MACHINE LEARNING APPLIED IN 2D PARASITIC EXTRACTION

A Thesis

by

ZHIXING LI

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Weiping Shi
Committee Members,	Peng Li
	Anxiao Jiang
Head of Department,	Chanan Singh

December 2014

Major Subject: Computer Engineering

ABSTRACT

With the scale of interconnect number grows to billions, parasitic capacitance extraction speed is an important issue for fast turn-around time for designers.

In this thesis, we propose to build a regression model for the input interconnect geometry to predict the parasitic capacitance based on machine learning. A simplification algorithm is proposed to reduce the number of conductors for quicker and easier regression modeling and the regression models can improve by machine learning technique.

Experimental results show that the proposed method is significantly faster than existing method and provides satisfactory accuracy.

DEDICATION

To my parents

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Weiping Shi, and my committee members, Dr. Peng Li, Dr. Anxiao Jiang, for their guidance and support throughout the course of this research.

Thanks also go to my colleague, Yuhan Zhou for discussion about the field solver and capacitance extraction with me. I also want to extend my gratitude to Hao Yang, Honghuang Lin, and Qian Wang for discussion on the SVM stuff, and to Hao He, Gongming Yang who encouraged me to keep fighting and never give up.

Finally, thanks to my mother and father who gave their all to me without asking anything in return.

NOMENCLATURE

SVM	Support Vector Machine
FinFET	Fin-shape Field Effect Transistor
EDA	Electronic Design Automation
RC	Resistor Capacitor
SVR	Support Vector Regression
SV	Support Vector

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER I

INTRODUCTION

1.1 Overview

With the device structure changing to FinFET, industry successfully scales the feature size down to 14 nanometers and the next-generation chip from Intel, called Broadwell as shown in Figure 1, which has several billions of transistors on the chip, is going to be released this October.

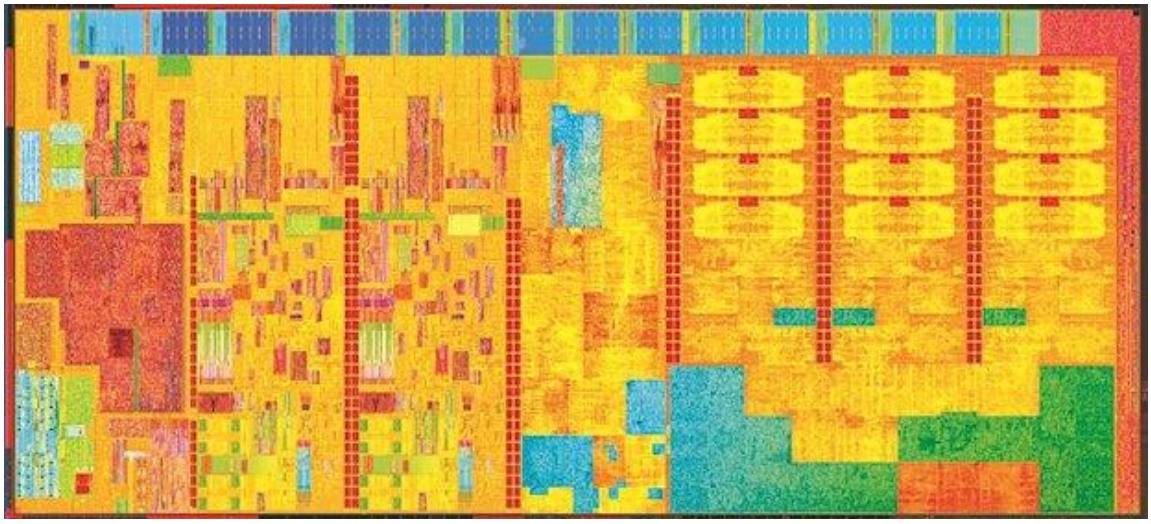

Figure 1. Intel's Broadwell chip with 14nm process technology **[1]**

Several years ago, semiconductor industry followed the Moore's Law to shrink the feature size to increase the speed of the circuits. However, nowadays, the size is being reduced for placing more transistors to build multiple cores to boost the chip

performance by parallel processing technique. Thus it can be seen that an explosive growth in the numbers of transistors on the chip is happening. In order to match the growing number of transistors, interconnect wires also increase to billion scale, which brings in new challenges for the chip designers and EDA tools.

1.2 Interconnect

What is interconnect? For a single chip, transistors on the silicon substrate need to be connected by different layers of metal wires with complicated structure which is called interconnect for function realization and signal communication.

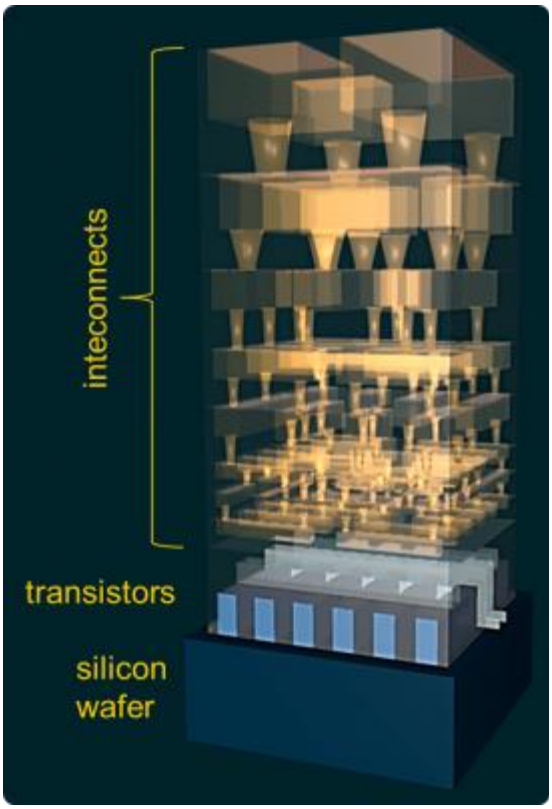

Figure 2. 3D illustration of interconnect in a chip **[2]**

This large and dense structure as illustrated in Figure 2 is interconnect and it can be easily seen that the maze-like interconnect metal wires take up most part of the chip. However, this cannot indicate the importance of interconnect for the chip design clearly.

1.3 Interconnect RC Delay

Made by metal, interconnect wire has its resistance R and capacitance C as drawn in Figure 3, which forms the RC delay τ for the signal processing. The RC delay τ is directly proportional to RC value and it is an important part of the total delay.

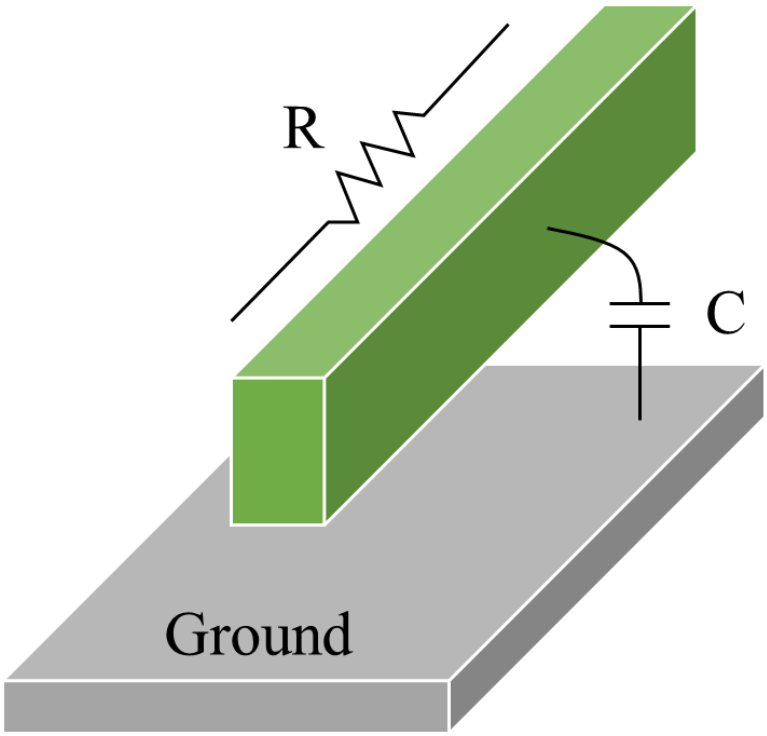

Figure 3. Interconnect metal wire with ground

In order to find the critical path of the chip to determine the circuit performance, parasitic total capacitance of all nets should be extracted fast and accurately so that the largest total delay can be calculated by sum of possible net delays on the path.

1.4 Problem Statement

For billion-scale interconnect wires, full-chip interconnect parasitic extraction becomes extremely time consuming. No chip designer is willing to wait several days only for the accurate extraction result. Shorter design cycle means commercial success. That is why modeling for fast parasitic capacitance extraction is so important.

Low extraction speed also leads to zero tolerance on the accuracy issue because any failure on result accuracy is a huge loss on time and money. Due to the smaller width and larger height of the interconnect wire, fringing capacitance is no longer the second order effect. Thus the old extraction method cannot guarantee the accuracy any more. Then machine learning is introduced to improve the modeling accuracy.

In next chapter, the parasitic extraction background will be presented and different kinds of methods are listed and discussed.

CHAPTER II

BACKGROUND

What are the existing methods applied for the parasitic extraction? Is there any method which is both efficient and accurate? As mentioned above, efficiency and accuracy are the most important two factors for full-chip parasitic extraction. However, there is an obvious trade-off which divides the existing methods into two basic categories: field solver and rule-based extraction.

2.1 Field Solver

Given enough information of the input interconnect, including the conductor geometry and process technology, the field solver can build a matrix equation for voltage potentials based on the Poisson equation with boundary conditions. After solving the matrix equation to get the voltage potential distribution, the parasitic capacitance is calculated by the cumulated charges on the conductor. For the field solver, there are many different numerical methods to set up the matrix equation:

- Finite Difference Method
- Finite Element Method
- Boundary Element Method

Without doubt, field solver is the parasitic extraction tool with the highest accuracy and it is taken as the industry parasitic extraction golden standard. Because of the heavy numerical computation work load to guarantee the accuracy, the extraction speed is

really low. So the field solver is only applied in small cases or cells for extraction reference and it is impossible to handle the full-chip extraction directly in a short time.

2.2 Rule-Based Extraction

For the rule-based extraction, as the name indicates, the layout rule file or the pattern matching technique are usually adopted to identify very basic interconnect structure for capacitance extraction and the result is the sum of all the pattern capacitance. These pre-computed methods are able to speed up the extraction tremendously compared with the field solver so that the parasitic extraction can be finished in an acceptable time period with an acceptable accuracy. Three industry-widely-used methods are listed below.

2.2.1 Look-up Table Method

Fine-meshed look-up tables are generated by field solver based on different possible interconnect patterns for capacitance calculation before the extraction. For the corresponding pattern, parasitic capacitance is read directly from the table or calculated by interpolation technique. For the non-matching pattern, field solver is called to generate the reference value and the look-up table is updated. The more data in the table, the more accurate the interpolation will be. That is the reason this method usually consumes large amount of disk resources and costs a lot of preparation time before extraction begins.

2.2.2 Analytical Formula Method

Many research works have been done in the analytical capacitance modeling. This method is based on the integration analysis of electrical field lines over the

conductor areas. Many models for different possible patterns have been presented and compared in [3], including the single line and parallel line formula [4]. The preset formula is used to calculate parasitic capacitance of interconnect pattern, however, this method is no longer accurate enough for the fringe capacitance increasing and the interconnect structure goes more complex and denser today.

2.2.3 Polynomial Fitting Method [5]

In this method, interconnect pattern is neglected and all the parameters related to the parasitic capacitance of input interconnect are collected in a polynomial fitting model, including conductor width, height, dielectric constant and coordinates of the center point. After initial sampling, the error is checked to see whether the model needs to be improved by more samples or adding more high order terms until the fitting model is satisfying. The result model can be directly used to calculate the parasitic capacitance. However, there are too many parameters to fit, thus sampling is an extremely large and time-consuming project. Also it is hard to pick the right high order term to be added to improve the model accuracy. In other words, this modeling method cannot improve the model accuracy with experience and time.

2.3 Proposed Method

A proper regression modeling method is needed for fast capacitance extraction and the self-improving feature is required to achieve good accuracy for this method. So machine learning comes into view and is applied in the proposed method.

First of all, what is machine learning?

Machine learning is a study of algorithms that improves the performance P at some task T with experience E, which is first proposed by Tom M. Mitchell [6]. Now the machine learning has already been applied in several fields as following:

- Natural language processing
- Speech recognition
- Computer vision
- Robot control

Why machine learning is so important to modeling, or to be specific, parasitic extraction? Through machine learning, the learnt regression model can keep learning new entry data and the model prediction becomes more and more precise.

Machine learning has three basic approaches:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

The support vector machine from supervised learning is the exact applied method because it can both generate regression model without overfitting and do a better classification job which is necessary in the proposed method.

In next several chapters, the proposed method is discussed step by step and the details will also be presented.

CHAPTER III

SIMPLIFICATION

To generate a parasitic capacitance regression model for the input interconnect geometry, enough samples are necessary for no matter what kind of regression method. However, there are so many conductors in the input interconnect geometry, so it is almost an impossible task to create enough samples for regression.

In order to reduce the number of conductors for quicker and easier regression modeling, simplification process without losing too much accuracy is a must.

3.1 Interconnect Definition

In Figure 4, it is an example of five-layer interconnect cross-section.

