pre-compute and cache the full version of this small kernel matrix.

6.4.3 Sparse data representation

To reduce the storage requirements, the sample vectors x_i are stored by sparse representation. When calculating a kernel column $(Q_{ij}, i = 1, 2, ..., l)$, we need to unroll the j^{th} sample vector to dense format and then calculate the dot products of this vector with the other l sample vectors.

6.4.4 Performance analysis

The computation of the algorithm 6.1 comprises two main procedures, initialization and iteration loops. We can formulate the running time of the iteration loops as $T = t_r * t_i$, where t_r denotes the convergence rate and t_i denotes the computation in one iteration. Since t_r depends on the datasets, normally it is in the range $l \sim l^{2.2}$ [158], it is difficult to specify its value. The following is a step-by-step analysis of one iteration.

- Select working set B: The complexity is n.

- Update α_i's by SMO solver: The size of the inner problem is n, thus the complexity is between n and $n^{2.2}$, we denote it as t_{smo}.
- Update f_i's: The parallel computation is in this procedure. Suppose the number of features is m, the kernel function is RBF kernel, there are p threads. we analyze the performance in the following two situations, depending on whether the kernel columns are cached.
  - Suppose the whole n kernel columns are cached, there is no need to calculate the kernel elements, thus the computation in each thread is $n * l/p$. The global running time is $n * l/p + t_o$, where t_o is the parallel overheads.
  - Suppose the kernel columns are not cached, then the computation in each thread is $n * m * l/p$, and thus the global complexity is $n * m * l/p + t_o$.
- Calculate $m(\alpha)$, $M(\alpha)$: The main work of this procedure is a quick sort of all α_i's, thus we can note the complexity as $l * log(l)$.

From the above analysis, we can see that it is difficult to precisely analyze the algorithm complexity since the convergence rate depends on the specific dataset and the hitting rate of the kernel columns is unpredictable. We roughly summarize that, when RBF kernel is used, if the kernel columns are cached, the complexity is $t_r * (n + t_{smo} + n * l/p + t_o + l * log(l))$. In contrast, if the kernel columns are not cached, the complexity is $t_r * (n + t_{smo} + n * m * l/p + t_o + l * log(l))$.

Suppose the cache size is $s_c * l$, where s_c is the maximum number of kernel columns that can be cached, then the space requirement is $O(l * x)$ where $x = \max(m, n, s_c)$.

6.4.5 Comparison of MRPsvm with Pisvm

Pisvm also uses decomposition method to train SVM in parallel. It is an efficient tool to analyze multiple buildings' energy behaviors as stated in our previous work [173]. However its implementation is different from our new implementation MRPsvm in many aspects. First, Pisvm is based on MPI implementation and aims at extracting parallelism from distributed memory systems, while our parallel algorithm is conducted by Map-Reduce threads on shared memory system. The two implementations are based on totally different models. Second, in Pisvm, each process stores a copy of data samples, while on the contrary, MRPsvm stores only one copy in the shared memory.

This means MRPsvm can save large amount of storage when the dataset is huge. The saved space can be used to cache more kernel matrix in order to further improve training speed. Third, Pisvm adopts a distributed cache strategy in order to share the saved kernel elements among all of the processes. Each process stores a piece of the cache locally. Consequently, the work for updating gradients is divided and assigned globally to proper processors according to the cache locality. In contrast, MRPsvm has only one copy of the cache, and each processor accesses the cache equally, so that the overhead of global assignment is avoided. However, we have to note that synchronization on cache write is required.

In the next section, we will compare the performance of Pisvm with that of MRPsvm in real application datasets, providing direct evidence that our system is more efficient and suitable than the MPI implementation on multi-core systems.

6.5 Comparative experiments on benchmark datasets

We test MRPsvm by comparing it with the parallel implementation Pisvm and the serial implementation Libsvm on five widely used benchmark datasets. Although this comparison may not be based on systems especially designed for multi-core architecture, we still have good reasons for doing so. Firstly, to the best of our knowledge, there is no existing parallel implementation of general SVM that is specially developed for multi-core systems. Therefore there is a strong need to verify if our system could outperform the state-of-the-art parallel implementation. Secondly, most of the systems surveyed in Section 6.2 are not available to the public, while Pisvm, as a typical parallel implementation of SVM, is easy to obtain. Thirdly, the quadratic problem solver of Pisvm is the same as MRPsvm, hence, if we compare our system with Pisvm, the advantage of Map-Reduce framework is more convincing.

Two computers with different hardware architectures are adopted to check hardware effects. As shown in Table 6.2, the first computer has 4 cores with a shared L2 cache and memory. The second one is a dual-processor system with 4 cores in each processor. The cores in the same processor share one cache, and the main memory is shared among all of the cores. Both of the two computers are running Linux 2.6.27-7.

The five datasets are shown in Table 6.3. They vary in sample size and dimension. We train all SVMs with Gaussian kernel. The tolerance of the termination criterion is

Table 6.2: The physical features of the multi-core systems.

Features	Computer 1	Computer 2
# of CPUs	1	2
# of cores	4	8
Frequency	1600MHz*4	2327MHz*8
Memory	2G	4G
L2 cache	4M	6M*2

Table 6.3: Description of the five datasets and the two parameters of SVM on each dataset.

	Web	Adult	Mnist	Covtype	Kddcup99
# training samples	24,692	32,561	60,000	435,759	898,430
# testing samples	25,075	16,281	10,000	145,253	311,029
# Classes	2	2	2	8	2
# Dimensions	300	123	576	54	122
C	64	100	10	10	2
γ	7.8152	0.5	1.667	2e-5	0.6

set to 0.01. Since we focus on comparing three systems, the outputs of these classifiers may not be optimal. In other words, we do not guarantee the parameters of SVM, i.e., C and γ, to reach optimal values. They are just chosen from the literature as shown in the last two lines of Table 6.3.

Since the caching technique is crucial for performance, for a reliable comparison, we set the cache size to be the same for all three systems. Furthermore, we restrict the cache size to be far smaller than the memory size in order to minimize page faults in runtime. Here we have to emphasize that the following reported performance might not be optimal for all three systems, only serving for comparison purpose.

Table 6.4 shows the results of the three implementations performed on computer 1 (with 4 processors). The time columns represent the whole training time, i.e., from reading the problem to writing the outputs. Here we use "speedup" to denote how many times faster parallel implementation is over sequential implementation:

$$Speedup = \frac{Time\ of\ Libsvm}{Time\ of\ parallel\ implementation} \tag{6.12}$$

By analyzing the results, we can see that MRPsvm has successfully parallelized SVM

Table 6.4: The training time and accuracy of the three systems on five datasets performed on computer 1. The unit of time is second.

		Web	Adult	Mnist	Covtype	Kddcup99
Libsvm	Time	306.4	311.6	517.8	20260.7	726.8
	Accuracy	97.6%	82.7%	99.8%	51.0%	92.0%
Pisvm	Time	117.5	91.4	148.7	5612.6	415.5
	Accuracy	97.6%	82.7%	99.8%	51.0%	92.0%
	Speedup	2.6	3.4	3.5	3.6	1.7
MRPsvm	Time	65.8	59.2	123.2	3895.1	351.9
	Accuracy	97.6%	82.7%	99.8%	51.0%	92.0%
	Speedup	4.7	5.3	4.2	5.2	2.1

training. For all five datasets, much more time is saved when running MRPsvm rather than Libsvm. Especially in the first four cases, the speed of MRPsvm is more than 4 times higher than that of Libsvm. In all of the cases, MRPsvm achieves outstanding higher speedup than Pisvm, indicating that MRPsvm is more suitable than Pisvm on multi-core systems.