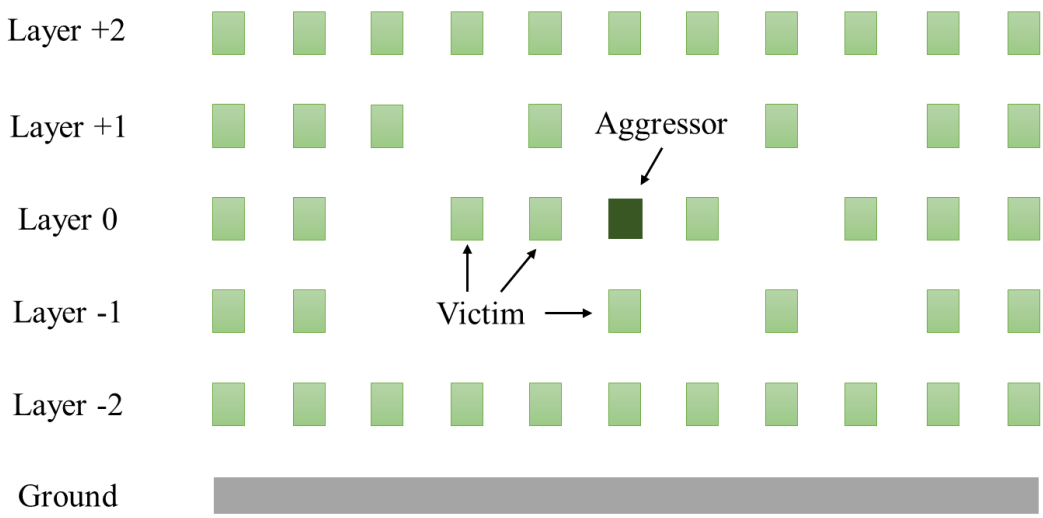

Figure 4. Interconnect cross-section

The conductor in the very middle of layer 0 is the one with positive voltage potential called “Aggressor” and it is the conductor whose total parasitic capacitance is going to be extracted. The neighbors of the aggressor have zero voltage potentials on them are called “Victims”. There is always a ground with interconnect.

For the input interconnect geometry as shown in Figure 5, define the parameter set as P and $P = \{(x_i, y_i, w_i, h_i), i \in 1,2, \dots, n\}$, where number of conductors, n varies.

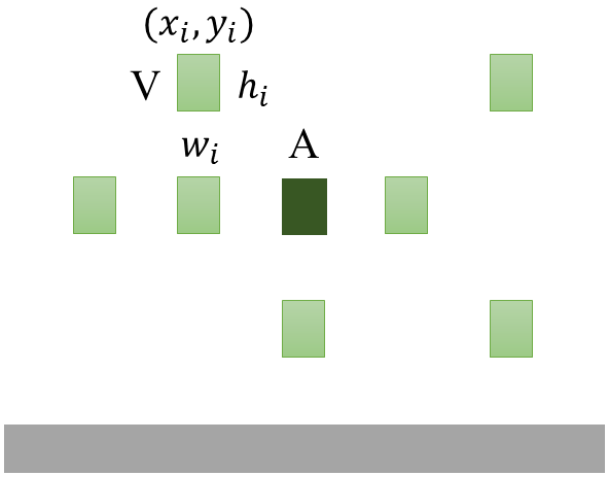

Figure 5. Input geometry example

The coordinates of the center of conductor i is (x_i, y_i) and the coordinates of the center of the aggressor is always (0,0). Width of conductor i is defined as w_i and height of conductor i is defined as h_i.

3.2 Shielding Effect Experiments

What is shielding effect?

For a dense interconnect structure, shielding effect is the fact that one conductor can shield another from the coupling effect so that coupling capacitance of the shielded conductor gets a huge reduction. Experiments are designed to study the shielding effect under different situations to see the error of dropping the shielded conductors [7].

In Figure 6, there are three conductors above the ground: aggressor A, victim V and the shielded conductor D. Let the total capacitance of aggressor A with conductor D be C_t and the total capacitance of aggressor A without conductor D be C_t'. Assume the minimum space between victim V and the shielded conductor D is S_{min}. Then compute the error of dropping conductor D: $e = (C_t - C_t')/C_t$.

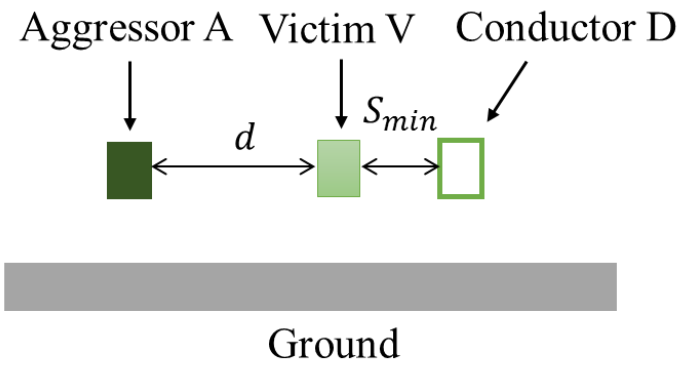

Figure 6. Shielding effect experiment set up

3.2.1 Same-Layer Shielding

To study the shielding effect of the victims on the same layer with the aggressor, an experiment is set up as shown in Figure 6. Victim V and the shielded conductor D are

on the same side of A and the distance between V and A is d as indicated. In this experiment, d is increasing to see the varying error of the total parasitic capacitance after dropping the shielded conductor. Meanwhile, V and D keep minimum space S_{min} to make sure that it is always the worst case for dropping D.

Experiment result is shown in Figure 7 and two curves are plotted based on the metal layer position of aggressor A. From the graph, it can be seen that the largest error occurs when distance d is around two times the minimum space. Thus in this case, dropping the shielded conductor D causes at most $e_1 = 0.6\%$ error.

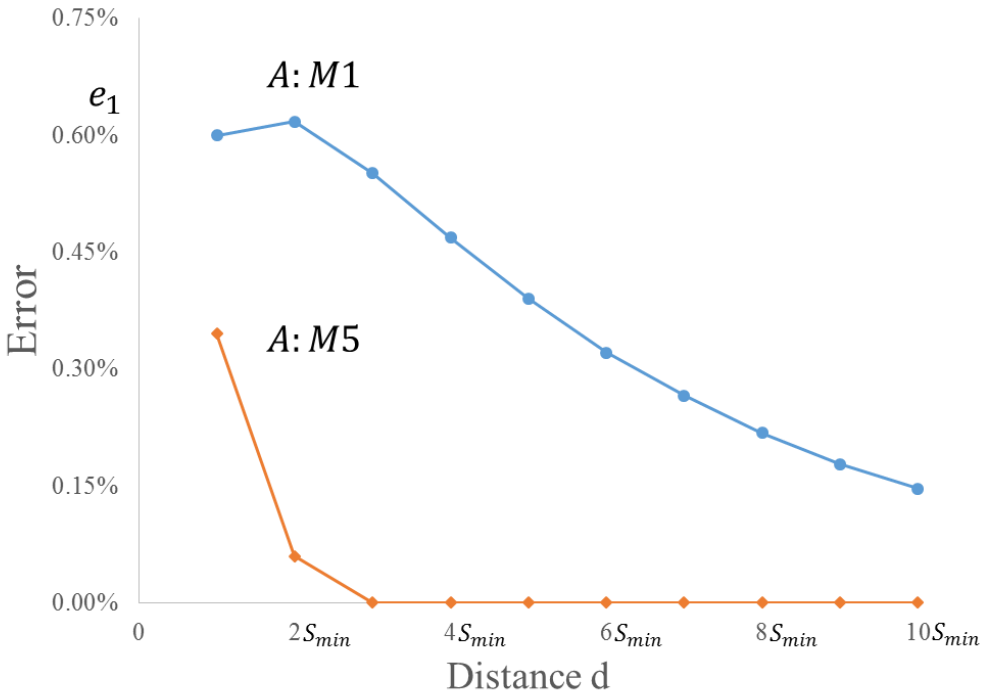

Figure 7. Experiment result for same-layer shielding effect: 1

Another experiment is conducted to study the shielding effect of the same-layer victims on the different layer with the aggressor as shown in Figure 8. Victim V and the shielded conductor D are on the same side but upper layer of A this time. Everything else keeps the same with last experiment.

This case is the worst case compared to V and D both on the lower layer of A because the ground will absorb less electrical fields and the coupling capacitance between the aggressor and shielded conductor is larger. Thus there is no need for experiments on that situation.

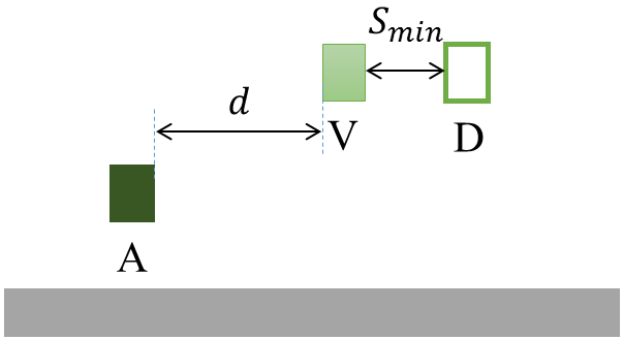

Figure 8. Experiment geometry for same-layer shielding effect

Experiment result is shown in Figure 9 and two curves are plotted based on the metal layer position of aggressor A as in last experiment. From the graph, it can be seen that the largest error exists at the minimum space place. However, the process technology limits the space between two conductors so that the error will not keep

increasing. Thus in this case, dropping the shielded conductor D causes at most $e_2 =$ 1.1% error.

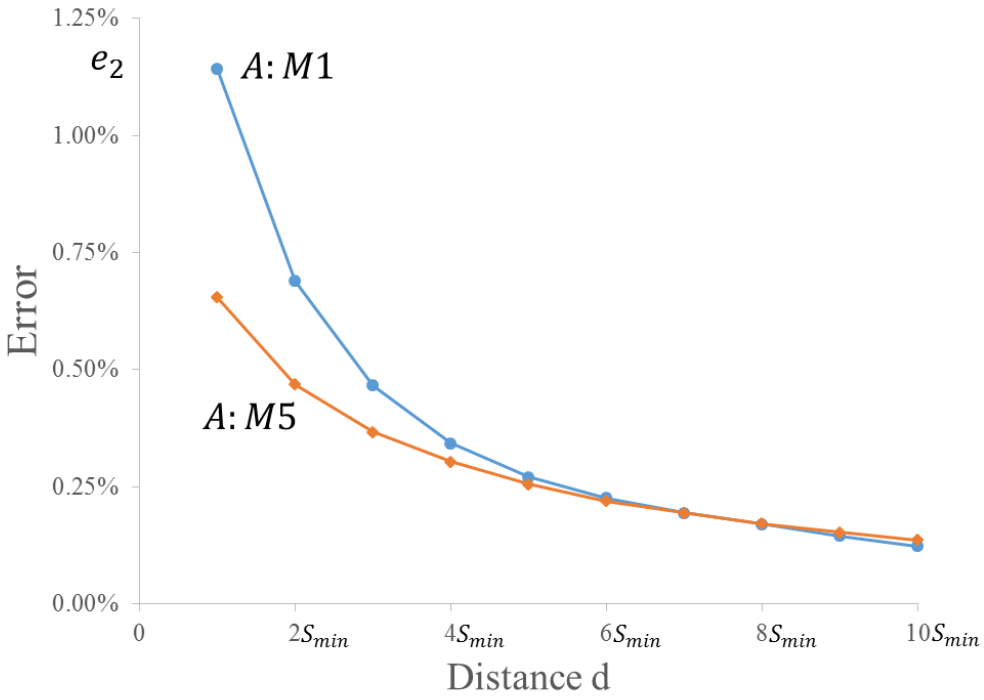

Figure 9. Experiment result for same-layer shielding effect: 2

3.2.2 Different-Layer Shielding

To study the shielding effect of both victims on the different layer with the aggressor separately, an experiment is set up as shown in Figure 10. Victim V and the shielded conductor D are on the upper side of A and the horizontal distance between V and A is d as indicated. In this experiment, d is increasing from zero to see the varying error of the total parasitic capacitance after dropping the shielded conductor. Meanwhile,

the shielded conductor D is always one layer above V to make sure that it is the worst case for dropping D compared with other cases.

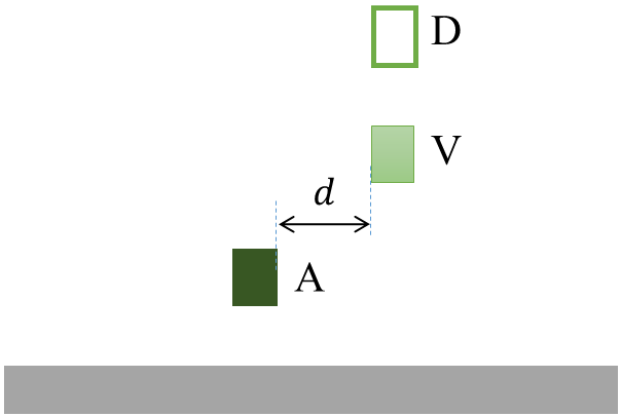

Figure 10. Experiment geometry for different-layer shielding effect

Experiment result is shown in Figure 11 and two curves are plotted based on the metal layer position of aggressor A. From the graph, it can be seen that the largest error exists when distance d is around five times the minimum space. Thus in this case, dropping the shielded conductor D causes at most $e_3 = 1.8\%$ error.

3.2.3 Experiment Summary

For two victim neighbors V and D on one side of the same layer with the aggressor A, dropping the shielded conductor D causes at most 0.6% error.

For two victim neighbors V and D on one side of the different layer with the aggressor A, dropping the shielded conductor D causes at most 1.1% error.

For two victim neighbors V and D that are on the different layer with the aggressor A separately, dropping the shielded conductor D causes at most 1.8% error.

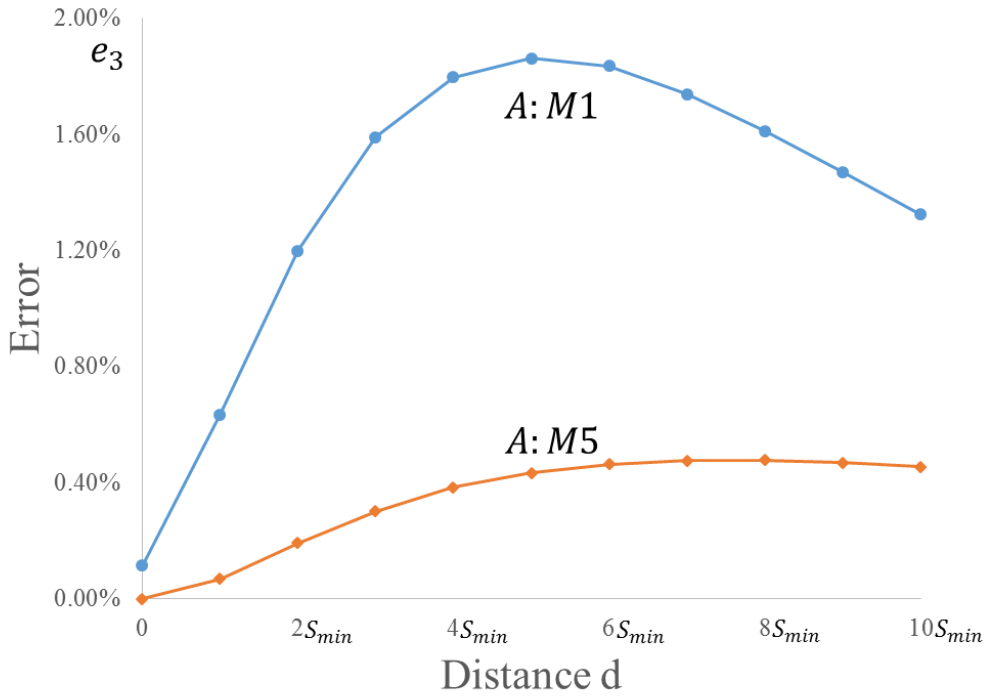

Figure 11. Experiment result for different-layer shielding effect

3.3 Simplification Algorithm

Based on the three experiments on the shielding effect, the simplification algorithm describes how to drop the shielded conductors to reduce the number of conductors in the input geometry as seen in Figure 12. After running the simplification algorithm, all of the shielded conductors have been removed and only the simplified geometry is left.

Algorithm 1 Simplification algorithm

for any given aggressor A at layer i **do**
  for each side of the same layer i **do**
    Find the nearest neighbors V_L^i on left and V_R^i on right.
    Omit other neighbors shielded by V_L^i and V_R^i.
  end for
  for each side of upper or lower layer j, $j = i-1, i+1, i-2, i+2$ **do**
    Find the nearest neighbors V_L^j on left and V_R^j on right.
    Omit other neighbors shielded by V_L^j and V_R^j.
    If a neighbor V is above or below A and $|x_v| \leq s_{min}/2$. keep V.
  end for
end for

Figure 12. Simplification algorithm

The error bound needs to be calculated to see whether it is useful or not.

3.3.1 Simplification Algorithm Error Bound

Two experiments are designed to find the maximal error of the total capacitance. The first one is a symmetrical structure with four shielded conductors on the upper and lower layers of the aggressor as shown in Figure 13. The victims and the shielded conductors are on the same layer and every distance d is varying at same time as last experiment to find the maximal error.

Experiment result is shown in Figure 14. It is seen that the maximal error occurs when the distance is equal to minimum space. The error bound equals 3.6% and is smaller than four times of the error e_2.

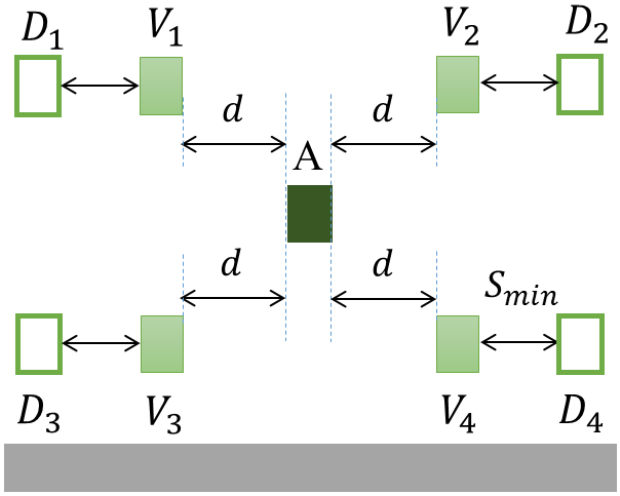

Figure 13. Experiment geometry for error bound worst case 1

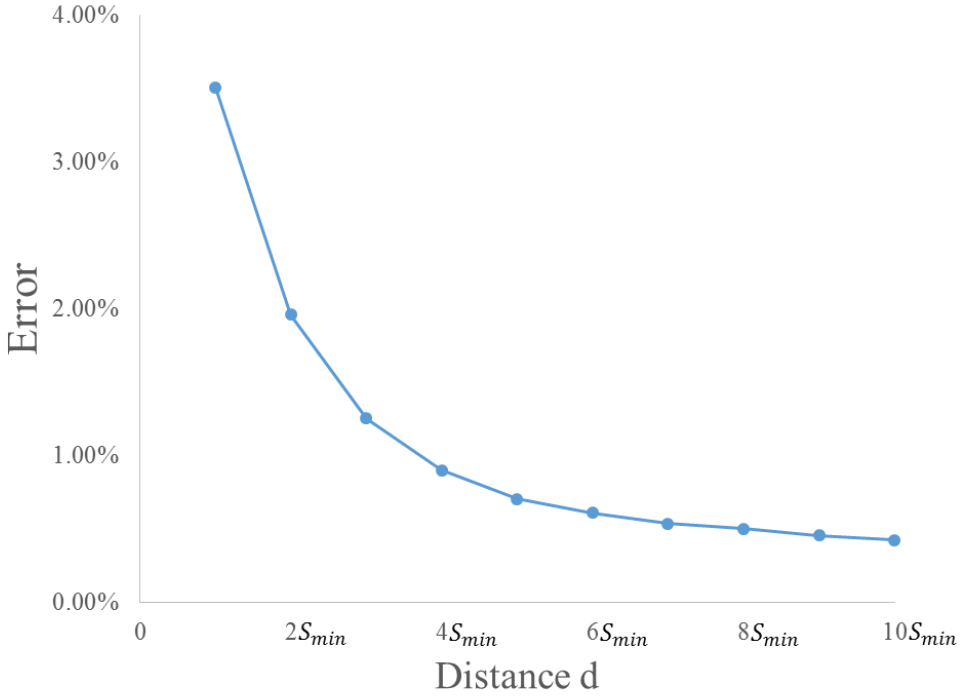

Figure 14. Experiment result for error bound worst case 1

The second one is a symmetrical structure with four shielded conductors on the upper and lower layers of the victims as shown in Figure 15. The victims and the shielded conductors are on the different layers and every distance d is varying simultaneously to find the maximal error. Meanwhile, the shielded conductor D is always one layer above or below V.

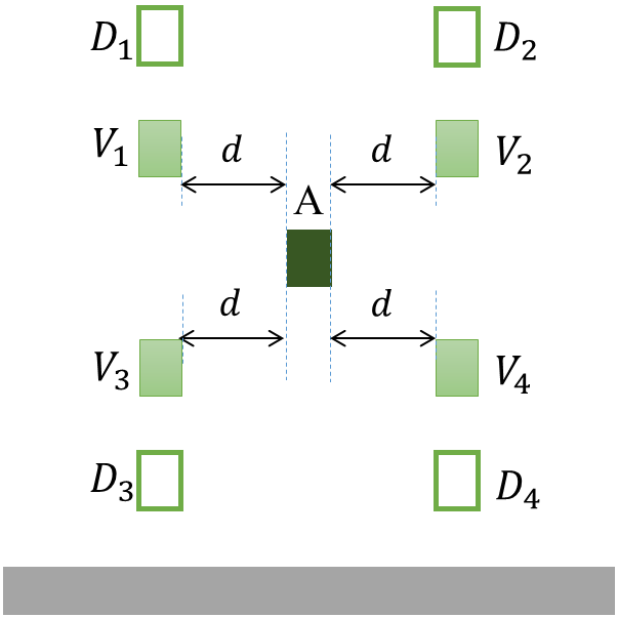

Figure 15. Experiment geometry for error bound worst case 2

Experiment result is shown in Figure 16. It is seen that the maximal error occurs when the distance is twelve times the minimum space. The error bound also equals 3.6% and is smaller than four times of the error e_3.

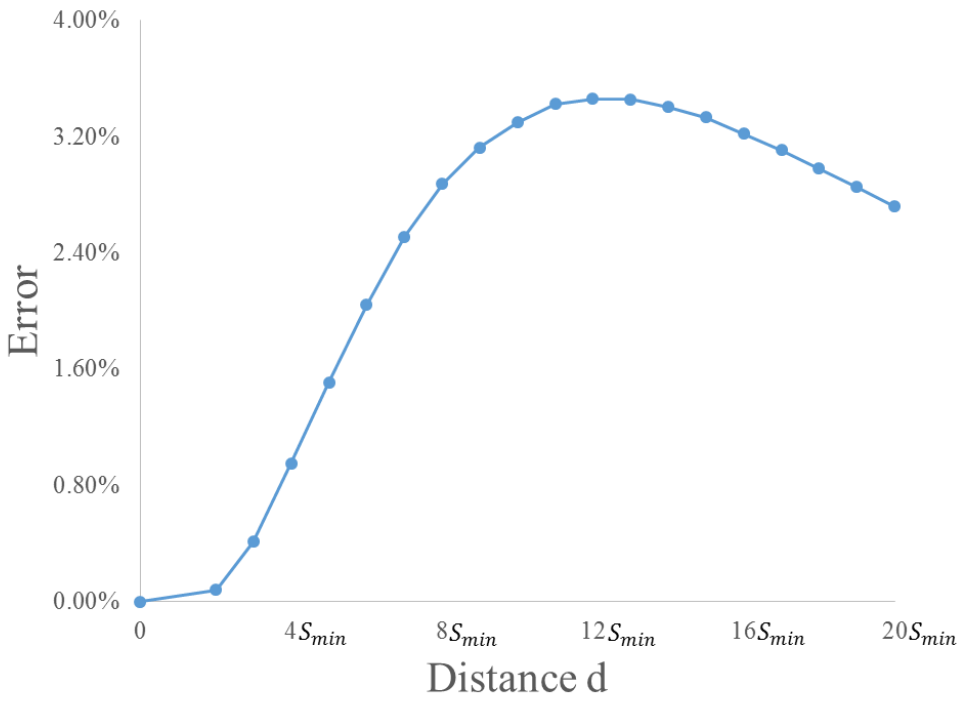

Figure 16. Experiment result for error bound worst case 2

Thus the maximal error bound of the simplification algorithm is $4e$, where error e is the maximal error in the shielding effect foundation experiments. For the first error bound case, the error is decreasing with the distance and for the second case, the error is increasing with the distance until it reaches the maximum. If combining these two cases, the maximal error will be even lower for the real simplification algorithm application.

To sum up, the simplification algorithm is valid to use and the algorithm illustration is shown in Figure 17. The shielded conductors D_1, D_2, D_3, D_4 have all been removed and the simplified geometry is generated.

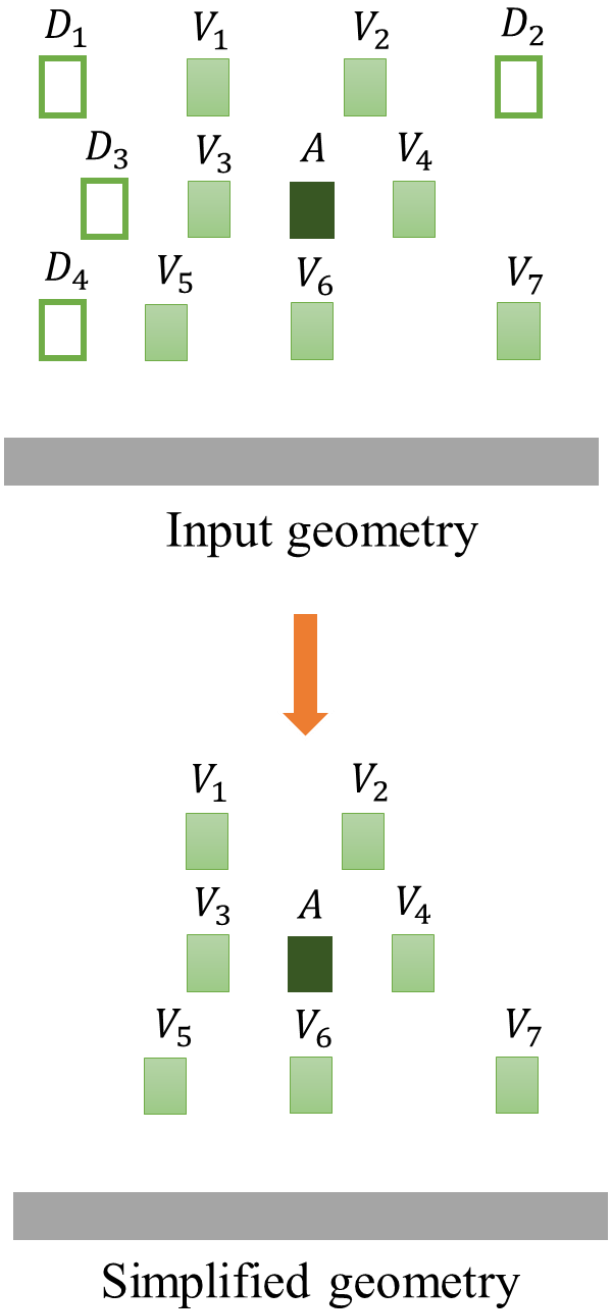

Figure 17. Simplification algorithm illustration

3.3.2 Simplified Geometry

After applying the simplification algorithm, the input interconnect geometry is reduced to the simplified geometry. As shown in Figure 18, simplified geometry is a nine-square form geometry and the aggressor A is always in the center square. Each neighbor square only has two states: 1 for a victim conductor inside and 0 for no victim inside. Depending on the number of present neighbors, there can be at most 8 neighbors and $2^8 = 256$ cases for three-layer input geometries.