We note that in these experiments, the accuracy of the three classifiers is almost the same. In fact, the numbers of support vectors generated by these classifiers are also quite close. Actually, these three implementations essentially have the same mechanism in quadratic problem solving, i.e., to iteratively optimize one pair of variables until achieving global optimization. The difference in runtime is mainly caused by the selected working set. Selecting different variables to perform optimization may induce totally different results in an iteration, but generally speaking, as long as global convergence is reached, the influence is marginal.

We show in Figure 6.2 the times up of the two parallel solvers over the sequential solver when running on the second computer. MRPsvm again outperforms Pisvm on all of the datasets. Among them the best speedup is achieved on Adult, while the worst is found on Kddcup99. This indicates that MRPsvm performs better on smaller datasets. The main reasons for worse performance on larger problems are due to locality and overheads of reduction. In each map, the updating of f requires accessing the whole data samples and several temporal vectors with the size close to l. Therefore, for large datasets, it is difficult to guarantee the locality for using L2 cache, especially when the cache is shared by several threads. Since we partition the global problem in columns,

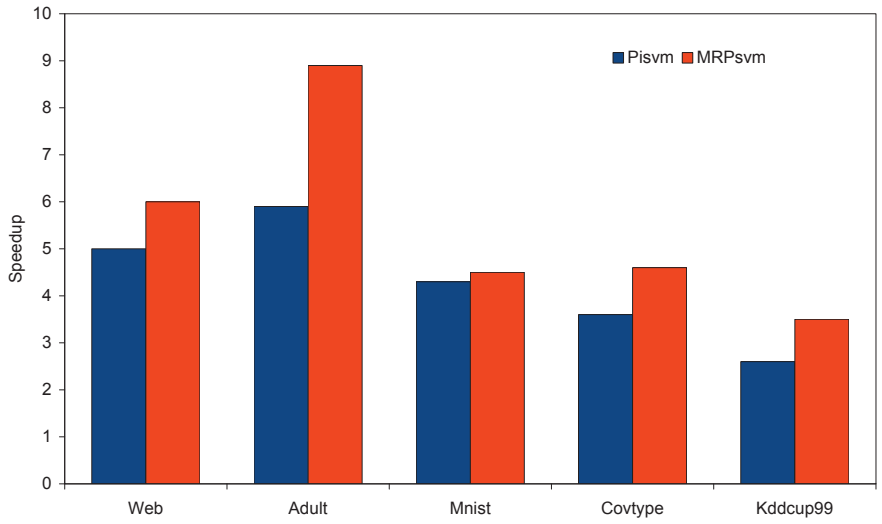

Figure 6.2: Speedup of Pisvm and MRPsvm over Libsvm when running on computer 2.

each map generates l intermediate f_i, so that the reduction is costly when l is very large.

In fact, the parallel performance on 8 cores only slightly outperforms that on 4 cores. As explained at the beginning of this section, this is because we did not make full use of the memory on computer 2. Far more time can be saved if we increase the cache size to the maximum with caution. In this optimal case, the cache size of MRPsvm and Libsvm is larger than that of Pisvm, since the former two systems generally require less memory. Therefore, it is implied that more improvements can be achieved for MRPsvm and Libsvm than Pisvm.

6.6 Parallel ε-SVR for solving energy problems

In this section, we use the same mechanism to parallelize SVR training process, and then apply this new parallel algorithm to predict building energy consumption. The above three implementations are compared again in this application.

Firstly, we reformulate ε-SVR in the same form as (6.1). Let us present the training data as $(x_1, z_1), ..., (x_l, z_l)$,where vector x_i is the i^{th} sample, z_i is the i^{th} target value corresponding to x_i, l is the number of samples. The dual form of the SVR can be

written in the following quadratic form:

$$\min_{\alpha} \quad \frac{1}{2}\alpha^T Q\alpha - \sum_{i=1}^{2l} p_i\alpha_i \tag{6.13}$$

subject to

$$y^T\alpha = 0 \tag{6.14}$$

$$0 \leq \alpha_i \leq C \quad \forall i = 1, ..., 2l \tag{6.15}$$

where α is a vector of $2l$ variables, Q is a $2l$ by $2l$ kernel matrix. Each element of Q has the following form:

$$Q_{ij} = K(x_m, x_n) \tag{6.16}$$

$$m = \begin{cases} i & \text{if} \quad i \leq l \\ i - l & \text{if} \quad i > l \end{cases} \tag{6.17}$$

$$n = \begin{cases} j & \text{if} \quad j \leq l \\ j - l & \text{if} \quad j > l \end{cases} \tag{6.18}$$

$$i, j = 1, ..., 2l \tag{6.19}$$

where $K(x_i, x_j)$ is a kernel function. The parameter p in (6.13) is defined as:

$$p_i = \begin{cases} \varepsilon + z_i & \text{if} \quad i = 1, ..., l \\ \varepsilon - z_i & \text{if} \quad i = l + 1, ..., 2l \end{cases} \tag{6.20}$$

y in the constraint (6.14) is defined as:

$$y_i = \begin{cases} 1 & \text{if} \quad i = 1, ..., l \\ -1 & \text{if} \quad i = l + 1, ..., 2l \end{cases} \tag{6.21}$$

In the constraint (6.15), C is the upper bound used to trade off between model performance on training data and its generalization ability. The objective of the problem is to find the solution of α which minimizes (6.13) and fulfills constraints (6.14) (6.15). After the optimum α is found, the decision function can be formulated as:

$$g(x) = \sum_{i=1}^{l}(-\alpha_i + \alpha_{i+l})K(x_i, x) + b \tag{6.22}$$

Only support vectors satisfy $(-\alpha_i + \alpha_{i+l} \neq 0)$.

6.6.1 Energy consumption datasets

Three datasets are prepared for the model training. They denote the historical energy consumption of one building, 20 buildings and 50 buildings respectively. All of these buildings are located in urban areas. We evenly distribute them into five typical cities in France. The dry bulb temperatures of these five places is drawn in Figure 5.3. Each building has similar structures, i.e., single-story, mass-built, one rectangular room with an attic roof and four windows without shading. Electrical equipment including lighting system, fans and water heaters, are scheduled as for common office use. In the winter season (from November 1^{st} to March 31^{st}), district heating is applied in order to keep the room temperature at a constant level. Ventilation is adopted for indoor thermal comfort. The number of occupants depends on the housing space and people density, with the average of 0.2 people per zone floor area. During the simulation, some input variables such as size, orientation, window area, scheduling, are set differently to achieve diversity among multiple buildings.

The dataset of one building is hourly energy dynamics in a period of one year. We select 8 important features as shown in the column "Case1" in Table 5.2 according to the evaluation of our feature selection method. For two other datasets, the recording period is from November to March which is the winter season in France, and we record 4 more features which generate the building diversity, i.e., height, length, width and window/wall area ratio. Since weekends and holidays have totally different energy behaviors from working days, to simplify the model in our practice, we only use the consumption data of working days in the experiments. One more building is simulated for model evaluation purpose. The attributes of the three datasets are shown in Table 6.5.