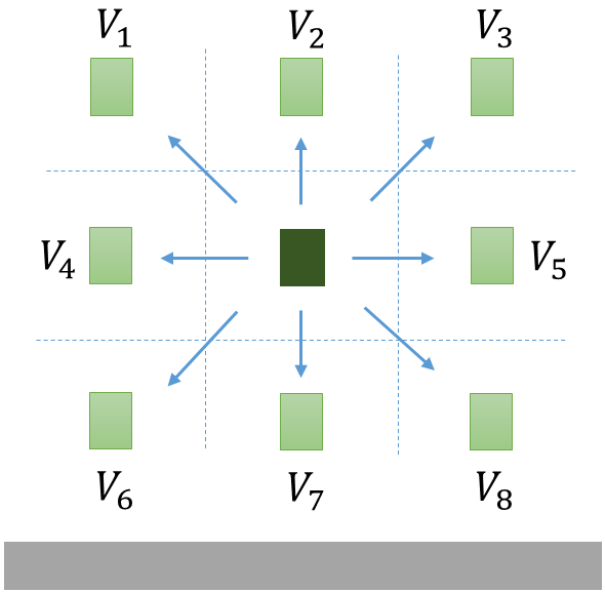

Figure 18. Simplified geometry

Now, it is ready to create samples and build a SVM regression model for the simplified geometry for total parasitic capacitance prediction.

CHAPTER IV

SVM REGRESSION

Every good regression model is based on enough training data, this is not an exception. After generating a simplified geometry for the input interconnect, it is necessary to find out the key parameters related with the total parasitic capacitance and create training samples by varying the corresponding parameters.

4.1 Create Samples

For the input geometry, there is a corresponding parameter set $P = \{(x_i, y_i, w_i, h_i), i \in 1,2, \dots, n\}$ which describes all the parameters of input interconnect. As is known to all, the interconnect process technology fixes the height of conductors on each different metal layers. Thus h_i and y_i are not variables for the chip designers to change and experiments need to be done to show that the left parameters do affect the total parasitic capacitance of the input interconnect.

4.1.1 Variables Experiments

The first experiment is designed to show that varying coordinate x_i affects the total capacitance. There is only aggressor A and victim V with the ground in the experiment geometry. With the coordinate x_i increases, total capacitance decreases as shown in Figure 19.

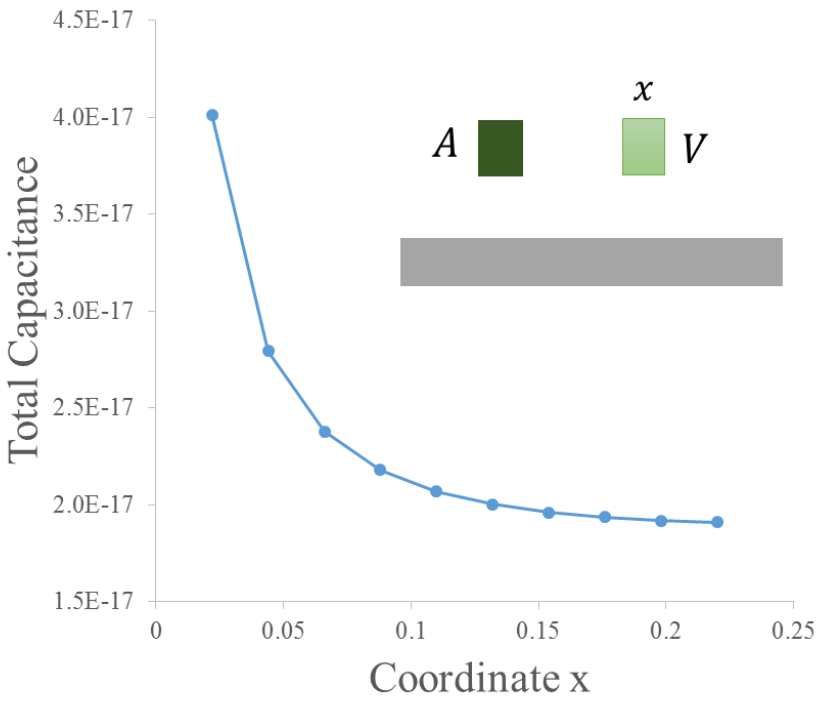

Figure 19. Experiment result for variable x

The second experiment has the same set up, however, the victim is on the upper layer of the aggressor and its width w_i varies this time. As seen in Figure 20, the total capacitance increases as w_i increases.

The total capacitance may not change with the two variables exactly as the curves in Figure 19, 20 show, but the two variables affecting total capacitance has been proven. Now samples for the initial models are going to be created based on these two parameters.

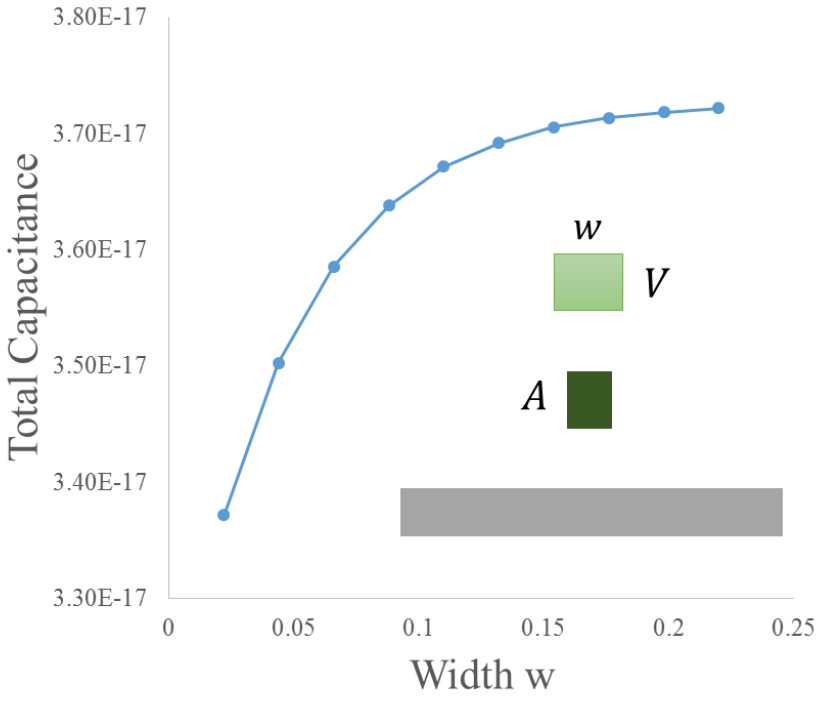

Figure 20. Experiment result for variable w

4.1.2 Rule for Creating Samples

For each simplified geometry, unbiased samples are necessary to create initial models for preventing regression overfitting. The left and right side of aggressor A are easily divided by positive or negative sign of the coordinate x_i. Then for each victim on one side, uniformly create 5 sample points for its coordinate x with step $2 \times S_{min}$ and for every conductor, create 2 sample points for its width w with step S_{min} increasing and decreasing. After each sample geometry has been set up, run field solver for it to get parasitic capacitance reference value r_r as target value for regression model.

4.2 SVM Regression

Regression is to find the relationship between the variables and the target value by building a model to describe and predict. For each simplified geometry, define variable vector f used for regression as $f = (x_1, w_1, x_2, w_2, \dots, w_7, w_7, x_8, w_8)$. After creating enough samples with reference value r based on vector f, a high-dimensional accurate regression model $M(f)$ is learnt by SVM. Taking f as input, SVM regression model is used to predict the total parasitic capacitance of aggressor A. This is the general idea about how to build and use the regression model. But now, why the SVM regression is applied here? What is the advantage compared with other regression technique?

4.2.1 Why SVM Regression?

Talking about the regression technique, from the most simple and easy-understanding, linear regression to the recent popular technique, SVM regression, it is a mathematical revolution process. Before the machine learning idea came into view, many regression methods have been tried to achieve a general model covering all sample data with good accuracy. Because coupling capacitance between victims and the aggressor follows the Gaussian distribution with respect to the coordinate x, it is no wonder that only the nonlinear regression method made it by Levenberg-Marquardt method [8]. However, no matter linear or nonlinear regression, they are old techniques to solve the numerical problem in its own space. SVM regression is different. In essence, it is still the linear regression after mapping all sample data into higher-dimension space with the penalty parameter preventing overfitting issue. This will be discussed later and now Table 1 lists major pros and cons of different regression methods.

Table 1. Comparison between different regression methods with SVM

Regression methods	Pros compared to SVM	Cons compared to SVM
Linear regression		Not accurate at all.
Linear piece-wise regression method		Needs lots of samples to guarantee accuracy.
Least square exponential regression method	Can predict data out of the range.	Generate too many models for single case.
Levenberg-Marquardt regression method	Can predict data out of the range.	Cannot improve the regression model.

From table 1, it is seen that SVM regression model cannot predict accurately for data out of the input sample range. This is the limitation of SVM regression model and it can be solved by building more regression models. The reason why building a new model is to prevent the overfitting issue and details are saved for later discussion. As mentioned before, the self-improving feature is the best reason for choosing SVM.

Next an example is presented to illustrate what SVM regression really does.

4.2.2 SVM Regression Example

In daily life, SVM regression can be used to predict the blood pressure for patients. Collecting all the related parameters including blood glucose, daily exercise time, weight, age and the target blood pressure value from patients 1 to 7, SVM can learn a regression model based on these already-known data as training data listed in Table 2. Then the regression model can be used to predict the blood pressure for the new patient, the conductor. His blood pressure is predicted as 68 mm hg.

Table 2. SVM regression example training data

Patient #	Blood glucose (mg/dL)	Daily exercise (hours)	Weight (lb)	Age (years)	Blood pressure (mm hg)
1	85	2.5	152	25	66
2	157	0.2	188	31	64
3	89	3	150	28	63
4	138	1	148	36	52
5	75	5	140	21	58
6	181	0.5	203	45	71
7	120	4	161	52	74
8	98	1.8	137	35	?

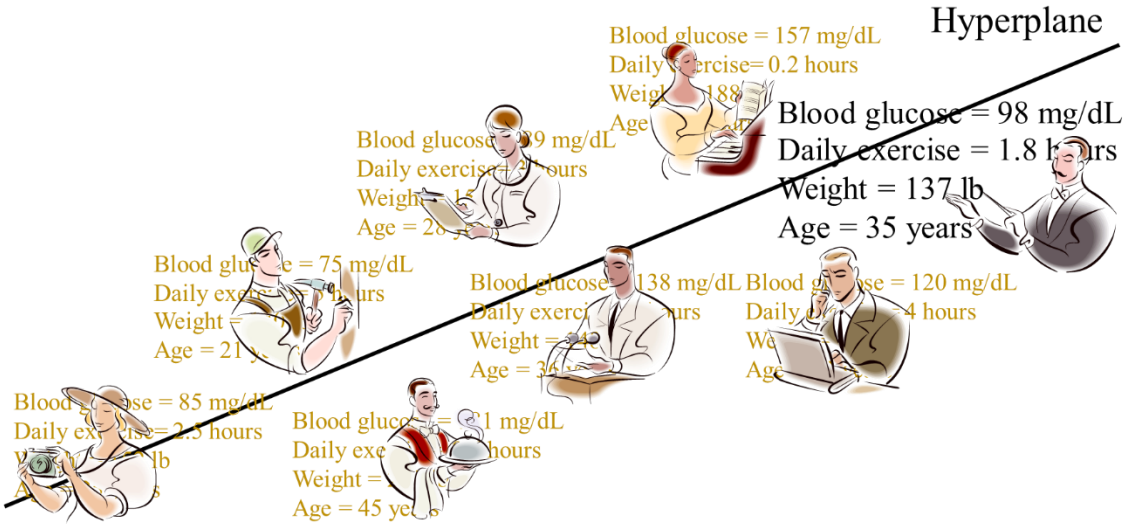

Figure 21. Blood pressure prediction

A hyperplane is generated by SVM as the regression model as seen in Figure 21.

Of course, this is only an intuitive example for people to have a straight-forward understanding on practical application of SVM regression. But for better understanding, it is necessary to know how SVM regression works.

4.2.3 SVM Regression Work Principles [9]

A linear regression model, or the hyperplane with respect to given m training samples $(\vec{x}_m, y_m) \in X \times \mathbb{R}$ is defined as a function $f(x) = (\vec{w} \cdot \vec{x}) + b$ by normal vector $\vec{w}$ and offset b. As mentioned before, SVM regression is linear regression in essence. In Figure 22, the solid line represents the function or the regression model and the dash lines indicates the tolerance range of predefined error ε. Error analysis is important to any regression method and here the Vapnik ϵ-SVR is applied [10].

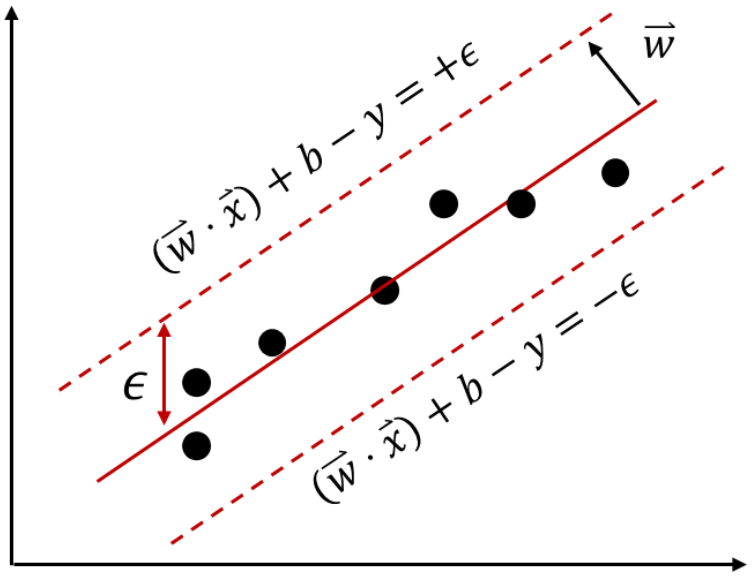

Figure 22. SVM regression work principle illustration

Define Vapnik's ϵ-insensitive loss function $|y - f(x)|_\epsilon = max\{0, |y - f(x)| - \epsilon\}$ where $\epsilon > 0$ is a predefined constant which controls the noice tolerance. This loss function ignores the error within dash lines and penalize the other data with large error ξ as indicated in Figure 23 which prevents the overfitting issue.