Table 6.5: Description of the three datasets and the three parameters of SVR on each dataset. #tr: number of training samples, #te: number of testing samples

Dataset	#tr	#te	Dimensions	C	γ	ε
One building	5064	1008	8	32	0.594	0.01
20 buildings	49940	2498	12	16	0.4193	0.01
50 buildings	124850	2498	12	16	0.4357	0.01

6.6.2 Experiments and results

We perform the experiments on two shared memory systems as outlined in Table 6.6. To vary from the above computers, this time we use two different ones, the first has 2 cores while the second has 4. Both of them have a shared L2 cache and memory and running 64 bit Linux system (kernel version 2.6.27-7). Their memory size is the same.

Table 6.6: The physical features of the experimental environment.

Features	Computer-I	Computer-II
# of CPUs	1	1
# of cores	2	4
Frequency	3.4GHz*2	1.6GHz*4
Memory	2G	2G
L2 cache	2M	4M

We train all SVRs with Gaussian kernel, the parameters as shown in the last three columns of Table 6.5. Again, we restrict the cache size to be far smaller than the memory size.

Table 6.7: The training time and performance of the three predictors on three datasets performed on computer-I. bd: building, nSVs: number of support vectors, MSE: mean squared error, SCC: squared correlation coefficient. The unit of time is second.

Data	Libsvm				Pisvm			
	nSVs	*MSE*	*SCC*	*Time*	*nSVs*	*MSE*	*SCC*	*Time*
1 bd	2150	6.16e-4	0.97	22.3	2162	6.10e-4	0.97	9.2
20 bd	9014	2.11e-3	0.96	3407.0	8967	2.12e-3	0.96	339.5
50 bd	22826	3.73e-4	0.97	44212.5	22823	3.74e-4	0.97	4179.8

Data	MRPsvm			
	nSVs	*MSE*	*SCC*	*Time*
1 bd	2168	6.14e-4	0.97	9.0
20 bd	8970	2.08e-3	0.96	212.7
50 bd	22799	3.73e-4	0.97	2745.8

On each dataset, we train Libsvm, Pisvm and MRPsvm on both computer-I and computer-II. Table 6.7 shows the results of the three implementations performed on dual-core processor, including the number of support vectors (nSVs), MSE, SCC and training time. We show the training time on quad-core system in Table 6.8. Since

nSVs, MSE and SCC in quad-core case are the same as those in dual-core case, we omit them in Table 6.8.

We can see that, for all three datasets, the accuracy and the nSVs of MRPsvm are quite close to that of Libsvm and Pisvm. MRPsvm runs faster than Libsvm and Pisvm in all tests.

We show the speedup of MRPsvm and that of Pisvm comparatively in Figures 6.3, 6.4, 6.5. For the case of one building, the speed of MRPsvm is twice as high as that of Libsvm. For the case of 20 and 50 buildings, MRPsvm performs more than 16 times faster than Libsvm. On both computers and on all three datasets, MRPsvm achieves remarkable higher speedup than Pisvm, indicating that MRPsvm is more suitable than Pisvm on multi-core systems. The speed improvement by MRPsvm is particularly obvious in multiple buildings cases, indicating that MRPsvm performs better than Pisvm on much larger datasets. However, the better performance is not guaranteed on even larger problems due to locality and overheads of reduction as stated in Section 6.5.

Table 6.8: The training time of the three predictors performed on computer-II. Time unit is second.

Dataset	Libsvm	Pisvm	MRPsvm
1 building	18.0	7.3	6.9
20 buildings	2532.1	214.1	133.5
50 buildings	32952.2	2325.5	1699.0

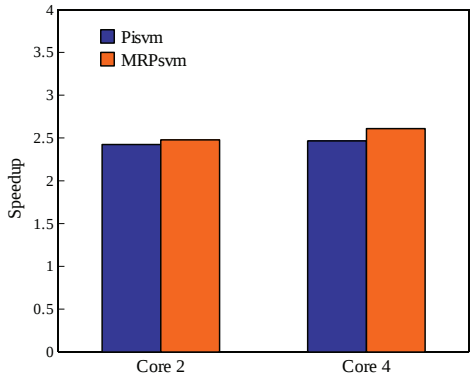

Figure 6.3: Speedup of Pisvm and MRPsvm on one building's data.

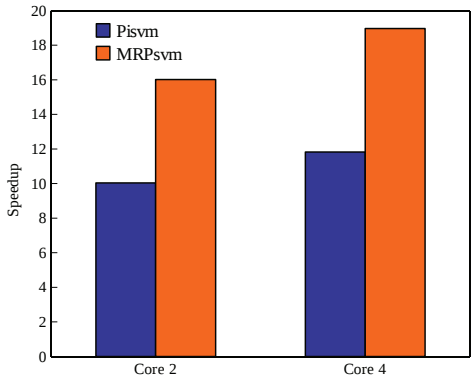

Figure 6.4: Speedup of Pisvm and MRPsvm on 20 buildings' data.

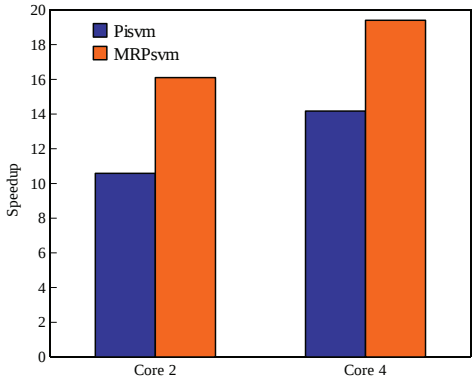

Figure 6.5: Speedup of Pisvm and MRPsvm on 50 buildings' data.

6.7 Discussion

This chapter proposes a parallel implementation of SVMs for multi-core and multiprocessor systems. It implements decomposition method and utilizes SMO as inner solver. The parallelism is conducted to update the vector f in the decomposition step and is programmed in the simple, yet pragmatic programming framework Map-Reduce. A shared cache is designed to save the kernel matrix columns when the data size is very large. Firstly, we use this mechanism to implement SVC, and the extensive experiments show that our system is very efficient in solving large scale problems. For instance, the speed on 4 processors can increase to more than 4 times that of Libsvm for most of the applications. It overwhelms the state-of-the-art Pisvm in all benchmark tests in terms of both speed and memory requirement.

We also parallelize SVR for solving large scale problems in building energy analysis.

Experimental results show that the new parallel system provides the same accuracy as Libsvm does, yet performs far more efficiently than the sequential implementation in solving stated problems. On the smallest dataset, MRPsvm achieves more than twice the speedup times of Libsvm while on the largest dataset, the speedup times can increase to 16-fold and 19-fold on dual-core system and on quad-core system, respectively. Again, the proposed implementation is superior to the Pisvm in all tests in terms of both speed and memory requirement.

Since the multi-core system is dominating the trend of processor development and is highly available in the modern market, MRPsvm is potentially very practical and feasible in solving large scale regression problems. Furthermore, the success of MRPsvm indicates that Map-Reduce is a possible option to parallelize machine learning algorithms.

However, the proposed system is not yet mature, there are still several aspects worth considering for further improvements, for instance, shrinking, finding the best granularity of parallel work for a particular dataset.

7

Summary and Future Work

7.0.1 Summary

The prediction of building energy consumption is an important task in building design, retrofit, and operation. A well designed building and its energy system may lead to energy conservation and CO_2 reduction. This thesis concentrates on up-to-date artificial intelligence models by solving problems related to this application.

Firstly, we review the recently developed models on predicting building energy consumption, including elaborate and simple engineering methods, statistical methods and artificial intelligence methods, especially ANNs and SVMs. This previous work includes solving all levels energy analysis with appropriate models, optimizing model parameters, treating inputs for better performance, simplifying the problems and comparing different models. We summarize the advantages and disadvantages of each model and point out that without complete comparison under the same circumstances, it is difficult to say which one is better than the others. Among existing methods, artificial intelligence models are attracting more and more attention in the research community.

Then, this thesis attempts to apply SVMs in predicting building energy consumption. We present the SVM principles in depth, as well as some important extensions, including SVC, SVR, one-class SVM, multi-class SVM and transductive SVM. These models have demonstrated superiority in many sorts of applications, due to ideas of maximum margin, regularization and kernel method.

Before using SVMs in our application, we need to consider how to obtain historical consumption profiles. We choose to simulate them in EnergyPlus based on three considerations. First, sufficient and precise data is difficult to collect in reality. Second, our

aim is to develop high performance mathematical models instead of physical profiles of a building and its energy system. Third, EnergyPlus is a powerful simulation tool. With calibration, it can produce very precise building energy profiles. In our work, office buildings located in France are generated. We use real weather conditions as the inputs in order to make the simulated buildings more like real ones. We develop an interface to control EnergyPlus to simulate multiple buildings. The diversity of multiple buildings comes from different structure characteristics, envelope materials, occupancy, etc.

As long as historical energy consumption data is recorded, SVR models are trained and tested on this prediction. Extensive experiments are designed to test the accuracy and robustness of this model by training on different types of historical data. Very high prediction accuracy is achieved. Generally speaking, the more training data, the better model performance. However, when the testing data has similar distribution with the training data, even small training datasets can derive high performance models. For instance, the energy consumption in November is similar to that in January, the model derived from January (one month only) achieves better performance than the model derived from January-August (8 months). SVR also shows high accuracy in predicting a completely new building by involving building structure characteristics.

In the next application, we use RDP neural network model to detect and diagnose building energy faults. We simulate abnormal consumption by manually introducing performance degradation to electric devices. In the experiment, RDP model shows very high detection ability. We propose a new approach to point out the reasons for the faults. Our method is based on the evaluation of several RDP models, each of which is designed to be able to detect a particular equipment fault. These models are trained on historical faulty consumption. Our diagnostic method successfully diagnosed Chiller faults in the experiment. This method is able to sort the possible sources according to their possibilities of failure.

A new feature selection method is proposed for reducing SVR input dimension. The features are selected according to their feasibility in practice and usefulness to the predictor. The last criterion is evaluated under two filter methods: the gradient guided feature selection and the correlation coefficients. To evaluate the proposed method, we use three training datasets to evaluate performance change of the model before and after feature selection. Experimental results show that the selected subset can provide

competitive predictors. The number of features is reduced without losing model performance, making the model easier to use in practice. Performance improvement is achieved in some cases. For instance, with both RBF and polynomial kernel on the data from fifty buildings, the model accuracy increases and the learning cost decreases apparently. This work serves as the first guide for selecting an optimal subset of features when applying machine learning methods on the prediction of building energy consumption.

When the training dataset is large, the SVM training process becomes costly and the storage requirement would be very high. For this reason, a new parallel approach is proposed to optimize the SVM training. It is based on a decomposition method. The variables are optimized iteratively. The parallelism is programmed in the simple, yet pragmatic programming framework Map-Reduce. A shared cache is designed to store kernel matrix columns. This implementation is specially suitable to multi-core and multi-processor systems. We have tested both SVC and SVR in extensive experiments. The results show that our system is very efficient in solving large scale problems. It achieves high speedup with regard to the sequential implementation. The results also show superior performance of our implementation over a state-of-the-art parallel one in both training speed and memory requirement.

7.0.2 Future work

As for the application of predicting building energy consumption, there are still many research problems. The future investigations may focus on the following points:

- Develop new and more effective, robust, reliable and efficient prediction models.
- Refine elements of system level energy consumption, compare candidate models and choose the best model for each component.
- Apply the energy prediction in the Building Energy Management System (BEMS) to achieve mutual benefits.
- Establish databases and collect precise and sufficient historical consumption data from various cases for further research use.
- Develop feature selection methods for other modeling methods in building energy analysis.

- Further optimize the parallel SVM algorithm by shrinking, finding the best granularity of parallel work for a particular dataset, etc.

References

[1] European Parliament and Council. Directive 2010/31/EU of the European Parliament and of the Council of 19 May 2010 on the energy performance of buildings. *Official Journal of the European Union*, L153:13–35, 2010. 1

[2] Commissariat général au développement durable. Bilan énergétique de la france pour 2010, 2011. www.statistiques.developpement-durable.gouv.fr. 1

[3] M. Krarti. An overview of artificial intelligence-based methods for building energy systems. *Journal of Solar Energy Engineering*, 125(3):331–342, 2003. 2

[4] A. I. Dounis. Artificial intelligence for energy conservation in buildings. *Advances in Building Energy Research*, 4(1):267–299, 2010. 2

[5] R. Yao and K. Steemers. A method of formulating energy load profile for domestic buildings in the UK. *Energy and Buildings*, 37(6):663 – 671, 2005. 2, 4

[6] S. Wang and X. Xu. Simplified building model for transient thermal performance estimation using GA-based parameter identification. *International Journal of Thermal Sciences*, 45(4):419 – 432, 2006. 5, 13

[7] S. Karatasou, M. Santamouris, and V. Geros. Modeling and predicting building's energy use with artificial neural networks: Methods and results. *Energy and Buildings*, 38(8):949 – 958, 2006. 11, 14, 78

[8] J. Liang and R. Du. Model-based fault detection and diagnosis of HVAC systems using support vector machine method. *International Journal of Refrigeration*, 30(6):1104 – 1114, 2007. 2, 12, 62, 63

[9] J. A. Clarke. *Energy Simulation in Building Design (2nd Edition)*. Butterworth-Heinemann, Oxford, 2001. 3

[10] F. C. McQuiston, J. D. Parker, and J. D. Spitler. *Heating, Ventilating and Air Conditioning Analysis and Design*. Wiley, 6 edition, 2005. 3

[11] ISO 13790:2008. *Energy performance of buildings Calculation of energy use for space heating and cooling*. ISO, Geneva, Switzerland, 2008. 3

[12] Building energy software tools directory, 2011. Available online at: http://apps1.eere.energy.gov/buildings/tools_directory/ (Accessed March 2011). 3, 42

[13] M. S. Al-Homoud. Computer-aided building energy analysis techniques. *Building and Environment*, 36(4):421 – 433, 2001. 3, 4, 5

[14] D. B. Crawley, J. W. Hand, M. Kummert, and B. T. Griffith. Contrasting the capabilities of building energy performance simulation programs. *Building and Environment*, 43(4):661 – 673, 2008. Part Special: Building Performance Simulation. 3

[15] J. A. White and R. Reichmuth. Simplified method for predicting building energy consumption using average monthly temperatures. In *Proceedings of the 31st Intersociety Energy Conversion Engineering Conference*, volume 3, pages 1834 – 1839, 1996. 4

[16] F. S. Westphal and R. Lamberts. The use of simplified weather data to estimate thermal loads of non-residential buildings. *Energy and Buildings*, 36(8):847 – 854, 2004. 4

[17] A. Rice, S. Hay, and D. Ryder-Cook. A limited-data model of building energy consumption. In *Proceedings of the 2nd ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings*, pages 67–72, 2010. 4

[18] C. S. Barnaby and J. D. Spitler. Development of the residential load factor method for heating and cooling load calculations. *ASHRAE Transactions*, 111(1):291 – 307, 2005. 4