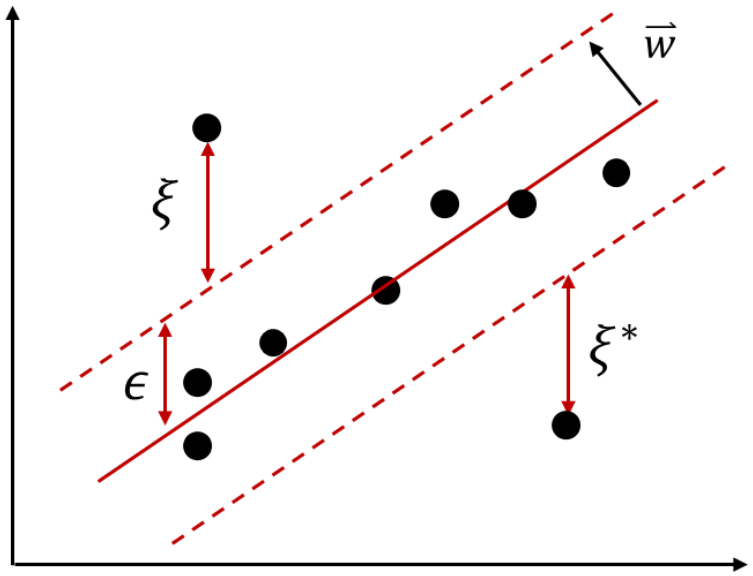

Figure 23. SVM regression error analysis illustration

After introducing the Vapnik loss function, the goal is to find a function $f(x)$ that has at most ϵ deviation from targets y_i for all m examples and as flat as possible. Here word flat means to keep the regression model as simple as possible to prevent it from being too complex.

Summarizing these two points, there is an optimal problem with the goal to minimize the model regularization and sum of penalized errors as following:

$$\text{Minimize } \frac{1}{2}\left|\left|w\right|\right|^2 + C \cdot \sum_{i=1}^{m}(\xi_i + \xi_i^*)$$

$$\text{Subject to } \begin{cases} f(x_i) - y_i \leq \epsilon + \xi_i \\ y_i - f(x_i) \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases}, \text{ where penalty parameter } C > 0 \text{ and } i = 1, \ldots, m.$$

The key idea to solve this problem is to construct a Lagrange function from both the objective function and the corresponding constraints by introducing a dual set of variables. It is purely mathematic problem thus the detail process is omitted. After solving this optimal problem, the SVM regression model is found.

As shown in Figure 24, linear regression is not a good choice to deal with the nonlinear problem. It is certainly a bad idea to use one piece of line to describe a circle. So here comes the question, how does SVM regression deal with nonlinear problem?

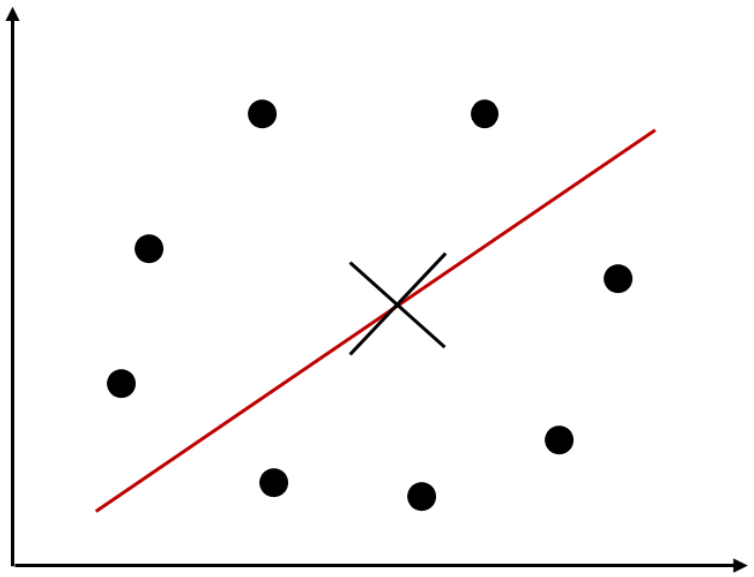

Figure 24. Linear regression fails in nonlinear problem in 2D space

Of course, using a circle to describe another circle is the most straight-forward method. However, SVM has a better solution. By mapping the original training data from 2D to 3D, it is easy to find a plane to cover all these points as shown in Figure 25.

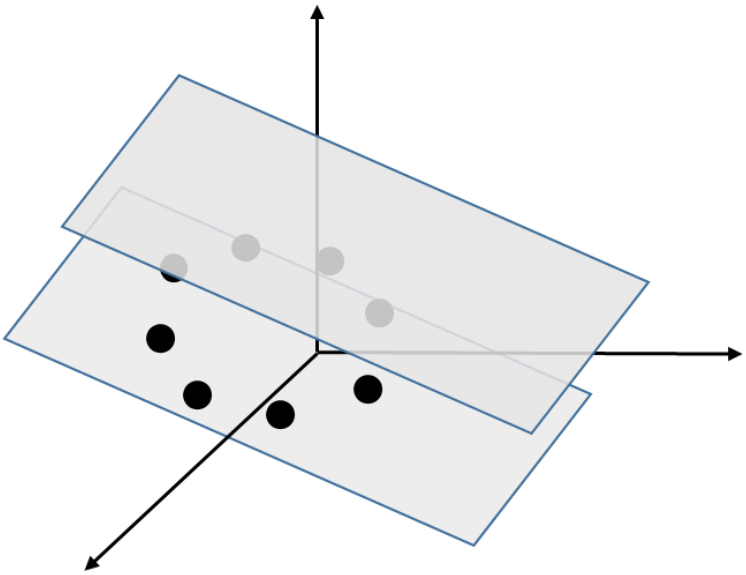

Figure 25. Linear regression solves nonlinear problem in 3D space

In other words, SVM transfers the low-dimensional nonlinear regression problem to high-dimensional linear regression problem by mapping the data $x \to \Phi(x)$.

It is a genius idea, but after mapping to the high-dimensional space, the number of dimensions is growing explosively. Thus the mathematical computation to solve the dot product becomes tremendously complex and impossible to be done.

Then the kernel method is introduced to solve this problem by conducting the mathematical computation in low-dimension space but gives out the result for dot product in high-dimensional space.

Standard SVM kernels which are usually applied in SVM are listed as following:

- Linear kernel: $K(x_i, x_j) = x_i^T x_j$
- Polynomial kernel: $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d$
- Gaussian kernel: $K(x_i, x_j) = \exp(-\gamma \parallel x_i - x_j \parallel^2)$
- Sigmoid kernel: $K(x_i, x_j) = \tanh(\gamma x_i^T x_j + r)$

4.2.4 SVM Training for Regression

After building a SVM regression model for a simplified geometry, it can be used to predict the parasitic capacitance of the input geometry that has the same simplified geometry as illustrated in the example.

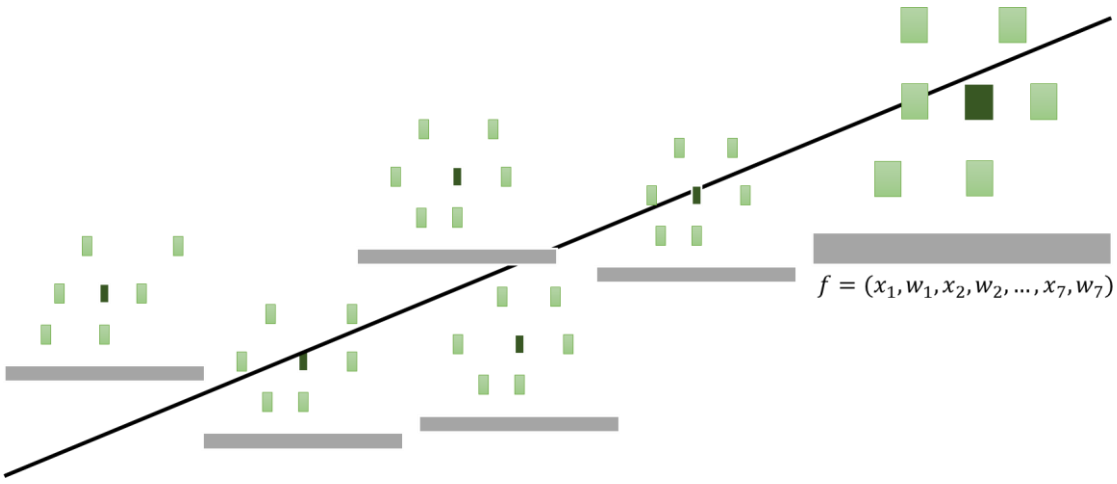

Figure 26. SVM regression applied in proposed method

As shown in Figure 26, the new simplified geometry is like the male conductor and the input variable vector f is like the health data of patients. Through this regression model, the target parasitic capacitance value is predicted just like the blood pressure.

How to train the SVM for regression model in the proposed method?

At first, enough samples have been created for the corresponding simplified geometry by varying the variables from vector f and field solver has been run to generate the reference value r_r. Then the training data t_r for SVM is created with reference r_r and vector f. For a simplified geometry, it is written in the format as below.

$$r_r \qquad 1{:}\,x_1\ 2{:}\,w_1\ 3{:}\,x_2\ 4{:}\,w_2\ \ldots\ 13{:}\,x_7\ 14{:}\,w_7\ 15{:}\,x_8\ 16{:}\,w_8$$

In the line, r_r is called label, the number is called index and x or w is called value.

Secondly, scale the range of training data. Scale target value r_r to $[0, +1]$ and scale the values to $[-1, +1]$, which is to prevent the small values from being dominated by the large ones.

Then choose Gaussian kernel for the nonlinear regression. Why it is a nonlinear regression problem? As mentioned before, the coupling capacitance between victims and the aggressor is following the Gaussian distribution, thus it is a nonlinear regression problem. Also Gaussian kernel is the universal kernel for SVM and its corresponding Hilbert space has infinite dimensions, so it is the best kernel here. Through cross validation method, the best Gaussian kernel parameter vector $p_r = (C, \gamma, \varepsilon)$ is found when the validation error is minimal.

Finally, train regression SVM to get model $M(f)$.

Is this model ready to predict? Will it have overfitting issue?

4.2.5 Overfitting

As mentioned for a lot of times, apparently overfitting is a common and important issue when dealing with the regression problem. From different perspectives, efforts have been done to prevent the overfitting happening. Generally, overfitting occurs when the model complexity to its training data size ratio is high, which means it is using a too complex model to describe a simple data or the data itself is biased so that the information is missing.

Thus, to prove that there is no overfitting issue in the regression model that used in the proposed method, both two perspectives have been considered and tested.

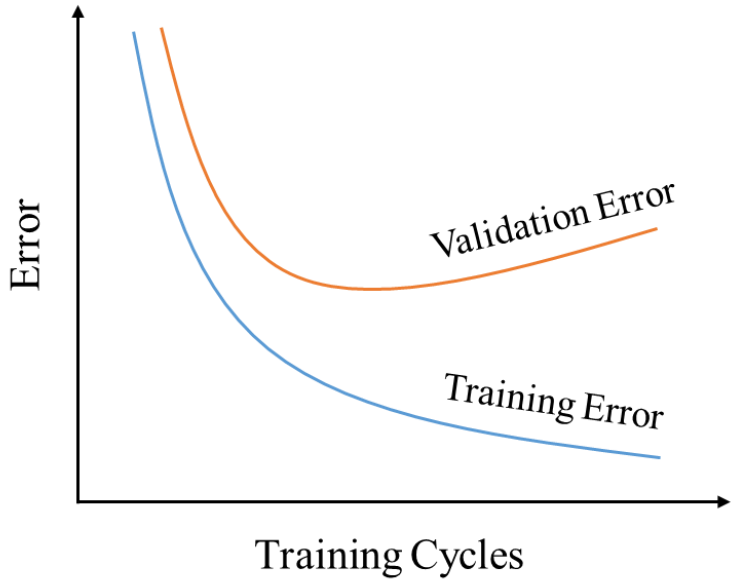

Figure 27. Graph comparing two errors with training cycles increasing **[11]**

For model complexity, five-times cross validation is applied in selecting parameters for the Gaussian kernel to achieve the minimal validation error. In general, the training error decreases with the training cycles and it is always small as in Figure 27. Only when the validation error approximately equals the training error, there is no overfitting, because the regression model describes the training data well. That is why the minimal validation error is required.

For training data size, unbiased samples of smaller size are tested to prove no overfitting.

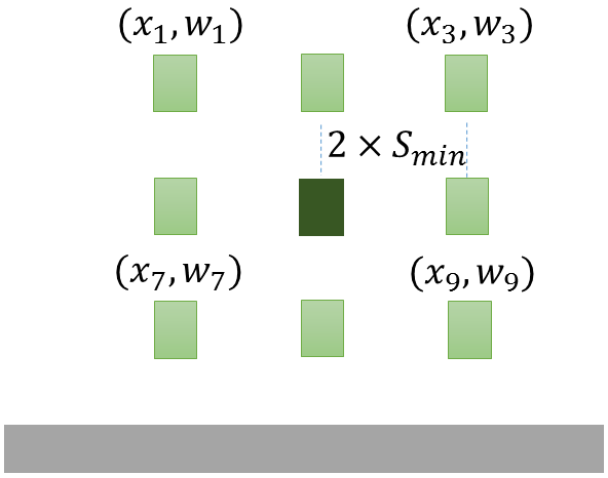

Figure 28. Simplified geometry to test overfitting

Following is an experiment conducted to study the overfitting issue. For a three-layer input geometry as shown in Figure 28, create a SVM regression model for it. When creating unbiased samples, uniformly pick 3 sample points for the coordinate x of victims on the left and right side with step $2 \times S_{min}$ and run field solver for each sample.