[19] F. W. H. Yik, J. Burnett, and I. Prescott. Predicting air-conditioning energy consumption of a group of buildings using different heat rejection methods. *Energy and Buildings*, 33(2):151 – 166, 2001. 5

[20] Y. Pan, Z. Huang, and G. Wu. Calibrated building energy simulation and its application in a high-rise commercial building in shanghai. *Energy and Buildings*, 39(6):651 – 657, 2007. 5

[21] A. Reddy. Literature review on calibration of building energy simulation programs : Uses, problems, procedures, uncertainty, and tools. *ASHRAE transactions*, 112(2):226 – 240, 2006. 5, 42

[22] D. B. Crawley, L. K. Lawrie, F. C. Winkelmann, W. F. Buhl, Y. J. Huang, C. O. Pedersen, R. K. Strand, R. J. Liesen, D. E. Fisher, M. J. Witte, and J. Glazer. Energyplus: creating a new-generation building energy simulation program. *Energy and Buildings*, 33(4):319 – 331, 2001. 5

[23] M. Bauer and J. L. Scartezzini. A simplified correlation method accounting for heating and cooling loads in energy-efficient buildings. *Energy and Buildings*, 27(2):147 – 154, 1998. 6, 14

[24] K-E. Westergren, H. Höberg, and U. Norlen. Monitoring energy consumption in single-family houses. *Energy and Buildings*, 29(3):247 – 257, 1999.

[25] J. Pfafferott, S. Herkel, and J. Wapler. Thermal building behaviour in summer: long-term data evaluation using simplified models. *Energy and Buildings*, 37(8):844 – 852, 2005. 6

[26] F. A. Ansari, A. S. Mokhtar, K. A. Abbas, and N. M. Adam. A simple approach for building cooling load estimation. *American Journal of Environmental Sciences*, 1(3):209 – 212, 2005. 6, 14

[27] A. Dhar, T. A. Reddy, and D. E. Claridge. Modeling hourly energy use in commercial buildings with fourier series functional form. *ASME Journal of Solar Energy Engineering*, 120:217 – 223, 1998. 6, 14

[28] A. Dhar, T. A. Reddy, and D. E. Claridge. A fourier series model to predict hourly heating and cooling energy use in commercial buildings with outdoor temperature as the only weather variable. *Journal of Solar Energy Engineering*, 121:47 – 53, 1999. 6, 14

[29] F. Lei and P. Hu. A baseline model for office building energy consumption in hot summer and cold winter region. In *Proceedings of International Conference on Management and Service Science*, pages 1 – 4, 2009. 6, 14

[30] Y. Ma, J q. Yu, C y. Yang, and L. Wang. Study on power energy consumption model for large-scale public building. In *Proceedings of the 2nd International Workshop on Intelligent Systems and Applications*, pages 1 – 4, 2010. 6, 14

[31] S-H. Cho, W-T. Kim, C-S. Tae, and M. Zaheeruddin. Effect of length of measurement period on accuracy of predicted annual heating energy consumption of buildings. *Energy Conversion and Management*, 45(18-19):2867 – 2878, 2004. 6, 14

[32] J. C. Lam, K. K. W. Wan, S. L. Wong, and T. N. T. Lam. Principal component analysis and long-term building energy simulation correlation. *Energy Conversion and Management*, 51(1):135 – 139, 2010. 6, 14

[33] C. Ghiaus. Experimental estimation of building energy performance by robust regression. *Energy and Buildings*, 38(6):582 – 587, 2006. 7, 14

[34] M. J. Jiménez and M. R. Heras. Application of multi-output arx models for estimation of the u and g values of building components in outdoor testing. *Solar Energy*, 79(3):302 – 310, 2005. 7, 14

[35] A. Kimbara, S. Kurosu, R. Endo, K. Kamimura, T. Matsuba, and A. Yamada. On-line prediction for load profile of an air-conditioning system. *ASHRAE Transactions*, 101(2):198 – 207, 1995. 7

[36] A. J. Hoffman. Peak demand control in commercial buildings with target peak adjustment based on load forecasting. In *Proceedings of the 1998 IEEE International Conference on Control Applications*, volume 2, pages 1292 – 1296, 1998. 7, 14

[37] G. R. Newsham and B. J. Birt. Building-level occupancy data to improve ARIMA-based electricity use forecasts. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, BuildSys '10, pages 13–18, New York, NY, USA, 2010. ACM. 7, 14

[38] M. Aydinalp-Koksal and V. I. Ugursal. Comparison of neural network, conditional demand analysis, and engineering approaches for modeling end-use energy consumption in the residential sector. *Applied Energy*, 85(4):271 – 296, 2008. 7, 11

[39] G Lafrance and D. Perron. Evolution of residential electricity demand by end-use in Quebec 1979-1989: a conditional demand analysis. *Energ Stud Rev*, 6(2):164 – 173, 1994. 7

[40] S. A. Kalogirou. Artificial neural networks in energy applications in buildings. *International Journal of Low-Carbon Technologies*, 1(3):201–216, 2006. 7

[41] S. A. Kalogirou, C. C. Neocleous, and C. N. Schizas. Building heating load estimation using artificial neural networks. In *Proceedings of the 17th international conference on Parallel architectures and compilation techniques*, 1997. 8, 14

[42] B. B. Ekici and U. T. Aksoy. Prediction of building energy consumption by using artificial neural networks. *Advances in Engineering Software*, 40(5):356 – 362, 2009. 8, 10, 14

[43] T. Olofsson, S. Andersson, and R. Östin. A method for predicting the annual building heating demand based on limited performance data. *Energy and Buildings*, 28(1):101 – 108, 1998. 8, 10, 14

[44] T. Olofsson and S. Andersson. Long-term energy demand predictions based on short-term measured data. *Energy and Buildings*, 33(2):85 – 91, 2001. 8, 10, 14

[45] R. Yokoyama, T. Wakui, and R. Satake. Prediction of energy demands using neural network with model identification by global optimization. *Energy Conversion and Management*, 50(2):319 – 327, 2009. 8, 10, 14, 75

[46] J. F. Kreider, D. E. Claridge, P. Curtiss, R. Dodier, J. S. Haberl, and M. Krarti. Building energy use prediction and system identification using recurrent neural networks. *Journal of Solar Energy Engineering*, 117(3):161–166, 1995. 8, 9, 14

[47] A. E. Ben-Nakhi and M. A. Mahmoud. Cooling load prediction for buildings using general regression neural networks. *Energy Conversion and Management*, 45(13-14):2127 – 2141, 2004. 8, 9, 10, 14, 75

[48] S. A. Kalogirou and M. Bojic. Artificial neural networks for the prediction of the energy consumption of a passive solar building. *Energy*, 25(5):479 – 491, 2000. 8, 14, 41

[49] C-w. Yan and J. Yao. Application of ann for the prediction of building energy consumption at different climate zones with HDD and CDD. In *Proceedings of 2010 2nd International Conference on Future Computer and Communication*, volume 3, pages 286 – 289, 2010. 8, 14

[50] Joint Center for Energy Management (JCEM). Final report: Artificial neural networks applied to LoanSTAR data. Technical Report TR/92/15, 1992. 8, 14

[51] S. S. A. K. Javeed Nizami and A. Z. Al-Garni. Forecasting electric energy consumption using neural networks. *Energy Policy*, 23(12):1097–1104, December 1995. 8

[52] P. A. González and J. M. Zamarreno. Prediction of hourly energy consumption in buildings based on a feedback artificial neural network. *Energy and Buildings*, 37(6):595 – 601, 2005. 8, 10, 14