Find the best parameters p_r for Gaussian kernel by five-times cross validation and train the regression SVM. Because there are 6 neighbors for both sides, the training data size is $3^6 = 729$. After the five-times cross validation and training process, the minimal validation error is $8.23436e^{-6}$ and training error is $7.55652e^{-6}$. Thus conclusion 1 is drawn that because training error is approximately equal to validation error, there is no overfitting for this regression model. This conclusion may seem not very persuading, thus test needs to be done to show the accuracy of the regression model.

Finally, for each victim on left and right side, uniformly create 3 test points for its coordinate x with step S_{min} and test them with the regression model. Also the field solver needs to run for the test data to get the reference values. Based on the prediction values and reference values, calculate errors to check whether there is overfitting or not. Set the error limit as 3%.

The size of test data is same as the training data and from Figure 29, it is easily found that all of the test errors fall into the range of $[-3\%, +3\%]$. Thus conclusion 2 is drawn that because the absolute value of every test error is smaller than 3%, it is a good regression model and there is no overfitting issue. Also it indicates that there will be no overfitting issue for the regression model generated by more unbiased training data.

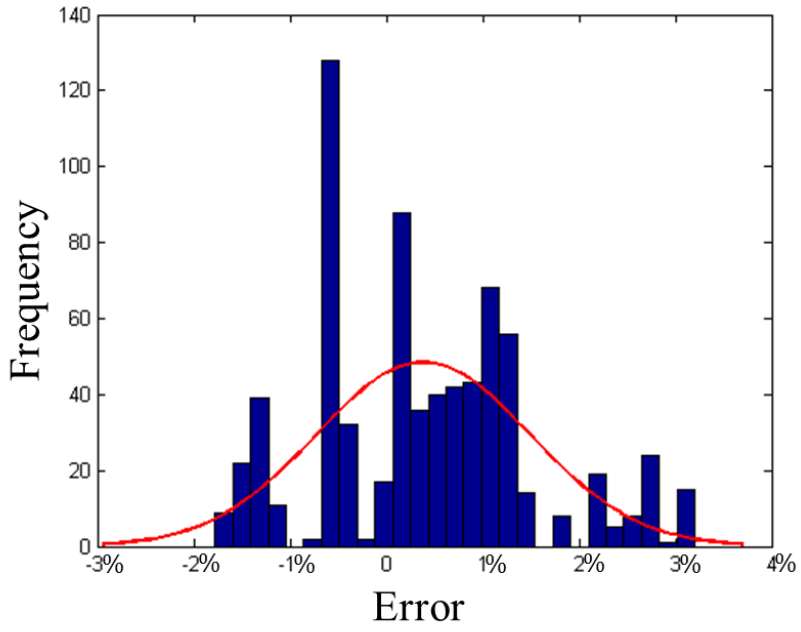

Figure 29. Test results for overfitting experiment

4.2.6 SVM Regression Applied

Having proved the SVM regression model is valid to use, it is ready for applying in parasitic capacitance prediction.

After the simplification process, the simplified geometry has the variable vector $f_{new} = (x_1', w_1', x_2', w_2', \dots, x_7', w_7', x_8', w_8')$ as the input vector. Based on the vector and any guessed value r_{guess}, create test data t_p as below.

$$r_{guess} \quad 1: x_1' \; 2: w_1' \; 3: x_2' \; 4: w_2' \; \dots \; 13: x_7' \; 14: w_7' \; 15: x_8' \; 16: w_8'$$

Scale the range of test data to $[-1, +1]$ as it did in training process. Then use regression model to predict the parasitic capacitance value $M_{l_1}(f_{new}) = r_{scaled\ predict}$. Because it is the scaled value, a backward-scaling is necessary to restore the value to normal range $r_{predict}$ and it is the predicted parasitic capacitance of new input geometry.

At last, check the accuracy by running field solver for this input geometry to get reference value r_r and calculate error with $(r_r - r_{predict})/r_r$. If the error is larger than the error limit ε which is usually preset as 3%, the prediction result is considered inaccurate. Otherwise, it is accurate. Then the training data is updated by adding r_r and vectorf_{new}. Also store $r_{predict}$ into prediction database DB_{p1} for model M_{l_1}.

Before the size of prediction database DB_{p1} reaches threshold T, the accuracy of model M_{l_1} needs to be checked every time when there is a new input geometry that is identified to use the same model. Until the size reaches threshold T, check the model accuracy periodically with the size increasing by T. If error is smaller than ε, then it must be also smaller than the satisfying limit $\epsilon = \varepsilon$. Round it to 1-significant figure to update ϵ and update T to $T_{new} = 10 \times T$ until the error is reduced to the satisfying level ϵ_s which can be preset as 0.1%. If the error is larger than ϵ but smaller than ε, the regression model is considered unsatisfying. Otherwise, it is satisfying and if the prediction error keeps being under the satisfying level ϵ_s for another T times parasitic prediction, then there is no need for checking accuracy any more. When checking accuracy, always update the training data set with the reference value and input variable vector.

If the prediction parasitic capacitance for the new input geometry is not accurate, a new regression model will be built for it. Why?

This question has been answered in previous chapters which discuss how to create samples for initial regression models and why SVM regression is better. The input vector f with large prediction error is usually out of the regression model range because SVM is not able to handle this kind of input. If creating new samples around the original values and merge them into the old training data, it will no longer be unbiased and the refined regression model trained by it will have overfitting issue. Also it is hard to create unbiased training data for unknown input range. It may cost a lot of time to create enough samples for that but only a little to build a new model. All in all, these are the reasons why to build a new model but not to refine the old one.

For each victim on left and right side, uniformly create 2 sample points for its coordinate x on each side with step S_{min}. Then follow the method above to select the best kernel parameter and train SVM to get the new model.

Next chapter, it is going to talk about how to choose the right corresponding regression model from the model library for the input geometry to predict its parasitic capacitance.

CHAPTER V

SVM CLASSIFICATION

After building regression models for each simplified geometry, a right corresponding regression model needs to be chose for the input interconnect geometry to predict accurate parasitic extraction. Certain number of initial regression models have already been generated and stored in a model library that is classed by number of conductors in the simplified geometry. For instance, there are 256 initial models for the three-layer input interconnect. However, according to the nine-square form and the status of victims, the corresponding model can be easily found by the binary codes based on the victim status, like 10011101 for model 157. Same method can be applied on the more layer input geometry by adding more squares. So why SVM classification is necessary and applied here?

5.1 Why SVM Classification?

Because not only there are many regression models stored in the same model library for different simplified geometries, but also there may be multiple models for the same simplified geometry. That forms a boundary problem which cannot be solved by the binary codes based on victim status, which is exactly the reason that SVM classification is important and useful here.

For the simplified geometry in Figure 30, there is a coordinate vector $g = (x_1,\ y_1, x_2, y_2, \dots, x_7, y_7, x_8,\ y_8)$ used for classification.

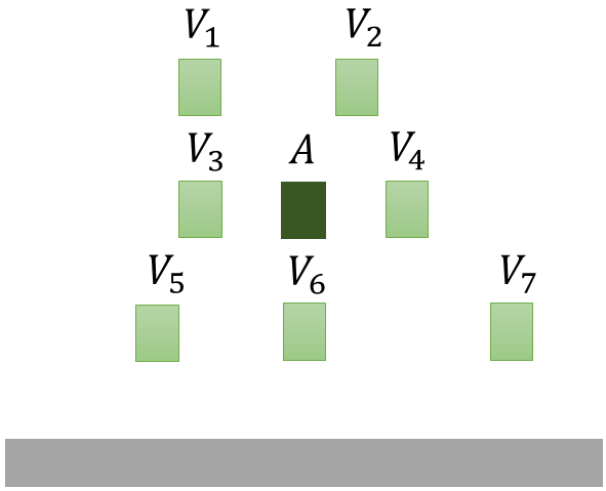

Figure 30. Simplified geometry for classification illustration

There are basically three classification jobs. The first one is to find the corresponding model library according to the number of conductors in the simplified input geometry. The second is to find whether there exists any model built for the geometry. If not, build a new initial model. Otherwise, find the corresponding regression model that has already been built for it.

5.2 SVM Classification Example [12]

In daily life, SVM classification can be used to tell whether the patients lack exercises or not. Collecting all the related parameters including blood glucose, blood pressure, weight, age and the target labels, yes or no from patients 1 to 7, SVM can learn a classifier based on these already-known data as training data listed in Table 3. Then the classifier can be used to predict the status for the new patient, the conductor.

Table 3. SVM classification example training data

Patient #	Blood glucose (mg/dL)	Blood pressure (mm hg)	Weight (lb)	Age (years)	Lack exercises
1	85	66	152	25	No
2	157	64	188	31	Yes
3	89	63	150	28	No
4	138	52	148	36	Yes
5	75	58	140	21	No
6	181	71	203	45	Yes
7	120	74	161	52	Yes
8	98	68	137	35	?

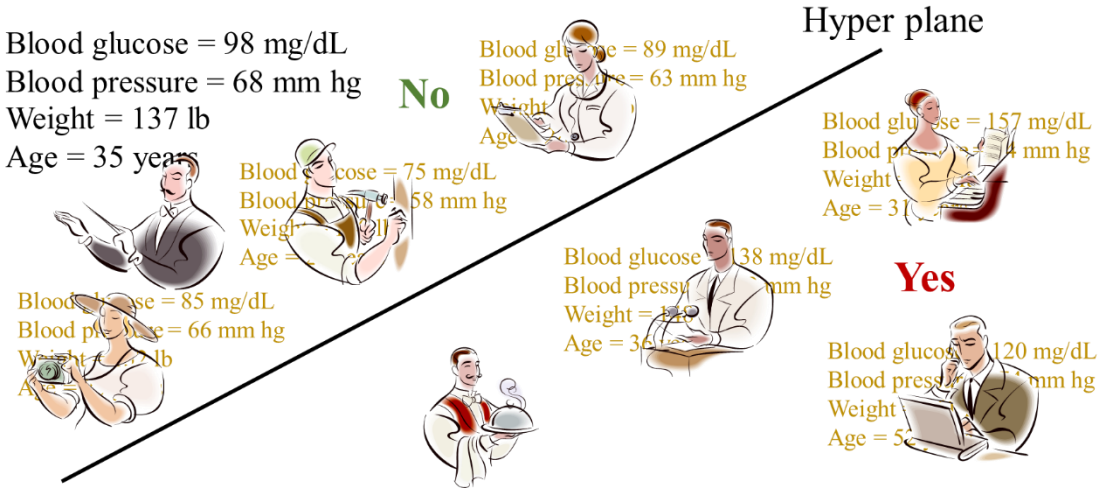

Figure 31. Health condition classification

His health condition is considered lacking exercise and a hyperplane is generated by SVM as the linear classifier as seen in Figure 31.

5.3 SVM Classification Work Principles [13]

A canonical hyperplane with respect to given m training samples $(\vec{x}_m, y_m) \in X \times \{\pm 1\}$ is defined as a function $f(x) = (\vec{w} \cdot \vec{x}) + b$ by normal vector $\vec{w}$ and offset b. The zero hyperplane h shown in Figure 32 is $(\vec{w} \cdot \vec{x}) + b = 0$, which linearly separates the points with two different labels.

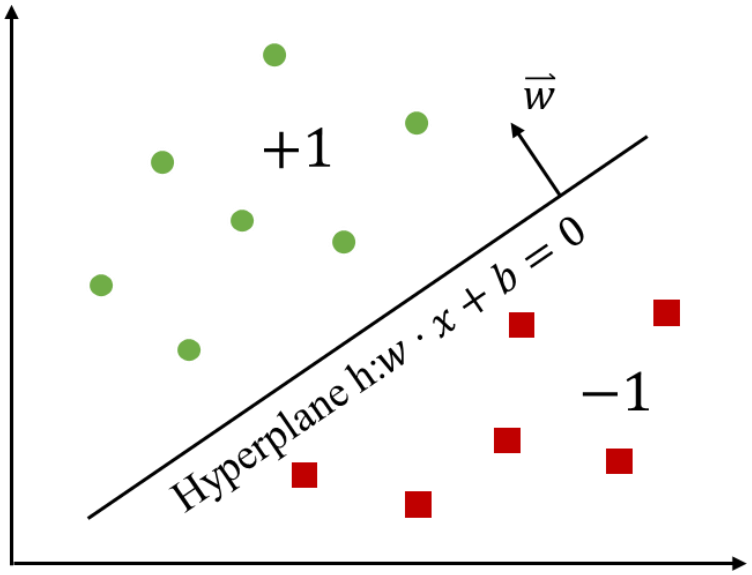

Figure 32. SVM classification work principle illustration

Then there is a classifier $h(x) = \begin{cases} +1 & if \ (\vec{w} \cdot \vec{x}) + b > 0 \\ -1 & otherwise \end{cases}$ generated by SVM to discriminate the different classes.

However, in SVM learning process, it always finds the hyperplane with maximal margin shows in Figure 33.