[53] A. Azadeh, S. F. Ghaderi, and S. Sohrabkhani. Annual electricity consumption forecasting by neural network in high energy consuming industrial sectors. *Energy Conversion and Management*, 49(8):2272 – 2278, 2008. 9, 11, 14, 75

[54] S. L. Wong, K. K. W. Wan, and T. N. T. Lam. Artificial neural networks for energy analysis of office buildings with daylighting. *Applied Energy*, 87(2):551 – 557, 2010. 9, 14

[55] Z. Hou, Z. Lian, Y. Yao, and X. Yuan. Cooling-load prediction by the combination of rough set theory and an artificial neural-network based on data-fusion technique. *Applied Energy*, 83(9):1033 – 1046, 2006. 9, 10, 14, 41

[56] W-Y. Lee, J. M. House, and N-H. Kyong. Subsystem level fault diagnosis of a building's air-handling unit using general regression neural networks. *Applied Energy*, 77(2):153 – 170, 2004. 9, 14, 62

[57] M. Aydinalp, V. I. Ugursal, and A. S. Fung. Modeling of the appliance, lighting, and space-cooling energy consumptions in the residential sector using neural networks. *Applied Energy*, 71(2):87 – 110, 2002. 9, 11, 14

[58] M. Aydinalp, V. I. Ugursal, and A. S. Fung. Modeling of the space and domestic hot-water heating energy-consumption in the residential sector using neural networks. *Applied Energy*, 79(2):159 – 178, 2004. 9, 14

[59] A. E. Ben-Nakhi and M. A. Mahmoud. Energy conservation in buildings through efficient A/C control using neural networks. *Applied Energy*, 73(1):5 – 23, 2002. 9, 14

[60] M. Yalcintas and S. Akkurt. Artificial neural networks applications in building energy predictions and a case study for tropical climates. *International Journal of Energy Research*, 29(10):891 – 901, 2005. 9, 14

[61] M. Yalcintas. Energy-savings predictions for building-equipment retrofits. *Energy and Buildings*, 40(12):2111 – 2120, 2008. 9, 14

[62] M. M. Gouda, S. Danaher, and C. P. Underwood. Application of an artificial neural network for modelling the thermal dynamics of a building's space and its heating system. *Mathematical and Computer Modelling of Dynamical Systems: Methods, Tools and Applications in Engineering and Related Sciences*, 8(3):333 – 344, 2002. 9, 14

[63] T. Olofsson and S. Andersson. Analysis of the interaction between heating and domestic load in occupied single-family buildings. In *Proceedings of the 5th Symposium on Building Physics in the Nordic Countries*, pages 473 – 480, 1999. 9, 14

[64] T. Olofsson and S. Andersson. Overall heat loss coefficient and domestic energy gain factor for single-family buildings. *Building and Environment*, 37(11):1019 – 1026, 2002. 14

[65] M. Lundin, S. Andersson, and R. Östin. Validation of a neural network method for estimation heat loss and domestic gain in buildings. In *Proceedings of the 6th Symposium on Building Physics in the Nordic Countries*, page 325332, 2002. 14

[66] M. Lundin, S. Andersson, and R. Östin. Development and validation of a method aimed at estimating building performance parameters. *Energy and Buildings*, 36(9):905 – 914, 2004. 9, 14, 41

[67] R. Zmeureanu. Prediction of the COP of existing rooftop units using artificial neural networks and minimum number of sensors. *Energy*, 27(9):889 – 904, 2002. 9, 14

[68] M. Yalcintas. An energy benchmarking model based on artificial neural network method with a case example for tropical climates. *International Journal of Energy Research*, 30(14):1158 – 1174, 2006. 10, 14

[69] M. Yalcintas and U. Aytun Ozturk. An energy benchmarking model based on artificial neural network method utilizing us commercial buildings energy consumption survey (CBECS) database. *International Journal of Energy Research*, 31(4):412 – 421, 2007. 10, 14, 41

[70] J. F. Kreider and J. S Haberl. Predicting hourly building energy use: the great energy predictor shootout–overview and discussion of results. *ASHRAE Transactions*, 100:1104–1118, 1994. 10, 14

[71] J. Yang, H. Rivard, and R. Zmeureanu. On-line building energy prediction using adaptive artificial neural networks. *Energy and Buildings*, 37(12):1250 – 1259, 2005. 10, 14, 78

[72] N. Kubota, S. Hashimoto, F. Kojima, and K. Taniguchi. GP-preprocessed fuzzy inference for the energy load prediction. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 1 – 6, 2000. 10, 14

[73] A. Kusiak, M. Li, and Z. Zhang. A data-driven approach for steam load prediction in buildings. *Applied Energy*, 87(3):925 – 933, 2010. 11, 14

[74] S. Kajl, R. Poulin, P. Malinowski, and M. A. Roberge. Fuzzy assistant for evaluation of building energy consumption. In *Proceedings of International Fuzzy Systems and Intelligent Control Conference*, pages 67 – 74, 1996. 11, 14

[75] S. Kajl, M. A. Roberge, L. Lamarche, and P. Malinowski. Evaluation of building energy consumption based on fuzzy logic and neural networks applications. In *Proc of CLIMA 2000 Conf*, page 264, 1997. 11, 14

[76] A. H. Neto and F. A. S. Fiorelli. Comparison between detailed model simulation and artificial neural network for forecasting building energy consumption. *Energy and Buildings*, 40(12):2169–2176, 2008. 11, 42, 75

[77] B. Dong, C. Cao, and S. E. Lee. Applying support vector machines to predict building energy consumption in tropical region. *Energy and Buildings*, 37(5):545–553, 2005. 12, 14, 75

[78] F. Lai, F. Magoulès, and F. Lherminier. Vapnik's learning theory applied to energy consumption forecasts in residential buildings. *International Journal of Computer Mathematics*, 85(10):1563–1588, 2008. 12, 14, 75

[79] Q. Li, Q. L. Meng, J. J. Cai, Y. Hiroshi, and M. Akashi. Applying support vector machine to predict hourly cooling load in the building. *Applied Energy*, 86(10):2249–2256, 2009. 12, 14

[80] Z. Hou and Z. Lian. An application of support vector machines in cooling load prediction. In *Proceedings of International Workshop on Intelligent Systems and Applications*, pages 1 – 4, 2009. 12, 14

[81] Q. Li, P. Ren, and Q. Meng. Prediction model of annual energy consumption of residential buildings. In *Proceedings of 2010 International Conference on Advances in Energy Engineering*, pages 223 – 226, 2010. 12, 14

[82] J. Lv, X. Li, L. Ding, and L. Jiang. Applying principal component analysis and weighted support vector machine in building cooling load forecasting. In *Proceedings of 2010 International Conference on Computer and Communication Technologies in Agriculture Engineering*, pages 434 – 437, 2010. 13, 14

[83] X. Li, L. Ding, J. Lv, G. Xu, and J. Li. A novel hybrid approach of kpca and svm for building cooling load prediction. In *Proceedings of 2010 Third International Conference on Knowledge Discovery and Data Mining*, pages 522 – 526, 2010. 13, 14

[84] X. Li, Y. Deng, L. Ding, and L. Jiang. Building cooling load forecasting using fuzzy support vector machine and fuzzy c-mean clustering. In *Proceedings of 2010 International Conference on Computer and Communication Technologies in Agriculture Engineering*, pages 438 – 441, 2010. 13, 14

[85] Y m. Zhang and W g. Qi. Interval forecasting for heating load using support vector regression and error correcting markov chains. In *Proceedings of the Eighth International Conference on Machine Learning and Cybernetics*, pages 1106 – 1110, 2009. 13, 14

[86] X. Wang, Z. Chen, C. Yang, and Y. Chen. Gray predicting theory and application of energy consumption of building heat-moisture system. *Building and Environment*, 34(4):417 – 420, 1999. 13

[87] J. J. Guo, J. Y. Wu, and R. Z. Wang. A new approach to energy consumption prediction of domestic heat pump water heater based on grey system theory. *Energy and Buildings*, 43(6):1273–1279, 2011. 13

[88] Q. Zhou, S. Wang, X. Xu, and F. Xiao. A grey-box model of next-day building thermal load prediction for energy-efficient control. *International Journal of Energy Research*, 32(15):1418–1431, 2008. 13

[89] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995. 19

[90] B. Schölkopf and Alexander J. S. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002. 24, 33

[91] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. Support vector regression machines. In *NIPS*, pages 155–161, 1996. 24

[92] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13:1443–1471, July 2001. 27

[93] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998. 28, 30

[94] J. Friedman. Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University, 1996. 29

[95] U. H.-G. KreBel. *Pairwise classification and support vector machines*, pages 255–268. MIT Press, Cambridge, MA, USA, 1999. 29

[96] T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, pages 507–513, Cambridge, MA, USA, 1998. MIT Press. 29

[97] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61–74, 1999. 29

[98] V. Roth. Probabilistic discriminative kernel classifiers for multi-class problems. In *DAGM Symposium Symposium for Pattern Recognition*, pages 246–253, 2001. 29

[99] K-B Duan and S. S. Keerthi. Which Is the Best Multiclass SVM Method? An Empirical Study. Technical Report CD-03-12, Department of Mechanical Engineering, National University of Singapore, 2003. 29

[100] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, May 2000. 29

[101] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992. 33

[102] S. J. Wright. *Primal-Dual Interior-Point Methods*. siam, 1997. 33, 36

[103] T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, page 116, New York, NY, USA, 2004. ACM. 36

[104] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *Proceedings of the 24th International Conference on Machine Learning*, pages 807–814, 2007. 37, 38, 87

[105] S. Shalev-Shwartz and N. Srebro. SVM optimization : Inverse dependence on training set size. *Communications*, pages 928–935, 2008. 37

[106] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE. 39

[107] F. Sebastiani and C. Nazionale D. Ricerche. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002. 39

[108] W. S. Noble. *Support vector machine applications in computational biology*, chapter 3. Computational molecular biology. MIT Press, 2004. 39

[109] A. Ben-Hur, C. S. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch. Support vector machines and kernels for computational biology. *PLoS Computational Biology*, 4(10), 10 2008. 39

[110] T. Jaakkola, M. Diekhans, and D. Haussler. Using the fisher kernel method to detect remote protein homologies. In *Intelligent Systems in Molecular Biology*, pages 149–158, 1999. 39

[111] C. H. Q. Ding and I. Dubchak. Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics/computer Applications in The Biosciences*, 17:349–358, 2001. 39

[112] B. Logan, P. Moreno, B. Suzek, Z. Weng, and S. Kasif. A study of remote homology detection. Technical report, Cambridge Research Laboratory, 2001. 39

[113] L. Liao and W. S. Noble. Combining Pairwise Sequence Similarity and Support Vector Machines for Detecting Remote Protein Evolutionary and Structural Relationships. *Journal of Computational Biology*, 10(6):857–868, 2003. 39

[114] L. Zhang, F. Lin, and B. Zhang. Support vector machine learning for image retrieval. In *International Conference on Image Processing*, pages 721–724, 2001. 40

[115] Energyplus, 2011. Available online at: http://www.EnergyPlus.gov (Accessed July 2011). 44, 46

[116] SVM software, 2011. Available online at: http://www.support-vector-machines.org/SVM_soft.html (Accessed July 2011). 52

[117] T. Joachims. Making large-scale support vector machine learning practical. *Advances in kernel methods: support vector learning*, pages 169–184, 1999. 52, 87, 90

[118] C. C. Chang and C. J. Lin. *LIBSVM: a library for support vector machines*, 2001. Available online at http://www.csie.ntu.edu.tw/~cjlin/libsvm. 52, 88, 90, 91

[119] I. W. Tsang, J. T. Kwok, and K. T. Lai. Core vector regression for very large regression problems. In *ICML'05: Proceedings of the 22nd International Conference on Machine Learning*, pages 912–919, New York, NY, USA, 2005. ACM. 54

[120] W. Y. Lee, J. M. House, and D. R. Shin. Fault diagnosis and temperature sensor recovery for an air-handling unit. *ASHRAE Transactions*, 103(1):621–633, 1997. 62

[121] J. M. House, W. Y. Lee, and D. R. Shin. Classification techniques for fault detection and diagnosis of an air handling unit. *ASHRAE Transactions*, pages 1087–1097, 1999. 62

[122] A. L. Dexter and D. Ngo. Fault diagnosis in air-conditioning systems: a multi-step fuzzy model-based approach. *HVAC&R Research*, 7(1):83 – 102, 2001. 62

[123] J. Du, M. J. Er, and L. Rutkowski. Fault diagnosis of an air-handling unit system using a dynamic fuzzy-neural approach. In *Proceedings of the 10th international conference*

on Artificial intelligence and soft computing: Part I, ICAISC'10, pages 58–65, Berlin, Heidelberg, 2010. Springer-Verlag. 62

[124] S. Wang and F. Xiao. AHU sensor fault diagnosis using principal component analysis method. *Energy and Buildings*, 36(2):147 – 160, 2004. 62

[125] J. Qin and S. Wang. A fault detection and diagnosis strategy of VAV air-conditioning systems for improved energy and control performances. *Energy and Buildings*, 37(10):1035 – 1048, 2005. 62

[126] F. Xiao, S. Wang, and J. Zhang. A diagnostic tool for online sensor health monitoring in air-conditioning systems. *Automation in Construction*, 15(4):489 – 503, 2006. The first conference on the Future of the AEC Industry (BFC05). 62

[127] F. Xiao, S. Wang, X. Xu, and G. Ge. An isolation enhanced PCA method with expert-based multivariate decoupling for sensor FDD in air-conditioning systems. *Applied Thermal Engineering*, 29(4):712 – 722, 2009. 62

[128] Z. J. Hou, Z. W. Lian, H. Yao, and X. J. Yuan. Data mining based sensor fault diagnosis and validation for building air conditioning system. *Energy Conversion and Management*, 47:2479–2490, 2006. 62

[129] M. Kim and M. S. Kim. Performance investigation of a variable speed vapor compression system for fault detection and diagnosis. *International Journal of Refrigeration*, 28(4):481 – 488, 2005. 62

[130] S. A. Tassou and I. N. Grace. Fault diagnosis and refrigerant leak detection in vapour compression refrigeration systems. *International Journal of Refrigeration*, 28(5):680 – 688, 2005. 62

[131] J. E. Braun. Automated fault detection and diagnostics for vapor compression cooling equipment. *ASME Transactions: Journal of Solar Energy Engineering*, 125:266–274, 2003. 62

[132] M. Liu, L. Song, and D. E. Claridge. Development of whole-building fault detection methods. *High Performance Commerical Building Systems*, 2002. 62, 63

[133] S-H. Cho, H-C. Yang, M. Zaheer-uddin, and B-C. Ahn. Transient pattern analysis for fault detection and diagnosis of HVAC systems. *Energy Conversion and Management*, 46(18-19):3103 – 3116, 2005. 62