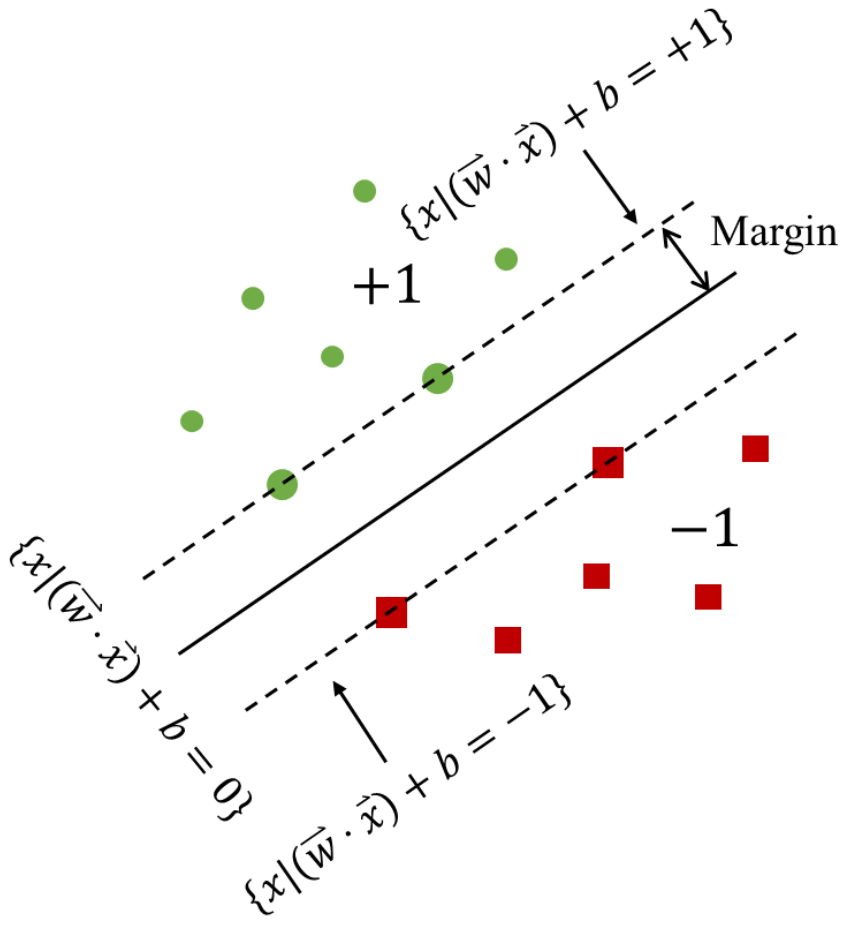

Figure 33. Support vectors and maximal margin for hyperplane

After normalizing $yf(x)$ as 1 and doing some mathematical calculation, the margin is represented as $\frac{1}{\|w\|}$. So there is another maximal problem to solve.

$$\text{Maximize } \frac{1}{\|w\|}$$

$$\text{Subject to } |(\vec{w}\cdot\vec{x})+b| \geq 1, \text{ where } i = 1,2,\dots,m$$

Like how the regression optimal problem being solved, Lagrange multipliers are introduced to transfer it into a dual problem. Then it is solved by quadratic programming method. Mathematical details are omitted here.

After finding the maximal margin, the support vectors $\mathrm{SV} = \{\vec{x}_j \mid |(\vec{w} \cdot \vec{x_i}) + b| = 1, j = 1 \ldots\}$ lie on the margin and they are the only useful points which determine the linear hyperplane with the hard margin. This is how the name of support vector machine comes. If introducing slack variables ξ and penalty parameter C, the hyperplane will have the soft margin with certain error tolerance.

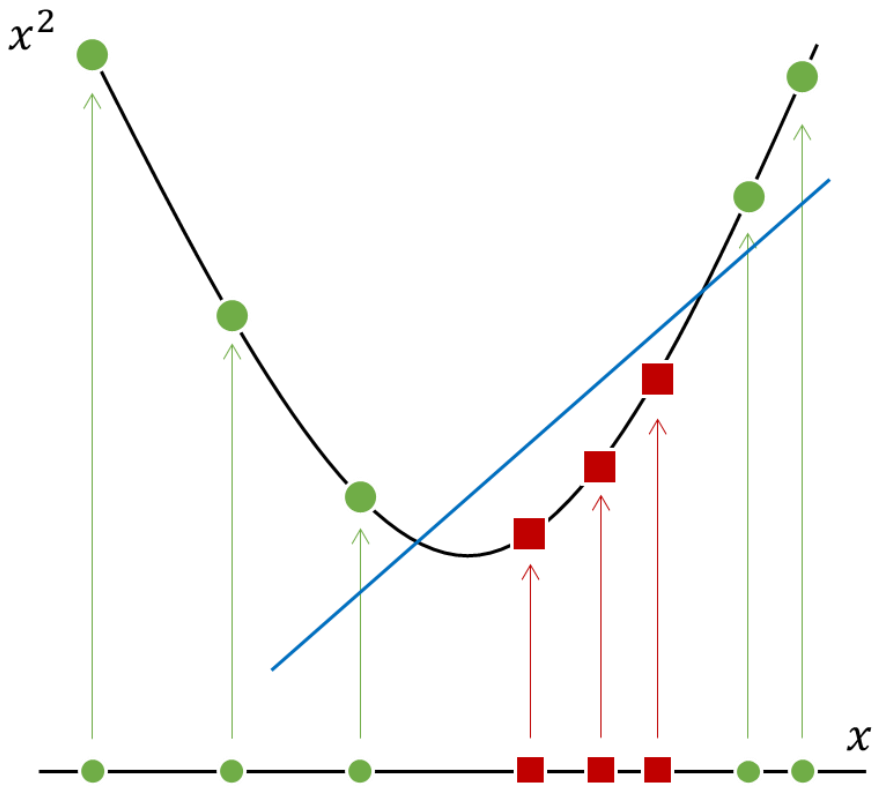

Figure 34. 1D non-separable case to 2D separable case

As shown in Figure 34, there is no point to discriminate these two classes in one-dimensional space. Like the SVM regression, mapping data x from 1D to 2D by $x \rightarrow \Phi(x) = (x, x^2)$ can solve this problem easily. Still in Figure 34, two classes are separated by a line $\left(\vec{w} \cdot \Phi(x)\right) + b = 0$, which is called hyperplane in high-dimensional space. Behind this mapping trick, there is a supporting mathematical theorem to guarantee an existing hyperplane. For any data set, there exists a mapping Φ to a higher-dimensional space such that the data is linearly separable.

Then the kernel method is introduced to solve this problem by conducting the mathematical computation in low-dimension space but gives out the result for dot product in high-dimensional space.

5.4 SVM Training for Classification

How to train SVM to generate different classifiers as needed?

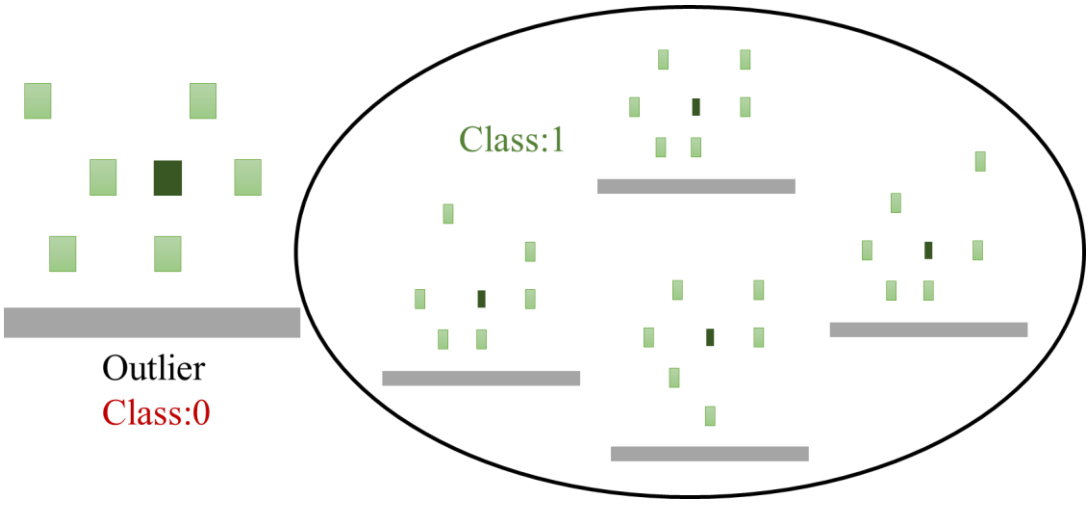

Figure 35. One-class classification illustration

After building a one-class SVM classifier for simplified geometries with same size of coordinate vectors g, it can be used to tell whether the simplified input geometry has any regression model built for it or not as illustrated in Figure 35. It is clearly seen that the simplified input geometry with no built regression model is treated as the outlier. If no, build a new initial model for it. If yes, a multi-class SVM classifier is needed to find the label of the right corresponding regression model for the simplified input geometry as illustrated in Figure 36.

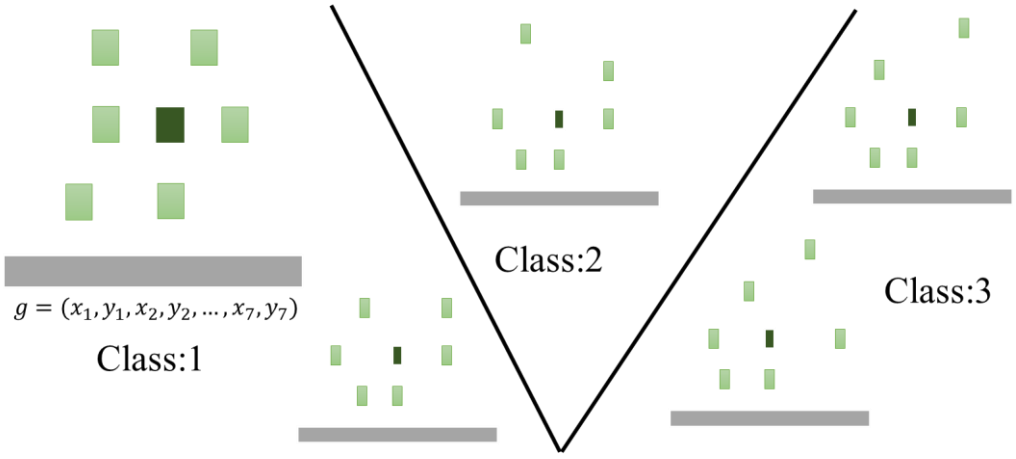

Figure 36. Multi-class classification illustration

The simplified input geometry with vector g is classified into class 1.

How to train the SVM for classification in the proposed method?

At first, every sample created for the simplified geometry with same size of the geometry vector g has been collected with its vector g and label l of the corresponding

regression model $M(f)$. Then the training data t_c for SVM is created with label l and vector g. For a sample geometry, it is written in the format as below.

$$l \quad 1{:}\,x_1\ 2{:}\,y_1\ 3{:}\,x_2\ 4{:}\,y_2\ \ldots\ 13{:}\,x_7\ 14{:}\,y_7\ 15{:}\,x_8\ 16{:}\,y_8$$

Secondly, scale the range of training data to $[-1, +1]$ to prevent the small values from being dominated by the large ones.

Then choose Gaussian kernel for the nonlinear classification. Through cross validation method, search for the best Gaussian kernel parameter vector $p_r = (C, \gamma)$ when the validation error is minimal.

Finally, train one-class SVM with the training data t_c to get classifier $OC(g)$ and train multi-class SVM with the same training data t_c to get classifier $MC(g)$.

5.5 SVM Classification Applied

Having trained SVM to get two different classifiers, it is ready for applying in choosing the corresponding regression model for the simplified input geometry.

After the simplification process, the simplified geometry has the geometry vector $g_{new} = (x_1', y_1', x_2', y_2', \ldots, x_7', y_7', x_8', y_8')$ as the input vector. Firstly check the number of conductors of the simplified geometry to find the corresponding model library. Based on the vector and any guessed label l_{guess}, create test data t_g as below.

$$l_{guess} \quad 1{:}\,x_1'\ 2{:}\,y_1'\ 3{:}\,x_2'\ 4{:}\,y_2'\ \ldots\ 13{:}\,x_7'\ 14{:}\,y_7'\ 15{:}\,x_8'\ 16{:}\,y_8'$$

Scale the range of test data to $[-1, +1]$ as it did in training process. Then use one-class SVM classifier to tell whether the simplified geometry has any regression model built for it or not by $OC(g_{new}) = 0\ or\ 1$ where 0 means no and 1 means yes. If yes, use multi-class SVM classifier to predict the label of the right corresponding

regression model for the simplified input geometry by $MC(g_{new}) = l_{predict}$ where $l_{predict}$ is the predicted label. At last the corresponding regression model is $M_{l_{predict}}(f)$.

However, as mentioned above, if the prediction accuracy for the new input geometry is not satisfied, a new regression model will be built for it and correspondingly the training data for classification needs to be updated too. It may happen that the classifier did not predict the right label of the regression model which leads to the parasitic capacitance prediction error and a newly-built model especially for the simplified input geometry. Thus there will be more and more models even for the same simplified geometry, which is making the structure of model library more complex and introducing more unnecessary boundary issues when doing SVM classification. How to solve this problem?

Regression models merging is another important process when checking accuracy. After the size of prediction database DB_{p1} reaches threshold T, merge the training data sets of all other different regression models with the same simplified geometry into one large training set. To prevent overfitting, check the merged training set to verify whether it is unbiased or not with step $\frac{1}{2} \times S_{min}$. If it is not unbiased, create samples for the missing part to make it unbiased. Then search for the best Gaussian kernel parameter vector $p_r = (C, \gamma, \varepsilon)$ with five-times cross validation and train SVM to get the merged regression model $M_m(f)$ instead of the old discrete ones. Meanwhile, update the training data t_c for classification and retrain SVM to get the merged one-class classifier OC_m and multi-class classifier MC_m. Therefore the prediction range of the

regression model is large and the number of models gets reduces. It is a process from complexity to simplicity.