[134] J. Schein, S. T. Bushby, N. S. Castro, and J. M. House. A rule-based fault detection method for air handling units. *Energy and Buildings*, 38(12):1485 – 1492, 2006. 62

[135] M. Tajine and D. Elizondo. The recursive deterministic perceptron neural network. *Neural Networks*, 11:153–170, 1998. 63

[136] D Elizondo. The linear separability problem: some testing methods. *Neural Networks, IEEE Transactions on*, 17(2):330 – 344, 2006. 64

[137] D. A. Elizondo, J. M. Ortiz de Lazcano-Lobato, and R. Birkenhead. Choice effect of linear separability testing methods on constructive neural network algorithms: An empirical study. *Expert Systems with Applications*, 38(3):2330 – 2346, 2011. 64

[138] M. Tajine and D. Elizondo. Growing methods for constructing recursive deterministic perceptron neural networks and knowledge extraction. *Artificial Intelligence*, 102(2):295 – 322, 1998. 64

[139] R. Rosipal, M. Girolami, and L. J. Trejo. Kernel PCA for Feature Extraction and De-Noising in Non-Linear Regression. *Neural Computing & Applications*, 10:231–243, 2001. 74

[140] B. Schölkopf, A. Smola, and K-R Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, July 1998. 74

[141] C. Liu and H. Wechsler. Comparative assessment of independent component analysis (ICA) for face recognition. In *International Conference on Audio and Video Based Biometric Person Authentication*, pages 22–24, 1999. 74

[142] L. J. Cao, K. S. Chua, W. K. Chong, H. P. Lee, and Q. M. Gu. A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine. *Neurocomputing*, 55(1-2):321–336, 2003. 74

[143] Y. Qi, D. Doermann, and D. DeMenthon. Hybrid independent component analysis and support vector machine learning scheme for face detection. In *International Conference on Acoustics, Speech, and Signal Processing* , 2001. 74

[144] O. Déniz, M. Castrillón, and M. Hernández. Face recognition using independent component analysis and support vector machines. *Pattern Recognition Letters*, 24:2153–2157, 2003. 74

[145] M. O. Faruqe and M. A. M. Hasan. Face recognition using PCA and SVM. In *Proceedings of the 3rd international conference on Anti-Counterfeiting, security, and identification in communication*, ASID'09, pages 97–101, Piscataway, NJ, USA, 2009. IEEE Press. 74

[146] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems*, volume 13, pages 668–674, 2000. 75

[147] H Fröhlic and A Zell. Feature subset selection for support vector machines by incremental regularized risk minimization. In *2004 IEEE International Joint Conference on Neural Networks*, volume 3, pages 2041–2045. IEEE Press, 2004. 75

[148] C. Gold, A. Holub, and P. Sollich. Bayesian approach to feature selection and parameter tuning for support vector machine classifers. *Neural Networks*, 18(5-6):693–701, 2005. 75

[149] O. L. Mangasarian and G. Kou. Feature selection for nonlinear kernel support vector machines. In *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, ICDMW '07, pages 231–236, Washington, DC, USA, 2007. IEEE Computer Society. 75

[150] H. Madsen and J. Holst. Estimation of continuous-time models for the heat dynamics of a building. *Energy and Buildings*, 22(1):67–79, 1995. 75

[151] C. A. Maia and M. M. Gonçalves. A methodology for short-term electric load forecasting based on specialized recursive digital filters. *Computers and Industrial Engineering*, 57(3):724–731, 2009. 75

[152] G. K. F. Tso and K. K. W. Yau. Predicting electricity energy consumption: a comparison of regression analysis, decision tree and neural networks. *Energy*, 32(9):1761–1768, 2007. 75

[153] S. L. Wong, Kevin K. W. Wan, and Tony N. T. Lam. Artificial neural networks for energy analysis of office buildings with daylighting. *Applied Energy*, 87(2):551 – 557, 2010. 75

[154] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003. 76, 79

[155] A. Navot, L. Shpigelman, N. Tishby, and E. Vaadia. Nearest neighbor based feature selection for regression and its application to neural activity. In Y. Weiss, B. Scholkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 995–1002. MIT Press, 2006. 76, 77

[156] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997. 87, 89

[157] E. Y. Chang, K. Zhu, H. Wang, H. Bai, J. Li, Z. Qiu, and H. Cui. PSVM: Parallelizing support vector machines on distributed computers. In *NIPS*, volume 20, 2007. 87, 88

[158] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods: support vector learning*, pages 185–208, 1999. 87, 90, 91, 94

[159] D. Brugger. Parallel support vector machines. In *Proceedings of the IFIP International Conference on Very Large Scale Integration of System on Chip*, 2007. 88

[160] Z. A. Zhu, W. Z. Chen, G. Wang, C. G. Zhu, and Z. Chen. P-packSVM: Parallel primal gradient descent kernel SVM. In *Proceedings of the 9th IEEE International Conference on Data Mining*, pages 677–686, 2009. 88

[161] D. Bickson, E. Yom-tov, and D. Dolev. A gaussian belief propagation solver for large scale support vector machines. In *Proceedings of the 5th European Conference on Complex Systems*, 2008. 88

[162] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, and V. Vapnik. Parallel support vector machines: The cascade SVM. In *In Advances in Neural Information Processing Systems*, volume 17, pages 521–528, 2005. 88

[163] J. X. Dong, A. Krzyzak, and C. Y. Suen. Fast SVM training algorithm with decomposition on very large data sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):603–618, 2005. 88

[164] L. Zanni, T. Serafini, and G. Zanghirati. Parallel software for training large scale support vector machines on multiprocessor systems. *Journal of Machine Learning Research*, 7:1467–1492, 2006. 88, 91, 92

[165] T. Hazan, A. Man, and A. Shashua. A parallel decomposition solver for SVM: Distributed dual ascend using fenchel duality. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pages 1–8, 2008. 89

[166] Y. Lu and V. Roychowdhury. Parallel randomized sampling for support vector machine (SVM) and support vector regression (SVR). *Knowledge and Information Systems*, 14:233–247, 2008. 89

[167] L. J. Cao, S. S. Keerthi, C. J. Ong, J. Q. Zhang, U. Periyathamby, J. F. Xiu, and H. P. Lee. Parallel sequential minimal optimization for the training of support vector machines. *IEEE Transactions on Nueral Networks*, 17(4):1039–1049, 2006. 89

[168] B. Catanzaro, N. Sundaram, and K. Keutzer. Fast support vector machine training and classification on graphics processors. In *Proceedings of the 25th International Conference on Machine Learning*, pages 104–111, 2008. 89

[169] R. E. Fan, P. H. Chen, and C. J. Lin. Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research*, 6:1889–1918, 2005. 91

[170] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008. 92

[171] C. T. Chu, S. K. Kim, Y. A. Lin, Y. Yu, G. R. Bradski, A. Y. Ng, and K. Olukotun. Map-reduce for machine learning on multicore. In *NIPS*, pages 281–288, 2006. 93

[172] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski, and C. Kozyrakis. Evaluating mapreduce for multi-core and multiprocessor systems. In *Proceedings of the IEEE 13th International Symposium on High Performance Computer Architecture*, pages 13–24, 2007. 93

[173] H. X. Zhao and F. Magoulès. Parallel support vector machines applied to the prediction of multiple buildings energy consumption. *Journal of Algorithms & Computational Technology.*, 4(2):231–249, 2010. 95