A question may be raised that why not build a large enough initial regression model at the very first beginning? First of all, it will cost too much time to create enough unbiased samples to generate a good regression model unless all of the samples are provided at the beginning. Although it is an offline task, it is still an impossible mission. The learning process is a process collecting all the necessary samples and a process improving itself. But is it the end? Can it do even better?

The answer is yes. After successfully predicting and collecting enough reference data for the merged model, it is believed that the SVM regression model is steady and mature. If the model needs even better accuracy by reducing the error from dropping the shielded conductors, more victims should be added in to form a more complex regression model.

For the steady merged regression model, revisit all of the corresponding input geometries in the prediction database and exclude all victims of their simplified geometries. Then rerun the simplification algorithm to get the secondary important victims for each simplified geometry. For all the secondary simplified geometries, scale the range of geometry vectors and discriminate them into different subclasses. New sub-models are built for each different subclass following the steps above. When finding model next time, if the model is found and it has sub-models, then run simplified algorithm again to look for the sub-model for better prediction accuracy. Models merging process can also be applied until the accuracy reaches preset satisfying level.

CHAPTER VI

CONCLUSION

Up till now, Chapter I is about introduction of interconnect parasitic extraction challenges and motivation of the fast extraction method. Chapter II explains the trade-off between different existing extraction methods and brings in the machine learning technique for this proposed method. From Chapter III to V, it presents the proposed method in details step by step. To sum it up, there is an overall flow in Figure 37.

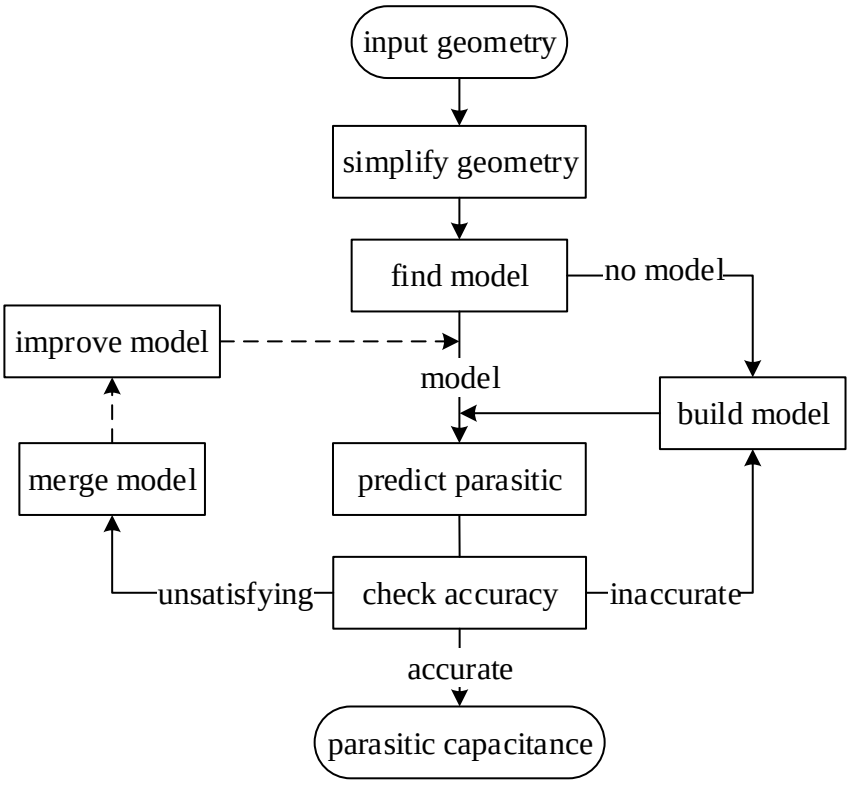

Figure 37. Overall flow

From Figure 37, the general idea of the proposed method is presented and each block has its explaining details in different chapters. The dash line here means the flow does not continue and the merged model or improved model is used for next time.

After presenting this overall flow, it is necessary to conduct experiments to show this proposed method does work. So a test three-layer interconnect geometry is shown in Figure 38 below.

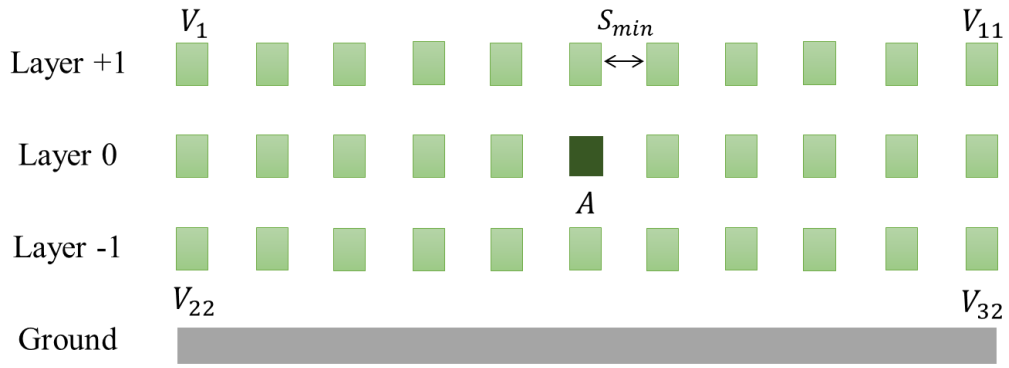

Figure 38. Test interconnect geometry

In this test geometry, victims $V_i, i \in 1 \ldots 32$ are randomly present and every pair of nearest neighbors on the same layer has minimum space between each other. Because it is a three-layer simplified geometry, 256 regression models have been built initially by LIBSVM [14] and each model has at most $4^8 = 65536$ training samples.

Next generate 200 random test geometries and predict the parasitic total capacitance r_p for each of them. Run the field solver Raphael [15] for reference value r_r and plot a histogram with errors calculated by $(r_r - r_p)/r_r$ in Figure 39.

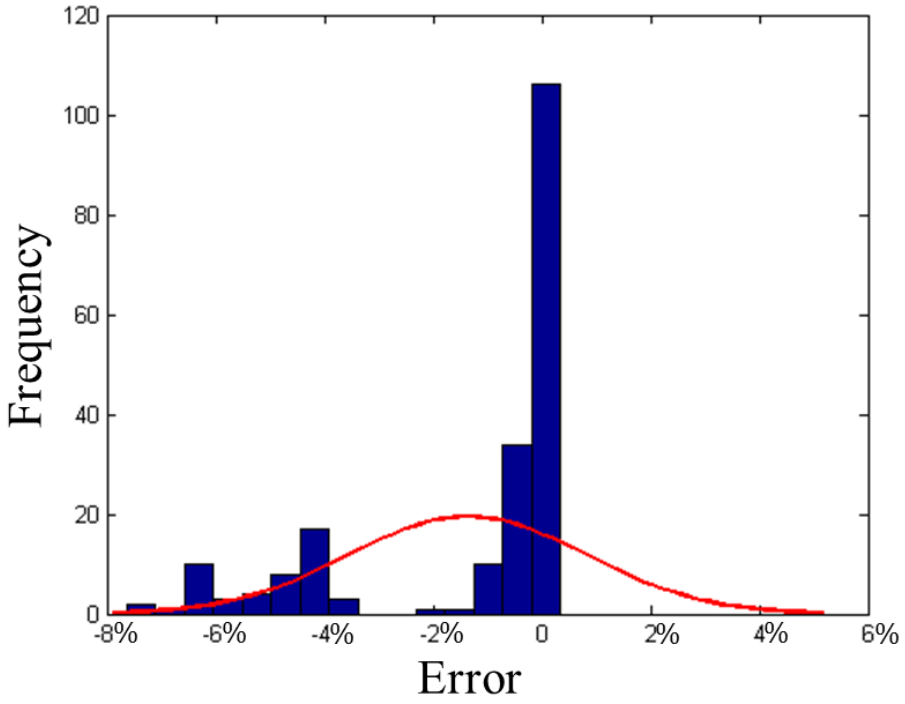

Figure 39. Histogram for initial 200 tests

From Figure 39, it can be seen that there are some large prediction errors varying from -8% to -4%. There may be many reasons for this situation. One is the error from dropping shielded conductors; one is the prediction error for the regression model itself; one maybe the error for input vector that is out of prediction range. According to this proposed method, 27 new models are generated for these inaccurate cases. Then another 200 random tests have been conducted with 283 models in the library.

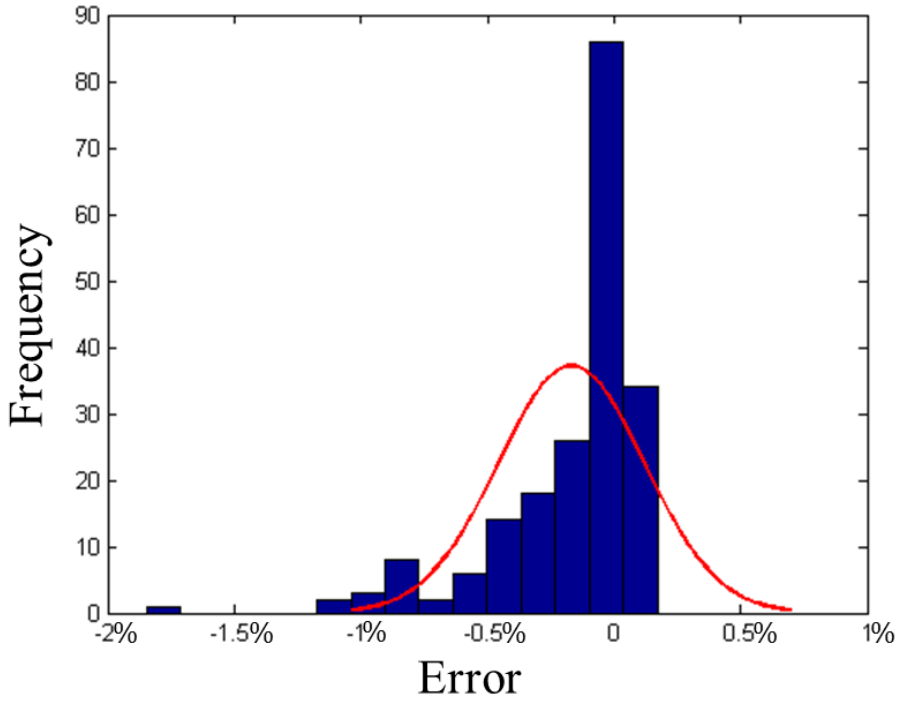

Figure 40. Histogram for another 200 tests

From Figure 40, it can be seen that the out of range prediction error has been removed and the regression model prediction error has also been reduced because of more training samples added in. For the sake of simple test geometries, the error seems small but it will be larger with a much more complex geometry. 200 also may not be a large number but it still proves that this proposed method works. It is believed that after the models merging and improving, the proposed method will perform better.

Last but not least, Table 4 is made for comparing the execution time between the proposed methods with the field solver.

Table 4. Execution time comparison table

Test #	Execution Time	
	SVM	Field Solver
200	0.013s	23m20s
Execution time ratio		$\approx 9/\text{Million}$

As seen in Table 4, the proposed SVM prediction method is significantly faster than the field solver and provides satisfactory accuracy.

REFERENCES

[1] M. Bohr, R. Borkar and S. Jourdan, "Advancing Moore's Law in 2014-The Road to 14 nm," 11 8 2014. [Online]. Available: http://www.intel.com/content/dam/www/public/us/en/documents/presentation/advancing-moores-law-in-2014-presentation.pdf. [Accessed 24 10 2014].

[2] R. Lewington, "Flowing Copper," Semimd, 31 07 2012. [Online]. Available: http://semimd.com/applied/2012/07/31/flowing-copper/. [Accessed 24 10 2014].

[3] Y. Cao, "Modeling of Interconnect Parasitics," in *Predictive Technology Model for Robust Nanoelectronic Design*, Springer, 2011, pp. 81-103.

[4] T.Sakurai and K.Tamaru, "Simple Formulas for Two- and Three-Dimensional Capacitances," *IEEE Transactions on Electron Devices,* vol. 30, no. 2, pp. 183-185, 1983.

[5] U. Choudhury and A. Sangiovanni-Vincentelli, "Automatic Generation of Analytical Models for Interconnect Capacitances," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 14, no. 4, pp. 470--480, 1995.

[6] T. M. Mitchell, The Discipline of Machine Learning, Pittsburgh: Carnegie Mellon University, School of Computer Science, Machine Learning Department., 2006.

[7] J. Cong, L. He, A. B. Kahng, D. Noice, N. Shirali and S. H.-C. Yen, "Analysis and Justification of a Simple, Practical 2 1/2-D Capacitance Extraction Methodology," in *Proceedings of the 34th annual Design Automation Conference*, 1997.

[8] H. Gavin, "The Levenberg-Marquardt Method for Nonlinear Least Squares Curve-fitting Problems," *Department of Civil and Environmental Engineering, Duke University,* 2011.

[9] J. S. Alex and S. Bernhard, "A Tutorial on Support Vector Regression," *Statistics and Computing,* vol. 14, no. 3, pp. 199-222, 2004.

[10] V. Vladimir, The Nature of Statistical Learning Theory, Springer, 2000.

[11] G. B. Orr, "Overfitting," Willamette University, 28 4 1999. [Online]. Available: http://www.willamette.edu/~gorr/classes/cs449/overfitting.html. [Accessed 24 10 2014].

[12] G. Kunapuli, "Support Vector Machines," [Online]. Available: http://pages.cs.wisc.edu/~dpage/cs760/SVMs_1.pdf. [Accessed 24 10 2014].

[13] J. S. Alex and S. Bernhard, Learning with Kernels, Berlin: GMD-Forschungsz, 1998.

[14] C. Chih-Chung and L. Chih-Jen, "LIBSVM : a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology,* vol. 2, no. 3, pp. 27:1--27:27, 2011.

[15] Synopsys, "Raphael Interconnect Analysis Program Reference Manual," Synopsys, 2013.